UMEÅ UNIVERSITY

Advancing Federated Learning: Algorithms and Use-Cases

Sourasekhar Banerjee

Doctoral Thesis, September, 2024
Department of Computing Science
Umeå University
Sweden

Department of Computing Science
Umeå University
SE-901 87 Umeå, Sweden

sourasb@cs.umu.se

ISBN 978-91-8070-463-2 (Print)
978-91-8070-464-9 (digital)
ISSN 0348-0542
UMINF 24.09

Cover illustrated by Ida Åberg.

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

Printed by Cityprint i Norr AB, Umeå, 2024

“Arise, awake, and stop not till the goal is reached."- Swami Vivekananda

Abstract

Federated Learning (FL) is a distributed machine learning paradigm that enables the training of models across numerous clients or organizations without requiring the transfer of local data. This method addresses concerns about data privacy and ownership by keeping raw data on the client itself and only sharing model updates with a central server. Despite its benefits, federated learning faces unique challenges, such as data heterogeneity, computation and communication overheads, and the need for personalized models. Thereby results in reduced model performance, lower efficiency, and longer training times.

This thesis investigates these issues from theoretical, empirical, and practical application perspectives with four-fold contributions, such as federated feature selection, adaptive client selection, model personalization, and socio-cognitive applications. *Firstly*, we addressed the data heterogeneity problems for federated feature selection in horizontal FL by developing algorithms based on mutual information and multi-objective optimization. *Secondly*, we tackled system heterogeneity issues that involved variations in computation, storage, and communication capabilities among clients. We proposed a solution that ranks clients with multi-objective optimization for efficient, fair, and adaptive participation in model training. *Thirdly*, we addressed the issue of client drift caused by data heterogeneity in hierarchical federated learning with a personalized federated learning approach. *Lastly*, we focused on two key applications that benefit from the FL framework but suffer from data heterogeneity issues. The first application attempts to predict the level of autobiographic memory recall of events associated with the lifelog image by developing clustered personalized FL algorithms, which help in selecting effective lifelog image cues for cognitive interventions for the clients. The second application is the development of a personal image privacy advisor for each client. Along with data heterogeneity, the privacy advisor faces data scarcity issues. We developed a daisy chain-enabled clustered personalized FL algorithm, which predicts whether an image should be shared, kept private, or recommended for sharing by a third party.

Our findings reveal that the proposed methods significantly outperformed the current state-of-the-art FL algorithms. Our methods deliver superior performance, earlier convergence, and training efficiency.

Sammanfattning

Federerat lärande (FL) är en distribuerad maskininlärningsparadigm som möjliggör träning av modeller ute hos kunder eller organisationer utan att överföring av lokala data krävs. Metoden behandlar farhågor om datasekretess och ägande genom att behålla rådata på klienten och endast dela modelluppdateringar med en central server. Trots dess fördelar står federerat lärande inför unika utmaningar, såsom dataheterogenitet, beräknings- och kommunikationskostnader samt ett behov av personliga modeller. Detta resulterar i försämrad modellprestanda, lägre effektivitet och längre träningstider.

Den här avhandlingen undersöker dessa frågor ur teoretiska, empiriska och praktiska tillämpningsperspektiv genom fyra olika bidrag, såsom federerat funktionsval, adaptivt klientval, modellpersonalisering samt sociokognitiva tillämpningar. *För det första* tog vi itu med dataheterogenitetsproblemen för federerat funktionsval i horisontellt FL genom att utveckla algoritmer baserade på ömsesidig information och multiobjektivsoptimering. *För det andra* tacklade vi systemheterogenitetsproblem som involverade variationer i beräknings-, lagrings- och kommunikationsmöjligheter mellan klienter. Vi föreslog en lösning som rankar klienterna med multiobjektivsoptimering för effektivt, rättvist och adaptivt deltagande i modellträningen. *För det tredje* behandlade vi frågan om klientdrift orsakad av dataheterogenitet i hierarkiskt federerat lärande där personligt federerat lärande tillämpats. *Till sist* fokuserade vi på två nyckelapplikationer som drar nytta av FL-ramverket men som lider av problem med dataheterogenitet. Den första applikationen försöker förutsäga nivån på självbiografiskt återkallande av händelser som är associerade med livsloggsbilden genom att utveckla klustrade personliga FL-algoritmer, som hjälper till att välja ut effektiva livsloggsbilder för kognitiva ingrepp för klienterna. Den andra applikationen är utvecklingen av en personlig bildintegritetsrådgivare för varje klient. Tillsammans med dataheterogenitet, står integritetsrådgivaren inför problem med databrist. Vi utvecklade en sammanlänksaktiverad klustrad personliga FL-algoritm, som förutsäger om en bild ska delas, hållas privat eller rekommenderas för delning av en tredje part.

Våra resultat visar att de föreslagna metoderna avsevärt överträffar de nuvarande toppmoderna FL-algoritmerna. Våra metoder ger överlägsen prestanda, tidigare konvergens och träningseffektivitet.

Preface

This thesis contains the following papers.

Paper I **Sourasekhar Banerjee**, Erik Elmroth, and Monowar Bhuyan. Fed-FiS: A Novel Information-Theoretic Federated Feature Selection for Learning Stability. *Proceedings of the* 28^{th} *International Conference on Neural Information Processing (ICONIP)*, Springer Cham., Communications in Computer and Information Science, Vol. 1516, pp. 480-487, 2021.

Paper II **Sourasekhar Banerjee**, Xuan-Son Vu, and Monowar Bhuyan. Optimized and Adaptive Federated Learning for Straggler-Resilient Device Selection. *Proceedings of the International Joint Conference on Neural Networks (IJCNN)*, IEEE, pp. 1-9, 2022.

Paper III **Sourasekhar Banerjee**, Devvjiit Bhuyan, Erik Elmroth, and Monowar Bhuyan. Cost-Efficient Feature Selection for Horizontal Federated Learning. *IEEE Transactions on Artificial Intelligence (TAI)*, IEEE, doi: `10.1109/TAI.2024.3436664`, 2024.

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

Paper V **Sourasekhar Banerjee**, Debaditya Roy, Vigneshwaran Subbaraju, and Monowar Bhuyan. Predicting Event Memorability using Personalized Federated Learning. *Submitted for publication*, 2024.

Paper VI **Sourasekhar Banerjee**, Vengateswaran Subramaniam, Debaditya Roy, Vigneshwaran Subbaraju, and Monowar Bhuyan. The Case for Federated Learning in Developing Personalized Image Privacy

Advisor. *Submitted for publication*, 2024.

In addition, the following works were carried out during PhD studies, but these are not part of this thesis.

Paper I **Sourasekhar Banerjee**, Yashwant Singh Patel, Pushkar Kumar, and Monowar Bhuyan. Towards Post-disaster Damage Assessment using Deep Transfer Learning and GAN-based Data Augmentation. *Proceedings of the* 24^{th} *International Conference on Distributed Computing and Networking (ICDCN)*,ACM, pp. 372-377, 2023.

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

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

Acknowledgements

The completion of this Ph.D. thesis has been a significant and challenging journey, achievable only through the support and guidance of many individuals. I am deeply honored to have had their support and would like to express my sincere appreciation to them.

First and foremost, I extend my heartfelt appreciation to my supervisor, **Dr. Monowar Bhuyan**. His steadfast support from the very beginning to the end of this journey has been indescribable. His constant presence and assistance have been invaluable in helping me overcome numerous obstacles. I am also profoundly grateful to his family for their support and understanding.

Secondly, I would like to express my profound gratitude to my co-supervisor, **Prof. Erik Elmroth**. His expertise, guidance, and continued encouragement have been truly invaluable throughout my research. His mentorship has not only enhanced my research skills but also inspired me to strive for excellence. I am deeply appreciative of all the time and effort he has invested in my development.

I also want to express my heartfelt gratitude for the financial support from the **Wallenberg AI, Autonomous Systems, and Software Program (WASP)**, which made this research possible. Additionally, I must mention the access to **Berzelius GPUs**, which played a crucial role in handling the computational demands of my work. The resources and backing from WASP have been a cornerstone of my ability to pursue and accomplish my research goals, and I am sincerely thankful for their support.

During my enriching four-year journey, I had the opportunity to spend two months as a research intern at the **A*STAR Institute of High Performance Computing in Singapore, within the Social and Cognitive Computing department**. During this period, I had the privilege of working closely with two esteemed senior scientists, **Dr. Vigneshwaran Subbaraju** and **Dr. Debaditya Roy**. Their mentorship was incredibly valuable, and our daily 4 pm discussions were particularly enlightening, deepening my understanding of interdisciplinary research. Additionally, I collaborated with **Vengateswaran Subhramaniam**, a skilled research engineer in the department, from whom I learned numerous engineering skills. This experience was essential in broadening my research and technical skills.

I would like to acknowledge my mentor, **Dr. Alp Yurtsever**, from the

Department of Mathematics and Mathematical Statistics at Umeå University, Sweden. He introduced me to the field of machine learning optimization and provided invaluable expertise that greatly enhanced my understanding of complex problems and their solutions. His guidance played a crucial role in helping me grasp Federated Learning from an optimization perspective. In addition, I extend my gratitude to **Ali Dadras**. Our collaboration was highly insightful, and the discussions we had were profoundly beneficial in enhancing my understanding of the subject.

I also wish to extend my gratitude to **Dr. Xuan-Son Vu**, a staff scientist at the Department of Computing Science at Umeå University, Sweden. His guidance and support have significantly advanced my research and contributed to my academic growth. I would also like to acknowledge the valuable experience of working with two interns, **Devvjiit Bhuyan** and **Himika Das**. Collaborating with them greatly expanded my knowledge of the subject.

Prof. Nabendu Chaki has been a pivotal figure in my life and career. His influence has profoundly shaped who I am today. As my first research mentor, he introduced me to the world of academic research and ignited a passion for it within me. Under his guidance, I developed the confidence and ambition to pursue my Ph.D. studies abroad. His unwavering support and encouragement were crucial in this decision, and I will always be grateful for his belief in my potential. Professor Chaki's mentorship has been invaluable, providing me not only with knowledge but also with inspiration. I want to extend my heartfelt thanks to him for his remarkable influence and for helping shape the person I have become today. Thank you, Professor Chaki, for everything.

I am forever grateful to my family, who are far away, waiting every day for my phone call. I dedicate this thesis to my parents, **Mr. Chandra Sekhar Banerjee** and **Mrs. Anuradha Banerjee**. Words cannot express the depth of my appreciation for everything you have done for me. I wish you both a long and healthy life, and I am committed to making you proud every moment. I am also indebted to my elder sister, **Dr. Somdutta Banerjee**, and my brother-in-law, **Dr. Montu Bose**. Without your support, I would not be who I am today. Your constant encouragement and well wishes have been invaluable, and I will always cherish your support. I also have lots of love for my niece, **Aaditri**, whose smiling face has provided me with daily motivation and joy. I would like to mention my maternal uncle, **Prof. Ambar Ghosal**. During my childhood, I aspired to be a theoretical physicist like you, but I ultimately became a computer scientist. I still remember the miniature Mercedes car you brought me from Italy, which I cherished dearly. It sparked a childhood dream that becoming a researcher could allow me to have many such toys. Jokes aside, you have been a significant motivation for me to pursue a career in research.

Klintvägen 7 is probably the best place in Umeå where I spent three important years of my life there, living with my neighbors **Jonas Öberg, Saloni Kwatra, and Yashwant Patel**. We lived like a family, sharing meals, watching movies, partying, and playing badminton together. Throughout my Ph.D. journey, we supported each other and maintained a healthy lifestyle,

which greatly contributed to my overall well-being.

I am deeply grateful to my colleagues in **the Autonomous Distributed Systems (ADS) lab**, including **Lidia, Charles, Pim, Nayereh, Aleksandra, Oliver, Rohail, Yangyang, Zhou, ILeet, Dr. Hajar Siar, Dr. Anil Singh, Dr. Clément Courageux-Sudan, Dr. Paul Townend, Dr. Chanh Le Tan Nguyen, and Dr. Per-Olov Östberg**. Their collaborative discussions have significantly enriched my research experience.

I would like to express my heartfelt gratitude to **Dr. Eunil Seo** for the enlightening discussions. I am also thankful to **Anindya**, a dear friend with whom I shared numerous meaningful personal and professional conversations. Additionally, I wish to acknowledge **Obaidullah** and **Dr. Kshira Sagar Sahoo**, whose thoughtful discussions greatly contributed to my understanding of the subject. My sincere thanks also go to **Dr. Javad Forough** and **Dr. Ali Rahmanian** for the many insightful and friendly conversations we had.

I would like to acknowledge the technical assistance team led by **Tomas Forsman** for their constant support with any technical issues. Special thanks to **Andreas Sandström** and **Mattias Åsander**.

Throughout this long journey, I have met many friends, some of whom have become very special to me. Among them are **Dr. Sudeb Majee, Dr. Shekhar Negi, Divya, Santhan, Ayush, and Karlo**.

I want to express my gratitude to all the anonymous reviewers for their valuable comments on my submitted papers. Their feedback has significantly improved the quality of our work. Lastly, I want to thank myself for believing in me, working hard, and staying motivated towards my goal. The journey of a Ph.D. student is much like stochastic gradient descent, involving small, iterative steps, learning from mistakes, and moving closer to the goal despite uncertainties.

Thanks and regards,

Sourasekhar Banerjee
September, 2024

Contents

Chapter 1

Introduction

The evolution of federated learning has significantly reshaped the landscape of distributed machine learning. By enabling the training of models across decentralized data sources without the need to share raw data, federated learning enhances data privacy and security. Moreover, it reduces the need for extensive data transfer, which can improve efficiency and reduce latency in model training. This chapter commences by outlining the motivation behind the thesis, followed by the research objectives and the research methodology. Then, the last part outlines the research contributions and concludes with a summary of the overall thesis structure.

1.1 Research Motivation

Over the decades, we witnessed the rapid growth in the Machine Learning (ML) paradigm, which has propelled the advancement of numerous Artificial Intelligence (AI) applications. Particularly, Deep Learning (DL) has thrived due to the availability of vast amounts of data around different applications. This wealth of data is crucial as these models, like GPT-3 [Bro+20], GPT-4 [Ach+23], LLaMA [Tou+23], Gemini [Tea+23], typically require significant volumes of data to perform effectively; for instance, GPT-3 was trained with approximately 570GB of data sourced from the Internet and licensed datasets[1]. However, as these models increasingly rely on large datasets, concerns about the source and ownership of this data are also important. Society is increasingly aware of data ownership issues, focusing on who has the right to access and use the data for development of AI methods. Typically, the process involves aggregating data from various sources into a central system with the computing power to train learning models. However, this has raised public concerns about the potential misuse of personal data for commercial or political gain without

[1] `https://openai.com/blog/data-partnerships`, `https://medium.com/@dlaytonj2/chatgpt-show-me-the-data-sources-11e9433d57e8`

explicit consent from the individuals. It emphasizes the delicate balance between utilizing up-to-date data for AI and protecting individual privacy and data security.

Despite new regulations such as the European Union's (EU) General Data Protection Regulation (GDPR) [PE16], the US's California Consumer Privacy Act (CCPA) [Cal18], Canada's Personal Information Protection and Electronic Documents Act (PIPEDA) [Par00] etc., aims to safeguard personal data while handling, exchanging, and aggregating data across different entities remains complex. The sensitive nature of certain data, like financial and health information, requires it to be securely stored in isolated silos controlled by the data creators or owners. Competitive tensions, privacy concerns, complex administrative procedures, and high data transfer costs complicate the integration of dispersed datasets needed for decision-making.

This complexity extends to the AI industries, where data collaboration is a major hurdle. When two organizations collaborate on ML model training, they typically share data directly or involve a third party to extract insights. This can lead to the original data owners losing control over their data, sparking concerns about data sovereignty and privacy. Additionally, when such collaborations lead to a superior model, it's challenging to distribute the benefits from the model fairly among all participants. The lack of clear guidelines on benefit sharing can cause disputes or reluctance to collaborate, underscoring the need for more defined data-sharing agreements that protect data rights and ensure the equitable benefit of distributions.

Moreover, with the rise of edge computing in the Internet of Things (IoT) ecosystem, data is no longer centralized but distributed across multiple locations and individuals. Edge devices at each site initially process and filter data before transmitting only essential information to the cloud [Kha+19]. This approach helps maintain data privacy and presents new challenges in sharing models securely and efficiently across multiple sites. As edge devices such as smartphones, laptops, tablets, and workstations grow more powerful, there is an increasing potential to leverage their computational capabilities to train models directly on these devices, enhancing privacy and efficiency in AI system development.

Intuitively, the increasing computational capabilities and storage capacity of edge devices[2] Within the distributed networks, it is now possible to use their power to perform on-device model training. But, on-device training is limited to the data available on that specific device, which may not represent broader patterns. This has led to a growing interest in Federated Learning (FL) [McM+17a]. FL works in a collaborative learning framework that learns by sharing the trained local model with a centralized server. In the FL setup, the server combines all local models to create a global model. Learning in these environments differs significantly from classical distributed settings, re-

[2]In this thesis, we used the terms edge device, participant, organization, institution, device, user, lifelogger, annotator, and client interchangeably. These terms indicate the same entity.

quiring fundamental advancements in privacy, scalable machine learning, and distributed optimization. It also raises new challenges that intersect various fields, including distributed machine learning and systems [Rat+19]. Federated learning suffers from four core challenges: communication overhead, system heterogeneity, statistical or data heterogeneity, and privacy concerns. These issues set apart the federated setting from distributed learning in datacenter or traditional private data analysis.

These four challenges pose several open research questions in federated learning, including *Q1: How can federated learning be made communication efficient? Q2: How can we reduce the impact of data heterogeneity to improve model convergence and performance?* Additionally, *Q3: How can we manage low-performing devices to mitigate the effects of system heterogeneity on federated learning model training?* These open research questions have motivated us to develop more efficient federated learning algorithms for both general and targeted use cases.

1.2 Research Objectives

This thesis aims to develop efficient algorithms for federated learning to mitigate the impact of challenges such as data heterogeneity and system heterogeneity during model training. The Research Objectives (RO) cover a wide range, exploring the topic from both theoretical and practical perspectives. The high-level research objectives are outlined as follows.

RO1 : To address the challenges of data heterogeneity in federated feature selection and develop algorithms for feature selection in federated settings.

RO2 : To address the challenges of system heterogeneity and develop algorithms for optimal client selection in federated learning.

RO3 : To address the data heterogeneity in hierarchical federated learning and develop personalized solutions.

RO4 : To identify applications where federated learning is beneficial and develop FL-enabled solutions for them.

1.3 Research Methodology

The methodology employed in this thesis follows the principles of Design Science Research (DSR) [VHM20], as depicted in Figure 1.1. The process begins with identifying and scientifically formulating the research problem. Next, an extensive literature review is conducted to gain a thorough understanding of the field and existing solutions to the identified problem. Then, an algorithm for the problem is designed, followed by its implementation. The performance of the proposed algorithm is evaluated using appropriate metrics and compared

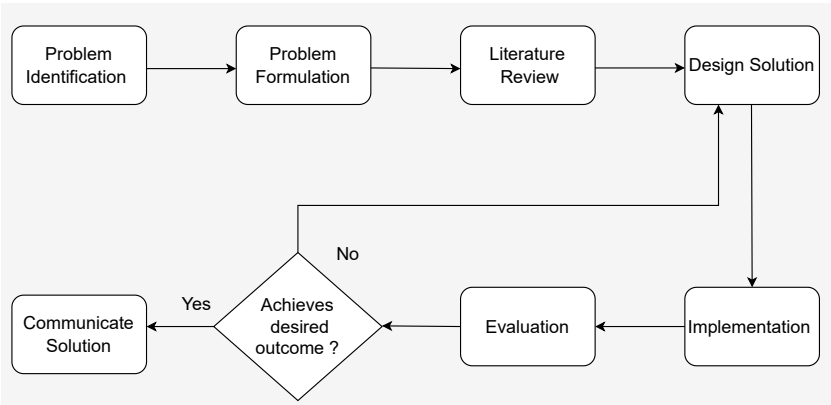


Figure 1.1: Research methodology

with state-of-the-art solutions across multiple datasets. If the proposed solution does not perform well or issues are identified, the design phase is reviewed, and improvements are made. Once the proposed solution achieves the desired outcomes, the findings, proposed solution, implementation, and results and analysis of the results are summarized in a scientific article. The article is then submitted to a peer-reviewed conference or journal for publication.

Our systematic investigation process includes: (1) Fundamental research, which aims to enhance our understanding of basic principles and theories, and (2) Applied research, which targets specific and practical problems. This later approach focuses on utilizing existing knowledge and theories to create solutions or new technologies that address real-world challenges in diverse application domains.

Throughout our studies, we utilized three different research methods: (1) Exploratory research that structures and identifies new problems (Paper I and III). (2) Constructive research to develop novel solutions to a specific persisting problem (Paper I - VI). (3) Empirical research or simulations to test the feasibility of a solution using empirical evidence (Paper I-VI).

1.4 Research Contributions

This study focuses on developing and implementing various Federated Learning algorithms to address the research objectives outlined in Section 1.2. An illustration of how the six papers contribute to **ROs** is provided in Figure 1.2.

Papers I and ***III*** focus on achieving the objective outlined in **RO1** by addressing the challenges associated with selecting features from datasets distributed in a federated manner while dealing with data heterogeneity. We introduced two approaches, Fed-FiS in ***Paper I*** and Fed-MOFS in ***Paper III***, to

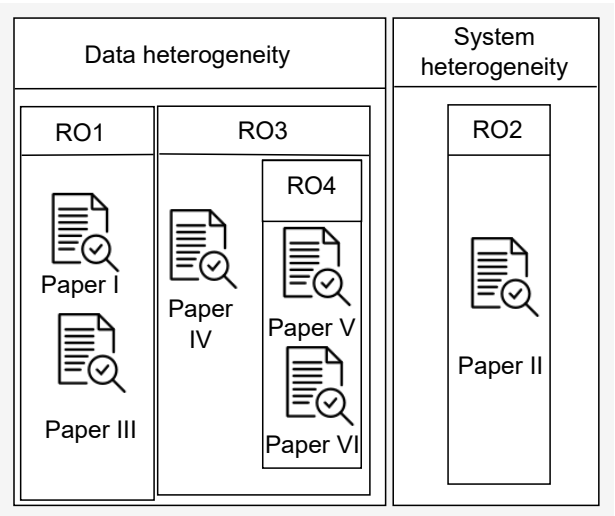


Figure 1.2: Illustration of the relationship between research objectives and contributory papers.

tackle this issue. Both Fed-FiS and Fed-MOFS employ the same local feature selection strategy, utilizing mutual information and clustering on individual clients to select local features. The key difference lies in their global feature ranking and selection methods. Fed-FiS utilizes a score function based on relevance and redundancy, while Fed-MOFS employs multi-objective optimization, simultaneously maximizing relevance and minimizing redundancy. Both methods efficiently select features from clients, enhance overall learning performance in the presence of data heterogeneity, improve stability, and are scalable across multiple clients. In ***Paper II***, we address the objective set out in **RO2** by introducing a method for ranking clients based on their system heterogeneity. Clients are then selected adaptively in FL training based on this ranking. The ranking process involves solving a multiobjective optimization problem, where we simultaneously maximize the available processing capacity, available memory, and available bandwidth. Once the ranking is complete, clients are selected adaptively for each global iteration based on their rank (Best-performing client to worst-performing client). The proposed algorithm is designed to efficiently identify low-performing or straggler devices in FL, leading to improved performance, convergence, fairness, and training time compared to random client selection. ***Paper IV*** addresses the objective given in **RO3**. We studied the problem of client drift that occurs due to statistical heterogeneity in multi-tier or hierarchical federated learning. we proposed a personalized multi-tier fed-

erated learning algorithm that simultaneously learns personalized models for devices, teams, and a global model. ***Papers V*** and ***VI*** cover both objectives **RO3** and **RO4** and focused problems from the application standpoints. In ***paper V***, we focused on predicting event memorability from lifelog images using federated learning. Visual lifelogging is an act of automatically and periodically capturing images using body-worn cameras, smart glasses, etc. The event memorability data is highly subjective to the clients and depends on several factors such as gender, age, etc. Event memorability in FL suffers from data heterogeneity. To solve this, we proposed clustered personalized federated learning solutions. These clusters can be formed dynamically based on model similarity with other lifeloggers, the distance between the global and local models, and the distance between the cluster and the local model. Alternatively, clusters can be formed based on the similarity between lifeloggers' memory scores in advance. In ***Paper VI***, our focus was on creating personalized image privacy advisor for each client. With minimal annotation and client preferences, our aim was to predict the privacy score of the images. Using these privacy scores, clients can then decide whether the image is shareable or not. This approach aims to reduce privacy breaches and make it easier to share images with friends or on social media. The privacy advisor model faces challenges due to data heterogeneity and data scarcity. To address this, we proposed daisy-chain-enabled clustered personalized federated learning algorithms. Similar to the previous application, the clusters can be formed dynamically based on model similarity between users, global model, and cluster model, or made apriori based on the personality scores of the participating clients.

1.5 Thesis Organization

The remainder of the thesis is organized as follows. In Chapter 2, we explore the fundamental concepts of federated learning, including its challenges, the baseline Federated Averaging (FedAvg) algorithm, categorizations of FL, data distribution strategies, evaluation metrics, and the datasets used to evaluate the proposed and state-of-the-art FL models in the associated papers. Chapter 3 is devoted to federated feature selection, covering its fundamentals, categorization, and providing a comprehensive overview of various federated feature selection approaches. Chapter 4 addresses system heterogeneity in FL, discussing its properties, client selection strategies, and straggler mitigation, along with a comparative study of different client selection methods. In Chapter 5, we examine the challenges of statistical heterogeneity, discuss the motivation behind personalization in federated learning, categorize various personalized FL strategies, and compare different personalized FL methods. Chapter 6 explores emerging application areas of federated learning. Chapter 7 provides a summary of the scientific contributions presented in the six associated papers (Papers I to VI). Finally, in Chapter 8, we discuss the potential future research directions.

Chapter 2

Foundation of Federated Learning

In this chapter, we explore the fundamentals of federated learning. We begin by introducing the federated learning problem and its associated challenges (Section 2.1). Next, we examine the various categories of federated learning frameworks and their topological distinctions (Section 2.2). The chapter also covers the data distribution strategies for both Independent and Identically Distributed (IID) and non-IID data (Section 2.3). Following this, we discuss the evaluation metrics (Section 2.4) employed to measure the performance of both the proposed and state-of-the-art algorithms. Finally, we provide a summary of the datasets (Section 2.5) used in the experimental studies.

2.1 Federated Learning and Its Challenges

In traditional centralized learning, all the data collected on local devices such as mobile phones, cameras, and watches needs to be moved and stored in a central cloud datacenter. This requirement raises concerns about privacy risks and data leakage and places a high demand on the storage and computing capacity of servers when dealing with large amounts of data. Additionally, transferring large amounts of data from a local device to a server is expensive and increases communication overhead.

To address these challenges, Federated Learning (FL) provides a promising solution without data displacement. FL is a collaborative and distributed machine learning framework that encourages multiple clients to participate and perform machine learning tasks jointly. FL ensures that raw data remains private and secure, protecting sensitive information while leveraging the combined knowledge of all clients to improve machine learning tasks. FL comprises four steps (see Figure 2.1) as follows.

1. **Client selection:** Initially, clients are chosen from the pool of available

clients before the training begins. This selection can be carried out randomly [McM+17a] or follow a specific strategy [NY19; Fu+23a; BVB22]. Selected clients initialize their local model using the global model.

2. **Client update:** The selected clients individually train their local model using private data and then send the model parameters or gradients to the server for a global update.

3. **Global update:** The server collects all model parameters or gradients from the clients and aggregates them to create a global model.

4. **Transfer of global model:** The trained global model is then distributed back to the available clients in the client pool.

After step 4, all clients update their local models with the global model, and a new set of clients is selected from the client pool for the next round of training. This process (steps 1 to 4) repeats until the global model converges.

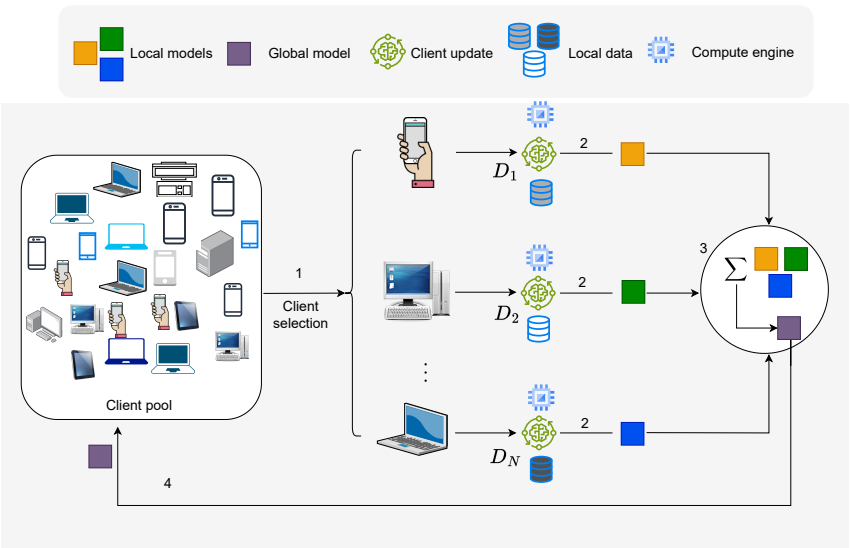


Figure 2.1: Federated learning architecture

A conventional federated learning problem, for example, FedAvg [McM+17a], focuses on training a unified, global model using data distributed across remote clients. The objective is usually to minimize the following objective function.

$$\min_{w \in \mathbb{R}} F(w), \ \ where \ F(w) = \sum_{i=1}^{N} p_i F_i(w) \tag{2.1}$$

Where N is the total number of available clients, $p_i \geq 0$, and $\sum_{i=1}^{N} p_i = 1$. The regularized term p_i specifies the relative impact of each client with two natural settings, $p_i = \frac{1}{N}$ or $p_i = \frac{D_i}{D}$, where $D = \sum_{i=1}^{N} D_i$. D_i represents the number of training samples available at the i^{th} client, while D represents the total samples available across all participating clients. w is the parameters of the models which are real numbers ($w \in \mathbb{R}$).

$F_i(w)$ is the local objective function for the i^{th} client. The local objective function is often defined as the empirical risk over local data, such as,

$$F_i(w) = \frac{1}{D_i} \sum_{j=1}^{D_i} f_i(w, x_j, y_j)$$

where, (x_j, y_j) represents the data point that belongs to D_i. Each client in FedAvg (see Algorithm 1) updates their local model by performing gradient descent (see Equation (2.2)) for a certain number of local iterations on their local data (D_i). α is the learning rate and b is the current batch of inputs.

$$w \leftarrow w - \alpha \nabla F_i(w, b) \tag{2.2}$$

Algorithm 1 FederatedAveraging (FedAvg)

Input: Total available client M, select N clients indexed by i; Total local minibatch (B) ; Local iterations (E); Global iterations (T); Learning rate (α).
Output: w_T ▷ The trained global model for which $w_T \approx w*$
Server executes:
Initialize the global model w_0
for each global iteration $t = 1, 2, \ldots, T$ **do**
  $N \leftarrow Select\_Client(M)$ ▷ Randomly select N clients from a pool of M clients
  send w_t to the N selected clients.
  for each client $N_i \in N$ **in parallel do**
    $\theta_i^{t+1} \leftarrow$ **ClientUpdate**(w_t) ▷ Updated local model
  end for
  $w_{t+1} \leftarrow \sum_1^N p_i \theta_i^{t+1}$ ▷ Aggregate all local models to have global model
end for
Client executes:
function ClientUpdate(w_t)
  $\theta_i^{0,t} \leftarrow w_t$ ▷ client i initializing local model with the global model w at the global iteration t
  $B \leftarrow$ (split D_i into batches of size B) ▷ Data is divided into B batches
  for each local iterations e from 1 to E **do**
    for batch $b \in B$ **do**
      $\theta_i^{e,t} \leftarrow \theta_i^{e,t} - \alpha \nabla F_i(\theta_i^{e,t}; b)$ ▷ Perform gradient descent
    end for
  end for
  return $\theta_i^{E,t}$ ▷ Return trained local model
end function

At first glance, FL and classical distributed learning both minimize global loss by dividing data into clients. However, some fundamental challenges are associated with solving the objective (see Equation (2.1)) in FL that distinguishes between FL and distributed learning.

2.1.1 Challenges

Federated learning has four fundamental challenges, as follows.

Statistical or data heterogeneity

In an FL system, statistical or data heterogeneity occurs due to the differing data distributions among the participating clients. The data across clients is distributed in a non-IID manner. Devices that generate or collect data can result in data samples on one device being significantly different from those on another in terms of feature and label distributions. This heterogeneity may cause the global model to perform differently between clients and lead to client drift [Kar+20]. We discuss more on this problem in *Chapter 5*. The non-IID paradigm in FL violates the IID assumptions in distributed machine learning [Zhu+21]. Furthermore, the skewness in the label distributions between devices may vary significantly [Zha+22b] and affect the learning model. In addition, skewness can also occur in features [Mou+23]. Different devices may see different features, leading to models biased towards the features they visit most frequently [Mou+23]. Some devices may have more samples of specific labels than others, which can skew the learning process towards overfitting to more common labels at the expense of rare ones [Zha+22b]. We further discuss the data distribution in *Section 2.3*.

System heterogeneity

Participating clients may exhibit heterogeneity in their devices, including variations in hardware (such as CPU and memory), network connectivity (like 3G, 4G, 5G, or Wi-Fi), and power (such as battery levels). These differences can make it difficult for some clients to participate in the FL process. They may cause some clients to drop out during training due to connectivity or energy constraints. These challenges make it harder to ensure the learning process is efficient and effective [Bon+19].To effectively address these challenges, federated learning methods must be designed to anticipate the infrequent availability of clients, be capable of accommodating various types of hardware, and be resilient enough to manage dropped clients within the network.

Communication overhead

Communication is a significant bottleneck in federated learning settings where clients interact via communication networks. In a cross-device setup, the network comprises many clients, such as millions of smartphones participating in learning. These clients are heterogeneous in computation and storage, and the communication bandwidth is also different. Furthermore, training computation-intensive deep neural network models at the edge makes federated learning more communication and computation expensive. Transferring those

heavy models in every global round consumes more data, and the heterogeneous communication channel causes significant delays in learning. Therefore, to mitigate this challenge, federated learning algorithms must be both computation and communication efficient. Additionally, the size of the model update that clients share should be small enough.

Privacy

Federated learning is called privacy-preserved collaborative machine learning because it does not involve sharing raw data with the other collaborator but shares model updates. However, communicating model updates throughout the training process can still leak sensitive information, either to a third party or to the central server [McM+17b]. Ensuring the privacy and security of the data and the learning process against various types of attacks such as data reconstruction attacks [Liu+23], model inversion attacks [Hua+21], membership-inference attacks [Zha+20], attribute inference attacks [Are+24], and model poisoning attacks [Fan+20] are crucial. While recent methods aim to enhance the privacy of federated learning using secure multiparty computation [Zha+22a; Kan+22] or differential privacy [Wu+22; CCK21], these approaches often provide privacy at the cost of reduced model performance or system efficiency.

The primary focus of this *thesis* is to address the challenges of statistical and system heterogeneity to enhance the performance of both the global and local models.

2.2 Categorization

Federated learning is categorized based on data availability, client participation, and decentralization schema.

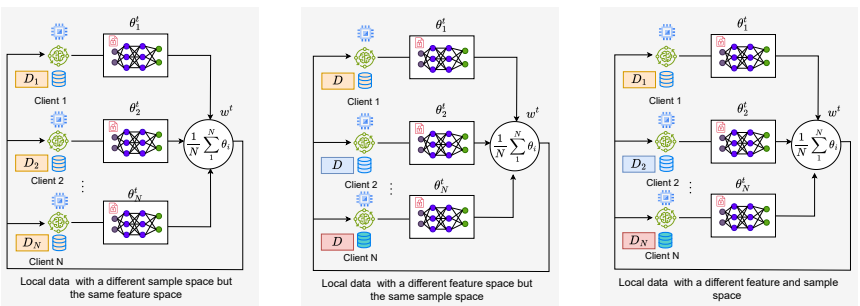
2.2.1 Categorization based on the data availability

Federated learning is classified into three categories [Yan+19] based on data availability among clients (see Figure 2.2). These are (a) Horizontal Federated Learning (HFL), (b) Vertical Federated Learning (VFL), and (c) Federated Transfer Learning (FTL).

Horizontal federated learning (HFL): Here, each client shares the same features but has different samples, meaning that clients have a common feature space but not a common sample space. As illustrated in Figure 2.2a, each client possesses distinct data samples. For example, client 1 has data sample D_1, and client 2 has data sample D_2, yet they share the same features represented by the same colored boxes in Figure 2.2a.

Vertical federated learning (VFL): Here, each client poses similar samples but different features; in other words, clients share overlapping sample spaces but do not have overlapping feature spaces. For example, in Figure 2.2b, the data sample D is distributed among N clients. However, these clients differ in feature space represented by the different colored boxes.

Federated transfer learning (FTL): In this scenario, clients don't have overlapping data or sample space. For example, in Figure 2.2c, clients have distinct data samples, such as D_1 for client 1 and D_2 for client 2, and different features are represented by different colored boxes.



(a) Horizontal FL (b) Vertical FL (c) FTL

Figure 2.2: Categories of federated learning based on data availability

2.2.2 Categorization based on client-participation

FL is classified into cross-silo FL and cross-device FL based on clients' participation [Kai+21].

Cross-silo FL: Cross-silo federated learning usually involves fewer clients, but each one has higher-quality data. In this type of federated learning, all clients are available for the entire duration of the training. System heterogeneity is not a problem in this setup because the client's device is powerful enough to compute the local model on time. Clients are connected to a reliable communication network that ensures minimal delay in data aggregation. However, cross-silo FL is challenged by data heterogeneity.

Cross-device FL: Cross-device federated learning allows machine learning to be performed across a large number of clients, including smartphones, tablets, and IoT gadgets. In cross-device FL, only a small number of clients participate in each global round, with some clients not participating in more than one global iteration. Cross-device FL encounters challenges due to variations in

data and system setups across clients, making it more complex than cross-silo FL.

2.2.3 Categorization based on decentralization schema

Federated learning architecture can be influenced by the level of decentralization among clients [Bel+23]. There are three different approaches (see Figure 2.3): Decentralized FL (DFL), Semi-Decentralized FL(SDFL), and Centralized FL (CFL).

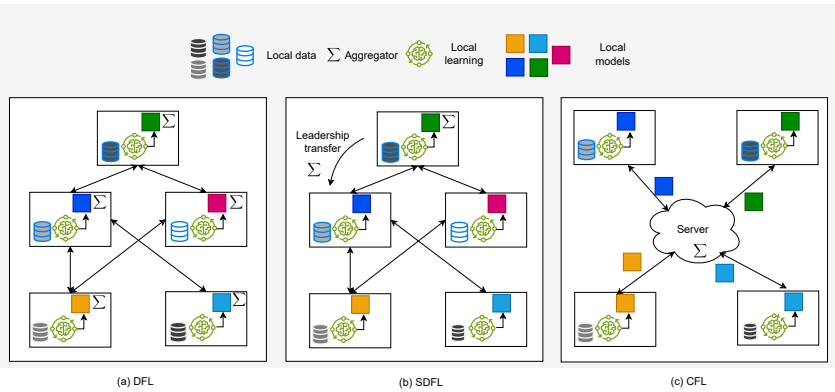


Figure 2.3: Decentralization architecture

Decentralized FL: In DFL architectures (see Figure 2.3a),the network operates autonomously. Each client trains local models on their individual datasets and communicates directly with other clients to transfer models for aggregation. In this architecture, each client functions as a local learner and aggregator, providing relaxation from synchronous model updates.

Semi-decentralized FL: Unlike the previous approach, SDFL architecture (see Figure 2.3b) assign an aggregator role to a client. This role rotates among the clients in the network and changes periodically. The next leader selection is either random or based on performance metrics such as network capacity, the computational power of the client, or energy efficiency [Yem+22; Lin+21]. Leader rotation in SDFL improves the system's performance, efficiency, robustness, and fault tolerance.

Centralized FL: In CFL-based architecture (see Figure 2.3c), data decentralization is managed by a single aggregator [McM+17a]. The server in the cloud collects models from clients and performs aggregation.

In this thesis, we considered all of our problems to be in a horizontal FL, where clients have the same feature space but a different sample space. We considered both cross-silo and cross-device setups, and we exclusively focused on CFL[1] architecture.

2.3 Data Division

The data division in clients follows Independent and Identical distribution (IID) or non-Independent and Identical (Non-IID). These data distribution strategies are described as follows,

2.3.1 Independent and Identically Distributed (IID)

In an IID setting, the data on each client is independently sampled from the same underlying distribution, which remains consistent across all clients. As illustrated in Figure 2.5a, all five clients have data from all ten classes, meaning each client represents the entire distribution of the full dataset. Consequently, in an IID scenario, the global model can be trained more efficiently, as each client's data provides a similar perspective on the overall data distribution. This allows the model to generalize well to new, unseen data.

2.3.2 Non-Independent and Identically Distributed (Non-IID)

In a non-IID setting, the data across clients are neither independent nor sampled from the same distribution. This situation arises when clients have different characteristics and diverse data sources. As shown in Figure 2.5b, each client has information about at most two different classes, while the actual dataset comprises 10 classes. Therefore, each client only has partial information about the whole dataset. Non-IID data introduces data heterogeneity in federated learning, making it challenging to train a global model.

Non-IID data generation strategies

In Figure 2.4, three distinct strategies are outlined for achieving non-IID data in federated learning: (1) Label-based division, (2) Feature-based division, and (3) Random sampling with bias. Detailed descriptions of these strategies are provided below.

1. Label-based division: This method ensures that different clients receive data with varying distributions of labels. Label-based division can be achieved through class partitioning and class imbalance. In class partitioning, each client

[1]Throughout the thesis, Centralized Federated Learning is referred to as Federated Learning (FL)

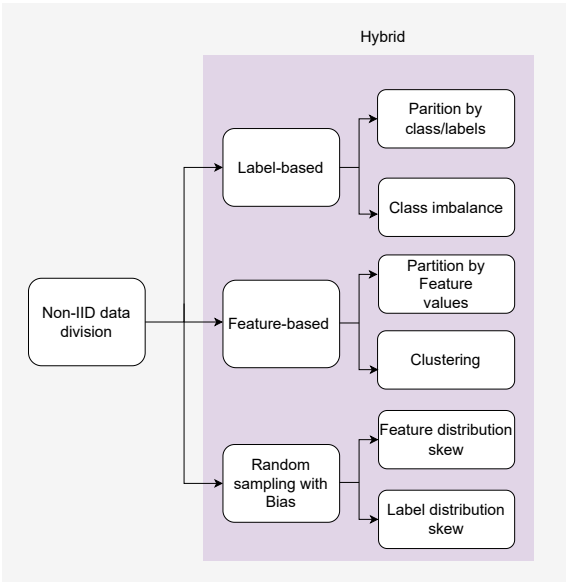
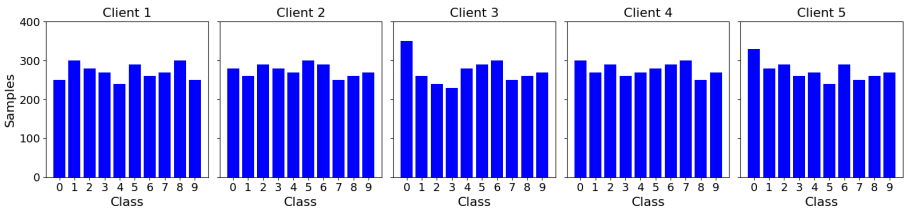


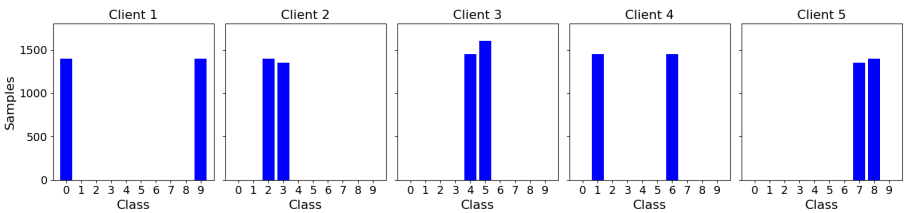
Figure 2.4: Non-IID data division strategy

receives data from only specific classes. For example, as shown in Figure 2.5b, 10 classes (0 to 9) are distributed among 5 clients. Client 1 might receive data from classes 0 and 8, client 2 from classes 3 and 4, client 3 from classes 1 and 6, client 4 from classes 2 and 7, and client 5 from classes 5 and 9. Class imbalance ensures that the class information available to clients is unevenly distributed. Techniques such as the Dirichlet distribution [Li+22], power law distributions [TTN20], are used to create this class imbalance among clients. For non-IID class distribution, it is important to consider imbalanced class proportions and partial class presence. For example, two clients might have data from classes 0 and 1, but client 1 could have 90% of its samples from class 0 and 10% from class 1, while client 2 could have 90% of its samples from class 1 and 10% from class 0. Additionally, class information can overlap, meaning that two clients might share the same classes but never have a complete representation of the entire class distribution. However, even though the two clients share similar class information, the samples they have are non-overlapping.

2. Feature-based division: The data can be distributed based on features in three ways: horizontal, vertical, and hybrid. In horizontal data division, all client devices have similar feature spaces but differ in samples. This means that the two clients do not have any overlapping samples. This type of data division is useful when the samples are the same, but the features differ between



(a) IID data division



(b) Non-IID data division

Figure 2.5: Data division MNIST dataset

clients. In the vertical data division, two clients do not have overlapping feature spaces, but they have overlapping samples. This type of division is commonly used in vertical federated learning. In hybrid data divisions, both horizontal and vertical divisions are combined. This means that two clients can have overlapping features and sample spaces.

Another method for dividing non-IID data based on features is clustering. Clustering algorithms such as K-Means [Llo82] are employed to group data points into clusters based on feature similarity. These clusters, which represent different feature spaces, can then be assigned to different clients. For example, in healthcare, data could be clustered based on patient demographics, such as age and health conditions, resulting in clients having datasets focused on specific patient groups.

3. Random sampling with bias: Another approach to creating non-IID data is to perform random sampling across clients with bias. It is important to incorporate bias to ensure that certain clients have a higher probability of receiving a certain type of data. This introduces skewness in the dataset, where a client can be biased towards particular features or classes.

2.4 Evaluation Metrics

In this thesis, we evaluated our proposed and state-of-the-art approaches using different metrics. The quality of the clusters is evaluated using the Silhouette Score. The performance of the model is evaluated across Accuracy, Precision,

Recall, and F1-Score. For the classification tasks, the loss is evaluated using Categorical Cross-Entropy (CE). Mean Absolute Error (MSE) and Root Mean Square Error (RMSE) are used to evaluate regression tasks. To perform the qualitative analysis of the clients for classification tasks, we used Confusion Matrix (CM). Wall-clock time is used to measure the time taken to train models in FL. The descriptions and equations for these metrics are provided in Table 2.1.

Table 2.1: Evaluation metrics

Metrics	**Description**	**Equation**
Accuracy	Quantifies the proportion of correct predictions (true positives (TP) and true negatives (TN)) out of the total number of predictions.	$\frac{TP+TN}{\text{Total Samples}}$
Precision	Measures the proportion of TP among all instances classified as positive (true positives and false positives (FP)).	$\frac{TP}{TP+FP}$
Recall	Measure the proportion of TP identified among all the actual positives (TP and False Negatives (FN)).	$\frac{TP}{TP+FN}$
F1 Score	The harmonic mean of precision and recall, balancing both metrics.	$2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision}+\text{Recall}}$
Categorical Cross-Entropy Loss	Measures the difference between predicted probabilities ($\hat{y}_c$) and actual class labels (y_c) for a multi-class (C) classification problem.	$-\sum_{c=1}^{C} y_c \log(\hat{y}_c)$
Confusion Matrix	Provides a detailed breakdown of model performance, displaying counts of TP, FP, FN, and TN.	$\begin{bmatrix} TP & FP \\ FN & TN \end{bmatrix}$
Wall-clock time	Measures the average training time (T_i) of the federated learning (FL) algorithm across all training iterations (T).	$\frac{1}{T}\sum_{i}^{T} T_i$
Mean Absolute Error (MAE)	The average (over N samples) of the absolute differences between predicted values ($\hat{y}_i$) and actual values (y_i) for a regression task.	$\frac{1}{N}\sum_{i=1}^{N} \lvert y_i - \hat{y}_i \rvert$
Root Mean Squared Error (RMSE)	The square root of the average (over N samples) of the squared differences between predicted values ($\hat{y}_i$) and actual values (y_i) for a regression task.	$\sqrt{\frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2}$
Silhouette Score	Measures how similar a data point is to its own cluster compared to other clusters. Here, a is the mean distance between a sample and all other points in the same cluster (intra-cluster distance), and b is the mean distance between a sample and all points in the nearest different cluster (inter-cluster distance).	$\frac{b-a}{\max(a,b)}$

2.5 Dataset Description

In this thesis, we evaluated the proposed and state-of-the-art approaches on various publicly available datasets. Table 2.2 provides descriptions of these datasets and references to the papers where they were utilized to demonstrate the efficiency of the proposed methods from the state-of-the-art. Among these, the Event Memorability and DIPA2 datasets are problem-specific. Additional details about these two problems can be found in *Chapter 6*.

Table 2.2: Datasets

Datasets	Description	Type	Paper
NSL-KDD99 [Tav+09]	Intrusion detection	Tabular	Paper I and III
ACC [Dal+17]	Credit card fraud transaction detection		
Wine [AF91]	Wine quality classification		Paper III
Vowel [Tur14]	Speech recognition from vowel sounds		
Vehicle [Kag20]	Vehicle type classification		
Segmentation [Sur]	Customer segmentation data for marketing		
WDBC [al95]	Breast cancer diagnosis		
Ionosphere [al89]	Radar scan classification		
Hill_valley [LF08]	Terrain classification		
ISOLET [RM94]	Isolated letter speech recognition		
Diabetes [DDD90]	Diabetes diagnostics		
IoT [Ant19]	Smart home activity recognition		
Synthetic [Ban+24a; Ban+24b]	Synthetic data for model evaluation		
MNIST [LeC+98]	Handwritten digit recognition	Grey-scale Images	Paper II and IV
FMNIST [Xia17]	Fashion item classification		
EMNIST [Coh17]	Extended handwritten digit and letter recognition		Paper IV
CIFAR-10 [KH09]	10-class object classification	Colored Images	Paper II and IV
CIFAR-100 [KH09]	100-class object classification		Paper IV
Event memorability [Xu+21b]	Event memorability prediction from visual lifelogs	colored Image and contextual text features (Multi-modal)	Paper V
DIPA2 [Xu+24]	An Image dataset with cross-cultural privacy perception annotations		Paper VI

2.5.1 Synthetic dataset

We generated the synthetic datasets for the experiments in papers III and IV using two strategies.

Strategy-I: This synthetic data is created for the federated feature selection experiments (Paper III [2]). The synthetic dataset has n samples, m features, and C classes. We divided m into three parts m_1, m_2, m_3. Among them, m_1 number of features are truly informative features, and m_2 features are linearly dependent on the m_1. The remaining $m_3 = m - m_1 + m_2$ are purely noise aimed

[2] `https://github.com/DevBhuyan/Horz-FL`

to reduce the learnability of models. A good feature selection algorithm will be able to filter this and provide high-quality data to a model. All the data points belong to one of the C classes. Then, we distributed the dataset among N clients. We ensured each client received the same features but non-overlapping samples. We incorporated skewness in the client data and ensured each client didn't hold information from all classes by doing label partition. For example, for a given iid ratio (IR), a client can have samples comprising not more than 20% of the total number of class labels if $IR = 0.2$. If the total number of classes in the dataset is 25, a single client would not have more than $\lfloor 25 \times 0.2 \rfloor = 5$ class labels. The first client is assigned samples having the first five class labels; the second client is assigned samples having another set of 5 class labels, and so on. No single client has more than (num_classes*IR) number of class labels, and the number of samples for each client is random.

Strategy-II: To conduct personalized multi-tier federated learning experiments (as described in Paper IV[3]), we create synthetic data by following these steps. First, we initialize the data's dimension d and number of classes C. We also set up a prior distribution, which includes weights and biases for each client, and initialize the covariance matrix of the data. Next, we generate unique weights and biases for each client. For IID data, all clients replicate the server's weights and biases. In contrast, for non-IID data, the weights and biases vary across clients. We then generate samples using a multivariate normal distribution and compute labels with softmax and argmax functions. Finally, the generated samples and labels are assigned to the respective clients, following a method similar to strategy-I.

[3] `https://github.com/sourasb05/PerMFL_1`

Chapter 3

Federated Feature Selection

Federated learning introduces unique challenges, such as data heterogeneity, that directly affect the learning of the model. It also affects several machine learning preprocessing tasks, such as feature selection, feature extractions, etc. This chapter focuses on the categorization of different feature selection methods. Federated feature selection methods and their comparative analysis.

3.1 Categorization of Feature Selection Methods

In many machine learning applications, such as text mining [Den+19], computer vision [BD01], and bioinformatics [WWC16], the amount of available data has dramatically increased, both in sample size and features. To facilitate knowledge acquisition, it is crucial to study methods for effectively utilizing these large-scale datasets. The presence of noisy, redundant, and irrelevant features can slow down the learning process, degrade performance, and complicate model explainability [Muñ+20]. Feature selection addresses these issues by identifying a small, relevant subset of features from the original set, thereby eliminating noise, redundancy, and irrelevant information.

Figure 3.1 illustrates the taxonomy of feature selection methods that include traditional machine learning, distributed learning, and federated learning. Feature selection methods are divided into three categories: (1) Label information-based methods, which can be further classified into three categories: supervised, unsupervised, and semi-supervised; (2) Search strategy-based methods, which can be further divided into four categories: filter, wrapper, embedded, and hybrid, and (3) Architecture-oriented methods that consider centralized, distributed, and federated settings. Federated feature selection is further classified into horizontal and vertical feature selection.

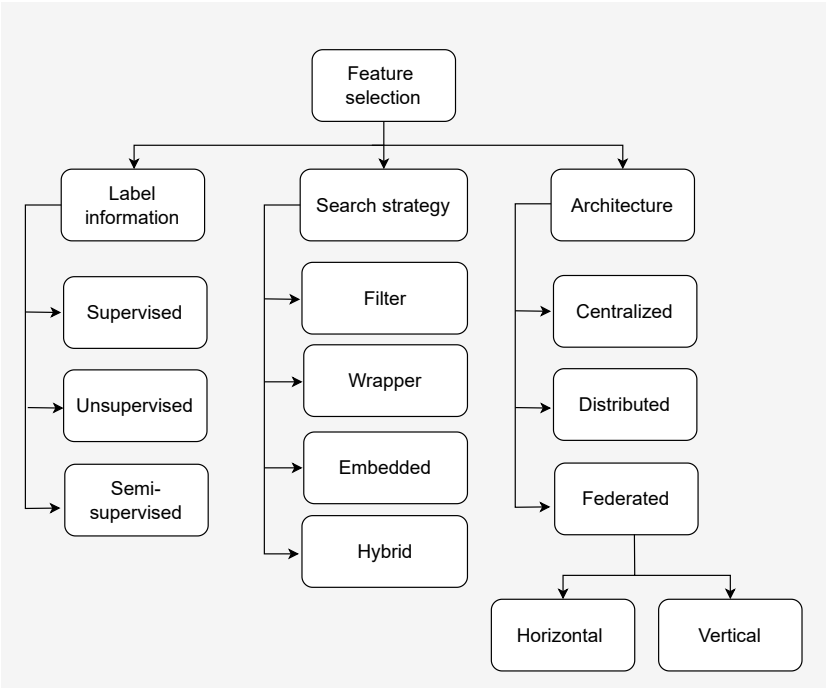


Figure 3.1: Taxonomy of feature selection methods

3.1.1 Label information-based feature selection

In supervised feature selection, label information is available for all examples. The feature selection algorithm efficiently selects relevant and non-redundant features that can effectively differentiate samples from different target variables [WSG05; Nie+10; Li+07; Hru+06]. In unsupervised feature selection, no label information is provided. This type of feature selection is unbiased and performs well when prior knowledge of the labels is unavailable. Additionally, it can reduce the risk of data overfitting compared to supervised feature selection methods [SCM20]. Semi-supervised learning falls between these two, where some examples have labeled information while others do not. This method is essential when the dataset contains limited labeled data but a large number of unlabelled data is available [ZL07; Xu+10]. Based on these three categories, feature selection algorithms also differ.

3.1.2 Search strategy-based feature selection

Search strategy-based feature selection techniques explore the feature space to select optimal subsets that maximize model performance and improve interpretability.

Filter

These methods select the most discriminative features based on the data characteristics. Typically, they perform feature selection before classification and clustering tasks, following a two-step strategy. First, all features are ranked according to specific criteria, and then the features with the highest rankings are selected. Many filter-type methods have been used, including correlation-coefficient [YL03], reliefF [RS04], F-statistic [DP05], mRMR [PLD05] and information gain [RS04]. For example, a feature with a high correlation to the target variable or one that significantly reduces uncertainty is considered highly relevant. One of the significant advantages of filter methods is their speed and scalability [HBL24], making them particularly suitable for high-dimensional datasets.

Wrapper

These methods employ the performance of the learning algorithms as the objective function to evaluate different subsets of variables. Evaluating all possible subsets (2^N) is an NP-hard problem. Therefore, wrapper methods employ heuristic search algorithms to identify suboptimal but effective subsets. These algorithms aim to minimize the subset of variables for which the performance, such as the classification accuracy of the learning models would be maximized. Various search strategies are used in wrapper methods. Sequential selection algorithms [PNK94], such as Sequential Forward Selection (SFS), start with an empty set and add one feature at a time toward setting a relevant feature set. It selects the feature that gives the highest value for the objective function and then evaluates the remaining features individually, adding them to the subset if they increase classification accuracy. This process continues until the required number of features are added. Another approach is Sequential Backward Selection (SBS), which is the opposite of SFS. It begins with the complete set and removes one feature at a time without compromising learning performance. More advanced versions like Sequential Floating Forward Selection (SFFS) introduce flexibility by allowing backtracking steps, further refining the feature subset. Heuristic search algorithms, such as Genetic Algorithms (GA) [Gol89] and Particle Swarm Optimization (PSO) [KE95], explore the search space broadly.

Embedded

These methods [BL97; Lan+94] incorporate feature selection directly into the model training process, reducing computation time effectively compared to wrapper methods, which evaluate multiple subsets of features. Embedded methods aim to identify the most relevant features while the model is being built, optimizing both feature selection and model performance simultaneously. These methods can dynamically adjust important features based on the model's learning progress by integrating feature selection into the training phase. Embedded feature selection algorithms use regularization techniques in linear models, such as Lasso (Least Absolute Shrinkage and Selection Operator) [FB17]. Lasso introduces a penalty term to the loss function, which shrinks certain coefficients to zero. This effectively performs feature selection by excluding less important features. Decision tree-based methods [WL08], such as random forests [ZZL16] and gradient boosting [Otc+22], also naturally perform feature selection by selecting features that best split the data at each node. Recursive feature elimination of the support vector machine (SVM-RFE) [Guy+02] is another example, which iteratively removes the least essential features based on the weights assigned by the SVM. Neural networks can also employ embedded methods by pruning connections during training, retaining only the most influential features. Embedded methods offer a balance between computational efficiency and effective feature selection, making them suitable for large datasets and complex models.

Hybrid

These methods combine the strengths of the filter and wrapper methods to enhance the feature selection process. Hybrid methods balance computational efficiency and selection accuracy. These methods involve two steps. First, a filter method is applied to perform the initial screening of features based on statistical criteria, quickly reducing the dimensionality of the dataset. This step ensures that only the most promising features are considered for the subsequent computationally intensive step. In the next phase, a wrapper method is applied to the reduced feature set to perform a more thorough assessment of features and their impact on the model's performance. By focusing only on the reduced subset, hybrid methods mitigate the computational burden that original wrapper methods often face when dealing with high-dimensional data. For example, a common hybrid approach [Sun+10] might involve using a Mutual information-based filter to select the top features, followed by a sequential forward selection (SFS) wrapper to fine-tune the feature set based on cross-validation performance.

These four feature selection methods are visually represented in Figure 3.2.

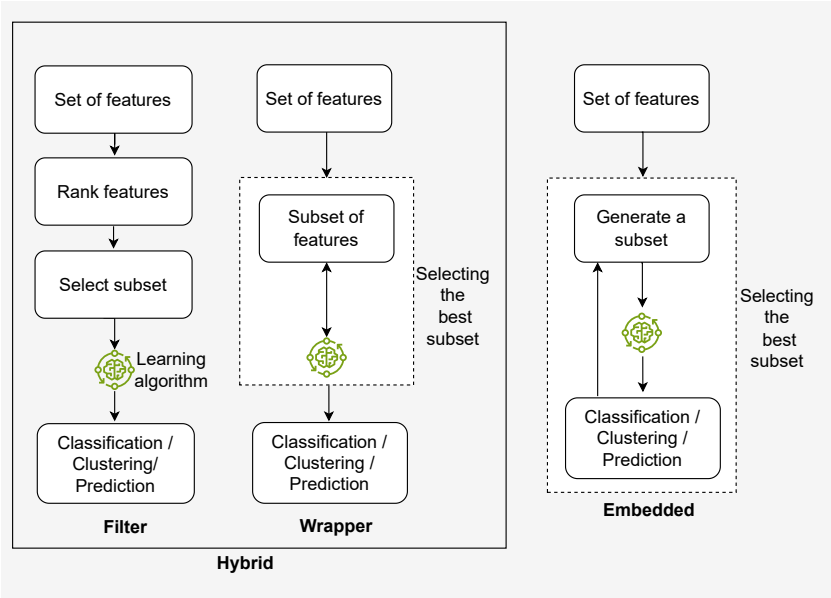


Figure 3.2: Search strategy-based feature selection

3.1.3 Architecture-based feature selection

Label information-based and search strategy-based feature selection can be utilized in various architectures, including centralized, distributed, and federated.

Centralized feature selection

In centralized feature selection, all the data is collected and stored in a single location, where the feature selection process is conducted. This method has the full dataset's information to identify the relevant features to improve the performance of the learning model, reduce computational complexity, reduce training time, and enhance interpretability. Centralized feature selection faces several challenges regarding handling large-scale data and privacy concerns. Handling large-scale datasets in a centralized manner requires significant computational and storage. Moreover, centralizing data from multiple sources can raise privacy and security issues, as it often involves transferring sensitive information to a single location, which might not be feasible due to legal and regulatory constraints, for example, GDPR [PE16] in the EU.

Distributed feature selection

Traditionally, feature selection methods have been developed for centralized computing environments. However, many distributed methods have emerged as alternatives to centralized approaches [MBA17]. Two main factors drive this paradigm shift. *Firstly*, advancements in network technologies have led to an increase in dataset sizes across various application fields, raising significant privacy concerns as data is often dispersed across geographical and organizational boundaries. It is often neither economical nor legal to consolidate such data in one location. *Secondly*, existing feature selection algorithms struggle to scale efficiently with large datasets, sometimes becoming impractical due to performance deterioration. A potential solution involves distributing the data, applying a feature selection method to each subset, and then merging the results. Data distribution can be either horizontal or vertical. Horizontal partitioning divides the dataset into multiple segments, each containing the same features as the original dataset but with a subset of the instances (see Figure 3.3a). Vertical partitioning, on the other hand, splits the dataset into several segments, each containing the same number of instances but with a subset of the original features (see Figure 3.3b).

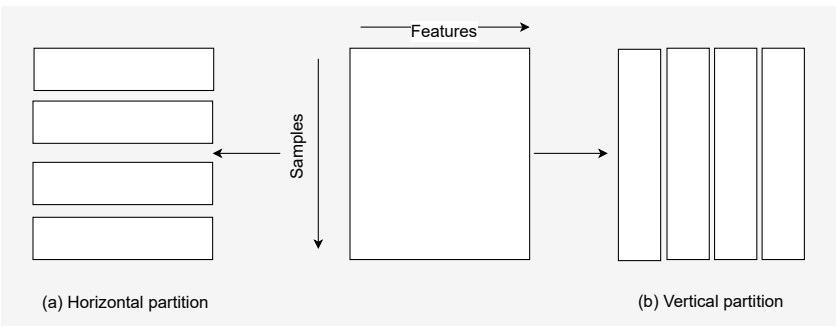


Figure 3.3: Distributed data partition

Federated feature selection

Federated learning shares similarities with distributed learning but encounters unique challenges such as data heterogeneity (non-IID nature of data), data scarcity, and model training on resource-constrained clients. These issues mean that many distributed feature selection techniques that perform well on IID data struggle with non-IID data. Additionally, wrapper or embedded methods work efficiently on distributed setups, which becomes very time-consuming on resource-limited clients in FL setups. Unlike distributed learning, which is typically designed to address big data problems and does not experience data scarcity, federated learning often deals with data scarcity and unbalanced la-

bel distributions. Therefore, specialized algorithms are essential for federated settings to overcome these challenges effectively. Federated feature selection can be categorized into two types based on data partitioning: Horizontal Feature Selection (HFS) and Vertical Feature Selection (VFS). HFS algorithms are designed for horizontal federated learning setups, while VFS algorithms are suitable for vertical federated learning scenarios.

Horizontal feature selection: Horizontal Federated Learning (HFL) enables learning with multiple clients using their private data sets, each of which carries an overlapping set of features, to jointly train a global model without sharing raw data. Feature selection is important in HFL to train a uniform global model across clients and enhance model performance by filtering out noisy and redundant features. Various methods have been proposed for HFS. For instance, In [Mas+22], LASSO regularization is utilized to perform embedded feature selection that reduces overfitting and enhances the interpretability of the global model while preserving the privacy of the client's data. Similarly, Qin et al. [QK21] employ a greedy algorithm, a wrapper method, which customizes feature selection based on attack types in intrusion detection systems. The mRMR approach in [HBL24] maximizes relevance and minimizes redundancy of features in an HFL environment. Conditional mutual information [Fu+23b] and mutual information with correlation distance [Zha+23b] are filter methods that enhance accuracy and communication efficiency. In [MK24], the Pareto-based bi-objective strategy effectively manages multi-label data, highlighting the nuanced trade-offs in horizontal feature selection. We proposed two HFS approaches, Fed-FiS [BEB21] and Fed-MOFS [Ban+24a], that use a mutual information-based approach to find relevant features. We give a comparative analysis of the HFS methods in Table 1 of paper-III.

Vertical feature selection: Vertical Federated Learning (VFL) enables multiple clients with their private datasets, each holding different subsets of features for largely overlapping datasets, to jointly train a global model without sharing raw data. Feature selection (FS) is important in VFL to enhance model performance by filtering out noisy and redundant features. Various methods have been proposed for VFS. For example, Zhang et al. [Zha+22c] employ Secret Sharing and Homomorphic Encryption to ensure privacy while performing FS. Another method uses Particle Swarm Optimization [Zha+23c], suitable for large-scale FS. Castiglia et al. [Cas+23] introduce Local Lasso and Group Lasso methods, balancing accuracy and communication costs. Feng et al. [Fen22] proposed a deep learning-based autoencoder that generalizes well to different data types but demands synchronous learning and incurs high computational costs. Jiang et al. [Jia+22] leverage Group Testing and Homomorphic Encryption to efficiently select clients, though it relies on a trustworthy key server. Li et al. [Li+23a] presented the FedSDG-FS that uses Gaussian Stochastic Dual-Gate with Partially Homomorphic Encryption and Gini Impurity for effective feature selection. These methods highlight the trade-offs between efficiency, accuracy,

and privacy in VFS. A comparative analysis of these vertical feature selection methods is given in Table 3.1.

Table 3.1: Summary of VFS methods

FS Method	Category	Advantages	Disadvantages
Secret sharing and homomorphic encryption [Zha+22c]	Embedded	Highly compatible with most of the ML models. Effective privacy preservation. Significant accuracy gains	High communication overhead for secret sharing. Moderate runtime for homomorphic encryption.
Particle Swarm Optimization (PSO) [Zha+23c]	Embedded	Suitable for large-scale feature selection. Fast convergence	Requires sharing some insensitive model parameters.
Local lasso and group lasso [Cas+23]	Wrapper, Embedded	Removes spurious features efficiently. High accuracy with selected features. Low communication cost	High pre-training epochs for some datasets. Complexity in regularization parameter tuning.
Deep learning-based Autoencoder [Fen22]	Embedded	Generalizable to different data types	Requires synchronous learning. High computation and communication burden with many clients.
Gaussian stochastic dual-Gate with PHE and gini impurity [Li+23a]	Embedded	Efficient in filtering noisy features. High accuracy in feature selection. Reduces communication. Secure through PHE and random noise	Have moderate computation overhead due to encryption.

Chapter 4

Straggler-Mitigation and Client Selection

The efficiency of Federated Learning (FL) is greatly affected by system heterogeneity, including diversity in clients, network conditions, and user behaviors, which can delay training. To accelerate FL, it's crucial to identify stragglers—clients with limited resources or poor connectivity—and implement a straggler-resilient client selection strategy. This chapter discusses the system heterogeneity, stragglers, and client selection strategies in detail.

4.1 System Heterogeneity

System heterogeneity can be categorized into (1) heterogeneity in clients' computational resources and (2) heterogeneity in the communication channel. Both aspects affect federated learning.

4.1.1 Heterogeneity in clients' computational resources

Heterogeneity in computing resources is a prevalent issue in cross-device FL, leading to variations in individual training times and causing delays in the overall FL process. For example, the computing devices in clients could range from low-end, such as the Raspberry Pi, to high-end, like GPUs (see Table 4.1). These devices[1] differ in terms of floating-point operations per second (FLOPS) and internal memory (RAM) capacity, which impacts their ability to participate in FL efficiently. Moreover, different neural network designs (see Table 4.2) have different Multiply-Accumulate (MAC) operations for a forward pass and, therefore, consume different amounts of memory. It usually takes three to five times as many MAC operations to train a full epoch because training includes a backward pass with about twice as many MACs as the forward pass [Amo+18],

[1] In this chapter, client and device are used interchangeably to represent the same entity.

plus any extra calculations needed for a stateful optimizer [2]. Additionally, a device must process hundreds to thousands of local training samples per FL round.

Table 4.1: Computational resources from low-end to high-end devices [Pfe+23]

Device	FLOPS	RAM
Raspberry Pi Ser.	10^8–10^{10}	512 MB–8 GB
Low-End Smartphones	10^{10}–10^{11}	1–2 GB
Nvidia Jetson Nano	10^{11}–10^{12}	2–4 GB
High-End Smartphones	10^{11}–10^{12}	4–8 GB
Server GPUs	10^{13}–10^{14}	32–100 GB

Table 4.2: The resource requirements for training a neural network model in PyTorch [Pas+19] on the CIFAR-10 dataset [KH09] with a batch size of 32 [Pfe+23]

ML Model	# MACs (Forward)	Memory (Training)
LeNet [LeC+98]	$6.7 \cdot 10^5$	0.5 GB
ResNet18 [He+16]	$5.6 \cdot 10^8$	0.8 GB
EfficientNet [TL19]	$3.2 \cdot 10^7$	0.9 GB
MobileNetV2 [San+18]	$9.6 \cdot 10^7$	1.4 GB
ResNet152 [He+16]	$3.7 \cdot 10^9$	5.3 GB

Not all devices are appropriate for federated learning. For instance, while a Raspberry Pi can train a neural network, it experiences significant delays. In contrast, powerful GPUs can perform the same task up to 100× faster. Consequently, the inclusion of these low-end devices in a federated learning system can increase training latency. These low-end devices are called stragglers in FL. The definition of stragglers is given below,

Definition 1. ***Stragglers:*** *In a federated learning system consisting of N devices, the computational power, communication capabilities, and storage capacity of each device can vary significantly. A subset of devices denoted as k (where $k \leq N$) may cause delays in the overall training due to their limited resources. These k devices are referred to as stragglers.*

4.1.2 Categorization of computation constraints

There are two main categories of computation constraints [Pfe+23] that make a device straggler, such as hard constraints and soft constraints.

Hard constraints, such as the limited memory capacity of a device, may hinder the training of a Deep Neural Network (DNN) model. For instance, MobileNetV2 consists of millions of parameters that require a significant amount of memory for training. If the selected model for FL exceeds the memory capacity of the available device, then that device will be unable to participate in the learning process. Despite potentially having high-quality data that could contribute to the learning, the device's limited resources prevent it from taking part.

The soft constraints allow DNN training on the device, but it hinders the device's ability to achieve high throughput, such as FLOPS, during the learning process. Soft constraints can cause a device to become a straggler. Several factors, as mentioned in [Pfe+23], such as micro-architecture, unstable power supplies, component degradation, or shared resource contention, can lead to slow model training. Soft constraints enable devices to participate in the learn-

[2] A stateful optimizer is an optimization algorithm that maintains and updates additional state information, such as momentum or adaptive learning rates, beyond just the model weights to enhance the training process.

ing process but restrict them from completing the training. The server rejects updates that are out of sync, preventing their incorporation into FL training.

4.1.3 Properties of heterogeneity

These constraints cause heterogeneity across devices or over the training iterations.

Heterogeneity across devices: The hard constraints are determined either at the design stage or prior to the commencement of training and remain constant over time. Various devices may have distinct constraints that impact their training throughput. For instance, there is notable heterogeneity among smartphone devices. Table 4.1 illustrates the differences in FLOPS and internal memory between a low-end smartphone and a high-end smartphone. For the same learning task, a low-end smartphone will have lower throughput compared to a high-end smartphone.

Heterogeneity over global iterations: Soft constraints of devices in FL systems can be influenced by the environments in which the devices operate and may vary across different training rounds. These constraints include factors such as the expected battery level during training, the current power supply, or ambient temperatures. These constraints are characterized by their relatively slow rate of change, typically occurring over minutes to hours, allowing them to be predicted and known before the start of an FL round. To accommodate such heterogeneity across rounds, FL algorithms should support dynamic changes in device resource availability. This adaptability ensures that the FL system can effectively manage variations in device performance and resource constraints, maintaining efficient and reliable training processes despite the diverse and fluctuating conditions of participating devices. This approach is crucial for optimizing the overall training throughput and robustness of federated learning systems.

4.2 Communication Heterogeneity

In FL, knowledge can be transferred to other devices and servers through gradient sharing, sharing model weights, logits, predictions, intermediate representations, and loss values. Since client devices are located far apart, this sharing is only feasible through communication over the network. The sharing of knowledge relies on the speed, timing, and trustworthiness of the communication channels. We can observe that these aspects are constrained across various slower client devices, which has hindered the overall FL system performance. Moreover, modern DNN models can have millions of parameters, therefore, transferring models with server or clients increase communication cost which makes the system non-scalable. To overcome this bottleneck, model

quantization [AG20] and compression [MHM19; SL21] based techniques have been utilized to accelerate the transmission. The communication-efficient process [Yu+15] begins with compressing the DNN model by removing redundant parameters that are not useful and keeping only the non-redundant parameter connections. Then, the shared weights are quantized to retain the effective weights and avoid redundant weights on the client devices. Similarly, when gradients are shared in FL, we can use gradient compression [Kam+19] to reduce the communication overhead.

4.3 Client Selection

Client selection has emerged as a significant area of research to address system heterogeneity issues in federated learning. In this process, the server determines which client devices to include in a global training round. Effective client selection can significantly enhance the global model's performance and minimize training overheads. In Figure 4.1, every client is measured for its utility or priority before each global round. The clients with high priorities are selected for model training and aggregation.

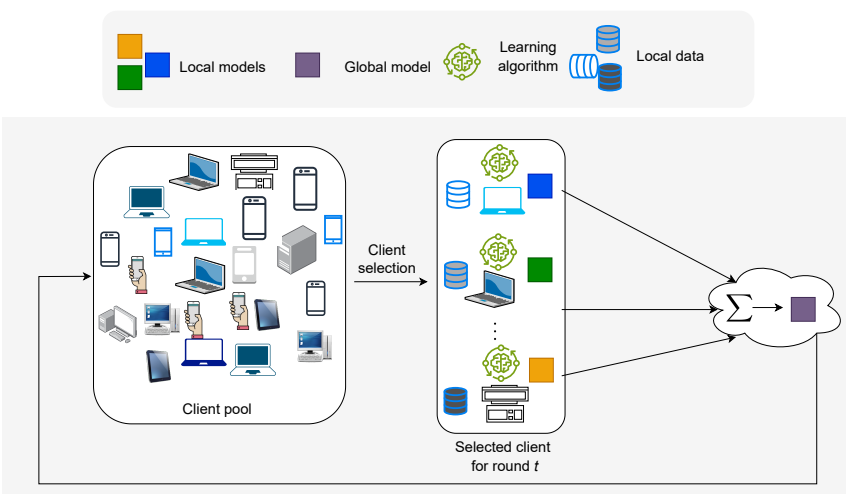


Figure 4.1: Client selection in FL

Figure 4.2 illustrates the client selection methods, which can be based on statistical or system information of devices.

