



ARTIFICIAL
INTELLIGENCE
RESEARCH GROUP



Master thesis submitted in partial fulfilment of the requirements for the degree of
Master of Science in Applied Sciences and Engineering: Computer Science

FEDERATED LEARNING IN NEUROIMAGING: PIONEERING PREDICTIVE MODELS FOR BRAIN AGE ESTIMATION

Advancing Brain Age Prediction through
Collaborative and Privacy-Conscious
Machine Learning Approaches

Alvaro Javier Vargas Guerrero

2023-2024

Promotor(s): Prof. Dr. Ann Nowé, Prof. Dr. Guy Nagels
Science and Bio-Engineering Sciences

Proefschrift ingediend met het oog op het behalen van de graad van Master of Science in de ingenieurswetenschappen: computerwetenschappen

FEDERATED LEARNING IN NEUROIMAGING: BAANBREKENDE VOORSPELLENDE MODELLEN VOOR HERSENLEEFTIJDSSCHATTING

Vooruitgang in de Voorspelling van
Hersenleeftijd door Samenwerkende en
Privacybewuste Machine Learning
Benaderingen

Alvaro Javier Vargas Guerrero

2023-2024

Promotor(s): Prof. Dr. Ann Nowé, Prof. Dr. Guy Nagels
Wetenschappen en Bio-ingenieurswetenschappen

Abstract

This thesis explores the integration of Federated Learning (FL) within the domain of brain age prediction using structural Magnetic Resonance Imaging (MRI) scans. Leveraging the FL framework Flower, this research enables medical institutions to collaboratively develop predictive models while ensuring data privacy and compliance with regulations such as GDPR and the California Consumer Privacy Act. By adopting domain adaptation techniques to ensure age representation across all nodes, this study mitigates the challenges posed by heterogeneous data distributions, enhancing model performance. The findings highlight the effectiveness of using pre-trained models, particularly with algorithms like FedProx, which correct client drift, demonstrating their advantage in scenarios with highly skewed data distributions. This research underscores the potential of FL to produce competitive models suitable for practical application in neuroimaging tasks. The implications extend beyond brain age prediction, suggesting broader applications in healthcare, where data privacy and collaborative learning are crucial. This thesis serves as a foundational effort, merging theoretical advancements with practical applications in medical diagnostics, and encourages further exploration and development in these domains.

Deze thesis onderzoekt de integratie van Federated Learning (FL) binnen het domein van het voorspellen van de hersenleeftijd met behulp van structurele Magnetic Resonance Imaging (MRI) scans. Door gebruik te maken van het FL-framework Flower stelt dit onderzoek medische instellingen in staat om samen voorspellende modellen te ontwikkelen en tegelijkertijd de gegevensprivacy te waarborgen en te voldoen aan regelgeving zoals GDPR en de California Consumer Privacy Act. Door domeinaanpassingstechnieken toe te passen om ervoor te zorgen dat de leeftijd op alle knooppunten vertegenwoordigd is, vermindert dit onderzoek de problemen die ontstaan door heterogene gegevensdistributies, waardoor de modelprestaties verbeteren. De bevindingen benadrukken de effectiviteit van het gebruik van voorgetrainde modellen, namelijk met algoritmen zoals FedProx, die cliëntdrift corrigeren en hun voordeel aantonen in scenario's met zeer scheve gegevensdistributies. Dit onderzoek onderstreept het potentieel van FL om competitieve modellen te produceren die geschikt zijn voor praktische toepassing in neuroimagingtaken. De implicaties reiken verder dan het voorspellen van de hersenleeftijd en suggereren bredere toepassingen in de gezondheidszorg, waar gegevensprivacy en samenwerkend leren cruciaal zijn. Deze thesis dient als een fundamentele inspanning, waarbij theoretische vooruitgang wordt samengevoegd met praktische toepassingen in de medische diagnostiek, en moedigt verdere verkenning en ontwikkeling in deze domeinen aan.

Acknowledgments

I would like to express my deepest gratitude to my promotor: Prof. Dr. Ann Nowé and Prof. Dr. Guy Nagels for their guidance, support, and encouragement throughout this research. And I would like to thank the AIMS group for allowing me to use their infrastructure and resources for this research. I've worked previously with the AIMS group, and I'm grateful for the opportunity to work with them again. I would also like to thank supervisor Stijn Denissen for joining the project and providing valuable insights and feedback about the principles of Federated Learning.

A special mention to my friends for keeping me company and inspiring me to achieve goals that I never thought possible. And my professors for teaching me the fundamentals of computer science and machine learning, effectively turning me from an interested student about the field to an incredible programmer.

Finally, I would like to thank my family for their unwavering support and encouragement throughout my academic journey. Their love and guidance have been my source of strength and motivation. In particular, I would like to thank two people, my mother and my stepfather. Having moved to Belgium at a young age, I was faced with the challenge of adapting to a new culture and language. I was not alone in this journey as my mother and stepfather were there to support me every step of the way. They have always believed in me and encouraged me to pursue my dreams. I am eternally grateful for their love, patience, and unwavering support. They granted me opportunities that I would not have had otherwise, and I am forever thankful for that. I have worked hard to make them proud, and I hope that this thesis is a testament to their love and support. I dedicate this work to them.

Contents

1	Introduction	5
2	Background	7
2.1	Federated Learning	7
2.1.1	Data-Aligned Federated Learning Paradigms	8
2.1.2	Scalability-Oriented Federated Learning Paradigms	9
2.1.3	Network-Architecture-Based Federated Learning Paradigms	11
2.1.4	Blockchain-Enabled Federated Learning	13
2.1.5	Comparison with Distributed Machine Learning	15
2.2	Challenges in FL	16
2.2.1	Model Security	16
2.2.2	Communication Overhead	16
2.2.3	Heterogeneity	17
2.3	Privacy & Security in FL	18
2.3.1	Regulations surrounding Data Protection	18
2.3.2	Threat Models	19
2.3.3	Poisoning & Inference Attacks	21
2.3.4	Privacy-Preserving Techniques	24
2.4	Model Consolidation Techniques	29
2.4.1	Federated Optimization	29
2.4.2	Model Updates	29
2.4.3	FedAvg: The Baseline Algorithm	29
2.4.4	Addressing Statistical and Systems Heterogeneity	31
2.4.5	Adapting Data Distributions	33
2.4.6	Addressing Heterogeneity as Tasks	35
2.4.7	Approaches Involving Multiple Models	35
2.5	Fairness in FL	37
2.5.1	Incentive Mechanisms	37
2.5.2	Fairness of Data	38
2.5.3	Fairness of Model	38
2.6	Federated Learning Frameworks	39
2.6.1	Criteria for Evaluating Federated Learning Frameworks	39
2.6.2	PySyft	40
2.6.3	Flower	41
2.6.4	OpenFL	41
2.6.5	TensorFlow Federated	42

2.6.6	NVFlare	42
2.6.7	FATE	43
2.6.8	LEAF	43
2.7	Medical Applications of Federated Learning	44
2.7.1	Brain Age Detection	44
2.7.2	Other applications of Federated Learning in Healthcare	45
2.8	Research Question	46
3	Experimental Setup	47
3.1	Data	47
3.1.1	Subject Dataset	47
3.1.2	Data Partitioning	49
3.2	Experimental Configuration	51
3.2.1	Model Starting Point	51
3.2.2	Federation	51
3.2.3	Local Training	52
3.2.4	Model Aggregation	52
3.2.5	Plotting	53
3.2.6	Hardware & Software Setup	53
4	Results	54
4.1	3 Nodes Experiment	54
4.2	6 Nodes Experiment	55
4.3	Model Performance	57
5	Discussion	63
5.1	Overview of Experimental Findings	63
5.1.1	Summary of Key Results	63
5.1.2	Contextual Comparison	63
5.2	Interpretation of Results	63
5.2.1	Comparative Analysis of Brain Age Models	63
5.2.2	Age Distribution	64
5.2.3	Aggregation Algorithm	65
5.2.4	Model Starting Point	65
5.2.5	Amount of Nodes	65
5.3	Challenges & Limitations	65
5.4	Future Research Directions	67
5.5	Theoretical and Practical Implications	69
6	Conclusion	70
Appendices		
A	Additional Plots	72

Chapter 1

Introduction

Artificial Intelligence (AI), more specifically Machine Learning (ML), has seen tremendous technological advances with the introduction of Neural Networks (NN) and Deep Learning (DL) [1]. Deep Learning algorithms have been shown to excel in recognizing advanced patterns, improving accuracy on predictive analytics and handling complex data structures. Deep learning models, owing to their advanced computational capabilities, have demonstrated significant efficacy in enhancing various business sectors [2]. This includes applications in stock market analysis, where predictive modeling aids in making informed decisions [3], and weather forecasting, which benefits from the models' capability to analyze extensive datasets for accurate predictions [4]. Additionally, in the realm of edge computing within the Internet of Things (IoT), these models enhance efficient data processing and decision-making at the device level, thereby improving operational efficiency [5].

However, the application of artificial intelligence in the medical domain has not yielded comparable advancements as observed in other sectors. This disparity arises primarily due to intrinsic challenges in the medical field that hinder the effective training of large deep learning (DL) models. A predominant issue is the scarcity of training data. Even large-scale hospitals possessing a substantial amount of data can't produce DL models that achieve the necessary accuracy for reliable medical application. Efforts to address the aforementioned challenge in the medical sector have led to several notable solutions. Primarily, medical institutions are increasingly collaborating by exchanging subject data to collectively solve a common goal. On a broader scale, the implementation of data lakes has emerged as a prevalent strategy for managing heterogeneous data within the healthcare domain. These data lakes, characterized by their capacity to store and process diverse data types, are not only utilized by medical institutions but also accessible to third-party entities, including clinical laboratories, pharmacies, and insurance companies [6]. Despite these efforts, data protection regulations have made it difficult to share data across institutions for model training. Notable examples include the General Data Protection Regulation (GDPR) [7], which has been enforced by the European Union since 2018 and the California Consumer Privacy Act [8], which got into effect in 2020.

In the specific context of neuroimaging, the quest to accurately predict brain age from structural Magnetic Resonance Imaging (MRI) scans has gained prominence, particularly for its potential applications as highlighted in [9]. The ability to predict an individual's brain age and compare it to their chronological age offers profound implications for the early detection and prognosis of age-related disorders, including Alzheimer's disease, schizophrenia, and bipolar disorder [10].

This process exemplifies the critical role of advanced ML models in bridging the gap between AI's potential and its practical utility in addressing complex medical challenges.

In addressing the highlighted challenges, this thesis adopts Federated Learning for brain age prediction, capitalizing on the joint efforts of various medical institutions to construct a unified model that ensures data privacy and security. This research aims to deliver a detailed exposition on Federated Learning, spotlighting its deployment, implications, and the obstacles it faces, especially within the healthcare landscape. The chapters that follow will investigate the core principles of Federated Learning, survey the existing technological frameworks and algorithms, address the specific challenges arising in this field, explore the utilization of Federated Learning for accurate brain age estimation, present the experiments conducted, discuss the results, and examine potential future research directions. Through this approach, the thesis endeavors to merge the theoretical advancements of Federated Learning with its practical applications in medical diagnostics, prognosis, and disease subtyping, underscoring its capacity to enhance predictive accuracy while safeguarding data privacy.

Chapter 2

Background

2.1 Federated Learning

Federated Learning (FL) [11, 12, 13, 14] has been introduced by Google in 2016. It was created with the intention of enabling more advanced models by allowing each participant to contribute to the training of a global model without having to compromise their data safety. Every participant starts by training a local model using their own available data. Then the models are sent to a global server where the local models are combined to create a global model. This global model is then sent back to every participant, where they will continue training the global model using their local data. This iterative process continues for a pre-determined amount of server rounds or until a desired level of accuracy is reached. The contents transmitted at each iteration varies depending on the specific model, typically encompassing weights or gradients. Given that the complexity of a model does not increase linearly with the volume of data, it is feasible to employ a substantial quantity of data for training a model. To summarize, there are five primary steps of federated learning: initializing the global model, selecting the clients, broadcasting the model, training the model using local data and model aggregation. Federated learning presents a multitude of benefits, notably in terms of data privacy and security. This approach eliminates the need for data exchange, thereby maintaining data within its original environment and diminishing the likelihood of any entity gaining unauthorized access to confidential information [15]. Federated learning facilitates collaborative model training among various entities, allowing them to benefit from a more sophisticated model than they could individually develop [16]. For instance, federated learning can be employed by commercial banks to identify instances of multi-party borrowing, a persistent challenge within the banking sector, particularly in the realm of online finance [17]. Yang et al. [18], Mammen [19] and Martínez Beltrán et al. [20] make the distinction between the following types of Federated Learning:

- Vertical Federated Learning: Employed when the different nodes operate on the same samples but have varying features.
- Horizontal Federated Learning : Utilized when the different nodes have the same feature space, but they contain separate data samples.
- Federated Transfer Learning: Designated for scenarios where the data across parties vary in both sample and feature spaces.

- **Cross-Silo Federated Learning:** Federated Learning with a small cadre of participants, each equipped with significant computational power and available for all rounds. This approach is particularly suited for collaborative efforts among organizations.
- **Cross-Device Federated Learning:** Federated Learning involving a multitude of participants on personal devices (such as cellphones, smartwatches, etc.) with small computational power. Due to their inherent unreliability, not all devices need to be present for every round.
- **Centralized Federated Learning:** This prevalent form of Federated Learning involves a central server that aggregates local models from various nodes and subsequently redistributes the updated model to them.
- **Semi-Decentralized Federated Learning:** In this federated architecture, the traditional role of a central aggregator is eliminated. Instead, a specific node, chosen by consensus among participating nodes, assumes the responsibilities typically managed by the central server.
- **Decentralized Federated Learning:** Characterized by the absence of a central aggregator, this architecture ensures a fully autonomous network. Nodes independently manage their communications and aggregate model parameters, fostering a decentralized learning environment.

It is important to recognize that these Federated Learning paradigms are not mutually exclusive and can exhibit overlapping characteristics in practice, allowing for a hybrid approach in their implementation.

2.1.1 Data-Aligned Federated Learning Paradigms

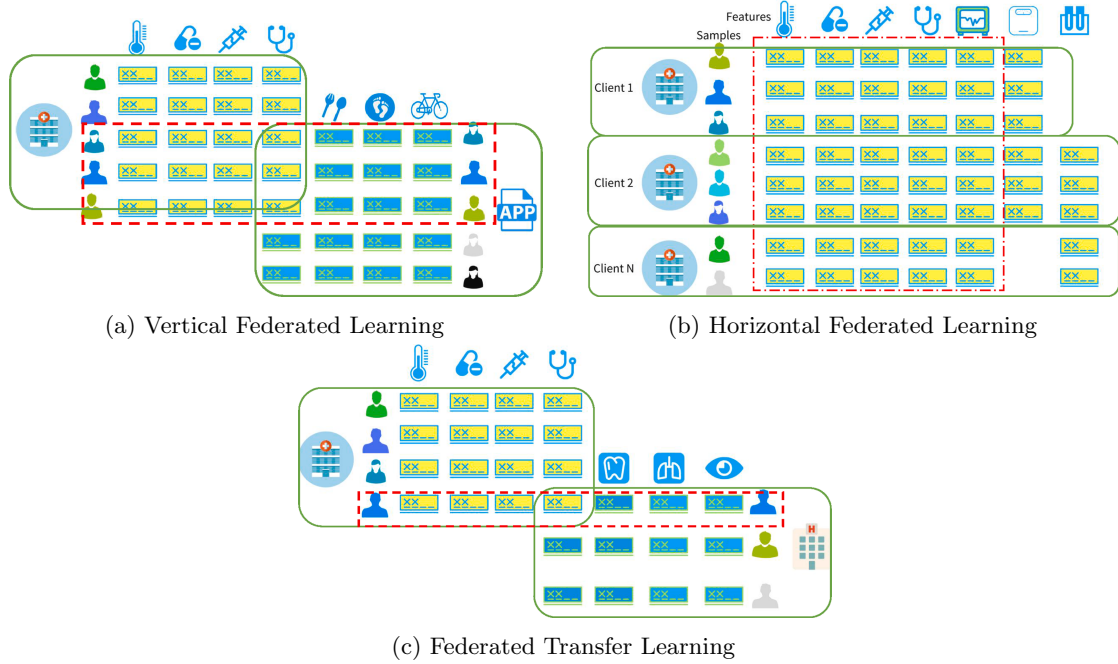


Figure 2.1: Data-Aligned Federated Learning Paradigms [21]

Horizontal Federated Learning (HFL) is employed in scenarios where various nodes possess

datasets with identical feature spaces but distinct data subjects. The term "horizontal" is derived from "horizontal partitioning" a concept prevalent in traditional database terminology. In this framework, particularly within a tabular database perspective, rows are segmented horizontally into various groups, with each row encompassing a full set of data features. Currently, this aspect represents the primary research focus within the field of Federated Learning due to its significant applicability across various business sectors. The initial, albeit unintentional, version of HFL was developed by Shokri and Shmatikov [13], before FL as a concept was formally established. In 2016, researchers at Google developed a HFL solution in order to process Android smartphone model updates [11], which was used to train the mobile keyboard prediction model on Android devices [22].