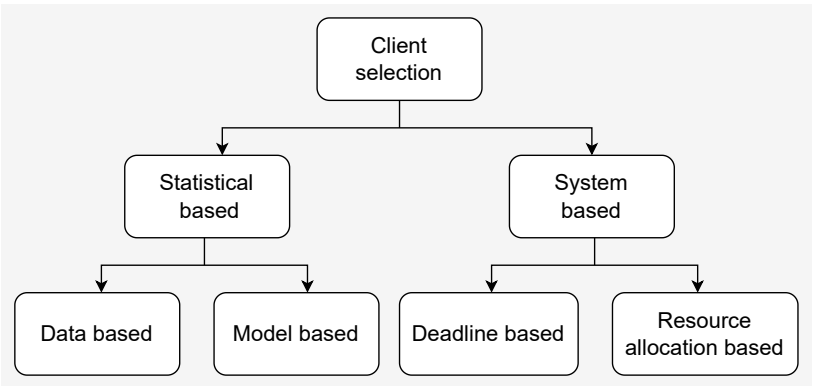


Figure 4.2: Client selection methods in FL

4.3.1 Statistical information based selection

Statistical information indicates the impact of a client's local update on the global model. This type of selection is further categorized into data sample-based selection and model-based selection approaches.

Data sample-based selection

Client selection in FL uses local data samples to determine their statistical utility. One method of measuring utility is based on the number of data samples a client has. Another approach involves giving importance scores [ZYZ21] to data samples, which is calculated using the L2-norm of each sample's gradient. An optimal solution can be represented as the average L2-norm of the gradients of a client's data samples. Alternatively, the loss value of each data sample can be used as a proxy for its gradient norm to reduce computational overhead. The cumulative loss [CWJ22] on a client's data samples is also an effective measure for client selection.

Model-based client selection

This approach quantifies the contribution of a client's local model updates to the global model. One method involves calculating the normalized model divergence, which is the average difference between the client's model weights and the global model weights [Hsi+17]. A smaller difference means that the local update has less impact on the global model. Another method measures the percentage of same-sign weights [LWB19] between the client model and the global model that represents the local model and global model aligned in the same direction. A few methods assess the similarity to the convergence trend, evaluating how the client's model weights move relative to the global model's

convergence direction [SWR20]. Comparing the change in a client's model before and after local training [Zha+22d] or using the L2-norm of the local model's gradients [CHR20] can also indicate the client's potential contribution. These techniques help prioritize clients whose updates are expected to be most beneficial for the federated learning process.

4.3.2 System-based client selection

System utility-based selection in federated learning takes into account the different hardware configurations and system capabilities of clients, which can significantly impact the overall training performance. This method prioritizes clients based on factors such as computation power, communication bandwidth, and energy availability to avoid prolonged training rounds due to straggler or resource-constrained devices. Here, we discuss two approaches: the deadline-based and the resource allocation-based.

Deadline-based client selection

This approach is typically remove clients who can't complete their local update within the set time frame in a synchronous FL aggregation system. The time limit can be either hard [NY19], meaning clients that take too long will be removed from the system, or soft [Lai+21], meaning the server will impose penalties on slow clients. This adds flexibility for the next round, where the straggler devices may send their updates on time. This deadline can be based on various metrics like computation speed (FLOPs) or transmission bandwidth (Mb/s). Additionally, algorithms like FedBalancer automatically adjust these deadlines to optimize performance. These strategies help to mitigate the adverse effect of system heterogeneity and offer a more efficient and balanced federated learning process. Adaptive client participation can significantly improve the global model's performance with fewer communication rounds [BVB22; Rei+22].

Resource allocation-based client selection

Optimized resource allocation is critical for Federated learning in presence of system heterogeneity. Given the decentralized nature of FL, the selection of clients and the allocation of resources are important to address the expensive communication costs and fairness. Resource allocation-based client selection methods consider factors like computational capacity, communication bandwidth, and energy consumption across clients. Selecting clients with higher computational resources and better bandwidth can reduce training time and improve the convergence of the global model [BVB22]. but this needs to be balanced with fairness and the inclusion of diverse data sources to prevent biases. Client selection approaches include managing clients based on real-time resource conditions, utilizing bandwidth allocation under long-term energy constraints, and using fuzzy logic to optimize selection. Efficient resource

allocation is crucial in mitigating the straggler effect, where training is delayed due to slower clients. Strategies to improve resource allocation involve prioritizing clients based on resource availability and energy consumption, maximizing participating clients while minimizing energy use, and select the best subset of clients. Metrics to evaluate client selection strategies typically include testing accuracy versus communication rounds and energy consumption versus convergence time. Studies indicate that reducing communication rounds and optimizing resource usage significantly enhances the efficiency of FL systems.

A comparative analysis of the client selection methods discussed is presented in Table 4.3. This analysis reveals that combining statistical information-based and system-based client selection methods can mitigate the effects of stragglers and data heterogeneity. Hence, a hybrid algorithm is required to balance the system and statistical heterogeneity effectively.

Table 4.3: Comparative study of different client selection methods

Client selection approach		**Benefits**	**Drawbacks**
Random selection [McM+17a]		Unbiased selection of clients ensure fairness.	Performance is lower in the presence of data and system heterogeneity. Training is also slower because of stragglers.
Statistical information based	Data Sample-Based [ZYZ21]	Optimized client selection based on data contribution that improves model accuracy	High computational overhead for importance sampling. Doesn't work well in presence of non-IID data.
	Model-Based [Hsi+17]	Prioritize clients based on model updates, and also this method enhances the convergence of global model.	This method works more during training due to the presence of stragglers.
System based	Deadline-based [NY19]	It reduces training time and mitigates stragglers effects	might overlook data heterogeneity. It may not maintain fairness.
	Resource allocation-based [BVB22]	Efficient management of resources for FL training across clients. It reduces energy consumption and communication costs	Might not take data heterogeneity into consideration.

4.4 Fairness in Client Selection

The selection of clients to address system heterogeneity may affect the overall performance of the global model due to the presence of statistical heterogeneity in the data. Therefore, the client selection algorithm should ensure effective and fair participation from all clients in the training process.

Fairness in client selection is defined as the uniformity of performance dis-

tribution across all participating clients, which means the performance of the global model should be relatively consistent across different clients, regardless of their individual data volume or computational capabilities. Client selection algorithms favor clients with more resources or faster response times, which can lead to biases in the global model that make the model less generalizable. For example, if the algorithm consistently selects only the fastest clients and deprives the participation of stragglers, that leads to a skewed representation of data and potential biases in the trained model. The impact of unfair client selection may lead to an increase in the number of training iterations without a guarantee of convergence.

Therefore, it is important to consider both the data heterogeneity and resource conditions of clients to achieve a fair client selection. Huang et al.[Hua+20b] introduced an FL algorithm that prioritizes fairness. The algorithm aims to enhance the overall performance of the global model while also ensuring that slower clients are not left out of the training process. Although this approach may reduce training efficiency, it results in a more balanced performance of the model across all clients. Another method, proposed by Jee Cho et al. [Cho+20], involves improving fairness through biased client selection based on higher local losses. This approach guarantees that clients with lower performance in previous global rounds are given the opportunity to participate in the next round.

Chapter 5

Personalized Federated Learning

In Federated Learning (FL), data may not be Independent and Identically Distributed among the participating devices. As a result, a global model generated using traditional federated learning algorithms such as FedAvg [McM+17a] may not perform equally well on all participating clients. To address this problem, Personalized Federated Learning (PFL) is introduced. In this chapter, we explore personalized federated learning, including its motivation, formulation, and various methods. We also discuss the advantages and disadvantages of different PFL approaches.

5.1 Motivation for Personalized Federated Learning

In Centralized Machine Learning (CML) [Dra+20], (see Figure 5.1a) data is gathered from end devices and sent to a cloud server for training a machine learning model. During training, the CML framework has access to the entire data distribution (IID), which aids in training a well-generalized model. However, CML does not prioritize privacy, as the data owner loses control over the data once it is shared with the server. Additionally, with the current surge in data generated at edge devices, predominantly images or videos, transferring this data to the server consumes bandwidth and increases cost.

To address these challenges, federated learning was introduced [McM+17a]. The FL settings (see Figure 5.1b) involve a federation of distributed clients, each with its own private local datasets. Creating isolated models for each client is not very useful because many clients suffer from a data scarcity problem that limits their ability to train effective models using only their own data. Therefore, they are encouraged to join the FL process, where even with a

limited contribution, they can obtain a better-performing model.

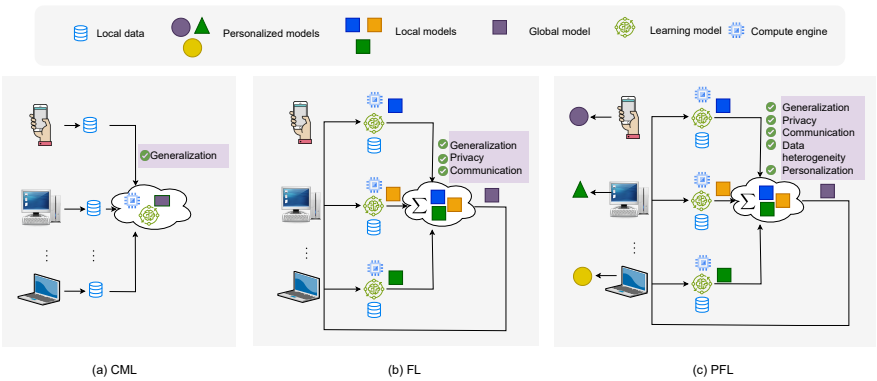


Figure 5.1: Illustration of the difference among (a) Centralized Machine Learning (CML), (b) Federated Learning (FL), and (c) Personalized Federated Learning (PFL)

The FL approach has fundamental challenges, such as the convergence issue of the global model on heterogeneous data and lack of solution personalization [Tan+22]. The accuracy of FedAvg is significantly reduced when training the model on non-IID data. This occurs due to client drift [Kar+20]. It refers to the phenomenon where the local models trained on different clients start to diverge significantly from each other and from the global model due to differences in local data distributions. Figure 5.2 depicts the effect of client drift on IID and non-IID data. The server updates move towards the average of client optima in FedAvg. When the data is IID, the averaged model $\bar{w}$ is close to the global optimum w^*, as it is equidistant to both local optima w_1^* and w_2^*. However, when the data is non-IID, the global optimum w^* is not equidistant to the local optima. In this case, w^* is closer to w_1^*. Consequently, the averaged model $\bar{w}$ will be far from the global optimum w^*, and the global model does not converge to its true global optimum. Therefore, FedAvg does not converge on non-IID datasets, and careful tuning of hyperparameters is needed to obtain learning stability [Li+19]. In the traditional FL setup, a single global model is trained to fit the average client. Consequently, this global model may not perform well for local distributions that significantly differ from the overall distribution. Relying on a single model is typically inadequate for real-world applications, which frequently encounter non-IID local datasets. Therefore, personalized federated learning is required to provide customized solutions to each client.

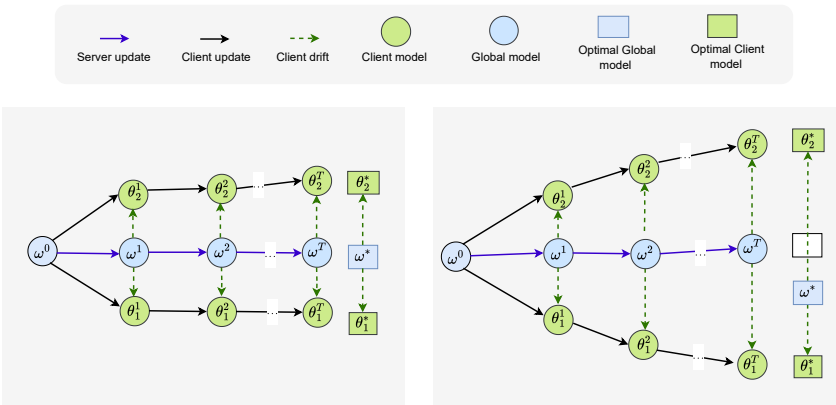


Figure 5.2: Illustration of client drift in FedAvg [McM+17a] for two client models (θ_1 and θ_2) on (a) IID and (b) non-IID settings

5.2 Formulation of Personalized Federated Learning

In the PFL model training, the optimization objective is formulated differently from the conventional FL settings. The global model objective aims to minimize the aggregate loss of overall clients' data distributions,

$$w^* = \arg\min_{w \in \mathbb{R}^d} \sum_{i=1}^{N} p_i F_i(w) \tag{5.1}$$

where N is the number of clients. p_i is the relative importance or weight of the i^{th} client often set to $\frac{n_i}{n}$, where n_i is the number of training samples at client i and n is the total number of training samples across all clients. $\mathcal{F}_i(w)$ is the loss function for client i, which depends on the model parameters w, defined as,

$$F_i(w) = \mathbb{E}_{(x,y)\sim D_i}[F_i(w; x, y)] \tag{5.2}$$

which is the expected loss over the data distribution D_i of client i. The FL formulation results in a uniform output for all clients using the global model, lacking personalization. In the presence of data heterogeneity, this approach will lead to subpar performance.

If we consider local models where clients aim to optimize their model θ_i locally without participating in the FL process, then each client will optimize the following problem on their private data.

$$\theta_i^* = \arg\min_{\theta_i \in \mathbb{R}^d} F_i(\theta_i) \tag{5.3}$$

where $F_i(\theta_i)$ is the local loss function for the i^{th} client. Here, the resulting model may not be well generalizable as the training samples are generally limited, and they do not participate in FL, so no knowledge sharing has been done.

In order to achieve a balance between generalization and personalization, the use of PFL is beneficial. PFL can be implemented in various ways, one of which involves including a regularized penalty term to the local objective to ensure that the personalized model θ_i remains close to the global model w:

$$\theta_i^* = \arg\min_{\theta_i} F_i(\theta_i) + l(\theta_i, w)) \tag{5.4}$$

where $l(\theta_i, w)$ is a regularization penalty term, which is typically formulated as a function of the global model and the local model of client i.

5.3 Strategies for Personalized Federated Learning

In Figure 5.3, we illustrate the categorization of personalized federated learning methods. These methods are divided into four main categories [Tan+22] such as model-based, data-based, architecture-based, and similarity-based. Model-based approaches focus on creating a strong global model for the future personalization of individual clients or improving the adaptation performance of local models. On the other hand, data-based approaches aim to minimize statistical differences among clients' datasets to address the client drift issue. Architecture-based approaches aim to develop personalized model architectures for each client. In contrast, similarity-based approaches utilize client relationships to enhance personalized model performance by constructing similar models for related clients.

5.3.1 Model-based approach

Model-based global model personalization focuses on learning a well-generalized model from which personalized models will be fine-tuned for each client or improving the adaptation performance of the local model. These methods are categorized into three parts: regularization-based (see Figure 5.4a), model adaptation through meta-learning (see Figure 5.4b), and transfer learning-based FL (see Figure 5.4c).

Regularization: Regularization techniques help to reduce the impact of local updates, which ultimately enhances the convergence and generalization of the global model. This approach is also employed to create a better-personalized model. It involves minimizing the objective function $F_i(\theta_i, w)$, which depends on the global model and the local model of i^{th} client (see Equation (5.4)). By regulating the variance between global and local models, the issue of client drift

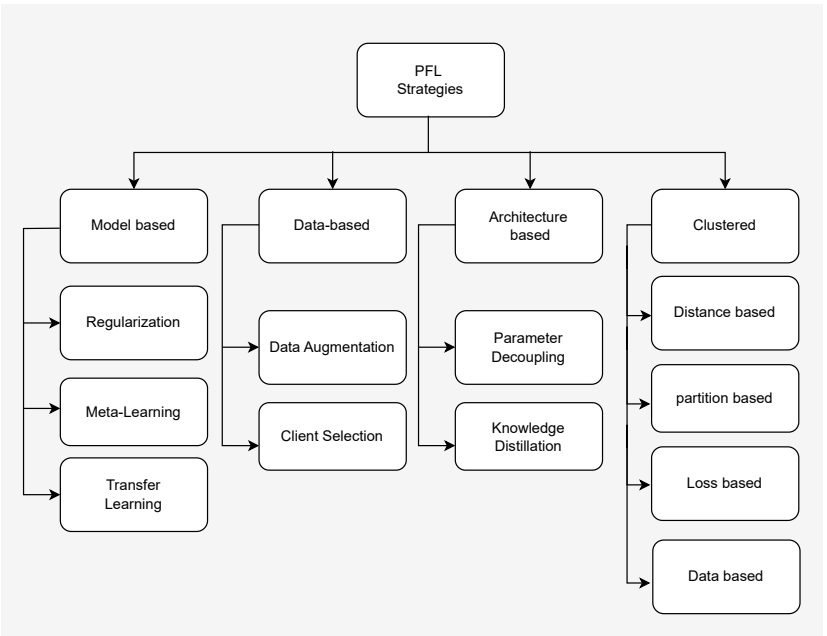


Figure 5.3: Taxonomy of PFL methods

is addressed [Li 20]. Another approach is to consider the history of the local models for personalization. MOON [LHS21] reduces the difference between the representations learned by the local models and global models in order to diminish the weight divergence. It also increases the difference between the current local model and the previous local model to accelerate convergence.

Meta-learning: The goal of meta-learning is to train a model on a diverse set of tasks so that it can efficiently tackle new tasks with only a few training examples. For these new tasks, the model should achieve good generalization with just a few gradient steps and minimal data. Model-Agnostic Meta-Learning (MAML) [FAL17] provides superior generalization and fast adaptation to new tasks. Meta-learning can also be applied to FL [Jia+19]. It involves two phases: meta-training and meta-testing. The meta-training phase corresponds to the global model training in FL, while the meta-testing phase is associated with the FL personalization step (see Figure 5.4b). FedAvg can be combined with MAML, which changes the global objective (see Equation (5.1)) into the following [FMO20],

$$w^* = \arg\min_{w \in \mathbb{R}^d} \sum_{i=1}^{N} p_i F_i(w - \alpha \nabla F_i(w)) \tag{5.5}$$

Here $\alpha > 0$ is the learning rate. The global objective is the average of meta-functions F_i of N clients. $F_i(w - \alpha \nabla F_i(w))$ is the meta-function associated with the i^{th} client. In FL, meta-learning also offers a better initialization that helps a personalized model to converge more quickly.

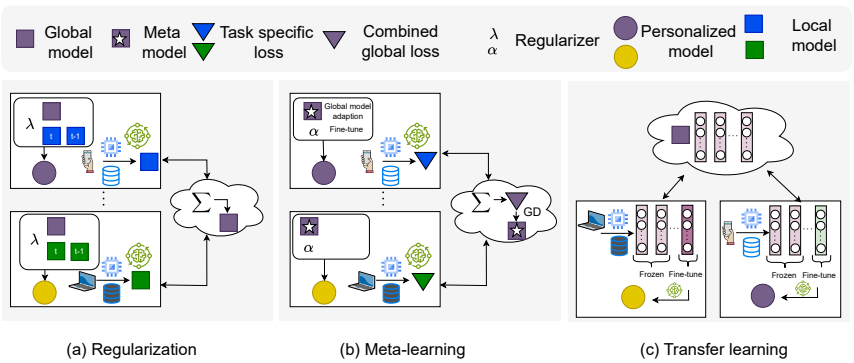


Figure 5.4: Model-based methods (a) Regularization, (b) Meta-learning, and (c) Transfer learning

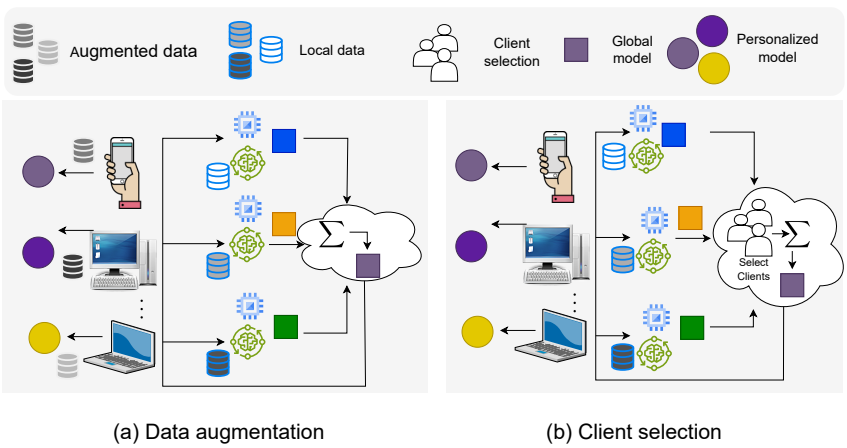
Transfer learning: Transfer Learning (TL) [PY09] is an effective technique for model adaptation, allowing knowledge transfer from a pre-trained model to a new task. It aims to transfer knowledge from a source domain to a related but different target domain. Domain adaptation techniques in TL are often used for PFL. These techniques aim to minimize the domain discrepancy between the global FL model (source domain) and local models (target domain) for better personalization [Che+20]. The process (see Figure 5.4c) typically involves three steps: (1) training a global model with FL, (2) adapting the global model to local data to train local models, and (3) refining local models into personalized models using the global model through TL. To minimize training overhead in deep neural networks, the lower layers of the global model, which capture generic features, are often transferred and reused in local models. The remaining layers are fine-tuned with local data to learn task-specific features for personalization.

5.3.2 Data-based methods

Data-based methods aim to reduce data heterogeneity issues for the client and improve generalization. This method is further classified into data augmentation and client selection.

Data augmentation: In machine learning, data augmentation methods are required to enhance the statistical homogeneity of the data and help to reduce data imbalance. Oversampling [Cha+02] and downsampling [KM+97] are two useful methods for balancing the data. Data augmentation in federated learning (see Figure 5.5a) is challenging because it usually requires some degree of data sharing [Zha+18], which violates the privacy constraints of federated learning or relies on having access to a proxy dataset that accurately represents the overall data distribution. Training a Generative Adversarial Network (GAN) based model on the FL server [Jeo+18] is useful to generate additional data that are distributed to the clients to improve model performance.

Client selection: This mechanism enables sampling from a more homogeneous data distribution, which improves the generalized performance of the model (see Figure 5.5b). Selected clients help to mitigate the bias due to non-IID data [Wan+20]. Tire-based FL (TiFL) [Cha+20] is also useful for reducing client drift by grouping clients into tiers based on their performance and adaptively selecting clients from the same tiers for each global round to optimize training time and improve accuracy.



(a) Data augmentation (b) Client selection

Figure 5.5: Data-based methods: (a) Data augmentation and (b) Client selection

Using data-based methods to address client drift problems can help improve the convergence and performance of a global model. However, modifying the local data distributions can lead to information loss and change client behavior. This information may be valuable for personalizing the global model for each client. It's important to note that data-based methods do not preserve privacy and, therefore, violate privacy-preserving federated learning assumptions.

5.3.3 Architecture-based approach

Architecture-based PFL customizes the model design for each client to improve performance. This category includes two types of methods: parameter decoupling and knowledge distillation.

Parameter decoupling: The concept of parameter decoupling entails the separation of local private parameters from the global FL model parameters [Ari+19]. Private parameters are trained locally and are not shared with the FL server. Only the generalizable layers are shared (see Figure 5.6a). Parameter decoupling shares similarities with Federated Split Learning (FSL) (see Figure 5.6b) [Vep+18], where the deep neural network is divided layerwise between the server and clients. Unlike parameter decoupling, in FSL, the server model is not transferred to the client for training. Instead, only the client model's split layer weights are shared during forward propagation, and the split layer gradients are shared with the client during backpropagation. This gives FSL a privacy advantage over FL, as neither the server nor clients have full access to the global and local models [Tha+22].

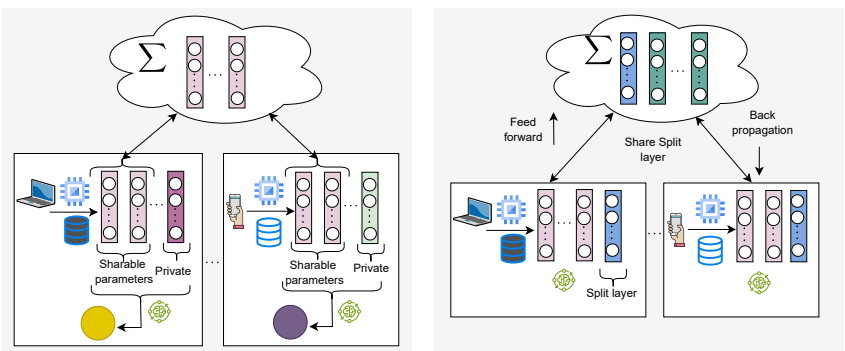


Figure 5.6: Difference between (a) Parameter decouple-based FL and (b) Federate Split learning

Knowledge distillation: It transfers knowledge from an ensemble of teacher models to a typically lightweight student model. Knowledge is usually represented as logit outputs. Knowledge Distillation (KD) can be implemented in four main ways: (1) distilling knowledge to each FL client to enhance personalization (see Figure 5.7a) [LW19], (2) distilling knowledge to the FL server to improve generalization (see Figure 5.7b)[Lin+20], (3) bidirectional distillation to both FL clients and the server (see Figure 5.7c) [HAA20a], and (4) distillation among clients (see Figure 5.7d) [BMB20]. Knowledge distillation in

federated learning provides flexibility for clients to customize model architectures. Moreover, it aims to address communication and computation challenges by lowering resource requirements.

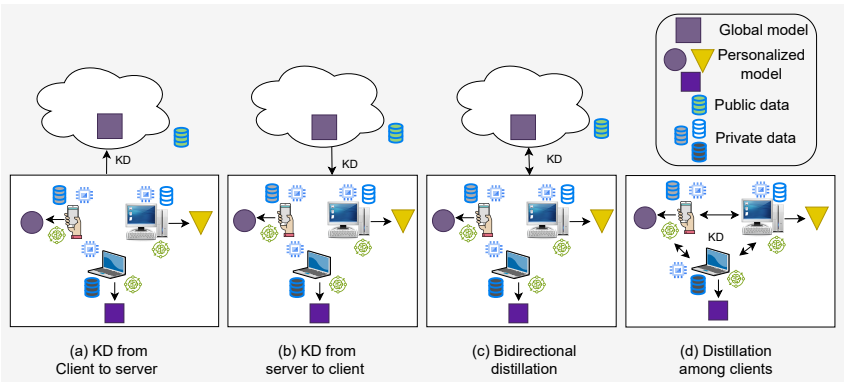


Figure 5.7: Different Knowledge Distillation(KD) based FL (a) KD from client to server, (b) KD from server to client, (c) Bideractional KD, and (d) KD among clients

5.4 Clustering-based approach

Clustered federated learning groups clients into clusters based on data or model similarity, where data distribution within each client is non-IID, and clients independently train their local models on their personal data. Clustered FL methods can be categorized into four types based on cluster formation criteria:

Distance between local and global model parameters: Clusters are formed based on the similarity between the local and global model parameters. If the distance between the local and global models is small, it means that the client is more useful for federated learning. A high distance indicates that the local model is diverging significantly from the global model. Instead of a single global model, clustered FL provides multiple global models or cluster models [Lon+23] to address heterogeneity. The clustering method could be partitioned based [Llo82], hierarchical [BFA20; Ngu+22], etc.

Clustering based on gradient information: Instead of using model parameters, the gradient updates of the local models are used to measure the similarity between clients [SMS20]. FedGroupProx [Dua+21] utilize the clients' optimization directions to determine similarity between clients and form clusters of similar clients.

Clustering based on the training loss: The server creates multiple global models instead of a single one and sends these models to all clients for local loss computation. Each client assesses multiple cluster models using their private data and chooses the cluster with the lowest loss. [Man+20; Gho+20].

Clustering based on the information about the data: This type of clustering relies on data. These methods necessitate sharing sensitive client information with the server for clustering. For instance, in [Hua+19], patients are grouped based on their electronic medical records.

Table 5.1 provides a comparative analysis of different PFL strategies :

Table 5.1: Summary of personalization strategies

Categories	Methods	Advantages	Disadvantages
Data-based	**Data augmentation** [Cha+02]	Can be built on the general FL training procedure	Privacy leakage can happen and a representative proxy dataset is required.
	Client selection [Wan+20]	Only modifies the client selection strategy of the general FL training procedure	Computation overhead for client selection.
Model-based	**Regularization** [Li 20]	Balance personalization and generalization trade-off	Performance of model depends on the hyper-parameters tuning
	Meta-learning [Jia+19]	Optimizes the global model for fast personalization	Computing second-order gradients (Hessian) is computation expensive
	Transfer learning [Che+20]	Improves personalization by reducing the domain discrepancy between the global and local models	Depends on the generalizability of the global model.
Architecture-based	**Parameter decoupling** [Ari+19]	Layer-wise flexibility in architecture design for each client	Difficult to determine the optimal privatization strategy
	Knowledge distillation [LW19]	Supports model heterogeneity	May require a representative proxy dataset. Moreover, clients need more resources to accommodate client models (teacher and student)
Clustering	**Distance between parameters** [Lon+23]	Cluster formation could be static or dynamic. It gives better performance on non-IID data	Computing clustering methods. Every iteration is computationally expensive and also depends on the scalability of the algorithm.
	Gradient information [Dua+21]	computation efficient	Privacy sensitive.
	clustering based on loss [Man+20]	Support model heterogeneity	Computationally expensive for resource constraint clients
	Clustering from data [Hua+19]	Can use the pattern from data to form clusters.	Privacy-sensitive

Chapter 6

Federated Learning Applications and Impacts

Federated Learning (FL) has proven its effectiveness across a wide range of industries, including healthcare, finance, IoT, cybersecurity, autonomous vehicles, and recommender systems. By enabling decentralized data collaboration while maintaining data privacy, FL offers significant advantages in these fields. For instance, in healthcare, FL enables collaborative model training without compromising patient privacy, and in finance, it enhances fraud detection and credit scoring by securely aggregating data from multiple organizations. In IoT, FL enhances the intelligence of edge devices. Additionally, FL has transformative potential in autonomous vehicles and personalized recommender systems. Beyond these established uses, we introduced two innovative applications from socio-cognitive sectors where FL-based solutions have shown promising outcomes.

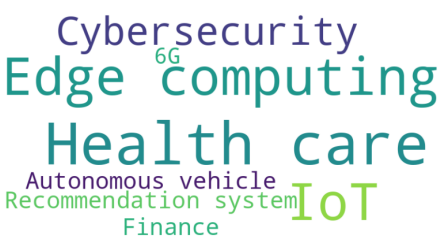


Figure 6.1: Word cloud for diverse FL applications

Our analysis of the various applications of FL published between 2020 and 2024, presented with a word cloud in Figure 6.1, highlights healthcare as the most explored application till now in FL, followed by edge computing, IoT, and cybersecurity. Furthermore, researchers are exploring the application of FL in autonomous vehicles, recommender systems, and finance, showcasing the wide-

reaching impact and versatility of FL. This chapter begins with a discussion of these general application areas where FL has demonstrated substantial benefits before moving into applications in socio-cognitive sectors.

6.1 Federated Learning in Healthcare

In the healthcare domain, patient data is highly sensitive, and hospitals generally do not share private patient information with third parties. This limited data availability poses challenges in developing accurate and efficient intelligent healthcare services. In this scenario, FL-based solutions have proven effective by ensuring that patients' private information remains undisclosed [Wen+23]. There are several applications in health care where FL is useful, such as medical imagery analysis, learning from Electronic Health Records (EHR), etc.

In medical imagery analysis, FL is employed to train models for segmenting brain Magnetic Resonance Imaging (MRI) data to identify brain tumors [Roy+19; She+20]. Additionally, FL models trained on chest X-rays have proven useful in predicting lung diseases, such as pneumonia [Ban+20], COVID-19 [Fek+21; Li+24; MS23; Gua+24], etc. Recently, FL models have also been used to diagnose conditions like Alzheimer's disease [Lu+19] and autism spectrum disorder [Li+20] through neurological MRI data.

In EHR, FL models are useful in several statistical analyses from the data such as Intensive Care Unit(ICU) mortality prediction [SSC19; Vai+21; Hua+20a], preterm birth prediction [Bou+19], etc. Moreover, FL is useful in building models that can help in disease diagnosis [Gra+20; Tul+20] and provide supports for personalized clinical decisions [Xue+21].

6.2 Federated Learning in Finance

Financial sectors, such as banking, insurance, etc., have observed criminal activities [Zhe+22]. In recent years, fraudulent transactions have become a significant problem for financial organizations worldwide. Consequently, the need for advanced fraud detection systems to protect assets and maintain customer trust is crucial in the financial sector. Furthermore, due to data privacy laws that all financial organizations must adhere to, sharing customer data to create a higher-performing centralized model is challenging [PE16; Con99; Cal18]. In fraudulent transaction detection in banking [ASP24], credit risk management [Xu+23b; Jov+24], loan risk assessment [Wan+23a], FL shows its supremacy in performance.

6.3 Edge Computing

Edge computing extends cloud computing services closer to data sources, allowing deep learning applications to run where the data is generated. Typically,

model inference is performed at the edge, while training data is often sent to third-party locations (cloud) to train centralized models. This approach requires high bandwidth and may pose privacy risks. Federated learning addresses these issues by enabling collaborative learning at the edge. The integration of FL with edge computing provides significant benefits such as improved privacy, lower latency, and better bandwidth utilization [Xia+21]. Employing FL for computation offloading and content caching [Cui+20] decisions proves to be beneficial [AHS22]. A dynamic cache allocation system based on FL allows edge nodes to learn locally and exchange knowledge collaboratively, optimizing resource allocations with minimal communication overhead [CP20]. The integration of FL and edge computing is being explored in various domains such as smart healthcare [Xu+21a], smart home [AGM19], surveillance [Yan+21], etc.

6.4 IoT and Cybersecurity

The decentralized approach of FL enables Internet of Things (IoT) devices to train local models and transmit updates to a central aggregator, thereby eradicating the necessity for centralized data processing. FL also provides services such as localization and attack detection, which enhance security and reduce network costs. FL transforms current systems and creates new opportunities for applications such as indoor navigation and mobile target tracking by leveraging the computational capabilities of distributed devices to facilitate efficient and privacy-aware IoT services.

Federated cybersecurity is an approach to enhance the safety and efficiency of the IoT. It is designed to detect security threats, implement countermeasures, and counter-spreading threats across IoT networks. It achieves cybersecurity objectives through its privacy-aware capabilities, which play a crucial role in securing the inherently vulnerable IoT environment. By leveraging FL, federated cybersecurity can provide robustness without compromising client privacy, thereby addressing the unique challenges posed by IoT devices and networks [GR22]. Many cybersecurity methods rely on federated learning to protect IoT networks. However, using a single global model from one service provider can cause problems due to data differences, making it less effective against cyberattacks. To address this, a clustered approach [Sáe+23] or a personalized federated learning approach [TSM24] could be advantageous. This would involve multiple global models and finely tuned local models managed by different service providers. In the field of cybersecurity, FL research can be defined in two ways. The first is federated learning for cybersecurity [GR22], where federated learning methods are developed to enhance privacy in cybersecurity. The second way is the consideration of cybersecurity for federated learning, where the FL system itself could become a target for various potential cyberattacks [Bla+17].

6.5 Autonomous Vehicles

Automated Vehicles (AVs) play a critical role in the advancement of intelligent transportation systems. Each AV generates a substantial amount of sensitive raw data, estimated at around 300 TB per vehicle annually [Blo24], which poses significant privacy and security risks if shared with third-party cloud services. FL addresses this challenge by employing collaborative learning, sharing only model updates instead of raw data. Additionally, FL reduces communication overhead by transmitting much smaller model updates instead of large datasets, making it feasible even in limited bandwidth scenarios. Collaborative learning in FL allows AVs to benefit from knowledge derived from diverse driving environments, thus improving the robustness and generalization of the global model. The decentralized framework of FL also enables direct vehicle-to-vehicle communication, enhancing scalability and ensuring that the model does not rely on a single point of failure. Furthermore, distributed computation in FL enhances resource efficiency, balances computational load, and accelerates training. Additionally, FL enables adaptive and continuous learning (Continual Federated Learning [Yua+23]), allowing AVs to learn from new data streams over time. These advantages make FL a promising tool for advancing machine learning in autonomous vehicles [Che+23].

6.6 Recommender Systems

Traditional recommender systems aggregate and analyze vast amounts of client data in a centralized server that includes the client's purchase history, browsing behavior, and personal preferences. clients are increasingly concerned about the potential misuse or mishandling of their sensitive information and the risks associated with data breaches. Motivated by privacy issues, federated learning is applied in recommender systems. The goal is to improve recommendation accuracy in a way that preserves privacy. The decentralized nature of FL helps maintain data privacy while still enabling the recommender systems to learn from diverse client behaviors across different devices, leading to more accurate and personalized recommendations. Moreover, FL addresses the personalized recommendations to the clients [Chr+24].

Apart from these common applications, we have identified two applications in the socio-cognitive areas where federated learning is beneficial.

6.7 Federate Learning for Predicting Epesodic Event Memorability

Episodic event memorability refers to how well an individual recalls an event. It is a fundamental part of human cognition and intelligence, playing a significant role in mental health and executive functioning [MJS19]. The memorability of

an event can be influenced by various factors, including the context in which the event occurs and the intrinsic visual attributes of image cues associated with the event [Iso+11]. Understanding episodic event memorability is crucial because enhancing episodic memory through cognitive training, such as regular photo review, can improve mental health and cognitive functioning. It helps in reactivating certain memory traces and maintaining high-level event memory [SOS13]. It is possible to implement lifelog-based memory interventions based on current technology. A lifelogger is someone who attaches wearable cameras or other recording devices to their body to continuously capture and record daily activities and events, creating a comprehensive log of their life experiences. However, designing effective cognitive intervention programs requires a clear understanding of what factors influence event memory, such as identifying which events have a higher training value and selecting effective photo cues [Xu+21b]. Federated learning is essential in predicting event memorability for several reasons,

Personalized model for each client: The memorability of an event is largely subjective, as it varies among individuals due to differences in their ability to remember events. Various personal attributes, such as age, race, ethnicity, and gender, contribute to this variability. Therefore, the presence of diverse data makes it difficult for a single global model to capture the memorability of events accurately. Hence, it is necessary to develop personalized models for individual clients.

Reducing self-rating effort: Although a siloed model may protect the privacy of clients by avoiding the transfer of knowledge through federated learning, it is not feasible for all individuals. In order to achieve satisfactory model performance, each client would be required to annotate a significant quantity of images. However, this task is not feasible for elderly or physically disabled individuals.

Preserve privacy: Information about the memorability of events can provide insights into a client's cognitive abilities, enabling the detection of conditions like Alzheimer's disease. As a result, this data is extremely private and cannot be disclosed to anyone else. Federated learning utilizes a privacy-preserving framework in which individual clients retain data and not exchange it among themselves. The secure channel guarantees that only model updates are transmitted to a trusted server, allowing clients to maintain complete control over their data.

In this thesis, we have developed FL-based predictive event memorability models utilizing the R3 dataset [Xu+21b] and performed several qualitative and quantitative analyses has been performed to predict event memorability. The characteristics of this dataset are given as follows.

6.7.1 Event memorability from R3 dataset

The R3 dataset [Xu+21b] is useful for studying event memorability in real-world contexts, with a focus on determining the factors that influence how events are remembered. The dataset is a comprehensive collection of data on visual semantics, event contexts, and memory outcomes that was obtained from a detailed and controlled experiment that involved elderly lifeloggers. The R3 dataset contains data from 47 elderly lifeloggers, of whom 37 were female. The mean age of the subjects is 62 years and eight months, and the standard deviation is six years and four months. The memory level and type of memory (remember, know, and new) were assessed, resulting in a score that ranged from 9 (strong recollection) to 0 (no memory). Additional information regarding the dataset is available in [Xu+21b].

We considered the annotations from 40 lifeloggers for FL; in total, 10,600 valid photo cues with associated memory scores were considered. In Table 6.1, we provide the number of images annotated by each lifelogger.

Table 6.1: Sample distribution across lifeloggers

Lifelogger ID	Sample Size	Lifelogger ID	Sample Size	Lifelogger ID	Sample Size	Lifelogger ID	Sample Size	Lifelogger ID	Sample Size
16	277	27	267	35	282	44	274	53	247
17	270	28	277	36	282	45	247	54	261
18	276	29	277	37	273	46	271	55	274
19	282	30	272	38	268	47	252	56	256
22	271	31	275	39	264	48	261	57	238
23	272	32	275	41	264	49	276	60	190
25	275	33	272	42	229	51	278	61	181
26	182	34	272	43	271	52	269	62	179

6.8 Federated Learning for Personal Image Advisor

With the rise of social media and cloud technology, many individuals store their private documents on various cloud platforms, often without considering the potential sensitivity of the information. While these services offer convenience, there is an increasing demand for data protection and privacy awareness. Many assume their documents are secure without fully understanding the security protocols. As a result, clients must be mindful of the nature of the information they upload to cloud services. An automated privacy advisor tool can play a valuable role in helping clients assess the sensitivity of their private documents before uploading them, allowing for more informed decisions about sharing. To support such tools, deep neural network models have been developed using public image datasets [OSF17; CKS21]. However, privacy sensitivity is a highly subjective matter, as perceptions and preferences regarding privacy risks can vary significantly between individuals, cultures, and even according to the privacy laws in different regions [Zho+17; JJS08; WNC11; Hen+12]. For in-

stance, some clients may view certain attributes in an image, such as a pet, as highly private, while others may not [Xu+24]. Therefore, personalized models that incorporate fine-grained annotations based on a client's perception of privacy risk for specific attributes or areas of interest are essential for accurately reflecting individual preferences.

DIPA2 [Xu+24] addressed personalized, attribute-specific privacy preference modeling by obtaining the annotations of privacy sensitivity scores for specific regions/attributes in images. They investigated the effect of personality-based and cross-cultural variations in privacy preferences when developing models for image privacy in DIPA2. Individuals from Japan and the UK provided their privacy annotations for the same set of images, and they found location-specific differences. Furthermore, they obtained the personality information of the annotators (clients) via a personality questionnaire and found that traits such as 'openness' and 'neuroticism' influence the sensitivity score provided by clients. Using this data, they developed their computational model, where the annotators' personality information, demographics, location, etc., was combined with the raw image features to predict the sensitivity of a given region of interest. A brief description of the DIPA2 dataset is as follows.

DIPA2: The DIPA2 dataset provides images with privacy sensitivity scores, focusing on client-perceive privacy and security from a cross-cultural perspective. DIPA2 provides detailed object-level annotations for 1,304 images, encompassing 5,897 annotations that describe perceived privacy risks for 3,347 objects. Each annotation includes four critical privacy-related metrics explained as follows.

- **Information type:** It specifies the type of information that might be revealed if the image is shared.
- **Informativeness:** Using a 7-point Likert scale, this metric assesses the perceived severity of potential privacy breaches, ranging from -3 (strongly disagree) to 3 (strongly agree). This helps measure the informativeness or threat level concerning privacy.
- **Sharing scope as a photo owner:** This metric identifies the groups with whom annotators would be willing to share the image if they were the owner.
- **Sharing scope by others:** This metric requires annotators to specify the groups they would allow to repost the image, using the same options as the sharing scope as a photo owner.

DIPA2 incorporated cultural differences by having annotations from Japan and the UK, with 300 participants from each country. Each image is annotated by two annotators from each country. This dual-perspective approach helps to understand how cultural backgrounds influence privacy concerns. DIPA2

collects extensive demographic information and personality traits of its annotators. Participants provide their age, gender, and nationality, and the *Big5* personality questionnaire, which measures the five personality dimensions: extraversion, agreeableness, conscientiousness, neuroticism, and openness. The details of the DIPA2 dataset are in [Xu+24].

The motivation to develop a personal image advisor using federated learning is because the centralized privacy models developed in [Xu+23a; Xu+24] assume the entire set of image annotations and the client's personality information, etc., are provided by all the clients are available in a centralized location. However, this may not be feasible if the clients are unwilling to share their private images and personality information for model development. Furthermore, the centralized approach suffers from conflicting privacy labels from clients [Suc+17] from different cultures and personality types. Therefore, each client should have a privacy model developed using their data. Still, they may not be able to annotate a sufficiently large dataset on their own to train a model with reliable results. Hence, clients should collaborate with other clients on model development without sharing the private raw images.

Chapter 7

Summary of Contributions

7.1 Paper I

Sourasekhar Banerjee, Erik Elmroth, and Monowar Bhuyan. Fed-FiS: A Novel Information-Theoretic Federated Feature Selection for Learning Stability. *Proceedings of the* 28^{th} *International Conference on Neural Information Processing (ICONIP), Springer Cham., Communications in Computer and Information Science, Vol. 1516, pp. 480–487, 2021.*

7.1.1 Contributions

In Paper I [BEB21], we focused on addressing research objective **RO1.** We investigated the challenges of feature selection in federated settings with heterogeneous data. Our proposed approach, Fed-FiS, effectively handles data heterogeneity in feature selection for federated settings. Fed-FiS comprises two main components: first, each client measures Feature Class Mutual Information (FCMI) or relevance and Feature Feature Mutual Information (FFMI) or redundancy scores for each feature and uses clustering to select local features; second, the selected features are sent to the server, where we propose a score function to generate global ranks for each feature. According to the ranks, the features are selected for FL.

Author's Contributions

Sourasekhar Banerjee identified the problem in collaboration with Monowar Bhuyan and Erik Elmroth. Sourasekhar Banerjee was primarily responsible for the problem formulation, algorithm design, implementation [1], and analysis. Monowar Bhuyan and Erik Elmroth provided critical feedback at each stage of experimentation and during the writing and finalizing the paper.

[1] Paper I and III have same repository `https://github.com/DevBhuyan/Horz-FL.git`

7.2 Paper II

Sourasekhar Banerjee, Xuan-Son Vu, and Monowar Bhuyan. Optimized and Adaptive Federated Learning for Straggler-Resilient Device Selection. *Proceedings of the International Joint Conference on Neural Networks (IJCNN), IEEE, pp. 1–9, 2022.*

7.2.1 Contributions

In Paper II [BVB22], we focused on identifying and mitigating the impact of stragglers in Federated Learning. Stragglers, which are low-performing devices, tend to delay the FL process and hinder learning efficiency. To address this, we developed Fed-MOODS, a method based on multi-objective optimization that ranks devices according to their processing power, memory, and bandwidth. We then proposed an adaptive device selection algorithm that prioritizes faster devices in the initial stages and progressively includes slower devices to enhance model accuracy over time.

Author's Contributions

Sourasekhar Banerjee, in collaboration with Monowar Bhuyan and Xuan-Son VU, identified the problem. Sourasekhar Banerjee was primarily responsible for problem formulation, algorithm design, implementation[2], and analysis. Xuan-Son Vu and Monowar Bhuyan provided critical feedback at each stage of experimentation and during the writing and finalizing of the paper.

7.3 Paper III

Sourasekhar Banerjee, Devvjiit Bhuyan, Erik Elmroth, and Monowar Bhuyan. Cost-Efficient Feature Selection for Horizontal Federated Learning. *IEEE Transactions on Artificial Intelligence (TAI)*, IEEE, doi: `10.1109/TAI.2024.3436664`, 2024.

7.3.1 Contributions

Paper III [Ban+24a] is a continuation of paper-I, where we studied research objective **RO1** for horizontal federated learning only. We introduced Fed-MOFS, a feature selection strategy based on multi-objective optimization. In this strategy, we assess features considering their relevance and redundancy. This process consists of two phases: local feature selection and global feature selection. The local feature selection phase is similar to Fed-FiS. For the global feature selection phase, Fed-MOFS utilizes multi-objective optimization to maximize relevance and minimize redundancy. Following this, all locally selected features

[2] `https://github.com/sourasb05/straggler_FL.git`

are ranked accordingly. We compared the performance of Fed-MOFS with conventional and federated feature selection methods. Moreover, we evaluated the scalability, stability, and effectiveness of both Fed-FiS and Fed-MOFS across various datasets for classification and regression problems. We also examined the effects of feature selection on model convergence and explored how data heterogeneity impacts feature selection.

Author's Contributions

Sourasekhar Banerjee identified the problem along with Monowar Bhuyan and Erik Elmroth. Sourasekhar Banerjee took the lead in formulating the problem, designing the algorithm, implementing it, and conducting the analysis. Devvjiit Bhuyan contributed to the partial implementation of the algorithm. Monowar Bhuyan and Erik Elmroth provided essential feedback during algorithm design, experimentation, paper writing, and finalization.

7.4 Paper IV

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

7.4.1 Contributions

In Paper IV [Ban+24b], we focused on the research objective **RO3**. We studied the data heterogeneity issue in multi-tier federated learning environments. The main challenge of personalized federated learning is to effectively capture the statistical heterogeneity characteristics of data through a communication-efficient way and to achieve customized performance for participating devices. We proposed PerMFL, a personalized federated learning algorithm for a multi-tier FL setup, to address this challenge. PerMFL produces optimized and personalized local models when clusters are known. We provided the theoretical guarantees of PerMFL for both convex and non-convex scenarios. PerMFL provides linear convergence rates for smooth, strongly convex problems and sub-linear convergence rates for smooth, non-convex problems. Empirically, we compared the performance and convergence of PerMFL with the state-of-the-art in multiple datasets. Moreover, we performed the ablation study of PerMFL on different hyperparameter settings, different types of team formation, and partial client participation. Also, we evaluated the effect of team iterations on the convergence of PerMFL.

Author's Contributions

Sourasekhar Banerjee, along with Ali Dadras, Alp Yurtsever, and Monowar Bhuyan, identified the problem and designed the algorithm. Sourasekhar Banerjee led the implementation [3], and empirical analysis. Ali Dadras handled the theoretical analysis part of the algorithm. Throughout the process, Alp Yurtsever and Monowar Bhuyan provided feedback on the empirical and theoretical analyses as well as on the writing and finalization of the paper.

7.5 Paper V

Sourasekhar Banerjee, Debaditya Roy, Vigneshwaran Subbaraju, and Monowar Bhuyan. Predicting Event Memorability using Personalized Federated Learning. *Submitted for publication*, 2024.

7.5.1 Contributions

In Paper V [Ban+24c], we focused on the research objectives **RO3** and **RO4**. Lifelog images are very helpful for people to remember past events. Predicting the memorability of a lifelog image for triggering memory recall in a person is crucial in cognitive interventions. Traditional methods for predicting event memorability typically followed a centralized training approach, requiring multiple users to share their lifelog images. However, because lifelog images are often very personal, sharing them may violate individual privacy. An alternative approach involved using a personal model trained on a lifelogger's own data, which preserved privacy but required the lifelogger to provide a large number of self-rated images to develop an effective model. This placed a significant burden on the lifelogger.

We proposed a clustered personalized federated learning method called FedMEM to address these challenges. This approach avoided sharing raw images while enabling collaborative learning through model sharing. We introduced two clustering methods within this approach. The first was Model Similarity-based Clustering (MSC), where clusters were formed based on the similarity between a lifelogger's local, global, and cluster models. After each global iteration, clusters were updated, allowing lifeloggers to switch to a different cluster if they found a cluster model that better matched their local model. The second method was Memory Score Distribution-based Clustering (MSDC), where clusters were formed based on the distribution of memory scores. In this approach, clusters were established at the beginning of the FL process, and lifeloggers remained in the same cluster throughout.

We conducted detailed evaluations of FedMEM combined with MSC and MSDC against the state-of-the-art FL algorithms using the event memorability dataset [Xu+21b]. Additionally, we performed a study to assess how FedMEM

[3] `https://github.com/sourasb05/PerMFL_1.git`

with MSC and MSDC performed with varying levels of lifelogger participation, data contributions, and convergence. Our analysis also examined the impact of FedMEM on individual lifeloggers.

Author's Contributions

Sourasekhar Banerjee identified the problem in collaboration with Vigneshwaran Subbaraju, Debaditya Roy, and Monowar Bhuyan. Sourasekhar Banerjee took the lead in problem formulation, algorithm design, implementation[4], and analysis. Vigneshwaran Subbaraju, Debaditya Roy, and Monowar Bhuyan provided feedback throughout the problem formulation, algorithm design, and analysis stages and assisted in writing and finalizing the paper.

7.6 Paper VI

Sourasekhar Banerjee, Vengateswaran Subramaniam, Debaditya Roy, Vigneshwaran Subbaraju and Monowar Bhuyan. The Case for Federated Learning in Developing Personalized Image Privacy Advisor. *Submitted for publication*, 2024.

7.6.1 Contributions

In Paper VI [Ban+24d], we focused on research objectives **RO3 and RO4**. Images often contain privacy-sensitive information, and sharing such personal images across different platforms can inadvertently expose this sensitive data. An automated image privacy advisor application can help reduce this risk by notifying users about the presence of privacy-sensitive information in their images. However, image privacy is highly subjective and depends on how users annotate their images. As a result, different users may assign varying privacy scores to the same image, leading to inconsistencies across users. A global model struggles to address this issue of heterogeneity. Additionally, users often have limited data, making it challenging to train personalized models for them.

To tackle the challenges of data heterogeneity and scarcity, we developed two daisy-chaining-enabled clustered personalized federated learning algorithms: Dynamic-Clustered-FedDC and Apriori-Clustered-FedDC. Alongside these algorithms, we introduced a lightweight image privacy model called Personality-Image attribute-Object network (PIONet), which is 20 times lighter than the baseline model [Xu+24]. We tested the performance of the PIONet trained on Dynamic-Clustered-FedDC and Apriori-Clustered-FedDC using an image privacy dataset [Xu+24] and compared it with state-of-the-art methods. To demonstrate the robustness of the algorithms against partial user participation, we conducted an ablation study on Dynamic-Clustered-FedDC and Apriori-Clustered-FedDC, examining different fractions of user involvement.

[4] `https://github.com/sourasb05/FedMEM.git`

Author's Contributions

Sourasekhar Banerjee identified the problem in collaboration with Vengateswaran Subramaniam, Vigneshwaran Subbaraju, Debaditya Roy, and Monowar Bhuyan. Sourasekhar Banerjee took the lead in problem formulation, algorithm design, implementation[5] of algorithms and analysis. Vengateswaran Subramaniam designed the PIONet. Vigneshwaran Subbaraju, Debaditya Roy, and Monowar Bhuyan provided feedback throughout the problem formulation, algorithm design, and analysis stages and assisted in writing and finalizing the paper.

[5] `https://github.com/sourasb05/Fed-DIPA2.git`

Chapter 8

Future Research Directions

This thesis addressed the challenges of statistical and system heterogeneity in federated learning. We focused on developing algorithms for horizontal federated learning where models are homogeneous, and the dataset remains static across clients. Despite the contributions made in this thesis, federated learning remains an active area of research with several critical open directions yet to be explored. This chapter provides a brief overview of several promising research directions of FL research.

8.1 Model Heterogeneity in Federated Learning

Due to varying design criteria and hardware capabilities [HAA20b; Wu+19], clients need to customize models. That eliminates the model homogeneity assumptions across clients and introduces a new challenge known as model heterogeneity [HYD22]. In the context of homogeneous federated learning, the methods are developed assuming that the server shares the initial model with the clients. The clients then train on their private data and share the parameters or gradients with the server for aggregation. This assumption cannot work on heterogeneous models. To address the challenge of model heterogeneity, model or knowledge distillation-based approaches were proposed [LW19; SL20], in which knowledge transfer occurs through labeled data. However, obtaining labeled data requires the server to collect data with distributions similar to private data, which involves costly human efforts and demands special domain expertise. Model sharing is another approach suggested in [Lia+20; Mat+22], but it increases computational cost and requires additional model structure on the client side. Moreover, simultaneously considering data and model heterogeneity in federated learning is a promising dimension for future research.

8.2 Federated Continual Learning

In federated learning, one challenge is the assumption that all data and classes are known in advance and will remain constant indefinitely. In reality, the entire dataset is not available at once but is received continuously in the form of task streams. Traditional federated learning can be seen as a two-step process: global or collaborative update and local update [McM+17a; Yan+19]. In the collaborative updating phase, clients learn from each other, and in the local update phase, the model is optimized on private data. However, traditional federated learning faces the issue called temporal catastrophic forgetting (Temporal-CF) when a local model is trained on new tasks. Temporal-CF is a significant challenge in continual learning [Die+19] that involves altering essential parameters in a single model while learning sequential tasks, resulting in degraded performance in previous tasks [MC89]. Another type of catastrophic forgetting discussed in [Yan+24] is known as spatial catastrophic forgetting (spatial-CF). This occurs during the aggregation of different local models. For example, in Federated Averaging (FedAvg), averaging the parameters of local models can result in overwriting critical parameters needed for specific local tasks. Consequently, the aggregated global model may underperform compared to the local models when evaluated on the local test sets [Gho+19; Luo+22; ZHZ21]. Federated Continual Learning (FCL) breaks the static assumption of federated learning and allows learning from dynamic task sequences. FCL suffers from Spacio-Temporal catastrophic forgetting. On the client side, users must address Temporal-CF when learning new tasks. On the server side, the server needs to overcome SpatialCF when aggregating different local models. Therefore, local and global models should be capable of continually fusing knowledge, as federated and continual learning share the common challenge of balancing knowledge from different data distributions [HYD22].

Knowledge can be expressed in various forms, including data, models, and outputs. From the literature [Yan+24], knowledge fusion can happen in seven ways, such as rehearsal [Luo+23; Wan+23b], clustering [Cas+22; Ton+21], sharing all parameters and regularization [Wan+23b; Zha+23d], parameter/layer isolation [Hu+22; Ji+19], dynamic architecture [MTF22; Ven+22], prototype [Hu+22] and knowledge distillation [HYD22; WL22]. Knowledge fusion-based federated continual learning holds significant potential and has not yet been thoroughly explored, making it a promising direction for future research.

8.3 Vertical Federated Learning

In this thesis, we focused on Horizontal Federated Learning (HFL), where all clients share a similar feature space but differ only in samples. Vertical Federated Learning (VFL) involves multiple clients with different feature spaces. Due to their difference in data partitions, HFL and VFL adopt very different training protocols. Unlike HFL, VFL keeps its data and model local but ex-

changes intermediate computed results. After training, each client in the VFL possesses a separate local model. During inference, each client in HFL applies the global model separately, while clients in VFL must collaborate to make inferences. In recent years, there has been an increasing demand for VFL in industries [Ou+20; Zha+24; Liu+21; Li+23b]. This is because clients from various industrial segments, such as banks and retailers, are more likely to collaborate than compete. Various VFL methods have been introduced recently to improve communication efficiency and model effectiveness. VFL is widely used in recommender systems, finance, healthcare, wireless networks, and smart grids. However, VFL still has significant challenges, such as interoperability and trustworthiness. The lack of interoperability of existing frameworks makes cross-platform collaboration difficult [Liu+24]. The VFL framework should have privacy, efficiency, fairness, explainability, and robustness. Protecting data during transit with well-defined privacy parameters is also crucial. The trade-off between privacy, efficiency, and fairness remains a central focus for future studies [Zha+23a] in VFL.

Bibliography

[Ach+23] Josh Achiam et al. "GPT-4 Technical Report". In: *arXiv preprint arXiv:2303.08774* (2023).

[AF91] Stefan Aeberhard and M. Forina. *Wine*. UCI Machine Learning Repository. DOI: https://doi.org/10.24432/C5PC7J. 1991.

[AG20] Mohammad Mohammadi Amiri and Deniz Gündüz. "Federated learning over wireless fading channels". In: *IEEE Transactions on Wireless Communications* 19.5 (2020), pp. 3546–3557.

[AGM19] Ulrich Matchi Aıvodji, Sébastien Gambs, and Alexandre Martin. "IoTFLA: A Secured and Privacy-Preserving Smart Home Architecture Implementing Federated Learning". In: *Proceedings of the IEEE Security and Privacy Workshops (SPW)*. IEEE, 2019, pp. 175–180.

[AHS22] Haftay Gebreslasie Abreha, Mohammad Hayajneh, and Mohamed Adel Serhani. "Federated learning in edge computing: a systematic survey". In: *Sensors* 22.2 (2022), p. 450.

[al89] Sigillito V. et al. *Ionosphere*. UCI Machine Learning Repository. 1989. URL: https://doi.org/10.24432/C5W01B.

[al95] Wolberg William et al. *Breast Cancer Wisconsin (Diagnostic)*. UCI Machine Learning Repository. 1995. URL: https://doi.org/10.24432/C5DW2B.

[Amo+18] Dario Amodei et al. *AI and Compute*. https://openai.com/blog/ai-and-compute/. Retrieved from https://openai.com/blog/ai-and-compute/. 2018.

[Ant19] Taranveer Singh Anttal. *Smart Home Dataset with Weather Information*. Kaggle. 2019. URL: https://www.kaggle.com/datasets/taranvee/smart-home-dataset-with-weather-information.

[Are+24] Caridad Arroyo Arevalo et al. "Task-Agnostic Privacy-Preserving Representation Learning for Federated Learning against Attribute Inference Attacks". In: *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 38. 10. 2024, pp. 10909–10917.

[Ari+19] Manoj Ghuhan Arivazhagan et al. “Federated learning with personalization layers”. In: *arXiv preprint arXiv:1912.00818* (2019).

[ASP24] Tomisin Awosika, Raj Mani Shukla, and Bernardi Pranggono. “Transparency and privacy: the role of explainable AI and federated learning in financial fraud detection”. In: *IEEE Access* (2024).

[Ban+20] Sourasekhar Banerjee et al. “Multi-diseases classification from chest-x-ray: A federated deep learning approach”. In: *Proceedings of 33rd Australasian Joint Conference, AI 2020, Canberra, ACT, Australia, November 29–30, 2020, Proceedings 33*. Springer. 2020, pp. 3–15.

[Ban+24a] Sourasekhar Banerjee et al. “Cost-Efficient Feature Selection for Horizontal Federated Learning”. In: *IEEE Transactions on Artificial Intelligence* (2024). DOI: 10.1109/TAI.2024.3436664.

[Ban+24b] Sourasekhar Banerjee et al. “Personalized multi-tier federated learning”. In: *arXiv preprint arXiv:2407.14251* (2024).

[Ban+24c] Sourasekhar Banerjee et al. “Predicting Event Memorability using Personalized Federated Learning”. In: *submitted for publication* (2024).

[Ban+24d] Sourasekhar Banerjee et al. “The case for federated learning in developing a personalized image privacy advisor”. In: *submitted for publication* (2024).

[BD01] José Bins and Bruce A Draper. “Feature selection from huge feature sets”. In: *Proceedings of Eighth IEEE International Conference on Computer Vision. ICCV 2001*. Vol. 2. IEEE. 2001, pp. 159–165.

[BEB21] Sourasekhar Banerjee, Erik Elmroth, and Monowar Bhuyan. “FedFiS: A novel Information-Theoretic Federated Feature Selection for Learning Stability”. In: *Proceedings of International Conference on Neural Information Processing (ICONIP)*. Springer. 2021, pp. 480–487.

[Bel+23] Enrique Tomás Martínez Beltrán et al. “Decentralized federated learning: Fundamentals, state of the art, frameworks, trends, and challenges”. In: *IEEE Communications Surveys & Tutorials* (2023).

[BFA20] Christopher Briggs, Zhong Fan, and Peter Andras. “Federated learning with hierarchical clustering of local updates to improve training on non-IID data”. In: *Proceedings of International Joint Conference on Neural Networks (IJCNN)*. IEEE. 2020, pp. 1–9.

[BL97] Avrim L Blum and Pat Langley. “Selection of relevant features and examples in machine learning”. In: *Artificial intelligence* 97.1-2 (1997), pp. 245–271.

[Bla+17] Peva Blanchard et al. "Machine learning with adversaries: Byzantine tolerant gradient descent". In: *Advances in neural information processing systems* 30 (2017).

[Blo24] Tuxera Blog. *Autonomous Cars: 300 TB of Data Per Year*. Accessed: 2024-07-26. 2024. URL: https://www.tuxera.com/blog/autonomous-cars-300-tb-of-data-per-year/.

[BMB20] Ilai Bistritz, Ariana Mann, and Nicholas Bambos. "Distributed distillation for on-device learning". In: *Advances in Neural Information Processing Systems* 33 (2020), pp. 22593–22604.

[Bon+19] Keith Bonawitz et al. "Towards federated learning at scale: System design". In: *Proceedings of the machine learning and systems* 1 (2019), pp. 374–388.

[Bou+19] Sabri Boughorbel et al. "Federated uncertainty-aware learning for distributed hospital ehr data". In: *arXiv preprint arXiv:1910.12191* (2019).

[Bro+20] Tom Brown et al. "Language models are few-shot learners". In: *Advances in neural information processing systems* 33 (2020), pp. 1877–1901.

[BVB22] Sourasekhar Banerjee, Xuan-Son Vu, and Monowar Bhuyan. "Optimized and Adaptive Federated Learning for Straggler-Resilient Device Selection". In: *Proceedings of International Joint Conference on Neural Networks (IJCNN)*. IEEE. 2022, pp. 1–9.

[Cal18] State of California. *California Consumer Privacy Act of 2018*. Accessed: 2024-06-23. 2018. URL: https://leginfo.legislature.ca.gov/faces/billTextClient.xhtml?bill_id=201720180AB375.

[Cas+22] Fabiola Espinoza Castellon et al. "Federated learning with incremental clustering for heterogeneous data". In: *2022 International Joint Conference on Neural Networks (IJCNN)*. IEEE. 2022, pp. 1–8.

[Cas+23] Timothy Castiglia et al. "LESS-VFL: Communication-efficient feature selection for vertical federated learning". In: *Proceedings of International Conference on Machine Learning*. PMLR. 2023, pp. 3757–3781.

[CCK21] Wei-Ning Chen, Christopher A Choquette-Choo, and Peter Kairouz. "Communication efficient federated learning with secure aggregation and differential privacy". In: *NeurIPS 2021 Workshop Privacy in Machine Learning*. 2021.

[Cha+02] Nitesh V Chawla et al. "SMOTE: synthetic minority over-sampling technique". In: *Journal of Artificial Intelligence Research* 16 (2002), pp. 321–357.

[Cha+20] Zheng Chai et al. “Tifl: A tier-based federated learning system”. In: *Proceedings of the 29th International Symposium on High-performance Parallel and Distributed computing*. 2020, pp. 125–136.

[Che+20] Yiqiang Chen et al. “Fedhealth: A federated transfer learning framework for wearable healthcare”. In: *IEEE Intelligent Systems* 35.4 (2020), pp. 83–93.

[Che+23] Vishnu Pandi Chellapandi et al. “Federated learning for connected and automated vehicles: A survey of existing approaches and challenges”. In: *IEEE Transactions on Intelligent Vehicles* (2023).

[Cho+20] Yae Jee Cho et al. “Bandit-based communication-efficient client selection strategies for federated learning”. In: *Proceedings of 54th Asilomar Conference on Signals, Systems, and Computers*. IEEE. 2020, pp. 1066–1069.

[Chr+24] Christos Chronis et al. “A survey on the use of Federated Learning in Privacy-Preserving Recommender Systems”. In: *IEEE Open Journal of the Computer Society* (2024).

[CHR20] Wenlin Chen, Samuel Horvath, and Peter Richtarik. “Optimal client sampling for federated learning”. In: *arXiv:2010.13723* (2020).

[CKS21] Zhang Chen, Thivya Kandappu, and Vigneshwaran Subbaraju. “PrivAttNet: predicting privacy risks in images using visual attention”. In: *Proceedings of 25th International Conference on Pattern Recognition (ICPR)*. IEEE. 2021, pp. 10327–10334.

[Coh17] Cohen, Gregory et al. “EMNIST: Extending MNIST to handwritten letters”. In: *Proceedings of International Joint Conference of Neural Networks (IJCNN)*. IEEE. 2017, pp. 2921–2926.

[Con99] United States Congress. *Gramm-Leach-Bliley Act*. Accessed: 2024-06-23. 1999. URL: `https://www.congress.gov/bill/106th-congress/senate-bill/900`.

[CP20] Shanti Chilukuri and Dirk Pesch. “Achieving optimal cache utility in constrained wireless networks through federated learning”. In: *Proceedings of 21st International Symposium on" A World of Wireless, Mobile and Multimedia Networks"(WoWMoM)*. IEEE. 2020, pp. 254–263.

[Cui+20] Laizhong Cui et al. “CREAT: Blockchain-assisted compression algorithm of federated learning for content caching in edge computing”. In: *IEEE Internet of Things Journal* 9.16 (2020), pp. 14151–14161.

[CWJ22] Yae Jee Cho, Jianyu Wang, and Gauri Joshi. "Towards understanding biased client selection in federated learning". In: *Proceedings of International Conference on Artificial Intelligence and Statistics*. PMLR. 2022, pp. 10351–10375.

[Dal+17] Andrea Dal Pozzolo et al. "Credit card fraud detection: a realistic modeling and a novel learning strategy". In: *IEEE transactions on neural networks and learning systems* 29.8 (2017), pp. 3784–3797.

[DDD90] National Institute of Diabetes, Digestive, and Kidney Diseases. *Diabetes Dataset*. Kaggle. 1990. URL: `https://www.kaggle.com/datasets/mathchi/diabetes-data-set`.

[Den+19] Xuelian Deng et al. "Feature selection for text classification: A review". In: *Multimedia Tools and Applications* 78.3 (2019), pp. 3797–3816.

[Die+19] Tom Diethe et al. "Continual learning in practice". In: *arXiv preprint arXiv:1903.05202* (2019).

[DP05] Chris Ding and Hanchuan Peng. "Minimum redundancy feature selection from microarray gene expression data". In: *Journal of Bioinformatics and Computational Biology* 3.02 (2005), pp. 185–205.

[Dra+20] Georgios Drainakis et al. "Federated vs. centralized machine learning under privacy-elastic users: A comparative analysis". In: *Proceedings of 19th International Symposium on Network Computing and Applications (NCA)*. IEEE. 2020, pp. 1–8.

[Dua+21] Moming Duan et al. "Fedgroup: Efficient federated learning via decomposed similarity-based clustering". In: *Proceedings of International Conference on Big Data and Cloud Computing (BdCloud)*. IEEE. 2021, pp. 228–237.

[FAL17] Chelsea Finn, Pieter Abbeel, and Sergey Levine. "Model-agnostic meta-learning for fast adaptation of deep networks". In: *Proceedings of International conference on machine learning*. PMLR. 2017, pp. 1126–1135.

[Fan+20] Minghong Fang et al. "Local model poisoning attacks to Byzantine-Robust federated learning". In: *Proceedings of 29th USENIX security symposium (USENIX Security)*. 2020, pp. 1605–1622.

[FB17] Valeria Fonti and Eduard Belitser. "Feature selection using lasso". In: *VU Amsterdam research paper in business analytics* 30 (2017), pp. 1–25.

[Fek+21] Ines Feki et al. "Federated learning for COVID-19 screening from Chest X-ray images". In: *Applied Soft Computing* 106 (2021), p. 107330.

[Fen22] Siwei Feng. "Vertical federated learning-based feature selection with non-overlapping sample utilization". In: *Expert Systems with Applications* 208 (2022), p. 118097.

[FMO20] Alireza Fallah, Aryan Mokhtari, and Asuman Ozdaglar. "Personalized federated learning: A meta-learning approach". In: *arXiv preprint arXiv:2002.07948* (2020).

[Fu+23a] Lei Fu et al. "Client selection in federated learning: Principles, challenges, and opportunities". In: *IEEE Internet of Things Journal* (2023). DOI: 10.1109/JIOT.2023.3299573.

[Fu+23b] Rui Fu et al. "FEAST: A communication-efficient federated feature selection framework for relational data". In: *Proceedings of the ACM on Management of Data* 1.1 (2023), pp. 1–28.

[Gho+19] Avishek Ghosh et al. "Robust federated learning in a heterogeneous environment". In: *arXiv preprint arXiv:1906.06629* (2019).

[Gho+20] Avishek Ghosh et al. "An efficient framework for clustered federated learning". In: *Advances in Neural Information Processing Systems* 33 (2020), pp. 19586–19597.

[Gol89] David E Goldberg. "Cenetic algorithms in search". In: *Optimization, Machine Learning* (1989).

[GR22] Bimal Ghimire and Danda B Rawat. "Recent advances on federated learning for cybersecurity and cybersecurity for federated learning for internet of things". In: *IEEE Internet of Things Journal* 9.11 (2022), pp. 8229–8249.

[Gra+20] Matei Grama et al. "Robust aggregation for adaptive privacy preserving federated learning in healthcare". In: *arXiv preprint arXiv:2009.08294* (2020).

[Gua+24] Hao Guan et al. "Federated learning for medical image analysis: A survey". In: *Pattern Recognition* (2024), p. 110424.

[Guy+02] Isabelle Guyon et al. "Gene selection for cancer classification using support vector machines". In: *Machine learning* 46 (2002), pp. 389–422.

[HAA20a] Chaoyang He, Murali Annavaram, and Salman Avestimehr. "Group knowledge transfer: Federated learning of large CNNs at the edge". In: *Advances in Neural Information Processing Systems* 33 (2020), pp. 14068–14080.

[HAA20b] Chaoyang He, Murali Annavaram, and Salman Avestimehr. "Towards non-IID and invisible data with FedNAS: Federated deep learning via neural architecture search". In: *arXiv:2004.08546* (2020).

[HBL24] Jorge Hermo, Verónica Bolón-Canedo, and Susana Ladra. "FedmRMR: A lossless federated feature selection method". In: *Information Sciences* 669 (2024), p. 120609.

[He+16] Kaiming He et al. "Deep residual learning for image recognition". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016, pp. 770–778.

[Hen+12] Xu Heng et al. "Effects of Individual Self-Protection, Industry Self-Regulation, and Government Regulation on Privacy Concerns: A Study of Location-Based Services". In: *Information Systems Research* 23.4 (2012), pp. 1342–1363.

[Hru+06] Eduardo R Hruschka et al. "Bayesian feature selection for clustering problems". In: *Journal of Information & Knowledge Management* 5.04 (2006), pp. 315–327.

[Hsi+17] Kevin Hsieh et al. "Gaia: Geo-Distributed machine learning approaching LAN speeds". In: *Proceedings of 14th USENIX Symposium on Networked Systems Design and Implementation (NSDI 17)*. 2017, pp. 629–647.

[Hu+22] Kai Hu et al. "A federated incremental learning algorithm based on dual attention mechanism". In: *Applied Sciences* 12.19 (2022), p. 10025.

[Hua+19] Li Huang et al. "Patient clustering improves efficiency of federated machine learning to predict mortality and hospital stay time using distributed electronic medical records". In: *Journal of biomedical informatics* 99 (2019), p. 103291.

[Hua+20a] Li Huang et al. "LoAdaBoost: Loss-based AdaBoost federated machine learning with reduced computational complexity on IID and non-IID intensive care data". In: *Plos one* 15.4 (2020), e0230706.

[Hua+20b] 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 (2020), pp. 1552–1564.

[Hua+21] Yangsibo Huang et al. "Evaluating gradient inversion attacks and defenses in federated learning". In: *Advances in neural information processing systems* 34 (2021), pp. 7232–7241.

[HYD22] Wenke Huang, Mang Ye, and Bo Du. "Learn from others and be yourself in heterogeneous federated learning". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2022, pp. 10143–10153.

[Iso+11] Phillip Isola et al. "Understanding the intrinsic memorability of images". In: *Advances in neural information processing systems* 24 (2011).

[Jeo+18] Eunjeong Jeong et al. “Communication-efficient on-device machine learning: Federated distillation and augmentation under non-iid private data”. In: *arXiv preprint arXiv:1811.11479* (2018).

[Ji+19] Shaoxiong Ji et al. “Learning private neural language modeling with attentive aggregation”. In: *Proceedings of International Joint Conference on Neural Networks (IJCNN)*. IEEE. 2019, pp. 1–8.

[Jia+19] Yihan Jiang et al. “Improving federated learning personalization via model agnostic meta learning”. In: *arXiv:1909.12488* (2019).

[Jia+22] Jiawei Jiang et al. “Vf-ps: How to select important participants in vertical federated learning, efficiently and securely?” In: *Advances in Neural Information Processing Systems* 35 (2022), pp. 2088–2101.

[JJS08] Iris A Junglas, Norman A Johnson, and Christiane Spitzmüller. “Personality traits and concern for privacy: an empirical study in the context of location-based services”. In: *European Journal of Information Systems* 17.4 (2008), pp. 387–402.

[Jov+24] Zorka Jovanovic et al. “Robust integration of blockchain and explainable federated learning for automated credit scoring”. In: *Computer Networks* 243 (2024), p. 110303.

[Kag20] Kaggle. *Vehicle Dataset*. Kaggle. 2020. URL: `https://www.kaggle.com/datasets/nehalbirla/vehicle-dataset-from-cardekho`.

[Kai+21] Peter Kairouz et al. “Advances and open problems in federated learning”. In: *Foundations and trends® in machine learning* 14.1–2 (2021), pp. 1–210.

[Kam+19] Michael Kamp et al. “Efficient decentralized deep learning by dynamic model averaging”. In: *Proceedings of Machine Learning and Knowledge Discovery in Databases: European Conference, ECML PKDD 2018, Dublin, Ireland, September 10–14, 2018, Proceedings, Part I 18*. Springer. 2019, pp. 393–409.

[Kan+22] Renuga Kanagavelu et al. “CE-Fed: Communication efficient multiparty computation enabled federated learning”. In: *Array* 15 (2022), p. 100207.