In subsequent sections, we will delve into the challenges, security considerations, and algorithms pertinent to Horizontal Federated Learning, as this constitutes the core focus of the thesis in exploring the application of Federated Learning in brain age prediction. To ensure a comprehensive background section, a concise overview of the other two data-aligned federated learning paradigms will also be provided. This approach ensures a thorough exploration of HFL while acknowledging the broader spectrum of federated learning methodologies.

Vertical Federated Learning (VFL) is deployed in scenarios where distinct nodes maintain datasets with divergent feature spaces, but overlap in terms of data subjects. The term "vertical" is inspired by the notion of "vertical partitioning" from conventional database parlance, emphasizing the columnar aspect. In this arrangement, particularly from the viewpoint of a tabular database, data columns are divided vertically, with each column representing a unique feature set across common data subjects. Vertical Federated Learning was first distinguished and proposed by Yang et al. [18]. VFL finds a significant application in enhancing Healthcare Recommendation systems by integrating features from both mobile network operators and healthcare providers [23]. Due to the considerable complexity involved in vertically integrating datasets, VFL is predominantly used within cross-silo contexts. The primary challenge within VFL lies in entity resolution [24], which involves accurately matching the same user's data across different datasets in a manner that safeguards user privacy. For further details regarding algorithms and security, we direct readers to the extensive study conducted by Liu et al. [25].

Federated Transfer Learning (FTL) is used in scenarios with minimal or no overlap in data and feature spaces. The integration of Federated Learning principles with transfer learning methodologies [26] helps develop effective and precise machine learning models for various businesses and applications with limited data, characterized by scarce overlapping samples and features, and insufficient supervision, indicated by limited labels, all while adhering to data privacy and security regulations [18, 25]. Research on the computational efficiency, deployability, and security of FTL remains limited. For more insights, refer to the study by Liu et al. [27], which introduces a framework for secure FTL.

2.1.2 Scalability-Oriented Federated Learning Paradigms

Cross-Device Federated Learning (CDFL) is primarily implemented across a broad spectrum of mobile and Internet-of-Things (IoT) [28] devices to conduct model training, characterized by its remarkable scale of parallelism, with the potential to engage up to 10^{10} devices. This extensive parallelism introduces significant challenges in communication and device reliability [29]. The devices, often limited by their computational capabilities and with sporadic availability for computation, raise critical issues regarding client reliability. To address these challenges and facilitate feasible training sessions, servers may employ strategies such as random node selection or algorithmic client selection for participation in training rounds [19].

CDFL contends with the heterogeneity of systems [30], where a substantial fraction of devices may fail during computations due to factors like battery depletion, network disruptions, or inactivity, leading to what is known as "stragglers". Managing these stragglers effectively remains an active area of research within Federated Learning. Moreover, the resource demands placed on both servers and clients by CDFL can pose challenges, particularly for client devices such as smartphones, which may experience diminished performance or battery life as a result. To mitigate these issues, servers might selectively recruit the most suitable nodes for participation. An illustrative strategy involves smartphones engaging in Federated Learning exclusively during nighttime charging periods, as highlighted by McMahan et al. [11]. While this approach promotes reliable training, it inadvertently introduces a time-dependent bias in the data, representing an undesirable outcome.

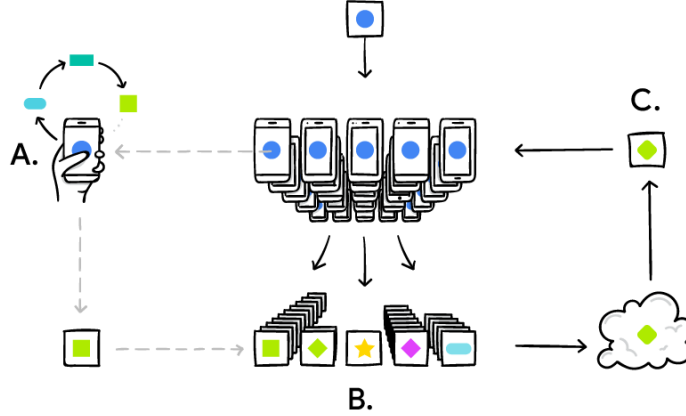


Figure 2.2: A single iteration of CDFL, using phones as participants. A phone trains a local model based on the user data (A), then each local model is aggregated in a central server (B) to form the global model (C). [31]

To address challenges associated with synchronous computations across unreliable devices, a method has been developed where nodes engage in learning only upon readiness, allowing them to submit updates at any computation stage while ensuring model convergence. This method enhances computational efficiency by eliminating the need for the server to await responses from slower clients before aggregating updates. This approach is known as Asynchronous Federated Learning (AFL).

Significant advancements in AFL have been made to address these challenges. Chen et al. [32] developed ASO-Fed, a novel framework wherein nodes engage only upon readiness, allowing them to asynchronously push model updates to the global server. It is important to note that the asynchronous nature of this learning model prevents its integration with synchronous privacy-enhancing techniques, such as Secure Multiparty Computation [33]. Asynchronous Federated Learning, particularly within the Cross-Device context, continues to be a fertile area for future research. For an extensive overview of the current developments in Asynchronous Federated Learning, the work of Xu et al. [34] is recommended.

Cross-Silo Federated Learning (CSFL), on the other hand, involves a comparatively smaller number of clients, typically ranging from 2 to 100. These clients are generally different organizations,

such as medical facilities or geographically distributed datacenters, which means that the data is both siloed and remains decentralized. The primary bottleneck in cross-silo federated learning can be either computation or communication, depending on the specific setup and network capabilities. Unlike in cross-device scenarios, clients in cross-silo federated learning are addressable and are expected to be more reliable and stateful, carrying the state from round to round of computation [29]. In a cross-silo environment, clients are expected to participate in every round of Federated Learning. This consistent participation allows the server to implement privacy-preserving techniques and reduce communication overhead. In specific sectors where CSFL is applicable, there may arise the challenge of model heterogeneity, which occurs when various clients develop their local models distinctively, influenced by their previous performance, environmental conditions, and resource availability. In the medical sector, CSFL has been used to collaboratively train a breast density classification model between 7 clinical institutions across the world [35].

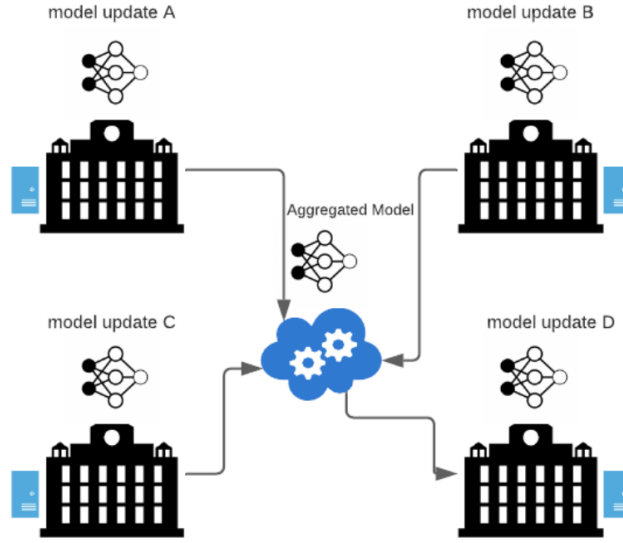


Figure 2.3: Cross-Silo Federated Learning, using institutions as participants [19]

It should be noted that although the experiments conducted in this thesis will take place assuming a cross-silo environment; discussions on heterogeneity from both cross-device and cross-silo viewpoints will be included in later sections, given their significance in existing literature. Addressing these issues and developing solutions contribute to the creation of models that are more resilient to heterogeneity, thereby benefiting the broader field of Federated Learning.

2.1.3 Network-Architecture-Based Federated Learning Paradigms

Centralized Federated Learning (CFL), initially introduced as the standard network structure in conjunction with Federated Learning in seminal works from McMahan et al. [11], Konečný et al. [12], Shokri and Shmatikov [13], and Konečný et al. [14], utilizes a central server, often referred to as the aggregator. In CFL, the central server functions as the hub in a hub-and-spoke topology. It acts as the coordinator for the federated learning process and alleviates client responsibilities: selecting clients for learning iterations when necessary, broadcasting the global

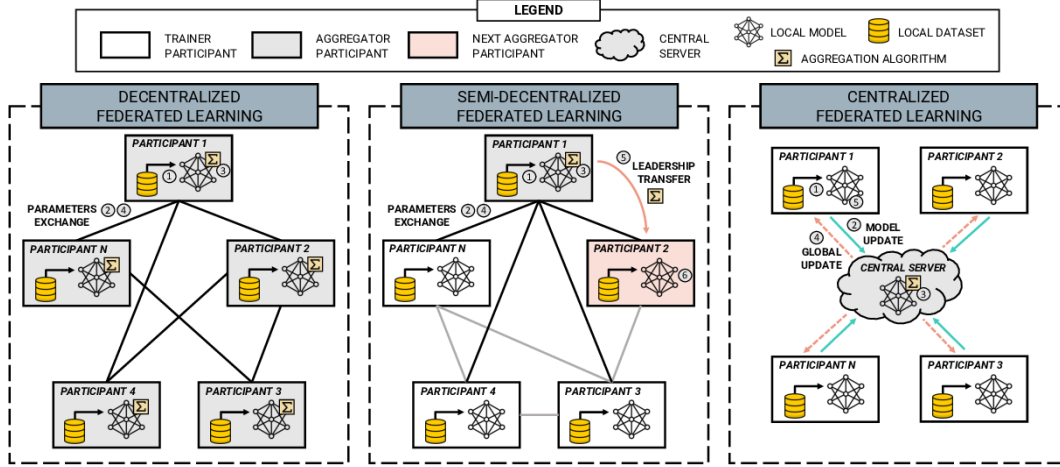


Figure 2.4: Network-Architecture-Based Federated Learning Paradigms [20]

model to the clients, collecting local updates from these clients, and incorporating these updates into the global model. This arrangement streamlines the communication and data exchange process, with the server handling the complexities of model updates and allowing client devices to focus on receiving the model and retraining it with local data. This arrangement enhances the cost-efficiency of Federated Learning by facilitating hardware server upgrades, such as adding more powerful processors, increasing memory capacity, and incorporating faster storage solutions. These improvements enable the servers to process data more quickly and efficiently, thereby accelerating the aggregation process. Particularly in Cross-Device Federated Learning, where numerous devices with varying computational capacities are involved, this centralized approach proves immensely beneficial. Additionally, centralizing this responsibility offers the flexibility to dynamically adjust how updates are processed. While most aggregation algorithms detailed in subsequent sections are designed for centralized networks, they can be adapted for other architectures with modifications. Despite these benefits, a centralized framework can introduce latency due to potential bottlenecks, increased susceptibility to system failures and concerns about the trustworthiness of the entity overseeing the aggregator. The designation of a central server may also give rise to various issues related to cost, bias and exposure to security risks, including but not limited to, man-in-the-middle and denial-of-service (DoS) attacks. Moreover, the selection of a central point in CFL can raise questions about fairness, as it may inadvertently favor certain clients over others based on their proximity to the server or their data's alignment with the global model's objectives. This can lead to imbalances in the contributions and benefits among participating clients, potentially skewing the model's development and its applicability across diverse datasets. Such disparities underscore the need for equitable strategies in the selection and operation of central servers to ensure that Federated Learning remains inclusive and fair for all participants.

Semi-Decentralized Federated Learning (SDFL) addresses the challenges identified in centralized models by distributing the computation and responsibilities of an aggregator server among the nodes. This system involves a periodic selection of a new aggregator from the nodes, a process that can be random or based on criteria like network speed, computational power, or energy capacity [36, 37]. The dynamic nature of the aggregator role enhances security by making it more challenging for adversaries to target a fixed central server, ensures more equitable latency

distribution by providing each node the opportunity to experience minimal network delay, and eliminates the need for a dedicated external server for aggregation tasks. However, this approach requires that all participating clients be capable of handling the demanding functions associated with the aggregator, thereby restricting its applicability predominantly to Cross-Silo Federated Learning scenarios. Additionally, future works can explore how often nodes are chosen as aggregators and the potential implementation of incentives to encourage nodes to take on this central role. The details of these incentive mechanisms will be addressed in Section 2.5.1.

Decentralized Federated Learning (DFL) presents a fully autonomous network structure where participants independently engage in local model training and subsequent parameter exchanges. This setup facilitates direct, often bidirectional communication between nodes, allowing for a highly flexible and self-sufficient system [20, 38]. Unlike centralized architectures, DFL does not rely on a singular server for parameter aggregation, thereby enabling asynchronous model updates and mitigating the constraints of synchronized operations. The network's topology, which can be either fixed or dynamically adjusted, plays a crucial role in the efficiency and stability of the learning process. While a fixed topology offers consistency and reliability, a dynamic arrangement introduces adaptability at the cost of increased management overhead. The choice between directed and undirected graphs further influences the nature of node interactions, with directed graphs accommodating asymmetric relationships and undirected graphs simplifying the modeling of symmetric connections. This decentralized approach inherently allows for greater scalability and resilience, avoiding single points of failure and facilitating robustness in the face of network changes and disruptions. While DFL offers significant autonomy and flexibility, it also encounters distinct challenges. The lack of a centralized aggregator can lead to increased complexity in coordinating model updates across the network, potentially resulting in inconsistencies and delays. This decentralized nature requires sophisticated algorithms to ensure convergence and maintain model quality, which can be computationally demanding for individual nodes. Additionally, the reliance on peer-to-peer communication raises concerns about network reliability and efficiency, as nodes must manage data exchanges without a central point to optimize traffic. The dynamic topology, although adaptable, introduces the need for continuous management of network connections, which can incur additional overhead and complicate the learning process. Furthermore, in DFL, ensuring data privacy and security becomes more challenging, as the decentralized framework broadens the attack surface, requiring advanced cryptographic techniques and trust mechanisms to safeguard communications and data exchanges. These disadvantages highlight the trade-offs involved in adopting a decentralized approach, necessitating careful consideration of the network's design and the implementation of robust protocols to mitigate potential issues.

2.1.4 Blockchain-Enabled Federated Learning

Decentralized Federated Learning has garnered significant attention, particularly through the integration of blockchain technologies [39, 38]. Blockchain, a decentralized digital ledger, ensures secure transaction recording across multiple computers [40]. Its most notable application is in Bitcoin 1.0, which facilitates decentralized peer-to-peer transactions without centralized oversight [41]. Another innovative application of blockchain technology is in Smart Contracts, which enable the execution of Turing-computable functions within the blockchain [42]. These contracts allow for the replacement of traditional aggregators in FL systems with blockchain-based mechanisms, thus facilitating the aggregation and redistribution of the global model. This approach essentially constitutes a form of SDFL, where the aggregator's responsibilities are managed through blockchain consensus algorithms.

Incorporating FL into blockchain offers numerous benefits, including enhanced authentication,

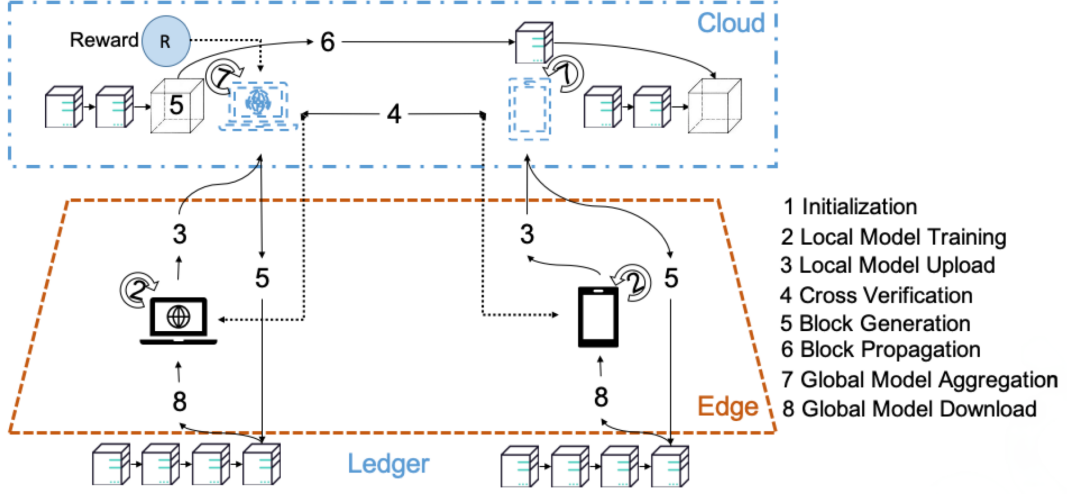


Figure 2.5: A generalized blockchain-based FL paradigm [39]

traceability, availability, scalability, decentralization, resilience to Byzantine faults, data persistence, and anonymity [43, 40, 44]. A critical and often neglected aspect of FL that blockchain addresses is the incentive mechanism, rewarding users or miners for their contributions. The integration of blockchain technology significantly enhances federated learning by incorporating both incentive mechanisms and membership selection. It rewards users or miners for their contributions by design and ensures only nodes with high-quality data and a commitment to system integrity are included. Although such mechanisms are crucial in cross-device architectures, they also hold value in cross-silo contexts. For a comprehensive understanding of the integration of FL and blockchain technology, please refer to the review study conducted by Wang and Hu [45] on blockchain-based approaches.

However, integrating blockchain with Federated Learning introduces certain challenges, particularly an increase in computational overhead. The issue of computational overhead already exists in conventional Federated Learning setups and should not be aggravated further. For instance, the Proof-of-Work (PoW) consensus algorithm, despite its security advantages, significantly increases computational demands. Another problem pertains to security; public blockchain data are accessible to anyone, raising privacy concerns [46].

To mitigate the aforementioned issues, a hybrid blockchain model has been suggested, where only the SHA256 hash of local model updates is recorded on a public blockchain, ensuring consistency while offloading intensive computations to a private blockchain [47]. Storing the SHA256 hash of local model updates on a public blockchain upholds security by providing a tamper-proof record that verifies the integrity and order of the updates. This public record ensures that any alteration of the updates would be easily detectable, as the hashes would no longer match. Consequently, this approach maintains transparency and trust without exposing the actual data, which remains on the private blockchain. The private blockchain handles the bulk of the computations, ensuring that sensitive data and intensive processes are kept secure and efficient. By separating the roles of the public and private blockchains in this manner, the model achieves a balance between transparency, security, and computational efficiency.

Another solution to the problem of additional computational overhead involves leveraging the computational effort required for training local models as a criterion for selecting the next aggregator node. This method, known as Proof-of-Federated-Learning (PoFL) [48], aims to balance security and efficiency within the blockchain-enabled Federated Learning framework.

In the subsequent sections of this thesis, our discussion will center on aggregation algorithms within the framework of networks that utilize a central server, as seen in centralized or semi-decentralized architectures. The consideration of decentralized aggregation algorithms will not be pertinent to this work, given that our experimental setup is situated within cross-silo contexts. In these environments, nodes are capable of either reaching consensus on a central server or possess sufficient resources to undertake central responsibilities themselves, thus diminishing the relevance of Decentralized Federated Learning advantages for our study.

2.1.5 Comparison with Distributed Machine Learning

Distributed Machine Learning (DML) [49] is considered the predecessor of Federated Learning, naturally leading to the presence of multiple shared characteristics between the two methodologies. Both frameworks involve the distribution of data across multiple nodes, with the primary distinction being the nature of the data distribution. Federated Learning operates in a decentralized manner, with the data distributed across nodes, each training its own model. DML was motivated either by the need to scale up the training process or by the desire to mitigate privacy concerns associated with centralized data repositories, the latter being a foundational principle for Federated Learning.

However, the primary distinction between the two paradigms lies in the nature of the data distribution. Scaling up the training process is primarily motivated by the expansive complexity of contemporary Deep Learning models. The training of these models is computationally intensive and may require a significant amount of time. Therefore, the machine learning process undergoes parallelization, where one or more aspects are distributed among worker nodes. The Data Parallelism aspect of DML closely aligns with Federated Learning in that it entails the distribution of data across multiple nodes, with each node independently training its own model [50]. Federated Learning treats data as decentralized and private, with each node independently training its model. In contrast, DML typically centralizes data, assigning data segments to nodes for collaborative model training. Although these methodologies share common ground, they each present unique challenges and opportunities.

The privacy of node-specific data in Federated Learning restricts insights into overall data distribution, compelling FL algorithms to inherently address non-independent and identically distributed (non-IID) data. DML, on the other hand, can operate under more predictable assumptions about data distribution, such as its skewness. The availability of data on a global scale in DML facilitates a more efficient redistribution. For instance, by allocating more resource-intensive tasks to better-equipped nodes, a flexibility that is not present in Federated Learning due to the intrinsic data and resource constraints of each node. Global data access in DML also enables the application of advanced conventional machine learning methodologies, including data preprocessing, feature engineering, and enhanced privacy protocols like comprehensive differential privacy, alongside more sophisticated optimization strategies. Nevertheless, both Federated Learning and DML grapple with the challenge of communication overhead, an area where future advancements could yield mutual benefits.

2.2 Challenges in FL

In this section, we will discuss the obstacles confronting Federated Learning. While these challenges are not exclusive to this domain, their complexity is magnified by the distributed structure of the learning mechanism. Preliminary discussions have already shed light on the difficulties encountered by Federated Learning in cross-device environments, with a particular focus on the diversity of systems. The narrative will now advance to a detailed exploration of the core challenges Federated Learning encounters within cross-silo configurations.

2.2.1 Model Security

A challenge within Federated Learning pertains to the security of the process. Theoretically, Federated Learning is designed to facilitate the exchange of training models without compromising data security, as the private data is never exchanged. However, the practical application reveals vulnerabilities; the model itself may inadvertently reveal information about the underlying data. It is crucial to ensure that the model's weights or gradients, exchanged during the process, remain secure from potential adversaries. Additionally, the training data and the learning process are susceptible to various forms of attacks, including the risk of being hijacked. Moreover, participants with honest-but-curious intentions pose a threat by potentially accessing more information than is deemed secure. This issue will be explored in greater depth in Section 2.3, dedicated to privacy and security.

2.2.2 Communication Overhead

Communication overhead presents a significant challenge to Federated Learning, as the process inherently involves transmitting models across the internet, often multiple times, until the global model reaches an acceptable standard. Particularly in cross-device contexts, where devices commonly operate on household networks characterized by asymmetric download and upload speeds, a notable performance bottleneck arises [51]. This issue is less pertinent in cross-silo scenarios, given the assumption that participants are equipped with advanced tools to minimize communication overhead. The extent of this overhead is influenced by several factors, including the network architecture. The assumption of node reliability can lead to delays in the learning process caused by less efficient nodes. To address these challenges, nodes may be excluded from the learning process if they do not complete their model update within a specified timeout. Asynchronous Federated Learning offers a mitigation strategy by allowing clients to transmit their model updates at their convenience. Decentralized Federated Learning, on the other hand, eliminates the need for a central authority and adopts a slower, distributed communication approach among nodes. Although these methods provide innovative solutions to communication overhead, they introduce additional complexities, such as issues related to convergence and data security.

Initiatives to minimize the volume of data transmitted have been undertaken, including the adoption of sparsification and quantization strategies. These methods effectively reduce the size of model updates prior to transmission. Notably, these approaches yield substantial decreases in communication overhead while maintaining the accuracy and efficacy of the federated learning process [14]. There are currently 3 methods available to further reduce the communication overhead, as stated by Kairouz et al. [29]:

1. Gradient Compression: reducing the size of the data communicated to the global server. Given the variability in reliability of edge devices and the general tendency for slower upload

speeds compared to download speeds on these devices, it is in this context that the greatest potential for improvement exists.

2. Model Broadcast Compression: reducing the size of the model broadcasted to all the participating nodes.
3. Local computation reduction: enhancing the computational efficiency of the local training procedure.

Compression techniques leveraging error correction are more effective in stateful cross-silo environments, where both clients and servers maintain state, allowing sophisticated algorithms to reduce data transmission size, minimize communication overhead, and enhance model accuracy and convergence [52, 53, 54, 55, 56]. Additionally, it is presumed that enhancing the central server’s performance is not imperative, as it does not constitute the primary source of bottlenecks within the system. The communication overhead is influenced not solely by the model’s size but also by the requirements of the selected aggregation algorithm and the implemented privacy safeguards. Thus, optimizing in this area necessitates a balance between acceptable performance levels and required privacy standards. Another innovative strategy for minimizing communication overhead involves decreasing the number of communication rounds. This approach was exemplified by Li, He, and Song [57], who developed an aggregation algorithm capable of converging within a single communication round, representing a significant advancement in this field.

2.2.3 Heterogeneity

In the context of federated learning, the challenge of heterogeneity in data is multifaceted, often leading to Non-Independent and Identically Distributed (non-IID) data among clients participating in the model training process. The heterogeneity of data can be broadly classified into two categories: statistical heterogeneity and structural heterogeneity, as elucidated by Li et al. [21].

Statistical heterogeneity addresses the discrepancies in data distributions across various clients. Traditional machine learning models operate under the assumption of Identically Independent Distribution (IID), where each data point is assumed to be drawn independently from the same distribution. However, in federated learning environments, this assumption does not hold due to a variety of factors that influence data collection, such as the type of device used, the context of data usage, time of data entry, and other external variables. These factors can lead to skewed data distributions that are unrepresentative of the global population, resulting in biased models with poor performance on underrepresented data segments. This skewness can arise from covariate shift, where the input distribution varies across clients even if the output remains consistent, from label distribution shift, where the same label might be interpreted differently across clients, or from quantity skew, where data contribution varies significantly among clients, potentially biasing the model towards larger data contributors.

Moreover, data independence violations can occur when client data exhibit temporal or spatial correlations, undermining the IID assumption. For instance, smartphone data might exhibit temporal patterns, leading to models biased towards specific usage times, or spatial patterns, leading to models biased towards certain locations.

Structural heterogeneity, conversely, relates to the variability in client participation and reliability within the FL process. Factors such as unstable connections or insufficient computational capabilities can hinder consistent client participation in training rounds, necessitating FL systems

to adapt to these structural changes by designing algorithms that accommodate the potential stateless nature of clients, thereby effectively managing non-IID data.

These challenges are central to the forthcoming discussion in Section 2.4, where we will explore viable strategies for addressing heterogeneity in Federated Learning.

2.3 Privacy & Security in FL

An important concern in the Federated Learning domain is how secure the procedure is. Any possible threats or attacks on the data, the model or the model aggregation can be disastrous. Guarantees must be given to all parties about how safe the system is before they participate in it.

2.3.1 Regulations surrounding Data Protection

Regulatory laws surrounding data privacy are crucial because they establish a framework for safeguarding personal information and ensuring responsible data management. The GDPR has been widely recognized as the benchmark for data protection standards, transcending its original jurisdictional boundaries. This broad adoption is evident even in nations where the regulation is not legally enforced, primarily due to its implementation across the European Union. Therefore, we consider a system safe if it can unambiguously comply with the GDPR. The GDPR outlines 6 principles of data protection (article 5.1 of the GDPR):

1. Fairness and transparency
2. Purpose limitation
3. Data minimization
4. Accuracy
5. Storage limitation
6. Integrity and confidentiality



Figure 2.6: GDPR: EU regulation on data protection [18]

The principles of purpose limitation, data minimization, and storage limitation are inherently satisfied through the operational framework of Federated Learning on the client side. On the server side, this is accomplished by typically involving the storage of minimal or no raw data, with the central server only aggregating and processing model updates. This approach ensures

adherence to the principle of storage limitation while maintaining the integrity and privacy of the client data. The efficiencies in communication rounds and the acceleration of the learning process necessitate the observance of purpose limitation and data minimization, further enhancing the overall effectiveness and security of the Federated Learning framework. The present thesis does not address data accuracy, under the assumption that all provided data is accurate and intentional. The challenge of transparency in Deep Learning models is recognized, particularly in relation to compliance with GDPR. While advancements in more interpretable models or potential modifications in GDPR transparency requirements are anticipated [58], these considerations are beyond the scope of this study.

Fairness entails that the data used from an individual is used in a way that is fair to the individual. This principle is not inherently satisfied by Federated Learning and methods for introducing fairness and solving bias will be discussed in Section 2.5. The principles of integrity and confidentiality necessitate that data be safeguarded against unauthorized access and processing. A scoping review conducted by Brauneck et al. [59] examined Federated Learning within the context of Data Protection legislation, concluding that Federated Learning alone is insufficient to meet data privacy mandates. The study asserts that additional security protocols, such as Secure Multiparty Computation and Differential Privacy, are essential to mitigate associated security vulnerabilities. Federated Learning systems must be designed to ensure the integrity of the data and the learning process. In the following subsections, we will explore various potential threats and attacks targeting Federated Learning frameworks, examining their implications and the measures that can be taken to mitigate these vulnerabilities.

2.3.2 Threat Models

Before we can assert what security measures are necessary, we must first identify the potential threats to the system. We inspect all the possible threat models in this subsection.

Outsider

Outsider attacks encompass a range of malicious activities that can be executed without direct involvement in the Federated Learning (FL) process. Such attacks are predominantly feasible in scenarios where gaining entry into the FL system is challenging. The primary objective of these attacks is to disrupt the communication channels linking clients to the central server. It is pertinent to note that executing these attacks becomes increasingly difficult in semi-decentralized or fully decentralized FL architectures. The following outsider attacks are identified as principal threats to FL systems:

- **Man-In-The-Middle (MITM) Attack:** These type of attacks involves an adversary intercepting the communication between two parties and impersonating one of the parties to the other [60]. This attack can be carried out during federated learning by intercepting the communication between the clients and the central server. The adversary can then impersonate the central server to the clients and vice versa. This would allow the adversary to obtain the model updates from the clients and send them a malicious global model while also inferring sensitive training data from the updates. Research by Zhu, Liu, and Han [61] has demonstrated the feasibility of an attacker stealing training data from gradient updates within a limited number of iterations. An example of this attack can be seen in Figure 2.7a.
- **Denial-of-Service (DoS) Attack:** This attack aims to obstruct the communication between entities by inundating the channel with superfluous requests [62, 63]. In the realm of FL, an adversary could execute this by overwhelming the communication pathway between

clients and the central server, thereby impeding the exchange of model updates and the redistribution of the updated global model. This disruption can effectively incapacitate the FL process. An illustration of this attack is provided in Figure 2.7b.

- **Software-based attacks:** As evidenced in instances involving Android systems [64], these attacks exploit vulnerabilities within the software utilized by either clients or the central server. Such vulnerabilities can provide adversaries with the means to launch attacks akin to those aforementioned, further compromising the integrity and confidentiality of the FL system.

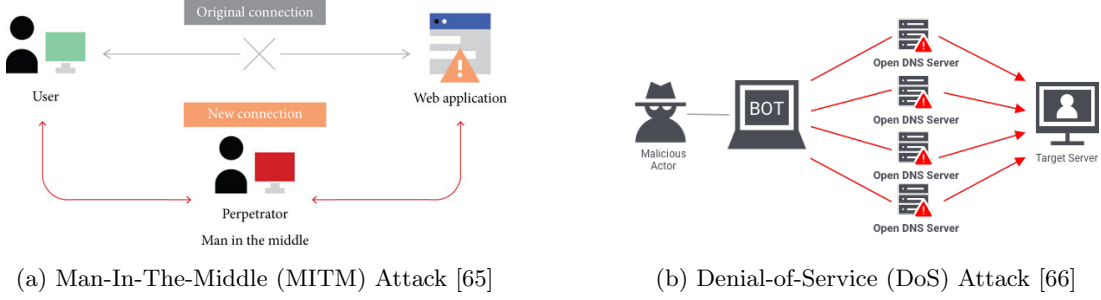


Figure 2.7: Illustrations of Network Security Attacks

Insider

Insider attacks describe malicious activities executed by participants within the FL system. These attacks are particularly challenging to detect, as the perpetrators are often authorized users with access to the system. The following insider attacks are identified as principal threats to FL systems [67]:

- **Single Attack:** This scenario involves a solitary malicious participant, acting independently, with the objective to deliberately misguide the model into erroneous classification of selected inputs with notable confidence [68, 69].
- **Byzantine Attack:** In this form of attack, malicious participants, known as Byzantines, may exhibit entirely unpredictable behavior. They meticulously craft their outputs to mimic the distribution of legitimate model updates, thereby complicating detection [70].
- **Sybil Attack:** This attack is characterized by adversaries creating multiple fictitious participant accounts or exploiting previously compromised participants to orchestrate more potent attacks on the FL system [71, 69].

Semi-Honest and Malicious

The majority of privacy-enhancing techniques and algorithms developed for Federated Learning are predicated on the assumption that all clients and the server adhere to an honest-but-curious threat model. Under this paradigm, while all participants are expected to comply with the established protocol, they will attempt to deduce information about fellow participants from the data received. In FL, two types of passive adversaries can occur: a passive client, who is presumed to only access the global model, and a passive server, which may seek to extract additional insights from the updates submitted by clients. This scenario is encapsulated within the semi-honest

threat model [72], predicated on the rationale that participants are motivated to abide by the protocol, which is inherently designed to mitigate the risk of data leakage.

However, it is important to note that this assumption is not always valid. In certain instances, participants might deviate from the prescribed protocol and attempt to gain access to the private information of other honest participants. This scenario is encapsulated within the malicious threat model, which is more realistic, given the difficulty in ensuring that all participants adhere to the protocol. Moreover, the malicious threat model is more dangerous, as adversaries can potentially execute more devastating attacks on the learning process, particularly if the adversary is the server. The malicious threat model is also more challenging to address, as it necessitates the implementation of more sophisticated cryptographic techniques and protocols.