[Kar+20] Sai Praneeth Karimireddy et al. “Scaffold: Stochastic controlled averaging for federated learning”. In: *Proceedings of International conference on machine learning*. PMLR. 2020, pp. 5132–5143.

[KE95] James Kennedy and Russell Eberhart. “Particle swarm optimization”. In: *Proceedings of International Conference on Neural Networks (ICNN)*. Vol. 4. ieee. 1995, pp. 1942–1948.

[KH09] Alex Krizhevsky and Geoffrey Hinton. *Learning multiple layers of features from tiny images*. Technical Report TR-2009. University of Toronto, 2009. URL: https://www.cs.toronto.edu/~kriz/learning-features-2009-TR.pdf.

[Kha+19] Wazir Zada Khan et al. "Edge computing: A survey". In: *Future Generation Computer Systems* 97 (2019), pp. 219–235.

[KM+97] Miroslav Kubat, Stan Matwin, et al. "Addressing the curse of imbalanced training sets: one-sided selection". In: *Proceedings of International Conference of Machine Learning*. Vol. 97. 1. Citeseer. 1997, p. 179.

[Lai+21] Fan Lai et al. "Oort: Efficient federated learning via guided participant selection". In: *Proceedings of 15th USENIX Symposium on Operating Systems Design and Implementation (OSDI)*. 2021, pp. 19–35.

[Lan+94] Pat Langley et al. "Selection of relevant features in machine learning". In: *Proceedings of the AAAI Fall symposium on relevance*. Vol. 184. California. 1994, pp. 245–271.

[LeC+98] Yann LeCun et al. "Gradient-based learning applied to document recognition". In: *Proceedings of the IEEE* 86.11 (1998), pp. 2278–2324.

[LF08] Graham Lee and Oppacher Franz. *Hill-Valley*. UCI Machine Learning Repository. 2008. URL: https://doi.org/10.24432/C5JC8P.

[LHS21] Qinbin Li, Bingsheng He, and Dawn Song. "Model-contrastive federated learning". In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2021, pp. 10713–10722.

[Li 20] Li, Tian et al. "Federated optimization in heterogeneous networks". In: *Proceedings of Machine learning and systems* 2 (2020), pp. 429–450.

[Li+07] Jianping Li et al. "Feature selection via least squares support feature machine". In: *International Journal of Information Technology & Decision Making* 6.04 (2007), pp. 671–686.

[Li+19] Xiang Li et al. "On the convergence of fedavg on non-iid data". In: *arXiv preprint arXiv:1907.02189* (2019).

[Li+20] Xiaoxiao Li et al. "Multi-site fMRI analysis using privacy-preserving federated learning and domain adaptation: ABIDE results". In: *Medical Image Analysis* 65 (2020), p. 101765.

[Li+22] Qinbin Li et al. "Federated learning on non-iid data silos: An experimental study". In: *Proceedings of 38th international conference on Data Engineering (ICDE)*. IEEE. 2022, pp. 965–978.

[Li+23a] Anran Li et al. “FedSDG-FS: Efficient and secure feature selection for vertical federated learning”. In: *Proceedings of Conference on Computer Communications (INFOCOM)*. IEEE. 2023, pp. 1–10.

[Li+23b] Qinbin Li et al. “Fedtree: A federated learning system for trees”. In: *Proceedings of Machine Learning and Systems* 5 (2023).

[Li+24] Zheng Li et al. “Integrated CNN and Federated Learning for COVID-19 Detection on Chest X-Ray Images”. In: *IEEE/ACM Transactions on Computational Biology and Bioinformatics* 21.4 (2024), pp. 835–845. DOI: 10.1109/TCBB.2022.3184319.

[Lia+20] Paul Pu Liang et al. “Think locally, act globally: Federated learning with local and global representations”. In: *arXiv preprint arXiv:2001.01523* (2020).

[Lin+20] Tao Lin et al. “Ensemble distillation for robust model fusion in federated learning”. In: *Advances in neural information processing systems* 33 (2020), pp. 2351–2363.

[Lin+21] Frank Po-Chen Lin et al. “Semi-decentralized federated learning with cooperative D2D local model aggregations”. In: *IEEE Journal on Selected Areas in Communications* 39.12 (2021), pp. 3851–3869.

[Liu+21] Yang Liu et al. “Fate: An industrial grade platform for collaborative learning with data protection”. In: *Journal of Machine Learning Research* 22.226 (2021), pp. 1–6.

[Liu+23] Hongfu Liu et al. “Privacy-Encoded Federated Learning Against Gradient-Based Data Reconstruction Attacks”. In: *IEEE Transactions on Information Forensics and Security* (2023).

[Liu+24] Yang Liu et al. “Vertical federated learning: Concepts, advances, and challenges”. In: *IEEE Transactions on Knowledge and Data Engineering* (2024).

[Llo82] Stuart Lloyd. “Least squares quantization in PCM”. In: *IEEE transactions on information theory* 28.2 (1982), pp. 129–137.

[Lon+23] Guodong Long et al. “Multi-center federated learning: clients clustering for better personalization”. In: *World Wide Web* 26.1 (2023), pp. 481–500.

[Lu+19] Songtao Lu et al. “Learn electronic health records by fully decentralized federated learning”. In: *arXiv preprint arXiv:1912.01792* (2019).

[Luo+22] Bing Luo et al. “Tackling system and statistical heterogeneity for federated learning with adaptive client sampling”. In: *Conference on Computer Communications (INFOCOM)*. IEEE. 2022, pp. 1739–1748.

[Luo+23] Yaxin Luopan et al. “Fedknow: Federated continual learning with signature task knowledge integration at edge”. In: *Proceedings of 39th International Conference on Data Engineering (ICDE)*. IEEE. 2023, pp. 341–354.

[LW19] Daliang Li and Junpu Wang. “Fedmd: Heterogenous federated learning via model distillation”. In: *arXiv preprint arXiv:1910.03581* (2019).

[LWB19] Wang Luping, WANG Wei, and LI Bo. “CMFL: Mitigating communication overhead for federated learning”. In: *2019 IEEE 39th international conference on distributed computing systems (ICDCS)*. IEEE. 2019, pp. 954–964.

[Man+20] Yishay Mansour et al. “Three approaches for personalization with applications to federated learning”. In: *arXiv:2002.10619* (2020).

[Mas+22] Carlotta Masciocchi et al. “Federated Cox Proportional Hazards Model with multicentric privacy-preserving LASSO feature selection for survival analysis from the perspective of personalized medicine”. In: *Proceedings of 35th International Symposium on Computer-Based Medical Systems (CBMS)*. IEEE. 2022, pp. 25–31.

[Mat+22] Koji Matsuda et al. “Fedme: Federated learning via model exchange”. In: *Proceedings of the SIAM international conference on data mining (SDM)*. SIAM. 2022, pp. 459–467.

[MBA17] Laura Morán-Fernández, Verónica Bolón-Canedo, and Amparo Alonso-Betanzos. “Centralized vs. distributed feature selection methods based on data complexity measures”. In: *Knowledge-Based Systems* 117 (2017), pp. 27–45.

[MC89] Michael McCloskey and Neal J Cohen. “Catastrophic interference in connectionist networks: The sequential learning problem”. In: *Psychology of learning and motivation*. Vol. 24. Elsevier, 1989, pp. 109–165.

[McM+17a] Brendan McMahan et al. “Communication-efficient learning of deep networks from decentralized data”. In: *Artificial intelligence and statistics*. PMLR. 2017, pp. 1273–1282.

[McM+17b] H Brendan McMahan et al. “Learning differentially private recurrent language models”. In: *arXiv preprint arXiv:1710.06963* (2017).

[MHM19] Jed Mills, Jia Hu, and Geyong Min. “Communication-efficient federated learning for wireless edge intelligence in IoT”. In: *IEEE Internet of Things Journal* 7.7 (2019), pp. 5986–5994.

[MJS19] Kevin P Madore, Helen G Jing, and Daniel L Schacter. "Selective effects of specificity inductions on episodic details: evidence for an event construction account". In: *Memory* 27.2 (2019), pp. 250–260.

[MK24] Afsaneh Mahanipour and Hana Khamfroush. "FMLFS: A federated multi-label feature selection based on information theory in IoT environment". In: *arXiv preprint arXiv:2405.00524* (2024).

[Mou+23] Yongli Mou et al. "pfedv: Mitigating feature distribution skewness via personalized federated learning with variational distribution constraints". In: *Proceedings of Pacific-Asia Conference on Knowledge Discovery and Data Mining*. Springer. 2023, pp. 283–294.

[MS23] Aaisha Makkar and KC Santosh. "SecureFed: federated learning empowered medical imaging technique to analyze lung abnormalities in chest X-rays". In: *International Journal of Machine Learning and Cybernetics* 14.8 (2023), pp. 2659–2670.

[MTF22] Junki Mori, Isamu Teranishi, and Ryo Furukawa. "Continual horizontal federated learning for heterogeneous data". In: *Proceedings of International Joint Conference on Neural Networks (IJCNN)*. IEEE. 2022, pp. 1–8.

[Muñ+20] Sergio Muñoz-Romero et al. "Informative variable identifier: Expanding interpretability in feature selection". In: *Pattern Recognition* 98 (2020), p. 107077.

[Ngu+22] Minh NH Nguyen et al. "Self-organizing democratized learning: Toward large-scale distributed learning systems". In: *IEEE Transactions on Neural Networks and Learning Systems* (2022).

[Nie+10] Feiping Nie et al. "Efficient and robust feature selection via joint l2, 1-norms minimization". In: *Advances in Neural Information Processing Systems* 23 (2010).

[NY19] Takayuki Nishio and Ryo Yonetani. "Client selection for federated learning with heterogeneous resources in mobile edge". In: *Proceedings of IEEE International Conference on Communications (ICC)*. IEEE. 2019, pp. 1–7.

[OSF17] Tribhuvanesh Orekondy, Bernt Schiele, and Mario Fritz. "Towards a Visual Privacy Advisor: Understanding and Predicting Privacy Risks in Images". In: *Proceedings of International Conference on Computer Vision (ICCV)*. Oct. 29, 2017.

[Otc+22] Daniel Asante Otchere et al. "Application of gradient boosting regression model for the evaluation of feature selection techniques in improving reservoir characterisation predictions". In: *Journal of Petroleum Science and Engineering* 208 (2022), p. 109244.

[Ou+20] Wei Ou et al. “A homomorphic-encryption-based vertical federated learning scheme for rick management”. In: *Computer Science and Information Systems* 17.3 (2020), pp. 819–834.

[Par00] Parliament of Canada. *Personal Information Protection and Electronic Documents Act*. Statutes of Canada 2000, c 5. 2000. URL: https://laws-lois.justice.gc.ca/eng/acts/P-8.6/.

[Pas+19] Adam Paszke et al. “PyTorch: An Imperative Style, High-Performance Deep Learning Library”. In: *Advances in Neural Information Processing Systems*. Vol. 32. 2019. URL: https://papers.nips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf.

[PE16] European Parliament and Council of the European Union. *Regulation (EU) 2016/679 of the European Parliament and of the Council of 27 April 2016 on the protection of natural persons with regard to the processing of personal data and on the free movement of such data, and repealing Directive 95/46/EC (General Data Protection Regulation)*. Accessed: 2024-06-23. 2016. URL: https://eur-lex.europa.eu/eli/reg/2016/679/oj.

[Pfe+23] Kilian Pfeiffer et al. “Federated learning for computationally constrained heterogeneous devices: A survey”. In: *ACM Computing Surveys* 55.14s (2023), pp. 1–27.

[PLD05] Hanchuan Peng, Fuhui Long, and Chris Ding. “Feature selection based on mutual information criteria of max-dependency, max-relevance, and min-redundancy”. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)* 27.8 (2005), pp. 1226–1238.

[PNK94] Pavel Pudil, Jana Novovičová, and Josef Kittler. “Floating search methods in feature selection”. In: *Pattern recognition letters* 15.11 (1994), pp. 1119–1125.

[PY09] Sinno Jialin Pan and Qiang Yang. “A survey on transfer learning”. In: *IEEE Transactions on Knowledge and Data Engineering* 22.10 (2009), pp. 1345–1359.

[QK21] Yang Qin and Masaaki Kondo. “Federated learning-based network intrusion detection with a feature selection approach”. In: *Proceedings of International Conference on Electrical, Communication, and Computer Engineering (ICECCE)*. IEEE. 2021, pp. 1–6.

[Rat+19] Alexander Ratner et al. “Mlsys: The new frontier of machine learning systems”. In: *arXiv preprint arXiv:1904.03257* (2019).

[Rei+22] 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.

[RM94] Cole Ron and Fanty Mark. *ISOLET*. UCI Machine Learning Repository. 1994. URL: `https://doi.org/10.24432/C51G69`.

[Roy+19] Abhijit Guha Roy et al. “Braintorrent: A peer-to-peer environment for decentralized federated learning”. In: *arXiv preprint arXiv:1905.06731* (2019).

[RS04] Laura Elena Raileanu and Kilian Stoffel. “Theoretical comparison between the Gini Index and Information Gain criteria”. In: *Annals of Mathematics and Artificial Intelligence* 41 (2004), pp. 77–93.

[Sáe+23] Xabier Sáez-de-Cámara et al. “Clustered federated learning architecture for network anomaly detection in large scale heterogeneous IoT networks”. In: *Computers & Security* 131 (2023), p. 103299.

[San+18] Mark Sandler et al. “MobileNetV2: Inverted Residuals and Linear Bottlenecks”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2018, pp. 4510–4520.

[SCM20] Saúl Solorio-Fernández, J Ariel Carrasco-Ochoa, and José Fco Martínez-Trinidad. “A review of unsupervised feature selection methods”. In: *Artificial Intelligence Review* 53.2 (2020), pp. 907–948.

[She+20] 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.

[SL20] Lichao Sun and Lingjuan Lyu. “Federated model distillation with noise-free differential privacy”. In: *arXiv preprint arXiv:2009.05537* (2020).

[SL21] Suhail Mohmad Shah and Vincent KN Lau. “Model compression for communication efficient federated learning”. In: *IEEE Transactions on Neural Networks and Learning Systems* 34.9 (2021), pp. 5937–5951.

[SMS20] Felix Sattler, Klaus-Robert Müller, and Wojciech Samek. “Clustered federated learning: Model-agnostic distributed multitask optimization under privacy constraints”. In: *IEEE transactions on neural networks and learning systems* 32.8 (2020), pp. 3710–3722.

[SOS13] Peggy L St. Jacques, Christopher Olm, and Daniel L Schacter. "Neural mechanisms of reactivation-induced updating that enhance and distort memory". In: *Proceedings of the National Academy of Sciences* 110.49 (2013), pp. 19671–19678.

[SSC19] Pulkit Sharma, Farah E Shamout, and David A Clifton. "Preserving patient privacy while training a predictive model of in-hospital mortality". In: *arXiv preprint arXiv:1912.00354* (2019).

[Suc+17] Jose M Such et al. "Photo privacy conflicts in social media: A large-scale empirical study". In: *Proceedings of the Conference on Human factors in Computing Systems*. 2017, pp. 3821–3832.

[Sun+10] Hongbin Sun et al. "PGFB: A hybrid feature selection method based on mutual information". In: *Proceedings of Seventh International Conference on Fuzzy Systems and Knowledge Discovery*. Vol. 6. IEEE. 2010, pp. 2862–2871.

[Sur] Kaushik Suresh. *Customer Segmentation Classification*. Kaggle. URL: `https://www.kaggle.com/datasets/kaushiksuresh147/customer-segmentation`.

[SWR20] Victor Sanh, Thomas Wolf, and Alexander Rush. "Movement pruning: Adaptive sparsity by fine-tuning". In: *Advances in neural information processing systems* 33 (2020), pp. 20378–20389.

[Tan+22] Alysa Ziying Tan et al. "Towards personalized federated learning". In: *IEEE transactions on neural networks and learning systems* 34.12 (2022), pp. 9587–9603.

[Tav+09] Mahbod Tavallaee et al. "A detailed analysis of the KDD CUP 99 data set". In: *Proceedings of IEEE symposium on computational intelligence for security and defense applications*. IEEE. 2009, pp. 1–6.

[Tea+23] Gemini Team et al. "Gemini: a family of highly capable multimodal models". In: *arXiv preprint arXiv:2312.11805* (2023).

[Tha+22] Chandra Thapa et al. "Splitfed: When federated learning meets split learning". In: *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 36. 8. 2022, pp. 8485–8493.

[TL19] Mingxing Tan and Quoc V Le. "EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks". In: *Proceedings of International Conference on Machine Learning*. PMLR. 2019, pp. 6105–6114.

[Ton+21] Guanghui Tong et al. "Gradmfl: Gradient memory-based federated learning for hierarchical knowledge transferring over non-iid data". In: *Proceedings of International Conference on Algorithms and Architectures for Parallel Processing*. Springer. 2021, pp. 612–626.

[Tou+23] Hugo Touvron et al. “Llama: Open and efficient foundation language models”. In: *arXiv preprint arXiv:2302.13971* (2023).

[TSM24] Thin Tharaphe Thein, Yoshiaki Shiraishi, and Masakatu Morii. “Personalized federated learning-based intrusion detection system: Poisoning attack and defense”. In: *Future Generation Computer Systems* 153 (2024), pp. 182–192.

[TTN20] Canh T Dinh, Nguyen Tran, and Josh Nguyen. “Personalized federated learning with moreau envelopes”. In: *Advances in neural information processing systems* 33 (2020), pp. 21394–21405.

[Tul+20] Anup Tuladhar et al. “Building machine learning models without sharing patient data: A simulation-based analysis of distributed learning by ensembling”. In: *Journal of biomedical informatics* 106 (2020), p. 103424.

[Tur14] Peter Turney. *Deterding Vowel Recognition Data*. OpenML. 2014. URL: `https://www.openml.org/search?type=data&sort=runs&id=58&status=active`.

[Vai+21] Akhil Vaid et al. “Federated learning of electronic health records to improve mortality prediction in hospitalized patients with COVID-19: machine learning approach”. In: *JMIR medical informatics* 9.1 (2021), e24207.

[Ven+22] Yeshwanth Venkatesha et al. “Addressing client drift in federated continual learning with adaptive optimization”. In: *Available at SSRN 4188586* (2022).

[Vep+18] Praneeth Vepakomma et al. “Split learning for health: Distributed deep learning without sharing raw patient data”. In: *arXiv preprint arXiv:1812.00564* (2018).

[VHM20] Jan Vom Brocke, Alan Hevner, and Alexander Maedche. “Introduction to design science research”. In: *Design science research. Cases* (2020), pp. 1–13.

[Wan+20] Hao Wang et al. “Optimizing federated learning on non-iid data with reinforcement learning”. In: *Proceedings of Conference on Computer Communications (INFOCOM)*. IEEE. 2020, pp. 1698–1707.

[Wan+23a] Sheng Wan et al. “Leveraging Federated Learning for Unsecured Loan Risk Assessment on Decentralized Finance Lending Platforms”. In: *Proceedings of International Conference on Data Mining Workshops (ICDMW)*. IEEE. 2023, pp. 663–670.

[Wan+23b] Zhe Wang et al. “Federated probability memory recall for federated continual learning”. In: *Information Sciences* 629 (2023), pp. 551–565.

[Wen+23] 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.

[WL08] YY Wang and J Li. “Feature-selection ability of the decision-tree algorithm and the impact of feature-selection/extraction on decision-tree results based on hyperspectral data”. In: *International Journal of Remote Sensing* 29.10 (2008), pp. 2993–3010.

[WL22] Guoyizhe Wei and Xiu Li. “Knowledge lock: Overcoming catastrophic forgetting in federated learning”. In: *Proceedings of Pacific-Asia Conference on Knowledge Discovery and Data Mining*. Springer. 2022, pp. 601–612.

[WNC11] Yang Wang, Gregory Norice, and Lorrie Faith Cranor. “Who Is Concerned about What? A Study of American, Chinese and Indian Users’ Privacy Concerns on Social Network Sites: (Short Paper)”. In: *Proceedings of 4th International Conference on Trust and Trustworthy Computing (RUST) 2011, Pittsburgh, PA, USA, June 22-24, 2011. Proceedings 4*. Springer. 2011, pp. 146–153.

[WSG05] Lior Wolf, Amnon Shashua, and Donald Geman. “Feature Selection for Unsupervised and Supervised Inference: The Emergence of Sparsity in a Weight-Based Approach.” In: *Journal of Machine Learning Research* 6.11 (2005).

[Wu+19] Bichen Wu et al. “Fbnet: Hardware-aware efficient convnet design via differentiable neural architecture search”. In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2019, pp. 10734–10742.

[Wu+22] Xiang Wu et al. “An adaptive federated learning scheme with differential privacy preserving”. In: *Future Generation Computer Systems* 127 (2022), pp. 362–372.

[WWC16] Lipo Wang, Yaoli Wang, and Qing Chang. “Feature selection methods for big data bioinformatics: A survey from the search perspective”. In: *Methods* 111 (2016), pp. 21–31.

[Xia+21] Qi Xia et al. “A survey of federated learning for edge computing: Research problems and solutions”. In: *High-Confidence Computing* 1.1 (2021), p. 100008.

[Xia17] Xiao, Han et al. “Fashion-Mnist: a novel image dataset for benchmarking machine learning algorithms”. In: *arXiv:1708.07747* (2017).

[Xu+10] Zenglin Xu et al. “Discriminative semi-supervised feature selection via manifold regularization”. In: *IEEE Transactions on Neural networks* 21.7 (2010), pp. 1033–1047.

[Xu+21a] Jie Xu et al. “Federated learning for healthcare informatics”. In: *Journal of healthcare informatics research* 5 (2021), pp. 1–19.

[Xu+21b] Qianli Xu et al. “Predicting event memorability from contextual visual semantics”. In: *Advances in Neural Information Processing Systems* 34 (2021), pp. 22431–22442.

[Xu+23a] Anran Xu et al. “DIPA: An Image Dataset with Cross-cultural Privacy Concern Annotations”. In: *Companion Proceedings of the 28th International Conference on Intelligent User Interfaces*. 2023, pp. 259–266.

[Xu+23b] Zhanyang Xu et al. “MSEs Credit Risk Assessment Model Based on Federated Learning and Feature Selection.” In: *Computers, Materials & Continua* 75.3 (2023).

[Xu+24] Anran Xu et al. “DIPA2: An Image Dataset with Cross-cultural Privacy Perception Annotations”. In: *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 7.4 (2024), pp. 1–30.

[Xue+21] Zeyue Xue et al. “A resource-constrained and privacy-preserving edge-computing-enabled clinical decision system: A federated reinforcement learning approach”. In: *IEEE Internet of Things Journal* 8.11 (2021), pp. 9122–9138.

[Yan+19] Qiang Yang et al. “Federated machine learning: Concept and applications”. In: *ACM Transactions on Intelligent Systems and Technology (TIST)* 10.2 (2019), pp. 1–19.

[Yan+21] Helin Yang et al. “Privacy-preserving federated learning for UAV-enabled networks: Learning-based joint scheduling and resource management”. In: *IEEE Journal on Selected Areas in Communications* 39.10 (2021), pp. 3144–3159.

[Yan+24] Xin Yang et al. “Federated Continual Learning via Knowledge Fusion: A Survey”. In: *IEEE Transactions on Knowledge and Data Engineering* 36.8 (2024), pp. 3832–3850.

[Yem+22] Michal Yemini et al. “Semi-decentralized federated learning with collaborative relaying”. In: *Proceedings of IEEE International Symposium on Information Theory (ISIT)*. IEEE. 2022, pp. 1471–1476.

[YL03] Lei Yu and Huan Liu. “Feature selection for high-dimensional data: A fast correlation-based filter solution”. In: *Proceedings of the 20th international conference on machine learning (ICML-03)*. 2003, pp. 856–863.

[Yu+15] Han Yu et al. “Efficient task sub-delegation for crowdsourcing”. In: *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 29. 1. 2015.

[Yua+23] Liangqi Yuan et al. "Peer-to-peer federated continual learning for naturalistic driving action recognition". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2023, pp. 5250–5259.

[Zha+18] Yue Zhao et al. "Federated learning with non-iid data". In: *arXiv preprint arXiv:1806.00582* (2018).

[Zha+20] Jingwen Zhang et al. "GAN enhanced membership inference: A passive local attack in federated learning". In: *Proceedings of International Conference on Communications (ICC)*. IEEE. 2020, pp. 1–6.

[Zha+22a] Chi Zhang et al. "Augmented multi-party computation against gradient leakage in federated learning". In: *IEEE Transactions on Big Data* (2022). DOI: 10.1109/TBDATA.2022.3208736.

[Zha+22b] Jie Zhang et al. "Federated learning with label distribution skew via logits calibration". In: *Proceedings of International Conference on Machine Learning*. PMLR. 2022, pp. 26311–26329.

[Zha+22c] Rui Zhang et al. "Secure feature selection for vertical federated learning in ehealth systems". In: *Proceedings of International Conference on Communications (ICC)*. IEEE. 2022, pp. 1257–1262.

[Zha+22d] Jianxin Zhao et al. "Participant selection for federated learning with heterogeneous data in intelligent transport system". In: *IEEE transactions on intelligent transportation systems* 24.1 (2022), pp. 1106–1115.

[Zha+23a] Xiaojin Zhang et al. "Trading off privacy, utility, and efficiency in federated learning". In: *ACM Transactions on Intelligent Systems and Technology* 14.6 (2023), pp. 1–32.

[Zha+23b] Xunzheng Zhang et al. "Federated feature selection for horizontal federated learning in IoT networks". In: *IEEE Internet of Things Journal* 10.11 (2023), pp. 10095–10112.

[Zha+23c] Yong Zhang et al. "An embedded vertical-federated feature selection algorithm based on particle swarm optimisation". In: *CAAI Transactions on Intelligence Technology* 8.3 (2023), pp. 734–754.

[Zha+23d] Zhao Zhang et al. "Communication-efficient federated continual learning for distributed learning system with Non-IID data". In: *Science China Information Sciences* 66.2 (2023), p. 122102.

[Zha+24] Rui Zhang et al. "Vertical Federated Learning Across Heterogeneous Regions for Industry 4.0". In: *IEEE Transactions on Industrial Informatics* 20.8 (2024), pp. 10145–10155.

[Zhe+22] Zhaohua Zheng et al. "Applications of federated learning in smart cities: recent advances, taxonomy, and open challenges". In: *Connection Science* 34.1 (2022), pp. 1–28.

[Zho+17] Haoti Zhong et al. “A Group-Based Personalized Model for Image Privacy Classification and Labeling.” In: *Proceedings of International Joint Conference on Artificial Intelligence(IJCAI)*. Vol. 17. 2017, pp. 3952–3958.

[Zhu+21] Hangyu Zhu et al. “Federated learning on non-IID data: A survey”. In: *Neurocomputing* 465 (2021), pp. 371–390.

[ZHZ21] Zhuangdi Zhu, Junyuan Hong, and Jiayu Zhou. “Data-free knowledge distillation for heterogeneous federated learning”. In: *Proceedings of International conference on machine learning*. PMLR. 2021, pp. 12878–12889.

[ZL07] Zheng Zhao and Huan Liu. “Semi-supervised feature selection via spectral analysis”. In: *Proceedings of the SIAM International Conference on Data Mining*. SIAM. 2007, pp. 641–646.

[ZYZ21] Xiao Zeng, Ming Yan, and Mi Zhang. “Mercury: Efficient on-device distributed DNN training via stochastic importance sampling”. In: *Proceedings of the 19th ACM Conference on Embedded Networked Sensor Systems*. 2021, pp. 29–41.

[ZZL16] Qifeng Zhou, Hao Zhou, and Tao Li. “Cost-sensitive feature selection using random forest: Selecting low-cost subsets of informative features”. In: *Knowledge-based systems* 95 (2016), pp. 1–11.

Paper I

Fed-FiS: A Novel Information-Theoretic Federated Feature Selection for Learning Stability

Sourasekhar Banerjee, Erik Elmroth, and Monowar Bhuyan

Proceedings of the 28^{th} *International Conference on Neural Information Processing (ICONIP)*, Springer Cham., Communications in Computer and Information Science, Vol. 1516, pp. 480-487, 2021.

Fed-FiS: A Novel Information-Theoretic Federated Feature Selection for Learning Stability*

Sourasekhar Banerjee, Erik Elmroth, Monowar Bhuyan

Department of Computing Science, Umeå University, Umeå, Sweden

sourasb@cs.umu.se, elmroth@cs.umu.se, monowar@cs.umu.se

Abstract: In the era of big data and federated learning, traditional feature selection methods show unacceptable performance for handling heterogeneity when deployed in federated environments. We propose Fed-FiS, an information-theoretic federated feature selection approach to overcome the problem that occurs due to heterogeneity. Fed-FiS estimates feature-feature mutual information (FFMI) and feature-class mutual information (FCMI) to generate a local feature subset in each user device. Based on federated values across features and classes obtained from each device, the central server ranks each feature and generates a global dominant feature subset. We show that our approach can find stable features subset collaboratively from all local devices. Extensive experiments based on multiple benchmark iid (independent and identically distributed) and non-iid datasets demonstrate that Fed-FiS significantly improves overall performance in comparison to the state-of-the-art methods. This is the first work on feature selection in a federated learning system to the best of our knowledge.

Key words: Federated Learning, Feature Selection, Mutual Information, Classification, Statistical Heterogeneity

1 Introduction

Feature subset selection is a crucial task in data mining, knowledge discovery, pattern recognition, and machine learning to construct cost-effective models for multiple applications. Information-theoretic measures have been widely used and established paradigm for feature selection. Specifically, mutual information-based feature selection (MIFS) empowers identifying relevant features subset by removing redundant and irrelevant features without impacting classifiers reachable performance. In general, feature selection techniques are classified into four categories [Man21] such as filter, wrapper, embedded, and hybrid. Traditional MIFS approaches [Hoq14; Liu21] are designed for centralized systems where data stored in the server. Having terabytes of user generated data and bringing them to the central server for constructing a model

*The paper has been re-typeset to match the thesis style. Reproduced with permission of Springer.

increases communication cost and also violate users privacy. Hence, the primary solution is to learn the model at the edge of the network using federated machine learning. Performing feature selection without relocating data to a centralized server is challenging because data present in local devices suffer from statistical heterogeneity. Real-world data combines iid and non-iid data and massively distributed across multiple devices. Moreover, local devices are low-end with limited resources, e.g., computation power. In such scenarios, traditional machine learning (ML) methods face difficulty in handling terabytes of data with statistical heterogeneity. Therefore, feature selection is essential and paramount to process such data and uncover useful knowledge for developing low-cost models. Feature selection is worthwhile for federated learning from many aspects, including (1) finding the common features set from the local device's data; hence, federated machine learning algorithms could learn efficiently, (2) dimensionality reduction leads to lowering the computational cost and the model size. The proposed work covered these aspects and introduced a federated feature selection method using mutual information.

Classical feature selection methods are widely developed for centralized systems to solve computational problems that occur due to higher dimensionality [Hoq14; Zhe21; Liu21]. Also, federated feature selection is different from the distributed feature selection [Gui20; Soh20; Mor17] being presence of heterogeneity . We propose a novel information-theoretic federated feature selection approach (Fed-FiS) for identifying relevant features subset from the federated dataset to learn ML models with stability. A federated dataset can be horizontal or iid where all devices have complete information of the features and classes and hybrid or non-iid where all devices don't have full details of the features and classes. But every device must have a features subset that is common in every device. Our major contributions are as follows.

- Fed-FiS introduces a local feature subset selection method by using mutual information and clustering.
- We develop a score function based on FFMI (minimize redundancy) and FCMI (maximize relevance) for global feature subset selection.
- Fed-FiS finds a most relevant features subset from all devices where data is distributed in horizontal or iid manner and in hybrid or non-iid manner.
- We evaluate Fed-FiS with multiple benchmark datasets and achieved cost-effective model performance in both iid and non-iid data partitions.

2 Problem Statement

Consider a federated learning system consists of q local devices ($\forall_{i=1}^{q} Cl_i$) and a server. We assume that $q \geq 2$, if $q = 1$, then it is considered as a centralized system having full information of the dataset. Suppose the dataset $D = \mathbb{R}^{m \times n+1}$ contains the features set $F = \{f_1, f_2, \ldots, f_n\}$, where $f_k \in \mathbb{R}^{n \times 1}$ is the k^{th} feature and n is the total number of features, and class $C \in \{0,1\}^{m \times |F|}$ where m is the number of data samples. D is distributed across q devices such that $\forall_i^q Cl_i$ contains the features set $F_{Cl_i} = \{f_1^{Cl_i}, f_2^{Cl_i}, \ldots, f_{|F_{Cl_i}|}^{Cl_i}\}$, where $f_k^{Cl_i}$ is the k^{th} feature of the i^{th} device, Cl_i. $\mid F_{Cl_i} \mid$

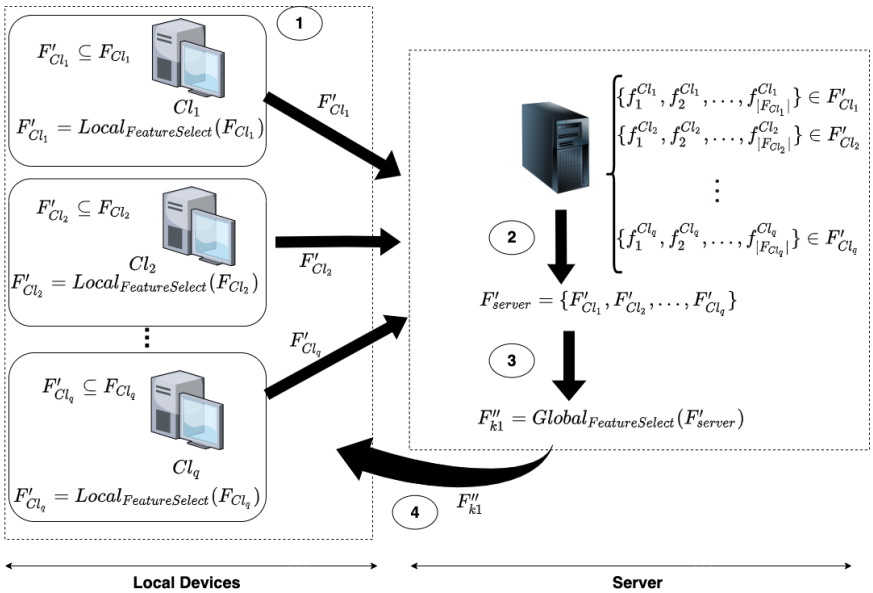


Figure 1: Fed-FiS: the proposed framework.

is the number of features present in Cl_i and $\mid F_{Cl_i} \mid \leq n$. Class $C^{Cl_i} \in \{0,1\}^{m^{Cl_i} \times |F_{Cl_i}|}$, where m^{Cl_i} is number of data samples in the i^{th} device and $m^{Cl_i} \leq m$. The data distribution creates statistical heterogeneity, i.e., each device (Cl_i) doesn't have complete information on the entire dataset. Our objective is to uncover λ strongly relevant features subset (F''_λ) that are present in every device. Hence, $accuracy(\psi(< F''_\lambda, C^{Cl_i} >)) \geq \delta$ and obtain stable global model performance. Here, $\psi(< F''_\lambda, C^{Cl_i} >)$ is the trained model on λ features at local device (Cl_i) and δ is the threshold for model accuracy.

3 Fed-FiS: The Proposed Approach

Fed-FiS is a mutual information-based federated feature selection approach that selects subset of strongly relevant features without relocating raw data from local devices to the server (see Figure 1 for proposed framework). Fed-FiS has two parts, local features selection and global features selection. Initially, local devices independently produce the local features subset, and later, the server generates global features subset. The proposed approach is described as follows.

3.1 Local feature selection

Fed-FiS employs MI to measure the amount of uncertainty in a feature variable with respect to a target variable. Algorithm 1 first computes aFFMI and FCMI scores of all features (steps 2 to 4). $F_{Cl_i}^{aFFMI}$ and $F_{Cl_i}^{FCMI}$ are two 1d vector that contains

aFFMI and FCMI scores of all features at local device, Cl_i (steps 5 and 6). The aFFMI score close to zero indicates low redundancy and FCMI score near to one means high relevance of a feature. Here, our objective is to find the features that have low redundancy or high relevance. Based on the aFFMI and FCMI scores, we compute CLUSTER($F_{Cl_i}^{aFFMI}$) and CLUSTER($F_{Cl_i}^{FCMI}$) (steps 7 and 8) using the procedure CLUSTER (step 11 to 24) to generate feature clusters with lower aFFMI scores and higher FCMI scores. Suppose clustering on aFFMI scores produce β_1 clusters, and similarly clustering on FCMI scores generate β_2 clusters then the objective function can be defined as follows, $F'_{Cl_i{}^{aFFMI}} = \arg\min_{\forall i \in \beta_1} centroid\ (cluster_i)$ and $F'_{Cl_i{}^{FCMI}} = \arg\max_{\forall i \in \beta_2} centroid(cluster_i)$, where $centroid(cluster_i)$ returns the centroid value of the i^{th} cluster, and $\mid cluster_i \mid$ is the cardinality of $cluster_i$. Union of the output of steps 7 and 8 produces the final local features subset $F'_{Cl_i} \subseteq F_{Cl_i}$ (steps 9). Figure 2 illustrates how MI and clustering help to obtain strongly relevant local features subset in i^{th} local device, Cl_i.

Algorithm 1 Fed-FiS (Local feature selection)

Input: $F_{Cl_i} = \{f_1^{Cl_i}, f_2^{Cl_i}, \ldots, f_{|F_{Cl_i}|}^{Cl_i}\}$ is the original feature set with $\mid F_{Cl_i} \mid$ dimension and class C^{Cl_i} for the i^{th} local device Cl_i

Output: F'_{Cl_i} is the local features subset from the i^{th} local device Cl_i

1: **procedure** $Local_{FeatureSelect}(< F_{Cl_i}, C^{Cl_i} >)$
2: **for** $f_k^{Cl_i} \in F_{Cl_i}$ **do**
3: $f_k^{aFFMI} = aFFMI(f_k^{Cl_i}) = \frac{1}{|F_{Cl_i}|-1} \sum_{j=1, f_j^{Cl_i} \in F_{Cl_i} \setminus f_k^{Cl_i}}^{|F_{Cl_i}|-1} FFMI(f_k^{Cl_i}, f_j^{Cl_i})$. ▷ return averaged FFMI score of f_k
4: $f_k^{FCMI} = FCMI(f_k^{Cl_i}, C^{Cl_i}) \triangleq \sum_{f_k^{Cl_i}, C^{Cl_i}} Pr(f_k^{Cl_i}, C^{Cl_i}) log \frac{Pr(f_k^{Cl_i}, C^{Cl_i})}{Pr(f_k^{Cl_i}) Pr(C^{Cl_i})}$. ▷ return FCMI score of f_k
5: $F_{Cl_i}^{aFFMI} = \{f_k^{aFFMI} \mid \forall_{k=1}^{d} f_k^{aFFMI} \in [0,1]\}$ ▷ vector of aFFMI scores of all features at Cl_i
6: $F_{Cl_i}^{FCMI} = \{f_k^{FCMI} \mid \forall_{k=1}^{d} f_k^{FCMI} \in [0,1]\}$ ▷ vector of FCMI scores of all features at Cl_i
7: $F'_{Cl_i{}^{aFFMI}}$ = CLUSTER($F_{Cl_i}^{aFFMI}$) ▷ return cluster of features with low aFFMI scores
8: $F'_{Cl_i{}^{FCMI}}$ = CLUSTER($F_{Cl_i}^{FCMI}$) ▷ return cluster of features with high FCMI scores
9: $F'_{Cl_i} = F'_{Cl_i{}^{FCMI}} \bigcup F'_{Cl_i{}^{aFFMI}}$ ▷ union of features with high FCMI and low aFFMI scores
10: return F'_{Cl_i} ▷ $F'_{Cl_i} \subseteq F_{Cl_i}$
11: **procedure** CLUSTER($F_{Cl_i}^{x}$) ▷ x is either aFFMI or FCMI
12: initialize μ random cluster centroid.
13: **repeat**
14: $\forall_{k=1}^{|F_{Cl_i}^{x}|} f_k \in F_{Cl_i}^{x}$
15: minimum $\leftarrow 0$
16: cluster_member $\leftarrow 0$
17: $\forall_{c=1}^{\mu} centroid\ c \in \mu$
18: dist $\leftarrow$ Distance(f_k, c)
19: **if** $dist < minimum$ **then**
20: minimum $\leftarrow$ dist
21: cluster_member $\leftarrow$ c
22: recalculate centroid(c)
23: **until** converge
24: return $F'_{Cl_i{}^{x}}$. ▷ it returns cluster of features with lowest centroid while clustering aFFMI scores and highest centroid while clustering FCMI scores.

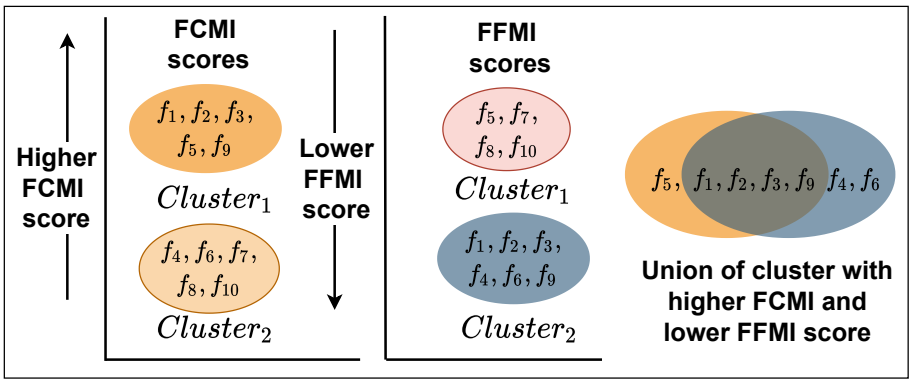


Figure 2: Illustration of the local feature selection.

3.2 Global feature selection

The steps for global feature selection is given in Algorithm 2. Where each local device (Cl_i) sends triplets of locally selected features ($\tau_k^{Cl_i} =< f_k^{Cl_i}, f_{k_{FCMI}}^{Cl_i}, f_{k_{aFFMI}}^{Cl_i} >$) to the server for computing the global scores. The triplet vector F'_{Cl_i} of the i^{th} device Cl_i is defined as: $F'_{Cl_i} = \{\tau_1^{Cl_i}, \tau_2^{Cl_i}, \ldots, \tau_k^{Cl_i}\}$. Server receives feature triplets from q local devices (step 2). A single feature f_k can be shared by multiple local devices. F_{server} may have multiple similar features with different aFFMI and FCMI scores. Server averages aFFMI and FCMI scores of the similar features, respectively (step 3). Server generates the unique features set F'_{server} (step 4). We propose a score function to globally rank each features and produce strongly relevant features subset.

Algorithm 2 Fed-FiS (Global feature selection)

Input: $F_{Server} = \{F'_{Cl_1}, F'_{Cl_2} \ldots F'_{Clq}\}$ ▷ collection of feature triplets from q local devices
Output: F''_k ▷ global feature subset

1: **procedure** $Global_{FeatureSelect}(F_{Server})$
2: server obtained $F_{server} = \{F'_{Cl_i} | \forall_{i=1}^{q}, F'_{Cl_i} \in Cl_i\}$.
3: obtain global feature triplet by performing average over aFFMI and FCMI scores individually.
4: obtain $\{F'_{server} | \forall f_k \in F'_{server}$ *are unique*$\}$
5: compute $S(f_k), \forall f_k \in F'_{server}$ using $S(f_k) = f_{k_{FCMI}} - \frac{1}{(|F'_{server}|-1)} \times f_{k_{aFFMI}}$, *where* $| F'_{server} | > 1$ *and* $S(f_k) \in [1, -1]$
6: $\forall_{i=1}^{q} Cl_i$ send $< S(f_k), f_k >$ to all Cl_i iff $f_k \in F'_{server}$ and $f_k \in F'_{Cl_i}$
7: $\forall_{i=1}^{q} Cl_i$, each Cl_i selects a feature subset $F''_{Cl_i} \subseteq F'_{server}$, where $accuracy(\psi(< F''_{Cl_i}, C^{Cl_i} >)) \geq \delta$ ▷ δ is an accuracy threshold and send F''_{Cl_i} to server.
8: server applies $\bigcap_{i=1}^{q} F''_{Cl_i} = F''_{\lambda}$ to generate global feature subset F''_{λ} of size λ
9: return F''_{λ} to all local devices.

3.2.1 Global score function

Server computes the score of each feature present in F'_{server} to rank features based on FCMI and aFFMI scores (Step 5) . $S(f_k)$ = 1 iff $f_{k_{FCMI}}$ = 1 and $f_{k_{aFFMI}} = 0$, similarly $S(f_k)$ = -1 iff $f_{k_{FCMI}}$ = 0 , $f_{k_{aFFMI}} = 1$ and $\mid F'_{server} \mid = 2$. The server ranks each feature based on the descending order of the global scores and sends the feature score vector $< S(f_k), f_k >$ to all local devices iff the device originally contains this feature (step 6). $\forall_{i=1}^{q} Cl_i$, each Cl_i selects a feature subset F''_{Cl_i} from the F'_{server}, where $accuracy(\psi(< F''_{Cl_i}, C^{Cl_i} >)) \geq \delta$ (step 7). Here, $\psi(< F''_{Cl_i}, C^{Cl_i} >)$ is a learning model, and δ is the benchmark accuracy collected for comparing with state-of-the-art methods. Server collects dominant features set from each local devices and perform intersection ($\bigcap_{i=1}^{q} F''_{Cl_i} = F''_{\lambda}$) and produce global features subset F''_{λ}(step 8) of size λ. Finally, server distributes optimal features subset to all local devices (step 9) for learning.

4 Evaluation

In this section, we were carried out exhaustive experiments to validate the performance of Fed-FiS using simulated environment with multiple benchmark datasets, considering numerous number of local devices. We used the NSL-KDD99 [Tav09], and the anonymized credit card transactions (ACC) [1] datasets for our experiments. We divide both datasets into horizontal (iid) and hybrid (non-iid) manner across five local devices. We carried out the analysis of the results in three parts: (1) cluster analysis at each local device for local features subset selection, (2) global features subset selection at server, and (3) performance of selected features with multiple learning models.

4.1 Cluster analysis and local features subset selection

We employ the k-means clustering to group the estimated aFFMI and FCMI scores obtain from each feature in NSL-KDD99 dataset distributed among five devices. The primary task is to identify lowest redundant and strongly relevant features subset. Each cluster verifies with silhouette coefficient (SC) to ensure obtaining quality and optimal clusters. From Figure 3(a), we observed that the cluster size two makes maximum averaged SC for all devices. Similarly, cluster size three provides maximum average SC observed from Figure 3(b). Hence, we consider the cluster size 3 for aFFMI and 2 for FCMI for our experiments. From Figure 3(c), we select cluster 2 that has maximum centroid value (centroid -2). Similarly, from Figure 3(d), we select cluster 1 with minimum centroid value (centroid -1). Union among these two selected clusters produce the local feature subset for the i^{th} device Cl_i. The local feature subset of device Cl_1

[1] Worldline and the ULB ML Group. Anonymized credit card transactions labeled as fraudulentor genuine.https://www.kaggle.com/mlg-ulb/creditcardfraud, 2020

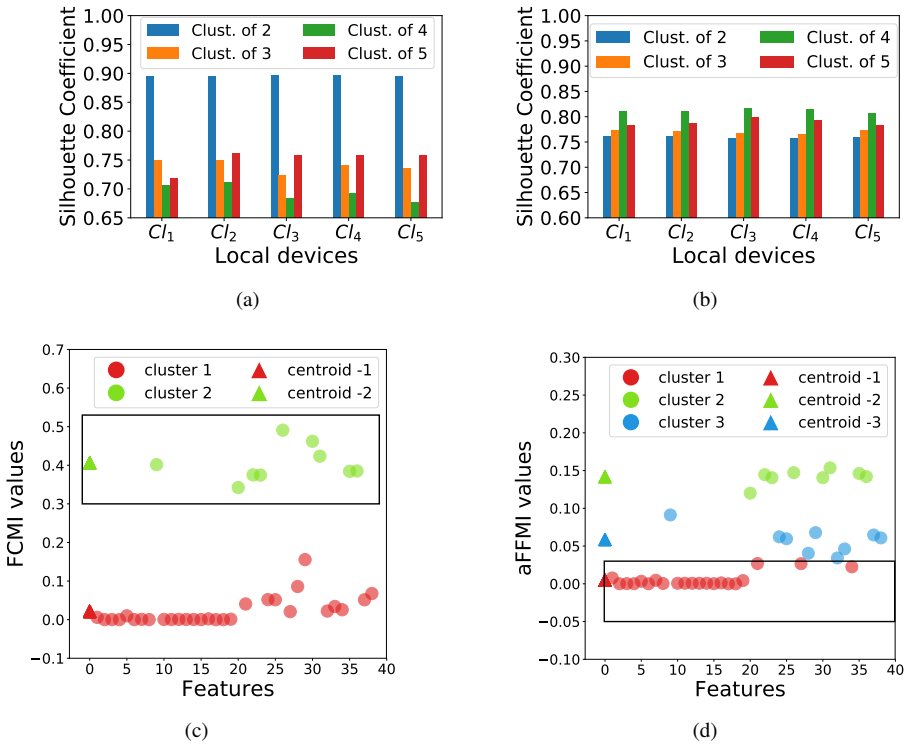


Figure 3: Cluster analysis for local feature selection on horizontal or iid distribution of NSL-KDD99 dataset among five local devices. (a) FCMI cluster analysis, (b) aFFMI cluster analysis, (c) Clusters of features with respect to the FCMI scores for device Cl_1, (d) Clusters of features with respect to the aFFMI scores for device Cl_1.

has 30 features. Now, we are able to exclude 11 irrelevant features. For other datasets, we followed the similar approach to obtain the strongly relevant and optimal features set.

4.2 Global features subset selection

We learned the KNN model at each local device keeping $\delta = 97\%$. All local devices reach the threshold (see Figure 4(a)) using the top 23 features for NSL-KDD99 iid dataset.Similarly for the NSL-KDD99 non-iid dataset, local devices C1 to C4 crossed the threshold for the top 23 features, and C5 has crossed the threshold with top 35 features (see Figure 4(b)). The intersection among these features generates the global relevant features subset of size 8.

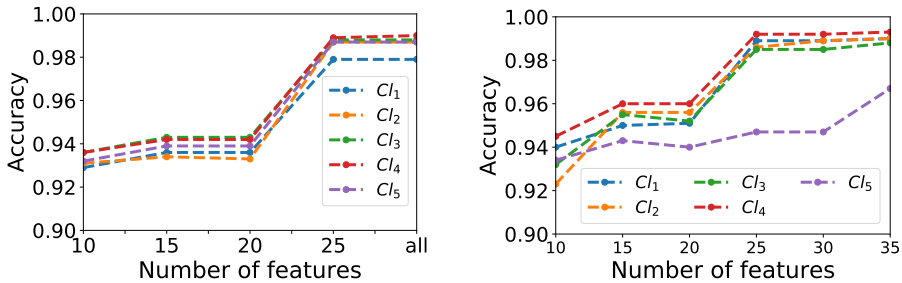


Figure 4: Global feature selection using single-objective optimization for NSL-KDD99 dataset. (a) Feature subset for iid dataset at five local devices, (b) Feature subset for non-iid dataset at five local devices.

Table 1: Performance of Fed-FiS with without feature selection and baseline models across Federated Forest

Datasets	**Fed-FiS**		without FS		classical MI-based FS	
	Optimal feature subset	Accuracy (%)	Feature set	Accuracy (%)	Feature subset	Accuracy (%)
NSL-KDD99 iid	23	99.3	41	99.5	23	99.24
NSL-KDD99 non-iid	9	89.33	15	54.07	7	92.75
ACC iid	10	99.95	29	99.96	6	99.3
ACC non-iid	9	99.83	14	99.81	10	99.83

4.3 Steady learning ability

We trained the federated forest model on both iid and non-iid division of NSL-KDD99 and the ACC datasets across 5 local devices. We compare the feature's subset size and the model's performance in Table 1 among Fed-FiS, MI-based baseline feature selection method[2](state-of-the-art), and without using feature selection (without FS). In the the baseline method, we computed MI of all features that are present in different devices and then does the intersection among them to obtain feature subset for learning.

On NSL-KDD99 and ACC iid dataset, Fed-FiS selects global features subset of sizes 23 and 10, respectively, and achieves 99.3% and 99.83% accuracy, respectively. In contrast, without choosing any features, the federated forest achieves 99.5% and 99.96% accuracy, respectively. The state-of-the-art method generates subsets of sizes 23 and 6, respectively and produces accuracy of 99.24% and 99.3%, respectively. The performance of proposed method is almost equivalent with minimum number of relevant features set in compare to the state-of-the-art and without FS.

For both NSL-KDD99 and ACC non-iid dataset, Fed-FiS selects a global feature subset of 9 and achieves 89.33% and 99.83% accuracy, respectively. The state-of-the-art method selects features subset of size 7 and 10, respectively and obtain 92.75% and 99.83% accuracy, respectively. Without FS, i.e., only selecting common features of

[2]https://scikit-learn.org/stable/modules/generated/sklearn.feature_selection.mutual_info_classif.html

five devices, it achieves 54.07% and 99.81% accuracy, respectively. For the non-iid scenario, the algorithm also depends on the common features set of each local device. If all devices share features more extensively, then the feature selection yields stability in learning.

5 Conclusion and Future Work

In this paper, we propose Fed-FiS, a mutual information-based federated feature selection method to select strongly relevant features subset for stable and low-cost federated learning. We used local feature selection on every local device using clustering of aFFMI and FCMI scores to select feature subset in federated settings. The server produces the global rank of each feature and generates a global feature subset. Fed-FiS achieved expected model performance with lower number of features set, verified with federated forest algorithm. The extension of this work for anomaly detection in edge clouds is underway.

Acknowledgments

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

References

[Gui20] Yu Gui. "ADAGES: adaptive aggregation with stability for distributed feature selection". In: *Proceedings of the ACM-IMS on Foundations of Data Science Conference*. 2020, pp. 3–12.

[Hoq14] N Hoque et. al. "MIFS-ND: A mutual information-based feature selection method". In: *Expert Systems with Applications* 41.14 (2014), pp. 6371–6385.

[Liu21] G Liu et. al. "Feature Selection Method Based on Mutual Information and Support Vector Machine". In: *Int. Journal of Pattern Recognition and Artificial Intelligence* (2021), p. 2150021.

[Man21] G Manikandan et. al. "Feature Selection Is Important: State-of-the-Art Methods and Application Domains of Feature Selection on High-Dimensional Data". In: *Applications in Ubiquitous Computing*. Springer, 2021, pp. 177–196.

[Mor17] L Morán-Fernández et. al. "Centralized vs. distributed feature selection methods based on data complexity measures". In: *Knowledge-Based Systems* 117 (2017), pp. 27–45.

[Soh20] M Soheili et. al. “DQPFS: distributed quadratic programming based feature selection for big data”. In: *Journal of Parallel and Distributed Computing* 138 (2020), pp. 1–14.

[Tav09] M Tavallaee et. al. “A detailed analysis of the KDD CUP 99 data set”. In: *IEEE symp. on computational intelligence for security and defense applications*. 2009, pp. 1–6.

[Zhe21] L Zheng et. al. “Feature grouping and selection: A graph-based approach”. In: *Information Sciences* 546 (2021), pp. 1256–1272.

Paper II

Optimized and Adaptive Federated Learning for Straggler-Resilient Device Selection

Sourasekhar Banerjee, Xuan-Son Vu, and Monowar Bhuyan

Proceedings of the International Joint Conference on Neural Networks (IJCNN), IEEE, pp. 1-9, 2022.

Optimized and Adaptive Federated Learning for Straggler-Resilient Device Selection*

Sourasekhar Banerjee, Xuan-Son Vu, Monowar Bhuyan

Department of Computing Science, Umeå University, Umeå, Sweden

sourasb@cs.umu.se, sonvx@cs.umu.se, monowar@cs.umu.se

Abstract: Federated Learning (FL) has evolved as a promising distributed learning paradigm in which data samples are disseminated over massively connected devices in an IID (Identical and Independent Distribution) or non-IID manner. FL follows a collaborative training approach where each device uses local training data to train local models, and the server generates a global model by combining the local model's parameters. However, FL is vulnerable to system heterogeneity when local devices have varying computational, storage, and communication capabilities over time. The presence of stragglers or low-performing devices in the learning process severely impacts the scalability of FL algorithms and significantly delays convergence. To mitigate this problem, we propose Fed-MOODS, a Multi-Objective Optimization-based Device Selection approach to reduce the effect of stragglers in the FL process. The primary criteria for optimization are to maximize: (i) the availability of the processing capacity of each device, (ii) the availability of the memory in devices, and (iii) the bandwidth capacity of the participating devices. The multi-objective optimization prioritizes devices from fast to slow. The approach involves faster devices in early global rounds and gradually incorporating slower devices from the Pareto fronts to improve the model's accuracy. The overall training time of Fed-MOODS is $\mathbf{1.8}\times$ and $\mathbf{1.48}\times$ faster than the baseline model (FedAvg) with random device selection for MNIST and FMNIST non-IID data, respectively. Fed-MOODS is extensively evaluated under multiple experimental settings, and the results show that Fed-MOODS has significantly improved model's convergence and performance. Fed-MOODS maintains fairness in the prioritized participation of devices and the model for both IID and non-IID settings.

Key words: Federated learning, Adaptive device selection, Statistical heterogeneity, Multi-objective optimization, and Straggler-resilient device

*The paper has been re-typeset to match the thesis style. Reproduced with permission of IEEE.

1 Introduction

Federated learning (FL) is a paradigm in distributed machine learning where multiple devices collaboratively train a model without sharing raw data [Ban20]. Apart from privacy, it reduces the communication burden by sending only the model parameters instead of sending terabytes of data to the server. Implementation of federated learning is very challenging, as it suffers from system (device) heterogeneity and statistical (data) heterogeneity. System heterogeneity refers to devices with varying computation capacity, memory capacity, bandwidth, etc., [Wan21; Yan21; Smi17], and statistical heterogeneity means Identical and Independent Distribution (IID) and non-Identical and Independent Distribution (non-IID) of data [Smi17; Ban21]. Due to stragglers in the federated learning system, keeping statistical accuracy high and dealing with system heterogeneity simultaneously is very challenging. Straggler devices are low performing devices that are incompetent in processing, communicating and storage. Involving stragglers causes significant delays from learning to inference [Kai21].

Federated learning is of two types based on how devices take part in learning: cross-device and cross-silo [Kai21]. In cross-silo FL, every device takes part in every round of the learning process. Compared to that, in cross-device FL, millions of devices are attached to the edge. Since devices are dynamic, all devices cannot be available for the entire process. Therefore, only a few devices participate in the learning process in each round. The server selects a subset of devices randomly for every round of training. However, the random selection of devices works better for straggler-free FL settings. In the presence of huge stragglers, mainly on non-IID data, the random selection based learning approach converges very slowly, and a high impact of randomness is present in the model training [Li 20; Li 19a]. The server's interest is most preferred in client selection i.e., the devices that respond quickly to the server only take part in the learning. As a result, stragglers can never contribute to the FL. Moreover, removing straggler devices and only training models based on the non-straggler devices may not generalize the final model properly and cause huge information loss, which may lead to unfairness in the learning process and jeopardize the sustainability of the FL system. Therefore, it is essential to choose devices such that the model converges quickly, produces sufficient accuracy, and maintains fairness.

To mitigate these problems, Reisizadeh et al. [Rei20] proposed an approach called FLANP that leverages the interplay between model accuracy and device heterogeneity. The algorithm includes faster devices based on computation capability in the early learning rounds and later involves stragglers. However, they only ranked devices based on their computational ability. We considered computation, communications, and storage characteristics of devices altogether and introduced Fed-MOODS, a multi-objective optimization-based adaptive device selection approach. We inferred multi-objective optimization to rank devices based on system performance, i.e., available processing, memory, and

bandwidth capacity. Devices are selected adaptively from the Pareto fronts to contribute in every global rounds. Multiple devices with varying computing and storage capabilities constitute a typical federated learning environment. Due to slow devices or stragglers, applying standard federated learning algorithms such as FedAvg [McM17] on highly heterogeneous devices result in significant and unanticipated delays. Our focus in this work is to mitigate these problems that aggravates from system heterogeneity in the FL framework while keeping the performance of the model stable. We employ interaction between statistical accuracy and system heterogeneity to design a straggler-resilient federated learning approach that selects a subset of available devices adaptively in each global round of training. Our main contributions are as follows.

- We introduce Fed-MOODS, a straggler-resilient multi-objective optimization-based adaptive prioritized device selection approach to mitigate the system heterogeneity problems in federated learning.
- Fed-MOODS considers computation, communications, and storage heterogeneity and formulates them as multi-objective functions to optimize and generate the rank of the local devices.
- Fed-MOODS minimizes the overall wall-clock training time of the model, improves the model's performance, maintains fairness in device selection, and generalizes the final model.
- We experimented Fed-MOODS across multiple benchmark datasets (MNIST, FMNIST, and CIFAR-10) and baseline models (FedAvg, FedProx) with random-device selection. We show that the proposed approach is superior to other baselines models for both IID and non-IID settings.

We assumed that (i) Devices would share the system level information with the server. (ii) Local devices and the server are both trustworthy. (iii) During the learning process, the device's local data remains unchanged, and (iv) all participating devices remain active for the whole learning.

Organization. The rest parts of paper is organized as follows: we provide a brief literature survey in Section 2. The proposed approach is given in Section 3. Experiments, results, and analyses are reported in Section 4. Finally, the conclusion and future work are discussed in Section 5.

2 Related Work

Federated Learning [McM17] allows users to learn a predictive model collaboratively while maintaining privacy, ownership, and data localization. Each participating device produces a model update during local training, which is sent to the server and aggregated with other devices' models to produce the global update [Wan21]. This global update is subsequently distributed to all participating devices, allowing them to improve their local models in next consecutive rounds. The participating devices are heterogeneous from the system and data perspective. Federated learning causes system heterogeneity problems

for devices' having different processing, communication, and storage capacities. Asynchronous approaches have shown considerable benefits in distributed or decentralized learning [Lia18], but these approaches are not very attractive in FL for the staleness of slow devices [Sti18; Xie19]. FLANP [Rei20], a straggler-resilient adaptive device participation algorithm to reduce the stragglers' effect in FL. The learning begins with computationally faster devices and then adaptively includes the slower devices. This process continues until the model converges. In [Wan21], the authors analyzed the impact of statistical heterogeneity on the device selection, the convergence of the model, and fault tolerance in FL settings. In [Mit21], the authors showed that the existing federated algorithms suffer from a speed-accuracy problem in presence of statistical and system heterogeneity. The algorithm finds global minima at a sub-linear rate. To solve that issue, they proposed FedLin, which guarantees linear convergence to the global minima. In [Yan21], the authors carried out empirical studies on the effect of system heterogeneity in the FL system. They built a heterogeneity-aware FL framework that compiles standard federated algorithms while considering the system heterogeneity. In [Shi21a], the authors proposed a FL model with attention transfer that reduces the effect of stragglers. A few more works on adaptive FL are in [Can21; Li 21].

Multi-objective optimization-based solutions in federated learning is interesting in finding model optimality by satisfying the fairness constraints of every participating device [Cui21; Hu 20]. Fairness in federated learning is a challenging task to accomplish [Shi21b]. Unfairness may arise in different phases of the FL process, starting from local device selection [Zho21; Hua20] to model optimization [Moh19; Li 19b]. The notions of fairness in FL can be categorized in different ways, such as, accuracy parity [Li 19b], good-intent fairness [Moh19], selection fairness [Zho21; Hua20], contribution fairness [Lyu20], and many more. In this paper, we only considered good-intent fairness and selection fairness.

Here, we employ the advantages of multi-objective optimization to achieve optimal performance of the learned model in presence of stragglers.

3 Proposed Approach

This section starts by describing the system model of the proposed approach. Then, we formulate the adaptive FL problem and the objectives of device selection. After that, we describe the Fed-MOODS algorithm and analyze the computation time for learning.

3.1 System model

Many heterogeneous devices are distributed across the edge of the network and connected to the global server in typical FL settings. A set of devices is selected adaptively from the Pareto fronts in every global round (see Figure 1). On the

other end, a global server is present to orchestrate the learning process and build the global model. The server broadcasts the learned model to all the devices. The procedure continues until the model converges.

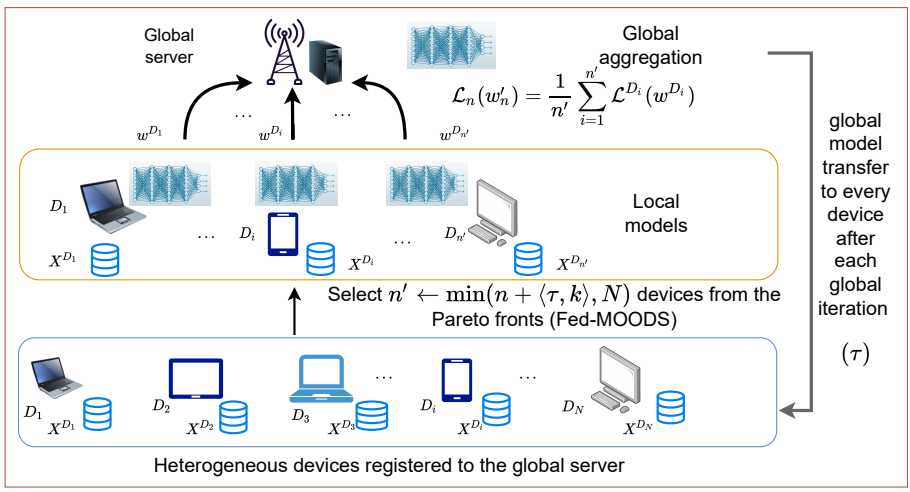


Figure 1: Overview of Fed-MOODS framework - an adaptive straggler-resilient device selection.

3.2 Problem formulation

The problem formulation is divided into two categories: (i) Multi-objective formulations of device heterogeneous properties, such as computation or processing, communication, and storage capacity. The primary goal is to optimize these functions or properties, generate Pareto fronts, and rank devices based on them. (ii) The formulation of the empirical loss function for the adaptive device selection based on Pareto fronts to mitigate the statistical heterogeneity problem in FL.

3.2.1 Multi-objective formulations

We formulate three objective functions based on the available device computation or processing, memory, and bandwidth capacity of each device.

Maximize available processing capacity: Low level performance can measure the instructions per cycle (IPC) of each device processing capacity[Yad16]. In a multi-core environment, a processor can handle multiple instructions per clock cycle. Multiple devices have CPU (central processing unit) with different processing capacities in a heterogeneous federated environment.

The CPU utilization (u) is estimated as: $u = 1 - p^a$ [Yad16], where a is the number of processes currently running, p is the average percentage of waiting time[Yad16]. If the device contains c multiple cores then the overall CPU utilization of each device can be defined as:

$$\begin{aligned} D_u &= \frac{1}{c}\sum_{i=1}^{c}(1-p^a) \\ where \quad & D_u \in [0,1] \end{aligned} \tag{1}$$

A device contains GPU (Graphics Processing Unit) along with CPU. Based on Equation (1), GPU utilization is D_{gu} and CPU utilization is D_{cu}. The operating system (OS) of each device checks how much GPU (D_g) and CPU(D_c) are free using Equations (2) and (3), respectively.

$$D_g = (1 - D_{gu})(\%) \tag{2}$$

$$D_c = (1 - D_{cu})(\%) \tag{3}$$

Suppose N devices are participating in FL. The server attempts to maximize the available processing capacity (D_i^{PA}) as in Equation (4).

$$\begin{aligned} \max_{i=1}^{N} D_i^{PA} &= \frac{1}{2}(D_g + D_c) \\ \text{s.t. } & 0 \leq D_g \leq 100, \\ & 0 \leq D_C \leq 100 \end{aligned} \tag{4}$$

Maximize available memory: Memory requirement (MR) for device D_i is the amount of memory required to train a neural network model[1]. While training a ConvNet, the total required memory includes storage for parameters, intermediate layers, and the gradient of each parameter. An extra memory is needed if the learning uses optimizers like momentum, RMSprop, Adams, etc. Hence, the memory requirement (D_i^{MR}) to learn a neural network is calculated as follows.

$$D_i^{MR} = B \times \sum_{l=1}^{L} MR_l \times Byte$$

where the neural network has L layers (including input and fully connected layers) and B is *Batch size*. Suppose the i^{th} device, D_i has current total memory D_i^{TM}. The available memory, D_i^{AMR} is computed as follows.

$$D_i^{AMR} = D_i^{TM} - D_i^{MR}$$

Now server collects D_i^{AMR} from the N devices and maximize in the following Equation (5).

[1] https://cs231n.github.io/convolutional-networks/#case

$$\begin{aligned} &\max_{i=1}^{N} D_i^{AMR} \\ &\text{s.t.} \frac{D_i^{TM}}{D_i^{MR}} \geq 1, \\ &0 \leq D_i^{TM} \leq 100, \\ &0 \leq D_i^{MR} \leq 100 \end{aligned} \tag{5}$$

Maximize available bandwidth: Network bandwidth can be estimated based on the amount of data transferred between devices and the server. Each device calculates the total amount of data (D_i^{TD}) to be replicated in gigabytes, data duplication ratio (D_i^{DR}), and length of the replication window time (D_i^{RWT}) in seconds. The server collects these information from each device and calculates the required network bandwidth[2] (D_i^{RNB})in Gbps for N devices using Equation (6).

$$D_i^{RNB} = \frac{D_i^{TD} * (100/D_i^{DR})}{(D_i^{RWT})} \tag{6}$$

Server computes the required network bandwidth for N local devices and perform the objective in the following Equation (7).

$$\begin{aligned} &\max_{i=1}^{N} D_i^{RNB} \\ &\text{s.t. } D_i^{RWT} \geq 1, \\ &D_i^{DR} \geq 100, \\ &D_i^{TD} \geq 0 \end{aligned} \tag{7}$$

Multi-objective optimization of these three functions generate the Pareto fronts where devices are arranged in ascending order from best to worst performing devices.

Table 1: Fed-MOODS: an example to illustrate how the multi-objective optimization process works across 5 devices.

Device(s)	PA	MA	AB	PA_d	MA_d	AB_d	$PA_d + MA_d + AB_d$
D_1	97	56	25	3	0	4	7
D_2	99	76	10	4	3	3	10
D_3	82	81	3	2	4	1	7
D_4	56	60	5	0	1	2	3
D_5	70	61	2.5	1	2	0	3

Illustration: We showcase a scenario in which there are 5 heterogeneous mobile devices for smooth understanding. These devices vary in configurations, i.e., different available processing capacities, memory, and bandwidth

[2]`https://bit.ly/ibm-itsm-srv-doc`

of the communication channels. According to Table 1, let 5 devices are: $D = \{D_1, D_2, D_3, D_4, D_5\}$. The server first computes the *available processing capacity (PA)*, *available memory (MA)*, and *available bandwidth (AB)* of each device. Next, the server calculates the *domination count* $\{(PA_d), (MA_d),$ and $(AB_d)\}$ of each device. *Domination count* of a device, D_i signifies how much better the device is in terms of PA, MA, and AB compared to the other participating devices. For example, in Device D_2, PA_d is 4, MA_d is 3, and AB_d is 3, i.e., Device D_2 has maximum available processing capacity. It has more available memory than 3 devices except for D_3 and the available bandwidth is also better than 3 devices except for D_1. We add all the domination counts $(PA_d + MA_d + AB_d)$ and rank each device based on them. The final list contains $\{\{D_2\}, \{D_3, D_1\}, \{D_5, D_4\}\}$. If there is a tie in domination count, we select the device with the highest available processing capacity to break the tie. To estimate the domination counts, we employ a multi-objective optimization approach, NSGA-II [Deb02] to obtain an optimal solution. NSGA-II is an evolutionary multi-objective optimization approach that can optimize three defined objectives efficiently.

3.2.2 Federated learning formulation

Considering a federated learning system consists of N local devices, $D = \{D_1, D_2, \ldots, D_N\}$, and a server. Each device $D_i \in D$ has access to m data samples denoted by $X^{D_i} = \{x_1^{D_i}, x_2^{D_i}, \ldots, x_m^{D_i}\}$, and $X^{D_i} \in \mathbb{R}$. The empirical loss function of device D_i is defined as:

$$\mathcal{L}^{D_i}(w^{D_i}) = \frac{1}{m} \sum_{j=1}^{m} l(w_j, x_j^{D_i})$$

where $l(w, x_j^{D_i})$ is the empirical loss of the model w trained on the j^{th} data sample of the i^{th} device D_i. For any global iteration, suppose the participating devices are n' ($n' \leftarrow \min(n + \langle \tau, k \rangle, N)$, and $n' \in [1, N]$) then the empirical loss for each global round $(\mathcal{L}(w_\tau))$ is defined as:

$$\mathcal{L}(w_\tau) \equiv \mathcal{L}_{n'}(w_{n'}) = \frac{1}{n'} \sum_{i=1}^{n'} \mathcal{L}^{D_i}(w^{D_i})$$

where $\mathcal{L}_{n'}(w_{n'})$ denotes the average empirical loss over n' devices. Number of devices are increasing adaptively in each global round (τ) until the global model converges. The adaptiveness is denoted as k. Here, the objective is to minimize the global loss. The empirical loss for G global rounds is defined as:

$$\mathcal{L}(w^*) = \min_{\tau=1}^{G} \mathcal{L}(w_\tau)$$

3.3 Algorithm

We propose Fed-MOODS, a multi-objective optimization-based adaptive device selection approach for FL that maximizes available processing capacity, memory, and bandwidth among N devices. Based on the three objectives mentioned in Equations (4, 5, and 6), Fed-MOODS ranks devices according to their performance and selects devices adaptively in each global round. We describe Fed-MOODS in two parts, Algorithm 1 for adaptive device selection and learning; and Algorithm 2 for multi-objective optimization based device ranking.

Algorithm 1: It has two phases. *Phase I* (steps 2 to 3) is to rank devices according to the Pareto fronts. *Phase I* is described in detail in Algorithm 2. In *Phase II* (steps 4 to 18), the server adaptively selects local devices from D (step 5) for each global iteration until the model converges. The algorithm is adaptive (steps 5 to 15), i.e., in every global round of learning, devices get an opportunity to contribute to the global model. At first, the algorithm selects the first n devices from the set D and then adaptively adds k devices in each global round (τ) until the model converges. In the worst case, all devices participate in the learning process.

Algorithm 1 Fed-MOODS - Adaptive Device Selection and Training

Input: D ▷ Collect meta-data to compute available processing capacity, memory, and bandwidth from N number of total devices
$X = \{\forall_{i=1}^{N} X^{D_i}\}$
Output: $\mathcal{L}(w^*)$ ▷ The optimal model, and loss function
initialize: $w_\tau = w_0$ ▷ Initialize global model weight

1: **procedure** FED-MOODS(D)
2: Phase 1:
3: $D \leftarrow$ call DEVICERANK(D) ▷ Rank all devices
4: Phase 2:
5: Select first n' devices from D
6: **for** each global iteration $\tau = 1, 2, \dots G$ **do**
7: Broadcast global model w_τ to n' devices
8: **for** each selected devices D_i in parallel **do**
9: **for** each local epoch E **do**
10: **for** batch $b \in X^{D_i}$ and $b \leq m$ **do**. ▷ Data divided in to m batches
11: $w^{D_i} \leftarrow w^{D_i} - \eta l(w_b, x_b^{D_i})$. ▷ Local model at device D_i
12: $\mathcal{L}^{D_i}(w^{D_i}) = \frac{1}{m} \sum_{j=1}^{m} l(w_j, x_j^{D_i})$ ▷ Empirical local loss function at device D_i
13: $w_\tau \leftarrow \frac{1}{n'} \sum_{i=1}^{n'} w_\tau^{D_i}$ ▷ Global model at round τ
14: $\mathcal{L}(w_\tau) \equiv \frac{1}{n'} \sum_{i=1}^{n'} \mathcal{L}^{D_i}(w^{D_i})$ ▷ Empirical loss at global round τ
15: $n' \leftarrow \min(n + \langle \tau, k \rangle, N)$ ▷ k devices are added from the Pareto fronts in every global iteration.
16: $w^* = \min_w \{\mathcal{L}(w) \equiv \sum_{\tau=1}^{G} w_\tau \mathcal{L}(w_\tau)\}$ ▷ Optimal global model
17: $\mathcal{L}(w^*) = \min_{\tau=1}^{G} \mathcal{L}(w_\tau)$ ▷ $\mathcal{L}(w^*)$ is the minimum global empirical loss among τ global models.
18: return $\mathcal{L}(w^*)$

Algorithm 2: The server initially collects meta-data from devices regarding the processing capacity, memory, and bandwidth, respectively. Then calculate D_i^{PA}, D_i^{AMR}, and D_i^{RNB} for each device ($i \in N$)(steps 3 to 5). Later, we employ NSGA-II to find the domination count of the devices and rank them accordingly to their Pareto fronts (step 6). Finally, the server generates a list of devices, D', based on their ranks (steps 7 to 9) and returns to the Fed-MOODS (step 10).