Training and Inference Phase

Training phase attacks in Federated Learning (FL) target the learning process itself, aiming to either compromise the model's accuracy or embed a backdoor within it [73]. Such attacks are predominantly initiated prior to the stabilization of the global model. In contrast, inference phase attacks occur after the development of the global model and are not designed to modify the model [74]. Instead, these attacks seek to extract information from the model or to exploit its vulnerabilities to produce erroneous outputs. Typically, these attacks are conducted by adversaries who lack access to the FL training phase, and their efficacy varies based on the adversary's knowledge of the model.

The less severe form of these attacks is termed a black-box attack, wherein the adversary has no prior knowledge of the model's internal workings and can only infer information through the model's outputs. On the other hand, a white-box attack represents a more severe threat, with the adversary possessing comprehensive knowledge of the model. This type of attack is particularly perilous in FL, as all global models, including the finalized version, are disseminated among clients. Therefore, ensuring the privacy of communication channels to prevent potential leaks of these models to adversaries is imperative.

2.3.3 Poisoning & Inference Attacks

Poisoning Attacks

Poisoning attacks are a form of training phase attack, wherein adversaries seek to compromise the integrity of the global model by injecting malicious data into the training process. Broadly, these attacks can be indiscriminate, merely seeking to degrade the model's overall accuracy, or they can be targeted, intending to manipulate the model into consistently misclassifying inputs towards a specific, adversary-chosen label [75]. Indiscriminate attacks, also known as convergence attacks, can typically be identified through diminished model accuracy on validation datasets. Conversely, targeted poisoning attacks, often referred to as backdoor attacks, pose a more substantial threat by embedding biases or backdoors within the model, albeit at the cost of increased execution complexity.

A prevalent method for executing backdoor attacks involves the use of trojans [76, 77]. In this context, a trojan is a meticulously designed pattern that triggers specific misclassifications by the model. For instance, in a system estimating house prices based on various features, an adversary might engineer a trojan to cause the system to undervalue houses equipped with swimming pools. By incorporating this trojan into the training data, the adversary could manipulate the system to consistently undervalue such properties. This scenario could arise in competitive environments

where companies engaged in the same FL project might attempt to compromise the model for strategic advantage.

Poisoning attacks are carried out in two forms: data poisoning or model poisoning. Data poisoning entails the introduction of maliciously altered data into the training set, whereas model poisoning involves the injection of harmful gradients. Data poisoning is typically effected through the manipulation of data labels, assuming adversary access to the training dataset [78]. Nonetheless, research by Shafahi et al. [79] demonstrates that adversaries can still orchestrate effective poisoning attacks without altering labels, by embedding watermarks in benign-looking data that is then inadvertently incorporated into the training set by data collection mechanisms. Although such label manipulations may not be feasible in sensitive fields like healthcare, it remains crucial to demonstrate model resilience against these attacks.

Model poisoning, on the other hand, operates under a different premise, where adversaries manipulate the model directly through the training process rather than the data it learns from. In model poisoning, attackers introduce malicious updates that, when aggregated with benign updates from other participants, subtly shift the model's behavior. This could involve the insertion of hidden backdoors [69]. These backdoors can allow malicious actors to manipulate the model's behavior subtly, causing it to produce incorrect outputs for specific inputs while remaining undetected during normal operation. The subtlety of model poisoning lies in its ability to remain undetected by blending malicious updates with legitimate ones, making it a significant challenge for FL systems. The alternating minimization strategy, as discussed by Bhagoji et al. [68], exemplifies the sophistication of such attacks. Here, adversaries alternate between optimizing for legitimate training objectives and their malicious goals, making the poisoned updates harder to identify amidst the noise of legitimate learning. Furthermore, the use of parameter estimation techniques to mimic benign updates adds another layer of stealth, complicating detection efforts.

Model poisoning subsumes data poisoning in FL settings, as data poisoning attacks eventually change a subset of updates sent to the model at any given iteration [71]. This is functionally identical to a centralized poisoning attack in which a subset of the whole training data is poisoned. Model poisoning attacks require sophisticated technical capabilities and high computational resources. Such attacks are generally less suitable for cross-device scenarios, but more likely to happen in cross-silo scenarios.

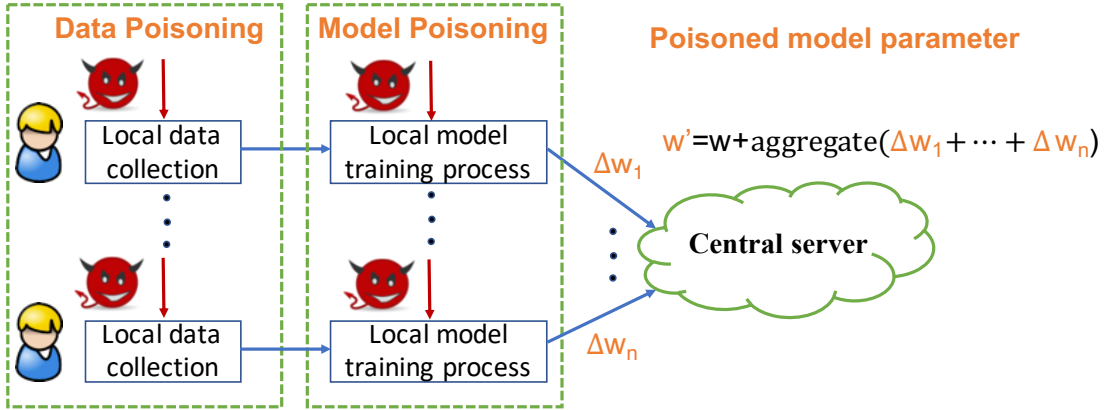


Figure 2.8: Data & Model poisoning attacks [67]

Inference Attacks

To comply with data protection laws, the data can only be used to achieve the model’s objective without revealing any additional information. Unfortunately, exchanging gradients during Federated Learning (FL) training can lead to serious privacy leakage [61, 80, 81, 82]. The goal of an inference attack is to infer properties of the local datasets. Observations of model updates can be used to infer class representatives, membership of a data point, properties of training datasets, and even training inputs with their labels [67]. In the worst case, an attacker can recover the original training data without requiring any prior information about the training data [61].

The work from Hitaj, Ateniese, and Perez-Cruz [83] has demonstrated that the FL process is susceptible to a unique form of active inference attack utilizing Generative Adversarial Networks (GANs). This GAN attack is particularly insidious because it allows a malicious participant within the FL system to generate synthetic samples that are representative of the private training data of other participants. These samples are not direct reconstructions of the actual inputs but are instead designed to capture the overall characteristics of the data belonging to a specific class. However, this attack may be limited in its effectiveness, as the synthetic samples generated by the GAN may not be representative of the actual data distribution and the GAN assumes that all training data for a class originates from a single participant. This assumption is invalid in both cross-device and cross-silo scenarios. Especially within the cross-device context, the computational demands associated with training and deploying GANs are likely to exceed the capabilities of most participants.

Membership inference is an interesting avenue in the domain of Federated Learning. An attacker can infer whether a specific data point is part of a participant’s private dataset by examining a singular model update from a node, or from any participant by examining the aggregated global model. These types of attacks can be carried out in either a passive or active manner [84, 82]. Passive attackers will not interfere with the FL process, they will just try to infer without changing their local or the aggregated model. This state of attackers can be classified as honest-but-curious and are part of the semi-honest threat model. Active attackers will tamper with the FL process to carry out more powerful attacks. Nasr, Shokri, and Houmansadr [84] demonstrated in their work that malicious model updates can be sent to the client to force the received global model to share more information about the local data of the participants. This attack is known as a gradient ascent attack, and it exploits the fact that SGD optimization updates model parameters in the opposite direction of the gradient of the loss.

Property attacks aim to infer characteristics from other participants’ data that are not tied to specific attributes [82]. It is assumed the adversary possesses auxiliary training data with accurate labels. A passive adversary might train a binary property classifier using model updates, whereas an active adversary could employ multi-task learning to coerce the FL model into distinguishing more effectively between data with and without the property, thereby gleaning additional information. An adversarial participant might also determine when a property emerges or vanishes in the dataset during training. Similar to the GAN attack, this approach is expected to be less viable in cross-device situations due to the extensive computational effort involved in training and deploying multiple models.

Research by Zhu, Liu, and Han [61] indicates that an adversary can ascertain the training inputs and labels of other participants through model updates. This form of attack poses a substantial risk as it can expose the training data of others, directly compromising participant privacy. Inference attacks necessitate advanced technical skills, significant computational resources, and involvement in numerous FL cycles. Such attacks are generally impractical for cross-device contexts but

are more plausible in cross-silo settings. However, considering that cross-silo scenarios typically involve thoroughly vetted participants, it is presumed that we are dealing with passive adversaries who do not engage in training additional models for inference attacks.

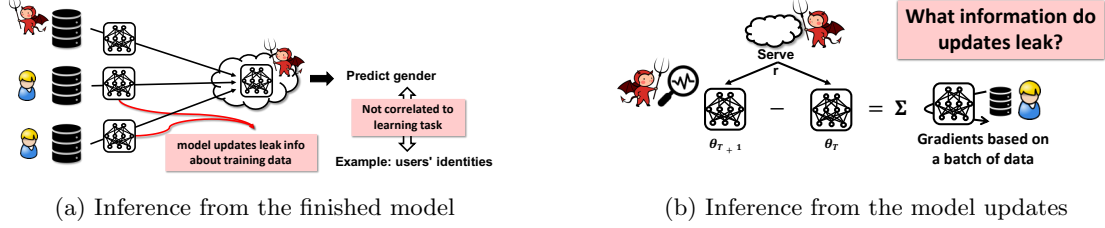


Figure 2.9: Inference Attacks [67]

2.3.4 Privacy-Preserving Techniques

Privacy-preserving techniques have emerged as pivotal mechanisms to address the inherent privacy challenges in collaborative computational frameworks, particularly within the context of Federated Learning. These methodologies are designed to enable the decentralized calculations, where the input remains distributed across a multitude of devices or institutions, thereby obviating the need for direct access to sensitive or proprietary information. As FL seeks to harness the collective power of data while safeguarding privacy and complying with regulatory mandates, the integration of privacy-preserving mechanisms is indispensable. This section elucidates four principal privacy-preserving techniques that are instrumental in reinforcing the privacy and security of FL processes: Secure Multiparty Computation (MPC), Differential Privacy (DP), Homomorphic Encryption (HE) and Trusted Execution Environments (TEEs). Each of these techniques contributes uniquely to the privacy-preserving capabilities of FL, ranging from the secure computation of aggregated data inputs without revealing individual contributions, to the obfuscation of data to prevent the deduction of individual participants' information, and the provision of secure computation environments to protect data during processing. Collectively, these techniques enable FL to navigate the complex balance between leveraging distributed data for model training and upholding stringent privacy and security standards.

Secure Multiparty Computation

Secure Multiparty Computation (MPC) was first proposed by Yao [33]. The referenced paper introduced a broad concept of secure computation, involving "m" parties collaboratively computing a function $f(x_1, x_2, \dots, x_m)$, where x_i represents the private input of the i^{th} party. According to Evans, Kolesnikov, Rosulek, et al. [85], Yao introduced the Garbled Circuits Protocol through a series of talks over the following years, although these were not included in any formal publication. We now present a simple example of MPC usage and then extend this example to Yao's Garbled Circuits algorithm.

Alice and Bob wish to determine who is older without revealing their actual ages. The goal is to set up a system where Alice sends questions to Bob, and he responds in a way that does not reveal his specific answers. The final output is simply the information about who is older. Alice designs a boolean circuit for the age comparison operation, taking two binary inputs (representing ages) and outputting 1 if Alice is older and 0 if Bob is older. She then garbles the circuit, encrypting each gate in the circuit and permutating the truth tables. For each binary digit of the age, Alice creates two pairs of keys, one for 0 and another for 1. Bob needs to choose keys corresponding to

his age without revealing his age to Alice. For each binary digit of his age, Bob and Alice engage in a 1-out-of-2 Oblivious Transfer (OT). Alice inputs two keys, and Bob secretly selects the key corresponding to his age's binary digit. Alice does not learn which key Bob chose. Bob uses the keys obtained through OT to evaluate the garbled circuit. He decrypts the gates corresponding to his inputs and obtains the output of the circuit. Bob learns the final result and shares this result with Alice. Both parties do not learn anything about the other's specific ages. This algorithm is secure against Alice because Bob's age bits are not revealed, and the OT prevents Alice from knowing which keys Bob picked and secure against Bob because he receives only keys corresponding to his age and the final output, learning nothing about Alice's age. This is an example of Boolean MPC with two-parties, as proposed by Yao [33].

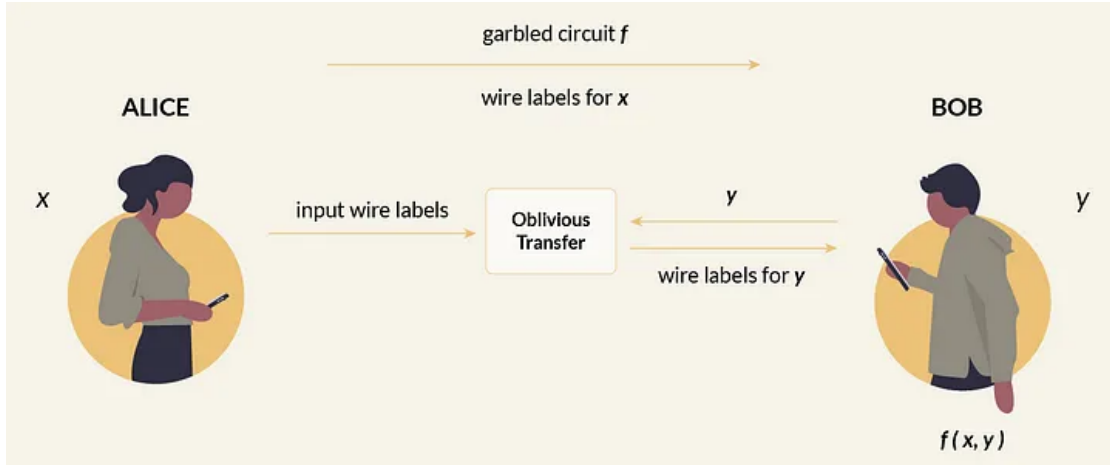


Figure 2.10: Yao's Garbled Circuit algorithm for Boolean MPC between two parties [86]

Multi-Party Computation has developed to a stage where it can be employed in scenarios involving more than two parties and it can evaluate arithmetic operations, specifically addition and multiplication. Additionally, various security parameters have been explored, encompassing the assessment of adversary capabilities (ranging from semi-honest/honest-but-curious to malicious and covert) and strategies for managing corrupt parties, as detailed in the papers written by Lindell [62] and Orlandi [87].

MPC has various use cases such as secure auctions, voting and secure machine learning. Although this domain has been mostly theoretical since its inception, we started seeing various real-world instances of MPC being utilized to realize a goal. Widely considered the first ever commercial use of MPC, Denmark developed an auction and bidding platform for sugar beet production contracts [88]. Several frameworks have been developed to write general algorithms leveraging the privacy guarantees of MPC, such as Fairplay [89] and Sharemind [90].

Attempts to incorporate Multi-Party Computation (MPC) into the healthcare sector have been undertaken. Smajlović et al. [91] introduced Sequare, a framework designed for developing high-efficiency algorithms supported by MPC. This framework guarantees data security and facilitates the sharing of biomedical data. Additionally, Senanayake et al. [92] developed NeuroCrypt, a system aimed at training machine learning models on distributed datasets, specifically for MRI brain scans. NeuroCrypt utilizes PySyft to merge Federated Learning with MPC, thereby achieving its objectives (more on FL frameworks in Section 2.6).

Federated learning has seen significant advancements through the incorporation of secure multi-party computation, which serves to mitigate the risk of model exposure amongst participating parties. This synergy is vital as machine learning models, if compromised, can potentially reveal sensitive information about the training data. Despite numerous improvements in MPC implementation, such as efficient garbling techniques, optimal circuit design, and sophisticated protocols, a predominant challenge persists in the form of communication overhead. The iterative nature of federated learning, compounded by the necessity for multiple communication rounds even under semi-honest adversary assumptions, results in a notable overhead. Another issue with MPC is that it is a synchronous procedure which cannot be ported to any asynchronous form of FL.

Differential Privacy

Differential Privacy (DP) [93, 94] is a rigorous mathematical framework designed to provide strong privacy guarantees in datasets while allowing for useful aggregate analysis. It achieves this by adding controlled noise to the data or query responses, ensuring that the inclusion or exclusion of any single individual's data does not significantly affect the outcome of the analysis. This makes it difficult for adversaries to infer private information about any individual, even with auxiliary information. Differential Privacy in Machine Learning has been applied to protect sensitive data during the training process [95]. By injecting noise into the input data or the gradients during training, it can be ensured that the resulting models do not memorize specific training examples, thereby preserving the privacy of the training data. Naturally, DP has been considered as a potential solution to the privacy concerns in Federated Learning. The core principle of differential privacy is the privacy budget, denoted as epsilon (ϵ), which quantifies the maximum allowable privacy loss. A smaller epsilon value indicates a stronger privacy guarantee, but it also results in a higher level of noise being added to the data, which can decrease model performance. The privacy budget is a finite resource, and once it is exhausted, no further queries can be made without compromising privacy. This necessitates a careful trade-off between the privacy budget and the utility of the data for producing useful, accurate results.