Algorithm 2 DeviceRank - Algorithm for Ranking Devices

Input: $D = \{\forall_{i=1}^{N} D_i < D_g, D_c, D_i^{TM}, D_i^{MR}, D_i^{TD}, D_i^{DR}, D_i^{RWT} >\}$ ▷ Server collects meta-data to compute available processing capacity, memory, and bandwidth from N number of total devices
Output: $\mathcal{D}'$ ▷ List of devices according to the maximum to minimum domination count.

1: **procedure** DeviceRank(D)
2: **for** i =1 to N **do**
3: Compute $D_i^{PA}(D_g, D_c)$ ▷ Compute available processing capacity of the i^{th} device.
4: Compute $D_i^{AMR}(D_i^{TM}, D_i^{MR})$ ▷ Compute available memory of the i^{th} device.
5: Compute $D_i^{RNB}(D_i^{TD}, D_i^{DR}, D_i^{RWT})$ ▷ Compute availble bandwidth of the i^{th} device.
6: Compute $\forall_{i=1}^{N}\ Dom(D_i(D_i^{PA}, D_i^{AMR}, D_i^{RNB}))$ ▷ Compute domination count of every devices using NSGA-II. and rank them according to the Pareto fronts
7: **for** i = 1 to N **do**
8: Select the device D_i successively from the Pareto fronts.
9: $D' = D' \cup D_i$ ▷ List of devices according o their Pareto fronts
10: Return D'

3.4 Computational time analysis

We characterize and compare the computational run-time of Fed-MOODS with random participation of devices as a baseline. Suppose the computation time of the N available devices are $\{T_{Cl_1}, T_{Cl_2}, \ldots T_{Cl_N}\}$. In Algorithm 1, each local device performs E local iterations and G global rounds until convergence. n is the initial set of devices, k is the adaptiveness factor. System heterogeneity causes different computation time for each device. Therefore, server waits until the slowest device responds. The computational run-time of $T_{Fed-MOODS}$ is defined below.

Definition 1. *For constants N, n, k, G, and E, the time required to train global model is* $\mathcal{O}(G * E * (T_n + T_{n+k} + \ldots + T_{n+\langle\tau,k\rangle}))$*, where* $0 \leq \tau \leq G$.

$T_{n+\langle\tau,k\rangle}$ is the maximum unit computation time of the slowest device at the τ^{th} global round. $T_{n+\langle\tau,k\rangle}$ can be defined as, $T_{n+\langle\tau,k\rangle} = \max_{j=1}^{n+\langle\tau,k\rangle} T_{Cl_j}$. From the Figure 5, we can observe that the computational run-time is exponential in nature, what we represent as e^{λ}, where λ is a constant. The average run-time of $T_{Fed-MOODS}$ can be written as, $\bar{T}_{Fed-MOODS} = \frac{1}{G}\sum_{\tau=1}^{G}(\max_{j=1}^{n\ +\ \langle\tau,k\rangle} T_{Cl_j}) \approx \mathcal{O}(e^{\lambda})$.
For the same settings, random device participation takes G' global rounds to converge. The computational run-time T_{Random} as the baseline is defined below.

Definition 2. *For constants N, G', and E, the time required to train global model is* $\mathcal{O}(G' * E * (T_1 + T_2 + \ldots + T_{G'}))$.

$T_{G'}$ is the maximum unit computation time of the slowest device (select n' from N) in random selection for the G^{th} global round. Similarly, from the Figure 5, we can observe that the computational run-time for T_{Random} is exponential in nature. Therefore, the average run-time for learning by randomly selecting devices is $\bar{T}_{Random} = \frac{1}{G'} \sum_{\tau=1}^{G'} (\max_{j=1}^{n'} T_{Cl_j}) \approx \mathcal{O}(e^{\lambda'})$, where λ' is a constant. In the worst case, $\lambda = \lambda'$, then $\bar{T}_{Fed-MOODS} = \bar{T}_{Random}$, otherwise, $\lambda < \lambda'$, and $\bar{T}_{Fed-MOODS} < \bar{T}_{Random}$. Experimentally, we have shown $T_{Fed-MOODS} \leq T_{Random}$ and $\bar{T}_{Fed-MOODS} \leq \bar{T}_{Random}$ in Section 4.7. To support our analysis of computational run-time for $T_{Fed-MOODS}$ when compares with T_{random} as baseline, we prove a lemma below.

Lemma 1. *The average run-time of Fed-MOODS ($\bar{T}_{Fed-MOODS}$) is always less than or equal to the baseline federated learning with random device selection ($\bar{T}_{Random}$) iff $\lambda \leq \lambda'$.*

Proof 1. *Let $e^{\lambda'} < e^{\lambda}$ as estimated run-time for T_{random} and $T_{Fed-MOODS}$, respectively. According to the Definition 1 and 2, the following conditions $\lambda \leq \lambda'$ and $\lambda : \lambda' \leq 1$ always hold. Therefore, the assumption is false. Hence, proved by contradiction.* □

4 Experiments and Analysis

4.1 Simulation setup

We simulate a FL environment in our local machine to reflect the effect of stragglers. In order to model the device heterogeneity, we incorporated two different approaches to validate Fed-MOODS. At first, we employ different local rounds to different devices to incorporate system heterogeneity as we simulated the federated network in a computer. We allowed a maximum of 10 local epochs to a non-straggler device and less than 10 local epochs to stragglers. Assuming that, for a synchronous federated learning, a straggler will perform less epoch than the non-straggler device. The simulation setup is given in Table 2. We compared the performance of Fed-MOODS with the randomly selected devices. Secondly, to measure the wall clock run-time of Fed-MOODS, we assumed the run-time of non-straggler devices is in the range of 10^2 ms to 10^3 ms, and for stragglers, it is in the range of 10^3 ms to 10^4 ms. All devices complete a fixed set of local rounds in each global round. For validation of run-time, we only used FedAvg as a baseline method. We used the early stopping mechanism to terminate the learning process if there is no improvement in loss function for 10 consecutive rounds.

Table 2: Simulation setup: parameters, values, and their description

Parameter(s)	Value	Description
local devices	100	Devices for a local update of the model
Server	1	For performing multi-objective optimization, model aggregation
Federated algorithm	2	FedAvg[Mit21], FedProx[Li 20]
local device's participation	Adaptive and random	Adaptive participation of devices for Fed-MOODS, by random, frequency of participation is 10%
Dataset	IID and non-IID	IID and non-IID division of MNIST, CIFAR-10, and FMNIST dataset
Local iteration	Maximum 10	Number of local iteration at each device for each global iteration.
Global iteration	Maximum 100, 500	100 global iterations for learning on MNIST and FMNIST dataset. 500 global iterations for learning model on CIFAR-10 datasets.
Presence of stragglers	10%, 50%, 70%, 90%	Presence of stragglers in each global iteration for different experiments.
Training network	3	Three Convolutional Neural Network (CNN) having two hidden layers for training on MNIST, CIFAR-10, and FMNIST datasets, respectively.
Optimizer	1	Stochastic Gradient Descent (SGD)
Performance metrics	2	Test accuracy, F1-score

4.2 Datasets and networks

We used three benchmark datasets, MNIST [Den12] (60,000 samples for training and validation, and 10,000 testing samples), CIFAR-10 [Kri09] (50,000 samples for training and validation, and 10,000 testing samples), and FMNIST [al17] (60,000 samples for training and validation, and 10,000 testing samples) to validate Fed-MOODS. All datasets are distributed to devices in IID and non-IID manner.

To implement Fed-MOODS, we created a federated network consisting of 100 heterogeneous devices. Each device trains a 2 layered convolutional neural network (CNN). The details of the neural network is given in the Table 3.

Table 3: Neural network architecture

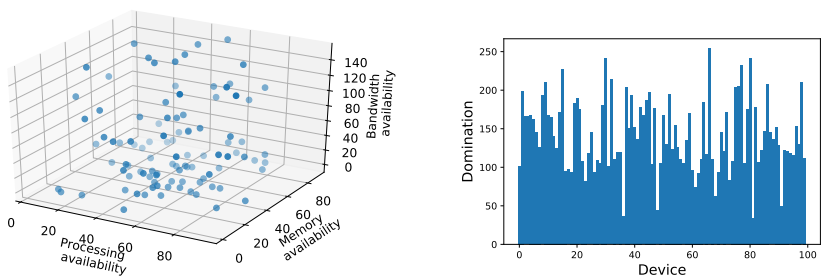
Neural Network	Number of Convolutional layer	In channel	Out channel	Kernel size	Number of Fully connected layer	In features	Out features	Activation function
CNNMnist	2	1	10	5	2	320	50	softmax
		10	20			50	10	
CNNFMnist	2	1	6	5	4	192	120	ReLU
						120	60	
		6	12			60	40	
						40	10	
CNNCifar10	2	3	6	5	3	400	120	softmax
		6	16			120	84	
						84	10	

4.3 Baseline algorithms

We consider FedAvg [McM17] and FedProx [Li 20] as federated algorithms integrated with Fed-MOODS to compare the performance of adaptive device selection with random partial participation of devices. We evaluated the performance of Fed-MOODS with these baselines, both with respect to global rounds and wall clock time simulations.

4.4 Rank devices based on NSGA-II

In *Phase-I* of the Fed-MOODS, we attempt to maximize three objective functions (see Fig. 2(a)) mentioned in subsection 3.2.1 to characterize each device and obtain the rank of devices (see Fig. 2(b)) based on their domination counts. A device with the highest domination count is the strongest device. Similarly, a device with the lowest domination count is the weakest device concerning its system heterogeneity.



(a) (b)

Figure 2: (a) Device characteristics - devices based on the three objective functions: available processing capacity, memory, and bandwidth. (b) Devices with the domination counts.

4.5 Comparison with random device participation

4.5.1 Convergence comparison

We verified the convergence of Fed-MOODS integrated with FedAvg[McM17] and FedProx[Li 20] separately with random device selection (selecting 10% of the total devices in each global round) in the presence of different fractions of stragglers (10%, 50%, 70%, and 90%) in Fig. 3 (left to right). The convergence curves are similar for the IID datasets (see Fig. 3(a), 3(c), and 3(e)). Fed-MOODS converges quickly; therefore, it maintains the model's fairness without involving all stragglers in the learning process. But for non-IID (see Fig. 3(b), 3(d), 3(f)), Fed-MOODS takes more global rounds to converge, but

it is faster than random device selection. The convergence curves are more stable compared to learning with random device selection.

4.5.2 Fairness in terms of performance

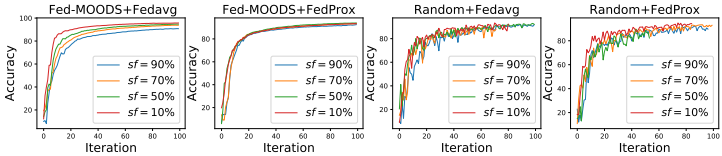
We compared the performance (see Table 4) of Fed-MOODS with baseline models with partial device selection based on the F1-score at a frequency of 90% stragglers in both IID and non-IID settings.

We compared the performance of Fed-MOODS both involving stragglers and without involving stragglers for the IID data. Here, without incorporating stragglers imply that we first divide data into 100 devices and then remove stragglers. We only kept 10 non-straggler devices for learning. We observed that Fed-MOODS gives a 97.6% F1-score for the MNIST IID dataset even if we do not incorporate stragglers. Similarly, for CIFAR-10 (51.79%) and FMNIST (78.16%), the performance is almost equivalent to the models incorporating stragglers. Even though we omit 90% of the devices, Fed-MOODS still maintains model fairness by giving an equivalent performance with baselines.

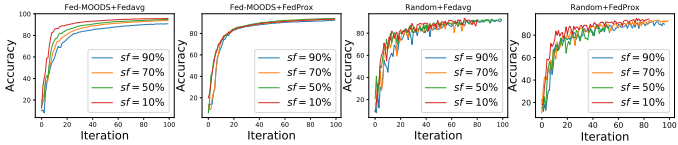
For non-IID division of data, the performance of Fed-MOODS (94.27% for MNIST, 49.33% for CIFAR-10, and 70% for FMNIST) is also better than the baseline models with randomly selected devices ((93% for MNIST, 9.37% for CIFAR-10, and 50% for FMNIST) at 90% straggler frequency. Even with high frequency of stragglers, Fed-MOODS can often achieve maximum performance. It also maintains fairness, as Fed-MOODS allows every device to contribute. Here, the performances of the models on CIFAR-10 and FMNIST are deficient because we used a simple 2-layer CNN for training and used the early stopping mechanism to terminate. However, since the main purpose of the experiment is to examine the behaviour of Fed-MOODS and the baselines regarding stragglers, the simple architecture suits the needs as well.

4.5.3 Fairness of probability of devices' appearance (PoA)

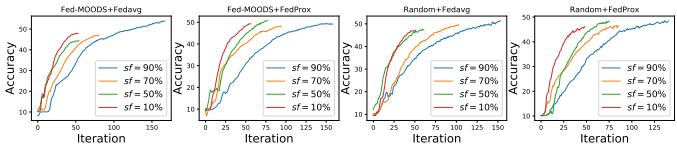
We assumed that N heterogeneous devices are available in the FL system, and all participate in learning. For random selection, if we select n' devices randomly from N devices in each global round, then the PoA of a device $p(D_i)$ for the G global rounds is $(1 - (\frac{N-1}{N})^{n'})^G$. Even though the straggler devices are present, random selection gives an equal PoA to each device. Fed-MOODS is biased toward non-straggler devices and does not assign equal PoA to every device. According to Algorithm 1, the PoA of a device in training rounds is $0 \leq P(D_i) \leq 1$, where $P(D_i) = \frac{\bar{G}}{G}$. Here, $\bar{G}$ is the number of appearances of a device (D_i) in total global rounds, and G is the total global rounds. Fed-MOODS adaptively incorporates devices in each training round, so that the PoA of a non-straggler device is always greater than the PoA of a straggler. For example, the first n devices appear in every global round ($\bar{G} = G$). Therefore, the PoA $= 1$ for the first n devices. Fed-MOODS is adding k devices adaptively in each round, so that the following k devices appear in $G-1$ global



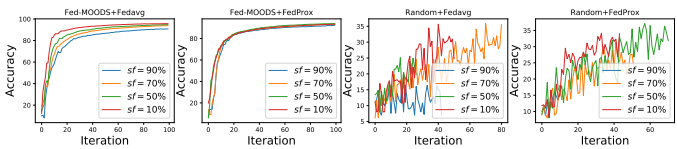
(a)



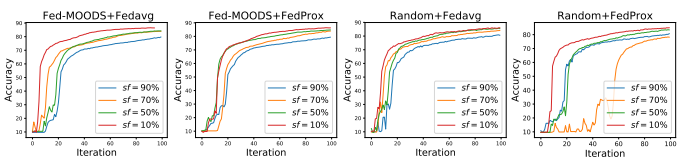
(b)



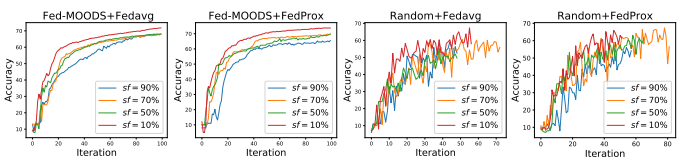
(c)



(d)



(e)



(f)

Figure 3: Convergence comparison of Fed-MOODS and baseline models with random device participation across (a) MNIST-IID, (b) MNIST-non-IID, (c) CIFAR-10 IID, (d) CIFAR-10 non-IID, (e) FMNIST IID, (f) FMNIST non-IID, with different straggler fractions (sf).

Table 4: Performance (F1-Score) comparison between Fed-MOODS and baseline models with random device participation in presence of 90% stragglers. ♥ and ♢ denote *involving stragglers* and *without involving stragglers*, respectively.

Dataset	Fed-MOODS + FedAvg	Fed-MOODS + FedProx	Random device selection + FedAvg ♥	Random device selection + FedAvg ♢	Fed-MOODS + + FedAvg ♢
MNIST IID	94.7	93.5	94.00	96.28	**97.00**
CIFAR-10 IID	48.65	**52.92**	49.51	49.67	51.79
FMNIST IID	78.66	78.48	80.48	79.01	78.19
MNIST non-IID	93.41	**94.27**	93.00	NA	NA
CIFAR-10 non-IID	**49.33**	48.79	9.37	NA	NA
FMNIST non-IID	63.12	**65**	50.25	NA	NA

rounds. Therefore, PoA $= 1 - \frac{1}{G}$. Similarly, the devices added in the $(G-1)^{th}$ global round, the PoA will be $\frac{1}{G}$. If $N = n + \langle G, k \rangle$, i.e., for G^{th} global round, all devices are participating in learning. If converging round $G^* > G$, then for the $G^* - G$ rounds, all devices participate in training. As Fed-MOODS considers the participation of every device until convergence (G^*), every device will get a chance to contribute its information to maintain high statistical accuracy. Therefore, fairness in device selection is maintained here in the presence of stragglers. If $n + \langle G^*, k \rangle < N$, i.e., partial participation of devices can produce an equivalent model performance to that of total involvement of devices. Therefore, Fed-MOODS is straggler-resilient as well as maintains fairness.

4.6 Test accuracy

In Table 5, we compared the test accuracy of the models for a different fraction of stragglers. Fed-MOODS produces similar results with a random selection of devices for IID datasets. Accordingly, in non-IID settings, Fed-MOODS outperforms the random selection approach by a maximum of 1.88% for MNIST, 34% for CIFAR-10, and 15% for FMNIST datasets, respectively.

4.7 Wall-clock time comparison

We compared the wall clock learning time of a neural network model using Fed-MOODS and baseline models on the MNIST and FMNIST datasets. We compared two cases (See Fig. 4 and 5) where the straggler frequency is very high (90%), and the straggler frequency is low (10%). From Fig. 4 we see that Fed-MOODS gradually involve straggler devices in each global round. Whereas in random partial device participation, the effect of randomness is

Table 5: Comparison of models among test accuracy

Dataset		SF %	Fed-MOODS + FedAvg	Fed-MOODS + FedProx	Random + FedAvg	Random + FedProx
MNIST	IID	90	**97.2**	96.31	97.2	96.89
		70	97.54	97.49	97.61	97.5
		50	**97.94**	97.76	97.74	97.61
		10	98.11	**98.39**	98.05	98.11
	Non-IID	90	92.31	91.93	92.04	91.47
		70	**93.91**	92.79	89.18	93.43
		50	**94.69**	93.47	93.05	89.61
		10	**95.74**	93.59	93.17	93.86
CIFAR-10	IID	90	**53.43**	50.20	49.15	48.86
		70	46.3	47.15	48.62	47.17
		50	43.59	**49.42**	46.25	48.9
		10	46.71	**47.33**	45.48	44.72
	Non-IID	90	49.23	**49.55**	15.84	10
		70	**48.75**	47.68	33.99	29.75
		50	**46.56**	45.93	24.98	38.44
		10	45.86	**47.81**	33.75	34.0
FMNIST	IID	90	78.66	78.48	80.48	79.44
		70	82.63	82.81	83.04	77.63
		50	83.32	83.89	85.17	82.59
		10	**85.39**	85.22	84.44	84.68
	Non-IID	90	63.22	**65.33**	50.26	58.18
		70	**67.16**	65.54	56.92	64.07
		50	70.0	**70.97**	55.56	61.81
		10	**71.76**	67.58	58.18	59.26

clearly visible. From table 6, to complete 100 global rounds, Fed-MOODS is 1.8× and 1.48× faster than the baseline model (FedAvg) with random device participation on the MNIST and FMNIST non-IID dataset, respectively. In Fig. 5, we measured validation loss with wall clock time to train a federated model. We observed that Fed-MOODS takes less time to converge than any baseline model with random device participation for the MNIST and FMNIST non-IID datasets.

4.8 Effect of adaptiveness

In Fig. 6, we compared the convergence of Fed-MOODS by adapting devices from the Pareto front in each round in the presence of 90% stragglers for the

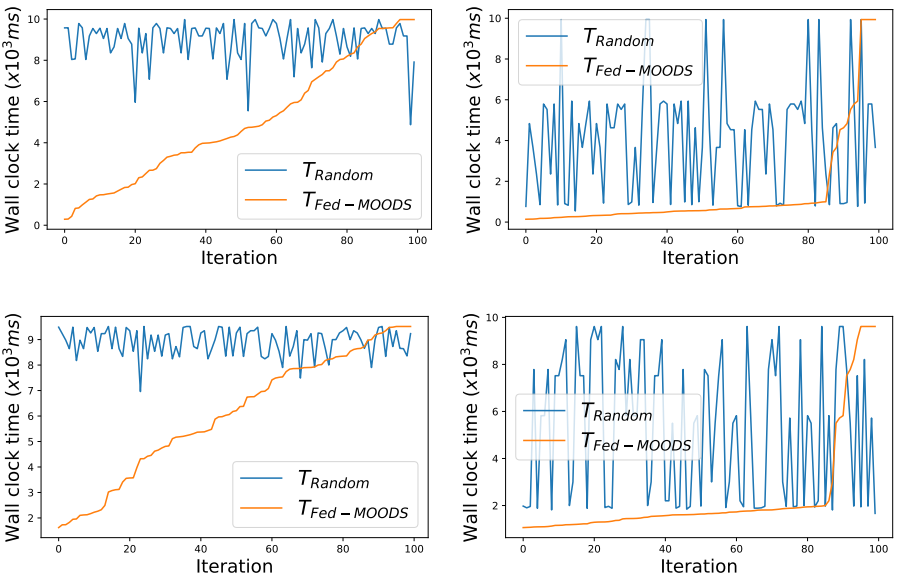


Figure 4: Compare wall-clock time vs global iterations between Fed-MOODS and baseline model with random device participation in the presence of 90% (left top and bottom) and 10% (right top and bottom) stragglers on MNIST-nonIID (top) and FMNIST-nonIID (bottom) datasets, respectively.

Table 6: Total and average wall clock time comparison between Fed-MOODS and baseline model with random device selection at presence of 90% stragglers on non-IID data.

Datasets	Random Device selection		Fed-MOODS	
	$T_{Random}(ms)$	$\bar{T}_{Random}(ms)$	$T_{Fed-MOODS}(ms)$	$\bar{T}_{Fed-MOODS}(ms)$
MNIST	9×10^5	9×10^3	$\mathbf{4.9 \times 10^5}$	$\mathbf{4.9 \times 10^3}$
FMNIST	8.9×10^5	8.9×10^3	$\mathbf{6 \times 10^5}$	$\mathbf{6 \times 10^3}$

MNIST, FMNIST, and CIFAR-10 non-IID datasets, respectively. We observed a significant increase performance in adaptiveness, making the model converge quickly, but it also incorporates more stragglers in the learning process.

5 Conclusion and Future Work

In this work, we proposed Fed-MOODS, a multi-objective optimization-based adaptive device selection approach to minimize the effect of stragglers in federated learning. We formulated every device's available processing capacity,

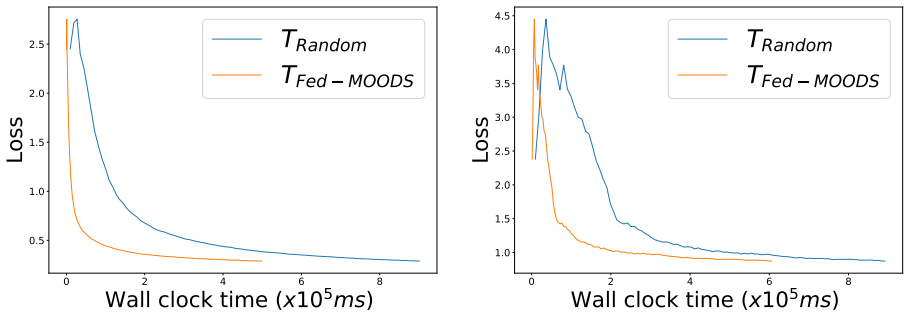


Figure 5: Training loss vs wall-clock time comparison of Fed-MOODS and baseline model with random device participation in presence of 90% stragglers on MNIST non-IID (left) and FMNIST non-IID (right) datasets, respectively.

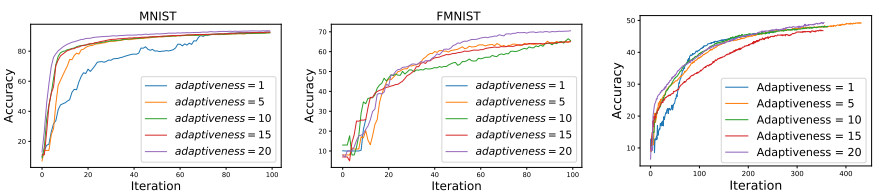


Figure 6: Convergence comparison of different adaptiveness level in presence of 90% stragglers on MNIST non-IID dataset (top left), FMNIST non-IID (middle) and, CIFAR-10 non-IID (right) dataset, respectively.

memory, and bandwidth as a multi-objective optimization problem. We generate the rank of devices from the Pareto fronts by solving the multi-objective functions. The algorithm adaptively selects devices for training according to their ranking. We verified the Fed-MOODS on three baseline datasets (MNIST, CIFAR-10, and FMNIST), considering both IID and non-IID divisions of data among 100 devices. Fed-MOODS is straggler-resilient with the ability to maintain the model's fairness and reduce overall training time by 1.8× and 1.48× faster than the baseline model (FedAvg) with random device participation on the MNIST and FMNIST non-IID dataset, respectively. Our work suggests several exciting directions, including the theoretical convergence analysis of Fed-MOODS, and understanding the trade-off between fairness and robustness issues in device selection in scalable FL.

Acknowledgments

This work was partially supported by the Wallenberg AI, Autonomous Systems and Software Program (WASP) funded by the Knut and Alice Wallenberg

Foundation.

References

[al17] H. Xiao et. al. “Fashion-MNIST: a Novel Image Dataset for Benchmarking Machine Learning Algorithms”. In: *arXiv:cs.LG/1708.07747* (2017).

[Ban20] S. Banerjee et. al. “Multi-diseases Classification from Chest-X-ray: A Federated Deep Learning Approach”. In: *Australasian Joint Conference on Artificial Intelligence* (2020), pp. 3–15.

[Ban21] S. Banerjee et. al. “Fed-FiS: a Novel Information-Theoretic Federated Feature Selection for Learning Stability”. In: *International Conference on Neural Information Processing* (2021), pp. 480–487.

[Can21] G. Canonaco et. al. “Adaptive Federated Learning in Presence of Concept Drift”. In: *2021 International Joint Conference on Neural Networks (IJCNN)* (2021), pp. 1–7.

[Cui21] S. Cui et. al. “Addressing Algorithmic Disparity and Performance Inconsistency in Federated Learning”. In: *Advances in Neural Information Processing Systems* 34 (2021).

[Deb02] K. Deb et. al. “A fast and elitist multiobjective genetic algorithm: NSGA-II”. In: *IEEE transactions on evolutionary computation* 6.2 (2002), pp. 182–197.

[Den12] Li. Deng. “The mnist database of handwritten digit images for machine learning research [best of the web]”. In: *IEEE signal processing magazine* 29.6 (2012), pp. 141–142.

[Hu 20] Z. Hu et. al. “Fedmgda+: Federated learning meets multi-objective optimization”. In: *arXiv:2006.11489* (2020).

[Hua20] T. 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 (2020), pp. 1552–1564.

[Kai21] P. Kairouz et. al. “Advances and Open Problems in Federated Learning”. In: *Foundations and Trends® in Machine Learning* 14.1–2 (2021), pp. 1–210.

[Kri09] A. Krizhevsky et. al. “Learning multiple layers of features from tiny images”. In: *Citeseer* (2009).

[Li 19a] X. Li et. al. “On the convergence of fedavg on non-iid data”. In: *arXiv:1907.02189* (2019).

[Li 19b] T. Li et. al. “Fair resource allocation in federated learning”. In: *arXiv preprint arXiv:1905.10497* (2019).

[Li 20] T. Li et. al. “Federated optimization in heterogeneous networks”. In: *Proceedings of Machine Learning and Systems* 2 (2020), pp. 429–450.

[Li 21] Li. Li et. al. “FedSAE: A novel self-adaptive federated learning framework in heterogeneous systems”. In: *International Joint Conference on Neural Networks (IJCNN)* (2021), pp. 1–10.

[Lia18] X. Lian et. al. “Asynchronous Decentralized Parallel Stochastic Gradient Descent”. In: *International Conference on Machine Learning* (2018), pp. 3043–3052.

[Lyu20] L. Lyu et. al. “Collaborative fairness in federated learning”. In: *Federated Learning, Springer* (2020), pp. 189–204.

[McM17] B. McMahan et. al. “Communication-Efficient Learning of Deep Networks from Decentralized Data”. In: *Artificial intelligence and statistics* (2017), pp. 1273–1282.

[Mit21] A. Mitra et. al. “Achieving Linear Convergence in Federated Learning under Objective and Systems Heterogeneity”. In: *arXiv preprint arXiv:2102.07053* (2021).

[Moh19] M. Mohri et. al. “Agnostic federated learning”. In: *International Conference on Machine Learning* (2019), pp. 4615–4625.

[Rei20] A. Reisizadeh et. al. “Straggler-Resilient Federated Learning: Leveraging the Interplay Between Statistical Accuracy and System Heterogeneity”. In: *arXiv:2012.14453* (2020).

[Shi21a] H. Shi et. al. “Towards Federated Learning with Attention Transfer to Mitigate System and Data Heterogeneity of Clients”. In: *proceedings. of the 4th Int. Workshop on Edge Sys., Analytics and Networking* (2021), pp. 61–66.

[Shi21b] Y. Shi et. al. “A Survey of Fairness-Aware Federated Learning”. In: *arXiv preprint arXiv:2111.01872* (2021).

[Smi17] V. Smith et. al. “Federated Multi-task Learning”. In: *Advances in neural information processing systems* 30 (2017).

[Sti18] S. Stich. “Local SGD Converges Fast and Communicates Little”. In: *arXiv:1805.09767* (2018).

[Wan21] Y Wang et.al. “Accelerated Training via Device Similarity in Federated Learning”. In: *”Proceedings of the 4th International Workshop on Edge Systems, Analytics and Networking”* (2021), pp. 31–36.

[Xie19] C. Xie et. al. “Asynchronous Federated Optimization”. In: *arXiv:1903.03934* (2019).

[Yad16] A Yadin. *Computer Systems Architecture*. CRC Press, 2016.

[Yan21] C. Yang et. al. “Characterizing Impacts of Heterogeneity in Federated Learning upon Large-Scale Smartphone Data”. In: *Proceedings of the Web Conference* (2021), pp. 935–946.

[Zho21] P. Zhou et. al. “Loss Tolerant Federated Learning”. In: *arXiv preprint arXiv:2105.03591* (2021).

Paper III

Cost-Efficient Feature Selection for Horizontal Federated Learning

Sourasekhar Banerjee, Devvjiit Bhuyan, Erik Elmroth, and Monowar Bhuyan

IEEE Transactions on Artificial Intelligence (TAI), IEEE, doi: 10.1109/TAI.2024.3436664, 2024.

Cost-Efficient Feature Selection for Horizontal Federated Learning*

Sourasekhar Banerjee[*], Devvjiit Bhuyan[†], Erik Elmroth[*], Monowar Bhuyan[*]

[*] *Department of Computing Science, Umeå University, Umeå, Sweden*

[†] *Department of Electronics and Communication Engineering, Tezpur University, Assam, India*

sourasb@cs.umu.se, ecb19050@tezu.ac.in, elmroth@cs.umu.se, monowar@cs.umu.se

Abstract: Horizontal Federated Learning exhibits substantial similarities in feature space across distinct clients. However, not all features contribute significantly to the training of the global model. Moreover, the curse of dimensionality delays the training. Therefore, reducing irrelevant and redundant features from the feature space makes training faster and inexpensive. This work aims to identify the common feature subset from the clients in federated settings. We introduce a hybrid approach called Fed-MOFS[1], utilizing Mutual Information and Clustering for local feature selection at each client. Unlike the Fed-FiS, which uses a scoring function for global feature ranking, Fed-MOFS employs multi-objective optimization to prioritize features based on their higher relevance and lower redundancy. This paper compares the performance of Fed-MOFS [2] with conventional and federated feature selection methods. Moreover, we tested the scalability, stability, and efficacy of both Fed-FiS and Fed-MOFS across diverse datasets. We also assessed how feature selection influenced model convergence and explored its impact in scenarios with data heterogeneity. Our results show that Fed-MOFS enhances global model performance with a 50% reduction in feature space and is at least twice as fast as the FSHFL method. The computational complexity for both approaches is O(d^2), which is lower than the state-of-the-art.

Key words: Horizontal Federated Learning, Feature Selection, Mutual Information, Clustering, Statistical Heterogeneity, Multi-objective Optimization

*The paper has been re-typeset to match the thesis style. Reproduced with permission of IEEE.

[1]This manuscript is an extension of Banerjee et al. [BEB21]

[2]We share our code and data through `https://github.com/DevBhuyan/Horz-FL/blob/main/README.md`.

1 Introduction

Federated Learning (FL) is a novel machine learning paradigm that facilitates collaborative model training among multiple data owners (a.k.a. clients) [BVB22; McM+17]. This occurs through the iterative exchange of model parameters via an FL server, all while maintaining the privacy of individual clients without sharing local data. According to the intersection or distribution of data among clients in terms of sample space or feature space, federated learning can be classified into three main categories: Horizontal Federated Learning (HFL), Vertical Federated Learning (VFL), and Federated Transfer Learning (FTL) [Yan+19]. HFL operates with data, sharing a uniform feature space across all clients. Whereas, VFL leverages dissimilar data with distinct feature spaces to train a global model collaboratively. Conversely, FTL uses a pre-trained model initially trained on similar data to solve different problems. HFL scenarios frequently occur in practical use cases, such as in internet-of-vehicles [Ham+22], smart grid analysis [Ren+23], e-commerce [Li+21], health-care [Ban+20], etc. The efficacy of local models is influenced by the quality of local features possessed by clients, consequently impacting the overall performance of the global model. Clients may have irrelevant or noisy features for the learning task, or they may have an excessive number of redundant features, leading to a significant degradation in the performance of the global model. Therefore, the absence of feature selection results in poor model performance and extends the duration of model training. Therefore, it is crucial to identify appropriate feature sets that overlap among the clients, as this can lead to reduced training time and energy consumption, which results in reduced communication rounds without compromising the global model's performance.

Feature Selection (FS) is a crucial preprocessing technique in a centralized Machine Learning (ML) framework. It has been extensively studied and proven valuable in data mining, knowledge discovery, and ML. The primary goal of FS is to identify the most pertinent, trustworthy, and non-redundant features from extensive datasets. This process enhances model performance and facilitates cost-effective learning of models. The choice of features is also very important when working with high-dimensional datasets, where the number of features is much higher than the number of samples. This makes it hard to find the right features for making cost-effective learning of models. In the context of deep learning, identifying meaningful features within a data set is significant. Representation learning algorithms extract valuable patterns from raw data, generating representations that simplify processing. These representations can be crafted for interpretability, to reveal hidden features, or utilized in transfer learning applications. In contrast to dimension reduction techniques such as principal component analysis (PCA) [Pea01], feature selection does not modify the original features. Instead, it identifies and chooses a subset of the most valuable features from the dataset during runtime. The feature selection process involves evaluating the importance of each feature using various methods, such as correlation [Hal99], mutual information [HBK14], chi-square [Aha+23], etc. After identifying crucial features, they are employed to construct precise, effective, and budget-friendly ML models for prediction or classification. Additionally, feature selection enhances interpretability by pinpointing the key variables influencing the model's predictions. Information-theoretic measures

have been widely utilized and established as a paradigm for filter-based feature selection. Specifically, Mutual Information-based Feature Selection (MIFS) empowers the feature selection method by removing redundant and irrelevant features without impacting the classifier's reachable performance. Traditional MIFS approaches [HBK14; AAB11; KSG04], such as MIFS-ND are designed for centralized systems where data is available in a centralized server.

Why does feature selection benefit in federated learning? When each client independently performs feature selection and builds individual local models based on their private data without exchanging the selected feature set with the server, it impacts the global model updates and performance. This results in objective shifts by cause of feature selection bias [KŁ16] and statistical heterogeneity among clients. Furthermore, if clients opt for distinct feature subsets, it poses challenges in ensuring homogeneous model training across all clients in HFL settings. These issues inspired us to develop a feature selection algorithm tailored to the context of HFL.

In our previous work, Fed-FiS [BEB21] evaluates the importance of the features by a score function and generates global ranks of each feature from higher to lower scores. Here, we introduce Fed-MOFS, a new federated feature ranking and selection method based on multi-objective optimization. The resulting method optimizes the relevance and redundancy of the feature set simultaneously. It produces a non-dominated solution set or Pareto fronts from where we get the ranking of the features. The local feature selection of Fed-FiS and Fed-MOFS remain the same, but the global feature selection and ranking differ.

Why does global feature ranking of individual features affect FL performance? All clients share a common set of features, but the importance of these features varies among clients. Determining a unified and relevant set of features across all clients is challenging, as the features selected locally by each client may differ. For instance, in Table 3, feature f_4 is crucial for clients Cl_2 and Cl_3 but not for Cl_1. If we choose features that are locally selected by all clients in common, we must omit f_4. In the worst-case, there might be no overlap in locally selected features among clients, making it difficult to train a global model. To address this challenge, we consider two approaches: Fed-FiS (refer to Table 4) and Fed-MOFS (refer to Table 5). These approaches provide global rankings for each feature. For example, f_4 obtains global rankings of 3, i.e., the third important feature using Fed-FiS and 4, which means the fourth important feature using Fed-MOFS. Thus, calculating the global ranking for each feature helps to achieve a fair assessment, ensuring a balanced representation of the importance of the feature in the training process.

The main contributions of this work are summarized as follows.

- We introduce Fed-MOFS, an approach for global ranking and feature selection based on multi-objective optimization for horizontal federated learning. It employs mutual information and 1-D clustering to choose a local set of features. Furthermore, it utilizes Pareto optimization to create a global ranking of these features and selects features from the Pareto fronts (see Section 4.3).
- We derived the computational complexity of both the Fed-FiS and Fed-MOFS algorithms (see Section 4.4) and compared them with the current state-of-the-art

horizontal federated learning algorithms (see Table 1) that explicitly demonstrate the computational cost, benefits and drawbacks.

- We conducted a comprehensive empirical study comparing the performance, scalability, efficiency, stability, and convergence of both Fed-FiS and Fed-MOFS across various datasets, including NSL-KDD99, Wine, Vowel, Vehicle, Segmentation, WDBC, Ionosphere, Hill-Valley, ISOLET, Diabetes, IoT, Anonymized Credit Card (ACC), Boston housing prices, California house pricing, and synthetic data. This analysis utilized conventional feature selection techniques within federated settings, such as RFE and ANOVA, as well as federated feature selection methods like FSHFL [Zha+23] and Fed-mRMR [HBL24]. For classification tasks, we trained Federated Forest [Liu20] and federated averaging [McM+17] on Deep Neural Networks (DNNs), and for regression tasks, we employed ridge regression, all following the application of various feature selection methods in federated settings (see Section 5).

2 Related Work

This section provides a comprehensive overview of three key related research areas: centralized feature selection, distributed feature selection, and horizontal federated feature selection.

2.1 Centralized feature selection

In centralized settings, computation happens on a single system. Data is collected and resides in a centralized server; hence, it is easy to access and build models for a specific task. Feature selection is comparatively easy as the server has complete information about the data. Regarding the availability of class information, feature selection methods are categorized as supervised [Son+07], semi-supervised [FDA22], and unsupervised [SCM20] approaches. Also, depending on how feature selection methods interact with the classifier, the existing works are categorized as filter [GE03], wrapper [BL97], embedded [LZL19], and hybrid [HHL11]. **Filter** methods are statistical approaches that select features based on their relevance to the class without considering the employed model. Examples include correlation-based [Hal99], mutual information-based [HBK14], and chi-squared [Aha+23] based feature selection methods. **Wrapper** methods involve training a model using a subset of features and evaluating the performance of the model [EB16; ALA16]. Examples of wrapper methods include forward selection, backward elimination, and recursive feature elimination. **Embedded** methods incorporate feature selection within the model-building process. For example, lasso regression [MR16], ridge regression [Zha+18], and elastic net [AH21]. **Hybrid** methods combine multiple feature selection techniques, such as filter and wrapper, to improve the model's overall performance.

The feature selection helps to describe data better for extracting valuable knowledge from high dimensional data [AAB11] and solve problems that occur due to the higher

dimensionality of data. This reduces the model's computation cost by reducing feature space and improves overall learning performance by selecting relevant and less redundant features. Mutual Information (MI) based feature selection is a filter method that chooses strongly correlated features with labels and has minimal redundancy among feature sets. The MI method adopts the entropy difference to calculate the information a feature contributes. MIM [Lew92] rapidly selects label-related features. However, the consideration of feature redundancy is absent. MIFS [Bat94] proposes feature redundancy as a metric for assessing the quality of features. Methods in [LT06; YM99; BHS15; HBK14] also consider relevance and redundancy relationship to select features. The application of feature selection is wide and beneficial. Some real-life applications of feature selection are: healthcare analytics [Pat+22; Che+20; RB19; Nag+22], image analysis [BR20] and recognition [Özy20], credit scoring [Koz+19; Tri20; KR23], marketing analysis [BSC23; Qia+22; YGX20; Haq+21], anomaly detection [BBK16; De +14; El +22; Nak+21; Ras+22], bio-informatics [WWC16; SIL07; Li+17], etc.

These feature selection methods are not directly applicable to federated learning because, in federated settings, not all clients have complete information about the data.

2.2 Distributed feature selection

Although centralized feature selection approaches are fast and effective. They struggle to achieve satisfactory performance when dealing with big data, which is characterized not only by its large volume but also by its diverse and intricate nature. Therefore, a specific distributed feature selection method is necessary. In [MBA17], the authors compare distributed vs centralized feature selection. Several rounds of feature selection are performed on horizontal as well as vertical partitions of data. Finally, the outputs of every round are combined and produce a single subset of relevant features using different data complexity measures [HBL06]. An adaptive aggregation (ADAGES) flexible distributed feature selection method is proposed in [Gui20]. A distributed fuzzy rough set (DFRS) based feature selection method to enable fuzzy rough set for big data analysis is presented in [Kon+19]. A distributed quadratic programming-based feature selection is reported in [SE20]. In [Zad+17], the authors formalized the feature selection problem as a diversity maximization problem by proposing an MI-based metric distance on features. They focused on vertically distributed feature selection that can deal with redundancy.

FL has additional characteristics, including imbalanced and massively distributed IID (see Definition 2) and non-IID (see Definition 3) data, and clients with limited computational capacity, despite the fact that distributed learning and federated learning appear to be similar. However, the idea behind FL is that data should remain private to the clients.

2.3 Horizontal federated feature selection

Federated feature selection is challenging because the distribution of the whole dataset is unknown for each client. Federated feature selection is first introduced in [BEB21], where authors formulated the feature selection problem for horizontal federated learning. The authors proposed a hybrid feature selection method using mutual information and clustering. Fed-FiS is stable while data distribution is IID, i.e., every client has information on features and classes but is not stable for vertical or hybrid distributions. Cassara et al. [CGV22] proposed federated feature selection for cyber-physical systems. Their approach involves a mutual information-based feature selection algorithm run by the autonomous vehicles (clients) and Bayes' theorem-based aggregation executed on the server. Hu et al. [Hu+22] proposed a federated feature selection algorithm using evolutionary computing techniques. Zhang et al. [Zha+23] proposed an unsupervised feature selection method for horizontal federated learning where clients share a common feature space but have different class labels. In [QK21], suggested a greedy algorithm for feature selection. In Table 1, we report a comparative study by considering multiple key factors between our method and the current state-of-the-art horizontal federated feature selection methods [BEB21; Zha+23; CGV22; QK21; HBL24].

3 Problem Statement

Consider a HFL system consists of q clients ($\forall_{i=1}^{q} Cl_i$) and a server. We assume that $q \geq 2$, if $q = 1$, it is considered a centralised system with full dataset information. Suppose the dataset D contains samples $S \in \mathbb{R}^{n \times d}$, the features set $F \in \mathbb{R}^{d \times 1}$, and class $C \in \{0, 1, \ldots, k\}^{n \times 1}$. D is distributed across q clients such that each client contains the features set F. Sample set $S_{Cl_i} \in \mathbb{R}^{m \times d}$, where $\cup_{i=1}^{q} S_{Cl_i} = S$, and $\cap_{i=1}^{q} S_{Cl_i} = \emptyset$ and class $C_{Cl_i} \in \{0, \ldots, k\}^{m \times 1}$, where $m < n$. In HFL, all clients have partial class information, which creates statistical heterogeneity. Our objective is to uncover relevant features subset (F'') and obtain stable and generalizable global model performance.

4 Proposed Approach

Our approaches comprise two components: (1) local feature selection performed independently by clients using mutual information and clustering, and (2) global feature selection achieved through a global score function for Fed-FiS and multi-objective optimization for Fed-MOFS.

4.1 Data division

For a given dataset $D(F, S) \in \mathbb{R}^{n \times d}$, consists of feature set $F = \{f_1, f_2, \ldots, f_d\}^T$, $F \in \mathbb{R}^{d \times 1}$, and sample set $S = \{s_1, s_2, \ldots, s_n\}$, $S \in \mathbb{R}^{n \times d}$. The dataset is distributed across q clients in a horizontal (see Figure 1) manner. In horizontal federated learning, all

Table 1: Comparisons among prior works on feature selection in FL settings. None of the current methods fully meet the specified requirements (C1)–(C5)

Federated Feature selection algorithms	Category (C1)	Feature Relevance (C2)	Feature Redundancy (C3)	Global Feature Selection using Multi-objective optimization (C4)	Computational Complexity (C5)	Benefits	Drawbacks
CE-based FFS [CGV22] [2022]	Filter	✓	✓	×	$O(n*d)$	• Robustness against non-IID data	• Complexity in the aggregation function
Greedy-Feature Selection [QK21] [2021]	Wrapper	×	×	×	$O(nd^2)$	• Adapt to various attack types by selecting different feature sets for each type of attack • The algorithm is straightforward and can effectively identify non-contributing or redundant features	• Suboptimal feature set due to greedy nature • The outcome heavily depends on the initial set of features
FSHFL [Zha+23] [2023]	Filter	✓	×	×	$O(d^2 \log d)$	• Improvement in performance • Reduced training time and energy consumption • Privacy benefit for federated learning	• High in complexity • Hierarchical clustering is time-consuming, therefore not scalable
Fed-mRmR [HBL24] [2024]	Filter	✓	✓	×	$O(nd^2)$	• Lossless feature selection • Capability of finding features from non-IID distributions	• Scalability issues during handling large datasets • High in computation complexity
Fed-FiS [BEB21] [2021]	Hybrid	✓	✓	×	$O(d^2)$	• Improved handling of heterogeneity • Reduced computation and communication cost • Stable and relevant feature selection	• Highly dependent on mutual information measures • Potential computational overhead on devices
Fed-MOFS This work	Hybrid	✓	✓	✓	$O(d^2)$	• Stable across IID and non-IID data distribution • Scalable across multiple clients • Stable and relevant feature selection	• Highly dependent on mutual information measures • Potential computational overhead on devices

Table 2: Notations

Symbol	Description	Symbol	Description
D	Dataset, $D \in \mathbb{R}^{n \times d}$.	F	Feature space $F = \{f_1, f_2, \ldots, f_d\}^\intercal$, $F \in \mathbb{R}^{d \times 1}$.
n	Number of samples	F'_{server}	The global feature set that contains unique features only.
d	Number of features	F''	The feature subset according to the rank of each feature.
S	Sample space, $S \in \mathbb{R}^{n \times d}$.	F'_{Cl_i}	Locally selected features at client Cl_i.
D_i	Dataset of client Cl_i	$\tau_k^{Cl_i}$	Feature triplet for k^{th} feature of Cl_i
β	Number of clusters or cluster centroids	ζ	Current cluster or cluster centroid
δ	Performance threshold of global model	ψ	Performance of global model.
C	Class, $C \in \{0, \ldots, \kappa\}^{n \times 1}$.	q	Number of clients.
θ_{Cl_i}	Local model of the i^{th} client.	ω	Global model.
Cl_i	i^{th} client from a pool of q clients.	$\tau_k^{Cl_i}$	Triplet of the k^{th} locally selected feature of the i^{th} client.
F_{Cl_i}	Feature set available at the i^{th} client.	S_{Cl_i}	Sample set available at the i^{th} client.
$FCMI$	Feature Class Mutual Information.	$aFFMI$	Averaged Feature Feature Mutual Information.
$f_{k_{FCMI}}^{Cl_i}$	FCMI of the k^{th} feature of the i^{th} client.	$f_{k_{aFFMI}}^{Cl_i}$	aFFMI of the k^{th} feature of the i^{th} client.
$f_i^{Cl_i}$	i^{th} feature of the client Cl_i.	C_{Cl_i}	Set of target class available at the i^{th} client.
$\mid F_{Cl_i} \mid$	Number of features at client Cl_i.	$\mid S_{Cl_i} \mid$	Number of samples at client Cl_i.
$F_{Cl_i}^{FCMI}$	Set of features with FCMI values for the i^{th} client	$F_{Cl_i}^{aFFMI}$	Set of features with aFFMI values for the i^{th} client
$F'_{Cl_i{}^{FCMI}}$	Set of features of Cl_i that belong to the clusters that have maximum centroid value.	$F'_{Cl_i{}^{aFFMI}}$	Set of features of Cl_i that belong to the clusters that have minimum centroid value.
$f_{k_{FCMI}}$	Average of FCMI value of feature f_k across all clients.	$f_{k_{aFFMI}}$	Average of aFFMI value of feature f_k across all clients.
T	Global rounds	ε	Number of features selected from the ranked features for training.
Δ	Fraction of clients participated $\Delta \in (0,1)$	γ	Non-IID factor $\gamma \in [0,1]$

clients have access to the same set of features but different samples. The dataset is considered to be Independent and Identically Distributed (IID) if each client possesses complete information about all the classes. On the other hand, if clients only have partial information about the classes, the data distribution is referred to as Non-Independent and Identically Distributed (non-IID). We introduced the following foundational definitions to understand the federated settings.

Definition 1. ***Horizontal:*** *For a given dataset $D(F,S) \in \mathbb{R}^{n \times d}$, consists of feature set F, and sample set S, distributed across q clients. Then all clients have similar feature sets but different samples, i.e., $\bigcap_{i=1}^{q} F_{Cl_i} = F$ but $\bigcap_{i=1}^{q} S_{Cl_i} = \emptyset$, where Cl_i is the i^{th} client, F_{Cl_i} and the S_{Cl_i} are the feature and sample set of Cl_i, respectively.*

Lemma 1. *All clients share common feature set, $F_{Cl_i} \cup F_{Cl_j} = F$.*

Proof. Lets consider clients Cl_i and Cl_j, if $F_{Cl_i} \bigcap F_{Cl_j} = \emptyset$, then there is no common features between clients Cl_i and Cl_j. According to Definition 1, $F_{Cl_i} \bigcap F_{Cl_j} \neq \emptyset$. By contradiction, we proved that the Lemma 1 is true. □

Definition 2. ***IID:*** *For a given dataset, $D(F,S) \in \mathbb{R}^{n \times d}$, $C = \{c_1, c_2, \ldots, c_\kappa\}$ consisting of κ classes, it would be called IID if the probability distribution of each class c_i is independent of other class $P(c_i | c_1, c_2, \ldots, c_\kappa) = \mathscr{P}(c_i)$ and all classes are drawn from the same underlying probability distribution $\mathscr{P}(c)$, $\mathscr{P}(c_1 = c, c_2 = c, \ldots c_\kappa = c) = \mathscr{P}(c)$.*

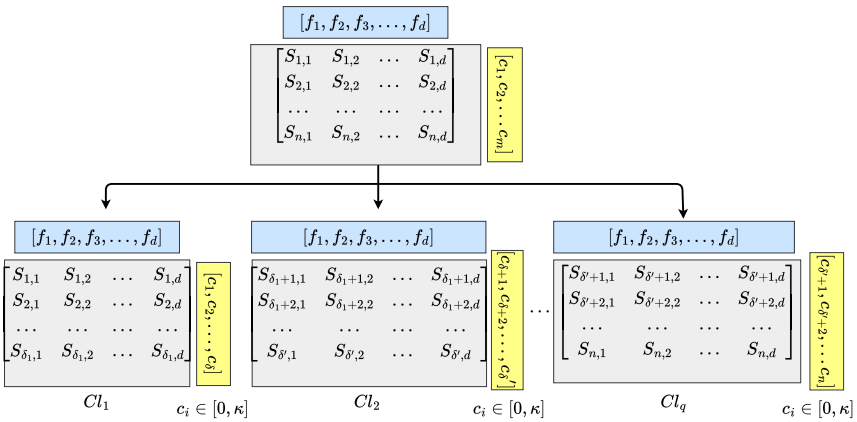
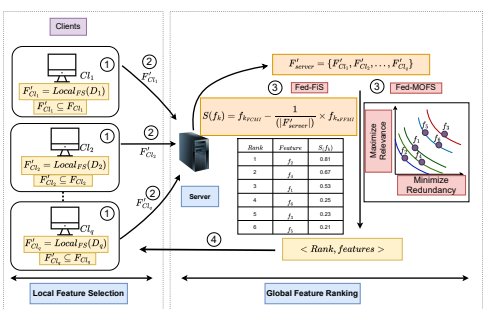


Figure 1: Horizontal data division in HFL

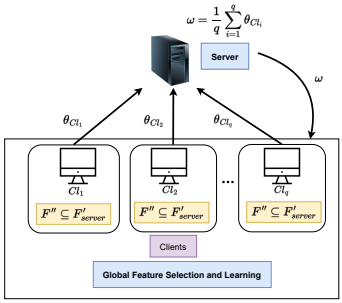
Definition 3. ***Non-IID:*** *For a given dataset, $D(F,S) \in \mathbb{R}^{n \times d}$, $C = \{c_1, c_2, \ldots, c_\kappa\}$ consisting of κ classes, it would be called non-IID if the probability distribution of each class c_i depends on the values or presence of other classes within dataset $\mathscr{P}(c_i|c_1, c_2, \ldots, c_\kappa) \neq \mathscr{P}(c_i)$, and the classes are drawn from different underlying probability distributions, across different clients, $\mathscr{P}(c_1) \neq \mathscr{P}(c_2) \neq \ldots \neq \mathscr{P}(c_\kappa)$.*

4.2 Framework

A conventional HFL consists of q clients $(Cl_1, Cl_2, \ldots, Cl_q)$ and a server. Figure 2a illustrates the proposed framework in four steps, as follows.



(a) Global feature ranking



(b) Feature selection and learning

Figure 2: Proposed framework

1. Each client Cl_i has its private feature set F_{Cl_i} and runs the procedure $Local_{FS}(D_i)$ independently to generate local feature subset F'_{Cl_i} ($F'_{Cl_i} \subseteq F_{Cl_i}$) .
2. Each client Cl_i sends the FCMI (see Definition 4) and aFFMI (see Definition 5 and 6) value of each $f_i^{Cl_i} \in F'_{Cl_i}$ to the server. Server averages these scores of similar features and generate a list of unique feature set F'_{server}.
3. Server applies Fed-FiS or Fed-MOFS to generate the global ranks of the locally selected features.
4. Server sends the global ranks of each feature to the clients.

In Figure 2b, after having the rank of each feature, all clients start FL with a feature subset ($F'' \subseteq F'_{server}$).

4.3 Algorithms

The federated feature selection approaches are described with four algorithms. Algorithm 1 for local feature selection, Algorithm 2 and 3 described global feature ranking using Fed-FiS and Fed-MOFS, respectively, and finally, Algorithm 4 for global feature selection.

4.3.1 Local feature selection

We adopted Mutual Information (MI) to measure the certainty of a feature variable with a target variable, which could be another feature or a class [Swi12; KSG04]. The MI-based feature selection approach depends on the relevance of each feature, measured by Feature-Class Mutual Information (FCMI, see Definition 4), as well as the redundancy, measured by, Feature-Feature Mutual Information (FFMI, see Definition 5) of each feature. The average of all FFMI values of features is called aFFMI (see Definition 6).

Definition 4. ***FCMI:*** *Given a feature $f_k^{Cl_i}$ at client Cl_i, and class C, then FCMI of feature $f_k^{Cl_i}$ and C can be computed as:*

$$FCMI(f_k^{Cl_i},C) \triangleq \sum_{f_k^{Cl_i},C} \mathscr{P}(f_k^{Cl_i},C) log \frac{\mathscr{P}(f_k^{Cl_i},C)}{\mathscr{P}(f_k^{Cl_i})\mathscr{P}(C)} \tag{1}$$

where $f_k^{Cl_i}$ is the k^{th} feature of the i^{th} client (Cl_i). $\mathscr{P}(f_k^{Cl_i})$ and $\mathscr{P}(C)$ are the marginal and $\mathscr{P}(f_k^{Cl_i},C)$ is the joint probability distribution for $f_k^{Cl_i}$ and C.

Definition 5. ***FFMI:*** *Given two features $f_k^{Cl_i}$ and $f_j^{Cl_i}$, at client Cl_i, then the FFMI of features $f_k^{Cl_i}$ and $f_j^{Cl_i}$ can be estimated as:*

$$\begin{aligned} FFMI(f_k^{Cl_i},f_j^{Cl_i}) \triangleq MI(f_k^{Cl_i};f_j^{Cl_i}) = H(f_k^{Cl_i}) + \\ H(f_j^{Cl_i}) - H(f_k^{Cl_i},f_j^{Cl_i}) \end{aligned} \tag{2}$$

where $f_k^{Cl_i}$ and $f_j^{Cl_i}$ are k^{th} and j^{th} feature of the i^{th} client (Cl_i), respectively, and $f_k^{Cl_i} \neq f_j^{Cl_i}$. $H(f_k^{Cl_i})$ and $H(f_j^{Cl_i})$ are marginal entropy. $H(f_k^{Cl_i}, f_j^{Cl_i})$ is the joint entropy of the $f_k^{Cl_i}$ and $f_j^{Cl_i}$, and $f_k^{Cl_i} \neq f_j^{Cl_i}$.

Definition 6. ***aFFMI:*** *Given a set of d features at client Cl_i, then averaged FFMI (aFFMI) value of $f_k^{Cl_i}$ can be calculated as:*

$$aFFMI(f_k^{Cl_i}) = \frac{1}{d-1} \sum_{j=1, f_j^{Cl_i} \in F_{Cl_i} \setminus f_k^{Cl_i}}^{d-1} FFMI(f_k^{Cl_i}, f_j^{Cl_i}) \tag{3}$$

where $f_k^{Cl_i}$ and $f_j^{Cl_i}$ are k^{th} and j^{th} feature of the i^{th} client (Cl_i), respectively, and $f_k^{Cl_i} \neq f_j^{Cl_i}$.

Algorithm 1 Local feature selection

Input: $F_{Cl_i} = \{f_1^{Cl_i}, f_2^{Cl_i}, \ldots, f_d^{Cl_i}\}$ is the original feature set and d is the dimension of the data for the i^{th} client Cl_i
Output: F'_{Cl_i} is the selected features for client i

1: **procedure** $Local_{FS}(D_i)$
2: **for** $f_k^{Cl_i} \in F_{Cl_i}$ **do**
3: $f_k^{FCMI} = FCMI(f_k^{Cl_i}, C)$ using Equation (1). ▷ return FCMI score of a feature
4: $f_k^{aFFMI} = aFFMI(f_k^{Cl_i})$ using Equation (2) and Equation (3). ▷ return averaged FFMI score of a feature
5: $F_{Cl_i}^{FCMI} = \{f_k^{FCMI} \mid \forall_{k=1}^{d} f_k^{FCMI} \in [0,1]\}$
6: $F_{Cl_i}^{aFFMI} = \{f_k^{aFFMI} \mid \forall_{k=1}^{d} f_k^{aFFMI} \in [0,1]\}$
7: $F'_{Cl_i FCMI}$ = CLUSTER($F_{Cl_i}^{FCMI}$) ▷ return cluster of features with high FCMI values
8: $F'_{Cl_i aFFMI}$ = CLUSTER($F_{Cl_i}^{aFFMI}$) ▷ return cluster of features with low aFFMI values
9: $F'_{Cl_i} = F'_{Cl_i FCMI} \bigcup F'_{Cl_i aFFMI}$
10: return F'_{Cl_i}
11: **procedure** CLUSTER($F_{Cl_i}^x$) ▷ here, $F_{Cl_i}^x$ is $F_{Cl_i}^{FCMI}$ or $F_{Cl_i}^{aFFMI}$
12: Initialize β random cluster centroid.
13: **repeat**
14: $\forall_{k=1}^{|F_{Cl_i}^x|} f_k \in F_{Cl_i}^x$
15: minimum $\leftarrow 0$
16: cluster_member $\leftarrow 0$
17: $\forall_{\zeta=1}^{\beta} centroid\ \zeta \in \beta$
18: dist $\leftarrow$ Distance(f_k, ζ)
19: **if then** $dist < minimum$
20: minimum $\leftarrow$ dist
21: cluster_member $\leftarrow \zeta$
22: recalculate centroid(ζ)
23: **until** Converge
24: return $F'_{Cl_i{}^x}$ ▷ here, $F'_{Cl_i{}^x}$ is $F'_{Cl_i FCMI}$ or $F'_{Cl_i aFFMI}$

Algorithm 1 computes the FCMI and aFFMI values of all features (Line 2 to Line 4) at each client using the Equation (1), Equation (2), and Equation (3), respectively. $F_{Cl_i}^{FCMI}$ and $F_{Cl_i}^{aFFMI}$ are two one-dimensional vectors that contain FCMI and aFFMI values of all features at client Cl_i. The size of the vector depends on the number of features present at the client, but it is bound to d. The FCMI and aFFMI values are within the range of 0 to 1 (Line 5 and Line 6). The FCMI value close to zero indicates the low

relevance of that feature. The aFFMI value close to one indicates the high redundancy of that feature. Here, we aim to find the optimal feature set by (1) maximizing the relevance and (2) minimizing the redundancy. Based on the FCMI and aFFMI values, we compute CLUSTER($F_{Cl_i}^{FCMI}$) and CLUSTER($F_{Cl_i}^{aFFMI}$) (Line 7 and Line 8) using the procedure CLUSTER (Line 11 to Line 24) to generate feature clusters with higher FCMI and lower aFFMI values. If there are β clusters of features, then the objective can be written as follows.

$$F'_{Cl_i{}^{FCMI}} = \arg\max_{\forall i \in \beta} Centroid(cluster_i) \tag{4}$$

$$F'_{Cl_i{}^{aFFMI}} = \arg\min_{\forall i \in \beta} Centroid(cluster_i) \tag{5}$$

where $Centroid(cluster_i)$ returns the centroid value of the i^{th} cluster, and $\mid cluster_i \mid$ is the cardinality of $cluster_i$.

In Equation (4), select the cluster with the maximum centroid($F'_{Cl_i{}^{FCMI}}$). It contains the features with utmost relevance and $F'_{Cl_i{}^{FCMI}} \subseteq F_{Cl_i}$. Similarly, Equation (5) returns the cluster with the minimum centroid ($F'_{Cl_i{}^{aFFMI}}$). It contains features with minimum redundancy and $F'_{Cl_i{}^{aFFMI}} \subseteq F_{Cl_i}$. Union of the output of Line 7 and Line 8 produces the final local feature subset (Line 9). The example of local feature selection is in illustration 4.3.1.

***Illustration*:** In Table 3, five features, f_1, f_2, f_3, f_4, and f_5 have been distributed across three clients, Cl_1, Cl_2, and, Cl_3 horizontally. The FCMI and aFFMI scores of each feature are estimated on each client independently, and then the clustering method is employed on the FCMI and aFFMI scores separately and divided into two clusters, C1 and C2. The value of cluster center of $C1 > C2$. Therefore, a union between C1 from Clustered FCMI and C2 from Clustered aFFMI has to be performed to get the locally selected features.

4.3.2 **Fed-FiS**

In Algorithm 2, each client, Cl_i sends triplets of locally selected features to the server. The definition of the feature triplet is given below.

Definition 7. ***Feature triplet*** ($\tau_k^{Cl_i}$)**:** *$\forall_{i=1}^{q} Cl_i$, a feature $f_k^{Cl_i} \in F'_{Cl_i}$ then feature triplet of $f_k^{Cl_i}$ can be defined as $\tau_k^{Cl_i} =< f_k^{Cl_i}, f_{k_{FCMI}}^{Cl_i}, f_{k_{aFFMI}}^{Cl_i} >$*

where $\tau_k^{Cl_i}$ is the triplet of the k^{th} locally selected feature of the i^{th} client Cl_i. $f_k^{Cl_i}$ is the unique identifier of the feature. $f_{k_{FCMI}}^{Cl_i}$, and $f_{k_{aFFMI}}^{Cl_i}$ are the FCMI and aFFMI score of $f_k^{Cl_i}$ respectively.

Each device sends a vector of triplets to the server. So a triplet vector F'_{Cl_i} of device Cl_i can be defined as $F'_{Cl_i} = \{\tau_1^{Cl_i}, \tau_2^{Cl_i}, \ldots, \tau_k^{Cl_i}\}$, where $k \leq d$. The server receives

Table 3: Local feature selection

Client(s)	Feature(s)	FCMI	aFFMI	Clustered FCMI	Clusered aFFMI	Locally selected features
	f_1	0.67	0.31			
	f_2	0.91	0.43	**C1:[** f_2,f_1,f_3**]**	C1:[f_2,f_4,f_5]	f_1,f_2,
Cl_1	f_3	0.57	0.21	C2:[f_4,f_5]	**C2:[** f_1,f_3**]**	f_3
	f_4	0.27	0.61			
	f_5	0.17	0.51			
	f_1	0.57	0.32			
	f_2	0.81	0.21	**C1:[** f_1,f_2,f_3,f_4**]**	C1:[f_4, f_5]	f_1,f_2,
Cl_2	f_3	0.37	0.42	C2:[f_5]	**C2:[** f_2, f_1, f_3**]**	f_3, f_4
	f_4	0.47	0.53			
	f_5	0.17	0.66			
	f_1	0.73	0.31			
	f_2	0.82	0.17	**C1:[** f_1,f_2,f_4**]**	C1:[f_4]	f_1,f_2
Cl_3	f_3	0.46	0.23	C2:[f_3,f_5]	**C2:[** f_2,f_3,f_1, f_5**]**	f_3, f_4
	f_4	0.51	0.41			f_5
	f_5	0.34	0.31			

feature triplets from q clients (Line 2). Multiple clients can share a single feature f_k. Therefore, F_{server} may have multiple similar features with different FCMI and aFFMI scores. Server averages the FCMI and aFFMI values of similar features using the Equation (6), and Equation (7) (Line 3).

$$f_{k_{FCMI}} = \frac{\sum_{i=1}^{j} f_{k_{FCMI}}^{Cl_i}}{j}, where\, j \in [1,q] \tag{6}$$

$$f_{k_{aFFMI}} = \frac{\sum_{i=1}^{j} f_{k_{aFFMI}}^{Cl_i}}{j}, where\, j \in [1,q] \tag{7}$$

Server generates the unique feature list F'_{server} (Line 4). For all features, the server computes a score $S(f_k)$ using Equation (8) (Line 5).

$$S(f_k) \ = f_{k_{FCMI}} - \frac{1}{(\mid F'_{server} \mid -1)} \times f_{k_{aFFMI}}, \tag{8}$$

where $\mid F'_{server} \mid > 1$ *and* $S(f_k) \in [1,-1]$
Finally, the server sends the score of each feature to the clients (Line 6). We give an example of the working of Fed-FiS in illustration 4.3.2.

***Illustration*:** The server initially computed the average FCMI and averaged aFFMI values for a set of five features, as presented in Table 4. Subsequently, the feature

Algorithm 2 Fed-FiS (Score-based global feature ranking)

Input: $F_{Server} = \{F'_{Cl_1}, F'_{Cl_2} \ldots F'_{Clq}\}$ ▷ collection of feature triplets from q clients
Output: $< rank, f_k >$ ▷ global rank of features

1: **procedure** $Fed - FiS(F_{Server})$
2: server obtained $F_{server} = \{F'_{Cl_i} | \forall_{i=1}^{q}, F'_{Cl_i} \in Cl_i\}$.
3: obtain global feature triplet by performing average over FCMI (Equation (6)) and aFFMI (Equation (7)) scores individually.
4: obtain $\{F'_{server} | \forall f_k \in F'_{server}$ *are unique*$\}$
5: compute $S(f_k), \forall\ f_k\ \in\ F'_{server}$ using Equation (8)
6: $\forall_{i=1}^{q} Cl_i$ send $< S(f_k), f_k >$ to all Cl_i iff $f_k \in F'_{server}$

scores were determined using Equation (8). These scores were then utilized to assign ranks to the features in descending order, starting from the highest score, $S(f_k)$. Therefore, The features are ranked in the following order from highest to lowest: f_2, f_1, f_4, f_3, and f_5.

Table 4: Illustration of feature ranking with score function in Fed-FiS

Feature(s)	**Averaged FCMI**	**Averaged aFFMI**	**S($\mathbf{f_k}$)**	**Rank**
f_1	0.66	0.31	0.58	2
f_2	0.85	0.27	0.78	1
f_3	0.47	0.29	0.398	4
f_4	0.52	0.47	0.403	3
f_5	0.34	0.31	0.26	5

Remark 1. *Fed-FiS employs $S(f_k)$ to determine the disparity between the average relevance and redundancy of a feature. When $S(f_k) \to 1$, i.e., this disparity is closer to 1 then the feature is significantly relevant and carries beneficial impact for learning. Conversely, when $S(f_k) \to -1$, it suggests that the feature is less relevant, making it less crucial for learning and potentially causing a negative impact. $S(f_k) = 0$ indicates that the feature possesses equal levels of relevance and redundancy. While it may have a positive impact on learning, it doesn't guarantee the absence of any negative effects.*

4.3.3 **Fed-MOFS**

Algorithm 3 is also a feature ranking algorithm similar to Fed-FiS, but here server applies a multi-objective optimization to find the dominant features based on the two objective functions, (1) maximizing the average FCMI score, and (2) minimizing the average aFFMI score (Line 6). This multi-objective optimization produces Pareto fronts from where we get the ranking of the features (Line 7). Finally, the server sends the feature with its rank to the clients (Line 8). The example of the working of Fed-MOFS is in illustration 4.3.3.

***Illustration*:** In Table 5, the averaged FCMI and averaged aFFMI values for features f_1 to f_5 are presented, which were collected from clients. The server then performs a

Algorithm 3 Fed-MOFS (Multi-objective optimization based global feature ranking)

Input: $F_{Server} = \{F'_{Cl_1}, F'_{Cl_2} \dots F'_{Cl_q}\}$ ▷ features from q clients
Output: $< rank, f_k >$ ▷ global rank of features

1: **procedure** $Fed-MOFS(F_{Server})$
2: server obtained $F_{server} = \{F'_{Cl_i} | \forall_{i=1}^{q}, F'_{Cl_i} \in Cl_i\}$.
3: obtains global feature triplet by performing average over aFFMI and FCMI scores individually.
4: obtain $\{F'_{server} | \forall f_k \in F'_{server}$ *are unique*$\}$
5: Optimize the following functions to find Pareto optimality:
6: (1) $\max_{\forall\ f_k\ \in F'_{server}} f_{k_{FCMI}}$ (2) $\min_{\forall\ f_k\ \in F'_{server}} f_{k_{aFFMI}}$
7: Get the ranking of the features from the Pareto fronts.
8: $\forall_{i=1}^{q} Cl_i$ send $< rank, f_k >$ to all Cl_i iff $f_k \in F'_{server}$

multi-objective optimization, aiming to maximize the FCMI scores while minimizing the aFFMI scores in order to determine the domination and dominated count for each feature. The difference between the domination and dominated count is used to rank the features. According to Table 5, feature f_2 has domination count 4, i.e., its averaged FCMI is higher than f_1, f_3, f_4, and f_5. It also has a dominated count 0 because its average aFFMI is the lowest among all features. Therefore, feature f_2 is not dominated by other features. Consequently, the difference between domination count and dominated count ($c_{dom} - f_{dom}$) is 4, which is the highest among all features. Hence, f_2 is the most important feature, followed by f_2, f_1, f_4, and f_5. The difference between domination and dominated count for f_4 and f_5 are the same. But the domination count of f_4 is better than f_5. Hence, the rank of f_4 is higher than f_5 as it has more relevance.

Table 5: Fed-MOFS - illustration of feature ranking through multi-objective optimization

Feature(s)	Averaged FCMI	Averaged aFFMI	Domination count (c_{dom})	Dominated count (f_{dom})	$c_{dom} - f_{dom}$	Rank
f_1	0.66	0.31	3	2	1	2
f_2	0.85	0.27	4	0	4	1
f_3	0.47	0.29	1	1	0	3
f_4	0.52	0.47	2	4	-2	4
f_5	0.34	0.31	0	2	-2	5

Remark 2. *Fed-MOFS utilizes a multi-objective optimization approximation algorithm to determine feature rankings. Therefore, Fed-MOFS never guarantees to find the exact optimal solutions but provides a set of non-dominated solutions (Pareto-optimal). But by prioritizing the domination count over the dominated count, we can get the unique rank of each feature.*

4.3.4 Global feature selection

Algorithm 4 describes the procedure for global feature selection after acquiring feature rankings from Fed-FiS or Fed-MOFS. Every client selects a common set of features ($F'' \subseteq F'_{Server}$) according to their individual rankings (Line 3), and then clients train the

local model (θ_{Cl_i}) (Line 6). The server collects the local updates from the clients and computes the global model (ω) (Line 7). After T global rounds, the performance (ψ) of global model (ω) is evaluated (Line 8) and checks whether ψ achieves a specified threshold (δ) (Line 9). If $\psi \geq \delta$, then, F'' is the selected feature subset (Line 10); otherwise, an additional set of ε features from the next-ranked features is to be incorporated (Line 12).

Algorithm 4 Global feature selection

Input: F'_{Server} ▷ features according to their global ranks
Output: F'' ▷ global feature subset

1: **procedure** $Global_{FS}(F'_{Server})$
2: **while** $\psi \leq \delta$ **do**
3: All Cl_i selects a common feature subset $F'' \subseteq F'_{server}$ based on their global ranks.
4: **for** $t = 1 \ldots T$ **do** ▷ T is the global rounds
5: **for** All clients in parallel **do**
6: θ_{Cl_i} = Local_update($D(F'', S_{Cl_i}, C_{Cl_i})$)
7: ω = Global_update($\theta_1, \theta_2, \ldots, \theta_q$)
8: Compute accuracy of the global model (ψ)
9: **if** $\psi \geq \delta$ **then**
10: F'' is the selected features.
11: **else**
12: $| F'' | = | F'' | + \varepsilon$ where $| F'' | + \varepsilon \leq |F_{Server}|$

4.4 Computational complexity

The computational complexity of Algorithm 1 depends on the dimensionality of the input data and clustering algorithm. For a given data $D_i \in \mathbb{R}^{n \times d}$ at client Cl_i, the computational complexity of calculating FCMI and aFFMI is $O(d^2)$. Clustering is a np-hard problem, but taking heuristics can make the time complexity of it to linear. FCMI and aFFMI are two one-dimensional vectors of size d, so the time complexity for clustering is $(O(d \cdot \beta \cdot \gamma))$ where β is the number of clusters, and γ is the number of iterations. So the computation complexity of Algorithm 1 is $O(d^2) + O(d \cdot \beta \cdot \gamma)$.

In Algorithm 2 The computation complexity to calculate $S(f_k)$ (see Equation (8)) is $O(| F'_{server} |)$ where $| F'_{server} | \leq d$, so in worst case the computation complexity would be $O(d)$.

In Algorithm 3, the computation of F'_{server} requires $O((d.q)^2)$ because in worst-case in each client, the local feature subset contains all features. The global feature selection with multi-objective optimization uses NSGA-II[Deb+02] algorithm so the time complexity of it is $O(M. | F'_{server} |^2)$, where $| F'_{server} | \leq d$ and M is the number of objectives. Here, we optimize two objectives, so the worst-case time complexity is $O(d^2)$.

The computational complexity of the Algorithm 4 depends on the specific learning problem. In the case of strongly convex and smooth problems, the convergence of FedAvg on non-IID dataset is $O\left(\frac{1}{T}\right)$ [Li+19], where T is the global rounds. If we have a dataset with d features, and selected features are ε, then the Algorithm 4 needs to be run a maximum of $\frac{d}{\varepsilon}$ times. Consequently, for strongly convex and smooth problems, the computational complexity of Algorithm 4 is $\frac{d}{\varepsilon} \cdot O\left(\frac{1}{T}\right)$.

4.5 Communication cost

To perform Algorithm 1, 2, and 3, both the client and server require a single communication round. A federated learning approach is utilised in the case of Algorithm 4. Consequently, the maximum number of required communication rounds can be expressed as $\frac{d}{\varepsilon} \cdot T$. Therefore, total communication cost $\leq 1 + \frac{d}{\varepsilon} \cdot T$.

5 Evaluation

We evaluated the performance of Fed-FiS and Fed-MOFS across multiple datasets. The details of the datasets and data statistics are given in Table 6. A detailed discussion of the datasets is in the supplementary copy. We evaluated both of our algorithms on both IID and non-IID data division.