Federated Learning presents distinct challenges for the implementation of differential privacy. The decentralized learning approach necessitates the adoption of local differential privacy, wherein noise is added to local model updates prior to their transmission to a central server. This method represents a practical means to uphold privacy within the FL context, given that proficiently trained models can disclose sensitive training data information. Although this technique is effective in safeguarding privacy, it also poses a challenge: the noise added to model updates can substantially reduce the model's accuracy. Managing the privacy budget proves challenging due to the vast volume of non-identically distributed data and the distributed nature of the nodes. Given the varied distributions, selecting an appropriate privacy budget is complex, and overseeing the budget across numerous training cycles is even more demanding. Nonetheless, the principal advantage of applying differential privacy in FL is its minimal impact on the communication process, unlike other more secure methods such as MPC. Although the reduced accuracy in model updates may necessitate additional communication rounds, the work by Li, He, and Song [57] has shown that it is possible to converge within a single communication round while adhering to differential privacy requirements.

In the scoping review from Brauneck et al. [59], the authors concluded that differential privacy on its own cannot fulfill the legal data protection requirements from the GDPR and similar data protection laws. While theoretically no private data from individuals is exposed, local model updates could still be accessible to adversaries, rendering other inference attacks, such

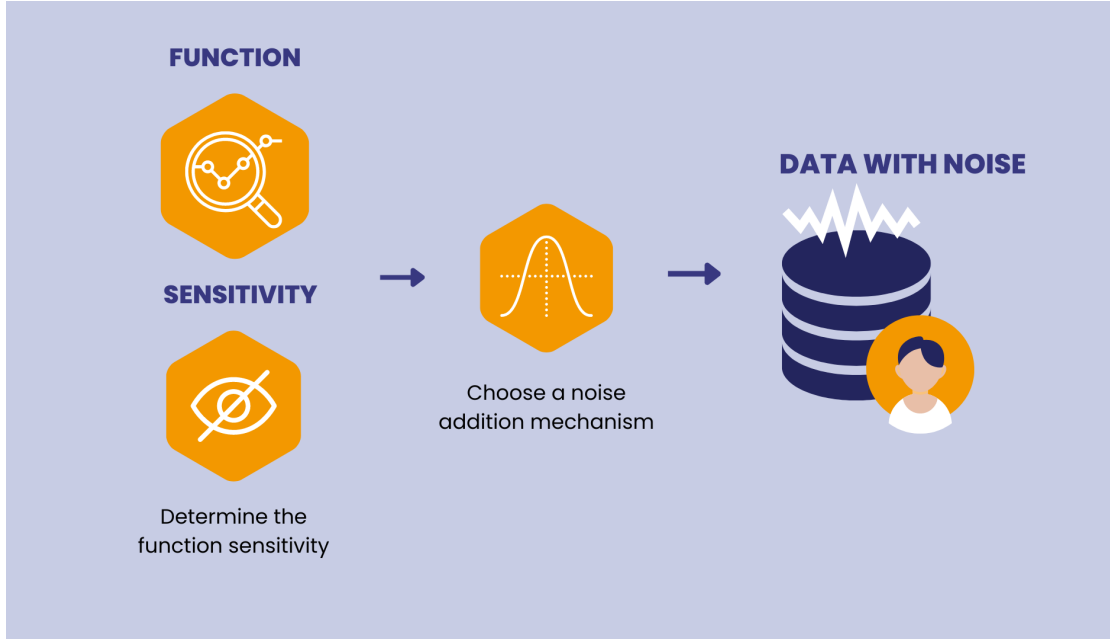


Figure 2.11: Differential Privacy [96]

as property attacks, plausible. However, integrating differential privacy with secure multiparty computation could establish a system resilient to data breaches and compliant with the GDPR. This combination offers a robust privacy guarantee, concealing model updates from adversaries and ensuring the global model does not memorize specific training examples.

Homomorphic Encryption

Homomorphic Encryption (HE) [97] represents a form of encryption that allows computations such as addition and/or multiplication to be performed on ciphertexts, generating an encrypted result which, when decrypted, matches the result of operations as if they had been performed on the plaintext. This property is particularly advantageous in the context of Federated Learning, as it allows a central server to process encrypted model updates without needing to access the raw data, thus potentially satisfying privacy requirements. The concept of HE was proposed in 1978 and was utilized in encryption systems proposed by Goldwasser, Micali, and Tong [98] and Paillier [99]. It only allowed a single addition over an encrypted bit. Work by Boneh, Goh, and Nissim [100] enabled unlimited additions and one multiplication. Finally, the first fully homomorphic encryption scheme was proposed by Gentry [101], which allowed for unlimited additions and multiplications by using ideal lattices.

HE schemes are typically defined by four functions: KeyGen, Enc, Dec, and Eval. KeyGen generates the public and private keys, Enc encrypts the plaintext, Dec decrypts the ciphertext, and Eval computes the function on the ciphertext. The Eval function is the most crucial, as it allows the server to perform computations on the encrypted data. The Eval function can be either partially homomorphic, supporting only one operation, or fully homomorphic, supporting both addition and multiplication. There are three classes of HE:

- Partially Homomorphic Encryption (PHE): This variant supports endless operations of

a single type (addition or multiplication) on encrypted data, leveraging the principles of group homomorphism.

- Somewhat Homomorphic Encryption (SHE): SHE permits a finite number of both additions and multiplications. However, it is limited by the operational complexity it can handle. SHE schemes incorporate noise as a security measure, with each operation on encrypted data incrementally increasing this noise. Exceeding a certain noise threshold renders accurate decryption unfeasible.
- Fully Homomorphic Encryption (FHE): FHE schemes facilitate unrestricted additions and multiplications on encrypted data. They manage to maintain security and functionality through circular security assumptions and the use of bootstrapping techniques to mitigate noise, making them the most secure and flexible, albeit computationally demanding.

In the FL environment, PHE could be utilized to securely implement a FedAvg algorithm, where addition is the primary operation needed. However, more complex model aggregation algorithms may necessitate scalar or ciphertext multiplications, for which SHE schemes, despite their noise management challenges, offer the necessary capabilities. These schemes are conducive to FL algorithms that ensure convergence within a predetermined number of communication rounds, as noise accumulation is a critical factor. FHE, despite being the most secure and flexible option, presents considerable computational challenges, potentially increasing communication overhead and preventing many practical use cases [102]. Therefore, in the integration of HE within the federated learning process, SHE algorithms emerge as a practical choice, balancing functionality, security, and efficiency. Homomorphic Encryption has been used in FL systems and we refer the readers to the papers of Fang and Qian [103] and Zhang et al. [104] for more detail.

Trusted Execution Environments

Trusted Execution Environments (TEEs) [105] play an essential role in contemporary computing by offering a secure section within a processor. This section safeguards sensitive information and code from external elements, including the operating system, other applications, and potential security threats, thus facilitating secure computation through the preservation of data confidentiality and integrity. This feature is particularly useful in thwarting software-based attacks. In multi-device scenarios, such secure environments are found in smartphones and tablets that operate on ARM architecture [106]. For cross-silo contexts, the primary CPU manufacturers for large-scale desktop environments are Intel and AMD, with Intel’s Software Guard Extensions (SGX) [107] and AMD’s Secure Encrypted Virtualization (SEV) [108] being the forefront of TEE technologies.

Intel SGX [107] encompasses a collection of security-centric instructions integrated into certain Intel CPUs, enabling the creation of secure enclaves within the processor for the safekeeping and processing of confidential data. SGX aims to establish a secure execution space that maintains high security levels, even when other system components are compromised. Designed to withstand both software and hardware attacks external to the enclave, SGX ensures the protection of data within the enclave by encrypting its memory, thereby restricting access to unauthorized processes. A pivotal feature of SGX is its attestation capability, which allows an enclave to confirm to a remote entity that it is executing genuine code within a secure setting. This function is particularly valuable in Federated Learning for the secure execution of model training, aggregation, and deployment, enabling trust verification between clients and servers regarding the legitimacy of model updates and deployments.

2.4 Model Consolidation Techniques

This segment will delve into the assortment of methodologies that Federated Learning can utilize to harness a global model effectively. Initially, the discussion will focus on the intricacies of the federated optimization challenge, followed by an examination of the various forms that model updates can take. Additionally, the discourse will encompass the foundational algorithm and the progressive enhancements that have been developed to refine it.

2.4.1 Federated Optimization

The concept of Federated Optimization was initially introduced by McMahan et al. [11], delineating the complexities associated with the optimization tasks in Federated Learning frameworks. This concept is characterized by several distinct features:

- **Non-Independent and Identically Distributed (Non-IID):** Data on individual clients' devices often reflects the unique interactions of the users with their devices, making local datasets not representative of the overall population distribution.
- **Unbalanced:** The volume of local training data varies across users due to differences in engagement levels with the service or application.
- **Massively distributed:** The optimization process involves a significantly larger number of client participants than the average dataset size per client.
- **Limited communication:** Connectivity issues are common with mobile devices, which may be offline or connected through slow or costly networks.

The primary objective of Federated Optimization lies in addressing the challenges that arise within cross-device contexts. Issues such as constrained communication are not as pertinent in cross-silo scenarios. However, extensive data distribution remains a concern, primarily due to the personalized nature of the data, though it is not as severe as in cross-device scenarios. In environments characterized by data silos, disparities in data volume can be managed through reduced incentives or by setting minimum data thresholds for participation. Nonetheless, the critical challenge in cross-silo contexts is the heterogeneity of data, which stands as the foremost concern to be addressed.

2.4.2 Model Updates

In the Federated Learning framework, participants have the option to transmit their local models using two distinct methodologies. The first method involves dispatching the gradients to the central server, where an aggregation of the incoming gradients occurs, a process often denoted as gradient averaging [109, 110]. The second method involves combining the actual model weights (the parameters that define the model) from each user's model, an approach frequently referred to as model averaging [11, 111, 112]. In the hypothetical scenario where model parameters are averaged following each epoch and all participants commence with an identical initial model, model averaging is equivalent to gradient averaging, as asserted by McMahan et al. [11] and Su and Chen [109]. A comparative analysis of these methodologies is presented in Table 2.1.

2.4.3 FedAvg: The Baseline Algorithm

In this subsection, we will present the foundational aggregation algorithm utilized for combining local model updates. Subsequently, we will discuss the limitations inherent to these baseline

Method	Advantage	Disadvantage
Gradient averaging	Accurate gradient information Guaranteed convergence	Heavy communication Require reliable connection
Model averaging	Not bound to SGD Tolerance of update loss Infrequent synchronization	No guarantee of convergence Performance loss

Table 2.1: Comparison of Gradient and Model Averaging Methods [16]

algorithms and provide examples of advanced algorithms designed to address these shortcomings. To maintain the coherence of this subsection, our primary focus will be on an in-depth exploration of the Federated Averaging approach, with additional emphasis placed on outlining the distinctions and enhancements offered by various implementations of this method. We encourage the reader to explore the original papers of each proposed algorithm for a deeper understanding, while also noting that this thesis does not delve into intricate mathematical concepts like convergence guarantees.

The Federated Averaging (FedAvg) algorithm, introduced by McMahan et al. [11], marked the inception of model aggregation techniques in Federated Learning. Regarded as the foundational algorithm, FedAvg serves as the cornerstone upon which numerous subsequent aggregation algorithms are developed. The process commences with the random selection of clients, each of whom is provided with the global model. Over a predetermined number of epochs, these clients partition their data into batches and execute the Stochastic Gradient Descent (SGD) algorithm on these batches. Following the retraining phase, the updated model weights from each client are transmitted back to the server. At the server level, an averaging of these weights from all participating local models is conducted. This cycle is iterated until the global model attains convergence. The algorithm for FedAvg is detailed in Algorithm 1.

Algorithm 1 Federated Averaging. The K clients are indexed by k ; B is the local minibatch size, E is the number of local epochs, and η is the learning rate. [11]

```

1: initialize  $w_0$ 
2: for each round  $t = 1, 2, \dots$  do
3:    $m \leftarrow \max(C \cdot K, 1)$ 
4:    $S_t \leftarrow$  (random set of  $m$  clients)
5:   for each client  $k \in S_t$  in parallel do
6:      $w_{t+1}^k \leftarrow \text{ClientUpdate}(k, w_t)$ 
7:   end for
8:    $w_{t+1} \leftarrow \sum_{k \in S_t} \frac{n_k}{n} w_{t+1}^k$ 
9: end for
10: function CLIENTUPDATE( $k, w$ ); ▷ Run on client  $k$ 
11:    $B \leftarrow$  (split  $P_k$  into batches of size  $B$ )
12:   for each local epoch  $i$  from 1 to  $E$  do
13:     for batch  $b \in B$  do
14:        $w \leftarrow w - \eta \nabla \ell(w; b)$ 
15:     end for
16:   end for
17:   return  $w$  to server
18: end function

```

The FedAvg algorithm’s straightforward design facilitates its implementation across a diverse range of edge devices, making the central server’s operations intuitively simple. The aggregator’s task, primarily involving summation and averaging, is inherently uncomplicated, which allows for seamless integration with secure protocols aimed at safeguarding model privacy, without significantly increasing computational burden. Within the context of blockchain technology, the algorithm’s simplicity renders it an ideal fit for lightweight Smart Contracts within public blockchain systems. The study by McMahan et al. [11] demonstrates that FedAvg is more robust to data heterogeneity compared to Federated Stochastic Gradient Descent (FedSGD) [113], a widely used algorithm in Distributed Machine Learning. However, despite this robustness, FedAvg still faces limitations when dealing with unbalanced and non-IID data. Although it represents an initial attempt to address the challenges presented by heterogeneity, there remains considerable scope for improvement, as will be evident in the discussion of more advanced algorithms that follow.

2.4.4 Addressing Statistical and Systems Heterogeneity

Client drift in the context of Federated Averaging (FedAvg) can be understood through the behavior of updates from multiple clients as they aim for their local optima. In Figure 2.12a, we observe two clients, each performing local updates (denoted as y_1 and y_2), which are visually represented in blue. These updates are directed towards each client’s individual optimal points, marked by orange squares (x_1^* and x_2^*), which we can interpret as the client optima. The server, on the other hand, conducts its own updates (shown in red), which instead of converging to the true global optimum (x^* , depicted as a blue square), tends towards the average of the client optima. This average is illustrated by grey squares and indicates the point where the server updates are directed. The phenomenon of client updates moving towards their local optima, as opposed to the true global optimum, and the server updates averaging these, illustrates the concept of client drift in FedAvg. This drift can potentially lead to suboptimal global convergence, as the server’s aggregated updates may not necessarily align with the true global optimum. This problem is addressed by SCAFFOLD [114] which uses control variates which are techniques used in statistics to reduce variance. The control variates act as corrective measure in the algorithm’s local updates, moving the server towards the global optimum, as seen in Figure 2.12b. Therefore, the federated learning process is more efficient and less susceptible to the adverse effects to statistical heterogeneity and client sampling. An important property of the SCAFFOLD algorithm is that the server needs to keep a control variate per client and that the clients in SCAFFOLD retain their control variate value across multiple rounds, making them stateful. This limits the possibility for stateless algorithms and can be a major constraint for cross-device settings. The MIME algorithm [115], as an innovative approach in cross-device federated learning, addresses the limitations of stateful algorithms like SCAFFOLD by being inherently stateless. Unlike SCAFFOLD, which requires server-side tracking of control variates and thus is less suited for cross-device contexts with potentially sporadic device participation, MIME circumvents these challenges. It achieves this by integrating novel mechanisms that prevent client drift without necessitating persistent client state, making it a more viable option for scenarios with resource-constrained devices and intermittent connectivity.

The choice in amount of epochs to perform locally presents some interesting trade-offs. On the one hand, more epochs means that more local computation is done and we can reduce the amount of communication before reaching convergence. However, a high amount of epochs can be computationally exhaustive, increase client drift and may not be realizable by all the nodes. This introduces two different but equally impactful problems:

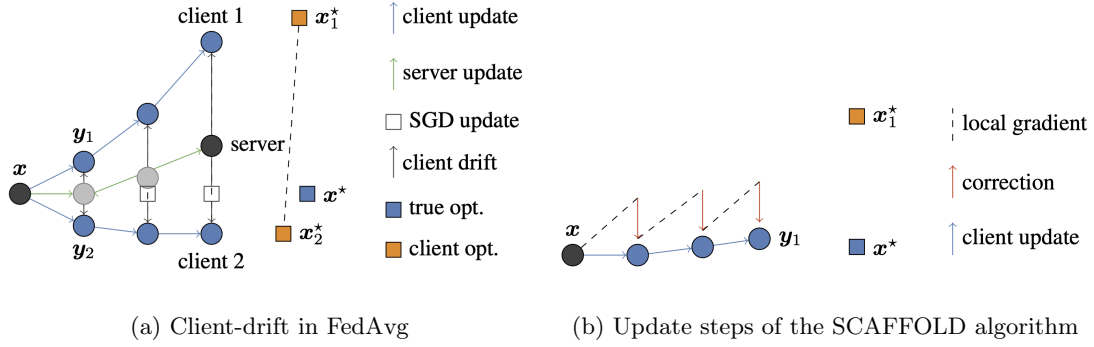


Figure 2.12: Comparative illustration of client-drift and SCAFFOLD algorithm correction. [114]

1. The nodes that do not have the resources to perform these epochs in a timely manner will either slow down the entire process of FedAvg or be dropped from the learning process. We commonly refer to these types of nodes as stragglers.
2. The nodes that can finish their computation on time, however these nodes might overfit their models to their local data, therefore causing client drift.