Table 6: Datasets

Datasets	Instances	Features	Classes	Type
NSL-KDD99 [Tav+09]	125973	41	2	Intrusion detection
ACC [Dal+17]	284807	30	2	Credit card fraud transaction detection
Wine [AF91]	178	13	3	Wine quality
Vowel [Tur14]	990	13	11	Speech recognition
Vehicle [Var20]	846	9	4	Vehicle ownership
Segmentation [Sur]	10695	10	4	Customer segmentation
WDBC [al95]	569	30	2	Breast cancer
Ionosphere [al89]	351	34	2	Radar scans
Hill_valley [LF08]	606	100	2	Terrain
ISOLET [RM94]	7797	617	26	Speech data
Diabetes [DDD90]	768	8	2	Diabetes diagnostics
IoT [Ant19]	503910	28	17	Smart home data
Boston [Koe23]	506	14	-	Regression dataset. Median Price of owner-occupied homes
California [Nug17]	20640	9	-	Regression dataset, Predicting median house values in California
Synthetic	100000	200	25	Synthetic data

5.1 Experimental setup

The experimental setup is described in Table 7. Each client is responsible for selecting local features and the local training of the model. Meanwhile, the server estimates the global feature scores and ranks them. Moreover, server performs aggregation of the local models. Our local feature selection approach is similar to [BEB21]. We employed the k-means clustering algorithm and set the number of clusters to 2 based on the highest silhouette score of the clusters. NSGA-II was employed to maximize relevance and minimize redundancy. Subsequently, we implemented federated forest and FedAvg algorithms, utilizing feature selection results from Fed-FiS and Fed-MOFS. In FedAvg, we are training a deep neural network (DNN). The description of the DNN

is given in the supplementary copy. Additionally, we performed feature selection at the client level using RFE and ANOVA-based methods. This means each client independently used RFE and ANOVA for feature selection and sent the information regarding the selected features to the server. The server performs an intersection to find common features, and clients perform federated learning on the selected features. Furthermore, we compared our feature selection techniques with FSHFL [Zha+23], and Fed-mRMR[HBL24], the state-of-the-art federated feature selection method. We ran all experiments 10 times and carried the mean of the results.

Table 7: Parameters description in federated feature selection

Parameter(s)	**Value**	**Description**
clients	5 to 100	Local feature selection, Learning local model
Server	1	Global feature selection, Learning global model
client's participation	10 to 100%	Partial or full participation of each client
Clustering algorithm	k-means	Cluster with nearest mean
Multi-objective optimization	NSGA-II	Non-dominated Sorting Genetic Algorithm
Feature selection state-of-the-art	4	ANOVA, RFE, FSHFL, Fed-mRMR
Learning algorithm	2	Federated Forest [Liu20], FedAvg [McM+17]
Performance metrics	7	Accuracy, Precision, Recall, and F1-Score, RMSE, MAE, Categorical cross-entropy loss.

5.2 Results and analysis

Here, we assessed the outcomes based on various aspects, including performance, scalability, stability, efficiency, and convergence of the global model.

5.2.1 **Performance evaluation**

We evaluated the performance of Fed-FiS and Fed-MOFS for both IID and non-IID data division on multiple datasets representing classification and regression tasks.

5.2.1.1 **Performance evaluation with IID data divisions:** In the following experiments in Table 8 and 9, we considered the data with random distribution among 5 clients and ensured that each client has information from all classes (IID). After conducting training with Federated Forest (Table 8) and training a deep neural network using federated averaging (Table 9) on both full feature sets and reduced feature sets

Table 8: Performance of federated forest on selected features (*mean* / *ratio of feature selected*)

Dataset	All Features		RFE		ANOVA		FSHFL		Fed-mRMR		Fed-FiS		Fed-MOFS	
	F1-Score	Accuracy	F1-Score	Accuracy	F1-Score	Accuracy	F1-Score	Accuracy	F1-Score	Accuracy	F1-Score	Accuracy	F1-Score	Accuracy
Ionosphere	0.93/1.0	0.92	0.91/0.97	0.91	0.92/0.91	0.91	**0.93/0.54**	**0.93**	0.94/0.87	**0.94**	0.92/0.15	0.92	0.93/0.76	0.92
WDBC	0.95/1.0	0.95	0.95/0.52	0.95	0.96/0.52	0.95	0.95/0.26	0.94	0.94/0.45	0.94	**0.96/0.23**	0.95	0.96/0.39	**0.96**
Wine	0.97/1.0	0.97	0.98/0.92	0.98	0.98/0.85	0.97	0.86/0.46	0.83	0.98/0.77	0.98	**0.98/0.54**	**0.98**	0.98/0.77	0.98
Hill-Valley	0.52/1.0	0.52	**0.55/0.05**	**0.54**	0.52/0.90	0.52	0.49/0.21	0.49	0.51/0.35	0.51	0.53/0.25	0.52	0.53/0.10	0.52
Vowel	0.91/1.0	0.9	0.90/0.92	0.9	0.83/0.50	0.82	0.79/0.58	0.77	0.80/0.91	0.8	**0.91/0.92**	**0.91**	**0.91/0.83**	0.9
Vehicle	0.83/1.0	0.83	0.81/0.75	0.81	0.80/0.88	0.8	0.67/0.62	0.66	0.76/0.88	0.76	0.83/0.87	0.83	**0.84/0.87**	**0.84**
ACC	0.99/1.0	0.99	0.99/0.67	0.99	0.99/0.83	0.99	0.99/0.67	0.99	0.99/0.68	0.99	0.99/0.70	0.99	**0.99/0.63**	**0.99**
Segmentation	**0.49/1.0**	0.49	0.48/0.89	0.48	0.47/0.89	0.47	0.41/0.67	0.41	0.41/0.78	0.41	0.48/0.78	0.48	0.48/0.89	0.48
ISOLET	0.92/1.0	0.92	0.89/0.91	0.88	0.90/0.91	0.91	0.88/0.38	0.88	0.89/0.78	0.9	**0.92/0.78**	**0.92**	**0.92/0.78**	**0.92**
IoT	0.98/1.0	0.98	0.98/0.46	0.98	0.96/0.68	0.96	0.89/0.64	0.89	0.99/0.52	0.99	**0.98/0.25**	**0.98**	**0.98/0.25**	**0.98**
Diabetes	0.79/1.0	0.78	0.79/0.75	0.79	0.78/0.88	0.78	0.76/0.50	0.76	0.68/0.75	0.68	0.79/0.75	0.79	**0.80/0.37**	**0.8**
NSL-KDD99	0.99/1.0	0.99	0.99/0.78	0.99	0.99/0.90	0.99	0.99/0.71	0.99	0.99/0.78	0.99	0.99/0.84	0.99	**0.99/0.63**	**0.99**

Table 9: Performance of FedAvg on selected features (*mean* / *ratio of feature selected*)

Dataset	All Features		RFE		ANOVA		FSHFL		Fed-mRMR		Fed-FiS		Fed-MOFS	
	F1-Score	Accuracy	F1-Score	Accuracy	F1-Score	Accuracy	F1-Score	Accuracy	F1-Score	Accuracy	F1-Score	Accuracy	F1-Score	Accuracy
Ionosphere	0.87/1	0.88	0.87/0.33	0.9	0.87/0.36	0.86	0.92/0.57	0.91	0.91/0.82	0.91	0.89/0.15	0.9	**0.94/0.21**	**0.91**
WDBC	0.94/1	0.94	0.94/0.32	0.95	0.95/0.48	0.95	0.92/0.25	0.93	0.95/0.57	0.95	**0.95/0.16**	**0.95**	**0.95/0.16**	**0.95**
Wine	0.93/1	0.92	0.87/0.53	0.94	0.93/0.61	0.96	0.93/0.38	0.93	0.96/0.84	0.96	0.91/0.46	**0.96**	0.93/0.38	0.94
Hill-Valley	0.5/1	0.51	0.53/0.1	0.5	0.5/0.35	0.51	0.51/0.21	0.5	0.50/0.35	0.44	0.51/0.1	0.51	**0.53/0.15**	**0.52**
Vowel	0.8/1	0.79	0.74/0.66	0.75	0.77/0.91	0.77	0.78/0.58	0.76	0.82/0.84	0.82	0.81/0.66	**0.83**	**0.82/0.66**	0.82
Vehicle	0.65/1	0.65	0.63/0.5	0.66	0.64/0.87	0.66	0.42/0.62	0.53	0.67/0.89	0.67	0.64/0.5	**0.67**	**0.67/0.67**	**0.67**
ACC	0.99/1	0.99	0.99/0.66	0.99	0.99/0.83	0.99	0.99/0.66	0.99	0.99/0.67	0.99	0.99/0.63	0.99	**0.99/0.46**	**0.99**
Segmentation	0.48/1	0.46	0.48/0.88	0.46	0.42/0.66	0.45	0.21/0.66	0.38	0.44/0.9	0.45	**0.48/0.66**	0.46	0.47/0.55	**0.46**
ISOLET	0.95/1	0.95	0.94/0.90	0.94	0.90/0.90	0.9	0.89/0.37	0.89	0.95/0.78	0.95	**0.95/0.77**	**0.95**	**0.95/0.77**	**0.95**
IoT	0.85/1	0.84	0.84/0.78	0.85	0.85/0.89	0.85	0.83/0.64	0.83	0.85/0.89	0.86	**0.87/0.71**	**0.88**	0.84/0.53	0.84
Diabetes	0.75/1	0.75	0.74/0.5	0.74	0.74/0.5	0.75	0.69/0.5	0.7	0.65/0.62	0.65	0.74/0.5	0.74	**0.76/0.5**	**0.77**
NSL-KDD99	0.99/1	0.99	0.99/0.76	0.99	0.99/0.81	0.99	0.99/0.71	0.99	0.99/0.85	0.99	0.99/0.68	0.99	**0.99/0.63**	**0.99**

Table 10: Performance of FedAvg on the selected features for regression tasks

Dataset/ratio of feature selected	Fed-mRMR		Fed-MOFS		Fed-FiS	
	RMSE	MAE	RMSE	MAE	RMSE	MAE
Boston/0.71	9.04	6.1	8.98	6.08	**8.96**	**6.03**
California/0.77	0.83	0.60	**0.79**	**0.57**	**0.79**	**0.57**

produced by feature selection algorithms (RFE, ANOVA, FSHFL, Fed-mRMR, Fed-FiS, and Fed-MOFS), we observed in Table 8, Fed-MOFS offers better accuracy and F1-Score than the state-of-the-art in 7 out of 12 datasets. Meanwhile, Fed-FiS performed better or equivalent when compared to Fed-MOFS in 4 datasets. For the Ionosphere dataset, FSHFL and Fed-mRMR both outperformed Fed-FiS and Fed-MOFS by 1%. In the Hill-Valley dataset, the performance of Fed-FiS and Fed-MOFS was similar, but their accuracy and F1-Scores were slightly lower (1% and 2%, respectively) compared to RFE. On the segmentation dataset, achieving the best performance requires the inclusion of all features. Fed-FiS and Fed-MOFS utilizes 78% and 89%

of the feature space, respectively to achieves 48% accuracy and F1-Score, which is just 1% lower than the optimal result. In Table 9, we observed that Fed-MOFS outperformed the state-of-the-art in 9 datasets in terms of accuracy and F1-Scores. Fed-FiS outperformed state-of-the-art on 6 and 4 datasets in terms of accuracy and F1-Score respectively. For WDBC and ISOLET, both Fed-FiS and Fed-MOFS give similar accuracy and F1-Score. For WDBC and ISOLET, both Fed-FiS and Fed-MOFS produce the same F1-Score. In the supplementary copy, we provide a set of experiments using Random Forest where we found that Fed-MOFS performed well in 7 datasets in terms of validation accuracy and F1-Score. Fed-MOFS is pretty consistent in 9 datasets, excluding Vehicle, Segmentation and Diabetes datasets. We evaluated Fed-FiS, Fed-MOFS, and Fed-mRMR on regression tasks (Table 10) using the Boston house price and California house price prediction dataset. Our results showed that when the feature subset contained 71% of the total features, Fed-FiS slightly outperformed Fed-MOFS on the Boston dataset. Conversely, Fed-MOFS performed equivalently with Fed-FiS on the California dataset by selecting 77% of the features. Both methods performed well compared to Fed-mRMR when selecting the same amount of features, while training a ridge regression model in a federated setup on both datasets.

5.2.1.2 **Performance evaluation with non-IID data divisions:** Here, we focused on the performance of Fed-FiS and Fed-MOFS on different non-IID setups given in Table 11. The table represents, each client has information from at most how many classes. We represent γ as a non-IID factor. For example, $\gamma = 0.2$ means each client has data from 2 out of 10 classes, while $\gamma = 0.8$ indicates that each client has data from 8 out of 10 classes. As γ approaches 1, the data distribution becomes more IID, with $\gamma = 1$ signifying that all clients have information about all classes, which is IID. Additionally, for each configuration, there are no overlapping samples among clients.

Table 11: Non-IID factors (γ) based on class information

Dataset	γ			
	0.2	0.5	0.8	1.0
	Number of classes per client			
IoT	4	9	14	17
ISOLET	5	13	21	26
Synthetic	8	13	20	25

- **Performance comparison with the state-of-the-art:** We compared the performance of Fed-FiS and Fed-MOFS with Fed-mRMR in Table 12 for 5 different datasets with $\gamma = 0.8$ and 100 clients. In all experiments, we fixed the proportion of features selected for each dataset. For instance, 67% of features were chosen for the vehicle dataset, 80% for the segmentation dataset, 60% for the IoT dataset, 39% for the ISOLET dataset, and 60% for the synthetic dataset. We observed from the experiments in 3 out of the 5 datasets, Fed-MOFS outperformed the Fed-FiS and Fed-mRMR. Specifically, Fed-FiS showed superior results on the IoT datasets. For the segmentation dataset, Fed-mRMR slightly surpasses (1%) both Fed-FiS and

Fed-MOFS in performance. The difference in results occurs due to the heterogeneity in the data.

Table 12: Performance of different federated feature selection algorithms with non-iid factor γ=0.8

Dataset/ratio of feature selected	Fed-mRMR		Fed-MOFS		Fed-FiS	
	Accuracy	F1-Score	Accuracy	F1-Score	Accuracy	F1-Score
Vehicle/0.67	0.57	0.61	**0.71**	**0.75**	0.7	0.74
Segmentation/0.8	**0.46**	**0.49**	0.45	0.47	0.45	0.48
IoT/0.6	0.86	0.85	0.91	0.9	**0.92**	**0.91**
ISOLET/0.39	0.92	0.91	**0.95**	**0.95**	0.94	0.94
synthetic/0.6	0.94	0.94	0.98	0.98	0.97	0.97

- **Comparative analysis of Fed-FiS and Fed-MOFS for different non-IID factors (γ) and the selected feature subset:** To compare across models and datasets, we trained two global models using the federated learning algorithms FedAvg and Federated Forest on the features selected by Fed-MOFS and Fed-FiS, results are reported in Figure 3. During the training process, 100 clients participated fully for both the IoT and synthetic datasets, while 75 clients were involved for the ISOLET dataset, maintaining full participation throughout the entire learning period. Following observations are made from there, in the ISOLET dataset (Figure 3i to 3l), Fed-MOFS outperformed Fed-FiS. For a fixed γ, Fed-MOFS achieves better accuracy with fewer features compared to Fed-FiS. However, in the IoT dataset (Figure 3e) with $\gamma = 0.5$, Fed-FiS selected better features than Fed-MOFS. In the synthetic dataset (Figure 3a, Figure 3c), the features selected by Fed-MOFS offered better performance than those selected by Fed-FiS. These observations indicate that the performance of Fed-FiS and Fed-MOFS varies across different datasets and depends on the learning algorithm employed.
- **Relation between γ and cardinality of feature subset:** In Figure 3, when we keep γ fixed and increase the feature subset (20%, 40%, 60%, 80%), the performance of learning improves as the number of features increases. Similarly, the model's performance is enhanced when we keep cardinality of feature subset fixed and increase γ from 0.2 to 0.8. A similar trend is observed for Fed-FiS. Thus, very low values of γ and feature subset fail to obtain optimal results. To enhance performance, we need to raise either γ or cardinality of the feature subsets, or both.

5.2.2 Scalability

To evaluate the scalability of Fed-FiS and Fed-MOFS, we carried out experiments with 100 clients on IoT and 75 clients on ISOLET datasets, as depicted in Figure 4. We set γ to 0.2 and varied client participation using the scale $\Delta \in$ (0.1, 0.5, 0.8, 1.0), where $\Delta = 1.0$ indicates full client participation and $\Delta = 0.1$ implies 10% client participation. Our results showed that on the IoT dataset (Figure 4a), Fed-MOFS performed better with increased client participation, achieving optimal performance

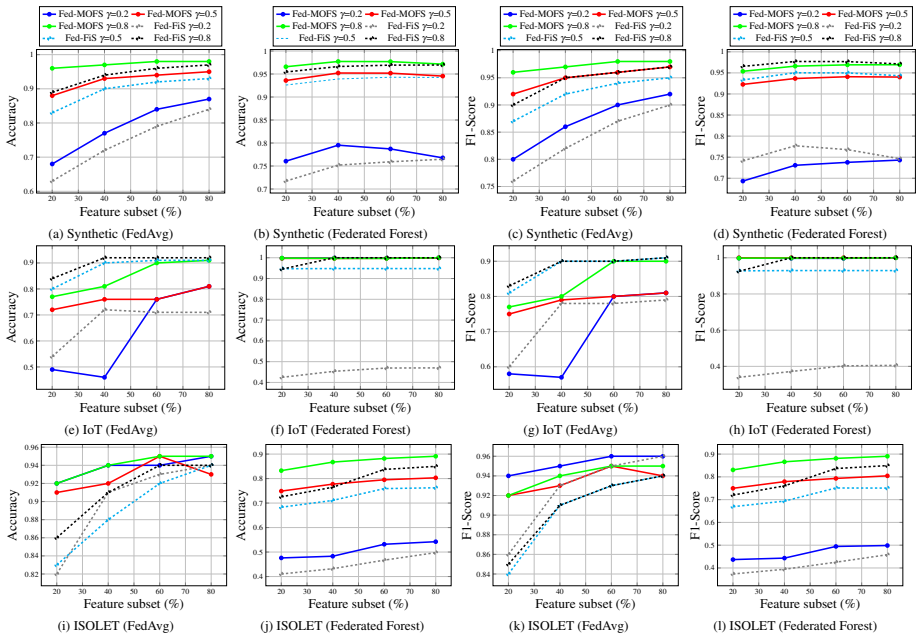


Figure 3: Performance comparison of Fed-FiS and Fed-MOFS on synthetic (Figure 3a, 3b, 3c, 3d), IoT (Figure 3e, 3f, 3g, 3h), and ISOLET (Figure 3i, 3j, 3k, 3l) datasets across different non-iid settings.

at $\Delta = 0.5$. Conversely, as shown in Figure 4b, Fed-FiS achieved the best results at $\Delta = 0.8$. For the experiments on ISOLET using Fed-MOFS (Figure 4c) and Fed-FiS (Figure 4d), increasing the value of Δ impacts the model's performance. Fed-MOFS achieved the highest accuracy and precision at $\Delta = 0.5$, and the highest recall and F1-score at $\Delta = 0.8$. For Fed-FiS, the model performed equivalently at $\Delta = 0.8$ and $\Delta = 1.0$, respectively.

From these observations, we can conclude that for Fed-MOFS, involving 50% participant in each global round is sufficient for achieving a generalized model. Similarly, for Fed-FiS, satisfactory performance is achieved with 80% client participation per global iteration. Therefore, both Fed-MOFS and Fed-FiS are effective with partial client participation.

5.2.2.1 **Effect of non-IID Factor (γ) on Partial Participation Factor (Δ):** In these experiments, we fixed the client participation factor at $\Delta = 0.1$, meaning only 10% of the total clients participate in each global iteration. For varying values of γ (0.2, 0.5, and 0.8), increasing γ led to improved performance for both Fed-FiS and Fed-MOFS. Notably, there were no significant changes in performance for both the ISOLET (Figure 5a) and IoT (Figure 5b) datasets when γ increased from 0.5 to 0.8. For the ISOLET dataset (Figure 5a), Fed-MOFS outperformed Fed-FiS, while for the

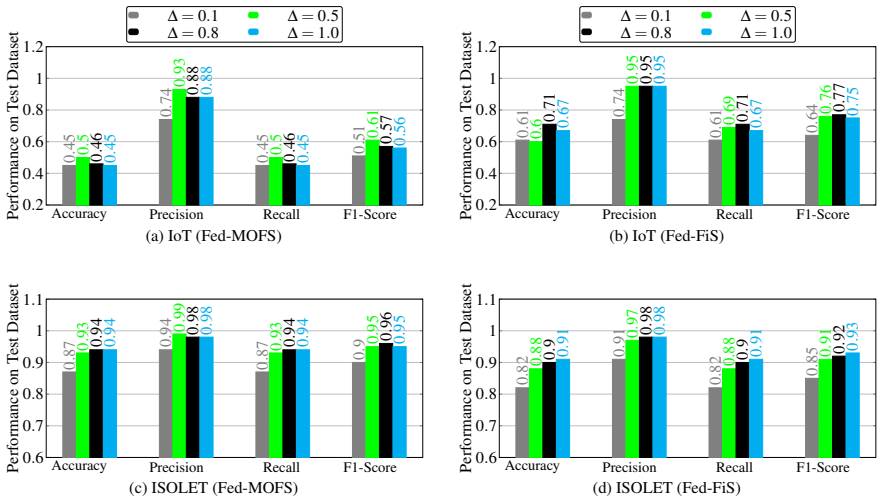


Figure 4: Performance of FedAvg with feature selection using Fed-MOFS on IoT (Figure 4a), and ISOLET (Figure 4c) datasets, compared to Fed-FiS on IoT (Figure 4b), and ISOLET (Figure 4d) datasets, for varying levels of client participation ($\Delta \in 0.1, 0.5, 0.8, 1.0$). The total number of clients is 100, with a non-IID factor (γ) of 0.2. The number of selected features is 17 for IoT (Figure 4a, 4b), and 240 for ISOLET (Figure 4c, 4d) datasets, using a DNN as the learning model.

IoT dataset (Figure 5b), Fed-FiS performed better than Fed-MOFS.

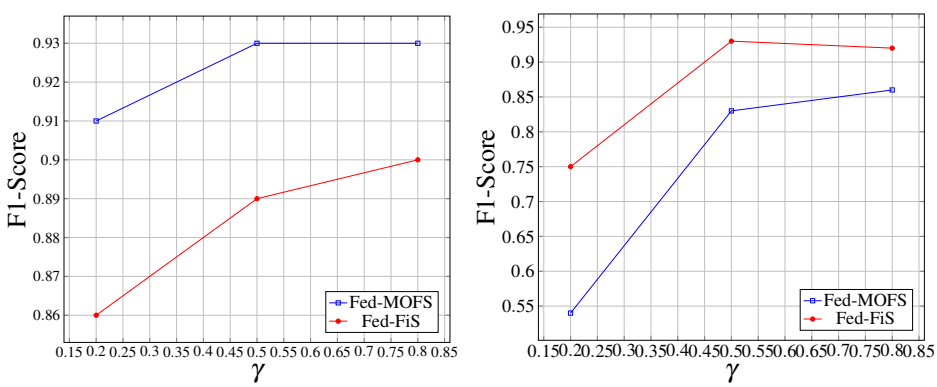
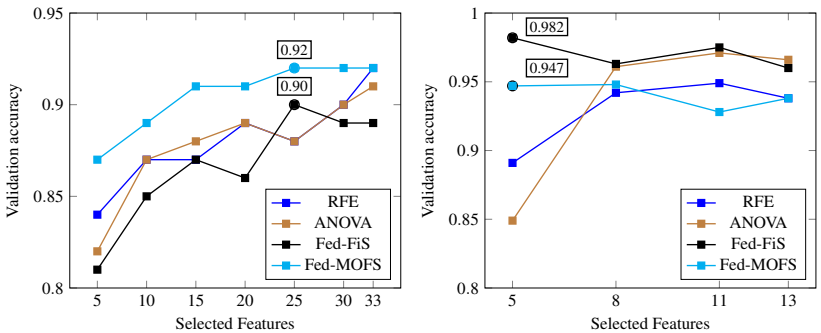


Figure 5: Performance comparison of Fed-FiS and Fed-MOFS for a fixed client participation ($\Delta = 0.1$) but varying non-IID factor $\gamma \in$ (0.2, 0.5, and 0.8)

When comparing Fed-MOFS and Fed-FiS based on client participation, Fed-FiS outperformed Fed-MOFS on IoT data (Figure 4a and Figure 4b). However, for the ISO-

LET datasets, Fed-MOFS (Figure 4c) showed better performance compared to Fed-FiS (Figure 4d). These observations suggest that the performance of the feature selection algorithm depends on the dataset, the non-IID factor (γ), and client participation (Δ).



(a) Ionosphere (b) Wine

Figure 6: Validation accuracy vs. number of selected global features

5.2.3 **Stability**

To assess stability, we conducted a comparison between Fed-FiS and Fed-MOFS against traditional feature selection methods like RFE and ANOVA. We evaluated their stability by examining the validation accuracy (Figure 6) of the global model using varying numbers of features. From Figure 6a, we observed that the ranking of features by Fed-MOFS is better than others. Based on the outcomes displayed in Figure 6b, it can be observed that both Fed-FiS and Fed-MOFS have effectively identified the top 5 significant features. As a result, their model's accuracy has demonstrated a remarkable improvement compared to other models despite reducing the feature space by over 50%. This indicates that no crucial information has been lost during the reduction process.

5.2.4 **Efficiency**

We performed a comparative analysis of the efficiency of Fed-FiS, Fed-MOFS, and FSHFL in terms of the time (wall-clock time) required for feature selection. As shown in Figure 7, our findings revealed that both Fed-FiS and Fed-MOFS demonstrate nearly identical performance in selecting the feature subset, while FSHFL takes considerably longer to achieve the same task. Particularly, in certain datasets, the difference in feature selection time is quite significant. For instance, in the ACC dataset, Fed-FiS outperformed FSHFL by 26.68 seconds, and Fed-MOFS was 26.54 seconds faster. Similarly, for IoT datasets, Fed-FiS surpassed FSHFL by 15.04 seconds, and

Fed-MOFS is 14.55 seconds faster than FSHFL. Additionally, when considering NSL-KDD99 datasets, Fed-FiS outperformed FSHFL by 23.1 seconds, and Fed-MOFS is 23 seconds faster than FSHFL.

Table 13 compares the validation accuracy vs. model size for Federated Forest. The number of estimators in random forests is predefined and remains constant; hence, to get a better understanding of the complexity of a forest, we focus on the maximum depth of trees in a forest. We computed the ratio of accuracy and depth of the forest. For 4 (Ionosphere, WDBC, Segmentation and NSL-KDD99) out of 6 datasets, Fed-MOFS produced a higher ratio than Fed-FiS and no feature selection (No-FS). And in other two datasets (Hill_valley and Vehicle) Fed-FiS produced higher ratio than No-FS.

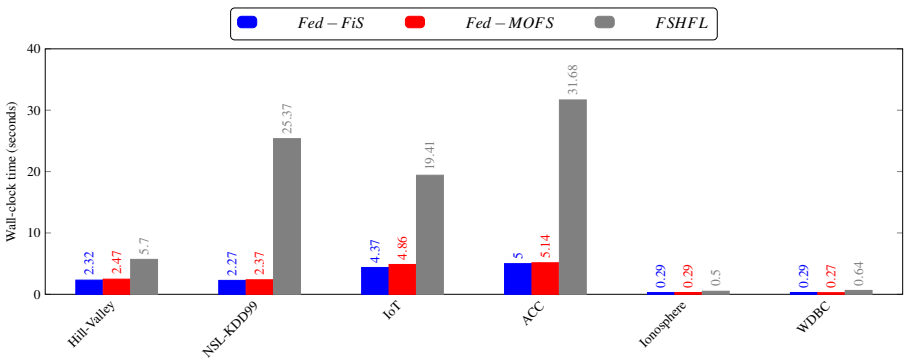


Figure 7: Wall-clock running time of federated feature selection algorithms

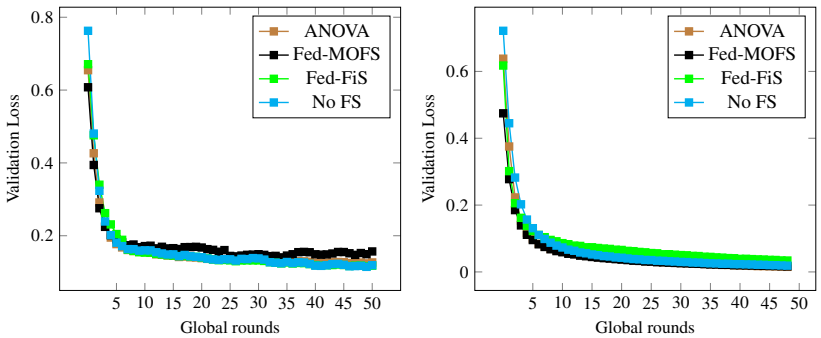
Table 13: Comparison between the performance of Federated Forest and model size

Dataset	Algorithm	Accuracy/depth	Dataset	Accuracy/Depth
	No-FS	0.091		0.136
Ionosphere	**Fed-FiS**	0.092	**WDBC**	0.134
	Fed-MOFS	**0.105**		**0.162**
	No-FS	0.019		0.031
Hill_valley	**Fed-FiS**	**0.021**	**Vehicle**	**0.033**
	Fed-MOFS	0.02		0.028
	No-FS	0.014		0.033
Segmentation	**Fed-FiS**	0.013	**NSL-KDD99**	0.030
	Fed-MOFS	**0.015**		**0.034**

5.2.5 **Convergence**

In the comparison depicted in Figure 8, we examined the convergence behavior of FedAvg on both NSL-KDD99 and WDBC datasets. Feature selection was carried out using three different methods: Fed-FiS, Fed-MOFS, and ANOVA. Additionally, we included experiments with the full dataset, where no feature selection was applied.

Our observations indicate that the convergence pattern remains consistent whether feature selection is performed or not prior to the learning process. Thus, we can conclude that the inclusion of feature selection does not hinder the convergence of the learning model.



(a) NSL-KDD99 (b) WDBC

Figure 8: Global rounds vs validation loss

6 Concluding Remarks

In this paper, we proposed Fed-MOFS and performed extensive studies on Fed-FiS and Fed-MOFS. Both are feature selection methods designed exclusively for horizontal federated learning. We made an extensive empirical, computational and communication complexity analysis of Fed-FiS and Fed-MOFS. In terms of performance, Fed-MOFS and Fed-FiS together outperformed the state-of-the-art on both classification and regression tasks. In terms of stability, Fed-FiS and Fed-MOFS show high performance while reducing more than 50% of the feature space. Regarding efficiency, Fed-FiS and Fed-MOFS are at least $2\times$ faster than FSHFL. We also observed feature selection doesn't influence the convergence of the model. Furthermore, we observed that both Fed-FiS and Fed-MOFS perform significantly well in the presence of non-IID data and partial client participation.

Federated feature selection has several potential applications, such as anomaly detection in human activity recognition, financial fraud detection, etc. In the future, we intend to implement federated feature selection that is capable of uncovering anomalies in human activity recognition with federated learning.

Acknowledgments

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

References

[AAB11] Antonio Arauzo-Azofra, José Luis Aznarte, and José M Benítez. "Empirical study of feature selection methods based on individual feature evaluation for classification problems". In: *Expert systems with applications* 38.7 (2011), pp. 8170–8177.

[AF91] Stefan Aeberhard and M. Forina. *Wine*. UCI Machine Learning Repository. acc.: 2024-06-09. 1991.

[AH21] Fatemeh Amini and Guiping Hu. "A two-layer feature selection method using genetic algorithm and elastic net". In: *Expert Systems with Applications* 166 (2021), p. 114072.

[Aha+23] Love Allen Chijioke Ahakonye et al. "SCADA intrusion detection scheme exploiting the fusion of modified decision tree and Chi-square feature selection". In: *Internet of Things* 21 (2023), p. 100676.

[al89] Sigillito V. et al. *Ionosphere*. UCI ML Repository. acc.: 2024-06-09. 1989. URL: https://doi.org/10.24432/C5W01B.

[al95] Wolberg William et al. *Breast Cancer Wisconsin (Diagnostic)*. UCI ML Repository. acc.: 2024-06-09. 1995. URL: https://doi.org/10.24432/C5DW2B.

[ALA16] Javier Apolloni, Guillermo Leguizamón, and Enrique Alba. "Two hybrid wrapper-filter feature selection algorithms applied to high-dimensional microarray experiments". In: *Applied Soft Computing* 38 (2016), pp. 922–932.

[Ant19] Taranveer Singh Anttal. *Smart Home Dataset with Weather Information*. Kaggle. acc.: 2024-06-09. 2019. URL: https://www.kaggle.com/datasets/taranvee/smart-home-dataset-with-weather-information.

[Ban+20] Sourasekhar Banerjee et al. "Multi-diseases classification from chest-x-ray: A federated deep learning approach". In: *Advances in Artificial Intelligence: 33rd Australasian Joint Conference, Canberra, ACT, Australia, November 29–30, 2020, Proceedings 33*. Springer. 2020, pp. 3–15.

[Bat94] Roberto Battiti. "Using mutual information for selecting features in supervised neural net learning". In: *IEEE Transactions on neural networks* 5.4 (1994), pp. 537–550.

[BBK16] Monowar Bhuyan, DK Bhattacharyya, and Jugal K Kalita. "A multi-step outlier-based anomaly detection approach to network-wide traffic". In: *Information Sciences* 348 (2016), pp. 243–271.

[BEB21] Sourasekhar Banerjee, Erik Elmroth, and Monowar Bhuyan. “Fed-FiS: A novel information-theoretic federated feature selection for learning stability”. In: *Neural Information Processing: 28th International Conference (ICONIP), Sanur, Bali, Indonesia, December 8–12, 2021, Proceedings, Part V*. Springer. 2021, pp. 480–487.

[BHS15] Mohamed Bennasar, Yulia Hicks, and Rossitza Setchi. “Feature selection using joint mutual information maximisation”. In: *Expert Systems with Applications* 42.22 (2015), pp. 8520–8532.

[BL97] Avrim L Blum and Pat Langley. “Selection of relevant features and examples in machine learning”. In: *Artificial Intelligence* 97.1-2 (1997), pp. 245–271.

[BR20] Veronica Bolon-Canedo and Beatriz Remeseiro. “Feature selection in image analysis: a survey”. In: *Artificial Intelligence Review* 53.4 (2020), pp. 2905–2931.

[BSC23] Sami Ben Jabeur, Nicolae Stef, and Pedro Carmona. “Bankruptcy prediction using the XGBoost algorithm and variable importance feature engineering”. In: *Computational Economics* 61.2 (2023), pp. 715–741.

[BVB22] Sourasekhar Banerjee, Xuan-Son Vu, and Monowar Bhuyan. “Optimized and Adaptive Federated Learning for Straggler-Resilient Device Selection”. In: *International Joint Conference on Neural Networks*. IEEE. 2022, pp. 1–9.

[CGV22] Pietro Cassará, Alberto Gotta, and Lorenzo Valerio. “Federated feature selection for cyber-physical systems of systems”. In: *IEEE Trans. on Vehicular Technology* 71.9 (2022), pp. 9937–9950.

[Cha+02] Nitesh V Chawla et al. “SMOTE: synthetic minority over-sampling technique”. In: *Journal of Artificial Intelligence Research* 16 (2002), pp. 321–357.

[Che+20] Chih-Wen Chen et al. “Ensemble feature selection in medical datasets: Combining filter, wrapper, and embedded feature selection results”. In: *Expert Systems* 37.5 (2020), e12553.

[Dal+17] Andrea Dal Pozzolo et al. “Credit card fraud detection: a realistic modeling and a novel learning strategy”. In: *IEEE transactions on neural networks and learning systems* 29.8 (2017), pp. 3784–3797.

[DDD90] National Institute of Diabetes, Digestive, and Kidney Diseases. *Diabetes Dataset*. Kaggle. acc.: 2024-06-09. 1990. URL: `https://www.kaggle.com/datasets/mathchi/diabetes-data-set`.

[De +14] Emiro De la Hoz et al. “Feature selection by multi-objective optimisation: Application to network anomaly detection by hierarchical self-organising maps”. In: *Knowledge-Based Systems* 71 (2014), pp. 322–338.

[Deb+02] K Deb et al. “A fast and elitist multiobjective genetic algorithm: NSGA-II”. In: *IEEE Trans. on Evolutionary Computation* 6.2 (2002), pp. 182–197.

[EB16] Naoual El Aboudi and Laila Benhlima. “Review on wrapper feature selection approaches”. In: *International Conference on Engineering & MIS*. IEEE. 2016, pp. 1–5.

[El +22] Mahmoud Said El Sayed et al. “A flow-based anomaly detection approach with feature selection method against DDoS attacks in SDNs”. In: *IEEE Transactions on Cognitive Communications and Networking* 8.4 (2022), pp. 1862–1880.

[FDA22] Vasilii Feofanov, Emilie Devijver, and Massih-Reza Amini. “Wrapper feature selection with partially labeled data”. In: *Applied Intelligence* 52.11 (2022), pp. 12316–12329.

[GE03] Isabelle Guyon and André Elisseeff. “An introduction to variable and feature selection”. In: *Journal of Machine Learning Research* 3.Mar (2003), pp. 1157–1182.

[Gui20] Yu Gui. “ADAGES: adaptive aggregation with stability for distributed feature selection”. In: *Proceedings of the ACM-IMS on Foundations of Data Science Conference*. 2020, pp. 3–12.

[Hal99] Mark A Hall. “Correlation-based feature selection for machine learning”. PhD thesis. The University of Waikato, 1999.

[Ham+22] Ahmad Hammoud et al. “On demand fog federations for horizontal federated learning in IoV”. In: *IEEE Trans. on Network and Service Management* 19.3 (2022), pp. 3062–3075.

[Haq+21] Anwar Ul Haq et al. “Forecasting daily stock trend using multi-filter feature selection and deep learning”. In: *Expert Systems with Applications* 168 (2021), p. 114444.

[HBK14] Nazrul Hoque, Dhruba K Bhattacharyya, and Jugal K Kalita. “MIFS-ND: A mutual information-based feature selection method”. In: *Expert Systems with Applications* 41.14 (2014), pp. 6371–6385.

[HBL06] Tin Kam Ho, Mitra Basu, and Martin Hiu Chung Law. “Measures of geometrical complexity in classification problems”. In: *Data complexity in pattern recognition*. Springer, 2006, pp. 1–23.

[HBL24] Jorge Hermo, Verónica Bolón-Canedo, and Susana Ladra. “Fed-mRMR: A lossless federated feature selection method”. In: *Information Sciences* 669 (2024), p. 120609.

[HHL11] Hui-Huang Hsu, Cheng-Wei Hsieh, and Ming-Da Lu. “Hybrid feature selection by combining filters and wrappers”. In: *Expert Systems with Applications* 38.7 (2011), pp. 8144–8150.

[Hu+22] Ying Hu et al. “Multiparticipant federated feature selection algorithm with particle swarm optimization for imbalanced data under privacy protection”. In: *IEEE Transactions on Artificial Intelligence* 4.5 (2022), pp. 1002–1016.

[KŁ16] Jerzy Krawczuk and Tomasz Łukaszuk. “The feature selection bias problem in relation to high-dimensional gene data”. In: *Artificial Intelligence in Medicine* 66 (2016), pp. 63–71.

[Koe23] Grandeur Koe. *Boston House Price Data*. acc.: 2024-06-09. 2023. URL: `https://www.kaggle.com/datasets/grandeurkoe/boston-house-price-data`.

[Kon+19] Linghe Kong et al. “Distributed feature selection for big data using fuzzy rough sets”. In: *IEEE Transactions on Fuzzy Systems* 28.5 (2019), pp. 846–857.

[Koz+19] Nikita Kozodoi et al. “A multi-objective approach for profit-driven feature selection in credit scoring”. In: *Decision support systems* 120 (2019), pp. 106–117.

[KR23] Nasser Khalili and Mohamad Ali Rastegar. “Optimal cost-sensitive credit scoring using a new hybrid performance metric”. In: *Expert Systems with Applications* 213 (2023), p. 119232.

[KSG04] Alexander Kraskov, Harald Stögbauer, and Peter Grassberger. “Estimating mutual information”. In: *Physical review E* 69.6 (2004), p. 066138.

[Lew92] David D Lewis. “Feature selection and feature extraction for text categorization”. In: *Speech and Natural Language: Proceedings of a Workshop*. Harriman, New York, February 23-26, 1992.

[LF08] Graham Lee and Oppacher Franz. *Hill-Valley*. UCI ML Repository. acc.: 2024-06-09. 2008. URL: `https://doi.org/10.24432/C5JC8P`.

[Li+17] Jinyan Li et al. “Elitist binary wolf search algorithm for heuristic feature selection in high-dimensional bioinformatics datasets”. In: *Scientific reports* 7.1 (2017), p. 4354.

[Li+19] Xiang Li et al. “On the convergence of fedavg on non-iid data”. In: *arXiv preprint arXiv:1907.02189* (2019).

[Li+21] Juntao Li et al. “Demand forecasting of e-commerce enterprises based on horizontal federated learning from the perspective of sustainable development”. In: *Sustainability* 13.23 (2021), p. 13050.

[Liu20] Liu, Yang and Liu, Yingting and Liu, Zhijie and Liang, Yuxuan and Meng, Chuishi and Zhang, Junbo and Zheng, Yu. “Federated forest”. In: *IEEE Transactions on Big Data* 8.3 (2020), pp. 843–854.

[LT06] Dahua Lin and Xiaoou Tang. “Conditional infomax learning: An integrated framework for feature extraction and fusion”. In: *Proc. of the 9th European Conference on Computer Vision, Graz, Austria, May 7-13, 2006, Part I 9*. Springer. 2006, pp. 68–82.

[LZL19] Haoyue Liu, MengChu Zhou, and Qing Liu. “An embedded feature selection method for imbalanced data classification”. In: *IEEE/CAA Journal of Automatica Sinica* 6.3 (2019), pp. 703–715.

[MBA17] Laura Morán-Fernández, Verónica Bolón-Canedo, and Amparo Alonso-Betanzos. “Centralized vs. distributed feature selection methods based on data complexity measures”. In: *Knowledge-Based Systems* 117 (2017), pp. 27–45.

[McM+17] Brendan McMahan et al. “Communication-efficient learning of deep networks from decentralized data”. In: *Artificial intelligence and statistics*. PMLR. 2017, pp. 1273–1282.

[MR16] Ramakrishnan Muthukrishnan and R Rohini. “LASSO: A feature selection technique in predictive modeling for machine learning”. In: *IEEE international conference on advances in computer applications*. IEEE. 2016, pp. 18–20.

[Nag+22] Senthil Murugan Nagarajan et al. “Innovative feature selection and classification model for heart disease prediction”. In: *Journal of Reliable Intelligent Environments* 8.4 (2022), pp. 333–343.

[Nak+21] Makiya Nakashima et al. “Automated feature selection for anomaly detection in network traffic data”. In: *ACM Trans. on Management Information Systems* 12.3 (2021), pp. 1–28.

[Nug17] Cameron Nugent. *California Housing Prices*. acc.: 2024-06-09. 2017. URL: `https://www.kaggle.com/datasets/camnugent/california-housing-prices`.

[Özy20] Fatih Özyurt. “Efficient deep feature selection for remote sensing image recognition with fused deep learning architectures”. In: *The Journal of Supercomputing* 76.11 (2020), pp. 8413–8431.

[Pat+22] Muhammad Salman Pathan et al. “Analyzing the impact of feature selection on the accuracy of heart disease prediction”. In: *Healthcare Analytics* 2 (2022), p. 100060.

[Pea01] Karl Pearson. “LIII. On lines and planes of closest fit to systems of points in space”. In: *The London, Edinburgh, and Dublin philosophical magazine and journal of science* 2.11 (1901), pp. 559–572.

[Qia+22] Hongyi Qian et al. “Financial distress prediction using a corrected feature selection measure and gradient boosted decision tree”. In: *Expert Systems with Applications* 190 (2022), p. 116202.

[QK21] Yang Qin and Masaaki Kondo. “Federated learning-based network intrusion detection with a feature selection approach”. In: *International Conference on Electrical, Communication, and Computer Engineering*. IEEE. 2021, pp. 1–6.

[Ras+22] ANM Bazlur Rashid et al. “Anomaly detection in cybersecurity datasets via cooperative co-evolution-based feature selection”. In: *ACM Trans. on Management Information Systems* 13.3 (2022), pp. 1–39.

[RB19] Beatriz Remeseiro and Veronica Bolon-Canedo. “A review of feature selection methods in medical applications”. In: *Computers in biology and medicine* 112 (2019), p. 103375.

[Ren+23] Chao Ren et al. “EFedDSA: An Efficient Differential Privacy-based Horizontal Federated Learning Approach for Smart Grid Dynamic Security Assessment”. In: *IEEE Journal on Emerging and Selected Topics in Circuits and Systems* (2023).

[RM94] Cole Ron and Fanty Mark. *ISOLET*. UCI MLRepository. acc.: 2024-06-09. 1994. URL: https://doi.org/10.24432/C51G69.

[SCM20] Saúl Solorio-Fernández, J Ariel Carrasco-Ochoa, and José Fco Martínez-Trinidad. “A review of unsupervised feature selection methods”. In: *Artificial Intelligence Review* 53.2 (2020), pp. 907–948.

[SE20] Majid Soheili and Amir Masoud Eftekhari-Moghadam. “DQPFS: Distributed quadratic programming based feature selection for big data”. In: *Journal of Parallel and Distributed Computing* 138 (2020), pp. 1–14.

[SIL07] Yvan Saeys, Inaki Inza, and Pedro Larranaga. “A review of feature selection techniques in bioinformatics”. In: *bioinformatics* 23.19 (2007), pp. 2507–2517.

[Son+07] Le Song et al. “Supervised feature selection via dependence estimation”. In: *Proceedings of the 24th international conference on Machine learning*. 2007, pp. 823–830.

[Sur] Kaushik Suresh. *Customer Segmentation Classification*. Kaggle. acc.: 2024-06-09. URL: https://www.kaggle.com/datasets/kaushiksuresh147/customer-segmentation.

[Swi12] Brian Swingle. “Rényi entropy, mutual information, and fluctuation properties of Fermi liquids”. In: *Physical Review B* 86.4 (2012), p. 045109.

[Tav+09] Mahbod Tavallaee et al. “A detailed analysis of the KDD CUP 99 data set”. In: *2009 IEEE symposium on computational intelligence for security and defense applications*. IEEE. 2009, pp. 1–6.

[Tri20] Shrawan Kumar Trivedi. “A study on credit scoring modeling with different feature selection and machine learning approaches”. In: *Technology in Society* 63 (2020), p. 101413.

[Tur14] Peter Turney. *Deterding Vowel Recognition Data*. OpenML. acc.: 2024-06-09. 2014. URL: https://www.openml.org/search?type=data&sort=runs&id=58&status=active.

[Var20] Various. *Vehicle Dataset*. Kaggle. acc.: 2024-06-09. 2020. URL: https://www.kaggle.com/datasets/nehalbirla/vehicle-dataset-from-cardekho.

[WWC16] Lipo Wang, Yaoli Wang, and Qing Chang. “Feature selection methods for big data bioinformatics: a survey from the search perspective”. In: *Methods* 111 (2016), pp. 21–31.

[Yan+19] Qiang Yang et al. “Federated machine learning: Concept and applications”. In: *ACM Trans. on Intelligent Systems and Technology* 10.2 (2019), pp. 1–19.

[YGX20] Binhang Yuan, Song Ge, and Wenhui Xing. “A federated learning framework for healthcare IoT devices”. In: *arXiv preprint arXiv:2005.05083* (2020).

[YM99] H Yang and John Moody. “Feature selection based on joint mutual information”. In: *Proceedings of international ICSC symposium on advances in intelligent data analysis*. Vol. 23. Citeseer. 1999.

[Zad+17] Sepehr Zadeh et al. “Scalable feature selection via distributed diversity maximization”. In: *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 31. 2017.

[Zha+18] Shichao Zhang et al. “Supervised feature selection algorithm via discriminative ridge regression”. In: *World Wide Web* 21 (2018), pp. 1545–1562.

[Zha+23] Xunzheng Zhang et al. “Federated feature selection for horizontal federated learning in IoT networks”. In: *IEEE Internet of Things Journal* 10.11 (2023), pp. 10095–10112.

Appendix A Additional Results and Analysis

Here, we discuss the datasets and the learning models in detail. After that, we have one additional result to show the superiority of the performance of Fed-FiS and Fed-MOFS over the state-of-the-art.

Appendix A.1 Datasets

In the empirical studies, we have experimented with Fed-FiS, Fed-MOFS, and state-of-the-art model FSHFL, ANOVA, and RFE on 12 datasets.

Appendix A.1.1 NSL-KDD99

The NSL-KDD [Tav+09] dataset is a refined version of the original KDD Cup 99 dataset for evaluating computer network intrusion detection systems. The KDD Cup 99 dataset was the benchmark for assessing network-based anomaly detection methods. The dataset contains 41 features. These features are derived from network traffic data and include various types of data, such as basic features of network connections, content features, and traffic features based on a two-second temporal window. The dataset has two main classes: normal (benign) and attack. However, the attack class is further divided into four major categories of network attacks, which are Denial of Service (DoS), Remote to Local (R2L), User to Root (U2R), and Probing. In this paper, we have only two classes: Benign and attack.

Appendix A.1.2 Annonymized Credit Card Fraud (ACC)

ACC dataset [Dal+17] is commonly used in machine learning and data science to detect fraudulent credit card transactions. The dataset is anonymized for privacy reasons. Sensitive personal information is either removed or transformed in a way that individual transactions cannot be traced back to individual cardholders. The dataset typically includes a mix of numerical features transformed using techniques like Principal Component Analysis (PCA). The datasets have 28 feature variables ($V_1, V_2, \ldots, V28$) and 2 classes. Class 1 represents fraudulent transactions, and 0 represents benign transactions. There are 492 frauds out of 284,807 transactions. The dataset is highly unbalanced. The fraudulent transactions account for 0.172% of all transactions. We applied SMOTE [Cha+02] to that unbalanced dataset to balance it, and after that, we distributed the data to clients.

Appendix A.1.3 Wine

Wine [AF91] is a classic dataset used in multivariate statistics and machine learning for classification tasks. These data are the results of a chemical analysis of wines grown in the same region in Italy but derived from three different cultivars. The dataset

characteristic is Tabular. It has 12 features such as Alcohol, Malic acid, Ash, Alcalinity of ash, Magnesium, Total phenols, Flavanoids, Nonflavanoid phenols, Proanthocyanins, Color intensity, Hue, OD280/OD315 of diluted wines, and Proline. The features are real and integer. The dataset has 178 samples.

Appendix A.1.4 Vowel

The primary goal of the Vowel [Tur14] dataset is to classify different spoken vowels. This dataset has 13 features, where the number of numeric features is 10 and 3 symbolic features. 990 instances and 11 classes.

Appendix A.1.5 Vehicle

The Vehicle dataset [Var20], focuses on types of car ownership. It comprises 846 samples and 9 attributes, and the target is categorized into 4 distinct classes. The dataset contained non-numeric data in features, which has been converted into categorical format for analysis. For instance, the ownership category has four types: 'First owner,' 'Second owner,' 'Third owner,' and 'Fourth or Above owner'. These have been encoded as 0, 1, 2, and 3, respectively.

Appendix A.1.6 Segmentation

This is a customer segmentation classification [Sur]. It has 10 features such as ID, Gender, Ever_married, Age, Graduated, Profession, Work_Experience, Spending_Score, Family_Size, and Var_1. The target variable is Segmentation. It has four classes. Similar to the Vehicle dataset, it has non-numeric data in some features (For example, Gender, Profession, etc.). Those are converted into categorical.

Appendix A.1.7 WDBC

This is a Wisconsin Diagnostic Breast Cancer dataset [al95]. The dataset characteristics are multivariate. The feature type is real. The dataset has 30 features and 569 samples. The dataset is useful for classification tasks. The number of classes is two (malignant and benign).

Appendix A.1.8 Ionosphere

This is a classification of radar returns from the ionosphere [al89]. The dataset characteristics are multivariate. The dataset has 34 features where the features are integer or real. The entire dataset has 351 samples.

Appendix A.1.9 Hill_valley

This dataset [LF08] has 100 features. All have floating-point values. The number of samples is 606. This dataset is used for 2 class classification problems. The class is represented in binary {0,1}. 0 means valley and 1 means hill.

Appendix A.1.10 ISOLET

This dataset [RM94] is useful for classification problems. The goal is to predict which letter name was spoken. Therefore, the target classes are 26. The characteristics of the dataset are multivariate. The number of features is 617, and each feature has real numbers. The total samples are 7797.

Appendix A.1.11 Diabetes

The objective of this dataset [DDD90] is to predict whether the patient has diabetes or not. This is a binary class classification dataset. The number of features is 8, and the number of samples is 768. All the features have a numeric value. The dataset has the following features: Number of times pregnant, plasma glucose concentration a 2 hours in an oral glucose tolerance test, Diastolic blood pressure (mm Hg), Triceps skin fold thickness (mm), 2-Hour serum insulin (mu U/ml), Body mass index (weight in $kg/(heightinm)^2$), Diabetes pedigree function, Age (years).

Appendix A.1.12 IoT

This dataset contains smart home data[Ant19]. It has 503910 samples. 28 Features and 18 classes. This dataset is useful for classification problems.

Appendix A.1.13 Synthetic data

We have used a synthetic dataset with 100,000 samples and 200 features specifically for our experiments with non-iid data. This dataset contains 35 truly informative features, and 65 features that are linearly dependent on the 35. The remaining 100 are purely noise aimed to reduce the learnability of models. A good Feature selection algorithm will be able to filter this and provide high-quality data to a model. All the data points belong to one of 25 classes With a randomly initialized class imbalance adding to the challenge of non-iidness.

For a given iid ratio (γ), a client can have samples comprising of not more than 20% of the total number of class labels if $\gamma = 0.2$. For example, if the total number of classes in the dataset is 25, a single client will have not more than floor(25*0.2) = 5 class labels. The first client may be assigned samples having the first five class labels, the second client will be assigned samples having another set of 5 class labels, and so on. Multiple checks and balances ensure that no sample is repeated in more than one client, all the samples are utilized in the distribution process, no single client has

more than (num_classes*γ) number of class labels, number of samples for each client is random.

Appendix A.2 Learning model

Appendix A.2.1 Neural Network

We created a neural network that has pass through 3 dense layer. The first hidden layer is a dense layer with 128 parameters. The second hidden layer is another dense layer with 64 parameters. The third hidden layer is a dense layer with 32 parameters. In all three hidden layer we have a ReLU activation function. We have an output layer where the number of parameters is equal to number of classes. The activation function of the output layer is either a sigmoid or a softmax, depending on whether the problem is binary classification or multi-class classification.

Appendix A.3 Performance analysis

After conducting training with distributed Random-Forest using both complete feature sets and reduced feature sets produced by feature selection algorithms (RFE, ANOVA, FSHFL, Fed-FiS, and Fed-MOFS), we observed (in Table 14) for most of the larger datasets, excluding Isolet (i.e. WDBC, HillValley, ACC, IoT, and NSL-KDD99 datasets), Fed-MOFS produced a smaller feature subset than others while maintaining a high test-accuracy. In terms of F1-Score (in Table 15), Fed-MOFS is pretty consistent across most datasets (excluding Vehicle, Segmentation, and Diabetes datasets), providing the highest F1-Score at minimal features.

Table 14: Test Accuracies of the model trained using Random-Forest Algorithm (*mean* $\pm$ *std* / *ratio of feature selected*)

Dataset	All Features	RFE	ANOVA	FSHFL	Fed-FiS	Fed-MOFS
Ionosphere	0.86±0.02/1	0.86±0.05/0.33	0.86±0.03/0.36	0.82±0.01/0.54	**0.89±0.02/0.15**	0.88±0.02/0.21
WBDC	**0.94±0.01/1**	**0.94±0.02/0.25**	**0.94±0.02/0.74**	**0.94±0.01/0.25**	**0.94±0.01/0.22**	**0.94±0.01/0.19**
WINE	0.94±0.02/1	0.94±0.03/0.53	0.94±0.03/0.61	0.91±0.01/0.46	**0.95±0.03/0.53**	**0.95±0.02/0.46**
Hill valley	0.51±0.03/1	0.50±0.02/0.45	0.51±0.03/0.65	0.51±0.00/0.21	0.51±0.01/0.25	**0.55±0.02/0.05**
Vowel	**0.80±0.02/1**	**0.80±0.02/0.83**	**0.80±0.02/0.91**	0.78±0.00/0.58	0.79±0.02/0.91	0.79±0.03/0.66
Vehicle	**0.81±0.01/1**	0.80±0.01/0.75	0.79±0.01/0.87	0.65±0.00/0.62	0.80±0.01/0.87	0.79±0.02/0.87
ACC	**0.99±0.00/1**	**0.99±0.00/0.66**	**0.99±0.00/0.7**	**0.99±0.01/0.66**	**0.99±0.00/0.7**	**0.99±0.00/0.56**
Segmentation	**0.43±0.00/1**	**0.43±0.01/0.88**	**0.43±0.01/0.88**	0.38±0.00/0.66	0.41±0.01/0.77	0.42±0.01/0.88
ISOLET	**0.90±0.00/1**	**0.90±0.00/0.64**	**0.90±0.00/0.77**	0.83±0.00/0.37	**0.90±0.00/0.77**	**0.90±0.00/0.64**
IoT	0.97±0.00/1	**0.98±0.00/0.32**	0.96±0.00/0.67	0.89±0.01/0.64	**0.98±0.00/0.25**	**0.98±0.00/0.25**
Diabetes	0.76±0.01/1	0.76±0.01/0.5	0.75±0.02/0.87	0.69±0.02/0.5	**0.77±0.01/0.75**	0.76±0.01/0.62
NSL KDD99	**0.99±0.00/1**	**0.99±0.00/0.81**	**0.99±0.00/0.86**	0.98±0.01/0.71	**0.99±0.00/0.84**	**0.99±0.00/0.65**

Table 15: F1-Scores of the model trained using Random-Forest Algorithm (*mean* $\pm$ *std* / *ratio of feature selected*)

Dataset	All Features	RFE	ANOVA	FSHFL	Fed-FiS	Fed-MOFS
Ionosphere	0.87±0.01/1	0.86±0.05/0.33	0.87±0.02/0.36	0.83±0.01/0.54	0.89±0.02/0.15	**0.94±0.02/0.21**
WBDC	0.94±0.01/1	0.94±0.02/0.25	0.94±0.01/0.74	0.94±0.01/0.25	**0.95±0.01/0.22**	0.94±0.01/0.19
WINE	0.95±0.01/1	0.95±0.03/0.53	0.95±0.03/0.61	0.93±0.01/0.46	**0.96±0.02/0.53**	**0.96±0.02/0.46**
Hill Valley	0.52±0.03/1	0.5±0.02/0.45	0.52±0.03/0.65	0.52±0.00/0.21	0.52±0.01/0.25	**0.56±0.02/0.05**
Vowel	**0.82±0.02/1**	**0.82±0.02/0.83**	**0.82±0.01/0.91**	0.78±0.00/0.58	0.81±0.02/0.91	0.80±0.03/0.66
Vehicle	**0.81±0.01/1**	0.80±0.01/0.75	0.79±0.01/0.87	0.65±0.00/0.62	0.80±0.01/0.87	0.79±0.02/0.87
ACC	**0.99±0.00/1**	**0.99±0.00/0.66**	**0.99±0.00/0.7**	**0.99±0.01/0.66**	**0.99±0.0/0.7**	**0.99±0.0/0.56**
Segmentation	**0.43±0.00/1**	**0.43±0.01/0.88**	**0.43±0.01/0.88**	0.38±0.00/0.66	0.41±0.00/0.77	0.42±0.01/0.88
ISOLET	**0.91±0.00/1**	**0.91±0.00/0.64**	**0.91±0.00/0.77**	0.84±0.00/0.37	**0.91±0.00/0.77**	**0.91±0.00/0.64**
IoT	0.97±0.00/1	**0.98±0.00/0.32**	0.96±0.00/0.67	0.89±0.01/0.64	**0.98±0.00/0.25**	**0.98±0.00/0.25**
Diabetes	0.77±0.01/1	0.76±0.03/0.5	0.76±0.02/0.87	0.69±0.02/0.5	**0.78±0.01/0.75**	0.76±0.02/0.62
NSL KDD99	**0.99±0.00/1**	**0.99±0.00/0.81**	**0.99±0.00/0.86**	**0.98±0.01/0.71**	**0.99±0.00/0.84**	**0.99±0.00/0.65**

Paper IV

Personalized Multi-tier Federated Learning

Sourasekhar Banerjee, Ali Dadras, Alp Yurtsever, and Monowar Bhuyan

Accepted for publication in the 31^{st} *International Conference on Neural Information Processing (ICONIP)*, 2024.

Personalized Multi-tier Federated Learning

Sourasekhar Banerjee*, Ali Dadras†, Alp Yurtsever†, Monowar Bhuyan*

* *Department of Computing Science, Umeå University, Umeå, Sweden*
† *Department of Mathematics and Mathematical Statistics, Umeå University, Umeå, Sweden*
sourasb@cs.umu.se, ali.dadras@umu.se, alp.yurtsever@umu.se, monowar@cs.umu.se

Abstract: The key challenge of personalized federated learning (PerFL) is to capture the statistical heterogeneity properties of data with inexpensive communications and gain customized performance for participating devices. To address these, we introduced personalized federated learning in multi-tier architecture (PerMFL[1]) to obtain optimized and personalized local models when there are known team structures across devices. We provide theoretical guarantees of PerMFL, which offers linear convergence rates for smooth strongly convex problems and sub-linear convergence rates for smooth non-convex problems. We conduct numerical experiments demonstrating the robust empirical performance of PerMFL, outperforming the state-of-the-art in multiple personalized federated learning tasks.

Key words: Personalized federated learning, Multi-tier federated learning, Hierarchical federated learning.

1 Introduction

Federated learning (FL) is a distributed on-device learning framework that employs the heterogeneous data privately available at the edge for learning. In classical machine learning, edge devices are supposed to send data to the centralized server for training. However, FL relaxes this restriction by enabling the training of each model on the end devices and aggregating them on the global server. Classical FL learns a single global model by utilizing the private data of the devices locally and exchanging only the model information in a communication-efficient and privacy-preserving manner [McM+17].

The early FL literature focuses mainly on a simplistic network architecture where all devices communicate directly with a single server [McM+17]. An example of such a network architecture is typical on a local area network (LAN)

[1] `https://github.com/sourasb05/PerMFL_1.git`

with all devices connected to a single server. However, real-world Internet applications often occur on more complex, multi-tiered network architecture, e.g., wide area networks (WAN) that span multiple geographic locations and connect heterogeneous LANs. Hence, the conventional FL architecture is not suitable for such a setting. To address this, we study a multi-tier FL model Figure 1b where an intermediate layer of mediators, called *team servers (TS_i)*, acts as a bridge between the end devices (N_i) and the global server (GS) and facilitates intermediate aggregations. From a systems standpoint, this multi-tier FL model resembles the cloud-edge continuum architecture [Wu+21]. In this model, the distant cloud serves as a central server responsible for generating a global model, and the edge servers located in various geographical regions act as team servers that establish connections with both the distant cloud and end devices. In contrast, the end devices perform local model computations. Multi-tier FL models, which are also referred to as hierarchical FL, have been studied by researchers in various applications and have shown significant advantages such as cost efficiency and scalability [Ekm+22], reduced communication overheads [Aba+20; Liu+22], enhanced privacy [Wai+20], improved system adaptability and performance [Lin+20], and faster convergence speeds (both theoretically and empirically) and reduced training time [Liu+20].

Data heterogeneity poses a significant challenge in FL, as data across different devices may exhibit varying characteristics and originate from diverse data distributions. The conventional FL systems assume that a single global model fits all devices, hindering the convergence speed and more crucially, the model accuracy when data is disseminated in a non-independent and identically distributed (non-IID) manner. In this scenario, using a 'global model for all' disallows adaptation to unique needs and preferences embedded in each user's data characteristics, possibly leading to subpar performance and user dissatisfaction [Tan+22]. To tackle this challenge, personalized FL (PerFL) methods learn local models suitable for each user's unique needs for downstream tasks while still benefiting from collaborative training to achieve customized performance. Popular PerFL approaches include regularization techniques that penalize the distance between local and global models [Li+21], smoothing techniques such as Moreau envelopes [TTN20], and other heuristics such as taking extra local steps after the global model has converged [FAL17; FMO20].

Multi-tier FL also suffers from data heterogeneity, but the existing literature on personalized multi-tier FL is limited. To address this challenge, we introduce a personalized multi-tier federated learning method (PerMFL). Our method is based on a new problem formulation that explicitly incorporates individual models for each team and device, in addition to the global server's model. We enforce the proximity between team models and the global model and between device models and their associated team models using squared Euclidean distance regularization. As a result, our method leverages the multi-tiered architecture and simultaneously learns three models: (1) a global model, (2) a personalized model for each team, and (3) a personalized model for each device. Geometrically, the global model serves as a central estimate that is agreed upon by all teams

and end devices. On the contrary, personalized models are designed to deviate from the global model in specific directions that align with their local data distributions. To facilitate efficient communication, our algorithm restricts direct communication between devices and the global server, allowing devices to communicate only with the team servers, which in turn communicate with the global server. Our primary goal with PerMFL is to achieve personalized on-device model performance while still maintaining comparably high accuracy for the global model, which can compete with conventional FL methods, all while ensuring efficient communication and collaboration among the devices and team servers.

Our key **contributions** are as follows:

1. We formulate an optimization problem for multi-tier personalized FL by introducing personal decision variables for teams and devices through squared Euclidean distance regularization, and we propose an algorithm (PerMFL) to solve this problem. Our algorithm flexibly accommodates local objectives during joint training of global and personalized models.
2. To provide a theoretical basis for our approach, we analyze the convergence guarantees of the proposed algorithm under the assumptions of smooth strongly convex and smooth non-convex loss functions. We show that the method converges with linear and sublinear rates, respectively, and we derive explicit theoretical bounds on hyperparameter settings to provide guidance for implementation.
3. We conduct extensive numerical experiments to evaluate the empirical performance of PerMFL. We compare our method to state-of-the-art (SOTA) approaches for both conventional and multi-tier FL settings using benchmark datasets (MNIST, FMNIST, EMNIST-10, FEMNIST,CIFAR100) and non-image tabular synthetic datasets with non-IID data dissemination. Moreover, we have examined the effect of hyperparameters on the convergence of PerMFL, ablation studies on different team formations, and ablation studies on team and client participation on the convergence of PerMFL.

2 Related Work

This section reviews existing studies and summarizes the differences between existing works and proposed efforts. We emphasize three different categories of FL models - multi-tier and personalized FL.

Multi-tier FL: A multi-tier (*aka* hierarchical) FL model leverages the combined capabilities of cloud and edge devices within the FL framework. In [Liu+20], the authors demonstrated that a multi-tier FL strategy, both theoretically and empirically, exhibits a faster convergence rate compared to

traditional FL algorithms. In [Wan+22], the concept of "upward" and "downward" divergences were introduced, exploring their implications in the context of multi-tier FL. Additionally, [Das+22] presented Cross-Silo FL, which encompasses multi-tier networks and utilizes vertical and horizontal data partitioning strategies. In [Ekm+22], they introduced FEDn, a multi-tier FL framework designed explicitly for horizontally scalable distributed deployments. [Wai+20] identified several potential advantages of multi-tier FL in addressing privacy concerns. Firstly, multi-tier FL helps reduce the concentration of power and control in a central server, promoting a more distributed and decentralized approach. Secondly, multi-tier FL allows for the flexible placement of defense and verification mechanisms within the hierarchical structure, enabling the practical application of these methods. Lastly, multi-tier FL leverages the trust between users to mitigate the number of potential threats.

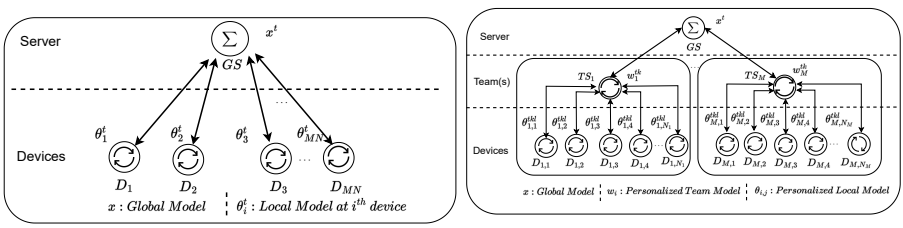
Personalized FL: FedAvg [McM+17] is a classical baseline method in FL, renowned for its simplicity and low communication cost. However, it faces challenges when dealing with heterogeneous (non-iid) data, leading to unstable performance due to the concept drift problem. Concept drift arises when a single global model does not perform well for all clients. To address these issues, FL has increasingly leaned toward personalized models [Kar+20; Man+20; Tan+22]. In [FAL17], the authors proposed a personalized version of model-agnostic meta-learning (MAML) for FL. They identified the problem of how fast the initial shared model adapts to their local dataset with fewer gradient descent steps using individual client data. In [Man+20], a systematic learning-theoretic study of personalization led to the proposal of three model-agnostic approaches: user clustering, data interpolation, and model interpolation. Another approach, presented in [TTN20], formulated a personalized FL problem that utilized Moreau envelopes to regularize devices' loss functions. In [LHK22], the authors introduced an Asynchronous Loopless Local Gradient Descent (Async-L2GD) method for users from multiple known clusters, simultaneously training three models: a global model, a model-specific cluster, and a personalized model for each device. The architecture is similar to PerMFL, except that L2GD is an asynchronous approach. [Ngu+22] introduced DemLearn, an FL algorithm that employs hierarchical agglomeration clustering. Unlike our model, where teams remain static throughout the FL process, DemLearn dynamically assembles teams after each global round. Recent research on personalized FL also includes works such as [Zha+22; Gas+22; Pil+22; Che+22; Tan+22; Tzi+22; Li+21; Zha24; Zha+24].

Motivation: A hierarchical structure in FL addresses the scalability and failure tolerance limitations observed in the centralized architecture [Wai+20]. In addition, it is also beneficial in addressing the management difficulties, system adaptivity, and performance [Lin+20] that arise due to fully decentralized architecture [Wai+20]. The key challenge of personalization is the heterogeneity of data, locally customized models, and identifying and collaborating among clients those having similar information [LHK22]. By adding personalization at team and device levels, we aim to capture customized and refined personalized

properties of the model that align well with real-world applications [Wu+21; Zho+23]. Communication with the global server is often the most expensive step in FL [Abd+22], and communications within a team are typically cheaper [Liu+20]. By employing the multi-tier architecture and accommodating a large portion of the communication within the teams, PerMFL economizes significantly on the communication iterations with the global server, preventing biases to local clients, and achieving faster convergence [Wai+20]. These advantages motivated us to propose multi-tier FL.

3 PerMFL

A multi-tier FL framework (see Figure 1b) is different from the conventional FL framework (see Figure 1a) as it follows a hierarchy. All devices are divided into M teams. Each team has a team server (TS_i), which is connected with N_i devices of the respective team. Global server (GS) only communicates with TS_i's team.



(a) Conventional FL framework (b) Multi-tier FL framework

Figure 1: Federated learning work-flow

3.1 Team formation

By definition, FL can be seen as a cross-silo or cross-device setting [Kai+21]. In cross-silo settings, a limited number of devices, typically between 2 and 100, participate in each round, and their participation remains fixed. On the other hand, in cross-device setups, the client pool is extensive, potentially in the millions, but only a small fraction of devices take part in each iteration [Kai+21]. In a multi-tier structure, teams can be constituted in four ways: (1) Both teams and devices within teams have full participation, (2) Teams have full participation, whereas devices within teams are participating partially, (3) Teams have partial participation, but all devices within teams are participating, and (4) Teams and devices both have partial participation.

Various FL methods with different team formation strategies have been proposed in the literature [Lon+23; YTW23]. PerMFL does not explicitly address the creation of teams. Instead, it exhibits adaptability to accommodate any team formation mechanism. It is important to note that PerMFL is most efficient when communication with the team servers is cheaper compared to communication with the global server. This situation often occurs when devices

are geographically distributed and connected to their nearest teams, similar to the Cloud-Edge model [Bit+18]. We can also argue that team-level personalization is more effective when the data within each team exhibits distinctive characteristics that differentiate it from the data in other teams. Nonetheless, we provide a comprehensive ablation study that thoroughly examines different types of team selection, including poorly constructed and randomly constructed team formations. In cases where teams do not exhibit such distinctive characteristics, our model can still benefit from device-level personalization and reduced communication costs with the global server.

3.2 PerMFL formulation

We consider a multi-tier FL setup, consisting of M teams, each with N_i devices $(i = 1, \ldots, M)$. In this setup, empirical risk minimization can be expressed as:

$$\min_{x \in \mathbb{R}^d} \quad \frac{1}{M} \sum_{i=1}^{M} \frac{1}{N_i} \sum_{j=1}^{N_i} f_{i,j}(x). \tag{1}$$

Here, $f_{i,j}(\cdot)$ represents the loss function of the j^{th} device from the i^{th} team. This formulation relies on a single decision variable x that all clients are expected to converge upon. However, this model is unsuitable for scenarios involving non-homogeneous data distributions, as the existence of a global model capable of accommodating all devices becomes unreasonable.

To address this challenge, we introduce decision variables for every team and device, denoted as w_i and $\theta_{i,j}$, respectively. Our goal is to find a global model representing a rough average, capturing common characteristics across all teams and devices; personalized team models that are close to the global model but can deviate from aligning with shared features within each team; and personalized device models that resemble the team model but can deviate to accommodate the unique characteristics of each device. We adopt a quadratic penalty approach to enforce that the device models are close to the team models and the team models to the global model. The parameters $\gamma \geq 0$ and $\lambda \geq 0$ control the degree of personalization impact at the team and device levels respectively.

$$\min_{x \in \mathbb{R}^d} \min_{w_i \in \mathbb{R}^d} \min_{\theta_{i,j} \in \mathbb{R}^d} \frac{1}{M} \sum_{i=1}^{M} \frac{1}{N_i} \sum_{j=1}^{N_i} \Big(f_{i,j}(\theta_{i,j}) + \frac{\lambda}{2} \|\theta_{i,j} - w_i\|^2 + \frac{\gamma}{2} \|w_i - x\|^2 \Big). \tag{2}$$

We are now prepared to outline the algorithm design. Our approach involves the use of three iteration counters: t for the global rounds, k for team-level rounds, and ℓ for device-level rounds.

Device-level updates. Given a team model $w_i^{t,k}$, the objective of the device (i, j) is to solve the following subproblem:

$$\tilde{f}_{i,j}^{\lambda}(w_i^{t,k}) := \min_{\theta_{i,j} \in \mathbb{R}^d} \; f_{i,j}(\theta_{i,j}) + \frac{\lambda}{2} \|\theta_{i,j} - w_i^{t,k}\|^2 \tag{3}$$

The exact solution to this problem is $\text{prox}_{f_{i,j}/\lambda}(w_i^{t,k})$; however, in general, there is no closed-form solution available for this proximal operator. Consequently, each device employs the gradient method to approximate the solution to Equation (3). Starting with an initial value of $\theta_{i,j}^{t,k,0} = w_i^{t,k}$, and utilizing a positive step-size α, we perform the following update rule for $l = 0, 1, \ldots, L-1$:

$$\theta_{i,j}^{t,k,l+1} = \theta_{i,j}^{t,k,l} - \alpha_{i,j}\nabla f_{i,j}(\theta_{i,j}^{t,k,l}) - \alpha_{i,j}\lambda(\theta_{i,j}^{t,k,l} - w_i^{t,k}). \tag{4}$$

Team-level updates. Similarly, the goal in team-level updates is to solve a regularized subproblem. We define the team-level loss function as

$$F_i(w_i) := \frac{1}{N_i}\sum_{j=1}^{N_i} \tilde{f}_{i,j}^{\lambda}(w_i). \tag{5}$$

With the addition of regularization towards the global server's model x^t, Team (i) aims to solve the following subproblem:

$$clust\tilde{F}_i^{\gamma}(x^t) := \min_{w_i \in \mathbb{R}^d} \; F_i(w_i) + \frac{\gamma}{2}\|w_i - x^t\|^2. \tag{6}$$

Once again, the solution is given by the proximal operator, $\text{prox}_{F_i/\gamma}(x^t)$, which can be difficult to compute. Instead, we can find an approximate solution by using the gradient method. Starting from $w_i^{t,0} = x^t$, and using a positive step-size $\eta > 0$, the gradient method update becomes

$$\begin{aligned} w_i^{t,k+1} &= w_i^{t,k} - \eta_i \nabla F_i(w_i^{t,k}) - \eta_i\gamma(w_i^{t,k} - x^t) \\ &= w_i^{t,k} - \frac{\eta_i}{N_i}\sum_{j=1}^{N_i} \nabla \tilde{f}_{i,j}^{\lambda}(w_i^{t,k}) - \eta_i\gamma(w_i^{t,k} - x^t). \end{aligned} \tag{7}$$

Here, the second line follows from the definition of F_i. Since we do not have the exact gradient $\nabla \tilde{f}_{i,j}(w_i^{t,k})$, we approximate it by:

$$\nabla \tilde{f}_{i,j}^{\lambda}(w_i^{t,k}) = \lambda\big(w_i^{t,k} - \text{prox}_{f_{i,j}/\lambda}(w_i^{t,k})\big) \approx \lambda(w_i^{t,k} - \theta_{i,j}^{t,k,L}). \tag{8}$$

By combining Equation (7) and Equation (8), we construct the following update rule for $k = 0, 1, \ldots, K-1$:

$$w_i^{t,k+1} = \big(1 - \eta_i(\lambda+\gamma)\big)w_i^{t,k} + \eta_i\gamma x^t + \frac{\lambda\eta_i}{N_i}\sum_{j=1}^{N_i}\theta_{i,j}^{t,k,L}. \tag{9}$$

Server-level updates. Finally, the server applies the gradient method for

$$\min_{x \in \mathbb{R}^d} \quad \phi(x) := \frac{1}{M}\sum_{i=1}^{M} \tilde{F}_i^{\gamma}(x). \tag{10}$$

Starting from an initial state $x^0 \in \mathbb{R}^d$ and using a positive step-size β, the gradient update becomes:

$$x^{t+1} = x^t - \frac{\beta}{M} \sum_{i=1}^{M} \nabla \tilde{F}_i^{\gamma}(x^t). \tag{11}$$

Again, we use an approximation for $\nabla \tilde{F}_i(x^t)$ based on the team-level models:

$$\nabla \tilde{F}_i^{\gamma}(x^t) = \gamma\big(x^t - \text{prox}_{F_i/\gamma}(x^t)\big) \approx \gamma(x^t - w_i^{t,K}). \tag{12}$$

Finally, we construct the server-level update rule by combining Equation (11) and Equation (12). For $t = 0, 1, \ldots, T-1$, the server performs the following update rule:

$$x^{t+1} = (1 - \beta\gamma)x^t + \frac{\beta\gamma}{M} \sum_{i=1}^{M} w_i^{t,K}. \tag{13}$$

Synthesis. By combining device, team, and server-level updates, we propose PerMFL (Algorithm 1) for personalized multi-tier FL.

Algorithm 1 PerMFL : Personalized Multi-tier FL

Input : x^0
Output : $\forall_{i=1}^{M} \forall_{j=1}^{N_i} \theta_{i,j}^{T,K,L}$, x^T
Initialize : $\forall_{i=1}^{M} w_i^{0,0} = x^0$, $\forall_{i=1}^{M} \forall_{j=1}^{N_i} \theta_{i,j} = w_i^{0,0}$, T,K,L,$\alpha_{i,j}$, β, γ, λ, η_i

for $t = 0, 1, \ldots, T-1$ **do** // Global iterations
  global server sends x^t to the teams. // Global model
  for $k = 0, 1, \ldots, K-1$ **do** // Team iterations
    Teams send $w_i^{t,k}$ to the devices. // Team-level personalized model
    for $l = 0, 1, \ldots, L-1$ **do** // Local iterations
      $\theta_{i,j}^{t,k,l+1} = \theta_{i,j}^{t,k,l} - \alpha_{i,j} \nabla f_{i,j}(\theta_{i,j}^{t,k,l}) - \alpha_{i,j} \lambda(\theta_{i,j}^{t,k,l} - w_i^{t,k})$ // Personalized local models
    end for
    $\bar{\theta}_i^{t,k} = \frac{1}{N_i} \sum_{j=1}^{N_i} \theta_{i,j}^{t,k,L}$ // Aggregation within a team
    $w_i^{t,k+1} = (1 - \eta_i\lambda - \eta_i\gamma) w_i^{t,k} + \eta_i\gamma x^t + \lambda\eta_i \bar{\theta}_i^{t,k}$ // Personalized team update
  end for
  $\bar{w}^t = \frac{1}{M} \sum_{i=1}^{M} w_i^{t,K}$ // Global aggregation
  $x^{t+1} = (1 - \beta\gamma)x^t + \beta\gamma\bar{w}^t$ // Global update
end for

Initialization: Global server initializes the global model (x^0). Every team connected to the global server, copies the global model to their respective team model ($\forall_{i=1}^{M} w_i^{0,0} = x^0$). Each device within a team copies the initial team model ($w_i^{0,0}$) as their local model ($\theta_{i,j}$). Global server initializes the total number of global iterations, team iterations, and local iterations as T, K, and L, respectively.

Iterations: At each global iteration t, the global server broadcasts the global model x^t to every team. Similarly, at each team iteration k, each team broadcasts $w_i^{t,k}$ to all the devices within the team. For each local iteration (l), each device solves Equation (3) separately, but in parallel to obtain the personalized model

$\theta_{i,j}^{t,k,l}$. The team server of each team collects the device updates from the respective devices after L local iterations and performs aggregation ($\bar{\theta}_i^{t,k}$) on the device updates. Each team broadcasts the updated team-level model to the devices registered with that team and continues steps 3 to 10 for the next $K-1$ team iterations. After all teams finished K team iterations, the global server collects team updates ($w_i^{t,K}$) from each team and performs averaging ($\bar{w}^t$) on team updates over M teams. The global server produces a global update (x^t) by solving Equation (13). The global server broadcasts the updated global model to teams and continues steps 1 to 13 for the upcoming $T-1$ global iterations.

Remark 1. *The quadratic penalty approach leads to the concept of Moreau envelopes, a mathematical tool frequently employed in optimization theory for smoothing functions. For a function $g : \mathbb{R}^d \to \mathbb{R}$, we define the Moreau envelope $\tilde{g}^\sigma : \mathbb{R}^d \to \mathbb{R}$ with parameter $\sigma \geq 0$ as*

$$\tilde{g}^\sigma(x) := \min_{u \in \mathbb{R}^d} \Big\{ g(u) + \frac{\sigma}{2}\|u - x\|^2 \Big\}. \tag{14}$$

Clearly, we can interpret Equation (3) *and Equation* (6) *as the Moreau envelopes of $f_{i,j}$ and F_i, respectively. It is important to note that Moreau envelopes have been used before in [TTN20] for personalization in the conventional FL setting.*

3.3 Convergence guarantees

This section presents the convergence guarantees[2] of PerMFL. We consider two different settings, with strongly convex and non-convex loss functions. In both cases, we assume that the loss functions are smooth in the sense that they have Lipschitz continuous gradients. The next Theorem formalizes the guarantees of when $f_{i,j}$ is strongly convex.

Theorem 1 (Strongly convex)**.** *Consider the minimization problem $\min_x \phi(x)$ when $\phi(x)$ is defined in Equation* (10) *with L_f-smooth and μ_f-strongly convex loss functions $f_{i,j}(x)$. For large enough numbers of inner iterations of orders $L = \Omega\big(K\big)$ and $K = \Omega\big(T\big)$, see the supplementary copy for the bounds, estimation $\{x^t\}_{t=0}^T$ generated by PerMFL with step-size β satisfies:*

$$\|x^T - x^*\|^2 \leq 2(1-\beta)^T \|x^0 - x^*\|^2. \tag{15}$$

where learning rates should satisfy $\beta \leq \frac{\mu_{\tilde{F}}}{4\gamma}$, $\eta_i \leq \frac{1}{2(\lambda+\gamma)}$, $\alpha_{i,j} \leq \frac{1}{L_f+\lambda}$, $\mu_{\tilde{F}} := \frac{\lambda\gamma\mu_f}{\lambda\mu_f+\gamma\mu_f+\lambda\gamma}$, and $\gamma > 2\lambda > 4L_f$.

Proof sketch. We analyze the algorithm in three levels: *(i)* The devices find approximate solutions to problem (Equation (3)) by using the gradient

[2]The proofs of the convergence guarantees are given in `https://arxiv.org/abs/2407.14251`

method. We can control the accuracy of this stage by choosing L (the number of iterations for the gradient method) large enough. *(ii)* The teams solve problem (Equation (6)) approximately, again by using a gradient method. We use an inexact gradient at this stage since the exact gradient requires exact solutions from the devices to which we do not have access. At this stage, K is the number of iterations, and L modulates the accuracy of our gradients. By choosing both K and L large enough, we can control the solution accuracy achieved at the teams' level. *(iii)* Finally, the Server solves the problem (Equation (10)), the original FL problem, by using the information provided by the Teams. It is worth noting that for a fixed T, we can decrease the error (down to a threshold) by increasing K and L. More precisely, we achieve linear convergence rates when we choose K and L in the order of $\Omega(T)$.

The next Theorem shows that PerMFL finds a first-order stationary point with sublinear rates when $f_{i,j}$ are smooth but non-convex.

Theorem 2 (Non-convex)**.** *Consider the minimization problem* $\min_x \phi(x)$ *when* $\phi(x)$ *is defined in Equation* (10) *with non-convex* L_f*-smooth loss functions* $f_{i,j}(x)$*. For large enough numbers of inner iterations of orders* $L = \Omega\big(K\big)$ *and* $K = \Omega\big(T\big)$*, see the appendix for the bounds, then, estimation* $\{x^t\}_{t=0}^T$ *generated by PerMFL with step-size* β *satisfies:*

$$\mathbb{E}\big[\|\nabla\phi(x^{\tilde{t}})\|^2\big] \leq \frac{\phi(x^0) - \phi(x^*)}{\beta T} \tag{16}$$

where $\beta \leq \frac{1}{4\gamma}$, $\eta_i \leq \frac{1}{\lambda+\gamma}$, $\alpha_{i,j} \leq \frac{1}{\lambda}$, $\gamma > 2\lambda > 4L_f$*, and* $\tilde{t}$ *is uniformly sampled from* $\{0, \ldots, T-1\}$

Proof sketch. The analysis follows a similar structure to the previous setting. The main difference is that the errors in subproblems are guaranteed as a bound on the gradient norms, which we translate to error bounds on objective residual by tuning λ and γ.

Remark 2. *At first glance, it may come as a surprise that our guarantees do not necessitate a bounded drift condition, which is common in FL methods involving multiple local steps. It is important to note a fundamental distinction between the conventional FL template Equation* (1) *and our personalized multi-tier FL template Equation* (2)*. The former lacks consideration for data heterogeneity as it relies solely on a single global variable. This can result in 'drift-away' issues when multiple local steps are taken, especially in the presence of data heterogeneity. In contrast, our formulation explicitly incorporates team and device-level variables, and our local steps are tailored to solve device and team-level subproblems Equation* (3) *and Equation* (6)*. The regularization employed in these subproblems prevents 'over-drifting' by explicitly penalizing the divergence between device, team, and global models in a suitable manner.*

4 Experiments

We studied classification problems to validate PerMFL using both benchmarks (MNIST [LeC+98], FMNIST [XRV17] EMNIST [Coh+17] with 10 classes (EMNIST-10), EMNIST with 62 classes (FEMNIST), CIFAR100 [KH09]) and non-image synthetic datasets. For the MNIST, FMNIST, EMNIST-10, and synthetic datasets, the data was distributed among multiple devices in a non-iid manner, ensuring each device had data from at most two classes. Subsequently, the devices were randomly grouped into four teams, each consisting of 10 devices, before performing PerMFL. We considered the full participation of teams and devices in each global round for the performance and convergence evaluation. For the FEMNIST and CIFAR100 datasets, the data was distributed to 3,500 and 350 devices, respectively, with each device holding data from the 3 classes. The devices are arranged into 5 teams. All datasets are split into training and validation sets with a 3:1 ratio.

We considered a multi-class logistic regression (MCLR) model with a softmax activation function for strongly convex scenarios. For synthetic datasets, we constructed deep neural networks with two hidden layers, while for image datasets, we built two-layered convolutional neural networks for non-convex scenarios. More details of the experimental setup, datasets, and learning models are in the supplementary copy.

In this paper, we conducted (1) the performance comparison between PerMFL with FedAvg [McM+17], pFedMe[TTN20], Per-FedAvg [FMO20], pFedBayes [Zha+22], Ditto [Li+21], and performance and convergence comparison with two hierarchical FL algorithms, such as hierarchical-SGD (h-SGD)[Liu+22], Asynchronous L2GD (AL2GD)[LHK22], and a hierarchical-clustered FL algorithm DemLearn [Ngu+22]. (2) We investigated the impact of β, γ, and λ on the convergence of PerMFL. (3) An ablation study to explore team formation. (4) An ablation study to analyze the influence of team and device participation. Moreover, in the supplementary copy, we gave an ablation study that explores the effects of team iterations on the convergence of PerMFL. Throughout the experiments, we denoted the personalized model and global model as (PM) and (GM), respectively We have made the implementation of PerMFL available at `https://github.com/sourasb05/PerMFL_1.git`

4.1 Results and Analysis

4.1.1 Performance

From Table 1, we observed that PerMFL(PM) outperformed the state-of-the-art for non-convex cases in all datasets. PerMFL(GM) outperformed other global models, including FedAvg(GM), pFedMe(GM), and Ditto(GM), and nearly equivalent with h-SGD(GM) and DemLearn(GM) for the MNIST dataset. For the synthetic dataset, PerMFL(GM) outperformed the state-of-the-art. For FMNIST and EMNIST-10 datasets, the performance of PerMFL(GM) is better than the conventional FL models and DemLearn(GM). For strongly

convex cases, PerMFL(PM) outperformed the state of the art in MNIST and Synthetic datasets. For the FMNIST dataset, PerMFL(PM)'s performance is better than the conventional FL state-of-the-art. Moreover, the performance of PerMFL(PM) is nearly equivalent to the DemLearn(PM) for FMNIST and EMNIST-10. PerMFL(PM) also achieved better performance than h-SGD and AL2GD on FEMNIST and CIFAR100 given in the supplementary copy. PerMFL(GM) outperformed the state-of-the-art in Synthetic and FMNIST datasets. PerMFL(GM) performs better than conventional methods in all datasets and is nearly equivalent to h-SGD(GM) on MNIST and EMNIST-10. From these observations, we can infer that PerMFL(PM) performs better than the 7 state-of-the-art methods on 6 out of 8 experiments. The reason could be, the personalization in both team and devices helps to get better performance.

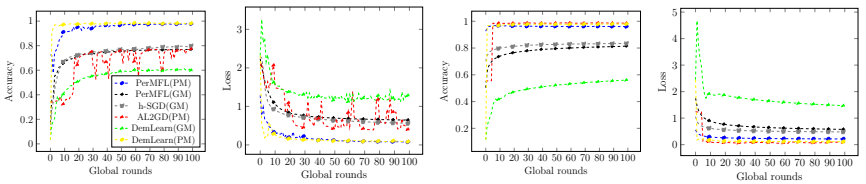
Table 1: Performance (Validation accuracy(mean/std)(%)) comparison of PerMFL with state-of-the-art.

Architecture	MCLR (Strongly convex)				
	Algorithm	MNIST	Synthetic	FMNIST	EMNIST-10
Conventional	FedAvg(GM)[McM+17]	84.87 (± 0.054)	84.87(± 0.054)	79.80 (±0.002)	91.60(±0.001)
	Per-FedAvg (PM)[FMO20]	94.81(± 0.00)	83.91(±0.15)	94.75 (±0.00)	97.57(±0.0)
	pFedMe(GM)[TTN20]	75.50(±0.00)	81.93(±0.21)	83.45(±0.21)	88.78(±0.01)
	pFedMe(PM)[TTN20]	88.89(±0.001)	87.61(±0.32)	91.32 (±0.08)	91.23(±0.01)
	pFedBayes(PM)[Zha+22]	94.13(±0.27)	87.05(±0.5)	92.14(±0.001)	94.13(±0.001)
	Ditto (GM) [Li+21]	84.81(±0.001)	82.35(±0.001)	74.02(±0.001)	91.03 (±0.0003)
Multi-tier	h-SGD (GM) [Wan+22]	**87.41**(±6.35)	84.29(±5.18)	81.653(±1.8)	**92.33**(±0.001)
	AL2GD(PM) [LHK22]	93.70 (±0.13)	84.75(±0.03)	**98.52(±0.004)**	**98.72(**±0.001**)**
	DemLearn (GM)[Ngu+22]	87.32(±0.002)	67.93(±0.04)	62.60(±0.002)	69.09(±0.12)
	DemLearn (PM)[Ngu+22]	91.26 (±0.01)	81.21(±0.01)	97.50(±0.0)	97.24(±0.005)
	PerMFL(GM) [ours]	86.92(±0.013)	**84.92**(±0.06)	**83.71**(±0.001)	91.68(±0.0)
	PerMFL(PM) [ours]	**96.87(**±0.0**)**	**87.94(**±0.001**)**	96.77 (±0.0)	96.49(±0.0)
DNN or CNN (Non-convex)					
Conventional	FedAvg(GM)	93.17 (±0.02)	84.53(±0.067)	**84.14**(±0.00)	92.73(±0.003)
	Per-FedAvg(PM)	91.845(±0.00)	75.93 (±0.18)	88.69(±0.269)	97.37(±0.01)
	pFedMe(GM)	80.12(±0.01)	81.23(±0.19)	68.64 (±0.009)	91.81 (±0.0002)
	pFedMe(PM)	97.40(±0.001)	87.86(±0.06)	96.30 (±0.001)	97.18(±0.0003)
	Ditto(GM)	87.30(±0.03)	81.12(±0.006)	57.80(±0.001)	90.58(±0.004)
Multi-tier	h-SGD(GM)	86.59 (±7.14)	87.42 (±5.67)	79.84 (±0.035)	96.03 (±0.001)
	AL2GD(PM)	91.04 (±0.035)	84.92 (±0.02)	71.32(±0.13)	92.94 (±0.14)
	DemLearn(GM)	**90.75(**±0.001**)**	68.91(±0.05)	64.84 (±0.002)	**96.63 (**±0.005**)**
	DemLearn(PM)	97.20(±0.001)	82.74(±0.008)	98.64(±0.0)	98.74(±0.0)
	PerMFL(GM) [ours]	89.39 (±0.001)	**87.53 (**±0.0**)**	79.15(±0.0)	93.12(±0.0)
	PerMFL(PM) [ours]	**98.15 (**±0.0**)**	**87.89(**±0.0**)**	**98.67**(±0.0)	**98.79**(±0.0)

4.1.2 Convergence

From Figure 2, we observed that the convergence of PerMFL(PM) is equivalent to DemLearn and is faster than h-SGD and AL2GD. Similar findings were also observed in EMNIST-10, Synthetic, and MNIST datasets given in the

supplementary copy. It is because, inside each team multiple iterations are happening, that helps the personalized model to converge quickly.

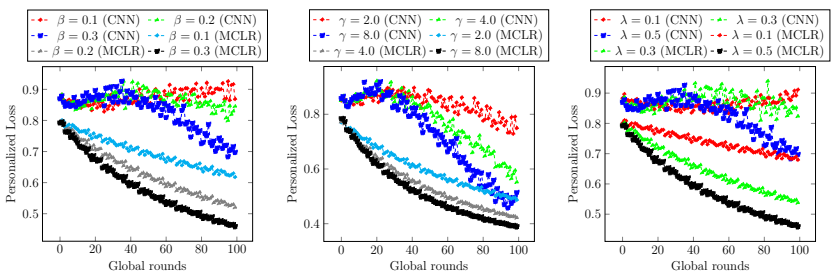


(a) non-convex (b) non-convex (c) strongly convex (d) strongly convex

Figure 2: Convergence of PerMFL with multi-tier SOTA in strongly convex and non-convex settings on FMNIST

4.1.3 Effect of hyghperparameters β, γ, and λ

From Figure 3, we observed if we increase the value of β, γ, and λ separately then PerMFL(PM) converge faster. A similar observation is found for FMNIST, and the synthetic dataset is given in the supplementary copy. In all experiments, the hyperparameters followed the bounds given in Theorem 1 for strongly convex and Theorem 2 for non-convex and smooth problems.



(a) Effect of β (b) Effect of γ (c) Effect of λ

Figure 3: Effect of β, γ, and λ on convergence of PerMFL(PM) in non-convex(CNN) and strongly convex(MCLR) settings using MNIST dataset

4.1.4 Ablation studies on team formation

In Table 2, we evaluated PerMFL for both worst-case (team 1 with labels $\{0, 1, 2, 3, 4\}$ and team 2 with $\{5, 6, 7, 8, 9\}$) and average-case (teams with overlapping labels, team 1 with labels $\{0, 1, 2, 3, 4, 5, 6\}$ and team 2 with $\{5, 6, 7, 8, 9, 0, 1\}$) over 400 global iterations which include 10 and 20, team and local iterations respectively, with the hyperparameter settings $\lambda = 0.5$, $\gamma = 1.5$, $\beta = 0.6$, and, $\alpha = 0.01$. PerMFL(GM) showed a 4% improvement in

average-case over the worst-case with FMNIST data. Likewise, PerMFL(PM) had slightly better results in average-case than worst-case for non-convex setups (CNN) with MNIST and FMNIST datasets, indicating PerMFL(PM)'s performance is mostly unaffected by team formation.

Table 2: Performance of PerMFL (Validation accuracy (%)) on worst-case and average-case team formation

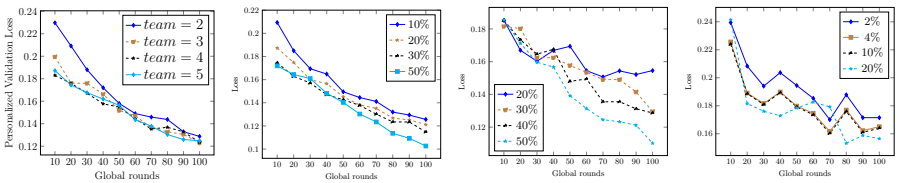
Team Formation	Algorithm	MNIST		FMNIST		EMNIST-10	
		MCLR(%)	CNN(%)	MCLR(%)	CNN(%)	MCLR(%)	CNN(%)
Worst case	PerMFL(PM)	96.86	95.80	**97.14**	95.62	**96.57**	98.13
	PerMFL(GM)	80.48	82.21	**76.18**	70.28	88.05	87.05
Average case	PerMFL(PM)	**97.01**	**97.02**	96.72	**97.38**	96.39	**98.15**
	PerMFL(GM)	**80.86**	**83.59**	74.45	**74.66**	**90.36**	**87.43**

4.1.5 Ablation study on teams and clients participation

PerMFL achieves quick convergence with complete participation from both teams and devices (see Figure 4a) or when teams fully participate but devices do so partially (see Figure 4b), in contrast to slower convergence under partial participation from both teams and devices (see Figure 4d). Moreover, expanding the number of teams does not impact the convergence speed of PerMFL(PM) when there is full participation from all teams and devices throughout all global rounds (see Figure 4a). Increased device involvement leads to faster convergence (see Figure 4b), whereas lower team engagement (see Figure 4c) decelerates it. Nonetheless, when team participation reaches 50% in each global round, the convergence rate is comparable to that observed in scenarios with complete participation (see Figure 4a). When all teams are fully participating, increasing the number of team iterations leads to quicker convergence. However, in scenarios where both team and device participation is minimal (2%) as shown in Figure 4d, PerMFL(PM) requires more global and team iterations to achieve convergence. Extended experimental results are reported in the supplementary copy.

5 Conclusions

We introduced, PerMFL, a personalized multi-tier federated learning approach involving global servers, teams, and devices. PerMFL utilized squared euclidean distance regularization on both devices and teams. By employing PerMFL, we are able to generate personalized models for each device and simultaneously obtain a global model. The performance of PerMFL demonstrates linear baseline rates for strongly convex scenarios and sublinear baseline rates for non-convex and smooth scenarios. Empirically we observed that PerMFL(PM) converges quickly and outperformed the state-of-the-art. PerMFL performs best when both teams and devices participate fully. Very low team and device participation



(a) (b) (c) (d)

Figure 4: Ablation study on team and client's participation on MNIST datasets in convex settings (MCLR): Figure 4a Full teams and devices participation, Figure 4b Full participation of 5 teams but partial participation of devices, Figure 4c partial participation of teams but full participation of devices, and Figure 4d Partial participation of teams (2%) with partial participation of clients

degrade the performance of PerMFL(GM). It requires more global and team iterations to converge. Also, in worst-case team formation, the performance of the global model decreases, while the personalized model is able to maintain its performance. Moreover right combination of λ, β, and γ enhances the convergence of PerMFL.

6 Acknowledgement

This work was supported by the Wallenberg AI, Autonomous Systems and Software Program (WASP) funded by the Knut and Alice Wallenberg Foundation. The computations were enabled by the Berzelius resource provided by the Knut and Alice Wallenberg Foundation at the National Supercomputer Centre. Alp Yurtsever acknowledges support from the Swedish Research Council, registration number 2023-05476.

References

[Aba+20] Mehdi Salehi Heydar Abad et al. "Hierarchical federated learning across heterogeneous cellular networks". In: *Proceedings of International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE. 2020, pp. 8866–8870.

[Abd+22] 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.

[Bit+18] Luiz Bittencourt et al. "The internet of things, fog and cloud continuum: Integration and challenges". In: *Internet of Things* 3 (2018), pp. 134–155.

[Cal+18] Sebastian Caldas et al. “Leaf: A benchmark for federated settings”. In: *arXiv preprint arXiv:1812.01097* (2018).

[Che+22] Daoyuan Chen et al. “pFL-Bench: A Comprehensive Benchmark for Personalized Federated Learning”. In: *arXiv preprint arXiv:2206.03655* (2022).

[Coh+17] Gregory Cohen et al. “EMNIST: Extending MNIST to handwritten letters”. In: *Proceedings of International Joint Conference on Neural Networks (IJCNN)*. IEEE. 2017, pp. 2921–2926.

[Das+22] Anirban Das et al. “Cross-silo federated learning for multi-tier networks with vertical and horizontal data partitioning”. In: *ACM Transactions on Intelligent Systems and Technology (TIST)* 13.6 (2022), pp. 1–27.

[Ekm+22] Morgan Ekmefjord et al. “Scalable federated machine learning with fedn”. In: *Proceedings of 22nd International Symposium on Cluster, Cloud and Internet Computing (CCGrid)*. IEEE. 2022, pp. 555–564.

[FAL17] Chelsea Finn, Pieter Abbeel, and Sergey Levine. “Model-agnostic meta-learning for fast adaptation of deep networks”. In: *International conference on machine learning, PMLR* (2017), pp. 1126–1135.

[FMO20] Alireza Fallah, Aryan Mokhtari, and Asuman Ozdaglar. “Personalized federated learning: A meta-learning approach”. In: *arXiv preprint arXiv:2002.07948* (2020).

[Gas+22] Elnur Gasanov et al. “FLIX: A Simple and Communication-Efficient Alternative to Local Methods in Federated Learning”. In: *International Conference on Artificial Intelligence and Statistics, PMLR* (2022), pp. 11374–11421.

[Kai+21] Peter Kairouz et al. “Advances and open problems in federated learning”. In: *Foundations and Trends® in Machine Learning* 14.1–2 (2021), pp. 1–210.

[Kar+20] Sai Praneeth Karimireddy et al. “Scaffold: Stochastic controlled averaging for federated learning”. In: *International Conference on Machine Learning, PMLR* (2020), pp. 5132–5143.

[KH09] Alex Krizhevsky and Geoffrey Hinton. *Learning multiple layers of features from tiny images*. Technical report 4. Toronto, Canada: University of Toronto, 2009.

[LeC+98] Yann LeCun et al. “Gradient-based learning applied to document recognition”. In: *Proceedings of the IEEE* 86.11 (1998), pp. 2278–2324.

[LHK22] Boxiang Lyu, Filip Hanzely, and Mladen Kolar. “Personalized Federated Learning with Multiple Known Clusters”. In: *arXiv preprint arXiv:2204.13619* (2022).

[Li+20] Tian Li et al. “Federated optimization in heterogeneous networks”. In: *Proceedings of Machine learning and systems* 2 (2020), pp. 429–450.

[Li+21] Tian Li et al. “Ditto: Fair and robust federated learning through personalization”. In: *International Conference on Machine Learning*. PMLR. 2021, pp. 6357–6368.

[Lin+20] Tao Lin et al. “Don’t use large mini-batches, use local sgd”. In: *Proceedings of International Conference on Learning Representations* (2020).

[Liu+20] Lumin Liu et al. “Client-edge-cloud hierarchical federated learning”. In: *Proceedings of International Conference on Communications (ICC)*. IEEE. 2020, pp. 1–6.

[Liu+22] Lumin Liu et al. “Hierarchical federated learning with quantization: Convergence analysis and system design”. In: *IEEE Transactions on Wireless Communications* (2022).

[Lon+23] Guodong Long et al. “Multi-center federated learning: clients clustering for better personalization”. In: *World Wide Web* 26.1 (2023), pp. 481–500.

[Man+20] Yishay Mansour et al. “Three approaches for personalization with applications to federated learning”. In: *arXiv:2002.10619* (2020).

[McM+17] Brendan McMahan et al. “Communication-efficient learning of deep networks from decentralized data”. In: *Artificial intelligence and statistics, PMLR* (2017), pp. 1273–1282.

[Ngu+22] Minh NH Nguyen et al. “Self-organizing democratized learning: Toward large-scale distributed learning systems”. In: *IEEE Transactions on Neural Networks and Learning Systems* (2022).

[Pil+22] Krishna Pillutla et al. “Federated learning with partial model personalization”. In: *International Conference on Machine Learning, PMLR* (2022), pp. 17716–17758.

[Tan+22] Alysa Ziying Tan et al. “Towards personalized federated learning”. In: *IEEE Transactions on Neural Networks and Learning Systems* (2022).

[TTN20] Canh T Dinh, Nguyen Tran, and Josh Nguyen. “Personalized federated learning with moreau envelopes”. In: *Advances in Neural Information Processing Systems* 33 (2020), pp. 21394–21405.

[Tzi+22] Isidoros Tziotis et al. “Straggler-Resilient Personalized Federated Learning”. In: *arXiv preprint arXiv:2206.02078* (2022).

[Wai+20] Aidmar Wainakh et al. “Enhancing privacy via hierarchical federated learning”. In: *2020 IEEE European Symposium on Security and Privacy Workshops (EuroS&PW)*. IEEE. 2020, pp. 344–347.

[Wan+22] Jiayi Wang et al. “Demystifying why local aggregation helps: Convergence analysis of hierarchical SGD”. In: *Proceedings of the AAAI Conference on Artificial Intelligence*. 2022, pp. 8548–8556.

[Wu+21] Jinze Wu et al. “Hierarchical personalized federated learning for user modeling”. In: *Proceedings of the Web Conference 2021*. 2021, pp. 957–968.

[XRV17] Han Xiao, Kashif Rasul, and Roland Vollgraf. “Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms”. In: *arXiv preprint arXiv:1708.07747* (2017).

[YTW23] Yihan Yan, Xiaojun Tong, and Shen Wang. “Clustered Federated Learning in Heterogeneous Environment”. In: *IEEE Transactions on Neural Networks and Learning Systems* (2023), pp. 1–14. DOI: 10.1109/TNNLS.2023.3264740.

[Zha+22] Xu Zhang et al. “Personalized Federated Learning via Variational Bayesian Inference”. In: *International Conference on Machine Learning, (PMLR)* (2022), pp. 26293–26310.

[Zha+24] Rongyu Zhang et al. “Multi-level Personalized Federated Learning on Heterogeneous and Long-Tailed Data”. In: *IEEE Transactions on Mobile Computing* (2024).

[Zha24] Ran Zhang et al. “CPPer-FL: Clustered Parallel Training for Efficient Personalized Federated Learning”. In: *IEEE Transactions on Mobile Computing* (2024).

[Zho+23] Hongliang Zhou et al. “Toward Robust Hierarchical Federated Learning in Internet of Vehicles”. In: *IEEE Transactions on Intelligent Transportation Systems* 24.5 (2023), pp. 5600–5614. DOI: 10.1109/TITS.2023.3243003.

Appendix A Additional Results and Analysis

We studied classification problems to validate PerMFL using both image (MNIST [LeC+98], FMNIST [XRV17] EMNIST [Coh+17] with 10 classes and 62 classes, CIFAR100 [KH09]) and non-image synthetic datasets, but PerMFL is not limited to these two categories of data. Data distributions are heterogeneous and non-iid. The distribution of the EMNIST dataset with 62 classes exhibits non-iid characteristics similar to the FEMNIST dataset. Consequently, we will distinguish EMNIST with 10 classes as 'EMNIST' and EMNIST with 62 classes as 'FEMNIST'. In the case of MNIST, FMNIST, EMNIST, and synthetic datasets, each device contains data for 2 classes. However, for FEMNIST and CIFAR100 datasets, each device carries data for 3 classes. There are no overlapping samples among devices. This supplementary material provides the following experiments and analysis to validate the PerMFL.

1. A detailed empirical study to investigate the impact of hyperparameters β, λ, and γ on the convergence of PerMFL (see Appendix A.4).
2. An ablation study to analyze the influence of team and device participation (see Appendix A.5).
3. An ablation study to explore team formation, i.e., the performance of PerMFL on worst-case and average-case team formation (see 4.1.4).
4. An ablation study that explores the effects of team iterations on the convergence of PerMFL (see Appendix A.6).
5. Convergence analysis on MNIST and synthetic datasets with the multi-tier SOTA such as (AL2GD[LHK22], h-SGD [Liu+22], [Ngu+22]) (see Appendix A.7).
6. Performance analysis of PerMFL on FEMNIST and CIFAR100 datasets (see Appendix A.8).

Each experiments ran 10 times, from there we produced mean and standard deviation.

Appendix A.1 Hardware specification

The experiments were conducted using an NVIDIA DGX-A100 GPU with 40 GB of RAM. The DGX-A100 is based on the NVIDIA A100 Tensor Core GPU architecture, and each A100 GPU in the system has 40 GB of high-bandwidth memory (HBM2). The standard configuration of the DGX-A100 includes eight A100 GPUs connected via NVLink, providing a total of 320 GB of GPU memory (40 GB per GPU $\times$ 8 GPUs). However, we could utilize only one GPU at a time for our experiments, which allowed us to utilize 40 GB of GPU memory per experiment.

Appendix A.2 Dataset description

We conducted our experiments using MNIST, FMNIST, EMNIST, FEMNIST, CIFAR100 and Synthetic datasets. FEMNIST, CIFAR100, and synthetic datasets are considered to create larger and more complicated scenarios from these datasets. The description of the datasets is given below.

Appendix A.2.1 MNIST

The primary purpose of the MNIST dataset is to serve as a widely used benchmark for evaluating and comparing the performance of various models in image classification tasks. It consists of a total of 70,000 examples, with 60,000 examples used for training and 10,000 examples used for testing. Each example is a grayscale image measuring 28x28 pixels, representing a handwritten digit ranging from 0 to 9. Every image in the dataset is associated with a label that denotes the correct digit it represents. The labels themselves are integers ranging from 0 to 9, corresponding to the handwritten digits in the images.

Appendix A.2.2 FMNIST

The FMNIST dataset serves the purpose of evaluating and benchmarking machine learning algorithms, particularly in the areas of image classification and pattern recognition. It differs from the original MNIST dataset by focusing on fashion-related images rather than handwritten digits, providing a more complex task. The dataset consists of 70,000 grayscale images with dimensions of 28x28 pixels. These images are split into 60,000 training examples and 10,000 testing examples. They depict various fashion items, including clothing, shoes, bags, and accessories. Each image in the FMNIST dataset is associated with a label representing the corresponding fashion item category. There are a total of 10 classes representing different types of clothing and fashion accessories such as T-shirt/top, Trouser, Pullover, Dress, Coat, Sandal, Shirt, Sneaker, Bag, and Ankle boot. Compared to the MNIST dataset, FMNIST presents a more significant challenge due to the diversity of clothing types and the increased complexity of the images. These datasets are often utilized to assess the robustness and generalization capabilities of machine learning models.

Appendix A.2.3 EMNIST

The dataset is a collection of handwritten characters, including lowercase and uppercase letters and digits. It consists of six different splits or variations, each representing a different task or scenario. The first split, called EMNIST ByClass, contains a total of 814,255 images representing 62 character classes. These classes include 26 uppercase letters, 26 lowercase letters, and 10 digits. The second split, EMNIST ByMerge, merges similar characters into a single class, resulting in 47 classes. This split is useful and challenging for scenarios where distinguishing between similar characters, such as uppercase and lowercase letters. The third

split, EMNIST Balanced, aims to balance the number of samples per class. It provides a balanced dataset with 131,600 images representing 47 classes. The fourth split, EMNIST Letters, focuses exclusively on uppercase and lowercase letters. It consists of 145,600 images representing 26 classes of letters. The fifth split, EMNIST Digits, contains only the digits from 0 to 9. It consists of 280,000 images representing the 10-digit classes. Lastly, the sixth split follows the structure of the original MNIST dataset that contains 70,000 images of digits from 0 to 9. Each image in the dataset is associated with a label indicating the corresponding character class that provided information about the specific representative character or digit. Here for all the experiments, we considered split by digits.

Appendix A.2.4 FEMNIST

It is a Federated EMNIST dataset. It is a ByClass split over the EMNIST dataset containing 814,255 images representing 62 character classes (0-9, A-Z, and a-z). The data are distributed among 3500 devices in an unbalanced manner, where each device has access to a maximum of 3 classes. FEMNIST is similar to [Cal+18].

Appendix A.2.5 CIFAR100

This dataset is a collection of 60,000 32x32 colour images in 100 classes, with 600 images per class. The dataset is distributed among 350 clients in an unbalanced manner where each client can access 3 classes.

Appendix A.2.6 Synthetic

We generate a synthetic dataset with $\bar{\alpha} = 0.5$ and $\bar{\beta} = 0.5$. The synthetic dataset is a tabular dataset. It has 60 features and 10 classes. The sample size of each client ranges from 250 to 25810, and each client has almost 2 classes. Finally, we distribute the data to N devices according to the power law in [Li+20].

Appendix A.2.7 Data division

MNIST, FMNIST, EMNIST, and synthetic datasets have ten different class labels representing distinct data distributions. The data was divided among multiple devices in a non-iid manner that ensured each device had information from two classes. To ensure each device has two classes, first, we gave specific data from each of these two classes to that device and then randomly distributed the remaining samples from these two classes. A similar approach is taken for the other devices. Following that, the devices were further divided into teams randomly. In the experiments, teams have an equal number of devices.

Appendix A.3 Learning models

Our study examined different scenarios and used specific models to handle them. We employed a multi-nomial logistic regression (MLR) model with l_2 regularization and a softmax activation function for strongly convex scenarios. To handle non-convex scenarios, we adopted different approaches depending on the dataset. For synthetic datasets, we constructed deep neural networks with two hidden layers. On the other hand, for the MNIST, FMNIST, and EMNIST datasets, which also involve non-convex scenarios, we built three two-layered convolutional neural networks (CNNs).

Throughout our experiments, we used the abbreviations (PM) and (GM) to refer to the personalized model and global model, respectively.

Appendix A.4 Effect of hyperparameters

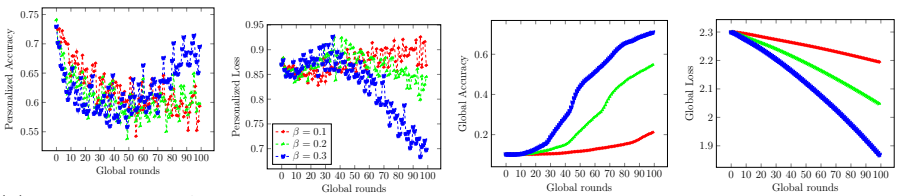
A series of tests were conducted on the MNIST, FMNIST, and Synthetic datasets to examine the impact of various hyperparameters, including λ, γ, and β, on the convergence of PerMFL. These tests were performed for both smooth strongly convex and smooth non-convex scenarios. The entire experiment is performed with the full participation of teams and devices for each global rounds. The number of teams is four, and each team has ten devices.

Effect of β: In this study, we made the following observation: when we increased the value of β (see Figure 5 to Figure 10) while other hyperparameters λ, and γ remains constant, both personalized and global model of PerMFL exhibited faster convergence. This behaviour was consistent in both convex and non-convex settings. Very low values of β do not generalize the global model well and delay the convergence of personalized and global models. A high value of β better generalizes the global model and the convergence faster. In all experiments from Figure 5 to Figure 10, we set the value of $\gamma = 3.0$ and $\lambda = 0.5$, $\eta = 0.03$, and $\alpha = 0.01$.

Effect of γ: In this study, we examined the influence of the hyperparameter γ on the convergence of PerMFL. From Figure 11 to Figure 16, we observed that increasing the value of γ led to faster convergence of PerMFL PM and GM. Like β, a low value of γ does not generalize the model well and slows the convergence speed of personalized and global models. Increasing the value of γ results in a better generalization with a faster convergence. In all experiments from Figure 11 to Figure 16, we set the value of $\lambda = 1.5$ and $\beta = 0.1$, $\eta = 0.03$, and $\alpha = 0.01$.

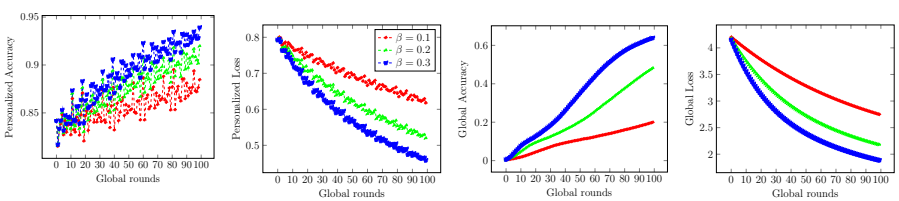
Effect of λ: Here, we examined the influence of the hyperparameter λ on the convergence of PerMFL. From the experimental results (see Figure 17 to Figure 21), we found that a low value of λ tended to impede the convergence process of both the personalized and global models. However, by increasing the value of λ, we observed a significant improvement in convergence. All the experiments from Figure 17 to Figure 21, we set the value of $\beta = 0.3$ and $\gamma = 3.0$, $\eta = 0.03$, and $\alpha = 0.01$.

Discussions : The hyperparameters β, γ, and λ exhibit interdependencies that should be taken into consideration. Modifying the value of a single hyperparameter can significantly influence the learning of both the personalized and global models. Consequently, it is crucial to recognize the intricate relationship among these hyperparameters, as changes in one hyperparameter can have consequential effects on the overall learning process. Therefore, thoroughly evaluating these hyperparameters is essential to achieve optimal model performance and convergence.



(a) Personalized Accuracy (Validation) (b) Personalized Loss (Validation) (c) Global Accuracy (Validation) (d) Global Loss (Validation)

Figure 5: Effect of β on convergence of PerMFL in non-convex settings (CNN) using MNIST dataset

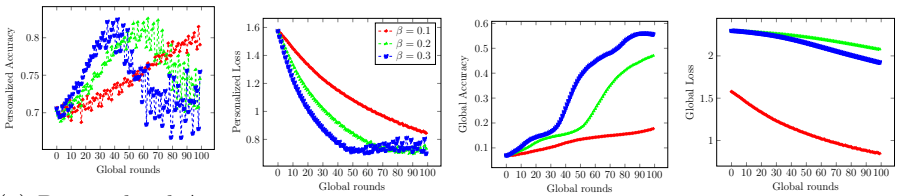


(a) Personalized Accuracy (Validation) (b) Personalized Loss (Validation) (c) Global Accuracy (Validation) (d) Global Loss (Validation)

Figure 6: Effect of β on convergence of PerMFL in strongly convex settings (MCLR) using MNIST dataset

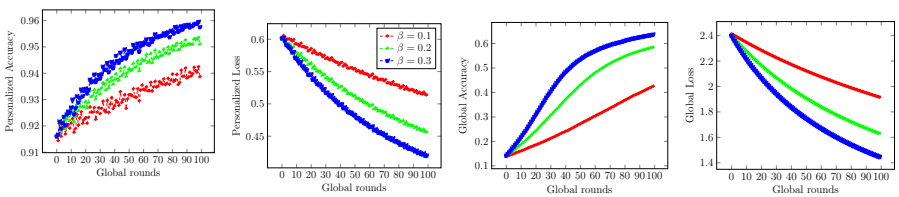
Appendix A.5 Ablation study on the effect of teams and devices participation on PerMFL

In a multi-tier architecture, teams and devices can be constituted in four ways: (1) Both teams and devices within teams have full participation (see Figure 23 to Figure 30), (2) Teams have full participation, but devices within teams are partially participating (see Figure 31 to Figure 42), (3) Teams have partial participation, but all devices within teams are participating (see Figure 43 to Figure 48), and (4) Teams and devices both have partial participation (see



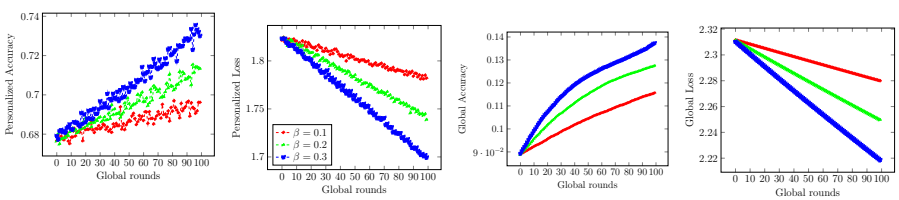
(a) Personalized Accuracy (Validation) (b) Personalized Loss (Validation) (c) Global Accuracy (Validation) (d) Global Loss (Validation)

Figure 7: Effect of β on convergence of PerMFL in non-convex settings (CNN) using FMNIST dataset



(a) Personalized Accuracy (Validation) (b) Personalized Loss (Validation) (c) Global Accuracy (Validation) (d) Global Loss (Validation)

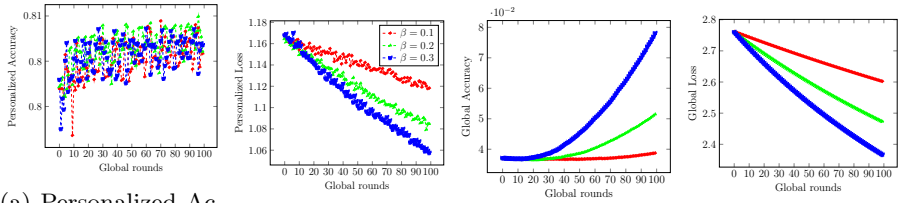
Figure 8: Effect of β on convergence of PerMFL in strongly convex settings (MCLR) using FMNIST dataset



(a) Personalized Accuracy (Validation) (b) Personalized Loss (Validation) (c) Global Accuracy (Validation) (d) Global Loss (Validation)

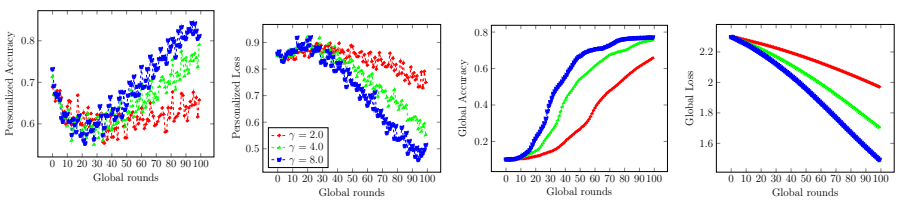
Figure 9: Effect of β on convergence of PerMFL in non-convex settings (DNN) using synthetic dataset

Figure 49 to Figure 51). In these experiments, we observed how well PerMFL performs and converges in various team combinations. We are also studying how PerMFL behaves when the number of participating teams and devices is limited.



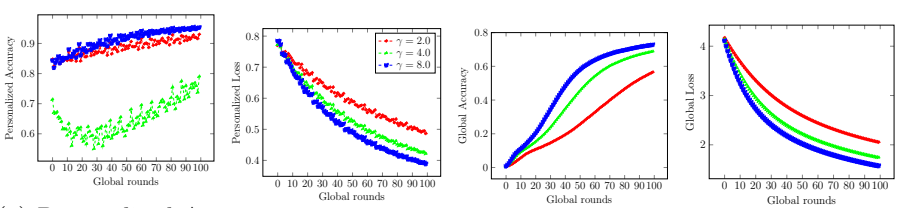
(a) Personalized Accuracy (Validation) (b) Personalized Loss (Validation) (c) Global Accuracy (Validation) (d) Global Loss (Validation)

Figure 10: Effect of β on convergence of PerMFL in strongly convex settings (MCLR) using synthetic dataset



(a) Personalized Accuracy (Validation) (b) Personalized Loss (Validation) (c) Global Accuracy (Validation) (d) Global Loss (Validation)

Figure 11: Effect of γ on convergence of PerMFL in non-convex settings (CNN) using MNIST dataset

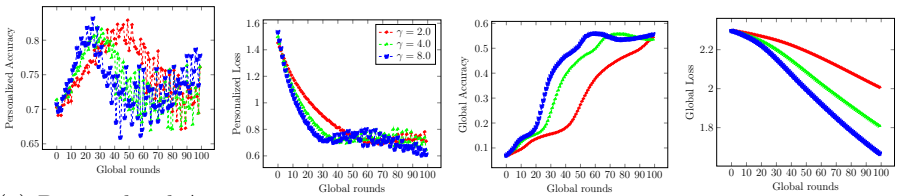


(a) Personalized Accuracy (Validation) (b) Personalized Loss (Validation) (c) Global Accuracy (Validation) (d) Global Loss (Validation)

Figure 12: Effect of the hyperparameter γ on convergence of PerMFL in strongly convex settings (MCLR) using MNIST dataset

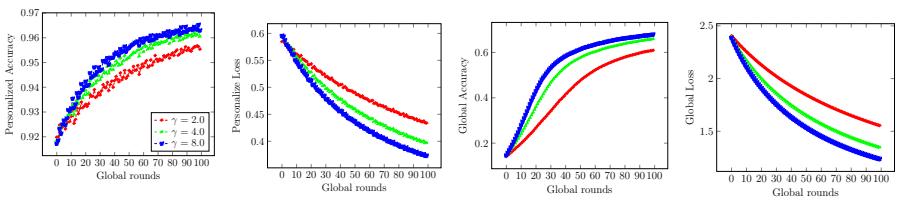
Appendix A.5.1 Full participation of teams and devices

We conducted experiments in both convex (see Figure 24, Figure 26, Figure 28, Figure 30) and non-convex (See Figure 23, Figure 25, Figure 27, Figure 29) settings using MNIST, FMNIST, EMNIST, and Synthetic datasets. Our findings indicate that when it comes to team and device selection strategies, having full participation of teams and devices yields the best results compared to the other



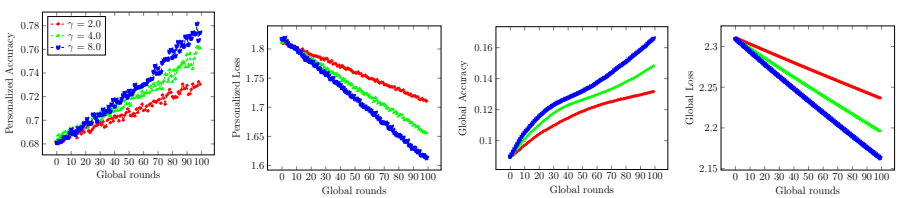
(a) Personalized Accuracy (Validation) (b) Personalized Loss (Validation) (c) Global Accuracy (Validation) (d) Global Loss (Validation)

Figure 13: Effect of γ on convergence of PerMFL in non-convex settings (CNN) using FMNIST dataset



(a) Personalized Accuracy (Validation) (b) Personalized Loss (Validation) (c) Global Accuracy (Validation) (d) Global Loss (Validation)

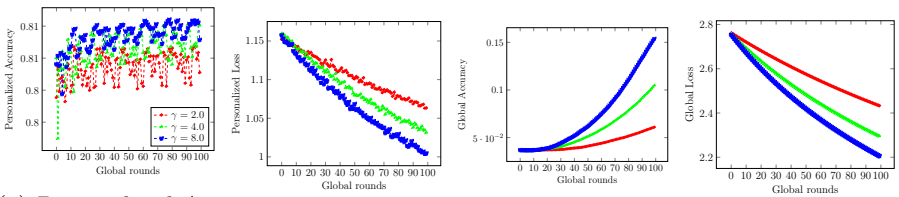
Figure 14: Effect of γ on convergence of PerMFL in strongly convex settings (MCLR) using FMNIST dataset



(a) Personalized Accuracy (Validation) (b) Personalized Loss (Validation) (c) Global Accuracy (Validation) (d) Global Loss (Validation)

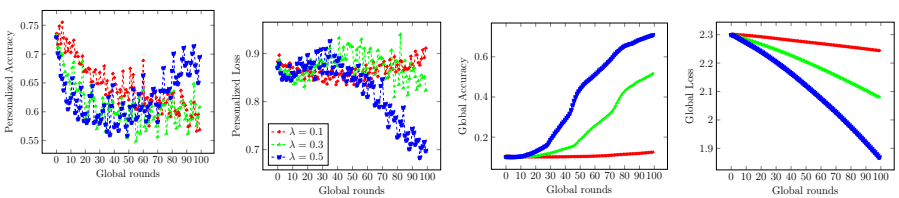
Figure 15: Effect of γ on convergence of PerMFL in non-convex settings (DNN) using synthetic dataset

three types of participation strategies. Increasing the number of teams does not negatively impact the performance of the personalized model. However, in some cases, such as those depicted in Figure 29a and Figure 29b, increasing the number of teams can lead to a decrease in the convergence of the global model.



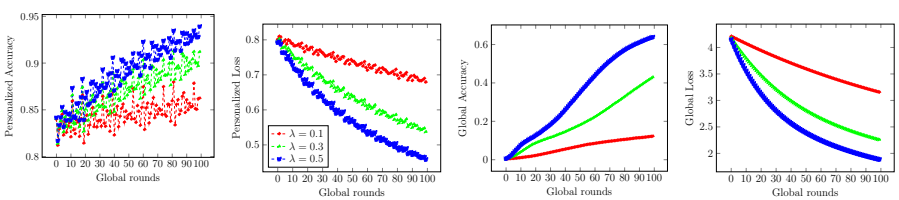
(a) Personalized Accuracy (Validation) (b) Personalized Loss (Validation) (c) Global Accuracy (Validation) (d) Global Loss (Validation)

Figure 16: Effect of γ on convergence of PerMFL in strongly convex settings (MCLR) using synthetic dataset



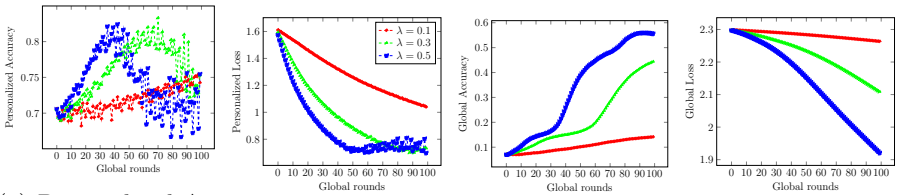
(a) Personalized Accuracy (Validation) (b) Personalized Loss (Validation) (c) Global Accuracy (Validation) (d) Global Loss (Validation)

Figure 17: Effect of λ on convergence of PerMFL in non-convex settings (CNN) using MNIST dataset



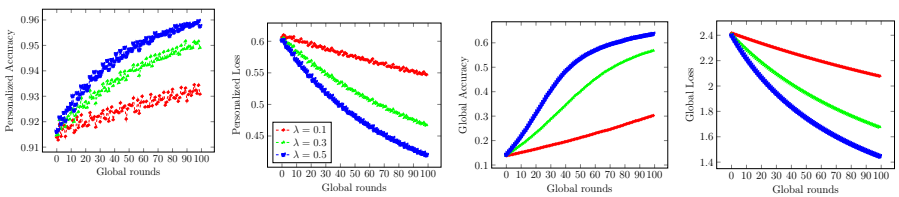
(a) Personalized Accuracy (Validation) (b) Personalized Loss (Validation) (c) Global Accuracy (Validation) (d) Global Loss (Validation)

Figure 18: Effect of λ on convergence of PerMFL in strongly convex settings (MCLR) using MNIST dataset



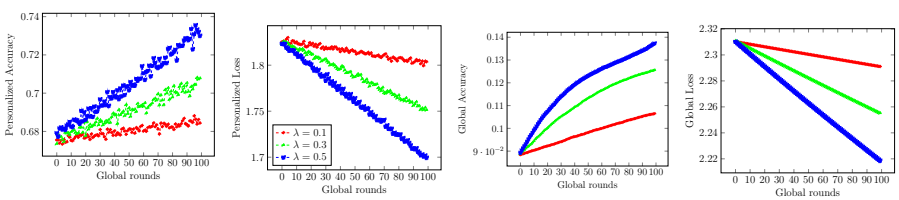
(a) Personalized Accuracy (Validation) (b) Personalized Loss (Validation) (c) Global Accuracy (Validation) (d) Global Loss (Validation)

Figure 19: Effect of λ on convergence of PerMFL in non-convex settings (CNN) using FMNIST dataset



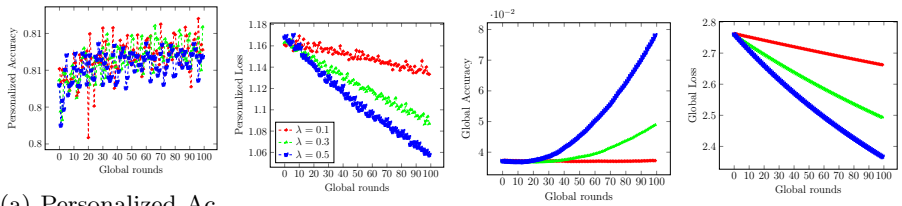
(a) Personalized Accuracy (Validation) (b) Personalized Loss (Validation) (c) Global Accuracy (Validation) (d) Global Loss (Validation)

Figure 20: Effect of λ on convergence of PerMFL in strongly convex settings (MCLR) using FMNIST dataset



(a) Personalized Accuracy (Validation) (b) Personalized Loss (Validation) (c) Global Accuracy (Validation) (d) Global Loss (Validation)

Figure 21: Effect of λ on convergence of PerMFL in non-convex settings (DNN) using synthetic dataset



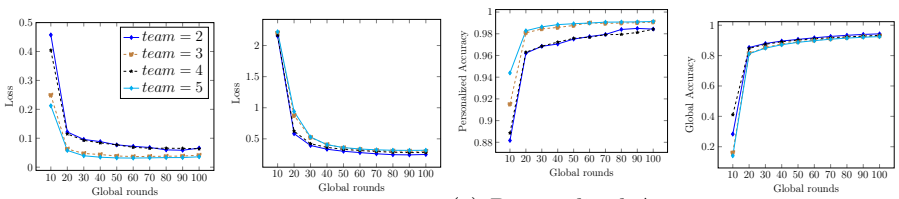
(a) Personalized Accuracy (Validation)

(b) Personalized Loss (Validation)

(c) Global Accuracy (Validation)

(d) Global Loss (Validation)

Figure 22: Effect of λ on convergence of PerMFL in strongly convex settings (MCLR) using synthetic dataset



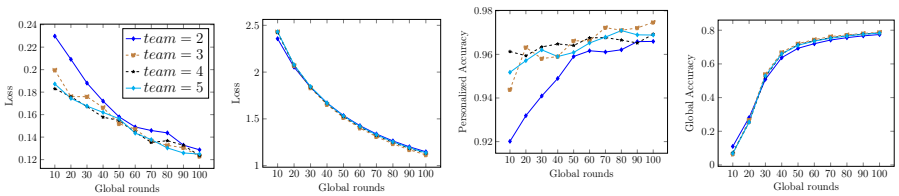
(a) Personalized Loss (Validation)

(b) Global Loss (Validation)

(c) Personalized Accuracy (Validation)

(d) Global Accuracy (Validation)

Figure 23: Full participation teams and devices on MNIST datasets in non-convex settings (CNN)



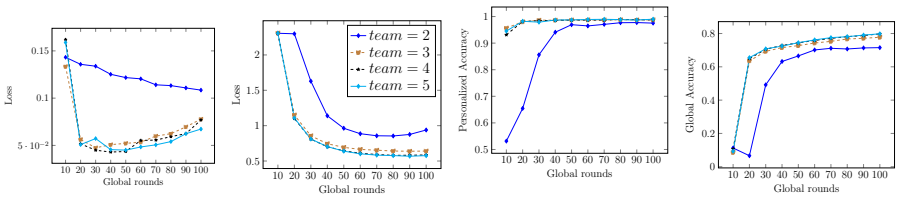
(a) Personalized Loss (Validation)

(b) Global Loss (Validation)

(c) Personalized Accuracy (Validation)

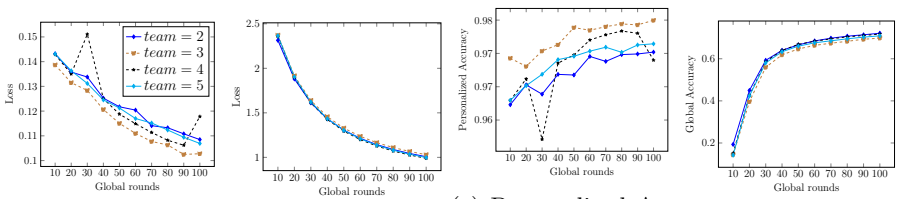
(d) Global Accuracy (Validation)

Figure 24: Full participation teams and devices on MNIST datasets in convex settings (MCLR)



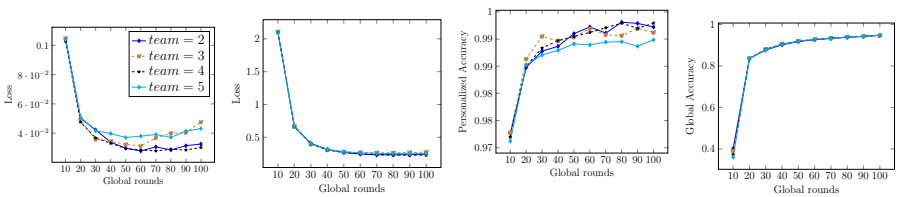
(a) Personalized Loss (Validation) (b) Global Loss (Validation) (c) Personalized Accuracy (Validation) (d) Global Accuracy (Validation)

Figure 25: Full participation teams and devices on FMNIST datasets in non-convex settings (CNN)



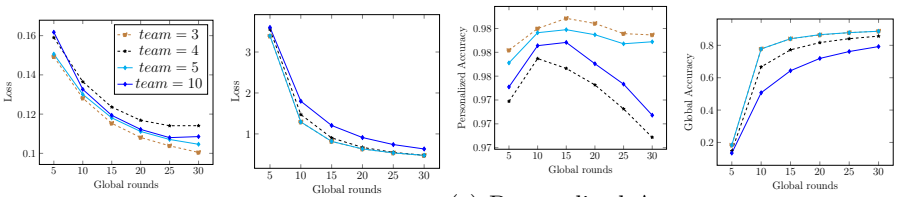
(a) Personalized Loss (Validation) (b) Global Loss (Validation) (c) Personalized Accuracy (Validation) (d) Global Accuracy (Validation)

Figure 26: Full participation teams and devices on FMNIST datasets in convex settings (MCLR)



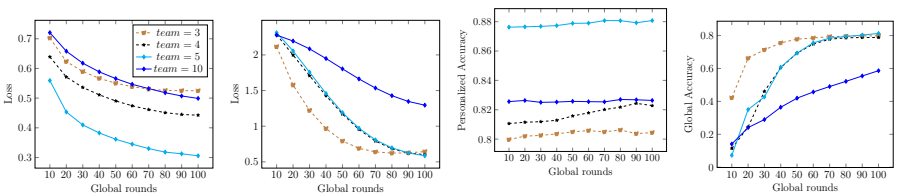
(a) Personalized Loss (Validation) (b) Global Loss (Validation) (c) Personalized Accuracy (Validation) (d) Global Accuracy (Validation)

Figure 27: Full participation teams and devices on EMNIST datasets in non-convex settings (CNN)



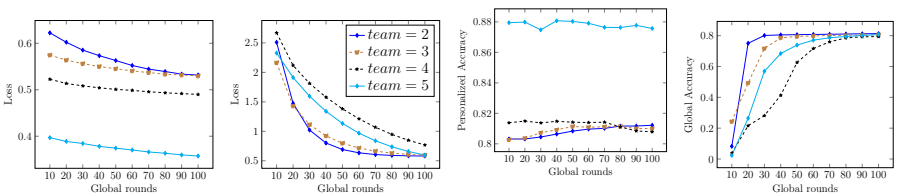
(a) Personalized Loss (Validation) (b) Global Loss (Validation) (c) Personalized Accuracy (Validation) (d) Global Accuracy (Validation)

Figure 28: Full participation teams and devices on EMNIST datasets in convex settings (MCLR)



(a) Personalized Loss (Validation) (b) Global Loss (Validation) (c) Personalized Accuracy (Validation) (d) Global Accuracy (Validation)

Figure 29: Full participation teams and devices on synthetic datasets in non-convex settings (DNN)

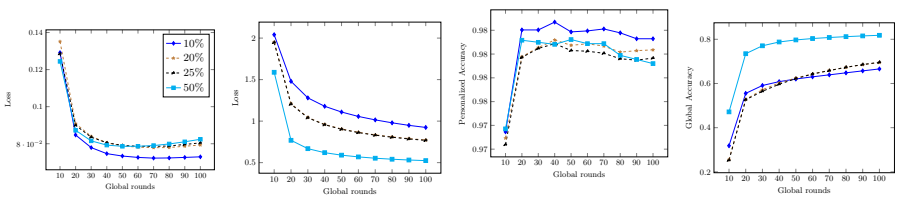


(a) Personalized Loss (Validation) (b) Global Loss (Validation) (c) Personalized Accuracy (Validation) (d) Global Accuracy (Validation)

Figure 30: Full participation teams and devices on synthetic datasets in convex settings (MCLR)

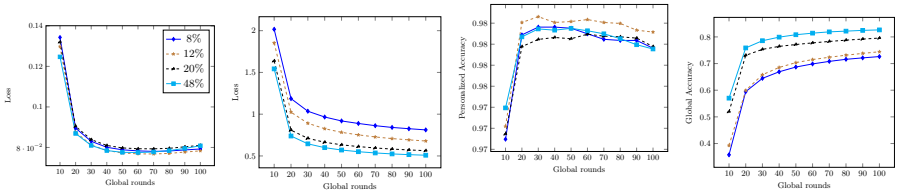
Appendix A.5.2 Full participation of teams and partial participation of devices

Here, we conducted experiments in convex settings using the MCLR approach; we observed that PerMFL with a low percentage of device participation converges slower compared to a high percentage of device participation (see Figure 31 to Figure 42). Increasing the number of participating devices resulted in faster convergence of the global model. Therefore, to achieve a better global model with faster convergence, it is recommended to increase the number of device participants per team iteration.



(a) Personalized Loss (Validation) (b) Global Loss (Validation) (c) Personalized Accuracy (Validation) (d) Global Accuracy (Validation)

Figure 31: Full team participation (5 teams) but partial devices participation on FMNIST datasets in convex settings (MCLR)

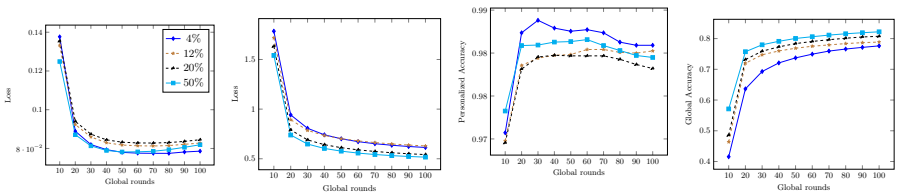


(a) Personalized Loss (Validation) (b) Global Loss (Validation) (c) Personalized Accuracy (Validation) (d) Global Accuracy (Validation)

Figure 32: Full team participation (4 teams) but partial devices participation on FMNIST datasets in convex settings (MCLR)

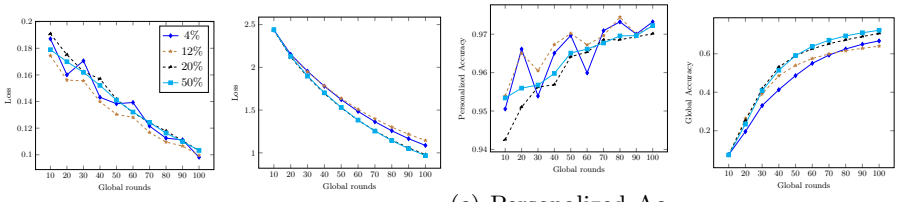
Appendix A.5.3 Partial participation of teams and full participation of devices

We conducted a series of experiments in both convex (see Figure 43, Figure 45, and Figure 47) and non-convex settings (see Figure 44, Figure 46, and Figure 48) using EMNIST, MNIST, and FMNIST datasets. From there, we observed if the participation of teams is limited, then the personalized and global model



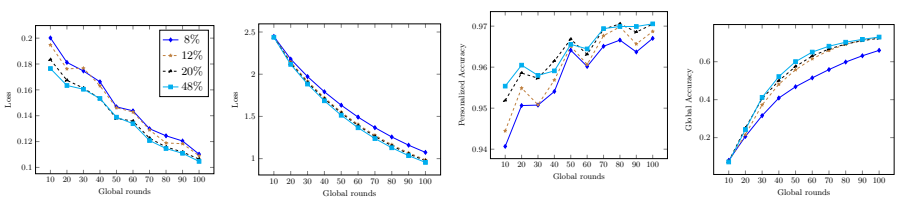
(a) Personalized Loss (Validation) (b) Global Loss (Validation) (c) Personalized Accuracy (Validation) (d) Global Accuracy (Validation)

Figure 33: Full team participation (2 teams) but partial devices participation on FMNIST datasets in convex settings (MCLR)



(a) Personalized Loss (Validation) (b) Global Loss (Validation) (c) Personalized Accuracy (Validation) (d) Global Accuracy (Validation)

Figure 34: Full team participation (2 teams) but partial devices participation on MNIST datasets in convex settings (MCLR)



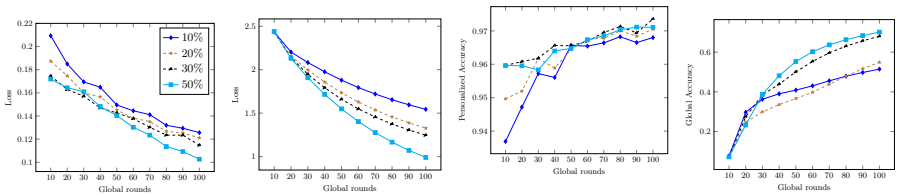
(a) Personalized Loss (Validation) (b) Global Loss (Validation) (c) Personalized Accuracy (Validation) (d) Global Accuracy (Validation)

Figure 35: Full team participation (4 teams) but partial devices participation on MNIST datasets in convex settings (MCLR)

both converges slowly. The same behaviour has been observed in all datasets for both convex and non-convex experiments (see Figure 43 to Figure 48).

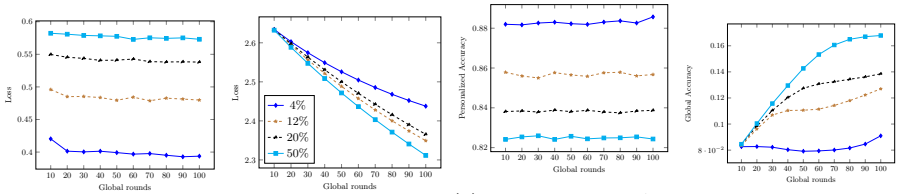
Appendix A.5.4 Partial participation of teams and devices

We conducted experiments for convex scenarios on EMNIST, MNIST, and FMNIST datasets (see Figure 49, Figure 50, and Figure 51). From the experiments,



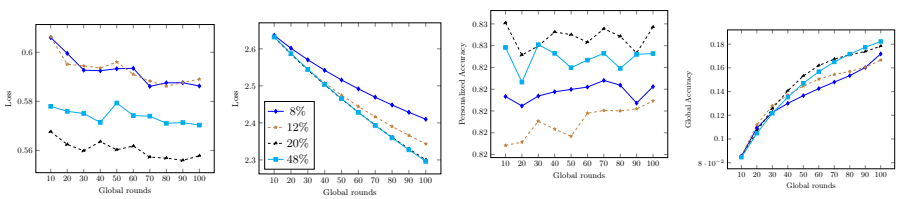
(a) Personalized Loss (Validation) (b) Global Loss (Validation) (c) Personalized Accuracy (Validation) (d) Global Accuracy (Validation)

Figure 36: Full team participation (5 teams) but partial devices participation on MNIST datasets in convex settings (MCLR)



(a) Personalized Loss (Validation) (b) Global Loss (Validation) (c) Personalized Accuracy (Validation) (d) Global Accuracy (Validation)

Figure 37: Full team participation (2 teams) but partial devices participation on Synthetic datasets in convex settings (MCLR)

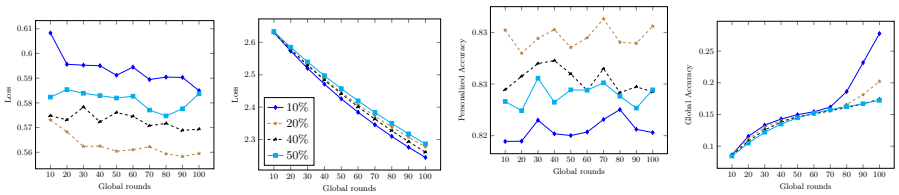


(a) Personalized Loss (Validation) (b) Global Loss (Validation) (c) Personalized Accuracy (Validation) (d) Global Accuracy (Validation)

Figure 38: Full team participation (4 teams) but partial devices participation on Synthetic datasets in convex settings (MCLR)

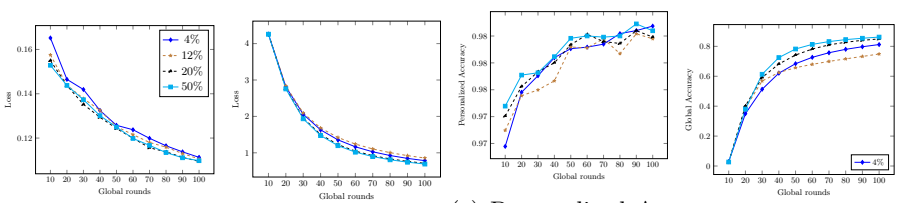
it was observed that when 20% of teams participated in each global round, increasing the number of devices participating in each team iteration resulted in improved convergence of both personalized and global models.

Discussions: Based on our empirical observations, we conclude that the convergence and performance of PerMFL are optimal when both teams and devices are fully present throughout the entire global iterations. However, PerMFL also demonstrates good performance even with variations in team and



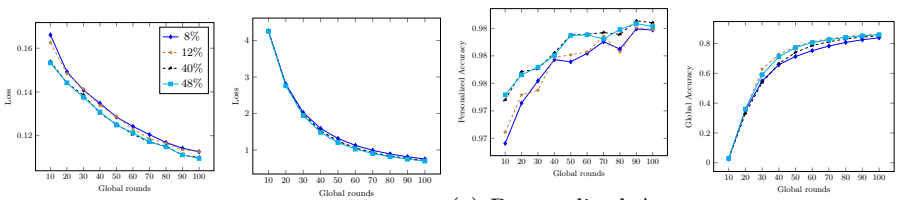
(a) Personalized Loss (Validation) (b) Global Loss (Validation) (c) Personalized Accuracy (Validation) (d) Global Accuracy (Validation)

Figure 39: Full team participation (5 teams) but partial devices participation on synthetic datasets in convex settings (MCLR)



(a) Personalized Loss (Validation) (b) Global Loss (Validation) (c) Personalized Accuracy (Validation) (d) Global Accuracy (Validation)

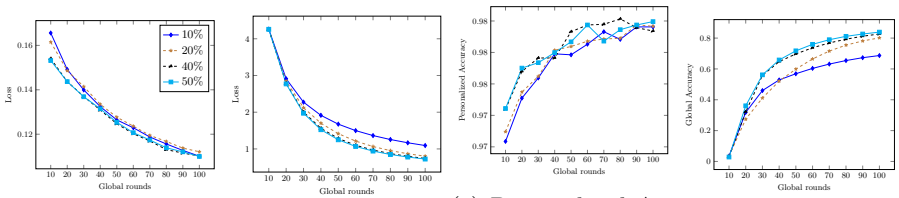
Figure 40: Full team participation (2 teams) but partial devices participation on EMNIST datasets in convex settings (MCLR)



(a) Personalized Loss (Validation) (b) Global Loss (Validation) (c) Personalized Accuracy (Validation) (d) Global Accuracy (Validation)

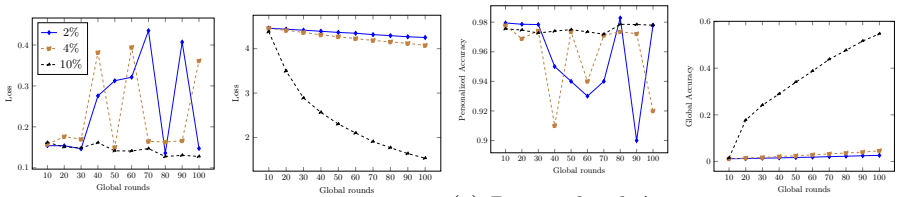
Figure 41: Full team participation (4 teams) but partial devices participation on EMNIST datasets in convex settings (MCLR)

device participation. When teams fully participate but there is partial device participation, PerMFL achieves fast convergence. On the other hand, when the number of participating teams is limited, the convergence of the global model is slower, requiring a higher number of global rounds to converge. It is important to note that the convergence of PerMFL is slowest when both teams and devices have very low participation (2%) in each global rounds.