FedProx [116] introduces a more adaptable model by integrating partial updates from individual nodes, effectively engaging stragglers in the learning process rather than excluding them. This method helps to mitigate the issue of system heterogeneity. A key feature of FedProx is the inclusion of a proximal term in the local optimization problem, which limits the divergence of local updates and prevents client drift, thus tackling statistical heterogeneity. By ensuring local updates remain closely aligned with the initial global model, FedProx negates the necessity for fine-tuning the count of local epochs. Additionally, this model naturally supports variable extents of local computation, enhancing its flexibility. It's crucial to note that FedProx is designed as a more generalized variant of the FedAvg algorithm, with the express purpose of simplifying its integration into existing federated learning frameworks. The algorithm for FedProx is detailed in Algorithm 2.

Algorithm 2 FedProx [116]

Require: $K, T, \mu, \gamma, w^0, N, p_k, k = 1, \dots, N$

- 1: **for** $t = 0, \dots, T - 1$ **do**
 - 2: Server selects a subset S_t of K devices at random (each device k is chosen with probability p_k)
 - 3: Server sends w^t to all chosen devices
 - 4: Each chosen device $k \in S_t$ finds a w_k^{t+1} which is a γ_k^t -inexact minimizer of:
 - 5:
$$\arg \min_w h_k(w; w^t) = F_k(w) + \frac{\mu}{2} \|w - w^t\|^2$$
 - 6: Each device $k \in S_t$ sends w_k^{t+1} back to the server
 - 7: Server aggregates the w 's as $w^{t+1} = \frac{1}{K} \sum_{k \in S_t} w_k^{t+1}$
 - 8: **end for**
-

Tuning hyperparameters to optimize FedAvg's performance can be challenging, often leading to suboptimal model convergence. In traditional non-federated learning environments, adaptive optimization techniques like ADAM [117], Adagrad [118], and Yogi [119] are employed to dynamically adjust gradients' learning rates, thereby reducing model loss and enhancing performance

[120]. Reddi et al. [121] propose integrating these adaptive optimizers into the federated learning framework. Their algorithm, FedOpt, aims to introduce adaptability into the realm of Federated Learning by enabling both client-side and server-side adoption of varied optimization techniques, thus addressing the need for more flexible optimization strategies. Both FedOpt and FedProx are developed to address the challenges associated with the adaptability limitations of FedAvg.

Algorithm 3 FedOpt

Require: x_0 , CLIENTOPT, SERVEROPT

```

1: for  $t = 0, \dots, T - 1$  do
2:   Sample a subset  $S$  of clients
3:    $x_{t,i,0} = x_t$ 
4:   for each client  $i \in S$  in parallel do
5:     for  $k = 0, \dots, K - 1$  do
6:       Compute an unbiased estimate  $g_{i,k}$  of  $\nabla F_i(x_{t,i,k})$ 
7:        $x_{t,i,k+1} = \text{CLIENTOPT}(x_{t,i,k}, g_{i,k}, \eta_l, t)$ 
8:     end for
9:      $\Delta_{t,i} = x_{t,i,K} - x_t$ 
10:  end for
11:   $\Delta_t = \frac{1}{|S|} \sum_{i \in S} \Delta_{t,i}$ 
12:   $x_{t+1} = \text{SERVEROPT}(x_t, -\Delta_t, \eta_g, t)$ 
13: end for
  
```

2.4.5 Adapting Data Distributions

Different data distributions can be viewed as distinct domains, necessitating the application of domain adaptation techniques to tackle the issue of heterogeneity. The concept of Federated Domain Adaptation has been previously introduced by Peng et al. [122] and Peterson, Kanani, and Marathe [123], and recent studies have explored its application in cross-device scenarios to address the challenge of data quality variability across devices (e.g., well-labeled versus poorly or unlabeled data) as well as in cross-silo contexts to enhance the utility of limited data by adjusting it to a more advantageous distribution. Li et al. [124] have developed a privacy-centric Federated Learning (FL) framework that incorporates Domain Adaptation to alleviate statistical heterogeneity. Their study delineates two specific domain adaptation strategies: the Mixture of Experts (MoE) and adversarial domain alignment. The MoE method aims to align the local data distribution with the global one by employing a gating network that determines the contribution of each local model. Within the Federated Learning paradigm, each node maintains a portion of private data, developing a local model that remains exclusive and hence, exempt from privacy considerations. This results in each node possessing a specialized model tailored to its unique data distribution. Subsequently, a gating function facilitates the integration of outcomes from both the private and global models, enhancing prediction accuracy and supporting continued model training. However, this approach necessitates the segregation of a data subset for the development of the private model, which limits the performance of the best possible global model.

The adversarial domain alignment method leverages a discriminator network to reconcile the disparities between local and global data distributions, aligning them more closely. This strategy, as highlighted by Li et al. [124] and Kang et al. [125], initiates with the training of site-specific local feature generators and classifiers, aiming for accurate data classification. To facilitate domain alignment, a domain discriminator for each source-target duo is introduced, which evaluates whether the generated features originate from the source or target domain. The training of feature

generators is conducted in an adversarial manner against these discriminators, with the goal of crafting features that challenge the discriminators' classification capabilities. This encourages the emergence of features that are invariant across domains. Throughout the training phase, the local models update their parameters using gradients derived from a cross-entropy classification loss for improving classification accuracy, and an adversarial loss to deceive the discriminators, promoting domain invariance. These adversarial updates are designed to penalize the discriminator for correctly identifying the domain of the data, thus pushing the feature generator towards producing domain-agnostic features. The algorithm for this approach is detailed in the paper by Li et al. [124].

An alternative strategy to alter the data distribution involves the centralization of a subset of the data to address statistical heterogeneity. This concept is explored through the analysis of findings by the authors in the study identified by Zhao et al. [126]. Weight divergence, as described in the study, refers to the deviation of model weights from the global objective towards a local objective, paralleling the issue known as client drift. The study demonstrates that data skewness, assessed using the Earth Mover's Distance, inversely affects model accuracy. To counteract this, a portion of the private data from all participants is centralized and stored on a global server. This global data is partitioned in two parts; one part is utilized for training the initial global model, and the remainder is redistributed among participants to harmonize their data distributions. Following this redistribution, the standard federated learning process commences, with the study showing improved model accuracy using the baseline FedAvg algorithm, thereby presenting a compelling case for further exploration of this method. Nevertheless, this methodology presents challenges. Centralizing a subset of the data contravenes the foundational principles of federated learning. Moreover, similar to MoE, this approach imposes constraints on the optimal performance achievable by the global model. Despite these drawbacks, the strategy offers a compelling means to alleviate statistical heterogeneity. This method might be viable with publically accessible data, thus safeguarding the privacy of data on individual devices. It is important to note the necessity of a central aggregator for this approach, which poses challenges in decentralized and semi-decentralized architectures, particularly in determining the entity responsible for initializing the global model and managing the redistribution of global data. An example of this approach can be found in Figure 2.13.

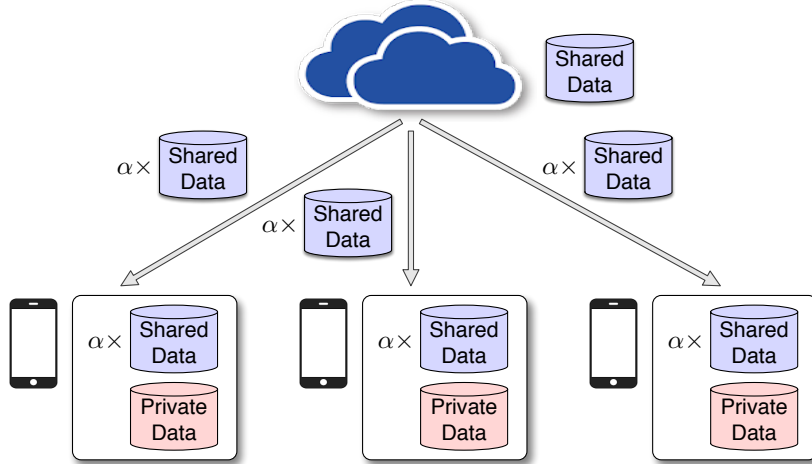


Figure 2.13: Illustration of the data sharing strategy. [126]

2.4.6 Addressing Heterogeneity as Tasks

A novel approach to the heterogeneity problem is to consider the different distributions across devices as separate tasks that must be learned, every unique distribution warranting a new task. Using this perspective, one can look into training models in order to fulfill as many tasks as possible, therefore solving the heterogeneity problem.

The FedCurv approach, as delineated by Shoham et al. [127], employs Lifelong Learning principles to address statistical heterogeneity in Federated Learning (FL). By leveraging the concept of Lifelong Learning, FedCurv aspires to equip the system with the capability to assimilate new data distributions without compromising the performance on previously learned distributions. This is achieved through the utilization of the Elastic Weight Consolidation (EWC) algorithm [128], which employs the diagonal of the Fisher information matrix to safeguard parameters critical to each task.

Crucially, FedCurv necessitates the transmission of model weights, along with the sum of squared and local gradients, from clients to the server. This information is pivotal for computing the regularization term that helps in retaining knowledge from prior tasks while accommodating new information. Such a mechanism allows for the estimation of parameter significance for preceding tasks, thereby enabling updates that preserve historical learning without necessitating direct access to old data. This aspect is particularly vital for managing non-IID data within FL environments.

However, it is important to note that the additional data transmission required by FedCurv does not involve patient data or other sensitive subject data. Instead, it pertains to model-related information, which introduces potential concerns regarding model security and increased communication overhead. Despite these concerns, FedCurv's mechanism for retaining knowledge from prior tasks while learning new ones is valuable, albeit with the trade-off of increased communication requirements.

The work by Smith et al. [129] proposes using Multi-Task Learning to train multiple highly related models which each aim to accurately predict their own participating node while also improving locally by sharing knowledge. The proposed optimization method is called MOCHA and is robust against stragglers and non-IID data. It uses a similar concept to the proximal term in FedProx to decide how much work the node has done in order to quantify how relevant the local update should be. The algorithm for MOCHA is detailed in Algorithm 4.

2.4.7 Approaches Involving Multiple Models

Rather than focusing on developing a singular model, employing an array of models can significantly enhance the efficacy of the federated learning framework. This concept has been previously touched upon in discussions regarding Multi-Task Learning and Domain Adaptation within earlier sections. Nonetheless, these instances represent merely a fraction of the potential strategies involving multiple models. This section aims to delve into additional methodologies that utilize multiple models.

Ensemble Learning

Ensemble learning, as detailed by Dong et al. [130], is a sophisticated machine learning paradigm that orchestrates the collaborative efforts of several models to refine the overall learning process. This technique employs a variety of algorithms to generate individual models, each contributing simple predictions. When these are aggregated, they collectively yield significantly more precise

Algorithm 4 MOCHA: Federated Multi-Task Learning Framework

```

1: Input: Data  $X_t$  from  $t = 1, \dots, m$  tasks, stored on one of  $m$  nodes, and initial matrix  $\Omega_0$ 
2: Starting point  $\alpha^{(0)} := 0 \in \mathbb{R}^n$ ,  $v^{(0)} := 0 \in \mathbb{R}^b$ 
3: for iterations  $i = 0, 1, \dots$  do
4:   Set subproblem parameter  $\sigma'$  and number of federated iterations,  $H_i$ 
5:   for iterations  $h = 0, 1, \dots, H_i$  do
6:     for all tasks  $t \in \{1, 2, \dots, m\}$  in parallel over  $m$  nodes do
7:       call local solver, returning  $\theta_t$ -approximate solution  $\Delta\alpha_t$  of the local subproblem,
       where:
8:        $\min_{\Delta\alpha_t} \mathcal{G}'_t(\Delta\alpha_t; v_t, \alpha_t) := \sum_{i=1}^{n_t} \ell_t^*(-\alpha_i - \Delta\alpha_i) + \langle w_t(\alpha), X_t \Delta\alpha_t \rangle + \frac{\sigma'}{2} \|X_t \Delta\alpha_t\|_{M_t}^2 + c(\alpha)$ 
9:       update local variables  $\alpha_t \leftarrow \alpha_t + \Delta\alpha_t$ 
10:      return updates  $\Delta v_t := X_t \Delta\alpha_t$ 
11:    reduce:  $v_t \leftarrow v_t + \Delta v_t$ 
12:  end for
13:  Update  $\Omega$  centrally based on  $w(\alpha)$  for latest  $\alpha$ 
14: end for
15: Central node computes  $w = w(\alpha)$  based on the latest  $\alpha$ 
16: end for
17: return:  $W := [w_1, \dots, w_m]$ 

```

forecasts than any standalone model could. Various methods, such as bagging, boosting, stacking, and voting, are employed to combine these models. In the context of enhancing federated learning, ensemble learning has been a focal point of investigation. The work by Mabrouk et al. [131] requires each node to train 8 well-known CNN models from which two will be selected to create the ensemble model that will be shared with the aggregator. This methodology is only viable within cross-silo scenarios due to its intensive computational demands. Conversely, a more adaptable strategy for cross-device frameworks is proposed by Hejazinia et al. [132]. This technique involves clustering devices based on either user-specific attributes or through clustering algorithms. Subsequently, each device partakes in federated learning to train a model representative of its cluster. These cluster-specific models are then combined by the aggregator into a global model. This strategy is particularly interesting for cross-device environments, as it allows for the scaling down of client models to accommodate the computational constraints of individual devices without compromising the integrity of the global model. However, this approach does entail a communication overhead, necessitating inter-device communication within clusters, likely through a peer-to-peer network, to facilitate model training.

Local Fine-Tuning & Meta-Learning

Local fine-tuning in Federated Learning involves initially training a global model in a federated manner, which is then customized at the client level through additional training on local data. This process aligns well with FL's model lifecycle, allowing the global model to be refined with local nuances post-deployment, enhancing its predictive performance on individual clients. However, this approach must be carefully managed to mitigate the risk of overfitting to local data.

This personalization strategy intersects with meta-learning concepts, particularly model-agnostic meta-learning (MAML), which facilitates the adaptation of a global model to specific tasks with minimal local adjustments. The integration of MAML and FL, exemplified by algorithms like Personalized FedAvg [133], introduces an interesting research area exploring the synergy between

FL personalization and meta-learning techniques, with a focus on optimizing models for non-IID data distributions and enhancing personalized model performance.

Personalization via Featurization

Employing multiple models in federated learning, such as the mobile keyboard prediction model discussed by Hard et al. [22], highlights the effectiveness of personalization due to varied language usage among clients. This approach, as adopted and advanced by Wang et al. [134], underlines the importance of client-specific models. Kairouz et al. [29] further explores when a global model might outperform individual client models. The Ditto framework, introduced by Li et al. [135], extends this discourse by offering a personalized federated learning solution that balances local customization with global model benefits, proving particularly advantageous in diverse data environments. However, this thesis will not delve into personalization, as this concept primarily applies to contexts where the results may be highly individualized, which is beyond the scope of this work. Nevertheless, it remains a pertinent topic for future exploration.

2.5 Fairness in FL

A frequently overlooked element of Federated Learning is the possibility of bias and unfairness within the models being developed. These issues are often overshadowed by more pressing concerns as outlined in Section 2.2. Bias and fairness are well-known issues in the wider field of Machine Learning [136]. Unfairness can relate to the data used for training the model, the model itself, or the incentives provided to participants. In this section, we will explore various aspects of fairness in Federated Learning and potential ways to address these issues.

2.5.1 Incentive Mechanisms

In any medium of collaboration, the incentive structure plays a pivotal role in determining the level of participation and the quality of contributions. This is particularly true in the context of Federated Learning, where the quality of the model is contingent on the quality of the data and the effort invested by participants [137]. In cross-device environments where clients can freely join and leave, some sort of incentive must be given to the client if they wish to participate in the learning process. Computational resources are needed to facilitate the learning process, and these resources are not free. Interestingly, incentive mechanisms were not initially discussed in the original Federated Learning papers from Google [11, 22]. This is likely because the inherent incentive for participation was the improvement in the quality of life for users of Android services. In cross-silo environments, the limited amount of participants and the fact that they are often large organizations means that the incentive structure can be more easily managed. These large organizations often have a vested interest in the model and that could suffice to ensure that every client is 100% compliant to the learning process. Incentive mechanisms can act as an effective deterrent against disruptive actions by malicious clients in the learning process. These mechanisms reward participants with monetary gains either for participating and actively improving the global model or for performing aggregation tasks. Such incentives not only encourage constructive participation but also penalize detrimental behavior. For further insight into incentive mechanisms in FL, the following papers provide comprehensive information [138, 139, 140].