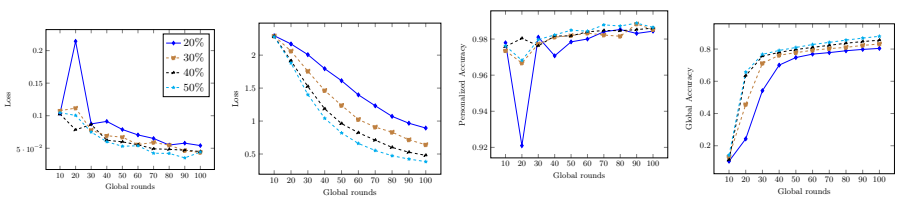
(a) Personalized Loss (Validation) (b) Global Loss (Validation) (c) Personalized Accuracy (Validation) (d) Global Accuracy (Validation)

Figure 42: Full team participation (5 teams) but partial devices participation on EMNIST datasets in convex settings (MCLR)



(a) Personalized Loss (Validation) (b) Global Loss (Validation) (c) Personalized Accuracy (Validation) (d) Global Accuracy (Validation)

Figure 43: Partial participation of team ($2\%, 4\%,$ *and* 10%) and full devices participation of devices on EMNIST datasets in convex settings (MCLR)

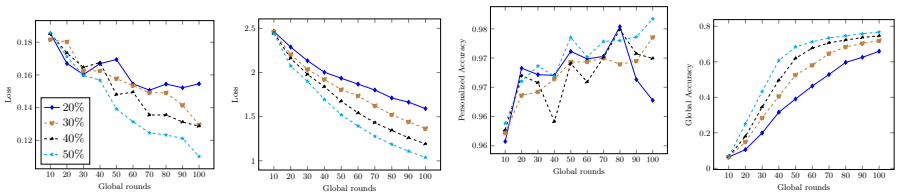


(a) Personalized Loss (Validation) (b) Global Loss (Validation) (c) Personalized Accuracy (Validation) (d) Global Accuracy (Validation)

Figure 44: Partial participation of team ($20\%, 30\%, 40\%,$ *and* 50%) and full devices participation of devices on EMNIST datasets in convex settings (CNN)

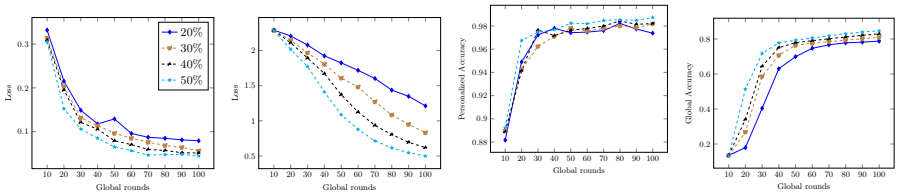
Appendix A.6 Effect of team iterations on convergence of PerMFL

In this study, experiments were conducted to investigate the impact of team iterations on the convergence of PerMFL. The objective was to gain insights into how varying the number of team iterations influences the convergence behaviour of the model. Here we studied (1) the effect of the number of team



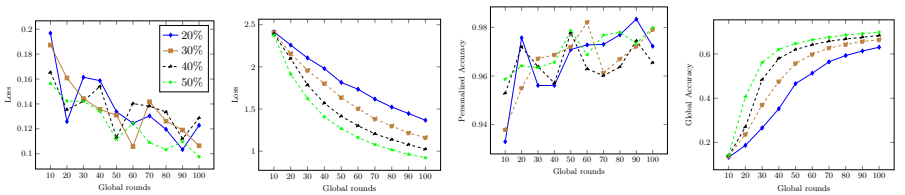
(a) Personalized Loss (Validation) (b) Global Loss (Validation) (c) Personalized Accuracy (Validation) (d) Global Accuracy (Validation)

Figure 45: Partial participation of team $(20\%, 30\%, 40\%,$ *and* $50\%)$ and full devices participation of devices on MNISTdatasets in convex settings (MCLR)



(a) Personalized Loss (Validation) (b) Global Loss (Validation) (c) Personalized Accuracy (Validation) (d) Global Accuracy (Validation)

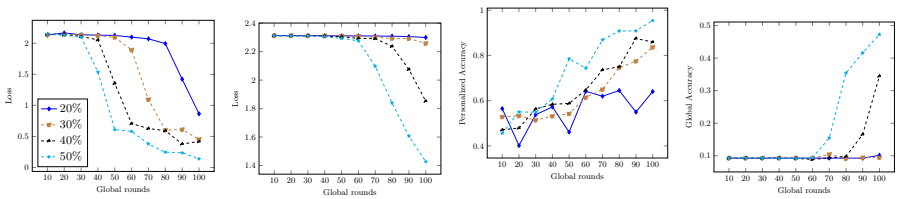
Figure 46: Partial participation of team $(20\%, 30\%, 40\%,$ *and* $50\%)$ and full devices participation of devices on MNIST datasets in convex settings (CNN)



(a) Personalized Loss (Validation) (b) Global Loss (Validation) (c) Personalized Accuracy (Validation) (d) Global Accuracy (Validation)

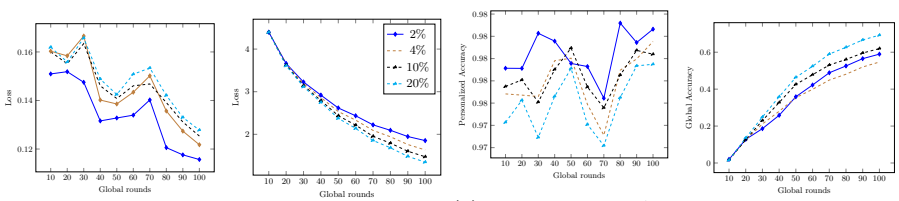
Figure 47: Partial participation of team $(20\%, 30\%, 40\%,$ *and* $50\%)$ and full devices participation of devices on FMNIST datasets in convex settings (MCLR)

iterations on the convergence of PerMFL when teams have full involvement but devices have fractional involvement in the entire learning process. (2) Effect of team iterations on the convergence of PerMFL while teams and devices both have fractional involvement. i.e., all devices are not participating in each global rounds only a fraction of devices are participating.



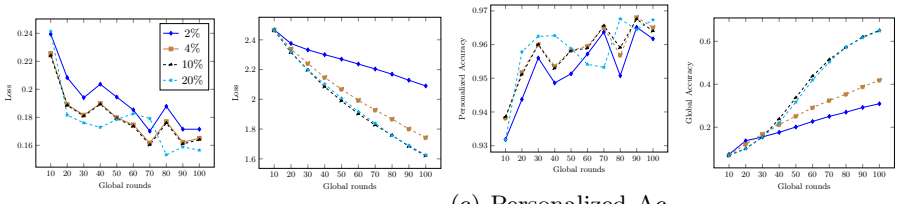
(a) Personalized Loss (Validation) (b) Global Loss (Validation) (c) Personalized Accuracy (Validation) (d) Global Accuracy (Validation)

Figure 48: Partial participation of teams $(20\%, 30\%, 40\%, \textit{and}\ 50\%)$ and full devices participation of devices on FMNIST datasets in convex settings (CNN)



(a) Personalized Loss (Validation) (b) Global Loss (Validation) (c) Personalized Accuracy (Validation) (d) Global Accuracy (Validation)

Figure 49: Partial participation of team (20%) and devices $(2\%, 4\%, 10\%, \textit{and}\ 30\%)$ on EMNIST datasets in convex settings (MCLR)

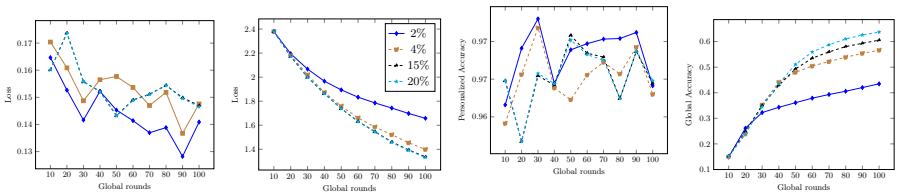


(a) Personalized Loss (Validation) (b) Global Loss (Validation) (c) Personalized Accuracy (Validation) (d) Global Accuracy (Validation)

Figure 50: Partial participation of team (20%) and devices $(2\%, 4\%, 10\%, \textit{and}\ 30\%)$ on MNIST datasets in convex settings (MCLR)

Appendix A.6.1 Effect of number of team iterations on the full participation of teams and partial participation of devices.

In our experiments (see Figure 52, Figure 53 Figure 54, Figure 55, and Figure 56), we observed for both convex and non-convex scenarios, the convergence of PerMFL's personalized and global model improves if we increase the team iter-



(a) Personalized Loss (Validation)

(b) Global Loss (Validation)

(c) Personalized Accuracy (Validation)

(d) Global Accuracy (Validation)

Figure 51: Partial participation of team (20%) and devices ($2\%, 4\%, 15\%,$ *and* 30%) on FMNIST datasets in convex settings (MCLR)

ations. This improvement occurs because when we increase the team iterations, more devices within the teams can actively contribute to the process.

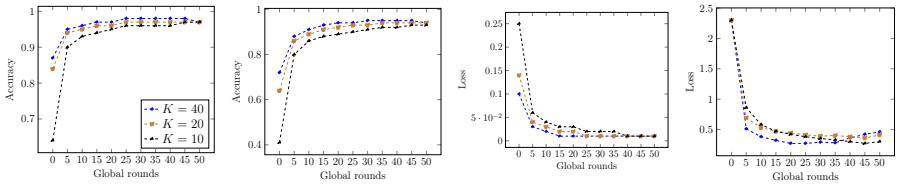
Appendix A.6.2 Effect of number of team iterations on the partial participation of teams and devices

In our experiments, we specifically investigated the impact of low team and device participation on the convergence of the global model. The objective was to determine whether increasing the number of team iterations could expedite the convergence process. Our findings, as depicted in Figure 57 and Figure 59, indicate that when team participation is set at 20%, increasing the team iterations lead to improved convergence of the global model. However, when team participation is extremely low (2%), as shown in Figure 58 and Figure 60, simply increasing team iterations is insufficient. In such cases, a higher number of global iterations is necessary to achieve convergence for the global model.

Discussions: Based on our findings, we can infer that team iterations play a crucial role in improving the performance of both the global and personalized models. However, when there is limited participation from teams and devices in each global round, relying solely on team iterations is inadequate. In such cases, it becomes necessary to increase the number of global rounds to enable more teams to participate and, consequently, enhance the performance of PerMFL.

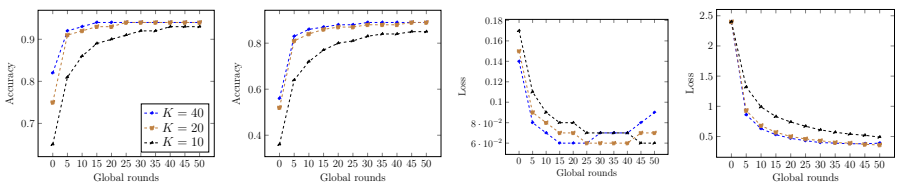
Appendix A.7 Convergence analysis

Based on the convergence results presented in Figure 62, Figure 61, and Figure 63, we observed that PerMFL(PM) achieved faster convergence compared to AL2GD. Additionally, the convergence of PerMFL(GM) and h-SGD was found to be similar for both strongly convex and non-convex scenarios.



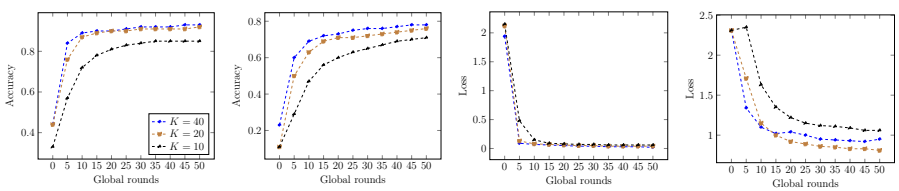
(a) Personalized Accuracy (Validation) (b) Global Accuracy (Validation) (c) Personalized Loss (Validation) (d) Global Loss (validation)

Figure 52: Effect of team iterations on the convergence of PerMFL in non-convex settings (CNN) on MNIST while teams and devices have full participation



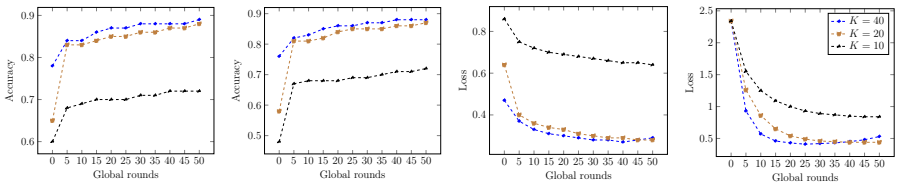
(a) Personalized Accuracy (b) Global Accuracy (Validation) (c) Personalized Loss (Validation) (d) Global Loss (Validation

Figure 53: Effect of team iterations on the convergence of PerMFL in strongly convex (MCLR) settings on MNIST while teams and devices have full participation



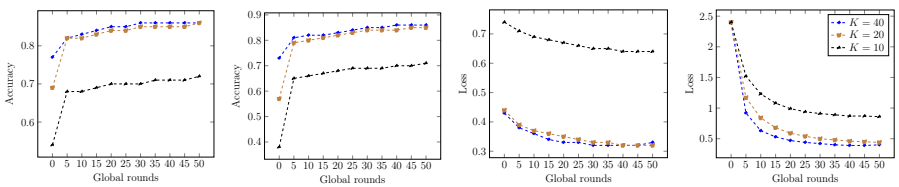
(a) Personalized Accuracy (Validation) (b) Global Accuracy (Validation) (c) Personalized Loss (Validation) (d) Global Loss (Validation)

Figure 54: Effect of team iterations on the convergence of PerMFL in non-convex settings (CNN) on FMNIST while teams and devices have full participation



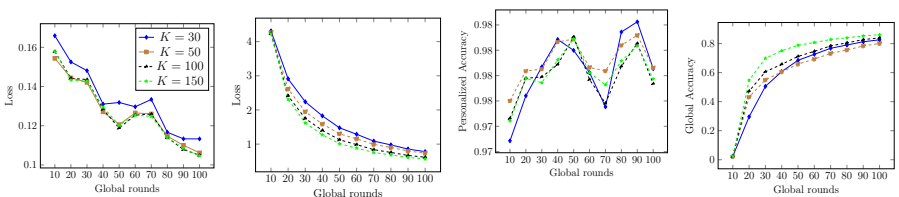
(a) Personalized Accuracy (Validation) (b) Global Accuracy (Validation) (c) Personalized Loss (Validation) (d) Global Loss (Validation)

Figure 55: Effect of team iterations on the convergence of PerMFL in non-convex settings (DNN) on Synthetic dataset while teams and devices fully participate.



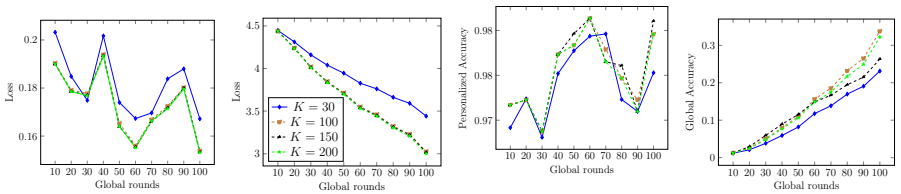
(a) Personalized Accuracy (Validation) (b) Global Accuracy (Validation) (c) Personalized Loss (Validation) (d) Global Loss (Validation)

Figure 56: Effect of team iterations on the convergence of PerMFL in strongly convex settings (MCLR) on Synthetic dataset while teams and devices fully participate.



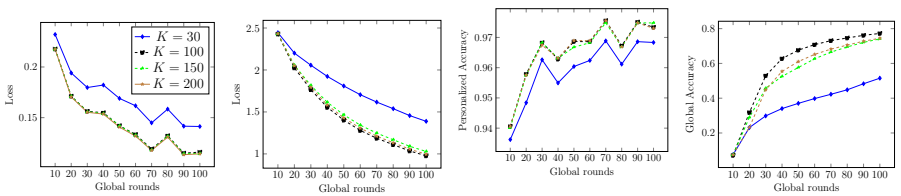
(a) Personalized Loss (Validation) (b) Global Loss (Validation) (c) Personalized Accuracy (Validation) (d) Global Accuracy (Validation)

Figure 57: Effect of team iterations $\{K = 30, 50, 100,$ *and* $150\}$ when team (20%) and devices (2%) are partially participated in the PerMFL using EMNIST datasets in convex settings (MCLR)



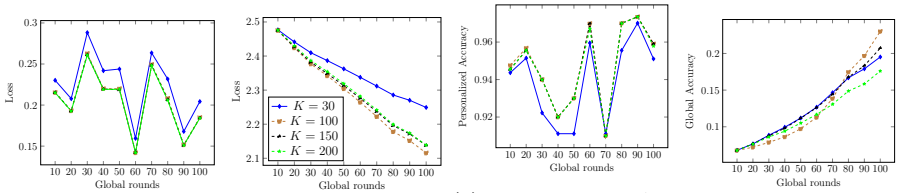
(a) Personalized Loss (Validation) (b) Global Loss (Validation) (c) Personalized Accuracy (Validation) (d) Global Accuracy (Validation)

Figure 58: Effect of team iterations $\{K = 30, 100, 150$ *and* $200\}$ when team (2%) and devices (2%) are partially participated in the PerMFL using EMNIST datasets in convex settings (MCLR)



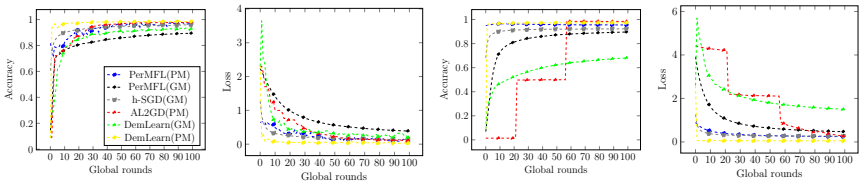
(a) Personalized Loss (Validation) (b) Global Loss (Validation) (c) Personalized Accuracy (Validation) (d) Global Accuracy (Validation)

Figure 59: Effect of team iterations $\{K = 30, 100, 150$ *and* $200\}$ when team (20%) and devices (2%) are partially participated in the PerMFL using MNIST datasets in convex settings (MCLR)



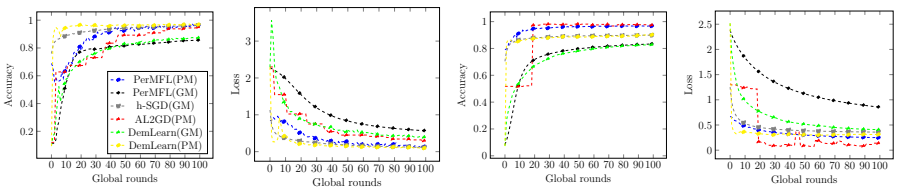
(a) Personalized Loss (Validation) (b) Global Loss (Validation) (c) Personalized Accuracy (Validation) (d) Global Accuracy (Validation)

Figure 60: Effect of team iterations $\{K = 30, 100, 150$ *and* $200\}$ when team (2%) and devices (2%) are partially participated in the PerMFL using EMNIST datasets in convex settings (MCLR)



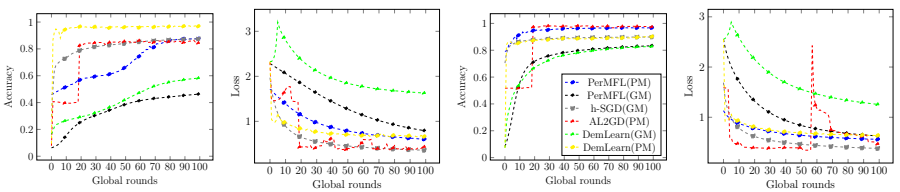
(a) non-convex (b) non-convex (c) strongly convex (d) strongly convex

Figure 61: Convergence comparison of PerMFL with multi-tier SOTA in strongly convex and non-convex settings on EMNIST



(a) Non-Convex (b) Non-convex (c) strongly convex (d) strongly convex

Figure 62: Convergence comparison of PerMFL with multi-tier SOTA in strongly convex and non-convex settings on MNIST



(a) Non-Convex (b) Non-convex (c) strongly convex (d) strongly convex

Figure 63: Convergence comparison of PerMFL with multi-tier SOTA in strongly convex and non-convex settings on synthetic datasets

Appendix A.8 Performance analysis on FEMNIST and CIFAR100 datasets

We have experimented the performance of PerMFL with the state-of-the-art multi-tire federated learning algorithms on Federated EMNIST (FEMNIST) and CIFAR100 datasets (see Table 3) for the strongly convex setup. From there, we observed PerMFL(PM) produce better results than AL2GD. h-sgd produces a better global model than PerMFL. For the CIFAR100 dataset, AL2GD overcame the performance of PerMFL(PM). The performance of PerMFL(GM) and h-SGD both are equivalent.

Table 3: Performance comparison of PerMFL with SOTA. (Validation accuracy(mean/std))

Algorithm	MCLR (Strongly convex)	
	FEMNIST	CIFAR100
h-SGD (GM)	0.6405(±0.005)	0.1232(0.001)
AL2GD (PM)	0.4467(±0.01)	0.65.87(±0.07)
PerMFL (GM)	0.5757 (±0.0)	0.1368 (±0.0)
PerMFL (PM)	0.8129 (±0.0)	0.6695 (±0.001)

Paper V

Predicting Event Memorability using Personalized Federated Learning

Sourasekhar Banerjee, Debaditya Roy, Vigneshwaran Subbaraju, and Monowar Bhuyan

Submitted for publication, 2024.

Predicting Event Memorability using Personalized Federated Learning

Sourasekhar Banerjee*, and Debaditya Roy†, and Vigneshwaran Subbaraju†, and Monowar Bhuyan*

* *Department of Computing Science, Umeå University, Umeå, Sweden*

† *A*STAR Institute of High Performance Computing (IHPC), Singapore*

sourasb@cs.umu.se, Roy_Debaditya@ihpc.a-star.edu.sg, Vigneshwaran_Subbaraju@ihpc.a-star.edu.sg, monowar@cs.umu.se

Abstract: Lifelog images are very useful as memory cues for recalling past events. Estimating the level of event memory recall induced by a given lifelog image (event memorability), is useful for selecting images for cognitive interventions. Previous works for predicting event memorability follow a centralised model training paradigm that requires several users to share their lifelog images. This risks violating the privacy of individual lifeloggers. Alternatively, a personal model trained with a lifelogger's own data guarantees privacy. However, it imposes significant effort on the lifelogger to provide a large enough sample of self-rated images to develop a well-performing model for event memorability. Therefore, we propose a clustered personalized federated learning setup FedMEM, that avoids sharing raw-images but still enables collaborative learning via model sharing. For an enhanced learning performance in the presence of data heterogeneity, FedMEM evaluates similarity among users to groups them into clusters. We demonstrate that our approach furnishes high-performing personalized models compared to state-of-the-art.[1]

Key words: Federated Learning, Event Memorability, Clustered Federated Learning, Classification, Lifelogs

1 Introduction

Personal photos or images are a commonly used medium that aid the experience of re-living memories from past events in a person's life. Visual lifelogging — the act of automatically and periodically capturing images using body-worn cameras, smart-glasses, etc., even as one goes about their regular life, is a potent tool for acquiring, storing and reviewing image cues that represent past

[1]Source code is available in `https://github.com/sourasb05/FedMEM.git`

moments from one's own life. Reviewing lifelog images is found to be useful for memory interventions [HLW16; LD07; Sil+18] by exploiting the ability of lifelog image cues to induce a vivid memory recall of the corresponding event in real-life. Serious games for general cognitive training [Xu+18] are also found to benefit from the use of lifelog image content by improving the enjoyment and engagement of users. However, since lifelog images are acquired automatically during the regular life of a person, a large part of the image collection comes from mundane events that may not be memory-worthy. Therefore, not every lifelog image may serve as an useful cue for memory recall. The delayed review of a given lifelog image can induce an event memory recall that varies from 'vivid recall of several episodic details about the event' to 'absolutely no memory of the event'. Hence, estimating the level of event memory recall induced by a given lifelog image – referred to as the *event memorability* of the image, is very useful for selecting images for memory or cognitive intervention. Recently, computational models have been developed [Xu+21] for this purpose such that the model is able to predict event memorability on a discrete, ordinal scale of 0 to 9 for a given lifelog image, as illustrated in Figure 1.

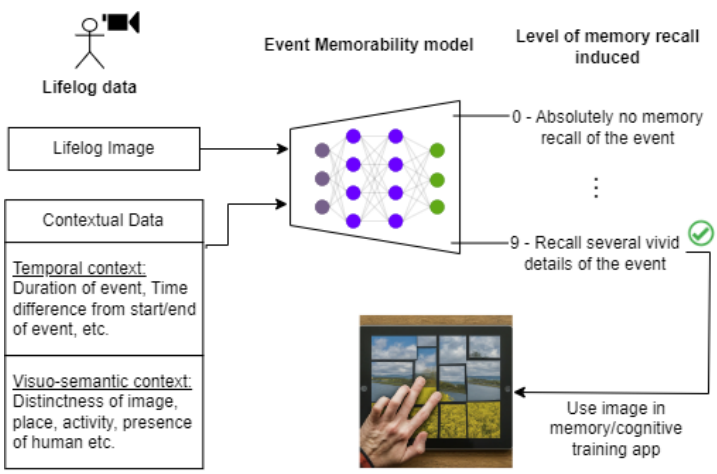
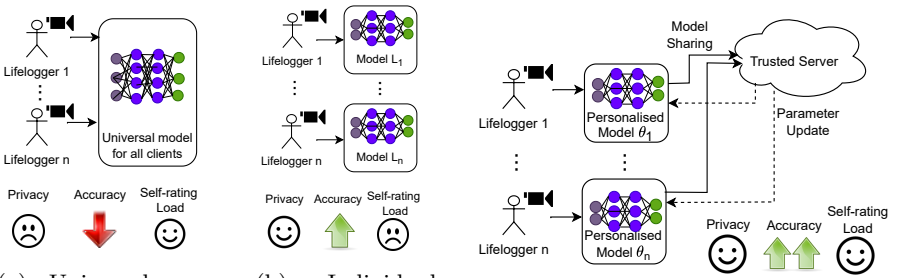


Figure 1: The event memorability model predicts the level of event memory recall induced by a given lifelog image. Apart from the raw image features, contextual factors such as distinctness of a lifelog image with respect to other lifelog images from the same user, temporal positioning of the lifelog image frame in relation to the start/end of the event, duration of the event, activity, place, presence of humans, etc. can also help in computing memory recall. Event memorability model can help in choosing the most useful images as cues for memory or cognitive intervention.

Essentially, these models are trained using a centralized approach as shown in Figure 2a, where data from the personal lifelogs of several users are aggregated together along with their self-reported scores of the induced memory recall (on



(a) Universal centralized model

(b) Individual silo models

(c) Federated personalized models

Figure 2: Three approaches to develop event memorability models with different implications for privacy, accuracy of event memorability prediction and the life-loggers' effort involved in providing sufficient number of images with self-rated event memorability scores (self-rating load). Personalised model development provides better privacy and boosts accuracy of individual models, while collaboration across clients reduces life-loggers' effort needed in providing self-rated images and improves model generalisation. We propose a federated learning-based personalized model development approach that simultaneously preserves privacy and also benefits from collaboration.

a scale of 0 to 9), to train the model. Such an approach suffers from two main disadvantages — (a) the inherent privacy risk in sharing sensitive life-log data makes the users wary of submitting their data and (b) while there are obvious benefits with regards to generalization and the diversity of the data used in developing an universal event memorability model, the lack of personalization in such a *one-size-fits-all* approach may lead to sub-optimal performance for individual lifeloggers, since event memorability is a highly personal phenomena.

As an alternative to the centralized approach, we can consider a siloed approach (illustrated in Figure 2b), where each lifelogger uses only their own data for building their personal models of event memorability. Such siloed models grant maximum privacy and highest level of personalization to the life-loggers. However, they would suffer from (a) poor generalization performance for images acquired from rare contexts appearing in a individual lifelogger's data and (b) the requirement of an extensive amount of self-rated data from the lifelogger to comprehensively cover diverse contexts of lifelog images to attain optimal performance. Unlike other common tasks in computer vision where the data labelling for model training can be done by professional or crowdsourced annotators, the self-rating of event memory has to be provided by the end-users (lifeloggers) themselves. The significant labor involved in such self-rating may hinder the adoption of the models. Therefore, we propose a federated learning-based approach (Figure 2c) for event memorability prediction where only model parameters and not images across the lifeloggers via a trusted global server. Through model sharing, we preserve privacy of the raw image data as well as

derive the performance benefits of collaborative model development.

Federated learning is a collaborative machine learning framework in which multiple clients participate in learning while maintaining privacy [McM+17] of their individual data. In a federated learning setup, clients (individual lifeloggers) contribute by sharing gradients or model weights of their locally trained model with the central server while maintaining full control over their data throughout the training phase. Only the encrypted model is disclosed to the server; the raw images remain private. This ensures that sensitive information stays with the clients. Additionally, it allows clients to participate in the learning process for its entirety or only a portion, as per their availability or preference.

Intuitively, Federated Learning (FL) is challenging when there is statistical heterogeneity, i.e., non-independent and identical distribution (non-i.i.d.) of data across clients. Such heterogeneity is expected in lifelogs where age, lifestyle, occupation, and activity level can cause vast differences in the events captured throughout the day. As a result, an FL global model may exhibit client drift – divergence of a client's model from the global model over time due to the local updates based on the client's unique data distribution. This drift causes FL to result in sub-optimal client models and global model as the aggregation becomes less effective for individual clients. Personalized FL addresses this problem and enables the global model to learn from this heterogeneous data while allowing for client models to retain their unique characteristics based on their own user data [TTN20; BYB22]. In literature [Ngu+22; Lon+23], clustering client models have been demonstrated to enhance personalization when clients contain non-i.i.d. data.

Therefore, we propose FedMEM, a personalized and clustered federated learning approach for event memorability prediction. We propose that lifeloggers share their model parameters in clusters that are formed using either the distribution of memorability scores across lifeloggers or model similarity across lifeloggers. We show that personalized event memorability models learned with either of the clustering criteria using FedMEM outperform other clustered FL approaches. Furthermore, we show FedMEM's efficacy as client models' performance improves even when there are a small number of participating clients. To the best of our knowledge, this is the first work on predicting event memorability using federated learning. Our key contributions are as follows:

- We formulated the event-memorability prediction as a personalized clustered federated learning problem where each client trains the personalized model, the server aggregates, and makes clusters of devices for better personalization. We show that this approach enhances privacy and improves model performance.
- We introduce a similarity measure that balances three criteria across personalized models – intra-cluster similarity, inter-cluster dissimilarity, and similarity to the global model.
- FedMEM outperforms the baseline centralized model [Xu+21] as well as SOTA clustered federated learning algorithms such as DemLearn, h-SGD,

and multi-centered FL and traditional federated learning algorithms like FedAvg, pFedMe, and FedProx.

2 Related Work

2.1 Event memorability from visual lifelogs

Memorability as an inherent characteristic of images and videos has been the subject of several prior works such as [BIO13; Iso+11; Iso+13]. Computational models that predict image/video memorability [Kho+15; Coh+19] have been developed by utilizing advanced techniques such as deep convolutional neural networks (CNN) and carefully designed content analysis mechanisms [Faj+18; Squ+18; Jin+16; Lu+20; KAC20; ABF22]. However, these works measure the intrinsic property of a given arbitrary image/video stimulus to stick into human memory in general. It is very important to distinguish this large body of work from the study of *event memorability* - the autobiographic memory recall that is induced by the review of visual lifelogs acquired from their personal lives, as it deals with different memory phenomena [BR18]. For example, viewing a personal lifelog image that was acquired during a leisurely stroll, could invoke memories of several associated details, such as the sounds of birds, the scent of flowers, etc., which may not even appear in the lifelog image.

Visual lifelogs acquired from wearable cameras and their ability to induce event memory recall have been studied in several works such as [Doh+12; Mil+11; Ris+16; Xu+21]. Despite the research interest, owing to difficulties in lifelog data collection and sharing, the only publicly available dataset which contains lifelog images along with self-reported event memorability annotations, is the R3 dataset [Xu+21; GLT18]. Using the R3 dataset, a Contextual Event Memory Network (CEMNet) [Xu+21] has been developed to process the multimodal input and predict the associated self-reported event memorability. The same dataset has also been used to show that personalized visual lifelog content can significantly enhance the engagement of users in serious games targeting cognitive training [Xu+18]. Despite compelling uses in memory augmentation [HLW16], privacy remains a major concern that hinders user adoption [FCJ17]. Therefore, in contrast to the centralised approach of CEMNet [Xu+21], which requires the lifeloggers to share raw image data, we explore privacy preserving machine learning approaches for developing computational models of event memorability.

2.2 Clustered and personalized FL

Federated learning is a privacy-preserving approach to distributed machine learning. FedAvg [McM+17] serves as a fundamental baseline in Federated Learning (FL) due to its simplicity and reduced communication overhead. Yet,

it struggles with data heterogeneity [Li+20], resulting in inconsistent performance and causes client drift [Kar+20]. This occurs when a global model fails to effectively serve all clients, and shifted the objective towards some clients, rest are not benefiting from a single global model. To mitigate these challenges, FL has progressively shifted towards adopting personalized models [TTN20]. Recent days, clustered personalized Federated learning algorithms produce promising solution to this problem [BYB22; Ngu+22]

Clustered Federated Learning organizes clients into clusters based on their similarity in data or model. Inside each cluster, data distribution in each client is non-IID. Clients train their local models independently on their personal data.

Clustered FL methods can be classified into four categories based on the cluster formation criteria. (1) **Distance between local and global model parameters.** In [Lon+23], the client's association in the cluster is measured by the distance between the local model and global models. Instead of a single global model, it offers multiple global models. A client chose the nearest global model to form a cluster. In [BFA20], devices are grouped using hierarchical clustering based on the similarity of their local updates. In [Ngu+22], Democratized Learning (DemLearn) was introduced by integrating self-organized hierarchical structuring, personalization, and hierarchical generalization. (2) **Partition based on gradient information:** [SMS20] used the cosine similarity between device's gradient updates. FedGroup and FedGroupProx [Dua+21] create clusters based on the similarity between the clients' optimization direction. It quantifies the similarity using the Euclidian distance of decomposed cosine similarity (EDC) between devices. (3) **Clustering based on the training loss:** In HyperCluster ([Man+20]), each client is evaluated themselves with multiple cluster models using their data, and select a cluster which that gives minimum loss, [Gho+20] proposed a similar concept to cluster clients. (4) **Clustering based on the information about the data:** [Hua+19] apply clustering of patients in FL, based on their electronic medical records.

We propose two new clustering schemes for our clustered FL approach – Model Similarity based Clustering (MSC) falls under the first category and Memorability Score Distribution-based Clustering (MSDC) that falls under fourth category of clustered Federated Learning approaches.

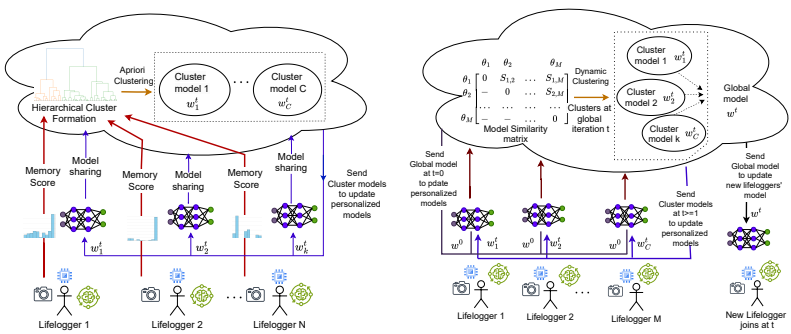
3 FedMEM: The Proposed Approach

In this section, we describe FedMEM, a personalized clustered federated learning approach for predicting event memorability from life-logs.

3.1 Framework

FedMEM follows the client-server federated learning framework (see Figure 3) that works in two phases. In phase 1, lifeloggers use their own data to train personalized local models (θ_i). In phase 2, server collects the personalized models

and assigns lifeloggers into clusters and aggregates their models $(\omega_1, \ldots, \omega_C, \omega)$. FedMEM employs two distinct clustering criteria: (1) Memorability Score Distribution-based lifelogger Clustering (MSDC, see Figure 3a), where lifeloggers transmit their memorability score distributions to the server before the beginning of the FL process and lifeloggers are clustered according to the similarities in their memory scores, and (2) Model Similarity-based Clustering (MSC, see Figure 3b), in which lifeloggers send their personalized models to the server and lifeloggers' are clustered based on the similarity of their model weights. MSC offers complete data privacy as the server does not have access to the data distribution among lifeloggers. MSDC has the advantage of conducting the clustering process just once before the commencement of Federated Learning, allowing lifeloggers to be aware of their cluster models in advance.



(a) FedMEM+MSDC (Apriori Clustering) (b) FedMEM+MSC (Dynamic Clustering)

Figure 3: Framework of FedMEM

In both the FedMEM + MSDC and FedMEM + MSC methods, the server calculates cluster models $(\omega_1, \ldots, \omega_C)$ and distributes them to the corresponding lifeloggers. The FedMEM + MSDC approach does not require a global model because every lifelogger is assigned to a cluster before the FL begins (Apriori clustering), and lifelogger never changes the cluster. FedMEM + MSC method transmits the global model to lifeloggers at the beginning (ω^0) of the FL or when a new lifelogger enters the FL process (ω^t) at global iteration t, as a reference to initialize the local models $(\theta_i$'s). After that, in every global iteration (t), only cluster models (ω_c^t) are transferred to the respective lifeloggers in the clusters. After cluster assignment is done, Each lifelogger trains personalized model (θ_i^t) and transfers it to the server for aggregation.

3.2 Clustering

Memorability-score distribution based clustering (MSDC) is performed before initiating the federated learning process. At first, each lifelogger sends

the distribution of memory scores based on their own training data (CD_i) to the server. The server computes the similarity between the memory score distributions (CD_i and CD_j) from lifelogger Lg_i and Lg_j respectively using KL divergence. A similarity matrix using KL divergence values is generated between every pair of lifeloggers. Hierarchical clustering is then applied to the similarity matrix to form a cluster tree of lifeloggers based on memory score distributions. We cut the cluster tree at a chosen height (empirically determined) to obtain C partitioned clusters of lifeloggers at a selected precision. Each lifelogger is assigned a cluster for the entirety of the FL process. Sharing memory score distributions across lifeloggers or to the server does not reveal the image content. Entire MSDC process is shown in Algorithm 1.

Algorithm 1 Memorability Score Distribution based Clustering (MSDC) of lifeloggers

Initialize: $\mathcal{N}$, $\forall_{i=0}^{\mathcal{N}} CD_i$
procedure MSDC($(\mathcal{N})$)
    Server collects the memory score distributions (CD_i) from $\mathcal{N}$ lifeloggers.
    for $i \leftarrow 0$ **to** $\mathcal{N}-1$ **do**
        for $j \leftarrow 0$ **to** $\mathcal{N}-1$ **do**
            $KL[i,j] \leftarrow$ KL_divergence(CD_i, CD_j)
        end for
    end for
    clusters $\leftarrow$ Hierarchical_Clustering(KL)
    Return clusters
end procedure

Model similarity-based clustering (MSC) begins by gathering the personalized models of the lifeloggers, which means that lifeloggers are not required to separately transmit their memory score distributions before the start of the Federated Learning process. The server computes a similarity graph whose edges are the similarity scores between two lifelogger's personalized models that is computed as:

$$\begin{aligned} s(\theta_i, \theta_j, \omega, \omega_c) =& (1-\lambda_1-\lambda_2) sim(\theta_i, \theta_j) \\ &+ \lambda_1 sim(\theta_i - \omega, \theta_j - \omega) \\ &+ \lambda_2 sim(\theta_i - \omega_c, \theta_j - \omega_c). \end{aligned} \tag{1}$$

Here, $sim(m,n) = exp(-\frac{||m-n||^2}{2\sigma^2})$ is the radial basis function, $sim(\theta_i, \theta_j)$ is the similarity between the personalized models (θ_i and θ_j) of two lifeloggers. $sim(\theta_i - \omega, \theta_j - \omega)$ measures the similarity between two lifeloggers by considering the deviation of their personalized model (θ_i) from the global model (ω). Similarly, $sim(\theta_i - \omega_c, \theta_j - \omega_c)$ measures the similarity between lifeloggers (θ_i, θ_j) based on their distance from the cluster model (ω_c). λ_1 and λ_2 serve as regularization factors, with the constraint that $\lambda_1 + \lambda_2 \leq 1$.

We compute a similarity graph $\mathcal{G}$ for $\mathcal{N}$ lifeloggers represented as a symmetric similarity matrix $S \in \mathbb{R}^{(\mathcal{N}-1)\times(\mathcal{N}-1)}$. Every $\{i,j\}^{th}$ element ($S[i,j] =$

$s(\theta_i, \theta_j, \omega, \omega_c))$ of the similarity matrix contains the similarity between two lifeloggers Lg_i and Lg_j computed using Equation (1). We employ spectral clustering [SM00] to cluster $\mathcal{G}$ based on S and a specified number of clusters (C) to assign a cluster to each lifelogger. The clustering process of MSC is repeated at every global iteration and lifeloggers are flexible to change clusters if a closer cluster center is identified. Entire MSC process is shown in Algorithm 2.

Algorithm 2 Model Similarity-based Clustering (MSC) of lifeloggers

procedure MSC($\theta_1, \theta_2, \ldots, \theta_M$)
    After a global iteration, the server collects client models from M clients $(\theta_i, \theta_j, ..., \theta_M)$ where $M \leq \mathcal{N}$.
    for $i \leftarrow 0$ **to** len(M) $- 1$ **do**
        for $j \leftarrow 0$ **to** len(M) $- 1$ **do**
            $S[i, j] \leftarrow s(\theta_i, \theta_j, \omega, \omega_c)$
        end for
    end for
    clusters = Clustering($S[i, j], C$)
    Return clusters
end procedure

3.3 Personalized clustered federated learning

In clustered federated learning, the objective function is to find the global model (ω) as follows:

$$\min_{\omega \in \mathbb{R}^d} f(\omega) \min_{\omega \in \mathbb{R}^d} \sum_{c=1}^{\mathcal{C}} \sum_{i=1}^{\mathcal{N}_c} r_c r_i f_i(\omega) \tag{2}$$

The function $f_i : \mathbb{R}^d \rightarrow \mathbb{R}$ denotes the local loss function computed for the model of lifelogger Lg_i in cluster c. The formulation involves a single global model (ω) that all lifeloggers agree on when the model converges. However, when data is heterogeneously across lifeloggers, the global model is less effective for all lifeloggers. We address this issue by introducing a cluster model (ω_c) for each cluster and reformulate Equation (2) as follows:

$$\min_{\omega_c \in \mathbb{R}^d} \min_{\theta_i \in \mathbb{R}^d} \sum_{c=1}^{C} \sum_{i=1}^{\mathcal{N}_c} r_c, r_i (f_i(\theta_i) + \frac{\eta}{2} ||\theta_i - \omega_c||^2) \tag{3}$$

The parameter $\eta \geq 0$ controls the contribution of the quadratic penalty $||\theta_i - \omega_c||^2$ to enforce that a lifelogger in a cluster has a personalized model close to cluster model.

FedMEM+MSC solves the optimization in Equation (3) as we compute a global model in addition to cluster models. But in FedMEM+MSDC, we do not compute a global model and the optimization is performed in each cluster

independently as follows:

$$\min_{\theta_i \in \mathbb{R}^d} \sum_{i=1}^{\mathcal{N}_c} r_i (f_i(\theta_i) + \frac{\eta}{2} ||\theta_i - \omega_c||^2). \tag{4}$$

Personalized update: For a cluster c at global round t, the cluster model is denoted as ω_c^t, and personalized model of lifelogger Lg_i as θ_i^t. The objective of personalized update is to solve the following objective:

$$\tilde{f}_i^{\eta}(\omega_c^t) := \min_{\theta_i^t \in \mathbb{R}^d} \; f_i(\theta_i^t) + \frac{\eta}{2} \|\theta_i^t - \omega_c^t\|^2. \tag{5}$$

We apply a gradient method to approximate the solution of Equation (5). We start with an initial value for the personalized model using the cluster model $\theta_i^{t,0} = \omega_c^t$. Using a positive learning rate α, we perform the following update for local iterations $l = 0, 1, \ldots, L-1$

$$\theta_i^{t,l+1} = \theta_i^{t,l} - \alpha_i \nabla f_i(\theta_i^{t,l}) - \alpha_i \eta (\theta_i^{t,l} - \omega_c^t). \tag{6}$$

Cluster update: For the cluster c, the objective of the cluster update for the global iteration t is to perform weighted aggregation over the personalized models.

$$F_c(\omega_c^t) := \sum_{i=1}^{\mathcal{N}_c} r_i \tilde{f}_i^{\eta}(\omega_c^t) \tag{7}$$

$$\omega_c^t := \sum_{i=1}^{\mathcal{N}_c} r_i \theta_i^t \tag{8}$$

where $\mathcal{N}_c$ represents the number of lifeloggers present in cluster c. $r_i = \frac{D_{Lg_i}}{D_c}$. D_{Lg_i} is the amount of data lifelogger Lg_i has. and D_c is the total amount of data the lifeloggers have in cluster c.

Global update: This is the weighted aggregation over the cluster models for global rounds $t = 0, \ldots, T$.

$$F(\omega^t) := \sum_{c=1}^{\mathcal{C}} r_c F_c(\omega_c^t). \tag{9}$$

$$\omega^t := \sum_{c=1}^{\mathcal{C}} r_c \omega_c^t. \tag{10}$$

where $r_c = \frac{D_c}{D}$, and D represents the amount of aggregated data from all lifeloggers across various clusters.

3.4 Fusing contextual information

The final personalized individual lifelogger models θ_i are obtained using only the raw-image features. However, each lifelog image in the R3 dataset is also associated with several other contextual data which have been shown to influence the estimation of event memorability. Measures of such contextual data have been provided in the R3 dataset for each lifelog image. Using these information, similar to what was done by the baseline CEMNet [Xu+21], we perform a late-fusion of the contextual features along with the image-based features that were obtained from the local personalised model for each lifelogger obtained from FedMEM. Figure 4 illustrates this late-fusion to predict the event memorability of a given lifelog image. This late fusion is performed in each individuals only after its final model θ_i is frozen. Several of these features are highly personal in nature such that they are relevant only for the individual lifelogger. For example, distinctness of an image can vary between individuals. It is also possible that these information may not be available for all lifeloggers. Hence, they were not used for the updates in the federated learning.

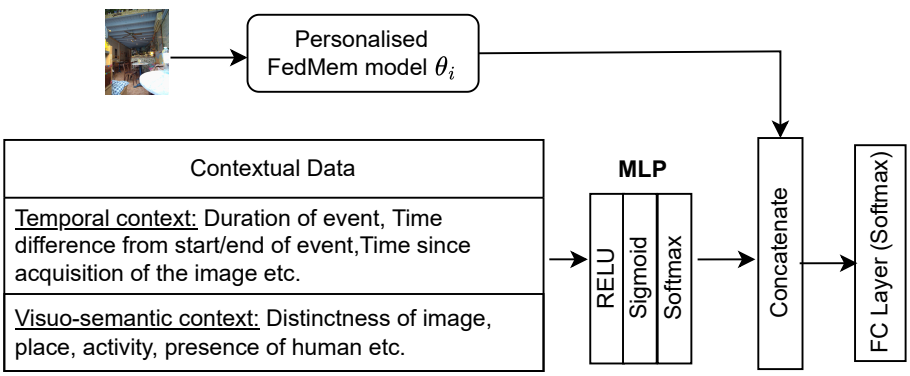


Figure 4: Late fusion of contextual information

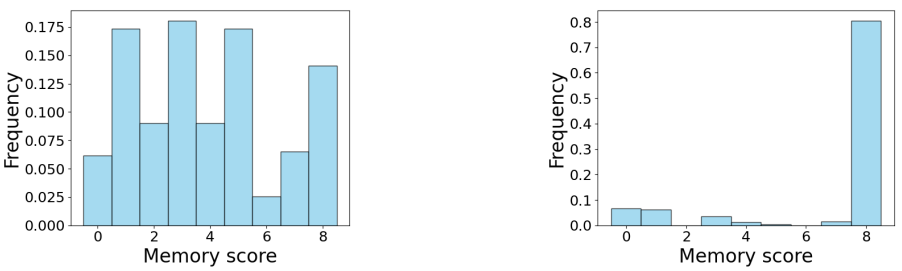
4 Experimental Evaluation

4.1 Dataset and model details

Dataset. Our experiments were conducted on the publicly available *R3* dataset [Xu+21]. This dataset is based on a user-study that involved 40 lifeloggers, each providing their own unique set of lifelog images acquired over a period of 2 to 4 weeks. A subset of approximately 250 images per lifelogger were selected and evaluated for event memorability via a self-reported score ranging from 0 to 9, where 0 indicates absolutely no recall and a score of 9 indicates that significant episodic details about the event were recalled by the lifelogger. In total,

there are about 10600 images with self-reported event memorability scores. The distribution of memory scores varies across the lifeloggers (as shown in Figure 5), leading to non-Independent and Identical (non-IID) and unbalanced data distributions among the lifeloggers. Apart from the image memorability scores, the dataset also provides quantified measures for about 28 contextual features such as distinctness, time difference from the start/end of an event, time elapsed between image acquisition and event memory evaluation, place, activity, presence of a human etc.

For each lifelogger, we split the data into train:val:test in 6:3:1 ratios. This was carried out for all the lifeloggers. During FL training iterations, only the train set from each client is utilized. After training is over, the personalised model for each individual client is employed to infer the memorability of images in the client's own test set images. The overall F1-score is calculated by considering the inferred and ground-truth memorability on all the test set images from all clients.



(a) Lg_16 (b) Lg_56

Figure 5: Difference in histograms of memory score distribution in individual lifeloggers (Lg)

Learning model: We adopted a transfer learning based approach for predicting the event memory score from a given lifelog image. The lifelog image is first passed through a frozen ResNet50 backbone to extract image features. These features are then processed through two fully connected layers with ReLU activation to predict the memory score. The first linear layer maps the 2048-D ResNet features to 512 and the next linear layer maps it to 10 output classes corresponding to the memorability scores. We use dropout between the first and second linear layers with probability of 0.5 and use batch normalization layer to reduce feature covariance shift after the first linear layer.

4.2 Comparison of event memorability prediction performance

In Table 1, we present the performance of FedMEM and compare against other state-of-the-art Federated and Non-Federated Learning approaches. We note that FedMEM+MSC outperforms all other approaches with a F1-score of 34.47%.

Comparison to non-FL approaches: FedMEM+MSC/MSDC performs better than the centralized model baseline reported in [Xu+21] in terms of F1 score by $> 13\%$. For a fair comparison, we compare with the image-only baseline provided in [Xu+21]. FedMEM+MSC also performs better than siloed models which demonstrates that a carefully designed clustered federated learning approach can benefit from model sharing across lifeloggers. Thus FedMEM+MSC realizes our primary objective of simultaneously improving privacy and model performance by avoiding raw-image sharing as well as enabling collaborative learning.

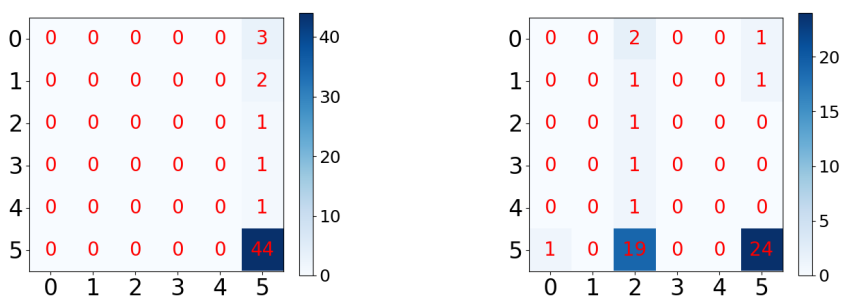
Comparision to FL approaches: We observe from Table 1 that the non-clustered approaches such as FedAvg [McM+17], FedProx [Li+20] and pFedMe [TTN20] obtain $9 - 12\%$ lower F1-score in comparison to FedMEM+MSDC. Other clustered federated approaches such as h-SGD [Liu+20] and DemLearn [Ngu+22] also outperform non-clustered approaches. This underscores the importance of clustering when dealing with heterogeneous data. Among the clustered FL models, our MSDC based apriori clustering obtains 8% higher F1-score compared to h-SGD [Liu+20] which also performs apriori clustering. Moving on to dynamic clustering based FL methods, FeSEM [Lon+23] does not compute a global model similar to FedMEM+MSC but our proposed similarity computation (Equation (1)) allows MSC to perform better by 23% on F1-score. When compared to DemLearn [Ngu+22], which is the state-of-the-art personalization model with dynamic clustering, FedMEM+MSC personalized models are able to achieve 5.5% higher F1 score. This demonstrates that our proposed penalty term enforcing similarity between personalized and cluster models creates better clusters that improves model performance.

Comparision between MSC and MSDC: FedMEM+MSC shows better performance than FedMEM+MSDC. We attribute the improvement to MSC's dynamic clustering and the inclusion of a global model, which facilitates knowledge transfer. In contrast, MSDC uses fixed clusters without a global model which can limit the transfer of knowledge between clusters.

Qualitative Analysis. Our motivation behind developing personalized model is to enhance the performance of each lifelogger's model. Therefore, in Figure 6, we show the impact of personalization on selected lifelogger's models using confusion matrices. The global model (FedAvg [McM+17]) tends to misclassify the images to higher memorability scores. The global model learn this pattern if a majority of lifeloggers memory score distributions (e.g. Figure 5b) are biased towards higher scores. However, it can affect the performance of lifeloggers when they have a different memorability score distribution as seen in Figure 6b.

Table 1: Comparison with different FL algorithms

Method					F1-Score
Non Federated	Centralized Model [Xu+21] (Image-only baseline)				15.00
	Siloed models per lifelogger				29.93
Federated	Non Clustered		Non Personalized	FedAvg [McM+17]	16.49
				FedProx [Li+20]	16.31
			Personalized	pFedMe [TTN20]	19.80
	Clustered	Apriori	Non Personalized	h-SGD [Liu+20]	20.39
		Dynamic		FeSEM [Lon+23]	11.46
		Dynamic	Personalized	DemLearn [Ngu+22]	28.86
		Apriori		FedMEM+ MSDC [Ours]	28.69
		Dynamic		FedMEM+ MSC [Ours]	**34.47**



(a) Lg 56 (Personalized)

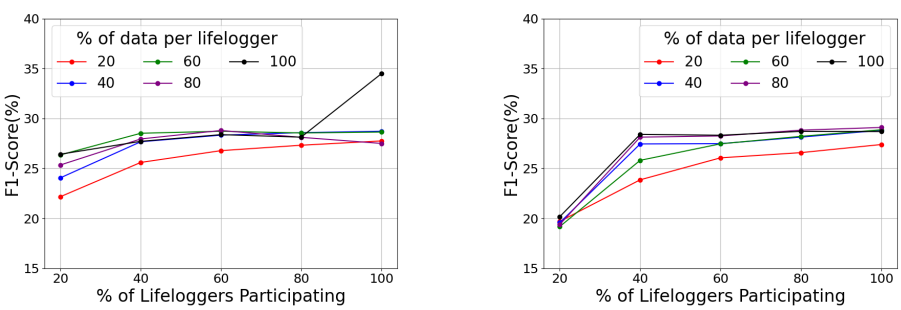
(b) Lg 56 (Global)

Figure 6: Confusion matrix – FedMEM vs. FedAvg [McM+17]

Personalized models capture the score distribution of each lifelogger resulting in better performance as shown in Figure 6a.

4.3 Partial participation of lifelogger and data contribution per lifelogger

In our previous experiments, we have assumed that all lifeloggers simultaneously participate in all the iterations and that they utilize all their available data when training the personalized FL models. However, all the lifeloggers may not be collecting data at the same time and they may not be able to provide their entire collection of lifelog images that are self-rated for event memorability in a single shot. Therefore, it is possible that some lifeloggers are not available when learning the FL model and also that some portion of their data is not available during learning. To investigate such scenarios, we measured the performance of FedMEM under such partial participation conditions as presented in Figure 7. Most importantly, we note that even under a very conservative assumption that only 20% of the lifeloggers are available, and each of them contribute only 20% of their labelled data, both MSC and MSDC are still able to obtain a performance that is better than the centralized model baseline. This shows that FedMEM can significantly lower the self-rating effort of lifeloggers. We also notice that when each lifelogger is contributing between 20% to 80% of the data, we are able to come close to peak performance with just 60% of lifeloggers participating. These results open-up interesting trade-offs with respect to lifelogger participation, their self-rating load and model accuracy.

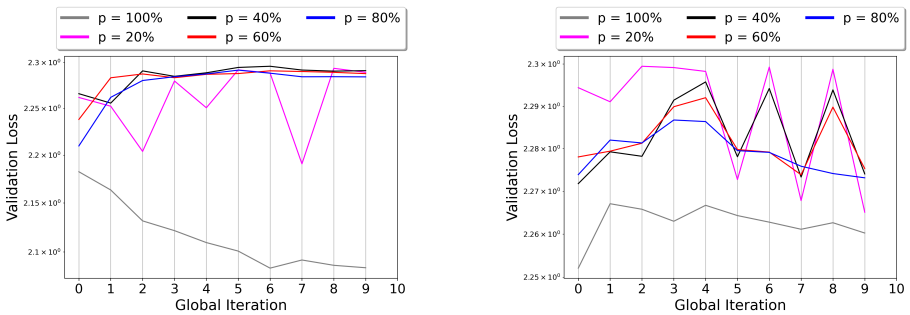


(a) FedMEM + MSC (b) FedMEM + MSDC

Figure 7: Ablation study on the performance of FedMEM with partial participation of lifelogger and data per lifelogger.

Convergence. We also study the convergence with partial lifelogger participation in Figure 8. The convergence study investigates whether a lifelogger needs to wait more to achieve a robust personalized model if less number of lifeloggers are available. FedMEM+MSDC is more robust than FedMEM+MSC after

5 iterations even with 20% due to apriori clusters that utilize the memorability score distribution. MSC requires more lifeloggers to form stable clusters that can benefit the personalized models.

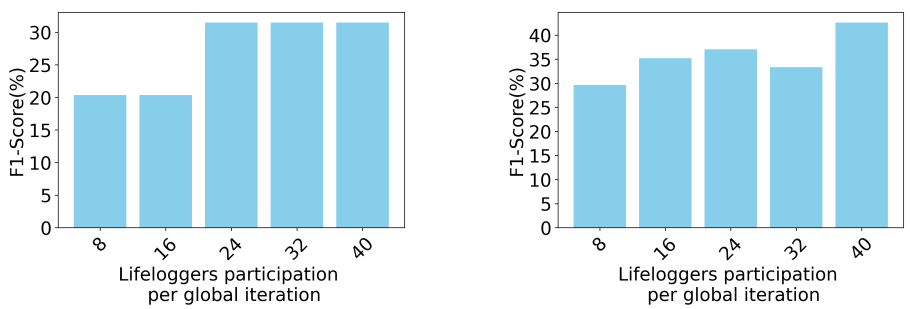


(a) FedMEM + MSC (b) FedMEM + MSDC

Figure 8: Convergence analysis of FedMEM with partial lifelogger participation (p = 20%, 40%, 60%, 80% and 100%)

Qualitative analysis. We also investigate the effect of partial participation on selected lifelogger's personalized models in Figure 9. From Figure 9a and 9b,

it is observed that model sharing with 24 fellow lifeloggers (60% participation) is sufficient to achieve nearly the same peak performance as obtained with all 40 lifeloggers.



(a) Lg 17 (b) Lg 52

Figure 9: Performance of personalized models for selected lifeloggers with partial participation of other lifeloggers (8 – 20%, 16 – 40%, 24 – 60%, 32 – 80%, and 40 – 100%)

4.4 Fusing contextual information

We observed that the performance of every FL model improved after incorporating contextual information as shown in Table 2, in comparison to the image-only results reported in Table 1. Moreover, FedMEM+MSC is better than the selected state-of-the-art FL methods as well as baseline non-federated learning approaches. FedMEM+MSDC gives better performance than the FedProx, and nearly equivalent performance with FedAvg, h-SGD.

Table 2: Comparison after fusing contextual visual semantic information

Method	**F1**	**Method**	**F1**
Centralized [Xu+21]	36.8	Siloed	34.94
FedAvg [McM+17]	35.26	FedProx [Li+20]	34.41
pFedMe [TTN20]	36.22	h-SGD [Liu+20]	34.89
FeSEM [Lon+23]	35.26	DemLearn [Ngu+22]	36.73
FedMEM + MSDC [Ours]	34.80	**FedMEM + MSC** [Ours]	**37.43**

5 Conclusion

We introduced FedMEM, a clustered personalized federated learning approach for predicting event memorability of visual lifelog images. FedMEM enhanced privacy by avoiding image sharing across lifeloggers while simultaneously reaping the benefits of collaborative learning via model sharing. We adopted a clustering mechanism in FedMEM to avoid the client drift problems. We explored two clustering strategies: apriori or memory score distribution-based clustering (MSDC) and dynamic or model similarity-based clustering (MSC). We also developed criteria for measuring similarity among the personalized models of lifeloggers, their cluster models, and the global model. FedMEM was implemented using convolutional neural networks and was further enhanced through the late fusion of contextual information. It was evaluated across diverse federated learning algorithms and in both siloed and centralized configurations. FedMEM+MSC consistently outperformed state-of-the-art methods for each test scenario.

Acknowledgments

This work was partially supported by the Wallenberg AI, Autonomous Systems and Software Program (WASP) funded by the Knut and Alice Wallenberg Foundation. The computations were enabled by the Berzelius resource provided by the Knut and Alice Wallenberg Foundation at the National Supercomputer Centre. This work is also partially funded by the A*STAR CDA project - 202D800021.

References

[ABF22] Safaa Azzakhnini, Olfa Ben-Ahmed, and Christine Fernandez- Maloigne. "Video Memorability Prediction using Deep Features and Loss-based Memorability Distribution Estimation". In: *MediaEval Benchmarking Initiative for Multimedia Evaluation*. 2022.

[BFA20] Christopher Briggs, Zhong Fan, and Peter Andras. "Federated learning with hierarchical clustering of local updates to improve training on non-IID data". In: *Proceedings of International Joint Conference on Neural Networks (IJCNN)*. IEEE. 2020, pp. 1–9.

[BIO13] Wilma A Bainbridge, Phillip Isola, and Aude Oliva. "The intrinsic memorability of face photographs." In: *Journal of Experimental Psychology: General* 142.4 (2013), p. 1323.

[BR18] Wilma A Bainbridge and Jesse Rissman. "Dissociating neural markers of stimulus memorability and subjective recognition during episodic retrieval". In: *Scientific Reports* 8.1 (2018), p. 8679.

[BYB22] Sourasekhar Banerjee, Alp Yurtsever, and Monowar Bhuyan. "Personalized multi-tier federated learning". In: *Workshop on Federated Learning: Recent Advances and New Challenges (in Conjunction with NeurIPS 2022)*. 2022.

[Coh+19] Romain Cohendet et al. "VideoMem: Constructing, analyzing, predicting short-term and long-term video memorability". In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 2019, pp. 2531–2540.

[Doh+12] Aiden R Doherty et al. "Experiences of aiding autobiographical memory using the SenseCam". In: *Human–Computer Interaction* 27.1-2 (2012), pp. 151–174.

[Dua+21] Moming Duan et al. "Fedgroup: Efficient federated learning via decomposed similarity-based clustering". In: *IEEE Intl Conf on Parallel & Distributed Processing with Applications, Big Data & Cloud Computing, Sustainable Computing & Communications, Social Computing & Networking*. IEEE. 2021, pp. 228–237.

[Faj+18] Jiri Fajtl et al. "Amnet: Memorability estimation with attention". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. IEEE, 2018, pp. 6363–6372.

[FCJ17] Md Sadek Ferdous, Soumyadeb Chowdhury, and Joemon M Jose. "Analysing privacy in visual lifelogging". In: *Pervasive and Mobile Computing* 40 (2017), pp. 430–449.

[Gho+20] Avishek Ghosh et al. "An efficient framework for clustered federated learning". In: *Advances in Neural Information Processing Systems* 33 (2020), pp. 19586–19597.

[GLT18] Ana Garcia del Molino, Joo-Hwee Lim, and Ah-Hwee Tan. "Predicting visual context for unsupervised event segmentation in continuous photo-streams". In: *Proceedings of the 26th ACM international conference on Multimedia*. 2018, pp. 10–17.

[HLW16] Morgan Harvey, Marc Langheinrich, and Geoff Ward. "Remembering through lifelogging: A survey of human memory augmentation". In: *Pervasive and Mobile Computing* 27 (2016), pp. 14–26.

[Hua+19] Li Huang et al. "Patient clustering improves efficiency of federated machine learning to predict mortality and hospital stay time using distributed electronic medical records". In: *Journal of biomedical informatics* 99 (2019), p. 103291.

[Iso+11] Phillip Isola et al. "Understanding the intrinsic memorability of images". In: *Advances in neural information processing systems* 24 (2011).

[Iso+13] Phillip Isola et al. "What makes a photograph memorable?" In: *IEEE transactions on pattern analysis and machine intelligence* 36.7 (2013), pp. 1469–1482.

[Jin+16] Peiguang Jing et al. "Predicting image memorability through adaptive transfer learning from external sources". In: *IEEE Transactions on Multimedia* 19.5 (2016), pp. 1050–1062.

[KAC20] Griffin E Koch, Essang Akpan, and Marc N Coutanche. "Image memorability is predicted by discriminability and similarity in different stages of a convolutional neural network". In: *Learning & Memory* 27.12 (2020), pp. 503–509.

[Kar+20] Sai Praneeth Karimireddy et al. "Scaffold: Stochastic controlled averaging for federated learning". In: *International conference on machine learning*. PMLR. 2020, pp. 5132–5143.

[Kho+15] Aditya Khosla et al. "Understanding and predicting image memorability at a large scale". In: *Proceedings of the IEEE international conference on computer vision*. IEEE, 2015, pp. 2390–2398.

[LD07] Matthew L Lee and Anind K Dey. "Providing good memory cues for people with episodic memory impairment". In: *Proceedings of the 9th international ACM SIGACCESS conference on Computers and accessibility*. ACM, 2007, pp. 131–138.

[Li+20] Tian Li et al. "Federated optimization in heterogeneous networks". In: *Proceedings of Machine learning and systems* 2 (2020), pp. 429–450.

[Liu+20] Lumin Liu et al. "Client-edge-cloud hierarchical federated learning". In: *Proceedings of IEEE international conference on communications (ICC)*. IEEE. 2020, pp. 1–6.

[Lon+23] Guodong Long et al. “Multi-center federated learning: clients clustering for better personalization”. In: *World Wide Web* 26.1 (2023), pp. 481–500.

[Lu+20] Jiaxin Lu et al. “Understanding and predicting the memorability of outdoor natural scenes”. In: *IEEE Transactions on Image Processing* 29 (2020), pp. 4927–4941.

[Man+20] Yishay Mansour et al. “Three approaches for personalization with applications to federated learning”. In: *arXiv:2002.10619* (2020).

[McM+17] Brendan McMahan et al. “Communication-efficient learning of deep networks from decentralized data”. In: *Artificial intelligence and statistics*. PMLR, 2017, pp. 1273–1282.

[Mil+11] F Milton et al. “The neural correlates of everyday recognition memory”. In: *Brain and Cognition* 76.3 (2011), pp. 369–381.

[Ngu+22] Minh NH Nguyen et al. “Self-organizing democratized learning: Toward large-scale distributed learning systems”. In: *IEEE Transactions on Neural Networks and Learning Systems* (2022).

[Ris+16] Jesse Rissman et al. “Decoding fMRI signatures of real-world autobiographical memory retrieval”. In: *Journal of cognitive neuroscience* 28.4 (2016), pp. 604–620.

[Sil+18] AR Silva et al. “A critical review of the effects of wearable cameras on memory”. In: *Neuropsychological Rehabilitation* 28.1 (2018), pp. 117–141.

[SM00] Jianbo Shi and Jitendra Malik. “Normalized cuts and image segmentation”. In: *IEEE Transactions on pattern analysis and machine intelligence* 22.8 (2000), pp. 888–905.

[SMS20] Felix Sattler, Klaus-Robert Müller, and Wojciech Samek. “Clustered federated learning: Model-agnostic distributed multitask optimization under privacy constraints”. In: *IEEE transactions on neural networks and learning systems* 32.8 (2020), pp. 3710–3722.

[Squ+18] Hammad Squalli-Houssaini et al. “Deep learning for predicting image memorability”. In: *Proceedings of IEEE international conference on acoustics, speech and signal processing (ICASSP)*. IEEE, 2018, pp. 2371–2375.

[TTN20] Canh T Dinh, Nguyen Tran, and Josh Nguyen. “Personalized federated learning with moreau envelopes”. In: *Advances in Neural Information Processing Systems* 33 (2020), pp. 21394–21405.

[Xu+18] Qianli Xu et al. “Personalized serious games for cognitive intervention with lifelog visual analytics”. In: *Proceedings of the 26th ACM international conference on Multimedia*. 2018, pp. 328–336.

[Xu+21] Qianli Xu et al. “Predicting Event Memorability from Contextual Visual Semantics”. In: *Advances in Neural Information Processing Systems* 34 (2021), pp. 22431–22442.

Appendix A Additional Results and Analysis

In this supplementary material, we provide additional details and results that were not included in the main paper, but we evaluated during the study. This appendix contains the following items,

- Detailed description of the FedMEM algorithm. (Algorithm 3 and Algorithm 4).
- Additional details about the experimental setup.
- Additional details about the distribution of lifelog data across clients (Figure 10).
- Description and the analysis of the effect of λ_1 and λ_2 when lifeloggers have partial participation (Figure 12).
- An ablation study on the impact of clusters on FedMEM+MSC (Figure 13).
- Additional details on the confusion matrix based qualitative analysis of lifelogger's personalized models and global model (FedAvg [McM+17]). (Figure 11)
- Additional metrics for the performance analysis of FedMEM with other federated learning approaches (Table 3)

Appendix A.1 Description of FedMEM algorithm

We have explored two clustering strategies: (1) Memorability Score Distribution-based Clustering (MSDC) for lifeloggers, and (2) Model Similarity-based Clustering (MSC) for lifeloggers. By integrating these clustering strategies with personalized federated learning, we have developed a clustered personalized federated learning algorithm, FedMEM, for event memorability prediction.

Appendix A.1.1 FedMEM+MSDC

In this Algorithm 3, all available life-loggers first send their memory score distribution (CD_i) to the server for clustering. The server performs KL divergence to determine the memory score similarity between pairs of clients, producing a similarity matrix $KL[,]$. Hierarchical clustering is then performed on this similarity matrix KL, dividing the clients into C_k clusters. Each client initializes their model from their respective cluster, such that if life-logger i belongs to cluster C, the initial model for life-logger i would be W_C. For each cluster, the $Personalized\_Local\_Update$ operation is performed, followed by the aggregation of local models within each cluster. FedMEM+MSDC provides a clustered federated learning approach in which lifeloggers remain in their designated clusters without migrating between them, ensuring that clusters are fully isolated from one another with no cross-cluster movements. Therefore, no cross-cluster knowledge sharing occurs.

Algorithm 3 FedMEM+MSDC

Initialize: $\omega_0^0 = \omega_1^0 = \ldots = \omega_C^0 = \omega^0$, T, L
Server receives distribution of the event memory scores $\{CD_0, CD_1, \ldots, CD_{\mathcal{N}-1}\}$ form $\mathcal{N}$ lifeloggers.
C ← MSDC(CD_0, CD_2, ..., $CD_{\mathcal{N}-1}$) ▷ Call Algorithm 1
for $t \leftarrow 0$ **to** T **do**
    $M \leftarrow Select\_clients(\mathcal{N})$
    Server sends the ω_c^t to the selected lifeloggers ▷ ω^t to the new lifelogger
    for M lifeloggers in parallel **do** ▷ Lifelogger assigned to any of the k clusters. c ∈ C
        θ_i = Personalized_Local_Update(ω_c^t)
    end for
    All lifeloggers send θ_i^t to their respective clusters.
    for $Lg_i \in M$ **do** in all C clusters perform the cluster update
        $\omega_c^t \leftarrow \omega_c^t + r_i\theta_i^t$ ▷ Equation (8)
    end for
end for
procedure Personalized_Local_Update($\bar{\omega}$)
    for $l \leftarrow 0$ **to** L **do**
        $\theta_i^{t,l+1} = \theta_i^{t,l} - \alpha_i \nabla f_i(\theta_i^{t,l}) - \alpha_i \eta(\theta_i^{t,l} - \omega_c^t)$. ▷ Equation (6)
    end for
end procedure

Appendix A.1.2 FedMEM+MSC

In Algorithm 4, initially, all available life-loggers do not have any information about the clusters. They start by initializing their local models with the same global model. In the initial round (t=0), the server sends the global model ω^0 to the selected life-loggers. In subsequent rounds, if a life-logger has been assigned to a cluster, the server sends the cluster model to that life-logger. If a new life-logger joins in a later round and does not belong to any cluster, their model is initialized with the global model. Similar to Algorithm 3, clients perform the *Personalized_local_update* operation to train their local models. Each life-logger then sends their trained local model to the server, where a similarity matrix is created using Algorithm 2. Spectral clustering is then performed to identify clusters. After cluster assignments are made, a cluster update is performed, and a new global model is created. This process continues until the personalized models of all life-loggers converge.

Appendix A.1.3 Comparison between FedMEM+MSDC and FedMEM+MSC

The FedMEM+MSDC approach forms clusters of lifeloggers based on their memory score distribution. These clusters remain static, meaning lifeloggers do not switch clusters during training. Unlike FedMEM+MSC, this method does not create a single global model. Instead, each cluster model acts as the global model for its respective cluster. There is no knowledge sharing between clusters since lifeloggers do not move across clusters.

On the other hand, FedMEM+MSC forms clusters based on model similarity among lifeloggers, the global model, and cluster models. In this approach, lifeloggers can move between clusters, enabling cross-cluster knowledge sharing.

Algorithm 4 FEDMEM+MSC

Initialize: $\omega_0^0 = \omega_1^0 = \ldots = \omega_C^0 = \omega^0$, T, L
for $t \leftarrow 0$ **to** T **do**
  $M \leftarrow Select\_clients(\mathcal{N})$
  Server sends the ω_c^t or ω^t to the selected lifeloggers ▷ ω^t to the new lifelogger
  for M lifeloggers in parallel **do**
    if $Lg_i \notin C$ **then** ▷ Lifelogger not assigned to any cluster
      θ_i = Personalized_Local_Update(ω^t)
    else ▷ Lifelogger assigned to any of the k clusters. c ∈ C
      θ_i = Personalized_Local_Update(ω_c^t)
    end if
  end for
  All lifeloggers send θ_i^t to the server.
  $C \leftarrow MSC(\theta_1^t, \theta_2^t, \ldots, \theta_M^t)$ ▷ Call Algorithm 2
  for $Lg_i \in M$ **do** in all C clusters perform the cluster update
    $\omega_c^t \leftarrow \omega_c^t + r_i \theta_i^t$ ▷ Equation (8)
  end for
  for all $c \in C$ in parallel **do** ▷ Only for MSC
    $\omega^t = \omega^t + r_c \omega_c^t$ ▷ Equation (10)
  end for
end for
procedure PERSONALIZED_LOCAL_UPDATE($\bar{\omega}$)
  for $l \leftarrow 0$ **to** L **do**
    $\theta_i^{t,l+1} = \theta_i^{t,l} - \alpha_i \nabla f_i(\theta_i^{t,l}) - \alpha_i \eta(\theta_i^{t,l} - \omega_c^t)$. ▷ Equation (6)
  end for
end procedure

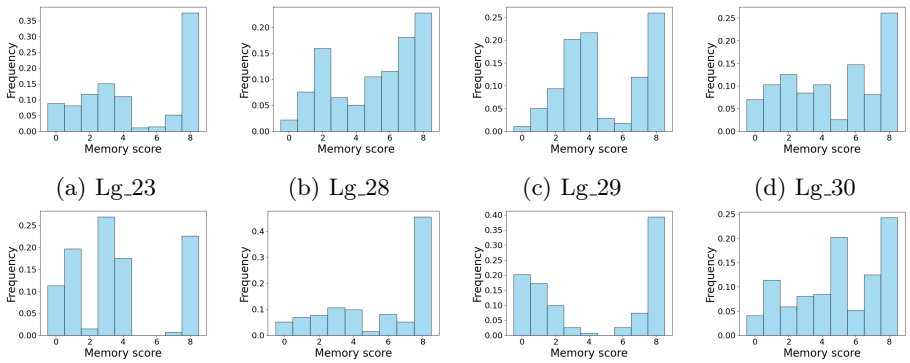
Appendix A.2 Experimental setup

For each lifelogger, we split the data into train:val:test in 6:3:1 ratios. This was carried out for all the lifeloggers. During FL training iterations, only the train set from each lifelogger is utilized. After training is over, the personalised model for each individual lifelogger is employed to infer the memorability of images in the lifelogger's own test set images. The overall F1-score is calculated by considering the inferred and ground-truth memorability on all the test set images from all lifeloggers. Cross-cluster/Cross-user testing was not carried out as our personalized approach produced individual models for every lifelogger.