2.5.2 Fairness of Data

The first notion of fairness in Federated Learning is the fairness of the data. This is a well-known issue in the wider field of Machine Learning, and it is particularly pertinent in Federated Learning due to the decentralized nature of the data. The data used for training the model can be biased, and this bias can be reflected in the model. This is particularly true in cross-device settings, where the data is often non-IID. Frameworks like agnostic federated learning [141] offer methods to mitigate bias within the training objective. When discussing data fairness, we distinguish two types of fairness: individual fairness and group fairness [142]. These concepts address the equitable treatment of data subjects from distinct but complementary perspectives.

Individual Fairness

The concept of individual fairness is grounded in the idea that people who are alike should be treated similarly in the context of a specific task. This concept requires a careful approach to define and measure similarity among individuals, which is challenging due to the varied nature of data and different situations where it applies. Individual fairness emphasizes the need for comparisons between individuals, ensuring fairness in treatment for each person in comparison to those who are similar to them. This approach also acknowledges the complexity of intersectionality, where biases may overlap and intensify for people who are part of multiple protected groups. The difficulty lies in creating fairness measures that can effectively identify and address these detailed similarities and overlapping biases.

Group Fairness

Conversely, group fairness focuses on ensuring equitable outcomes for various demographic or defined groups within a community, especially concerning sensitive traits like gender, race, and age. The aim is to counteract biases stemming from the categorization and usage of such sensitive traits, striving for equal treatment. This involves ensuring that each subgroup within a community receives fair treatment, with outcomes that align with those of the broader population, regardless of any protected characteristics. Achieving group fairness involves efforts to ensure that predictions are not influenced by these sensitive traits, thereby reducing bias. Within Federated Learning, achieving this balance is particularly challenging due to the necessity of maintaining client privacy and limiting direct access to sensitive client information.

These issues are paramount in Machine Learning, but to maintain the focus of this thesis on Federated Learning, we will not explore these issues in depth. For additional information about data bias, we direct the reader to the survey by Mehrabi et al. [136] and the paper by Vucinic and Zhu [137]. It is worth noting that although Differential Privacy aims to remove sensitive personal data from datasets, potentially contributing to fairness, this is not its primary purpose and it does not fully address fairness concerns.

2.5.3 Fairness of Model

In the context of Federated Learning, the fairness of the models emerges as a distinct challenge. These models are typically trained on data from diverse sources, which can introduce biases. This issue is particularly pronounced in cross-silo environments, where the data often differs significantly (non-IID) and involves a smaller number of participants. Consequently, models may become biased towards the data from larger contributors, disadvantaging smaller participants [30]. This concern extends beyond Federated Learning to encompass Distributed Machine Learning,

where it is commonly addressed through data partitioning strategies, a solution not feasible in Federated Learning contexts.

Moreover, biases in models can arise from the selection of hyperparameters. Certain hyperparameter configurations may not be suitable for all participants, potentially disadvantaging them, especially in scenarios where algorithms are designed to be robust to stragglers. This issue is particularly relevant in cross-device scenarios, where there can be significant variation in the computational capabilities of devices. In cross-silo settings, the hyperparameters chosen for the collective learning process might conflict with the optimal settings for an individual client's local model, potentially degrading performance compared to pre-participation levels. While personalization strategies as discussed in Section 2.4.7 offer some mitigation, they do not fully resolve these fairness concerns.

2.6 Federated Learning Frameworks

In this section, we will delve into the different frameworks available for Federated Learning. We will explore the different features of these frameworks and how they handle the challenges and opportunities of Federated Learning. We will also discuss how these frameworks can be adopted in the medical sector.

2.6.1 Criteria for Evaluating Federated Learning Frameworks

When evaluating Federated Learning frameworks, it is important to consider the following criteria proposed by Braungardt [143]:

- **Model Support:** The compatibility of frameworks with specific model types is crucial. Our focus is on a deep learning model that employs neural networks. Ensuring the chosen framework accommodates deep learning models is vital, though most contemporary Federated Learning frameworks do.
- **ML Framework Support:** The ease of integration with the Machine Learning library used for our model is significant for seamless interoperability. While many frameworks are compatible with popular libraries like TensorFlow and PyTorch, some are framework-agnostic, facilitating use with any ML library.
- **Expandability:** The flexibility to enhance the framework with new functionalities and to tailor it to novel applications is especially pertinent in healthcare, where innovation is constant. This thesis will explore the framework's adaptability to diverse Federated Learning architectures.
- **Aggregation Algorithms:** A fundamental component of Federated Learning is the aggregation algorithm. Although FedAvg is commonly supported, the ability to utilize alternative algorithms or devise custom ones is imperative.
- **Privacy Methods:** In healthcare, safeguarding data privacy is paramount. Selecting a framework that employs robust privacy measures is essential to ensure data protection.
- **Commercial Usage:** While not a primary concern within the research field, the framework's readiness for real-world deployment merits consideration. Frameworks offering additional features for deployment and management might be advantageous, albeit possibly at the expense of simplicity and flexibility.

- Device and OS support: Given that medical sector devices typically do not serve personal computing purposes, it is reasonable to prioritize frameworks compatible with Linux, which will likely be the operating system in use.

2.6.2 PySyft

PySyft, developed by OpenMined and supported by major tech firms, is a Python library tailored for secure and private Deep Learning. It builds upon PyTorch and TensorFlow, offering tools for privacy-preserving machine learning beyond its capabilities as a Federated Learning framework, positioning it as a broader remote data science platform [144, 145]. PySyft stands out for incorporating leading privacy protection methods like Secure Multiparty Computation, Differential Privacy, and Homomorphic Encryption. PySyft provides numerous ways to deploy applications to a Kubernetes cluster or any container engine [146].

The library enhances the Tensor class from PyTorch and TensorFlow, introducing a hierarchical model where tensor transformations are tracked through a parent-child relationship. This design facilitates an intuitive approach to managing FL model transformations, as depicted in Figure 2.14a. A key feature is the PointerTensor, enabling pointers to tensors on remote devices, which become available post model update transmission, illustrated in Figure 2.14b. For development ease, PySyft offers Virtual Workers to simulate secure communications and data federation across local virtual workers, mimicking a distributed computation graph as seen in Figure 2.14c. This graph outlines the Multi-Party Computation structure, supported by protocols such as SPDZ for secure operations [147] and SecureNN for key neural network layers [148]. Differential Privacy mechanisms in PySyft include noise addition and privacy budget management, while Homomorphic Encryption is facilitated via integration with libraries like PySEAL [149].

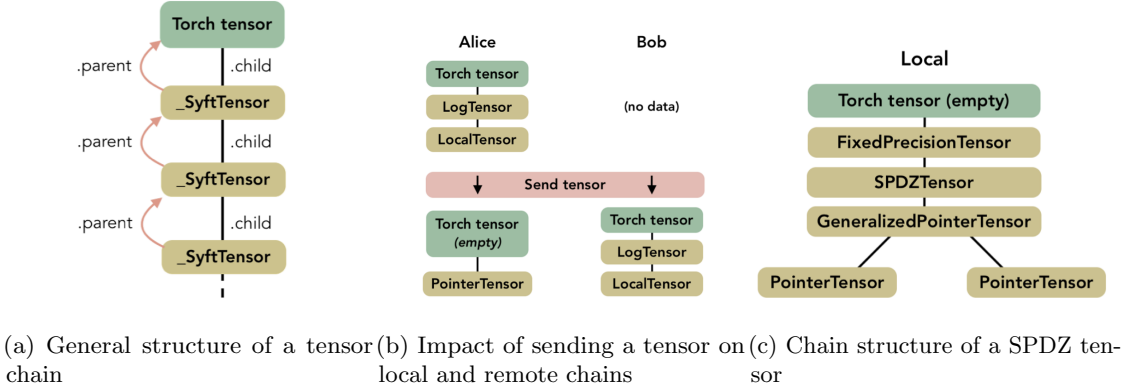


Figure 2.14: Tensors in PySyft [145]

Although PySyft currently supports only PyTorch and TensorFlow, there are initiatives to extend its compatibility to other models, aiming for a model-agnostic framework. PySyft's commitment to privacy, combined with its significant support and visibility within the tech community, renders it a leading choice for Federated Learning. Nevertheless, PySyft's focus on privacy and security does not exempt it from challenges and limitations inherent in such complex frameworks.

2.6.3 Flower

Flower, conceived at the University of Oxford and now managed by adap gmbh, a German AI firm, represents a paradigm shift in federated learning frameworks with its user-centric design [150]. Its creation aimed at demystifying FL application development, providing a high-level API conducive to research-oriented projects. Flower is distinguished by its exceptional flexibility, as it is not tied to specific programming languages, communication protocols, or machine learning models. Additionally, it is highly resilient to system diversity. This adaptability is evidenced by innovative adaptations for diverse research needs, such as integrating FL into Swift for iPhone applications [151] and embedding Ethereum blockchain technologies for enhanced FL processes [152].

The framework is distinguished by its comprehensive suite of FL algorithms, a robust communication infrastructure built on bidirectional gRPC streams (a Remote Procedure Calls framework developed by Google) for reliable client-server interactions, and practical guides for deployment on various platforms. Flower’s choice of gRPC is driven by its binary efficiency and continuous communication capability, which are essential for mobile networks with limited bandwidth. Its architecture promotes experimentation with advanced serialization techniques, including gradient compression and encryption, without being bound to gRPC, thereby supporting diverse client frameworks. The ClientProxy concept in Flower ensures seamless client management across different communication protocols, enhancing the framework’s adaptability.

Flower also introduces the Virtual Client Engine (VCE) to tackle the challenges posed by large-scale FL experiments. By optimizing resource distribution, VCE enables extensive experiments on modest hardware setups, leveraging the Ray framework [153] to manage and execute client tasks effectively, showcasing Flower’s capacity to accommodate a broad spectrum of FL applications and research endeavors.

However, Flower’s approach to privacy and commercial deployment presents limitations. Although it incorporates Secure Aggregation algorithms [154, 155] designed to protect against semi-honest adversaries, these do not match the comprehensive privacy features found in frameworks like PySyft. Moreover, Flower’s deliberate focus on flexibility and adaptability has led to a scarcity of built-in deployment and management tools, restricting its immediate applicability in commercial environments. This strategic choice, while enhancing its versatility for research purposes, necessitates further development for real-world FL implementations.

2.6.4 OpenFL

The Open Federated Learning (OpenFL) framework [156] is developed in Python 3 specifically for the Linux operating system and is dedicated to Federated Learning applications. Initially, it emerged from a joint research initiative between the University of Pennsylvania and Intel, aiming to devise the Federated Tumor Segmentation (FeTS) tool [157]. This framework includes ready-to-use integrations with TensorFlow and PyTorch for several aggregation algorithms, such as FedAvg, FedProx, and FedOpt, as well as a PyTorch-based implementation of FedCurv.

OpenFL enhances security by utilizing Intel SGX technologies [107] to enable secure aggregation and training within a Trusted Execution Environment. Moreover, the framework ensures the security of inter-node communications through the application of mutual Transport Layer Security (mTLS), a protocol designed to secure node-to-node interactions. Despite these robust security features against external threats, OpenFL’s current security framework does not incorporate specific privacy measures against internal risks, thus constraining its applicability primarily to cross-silo settings where there exists a baseline of trust among participants.

2.6.5 TensorFlow Federated

TensorFlow Federated (TFF) is a TensorFlow-based Federated Learning framework developed by Google, designed to facilitate the development and deployment of FL models [158].

TFF stands out with its two-layer architecture: the Federated Learning (FL) API and the Federated Core (FC) API. The FL API provides high-level interfaces that allow for easy application of federated training and evaluation to existing TensorFlow models. On the other hand, the FC API offers lower-level interfaces, giving developers the tools to express complex federated algorithms with TensorFlow's robust communication operations. One of the critical advantages of TFF is its seamless integration with TensorFlow/Keras, making it an excellent choice for developers already working within the TensorFlow ecosystem. It supports TensorFlow Board and allows for customized aggregation algorithms, which can be crucial for developers aiming to fine-tune federated learning models. Due to its integration with industry-level machine learning tools, TFF is a strong candidate for commercial deployment. However, TFF is only limited to TensorFlow, which is a significant drawback.

Privacy is another drawback of TensorFlow Federated. Although it officially supports differential privacy and unofficially supports secure aggregation, TFF does not support other privacy methods such as homomorphic encryption and secure multi-party computation. This limitation makes it less suitable for applications where privacy is a significant concern.

2.6.6 NVFlare

NVIDIA FLARE (NVFlare) is a domain-agnostic, open-source Federated Learning framework developed by NVIDIA, designed to facilitate the development and deployment of FL models [159]. It enables platform developers to build a secure, privacy-preserving FL system. It was originally developed to facilitate Nvidia Clara, a suite of AI solutions designed for the medical sector [160]. NVFlare has been made open source in 2021. This framework is equipped with ready-to-use FL algorithms such as FedAvg, FedProx, FedOpt, SCAFFOLD, Ditto, etc.

The versatility of NVFlare is evident in its support for various models, rendering it framework-agnostic. Designed with the medical sector in mind, NVFlare offers numerous privacy techniques such as differential privacy, homomorphic encryption, and private set intersection. These features, alongside its capabilities in data analysis and management of the machine learning lifecycle, position NVFlare as a viable option for commercial use. The framework boasts an extensive suite of security measures critical for sensitive fields like healthcare, encompassing authentication, authorization, and audit logging. These measures are vital for upholding data integrity and confidentiality within a federated learning setting.

NVFlare's design prioritizes extensibility, allowing for the integration of custom federated learning algorithms and compatibility with existing data processing and machine learning pipelines. This adaptability extends NVFlare's applicability beyond healthcare to any field where data privacy is paramount and centralization of data is impeded by privacy, bandwidth, or regulatory constraints. The framework includes a comprehensive toolkit for overseeing the federated learning process, including aspects like client management, job scheduling, and model evaluation.

An integral feature of NVFlare is its simulation environment, which facilitates the testing and development of FL algorithms without necessitating full deployment. This feature can significantly expedite the development process by enabling rapid iterations and testing within a controlled setting.

Another key aspect of NVFlare is its community and support. Being an open-source project,

it benefits from contributions from developers and researchers worldwide, which helps drive innovation and improve the framework. NVIDIA also provides support and documentation to assist developers in implementing federated learning solutions using NVFlare. However, community support for NVFlare is not as extensive as for other frameworks like PySyft and Flower, due to its relatively short time as an open-source project.

The main drawback of NVFlare is the learning curve required to understand the architecture. It is a complex framework that requires a deep understanding of federated learning and distributed systems. This can make it challenging for developers who are new to federated learning to get started with NVFlare, which can deter researchers from extending NVFlare for research purposes. However, the benefits of NVFlare, such as its comprehensive set of features, extensibility, and support for privacy and security, make it a strong candidate for commercial deployment in the healthcare sector.

2.6.7 FATE

The Federated AI Technology Enabler (FATE) [161], developed by WeBank, a subsidiary of Tencent in Shenzhen, is a leading framework in federated learning, offering functionalities similar to those of NVIDIA FLARE. Since its introduction in 2019, FATE has established itself as one of the first FL frameworks designed with business applications in mind. As an open-source project, it boasts an extensive range of modules, including preprocessing, various machine learning algorithms such as gradient-boosted decision trees, and tools for statistical analysis and privacy preservation, although it notably lacks support for differential privacy, homomorphic encryption, and secure multi-party computation.

FATE’s design is somewhat less flexible than its counterparts, yet it presents several key advantages through its ecosystem. Features like FATEBoard for live visualization and FATEServing for model deployment significantly enhance user experience. Additionally, its compatibility with Spark clusters and the ability to deploy using KubeFATE through Docker Compose and Kubernetes highlight its scalability and seamless integration capabilities for commercial use.

However, the framework’s focus on specific technologies rather than being framework-agnostic, along with a proprietary module architecture, may present challenges in adaptability and learning for new developers. The broad array of modules and features, while comprehensive, could potentially complicate customization and extension for specific user needs.

Overall, FATE is recognized as a solid and business-oriented FL framework, rich in pre-designed modules and supplementary features for effective orchestration. Despite facing some limitations in terms of flexibility and privacy features, alongside usability hurdles, FATE contributes significantly to the federated AI field, facilitating collaborative and secure data analysis within regulated environments.

2.6.8 LEAF

LEAF [162] is a toolkit designed for testing and comparing learning methods in federated settings. It includes three main parts that can be used with various systems:

- **Datasets:** LEAF offers a range of datasets suitable for federated learning, such as FEMNIST, Sent140, Shakespeare, CelebA, and Reddit, to help test different learning scenarios.
- **Algorithms:** It provides basic versions of some federated learning algorithms like minibatch SGD, FedAvg, and MOCHA, to serve as reference points or starting templates.