We utilized the same frozen ResNet50 as the baseline CEMNET [Xu+21]. Because we want to demonstrate the advantage of our personalized federated learning approach over the centralized CEMNET model.

Appendix A.3 Distribution of lifeloggers data

In this supplementary copy we provide the data distributions of some more lifeloggers Lg_{23} (Figure 10a), Lg_{28} (Figure 10b) to Lg_{34} (Figure 10h). We observed the data distribution across lifeloggers, and we noted that some lifeloggers do not have any data for certain memory scores (such as, for Lg_{31}, none of their images had memory scores of 5 and 6). This showed us that the memorability scores provided by the lifeloggers do not follow the same distribution. Therefore, we believe that there is case of non-IID (not independent and identical) distribution of data across lifeloggers.



(a) Lg_23 (b) Lg_28 (c) Lg_29 (d) Lg_30

(e) Lg_31 (f) Lg_32 (g) Lg_33 (h) Lg_34

Figure 10: Difference in histograms of memory score distribution in individual lifeloggers (Lg)

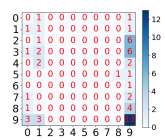
Appendix A.4 Qualitative analysis of lifelogger's personalized models and global model (FedAvg)

We also provide a confusion matrix-based analysis for each qualitative analysis of lifeloggers in Figure 11 . The color intensity represents the classifier's predictions. Darker colors usually indicate higher numbers. This means that areas in the matrix where the model made more predictions (correct or incorrect) will be highlighted with a darker shade. Lighter colors typically represent lower numbers. These areas indicate fewer predictions for those class combinations. The y-axis (rows) represents the actual classes, and the x-axis (columns) represents the predicted classes. We observed from the Figure 11 that the personalized model produces more true positive predictions, whereas the global model makes more false positive predictions. This further supports our claim that the personalized model is more effective than the global model for all the lifeloggers.

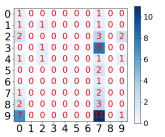
Appendix A.5 Ablation of the value of λ_1 and λ_2

We trained both FedMEM+MSC and MSDC with 5 clusters. We defined hyperparameters λ_1 and λ_2 in the range [0,1], ensuring $\lambda_1 + \lambda_2 \leq 1$. These parameters regularize the similarity measure. Setting both to 0 focuses on client similarity. Increasing λ_1 emphasized the global model while increasing λ_2 emphasized the clustered model. For equal consideration of both cluster and global models, λ_1 and λ_2 are set equally. In the experiments in the submitted paper we prioritized lifelogger's similarity and set λ_1 and λ_2 to 0.25.

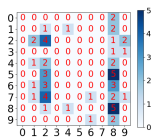
In Figure 12, we performed a comparative analysis of λ_1 and λ_2 to examine the impact of global and cluster models on clustering, which is reflected in the effectiveness of FedMEM. The model was trained for 30 global rounds, with



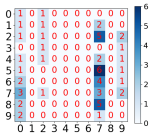
(a) Lg 23 (Personalized)



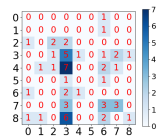
(b) Lg 23 (Global)



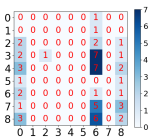
(c) Lg 28 (Personalized)



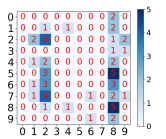
(d) Lg 28 (Global)



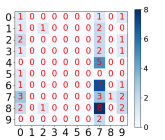
(e) Lg 29 (Personalized)



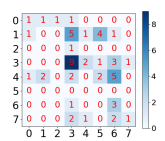
(f) Lg 29 (Global)



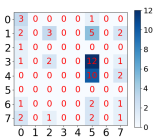
(g) Lg 30 (Personalized)



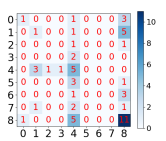
(h) Lg 30 (Global)



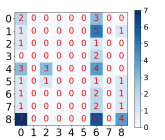
(i) Lg 31 (Personalized)



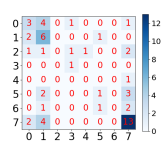
(j) Lg 31 (Global)



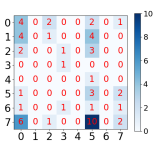
(k) Lg 32 (Personalized)



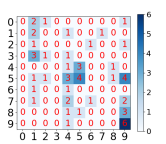
(l) Lg 32 (Global)



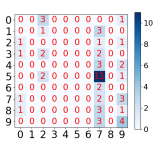
(m) Lg 33 (Personalized)



(n) Lg 33 (Global)



(o) Lg 34 (Personalized)



(p) Lg 34 (Global)

Figure 11: Personalized model (FEDMEM) compared to global model (FedAvg [McM+17]) for selected lifeloggers (Lg). Global model favors higher memorability scores but FEDMEM is able to capture the characteristics of each lifelogger's memorability score distribution resulting in improved F1 score.

50% of the lifeloggers participating in each round. We found that increasing the value of λ_1 improves the performance of FedMEM. Furthermore, when λ_1 and λ_2 are in the range of $(0.25, 0.75)$, the performance of FedMEM remains relatively consistent.

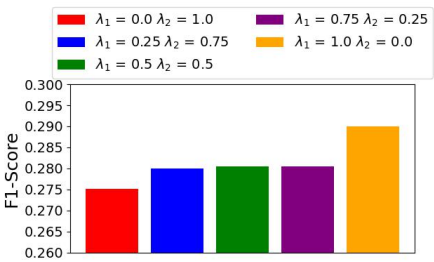


Figure 12: Ablation on value of λ_1 and λ_2

Appendix A.6 Ablation study on the impact of clusters on FedMEM+MSC

In Figure 13, we performed the experiment across 40 life loggers, but 50% of them were available at each global round. We train for 30 global rounds. The values of the hyperparameters λ_1 and λ_2 are 0.25. We observed if we increase the number of clusters, the performance of FedMEM decreases. But the decrease is not significant.

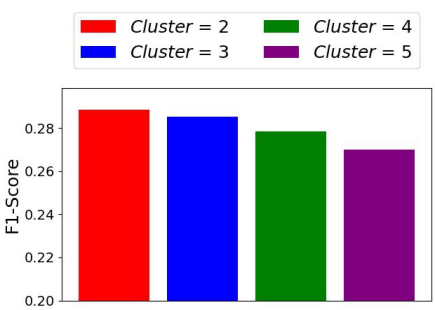


Figure 13: Ablation on the number of clusters

Table 3: Comparison of FedMEM with different FL algorithms

Method					Precision	Recall	F1-Score (weighted)	Mean Absolute Error
Non Federated	Centralized Model [Xu+21] (Image-only baseline)				17.10	17.90	-	3.03
	Siloed models per lifelogger				21.10	29.9	24.74	2.30
Federated	Non Clustered		Non Personalized	FedAvg [McM+17]	12.27	16.49	14.07	3.69
				FedProx [Li+20]	12.22	16.31	13.97	3.78
			Personalized	pFedMe [TTN20]	15.60	19.80	17.45	3.39
	Clustered	Apriori	Non Personalized	h-SGD [Liu+20]	07.00	20.39	10.42	3.96
		Dynamic		FeSEM [Lon+23]	03.37	11.46	05.21	4.06
		Dynamic	Personalized	DemLearn [Ngu+22]	**21.54**	28.86	24.67	2.59
		Apriori		**FedMEM+ MSDC [Ours]**	21.30	28.69	24.45	2.68
		Dynamic		**FedMEM+ MSC [Ours]**	19.86	**34.47**	**25.20**	**2.28**

Appendix A.7 Performance analysis with the state-of-the-art

In Table 3, we present a comparative analysis of the performance of FedMEM with different state-of-the-art Federated Learning algorithms. In the main article, we had given only one performance metric, which is the F1-score. Therefore, in this supplementary, we provide additional metrics here for comparison such as the Precision, Recall, weighted-F1 score, and the Mean Absolute Error (MAE). Since the memory score is an ordinal number, we obtain the absolute value of the error between the predicted and ground-truth memorability scores. Then we calculate the average value of this absolute error during inference time over the images in the testset, which is presented as MAE. From Table 3 we observed FedMEM+MSC performance is better than the other FL algorithms.

Paper VI

The Case for Federated Learning in Developing Personalized Image Privacy Advisor

Sourasekhar Banerjee, Vengateswaran Subramaniam, Debaditya Roy, Vigneshwaran Subbaraju, and Monowar Bhuyan

Submitted for publication, 2024.

The Case for Federated Learning in Developing Personalized Image Privacy Advisor*

Sourasekhar Banerjee[§†], Vengateswaran Subramaniam[†], Debaditya Roy[†], Vigneshwaran Subbaraju[†], and Monowar Bhuyan[§]

§ *Department of Computing Science, Umeå University, Umeå, Sweden*

† *A*STAR Institute of High Performance Computing (IHPC), Singapore*

sourasb@cs.umu.se, Vengateswaran_Subramaniam@ihpc.a-star.edu.sg, Roy_Debaditya@ihpc.a-star.edu.sg, Vigneshwaran_Subbaraju@ihpc.a-star.edu.sg, monowar@cs.umu.se

Abstract:

Sharing privacy-sensitive information poses a significant risk when private information is carelessly exposed to individuals or groups through various media. With minimal annotation and user preference, users can decide whether to share or keep an image based on its privacy score. This approach helps reduce the likelihood of privacy breaches and makes it easier to share images with friends or on social media platforms. Federated Learning (FL) allows collaborative learning of models without sharing personal data, which is crucial for sharing privacy-sensitive information. However, there are several challenges associated with learning from private data. This paper proposes two FL algorithms, namely Dynamic-Clustered-FedDC and Apriori-Clustered-FedDC, to address key problems, such as data scarcity, data heterogeneity, and model complexity in order to ensure personalized image privacy. Both algorithms train personalized models for each annotator using clustered federated learning to address data heterogeneity. Additionally, both algorithms utilize daisy chaining-based knowledge sharing between annotators to mitigate data scarcity issues during training. In addition to these two algorithms, we propose the PIONet model, which is 20× lighter compared to baseline models and retains equivalent performance. PIONet, along with Dynamic-Clustered-FedDC and Apriori-Clustered-FedDC, outperformed the state-of-the-art federated learning algorithms and the baseline.

Key words: Federated learning, Statistical heterogeneity, Data scarcity, Personalized federated learning, Daisy chaining

*This work was completed as part of an internship in the Social and Cognitive Computing department at the A*STAR Institute of High-Performance Computing in Singapore.

1 Introduction

Privacy sensitive information are commonly found in images and users sharing such personal images across various media are at risk of exposing sensitive information inadvertently. An automated *image privacy advisor* application can mitigate this risk by informing the user about the presence of privacy sensitive information in a given image (as shown in Figure 1), thereby enabling them to make informed decisions regarding sharing or obfuscation. To support such

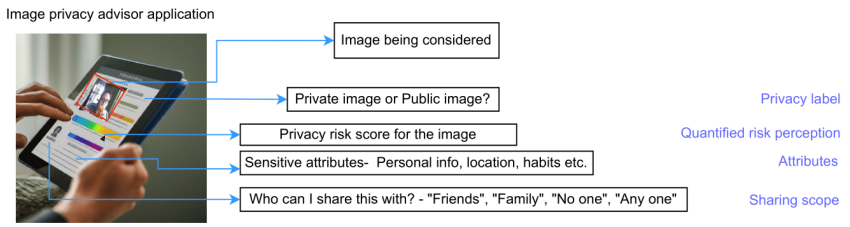
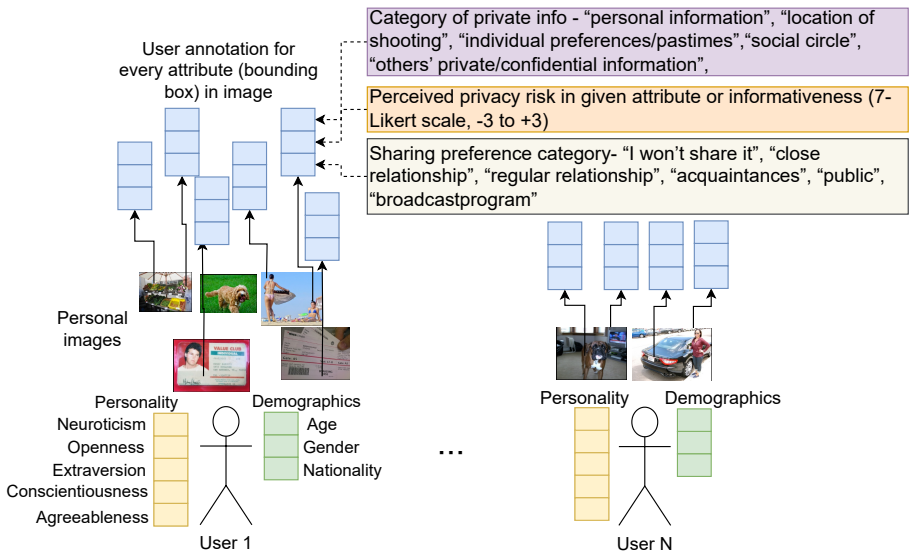


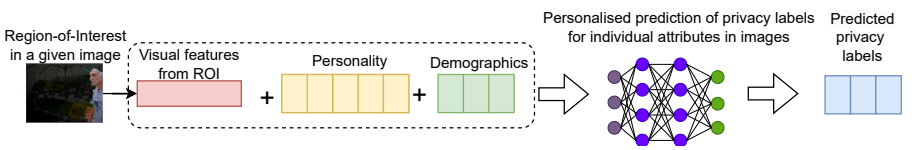
Figure 1: An hypothetical image privacy advisor application, that can inform the image owner about various aspects of privacy in a given image to enable sharing and obfuscation related decisions.

an application, deep neural network models trained on public image datasets have been developed. These models (a) detect the presence of some pre-defined categories of privacy sensitive attributes such as name, face, licence plate, location etc. [OSF17; Yu+16; Gur+19; CKS21], (b) classify an image as private or public [TC16; Zho+17; TC19], and/or (c) provide a score that reflects the privacy risk of an image [OSF17; CKS21]. However, privacy sensitivity is a highly subjective phenomenon where preferences and perceptions of privacy risks can vary from person to person, culture to culture and it could also be dependent on the privacy laws that are in force at different locations [JJS08; WNC11; She02; Hen+12]. For example, certain attributes in an image (e.g. a pet animal) may be considered to be highly private information by some users, while other may not. Therefore, personalised models that are trained on fine-grained annotations of a user's own perception of privacy risk for specific attributes or regions-of-interest in their images (as illustrated in Figure 2) are needed to accurately reflect individual preferences.

Recent works such as DIPA [Xu+23] and DIPA2 [Xu+24] have sought to address personalised, attribute specific privacy preference modeling by obtaining the annotations of privacy sensitivity scores for specific regions/attributes in images. They investigate the effect of personality-based and cross-cultural variations in privacy preferences when developing models for image privacy in DIPA2 [Xu+24]. Individuals from Japan and the UK provided their privacy annotations for the same set of images and they found location-specific differences. Furthermore, they obtained the personality information of the annota-



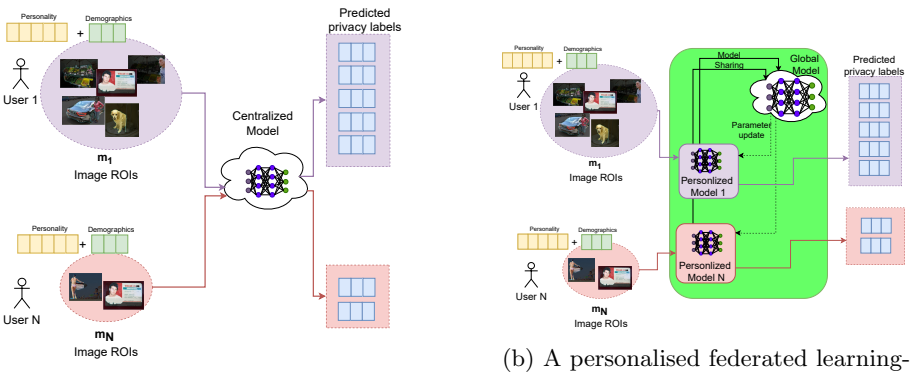
(a) Gathering attribute level annotation of privacy perception from individual users from their personal images



(b) Personality and attribute specific image privacy advisor

Figure 2: Personalised model that can accommodate user specific variations in privacy preferences for individual attributes. Each user annotates some of their personal images, demarcating sensitive attributes or regions-of-interest in them and providing separate privacy labels for each region/attribute. Further, each user provides their demographic information and personality scores.

tors via a personality questionnaire and found that traits such as '*openness*' and '*neuroticism*' influence the sensitivity score provided by users. Using this data, they develop their computational model, where the annotators' personality information, demographics, location etc. was combined with the raw image features to predict the sensitivity of a given region-of-interest.



(a) Centralised approach

(b) A personalised federated learning-based approach

Figure 3: Comparison of centralised and federated approaches. In the centralised approach, all the users upload their annotated images, personality information, and demographics to enable model development. The resultant model accommodates individual variations in privacy preferences by using personality and demographic information. In the federated approach, the user's data stays within the client to enable local, highly personalised model development. Only the model parameters are shared with a global aggregator that enables collaborative learning from other users' data.

A significant drawback of the centralized privacy models developed in [Xu+23] [Xu+24] is that they assume that the entire set of image annotations as well as the user's personality information etc., are provided by all the users to a server which aggregates all such annotations to build the model (see Figure 3a). However, this may not be feasible if the users are not willing to share their private images and personality information for the purpose of model development. Furthermore, the centralised approach suffers from conflicting privacy labels provided by annotators [Suc+17] from different cultures and personality types. Therefore, each user should have a privacy model developed using their own data but they may not be able to annotate a sufficiently large dataset on their own to train a model with reliable results. *Hence, users should collaborate with other users on model development without actually sharing the private raw images.* In this paper, we propose to develop such a collaborative approach model development via Federated Learning (FL) [McM+17]. Each user develops their own models with their personal images annotated with their own perception of privacy sensitivity of specific regions or objects found in their image, and they share these models under the FL framework with collaborative

users as shown in Figure 3b.

Despite FL being a suitable collaborative approach to building personalized image privacy models, the following challenges exist in designing an FL-based solution:

1. **Data scarcity:** Individual users may contribute varying amounts of data samples depending on their convenience. Based on the amount of data each user provides, we classify them into two groups: Resourceful (RF), who have enough data to train individual models, and Resource-Limited (RL), who possess only a minimal amount of data, insufficient for creating private models. As depicted in Figure 4, in DIPA2 [Xu+24] dataset, only a small number of users (fewer than 5) have annotated more than 30 samples. A total of 240 users have provided annotations for only 3-10 samples each. This imbalance leads to a data scarcity issue when training individual models at the users' end, making it challenging to effectively train a local model due to the limited amount of data available. This data scarcity problem motivated us to incorporate daisy chaining in federated learning.
2. **Data heterogeneity:** Subjective phenomena like privacy sensitivity involves high amount of variations owing to location and personality of individual users. This results in a non-iid distribution of data and models shared between users may not be consistent.
3. **Model complexity:** Deep learning models are generally complex and involve significant resources for development in a federated learning setup due to the creation of multiple models and incur communication overheads for model sharing and parameter updates. Hence, it is desirable to have a lightweight model that can reduce the overheads.

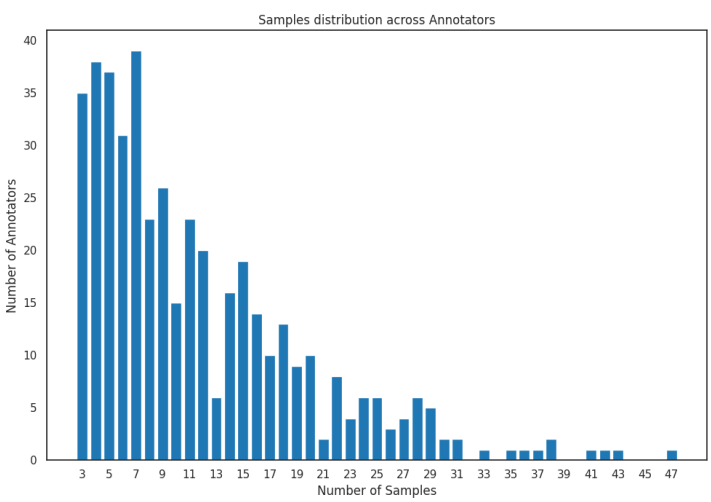


Figure 4: Samples distribution across users

To address these challenges, we introduced two algorithms, Dynamic-Clustered-FedDC and Apriori-Clustered-FedDC, designed to tackle issues of data heterogeneity and scarcity. Both Dynamic-Clustered-FedDC and Apriori-Clustered-FedDC incorporate clustered personalized federated learning to manage data heterogeneity and utilize daisy chaining to mitigate data scarcity. The key difference between the two algorithms lies in their clustering mechanisms and daisy-chaining processes. Dynamic-Clustered-FedDC forms clusters based on a model similarity-based metric. Clusters are re-formed after each global iteration, allowing users to move between clusters flexibly. In contrast, Apriori-Clustered-FedDC, clusters are established based on users' personality scores. The clusters remain fixed throughout the entire learning process. The daisy-chaining methods also differ: Dynamic-Clustered-FedDC implements daisy chaining between resourceful and resource-limited annotators, whereas Apriori-Clustered-FedDC restricts daisy chaining to resource-limited annotators, excluding the resourceful ones. In both algorithms, daisy chaining is performed within clusters, with no cross-cluster daisy chaining taking place. Moreover, along with these two algorithms, we propose a lightweight deep neural network model called Personality-Image attribute-Object Network (PIONet), which is suitable for federated learning. Our contributions are summarized as follows contributions:

- We developed two federated learning algorithms: Dynamic-Clustered-FedDC and Apriori-Clustered-FedDC, that utilize clustering and daisy chaining to overcome the problems of data heterogeneity and data scarcity.
- We developed a light-weight model for predicting the user-specific privacy preferences for a given image. When compared to the DIPA2 baseline, our new model is 20× lighter than SOTA and is able to achieve similar performance.
- We empirically evaluated the performance of PIONet enabled Dynamic-Clustered-FedDC and Apriori-Clustered-FedDC with state-of-the-art federated learning algorithms such as FedAvg[McM+17], FedProx [Li+20b], and FedDC [KFV22], FedMEM on DIPA2 [Xu+24] dataset.

2 Related Work

Detecting privacy-sensitive regions in an image has been a matter of serious concern with the proliferation of online social media and ubiquitous camera hardware found in personal devices. With the advent of deep learning, several computational models have been developed to detect privacy sensitive information in images. A majority of such models in image privacy deal with broadly classifying images into private vs public. [TC16] deep features from CNN-based visual backbones were found to be useful for this purpose, and a combination of user-generated as well as machine-generated tags were considered to be effective. [TC19] developed a multi-modal fusion mechanism by using object,

scene context, and image tags as features to predict the binary privacy label of a given image.

Moving on from broad image-level labels, some studies considered detecting the presence of specific privacy-sensitive attributes and potentially localizing them in a given image. VISPR [OSF17] introduced a large set of about 22000 images and provided annotations of about 68 different privacy-sensitive attributes (objects, text, multi-modal, etc.) in these images. This set of 68 attributes (Eg. name, location, passports, credit card, receipt, handwriting, fingerprints, receipts, etc.) was chosen by the authors by studying the various privacy laws across the world. Using this data, they developed a deep-learning model to detect these attributes in the images. They also conducted a user study to evaluate the risk perception of individual attributes on a scale of (0-5) and provided an aggregated risk score for each image. They found considerable variations in individual risk perceptions of the images across different users. Viz-WizPriv [Gur+19] is another work that considers the privacy of images acquired by blind and visually impaired people. The images considered in their work are acquired from the first-person point-of-view, and they have provided annotations of objects and text-based private information in these photographs. However, all these methods do not account for individual-specific variations in the perception of privacy risks. It has been very well understood from literature in psychology that privacy perceptions are highly subjective. [JJS08] showed that the *Big5* personality scores can be broadly used to infer a person's perception of privacy risk. Other works, such as [Zho+17] - attack the problem of conflicting privacy labels provided by different annotators. They group the users into several clusters and build separate classifiers for each group. However, they consider global binary labels for individual images and do not account for category-specific variations in privacy preferences. To account for such variations, very recently [Xu+23] and [Xu+24] released a dataset with fine-grained annotation labels for individual attributes in images along with scores for personality and demographic information of the annotators. This is valuable given that the perceptions of privacy risks are known to vary with age group, nationality, and individual personality traits [WNC11; She02; Hen+12; JJS08]. Using this unique dataset in [Xu+24], the authors investigated cross-cultural variations in privacy perceptions as users from Japan and the UK provided their perceived scores for the same set of images and attributes.

However, all the above works assume that individual annotators share the image and personality information in a centralized location to build a single model. This is not feasible if the individual users are annotating their own private images and do not wish to share the images and their personality scores. In such situations, a federated learning-based approach could provide a mechanism for collaborative model development without actually sharing the sensitive raw data. Instead, individual clients share the model parameters alone for learning purposes.

Federated learning is a collaborative learning framework where we can generate

a universal model for all collaborators. FedAvg [McM+17] was a first-of-a-kind algorithm for federated learning. FedAvg performs local training on each client and performs aggregation on the server. This method suffers from statistical heterogeneity [Li+20a], resulting in inconsistent performance. Where the global model primarily benefits a select group of clients, causing the global objective to be skewed in favor of these benefited clients. Consequently, the remaining clients do not gain from a single global model. To address this issue, personalized federated learning [TTN20] has been developed, allowing each client to collaborate in FL while ultimately obtaining a model that is personalized to their specific needs. The trade-off between personalization and generalization among clients and servers is a persistent issue. Clustered federated learning [Ngu+22] addresses this by grouping clients with similar characteristics into clusters, based on either model [Lon+23; BFA20; Ngu+22; SMS20] or data similarities [Man+20; Hua+19]. This approach aims for the cluster head model to provide a more generalized model for its cluster while also enhancing the personalized models of individual clients through shared knowledge. Additionally, federated learning faces challenges due to data scarcity, where individual clients lack enough data to effectively train deep neural network models. A daisy chaining method could be beneficial in improving the performance of the overall global model [KFV22; Mat+22; HHB22].

Motivated by these issues, we present Dynamic-Clustered-FedDC and Apriori-Clustered-FedDC, which leverage the strengths of clustering and daisy chaining to effectively learn models despite data heterogeneity and scarcity.

3 Proposed Approach

We initially introduced a lightweight image privacy model suitable for federated learning and later developed a personalized clustered federated learning approach enabled by daisy chaining to train a privacy advisor model.

3.1 Light-weight image-privacy model - PIONet

In [Xu+24], authors inflate the annotator personality information and object category information to the same size as the input image. The resultant input is a 13-channel image that is processed by a parameter-heavy CNN model. Such a model cannot be trained by every client in federated learning setup given the limited annotated data and computational resources. Therefore, we design a new baseline centralized model for privacy prediction called Personality-Image attribute-Object network (PIONet see Figure 5). We introduce the various inputs – entire image, the bounding boxes of the objects, and the annotator personality information and object category at different stages in PIONet. First, we extract the features for the entire image $f_I = 2048 \times 7 \times 7$ using a pretrained ResNet50 model. Next, we extract the object features for the bounding boxes using RoIAlign [He+17] from the extracted image features to

obtain $f_O = 2048 \times 7 \times 7$ features per object bounding box. We consider up to 31 objects. We concatenate the image and object features and pad zeros for the remaining non-present objects to obtain a $32 \times 2048 \times 7 \times 7$ input f_{IO}. We send f_{IO} through the pretrained *adaptive average pool* layer from ResNet50 to obtain a $f'_{IO} = 32 \times 2048$ feature. Note that we have no trainable layers till this point. We train on the pooled features f'_{IO} with an linear layer and *ReLU* activation that maps the 2048-dimensional input to 256-dimensions. We then add a 1-layer transformer block with 16 heads to capture the interactions between objects and the overall image. The image provides the setting where the object is placed which can point to its informativeness/sensitivity. For example, a patient name tag in a hospital setting is more sensitive than the name on a visiting card. We obtain a $f''_{IO} = 32 \times 256$ feature from the transformer which are then passed to a linear layer to form 32x16 and flattened to result in a 512-dimensional output f'''_{IO}. Next, we concatenate the object image features f'''_{IO} with the annotator personality information and object category f_{AC}. The annotator personality and object category form a 10-dimensional input and is passed through two linear layers with a *ReLU* activation in between to obtain a 512-dimensional f_{AC}. The concatenated input $f_{ALL} = [f'''_{IO}; f_{AC}]$ is then processed through an *ReLU* activation, followed by 2 MLP blocks. Each MLP block comprises of a linear layer, a sigmoid linear activation unit (*SiLU*) [EUD18] and a dropout layer (with dropout probability of 0.2) following [Xu+24]. We use a linear layer to map the output to a 21-dimensional output.

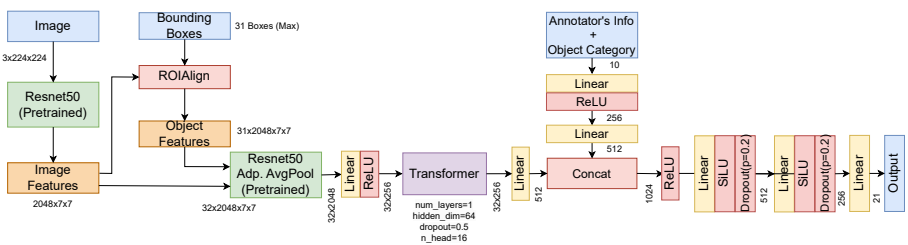


Figure 5: Our new light-weight image-privacy baseline model

3.2 Formulation

In a classical FL setting, the overall objective of FL is to minimize the global loss by aggregating the local loss from the participated users. The formulation is given as follows,

$$w^* = \arg\min_{w \in \mathbb{R}^d} \sum_{i=1}^{N} \frac{D_i}{D} F_i(w) \tag{1}$$

where N is the number of users in FL. D_i is the number of training samples that user i poses. D is the total number of samples. $\mathcal{F}_i(w)$ is the loss function

for user i, which depends on the model parameters w.

$$F_i(w) = \mathbb{E}_{(x,y)\sim D_i}[F_i(w; x, y)] \tag{2}$$

$F_i(w, x, y)$ is called the supervised loss that is formulated as the expected loss over the data distribution D_i of user i as given in Equation (2). The FL formulation results in a uniform output for all users using the global model, lacking any personalization.

As image privacy is very subjective, which causes data heterogeneity, this approach will lead to subpar performance and introduce client drift. In order to achieve a balance between generalization and personalization, the use of Personalized Federated Learning (PFL) is beneficial. We formulate the PFL problem in Equation (3).

$$\theta_i^* = \arg\min_{\theta_i} F_i(\theta_i) + \mathcal{D}ist(\theta_i^t, \theta_i^{t-1}, w) \tag{3}$$

where $\mathcal{D}ist(\theta_i^t, \theta_i^{t-1}, w)$ is a penalty term, which is typically formulated as a function of the global model, current local model, and previous local model of client i. $\mathcal{D}ist(\theta_i^t, \theta_i^{t-1}, w)$ can be formulated as a combination of similarity and stability, as given in Equation (4)

$$\mathcal{D}ist(\theta_i^t, \theta_i^{t-1}, w) = (1-\lambda)\mathcal{D}ist_{similarity} + \lambda\mathcal{D}ist_{stability} \tag{4}$$

The similarity is measured by the distance between the current local model and the global model as given in Equation (5). By minimizing this term, the local model is encouraged to stay close to the global model, ensuring that the user's model does not drift too far from the global consensus

$$\mathcal{D}ist_{\text{similarity}} = \frac{1}{2}\|\theta_i^t - w\|_2^2 \tag{5}$$

The stability is measured by the distance between the current local model and the previous local model as given in Equation (6). By minimizing this term, the local model is encouraged to change smoothly over iterations, preventing abrupt changes that might destabilize the training process.

$$\mathcal{D}ist_{\text{stability}} = \frac{1}{2}\|\theta_i^t - \theta_i^{t-1}\|_2^2 \tag{6}$$

λ is the hyperparameter to control the trade-off between similarity and stability. A lower λ places more emphasis on the similarity term, pushing the local model to align closely with the global model. A higher λ places more emphasis on the stability term, ensuring that the local model changes gradually between iterations. Based on this, we propose Algorithm 1 for the local update operation.

Algorithm 1 Local Update Algorithm

1: **Initialization:** $\theta_i^{t,s} = w^t, \theta_i^t = \theta_i^{t-1}$ ▷ $\theta_i^{t,s}$ is the current local model, θ_i^{t-1} is the previous local model, w^t is the current global model
2: **Input:** w^t , S, D_i, x, y
3: **Output:** θ_i^t
4: **for** local step $s \leftarrow 0, 1, \ldots, S-1$ **do**
5: Compute gradient of local loss:

$$\nabla F_i(\theta_i^{t,s}) = \frac{1}{|D_i|} \sum_{(x,y) \in D_i} \nabla F_i(\theta_i^{t,s}; x, y)$$

6: Update local model using gradient descent:

$$\theta_i^{t,s+1} = \theta_i^{t,s} - \eta \left(\nabla F_i(\theta_i^{t,s}) + \mathcal{D}ist(\theta_i^{t,s}, \theta_i^{t-1}, w^t) \right)$$

7: Set $\theta_i^t = \theta_i^{t,S}$
8: Send updated local model θ_i^t to the server

3.2.1 Clustered FL with daisy chaining

Users are categorized into two groups based on the volume of data they possess. The Resourceful (RF) group consists of users who have 50% of the total available data, while the resource-limited (RL) group comprises those with the remaining 50% of the data. The resourceful users possess enough data to train their individual models, whereas the resource-limited users do not. To address this, resource-limited users engage in daisy chaining (see Figure 6). We proposed two different strategies for performing Daisy Chaining.

1. RL trains models on RF's data: Resource-limited users transfer their local models to the resourceful(RF) user who is part of the same cluster. We propose Dynamic-Clustered-FedDC in Algorithm 2 to train personal models for each user using clustered federated learning and daisy chaining.

2. RL transfer model to another RL: In this strategy, RF users do not take part in daisy chaining. RF users form clusters based on their *Big5* (extraversion, agreeableness, conscientiousness, neuroticism, openness) personality scores. Each RL user identifies their clusters, and Once clusters are identified, they carry out a daisy-chaining operation involving only the remaining RL users within the cluster. We propose Apriori-Clustered-FedDC (see Algorithm 3) based on this strategy.

Dynamic-Clustered-FedDC: In Algorithm 2, at the beginning of the training, N users broadcast their data availability to the server. Then, the Server calls $Divide\_users(N)$ to separate the resourceful users from the resource-limited ones. Resourceful or resource-limited users can have full participation or partial participation in the learning. Therefore in each global iteration t

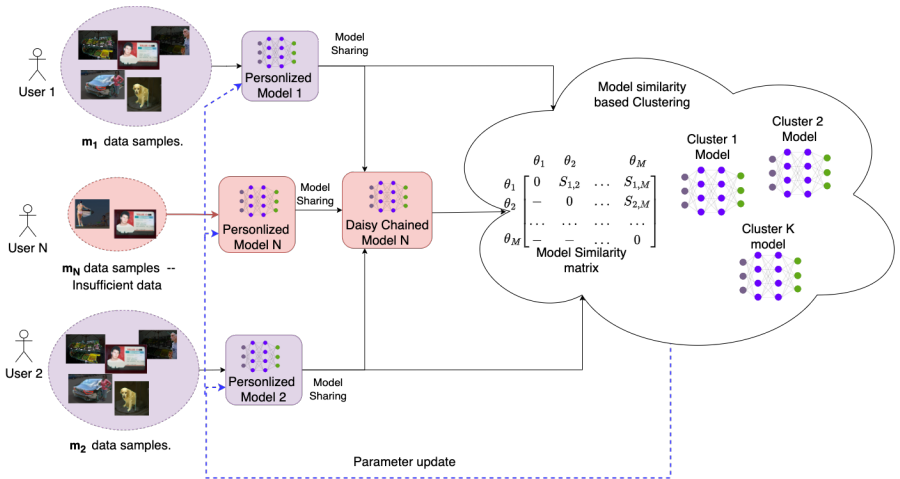


Figure 6: Clustering and daisy chaining

server calls $Select\_user(N)$ to randomly select RL_κ and RL_δ users. where κ *and* $\delta \in (0, 1)$. All users in RL_κ and RL_δ train their individual models in parallel. After that, resource-limited users share their model with resourceful users $(Transfer\_model(RL_{\kappa,i}), RF_{\delta,j})$ to train on the resourceful's data. After training, the server first performs the clustering based on the user's model similarity and then aggregates the models in the same clusters to get the cluster model (w_c^t). After generating cluster models, all the cluster aggregate their model to produce the global model (w^t). The clusters are formed dynamically. At the beginning of the FL, there are no clusters available. Resource-limited users randomly select resourceful users to daisy chain their models. At the end of the first iteration, the clusters are formed based on the similarity score between the local models of the users and their global model. We have used spectral clustering on the similarity matrix formed by the user's similarity score.

Apriori-Clustered-FedDC: In Algorithm 3, similar to Dynamic-Clustered-FedDC, at the initial phase, the server categorizes users into two groups: resourceful (RF) and resource-limited (RL), based on the available annotations from the users. The resourceful users are organized into C clusters $(RF_{c_1}, RF_{c_2}, \ldots, RF_C)$ according to their *big5* personality scores. We used the K-means algorithm to perform clustering. From each cluster, a fraction κ of the users is selected for participation in each global iteration. For instance, from cluster c_i, κ fraction of users, denoted as $RF_{c_i}^\kappa$, are chosen. Additionally, from the pool of resource-limited users (RL), a fraction δ is selected. Both κ and δ have values in the range from 0 to 1. When κ and δ are equal to 1, it indicates that all users are available for every global iteration of FL. All selected resourceful (RF) users in each cluster perform the *Local_Update* operation, as

Algorithm 2 Dynamic-Clustered-FedDC (Daisy-chaining between RL and RF)

Initialize: $w_0^0 = w_1^0 = \ldots = w_C^0 = w^0$, T, L ▷ Initialize C clusters, initialize Global iterations, and Local iterations
All users broadcast their data availability to the server.
RF, RL = Divide_users(N)
for $t \leftarrow 0$ **to** T **do**
    RF_κ, $RL_\delta \leftarrow$ Select_users($\mathcal{N}$) ▷ Select subset of resourceful and resource-limited users.
    Server sends the w_c^t or w^t to the selected users ▷ Send w^t to the selected users
    for each user in RF_κ and RL_δ in parallel **do**
        if $Lg_i \notin C$ **then** ▷ Users not assigned to any cluster
            θ_i = Local_Update(w^t)
        else ▷ User assigned to any of the c clusters, $c \in C$
            θ_i = Local_Update(w_c^t)
    for each user in RL_δ in parallel **do**
        Select user from the same cluster in RF_κ
        θ_i = Transfer_model($RL_{\delta,i}$, $RF_{\kappa,j}$) ▷ RF users transfer model with RL users for training their model on RL's data
    $C \leftarrow$ Clustering($\theta_1^t, \theta_2^t, \ldots, \theta_M^t$) ▷ Clustering based on model similarity
    for each $Lg_i \in M$ in all C clusters **do**
        $w_c^t \leftarrow w_c^t + r_i \theta_i^t$
    for each $c \in C$ in parallel **do** ▷ Global update
        $w^t = w^t + r_c w_c^t$

described in Algorithm 1, to train local models on their private data in parallel. Within each cluster, these users aggregate their local updates ($\theta^t_{RF^\kappa_{c_i}}$), but there is no cross-cluster aggregation. Users within their respective clusters aggregate their local updates to generate a cluster model ($w^t_{c_i}$). Subsequently, all cluster models from the C clusters, namely $w^t_{c_1}, w^t_{c_2}, \ldots, w^t_{c_C}$, are sent to the selected resource-limited (RL) users, denoted as RL_δ.

Each RL in RL_δ evaluates the cluster models on their private data. Based on the evaluation performance, each RL selects its corresponding cluster. For example, as shown in Table 1, there are five clusters and ten RLs. The evaluation is based on the performance of the five cluster models, measured by the F1 Score in informativeness prediction. $RL_1, RL_3,$ and RL_4 select cluster c_1 because the model of c_1 provides a better F1 Score compared to other cluster models. Similarly, RL_2 selects c_2, RL_{10} chooses c_3, $RL_5, RL_6,$ and RL_7 opt for c_4, and RL_8 and RL_9 choose cluster c_5.

Table 1: Cluster assignment for RLs

	RL_1	RL_2	RL_3	RL_4	RL_5	RL_6	RL_7	RL_8	RL_9	RL_{10}
c_1	0.21	0.19	0.16	0.23	0.2	0.2	0.2	0.2	0.2	0.2
c_2	0.19	0.21	0.15	0.1	0.19	0.19	0.19	0.19	0.17	0.19
c_3	0.2	0.2	0.13	0.21	0.17	0.19	0.17	0.2	0.17	0.22
c_4	0.2	0.17	0.14	0.2	0.21	0.21	0.21	0.19	0.16	0.2
c_5	0.17	0.2	0.15	0.17	0.12	0.19	0.12	0.21	0.21	0.21

After the clusters are assigned, resource-limited users (RLs), due to their limited data, are unable to train their individual effective model independently.

To address this, RLs utilize daisy chaining to collaboratively train a single model across all available RLs within the cluster. This daisy chaining occurs in a circular manner. For instance, as shown in Table 1, within cluster c_4, the model $w^t_{c_i}$ is first trained on RL_5's data, then transferred to RL_6 to be further trained on RL_6's data, and subsequently transferred to RL_7 for training on RL_7's data. Each RL in the cluster participates in training a single model using this daisy chaining method. Finally, the daisy-chained model $w^t_{RL_{c_i}}$ is aggregated with the cluster model ($w^t_{c_i}$) that is computed by resourceful users to produce a cluster model. It is important to note that there is no universal global model computed for all users; instead, personalized models are created for resourceful users and cluster models for resource-limited users.

Algorithm 3 Apriori-Clustered-FedDC (Daisy-chaining on RL only)

Initialize: $w^0_0 = w^0_1 = \ldots = w^0_C = w^0$, T, L ▷ Initialize C clusters, initialize Global iterations, and Local iterations
All users broadcast their data availability to the server.
RF, RL = Divide_users(N)
Based on the informativeness score given by each resourceful user, divide them into C clusters. ▷ $RF_{c_1}, RF_{c_2}, \ldots RF_C$
for $t \leftarrow 0$ **to** T **do** ▷ T is the global iterations, t is the current global iteration
  for all $c_i \in C$ in parallel **do**
    $RF^{\kappa}_{c_i} \leftarrow$ Select_users(RF_{c_i}) ▷ Select subset of resourceful users from their cluster $c_i \in C$
  $RL_\delta \leftarrow$ Select_users(RL) ▷ Select subset of resource-limited users from the user's pool
  for all $c_i \in C$ in parallel **do**
    Compute $\theta^t_{RF^{\kappa}_{c_i}} \leftarrow Local\_Update(w^t_{c_i})$ ▷ Perform Algorithm 1
  for all $c_i \in C$ in parallel **do**
    Compute $w^t_{c_i} = \frac{1}{RF^{\kappa}_{c_i}} \sum_{1}^{RF^{\kappa}_{c_i}} \theta^t_{RF^{\kappa}_{c_i}}$ ▷ Compute cluster model based on RF's only
  Server sends all cluster models $w^t_{c_i} \in C$ to the all user users in RL_δ
  for all user in RL_δ in parallel **do**
    Evaluate all cluster models on their private data and select the cluster which one is giving maximum accuracy.
  A subset of RL users are selected from each clusters. ▷ $RL_{c_1}, RL_{c_2}, \ldots, RL_C$
  for $c_i \leftarrow 0$ to RL_{c_i} in parallel in all C **do**
    Perform $w^t_{RL_{c_i}} \leftarrow Daisy\_chain(w^t_{c_i}, RL_{c_i})$
  for all $c_i \in$ C in parallel **do**
    $w^t_{c_i} \leftarrow \frac{w^t_{c_i} + w^t_{RL_{c_i}}}{2}$ ▷ $\bar{w^t_c}$ is the cluster model for cluster $c \in C$

4 Evaluation

4.1 Experimental setup

We conducted experiments using the DIPA2 dataset [Xu+24]. This dataset offers images along with privacy sensitivity scores, emphasizing user-perceived privacy from a cross-cultural viewpoint. DIPA2 includes comprehensive object-level annotations for 1,304 images, totaling 5,897 annotations that detail perceived privacy risks for 3,347 objects. Each annotation provides four key

privacy-related metrics: information type, informativeness, sharing scope as perceived by the photo owner (sharing ownar), and sharing scope by others (sharing others). The dataset incorporates cultural variations by including annotations from both Japan and the UK, with contributions from 300 participants in each country. Each image is annotated by two individuals from each country, resulting in annotations that may differ between annotators from Japan and the UK. Consequently, each image's annotations are subjective to the individual annotators. In our experiments, we included 525 annotators out of the original 600, as 75 annotators only provided one annotation each. Among the 525 annotators, 112 (21%) are classified as resourceful annotators, while the remaining 413 (79%) are considered resource-limited. We trained the PIONet model on individual annotators' data using either direct local updates or a daisy-chaining approach. Note that, annotators are the users where the local update operation is performed. For each annotator, the data is split into training, validation, and test sets in a 7:2:1 ratio. All the experiments are performed on the NVIDIA A100 Tensor Core GPU enabled with 40GB of high-bandwidth memory.

4.2 Performance

In our initial comparison between PIONet and the baseline model [Xu+24] in centralized settings, we observed that PIONet does not surpass the baseline in performance across "Information Type" and "Informativeness". However, PIONet outperforms the baseline in the "Sharing Owner" category and performs on par with the baseline in "Sharing Others". One of the most significant advantages of PIONet is its ability to reduce the model size by at least 20 $\times$. This substantial size reduction makes PIONet particularly advantageous for Federated Learning applications, where smaller models can enhance communication efficiency and reduce computational demands. We trained PIONet on various state-of-the-art Federated Learning algorithms, including FedAvg, FedProx, FedDC, and FedMEM, as well as on our proposed Dynamic-Clustered-FedDC and Apriori-Clustered-FedDC. The experiments in Table 2 consider the involvement of all RLs and RFs in every global iteration of FL.

We observed that Federated Learning methods like FedAvg, FedProx, and FedDC have poor performance, reflected in low scores across all metrics. The global model of Dynamic-Clustered-FedDC also shows weak performance in the "Information Type" category, with scores similar to those of FedAvg and FedProx. For the "Sharing Owner" category, the metrics improve slightly but remain low overall, while "Sharing Others" scores are also low, and "Informativeness" scores are minimal. In contrast, the cluster model in Apriori-Clustered-FedDC demonstrates mediocre performance, with scores slightly better than FedAvg and FedProx. Compared to Dynamic-Clustered-FedDC, the global model achieves higher precision, recall, and F1 Scores for the "Information Type" category, similar scores for "Sharing Owner" and "Informativeness," but lower scores for "Sharing Others." The personalized model in

Dynamic-Clustered-FedDC achieved the highest scores across all four categories, surpassing PIONet (Centralized). The personalized model in Apriori-Clustered-FedDC shows scores comparable to PIONet in the "Information Type" category, slightly lower scores in "Sharing Owner" and "Sharing Others", but better performance in "Informativeness". Also, the personalized models in Dynamic-Clustered-FedDC and Apriori-Clustered-FedDC outperformed the baseline by 5% in the "Informativeness" category, while the personalized model in Dynamic-Clustered-FedDC outperformed the baseline by 7% in the "Sharing Owner" category. Moreover, compared with FedMEM with dynamic clustering, the personalized model of Dynamic-Clustered-FedDC produces better F1 Scores in all four categories. Whereas the personalized model of Apriori-Clustered-FedDC only gives a better F1 Score on "Informativeness" and an equivalent F1 Score on "Information Type".

From the observations, we inferred that global models are not effective because of the presence of heterogeneity in the data. Daisy chaining boosts performance but not significantly. Proposed algorithms utilized the benefits of clustering and personalization with daisy chaining to produce a better-personalized model and a global model.

Table 2: Performance analysis of PIONet on centralize and federated setup

Model	Information Type			Sharing Owner			Sharing Others			Informativeness		
	Prec. ↑	Recall ↑	F1 ↑	Prec. ↑	Recall ↑	F1 ↑	Prec. ↑	Recall ↑	F1 ↑	Prec. ↑	Recall ↑	F1 ↑
Centralized												
Baseline [Xu+24]	**0.62**	**0.60**	**0.61**	0.60	0.57	0.58	0.60	0.57	0.58	**0.32**	0.26	0.24
PIONet	0.57	0.47	0.47	**0.66**	0.54	0.59	0.64	0.53	0.58	0.22	0.17	0.15
Federated												
PIONet + FedAvg [McM+17]	0.22	0.38	0.28	0.17	0.33	0.22	0.00	0.00	0.00	0.01	0.14	0.03
PIONet + FedProx [Li+20b]	0.23	0.37	0.28	0.18	0.34	0.24	0.0	0.0	0.0	0.01	0.14	0.02
PIONet + FedDC [KFV22]	0.32	0.39	0.34	0.27	0.33	0.29	0.0	0.0	0.0	0.06	0.09	0.07
PIONet + FedMEM (Personalized)	0.55	0.46	0.47	**0.66**	0.55	0.60	**0.66**	0.53	0.58	0.26	0.26	0.24
PIONet + Dynamic-Clustered-FedDC(Global)	0.19	0.35	0.25	0.17	0.33	0.22	0.18	0.36	0.24	0.01	0.14	0.02
PIONet + Dynamic-Clustered-FedDC(Personalized)	0.56	0.56	0.56	0.63	**0.67**	**0.65**	0.64	**0.63**	**0.64**	**0.32**	**0.31**	**0.29**
PIONet + Apriori-Clustered-FedDC(Cluster)	0.26	0.56	0.35	0.17	0.33	0.22	0.10	0.27	0.15	0.01	0.14	0.02
PIONet + Apriori-Clustered-FedDC(Personalized)	0.54	0.45	0.47	0.64	0.52	0.57	0.62	0.52	0.56	0.3	0.3	0.28

According to Figure 7, in centralized settings, the baseline [Xu+24] and PIONet achieved MAE values of 1.24 and 1.22, respectively. Among the federated methods, FedAvg, FedProx, and FedDC, which are state-of-the-art techniques, yielded higher MAE values of 1.43, 1.44, and 1.4, respectively. In our approach, the personalized model in Dynamic-Clustered-FedDC showed the lowest MAE at 1.14, followed by FedMEM with an MAE of 1.17. The personalized model in Apriori-Clustered-FedDC showed an MAE of 1.19. This observation suggests that our dynamic clustering approaches in federated learning (Dynamic-Clustered-FedDC and FedMEM) can significantly reduce MAE compared to

other federated methods.

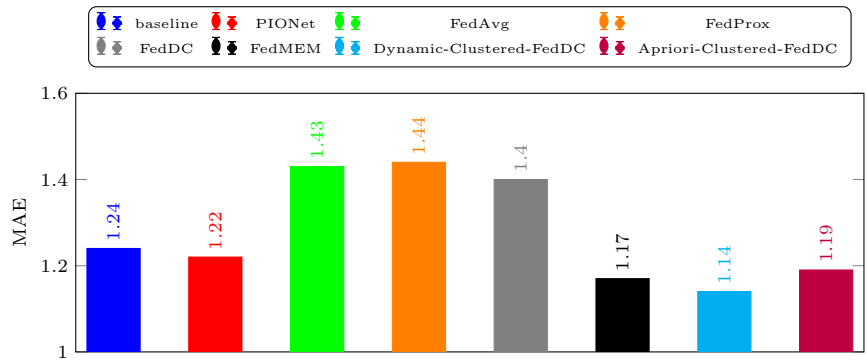


Figure 7: Mean Absolute Error (MAE) by different centralized (Baseline, and PIONet) and federated (FedAvg, FedProx, FedDC, FedMEM, Dynamic-Clustered-FedDC, Apriori-Clustered-FedDC,) learning approaches

4.3 Ablation study on partial participation of users in FL training

4.3.1 Dynamic-Clustered-FedDC

Varing participation of RF users: The ablation studies in Figure 8 show that the F1 Score generally increases with higher values of κ, meaning that higher participation of RL clients has a positive impact on the performance of Dynamic-Clustered-FedDC. In the "Information Type" category, the F1 Score remains constant at 0.38 for $\kappa = 0.1$ and $\kappa = 0.5$ but increases to 0.40 for $\kappa = 0.8$, indicating a slight improvement. Similarly, for the "Sharing Owner" category, the F1 Score improves from 0.42 at $\kappa = 0.1$ to 0.46 at $\kappa = 0.8$. The "Sharing Others" category shows a significant rise in the F1 Score from 0.48 at $\kappa = 0.1$ to 0.52 at $\kappa = 0.8$, despite a slight dip at $\kappa = 0.5$. However, in the "Informativeness" category, the F1 Score remains relatively stable, with minimal change, hovering around 0.20 to 0.21 across different κ values. This suggests that while a higher number of resourceful user participation generally enhances performance in most categories, informativeness is less influenced by changes.

Varing participation of RL users: The ablation studies presented in Figure 9 demonstrate the impact of varying the participation (δ) of RL users on the F1 Score across four different categories. In Figure 9a, which focuses on the "Information Type", the F1 Score decreases as δ increases, with the highest score observed at $\delta = 0.1$ (0.38) and the lowest at $\delta = 0.8$ (0.35). Conversely, Figure 9b examines the "Sharing Owner", where the F1 Score increases with δ,

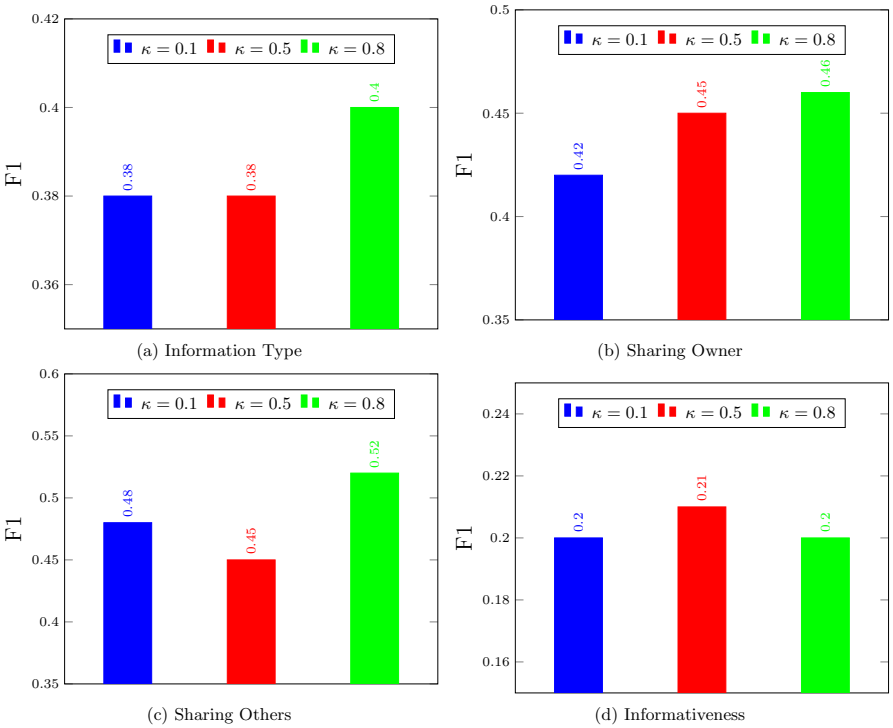


Figure 8: Ablation studies of Dynamic-Clustered-FedDC based on RF's partial participation ($\kappa = 0.1$, 0.5, and 0.8) while RL's participation is $\delta = 0.1$

peaking at $\delta = 0.8$ (0.45) and reaching a minimum at $\delta = 0.5$ (0.40).Figure 9c explores the Sharing Others scenario, where the F1 Score shows a slight decline as δ increases, with the highest score at $\delta = 0.1$ (0.48) and a minor reduction at $\delta = 0.5$ (0.45), followed by a slight increase at $\delta = 0.8$ (0.47). Finally, Figure 9d analyzes Informativeness, demonstrating a consistent increase in the F1 Score as δ rises, from 0.20 at $\delta = 0.1$ to a maximum of 0.26 at $\delta = 0.8$.We've observed that even with only partial involvement of users, Dynamic-Clustered-FedDC performs well. In two out of four categories, a low participation rate (delta=0.1) of RL clients results in the best F1 Score. For the informativeness category, 50% of total RLs' involvement in FL can yield the maximum F1 Score. For the "Sharing Owner" category, having 80% of RLs participate in every global iteration produces the maximum F1 Score.

4.3.2 Apriori-Clustered-FedDC

Varing participation of RF users while RL has full participation: In Figure 10a, focusing on the "Information Type", the F1 Score increases as RFs

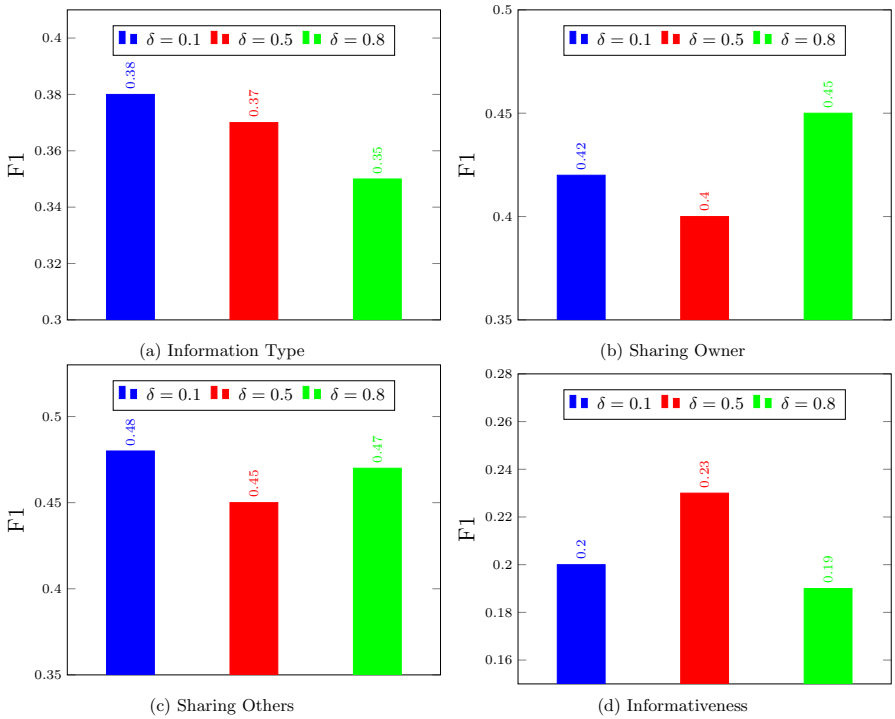
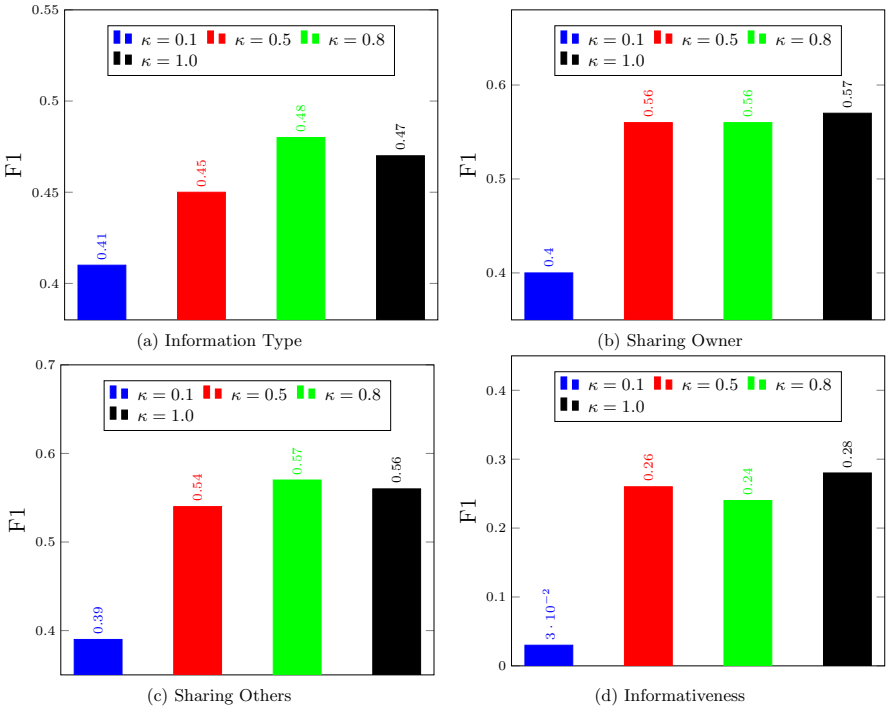


Figure 9: Ablation studies of Dynamic-Clustered-FedDC based on RL's partial participation ($\delta = 0.1$, 0.5, and 0.8) while RF's participation is $\kappa = 0.1$

participation (κ) in each global iteration increases. Starting from 0.41 at 10% ($\kappa = 0.1$) participation and reaching a peak of 0.48 at 80% ($\kappa = 0.8$) participation, followed by a slight decrease to 0.47 when users fully participated ($\kappa = 1.0$). Figure 10b examines the influence of RFs participation in the "Sharing Owner" category. F1 Score substantially increases from 0.40 to 0.56 while RF participation increases from 10%($\kappa = 0.1$) to 50% ($\kappa = 0.5$). After that, the performance stabilizes around 0.56 to 0.57 as κ increases further to 0.8 and 1.0. In Figure 10c F1 Score demonstrates a consistent rise from 0.39 at 10% ($\kappa = 0.1$) participation to 0.57 at 80% ($\kappa = 0.8$) participate. After that, a slight reduction to 0.56 is observed when all users are participating ($\kappa = 1.0$) in the "Sharing Other" category. Finally, Figure 10d looks at "Informativeness", where the F1 Score increases sharply from 0.03 to 0.26 when users participation increases from 10% ($\kappa = 0.1$) to 50% ($\kappa = 0.5$) and continues to rise moderately to 0.28 with full participation ($\kappa = 1.0$).

Overall, the performance of Apriori-Clustered-FedDC remains stable with the involvement of users being more than 50% in each global iteration.



(a) Information Type (b) Sharing Owner

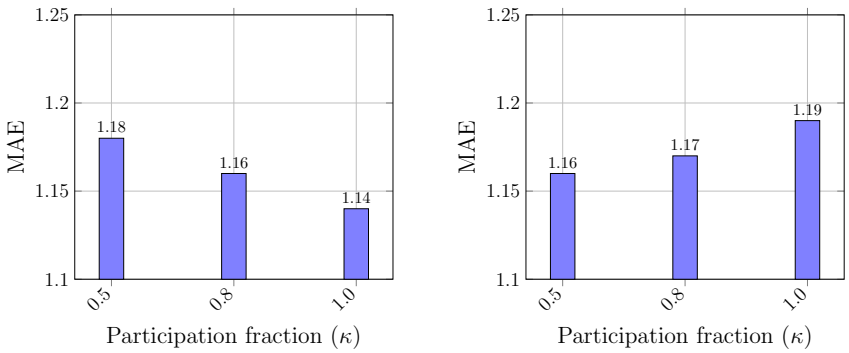
(c) Sharing Others (d) Informativeness

Figure 10: Ablation studies of Apriori-Clustered-FedDC based on RF's partial participation (10%, 50%, and 80%) while RL's participation is 100%

4.4 Comparative study between Dynamic-Clustered-FedDC and Apriori-Clustered-FedDC

We compared the performance of Dynamic-Clustered-FedDC and Apriori-Clustered-FedDC under varying participation fractions (κ) of RFs. In Figure 11a, for the Dynamic-Clustered-FedDC, the Mean Absolute Error (MAE) decreases as the participation of resourceful users increases. For 50% participation($\kappa = 0.5$) Dynamic-Clustered-FedDC produce MAE 1.18 that consistently declines to 1.14 as κ reaches 1.0 i.e., full participation of users. This trend suggests that the Dynamic-Clustered-FedDC model benefits from higher participation fractions, leading to more accurate predictions. In contrast, Apriori-Clustered-FedDC (see Figure 11b), shows a different pattern. Here, the MAE slightly increases with higher participation fractions, starting at 1.16 for $\kappa = 0.5$ and rising to 1.19 when κ is 1.0. This indicates that the Apriori-Clustered-FedDC may experience slight degradation in performance with increased participation. Overall, while higher participation improves the performance of the Dynamic-Clustered-FedDC, it appears to have an opposite

or negligible effect on the Apriori-Clustered-FedDC.



(a) Dynamic-Clustered-FedDC (b) Apriori-Clustered-FedDC

Figure 11: Informativeness MAE measurements under partial participation (κ) of RFs

5 Conclusion and Future Work

In this paper, we proposed two novel daisy-chaining enabled clustered federated learning algorithms, Dynamic-Clustered-FedDC and Apriori-Clustered-FedDC, to develop personalized image privacy advisors. We also introduced a new baseline model, PIONet, which is 20× lighter than the original DIPA2 baseline [Xu+24]. Our experiments demonstrate that PIONet in a centralized setup performs comparably to the DIPA2 baseline. Moreover, PIONet, together with Dynamic-Clustered-FedDC, outperformed the centralized baseline in three out of four categories, highlighting the efficacy of our approach. Specifically, Dynamic-Clustered-FedDC and Apriori-Clustered-FedDC achieved a 5% and 4% improvement in F1-score, respectively, over the baseline in the informativeness category. Importantly, both Dynamic-Clustered-FedDC and Apriori-Clustered-FedDC support the partial participation of annotators in federated learning, maintaining strong performance without degradation. These findings underscore the potential of our proposed models in enhancing the effectiveness and scalability of privacy advisory systems in federated learning environments. In the future, we aim to further enhance the performance of the model.

Acknowledgments

This work was partially supported by the Wallenberg AI, Autonomous Systems and Software Program (WASP) funded by the Knut and Alice Wallenberg Foundation. The computations were enabled by the Berzelius resource

provided by the Knut and Alice Wallenberg Foundation at the National Supercomputer Centre. This work was also partially funded by the A*STAR CDA project—202D800021.

References

[BFA20] Christopher Briggs, Zhong Fan, and Peter Andras. "Federated learning with hierarchical clustering of local updates to improve training on non-IID data". In: *2020 International Joint Conference on Neural Networks (IJCNN)*. IEEE. 2020, pp. 1–9.

[CKS21] Zhang Chen, Thivya Kandappu, and Vigneshwaran Subbaraju. "PrivAttNet: predicting privacy risks in images using visual attention". In: *2020 25th International Conference on Pattern Recognition (ICPR)*. IEEE. 2021, pp. 10327–10334.

[EUD18] Stefan Elfwing, Eiji Uchibe, and Kenji Doya. "Sigmoid-weighted linear units for neural network function approximation in reinforcement learning". In: *Neural networks* 107 (2018), pp. 3–11.

[Gur+19] Danna Gurari et al. "Vizwiz-priv: A dataset for recognizing the presence and purpose of private visual information in images taken by blind people". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2019, pp. 939–948.

[He+17] Kaiming He et al. "Mask r-cnn". In: *Proceedings of the IEEE international conference on computer vision*. 2017, pp. 2961–2969.

[Hen+12] Xu Heng et al. "Effects of Individual Self-Protection, Industry Self-Regulation, and Government Regulation on Privacy Concerns: A Study of Location-Based Services". In: *Information Systems Research* 23.4 (2012), pp. 1342–1363.

[HHB22] Hannes Hilberger, Sten Hanke, and Markus Bödenler. "Federated learning with dynamic model exchange". In: *Electronics* 11.10 (2022), p. 1530.

[Hua+19] Li Huang et al. "Patient clustering improves efficiency of federated machine learning to predict mortality and hospital stay time using distributed electronic medical records". In: *Journal of biomedical informatics* 99 (2019), p. 103291.

[JJS08] Iris A Junglas, Norman A Johnson, and Christiane Spitzmüller. "Personality traits and concern for privacy: an empirical study in the context of location-based services". In: *European Journal of Information Systems* 17.4 (2008), pp. 387–402.

[KFV22] Michael Kamp, Jonas Fischer, and Jilles Vreeken. "Federated Learning from Small Datasets". In: *The Eleventh International Conference on Learning Representations*. 2022.

[Li+20a] Tian Li et al. “Federated learning: Challenges, methods, and future directions”. In: *IEEE signal processing magazine* 37.3 (2020), pp. 50–60.

[Li+20b] Tian Li et al. “Federated optimization in heterogeneous networks”. In: *Proceedings of Machine learning and systems* 2 (2020), pp. 429–450.

[Lon+23] Guodong Long et al. “Multi-center federated learning: clients clustering for better personalization”. In: *World Wide Web* 26.1 (2023), pp. 481–500.

[Man+20] Yishay Mansour et al. “Three approaches for personalization with applications to federated learning”. In: *arXiv preprint arXiv:2002.10619* (2020).

[Mat+22] Koji Matsuda et al. “Fedme: Federated learning via model exchange”. In: *Proceedings of the 2022 SIAM international conference on data mining (SDM)*. SIAM. 2022, pp. 459–467.

[McM+17] Brendan McMahan et al. “Communication-efficient learning of deep networks from decentralized data”. In: *Artificial intelligence and statistics*. PMLR, 2017, pp. 1273–1282.

[Ngu+22] Minh NH Nguyen et al. “Self-organizing democratized learning: Toward large-scale distributed learning systems”. In: *IEEE Transactions on Neural Networks and Learning Systems* 34.12 (2022), pp. 10698–10710.

[OSF17] Tribhuvanesh Orekondy, Bernt Schiele, and Mario Fritz. “Towards a visual privacy advisor: Understanding and predicting privacy risks in images”. In: *Proceedings of the IEEE international conference on computer vision*. 2017, pp. 3686–3695.

[She02] Kim Bartel Sheehan. “Toward a typology of Internet users and online privacy concerns”. In: *The information society* 18.1 (2002), pp. 21–32.

[SMS20] Felix Sattler, Klaus-Robert Müller, and Wojciech Samek. “Clustered federated learning: Model-agnostic distributed multitask optimization under privacy constraints”. In: *IEEE transactions on neural networks and learning systems* 32.8 (2020), pp. 3710–3722.

[Suc+17] Jose M Such et al. “Photo privacy conflicts in social media: A large-scale empirical study”. In: *Proceedings of the 2017 CHI conference on human factors in computing systems*. 2017, pp. 3821–3832.

[TC16] Ashwini Tonge and Cornelia Caragea. “Image privacy prediction using deep features”. In: *Proceedings of the AAAI conference on artificial intelligence*. Vol. 30. 1. 2016.

[TC19] Ashwini Tonge and Cornelia Caragea. “Dynamic deep multi-modal fusion for image privacy prediction”. In: *The World Wide Web Conference*. 2019, pp. 1829–1840.

[TTN20] Canh T Dinh, Nguyen Tran, and Josh Nguyen. “Personalized federated learning with moreau envelopes”. In: *Advances in Neural Information Processing Systems* 33 (2020), pp. 21394–21405.

[WNC11] Yang Wang, Gregory Norice, and Lorrie Faith Cranor. “Who Is Concerned about What? A Study of American, Chinese and Indian Users’ Privacy Concerns on Social Network Sites: (Short Paper)”. In: *Trust and Trustworthy Computing: 4th International Conference, TRUST 2011, Pittsburgh, PA, USA, June 22-24, 2011. Proceedings 4*. Springer. 2011, pp. 146–153.

[Xu+23] Anran Xu et al. “DIPA: An Image Dataset with Cross-cultural Privacy Concern Annotations”. In: *Companion Proceedings of the 28th International Conference on Intelligent User Interfaces*. 2023, pp. 259–266.

[Xu+24] Anran Xu et al. “DIPA2: An Image Dataset with Cross-cultural Privacy Perception Annotations”. In: *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 7.4 (2024), pp. 1–30.

[Yu+16] Jun Yu et al. “iPrivacy: image privacy protection by identifying sensitive objects via deep multi-task learning”. In: *IEEE Transactions on Information Forensics and Security* 12.5 (2016), pp. 1005–1016.

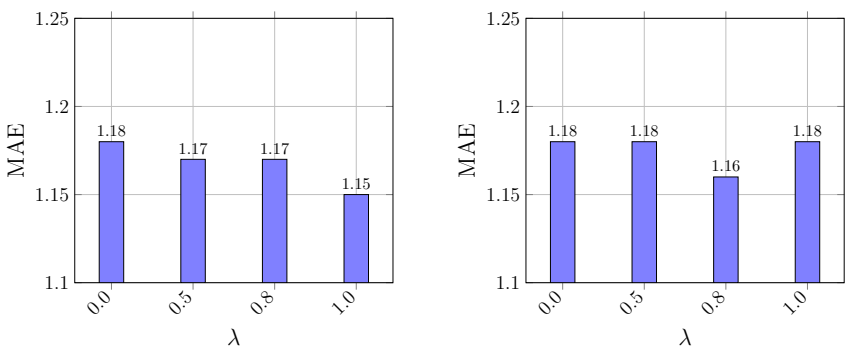
[Zho+17] Haoti Zhong et al. “A Group-Based Personalized Model for Image Privacy Classification and Labeling.” In: *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence (IJCAI)*. Vol. 17. 2017, pp. 3952–3958.

Appendix A Additional Results and Analysis

Appendix A.1 Ablation study on the value of λ

In Figure 12 and 13, we trained Dynamic-Clustered-FedDC and Apriori-Clustered-FedDC across various values of λ. When λ is set to 0.0, the stability term is eliminated, placing full emphasis on similarity. Conversely, when λ is set to 1.0, the similarity term is eliminated, shifting the full emphasis to stability.

In Figure 12a, for the Dynamic-Clustered-FedDC model, as λ increases from 0.0 to 1.0, the MAE gradually decreases. The MAE decreases from 1.18 at $\lambda = 0.0$ to 1.15 at $\lambda = 1.0$. In Figure 12b, for the Apriori-Clustered-FedDC model, the MAE also shows a slight decrease as λ increases. It decreases from 1.18 at $\lambda = 0.0$ to 1.16 at $\lambda = 0.8$, but then remains constant at 1.18 for $\lambda = 1.0$.

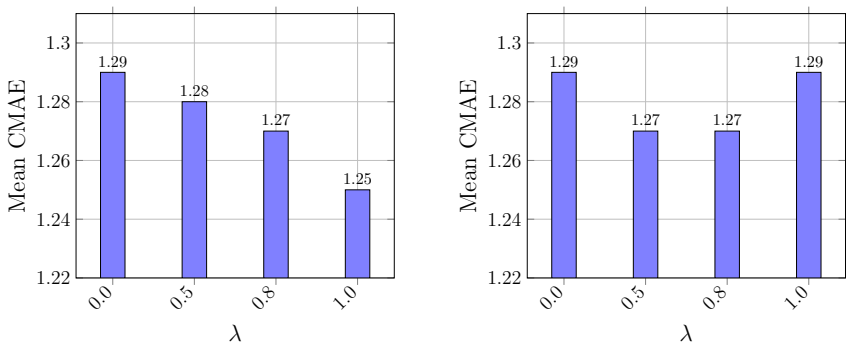


(a) Dynamic-Clustered-FedDC (b) Apriori-Clustered-FedDC

Figure 12: Comparing Informativeness MAE of Dynamic-Clustered-FedDC and Apriori-Clustered-FedDC for different λ

In Figure 13a, for the Dynamic-Clustered-FedDC, the similar to MAE, Mean Classwise MAE (CMAE) also decreases as λ increases. It drops from 1.29 at $\lambda = 0.0$ to 1.25 at $\lambda = 1.0$. In Figure 13b, for the Apriori-Clustered-FedDC, the Mean CMAE decreases from 1.29 at $\lambda = 0.0$ to 1.27 at $\lambda = 0.5$ and $\lambda = 0.8$, but then returns to 1.29 at $\lambda = 1.0$.

From the observations, we infer that Dynamic-Clustered-FedDC consistently improves both MAE and Mean CMAE as λ increases. This suggests that putting more emphasis on stability benefits the performance of Dynamic-Clustered-FedDC, leading to reduced error rates. The Apriori-Clustered-FedDC shows some improvement in both MAE and Mean CMAE with increasing λ, particularly around the middle values (0.5 and 0.8). However, at $\lambda = 1.0$, the performance seems to revert to 1.29, indicating that an excessive focus on stability or similarity might not benefit Apriori-Clustered-FedDC. A



(a) Dynamic-Clustered-FedDC (b) Apriori-Clustered-FedDC

Figure 13: Comparing Informativeness CMAE of Dynamic-Clustered-FedDC and Apriori-Clustered-FedDC for different λ

combination of similarity and stability is needed for Apriori-Clustered-FedDC.