- **Metrics:** The toolkit supplies a comprehensive set of metrics to evaluate how well learning methods perform across various devices. This involves checking performance at different levels (10th, 50th, and 90th percentiles), analyzing outcomes based on data categories, and assessing the computational and data transfer demands on devices. LEAF also encourages considering how to balance the importance of each device or data point, whether treating them all equally or giving more weight to those used more frequently.

While LEAF is excellent for testing and comparing federated learning approaches, it's more of a benchmarking tool than a full-fledged framework. It doesn't offer the tools needed to develop federated learning algorithms but provides basic algorithm implementations for reference. Its primary use is in research or testing settings rather than for building commercial applications. In our research, we opted not to use LEAF's datasets, though its metrics could be beneficial for evaluating our methods. Similar metrics might also be available in other frameworks we are considering. Instead, we utilized brain age metrics from other brain age prediction studies, such as Mean Absolute Error, to validate our results. Despite not utilizing LEAF's datasets, it remains an essential benchmarking toolkit for those involved in federated learning research or development.

2.7 Medical Applications of Federated Learning

The thesis will examine the role of Federated Learning in the prediction of brain age, a critical aspect of healthcare. We will begin by investigating the application of FL in the accurate prediction of subjects' brain ages. Following this, the discussion will broaden to encompass the application of FL across the healthcare sector, evaluating both its advantages and the obstacles it presents. The healthcare sector, characterized by its handling of sensitive and confidential data, presents an ideal environment for the implementation of FL. This technology facilitates collaboration among diverse medical institutions, allowing them to jointly develop and refine predictive models without compromising the privacy of subject data. This section does not provide a complete overview of FL in healthcare.

2.7.1 Brain Age Detection

The process of predicting brain age involves analyzing a subject's brain data, which may include imaging techniques such as Magnetic Resonance Imaging (MRI) scans, as well as neurophysiological data like Electroencephalography (EEG) and Magnetoencephalography (MEG). These methods provide detailed information about the brain's structure and function, allowing for the assessment of various biomarkers indicative of neurological health and aging. The brain age prediction model, a sophisticated algorithm, is meticulously trained on a comprehensive dataset that includes brain data alongside the actual ages of the healthy subjects from whom the data was collected. Through this training, the model acquires the ability to discern the intricate patterns and relationships between the aging process captured in the brain imaging data and the chronological age of the subjects. This enables the model to estimate the age of a new subject's brain with precision, based on their unique brain imaging data. Accurate prediction of brain age is an important research tool in the medical field, particularly for the diagnosis and ongoing monitoring of neurological disorders.

Deviations from the expected brain age, as predicted by the model, can serve as critical indicators of potential neurological health issues. While initial attempts to develop accurate models for brain age prediction relied on traditional machine learning techniques, these efforts often did not achieve the level of accuracy and robustness required for effective clinical application [163]. In

contrast, Deep Learning methods, and more specifically Convolutional Neural Networks (CNNs), have shown exceptional promise in addressing these challenges, achieving remarkable accuracy in brain age predictions. This breakthrough is supported by research and documentation from sources like Tanveer et al. [164], Wood et al. [165], Peng et al. [166], and Leonardsen et al. [167], highlighting the significant advancements made in this area through the application of CNNs.

Federated Learning offers a promising approach to training brain age prediction models while preserving the privacy of the subject data. By training the model on data distributed across multiple medical institutions, FL enables the development of more accurate and robust models without compromising data privacy. The use of Federated Learning has been explored in recent research by Stripelis et al. [168], in which they explore brain age prediction in order to develop an Alzheimer’s disease detection model.

In the thesis by Silva [169], a deep learning model is developed using DenseNets [170] to predict brain age from T1-weighted MRI scans. The study emphasizes the impact of data heterogeneity, model initialization, and bias correction on the performance of brain age prediction models. Silva applied a bias correction to address issues highlighted by Le et al. [171], which noted that brain age predictions tend to overestimate the age of younger subjects and underestimate the age of older subjects. However, Silva found that the bias correction algorithm actually increased the model’s error. Therefore, while we will use Silva’s work as a baseline for our experiments, we will forego the bias correction to avoid this increased error in our federated models.

2.7.2 Other applications of Federated Learning in Healthcare

In healthcare, Federated Learning involves fewer participants compared to other fields because fewer institutions are equipped to store and manage the necessary data. This is due to the high costs, stringent regulations, and specialized infrastructure required to securely handle sensitive healthcare information. For instance, research by Brencic et al. [172] highlights the use of personalized Federated Learning models in emergency situations like detecting falls or identifying dangers. Healthcare organizations might also explore the advantages of edge computing through Federated Learning [15], presenting a promising area for future research. However, as mentioned earlier, our thesis will focus on cross-silo scenarios where a few healthcare institutions, each with limited subject data, collaborate. They aim to develop more precise models for disease detection. Given the small number of participants, a completely decentralized model is impractical. Thus, we will examine both centralized and semi-decentralized Federated Learning approaches in healthcare. Ensuring data privacy is crucial, and it must be maintained at all costs, even if it affects model precision.

Regarding data alignment, all three types have been applied in healthcare. Although horizontal Federated Learning is most common, vertical Federated Learning has been used to expand a subject database’s features for better classification [173]. Federated Transfer Learning, as introduced in the FedHealth framework by Chen et al. [174], trains healthcare models on wearable devices with differing data characteristics.

One significant application of Federated Learning was the development of a model to detect COVID-19 from chest scans during the pandemic [175]. This research continues to be relevant post-pandemic, as demonstrated by ongoing studies [176, 177, 178]. Additionally, Federated Learning has played a crucial role in developing models for tumor and cancer detection [179, 180].

An important aspect of Federated Learning in healthcare is ensuring the privacy of medical data. Prakash et al. [181] discuss the integration of Federated Learning with differential privacy to enable private deep learning for hospitals. This approach allows for the training of robust medical

models without compromising patient data privacy. Their work emphasizes the significance of maintaining data confidentiality while collaborating on model development, which aligns with the central goal of protecting sensitive healthcare information.

2.8 Research Question

Given the aforementioned background information, our goal is to develop a more robust and accurate model for predicting brain age. The primary motivation for this endeavor is to detect various biomarkers that may indicate neurological or psychological issues in patients. By accurately predicting the chronological age with a brain age model, we can potentially diagnose subjects with possible disorders, as these disorders can impact brain age.

However, gathering sufficient data from multiple institutions while adhering to data privacy regulations is a significant challenge and can be very time-consuming, hindering the training of an accurate model. To address this issue, we are utilizing Federated Learning, which allows us to leverage a maximal amount of data for machine learning without violating data privacy laws.

Despite the advantages, FL comes with its own set of limitations, including issues related to data heterogeneity, security, fairness, etc. This thesis specifically focuses on addressing data heterogeneity in the context of brain age models. Brain age prediction generally falls within a limited domain of natural numbers, with the majority of the population ranging from 0 to 100 years old. Since most databases exclude minors, we have a reasonable age range for our models.

Nonetheless, significant variability can still occur as different institutions may have diverse age distributions among their data, thereby violating the Independent and Identically Distributed condition and complicating the overall process. Our research aims to investigate the effects of age distribution across various institutions and to identify the most effective methods for improving brain age prediction models in this context. By doing so, we seek to produce a more accurate and reliable model for predicting brain age, ultimately aiding in the early diagnosis and monitoring of neurological and psychological disorders.

Chapter 3

Experimental Setup

These experiments have been implemented in Python, and the code is available in the following GitHub repository ¹. It is important to note that running this code directly is not feasible due to the absence of the MRI scans assembled by the AIMS lab in this repository. However, users can create their own patient datasets by adhering to the structure of the patient CSV files provided in the repository.

3.1 Data

3.1.1 Subject Dataset

The incorporation of Federated Learning into the process of predicting brain age enables the utilization of a more extensive dataset during the training phase, potentially yielding a more precise predictive model. Such an approach facilitates collaboration among data silos without necessitating the exchange of private data or obtaining specific permissions, thereby accelerating collaborative efforts and enhancing the accuracy of brain age predictions. In an effort to emulate this environment, a strategy is devised wherein the training data is distributed across client nodes, ensuring that a given subject within a node does not appear in multiple nodes. This approach aims to mitigate issues such as oversampling and contamination, and it more accurately reflects international scenarios where it is unlikely for a subject to be recorded in two separate hospitals in two different countries.

Given the variability in population demographics, subject sampling strategies, and other contextual factors, it is imperative to examine the impact of diverse age distributions across the nodes on the accuracy of the model. To this end, a series of Federated Learning experiments are planned, wherein subjects will be allocated according to predetermined age distributions. This methodological approach will allow for a nuanced understanding of how differences in age demographics across the federated network influence the model's predictive accuracy. By systematically varying the age distribution within the client nodes, we aim to identify patterns and potential biases introduced by demographic disparities, thereby enhancing the model's reliability and applicability across a wide range of settings.

For our research, we will utilize the subject database compiled by the AIMS research lab. This

¹<https://github.com/AIMS-VUB/BrainAgeFederated>

compilation encompasses healthy control subjects extracted from 216 distinct datasets, providing a diverse and comprehensive selection of subjects for our analysis. It is important to note that the individuals in this database have been compiled from several datasets. This compilation includes both healthy controls, whose brain age is similar to their actual age, and subjects with pathological conditions, who may have learning issues and an inaccurate brain age compared to their actual age. A comprehensive breakdown of the origins of the subjects can be found in Table 3.1 and details about the pathological conditions of the subjects are provided in Table 3.2.

Table 3.1: Number of images per open-source dataset in the 9,573 subject dataset.

Source	Name/Description	Images
ABIDE	Autism Brain Imaging Data Exchange	81
CamCAN	Cambridge Centre for Ageing and Neuroscience	653
CORR	Consortium for Reliability and Reproducibility	640
Openneuro (DS)	Open platform for sharing and analyzing neuroimaging datasets	5263
IXI	Information eXtraction from Images	562
OASIS_3	Open Access Series of Imaging Studies version 3	1019
PDD	Pediatric Developmental Disorder	80
RocklandSample	Sample from 1000 Functional Connectomes Project	940

Diagnosis	Count
No Diagnostic Info	4634
Control	4629
Pathology	244
Unknown	1
Total	9573

Table 3.2: Pathological conditions of subjects in the 9,573 subject dataset.

Data organization was performed using JSON metadata files and structured according to the Brain Imaging Data Structure (BIDS) [182] standard to handle heterogeneity. The data cleaning process filtered out subjects with missing or ambiguous age information, pathologies, and gender data to avoid bias, resulting in a dataset of healthy controls aged between 18 and 97 years. Image preprocessing included skull stripping to remove non-brain information, bias field correction to address image distortions, linear registration to align images to a common template, and resampling to a standardized size. Normalization using Z-score normalization was applied to rescale the images for consistent input into the model. For more information about data collection and preprocessing, we refer to the master thesis by Silva [169].

It is important to acknowledge the existence of two distinct databases in the context of this research. The first database comprises 6,326 subjects, inclusive of supplementary gender information, while the second database encompasses 9,573 subjects, devoid of such gender-related data. Despite some degree of overlap between these databases, there are notable discrepancies in the subject counts and the origins of the datasets. The decision was made to utilize the larger database, prioritizing the advantage of a greater number of data points for the experiments. Given that gender is not a focal point of these experiments, the absence of gender-related information does not detract from the value of the data being used. Accordingly, any potential influence of gender on the outcomes of the experiments is acknowledged but deemed beyond the current scope of inquiry. The age distribution specific to the dataset containing 9,573 subjects is delineated in

Figure 3.1, providing a foundational reference for understanding the demographic composition of the study population.

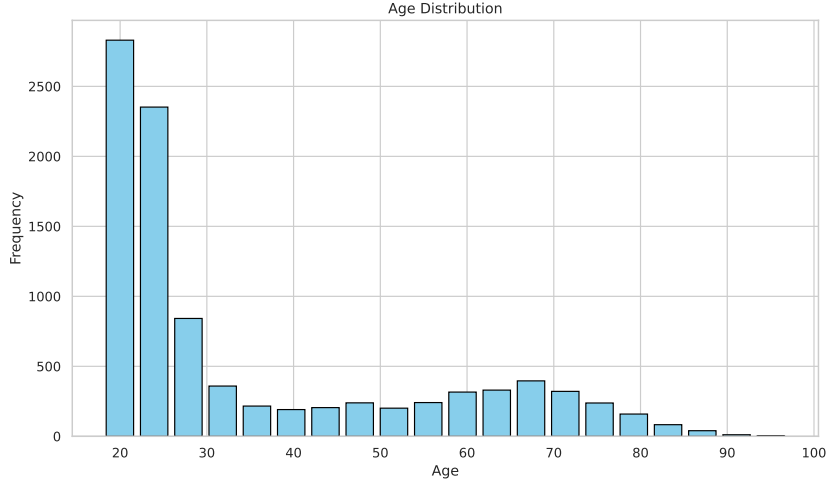


Figure 3.1: Age distribution of the dataset containing 9573 subjects

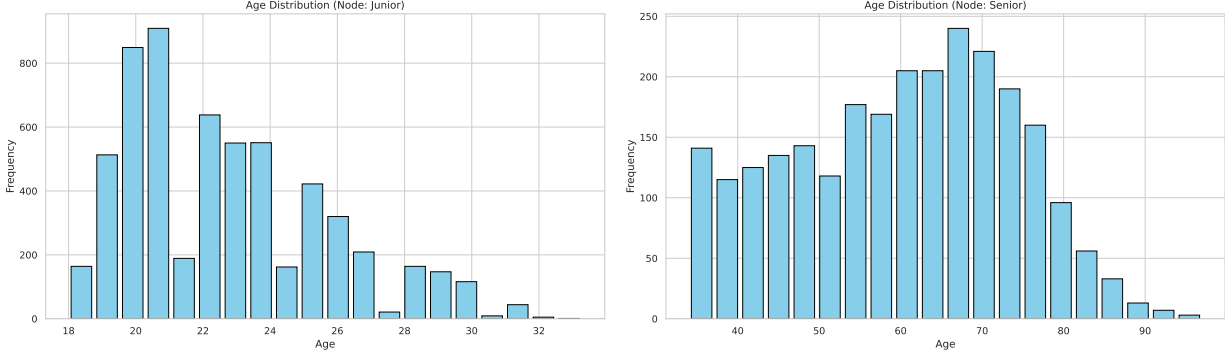
As seen in Figure 3.1, the age distribution does not seem to follow a standard Gaussian distribution but rather can be made as a mixture of two Gaussian/normal distributions. Using the `gmm` tool in Python, we fitted the age distribution to a Gaussian mixture model, which combines multiple Gaussian distributions. Since all the subject data we have is from adults, we decided to discard any age samples below 18 to ensure that no minors are included in the database. This mixture distribution can be found in Figure 3.2c. These are the distributions:

- The first normal distribution accounts for 62,5% of all subjects. It has a mean of 22,4 and a standard deviation of 2,9. A visualisation of the distribution can be found in Figure 3.2a. This will be referred to as the Junior distribution.
- The second normal distribution accounts for the remaining 37,5% of all subjects. It has a mean of 56,4 and a standard deviation of 16,2. A visualisation of the distribution can be found in Figure 3.2b. This will be referred to as the Senior distribution.

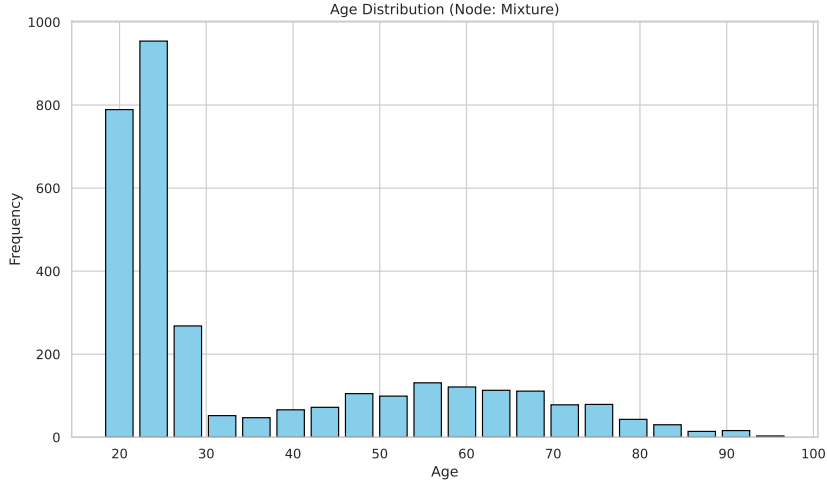
3.1.2 Data Partitioning

For these experiments, we divided the subject dataset into two subsets: a training dataset comprising 80% of the subjects and a test dataset comprising the remaining 20%. The test dataset will be used for evaluating the global models generated after every server round. Then, we distribute the training dataset into multiple client nodes (henceforth referred to as nodes). These nodes sample from the dataset according to a specific distribution. Here are the various distribution profiles:

- Original distribution profile: nodes will randomly sample from the subject dataset, thereby ensuring that all nodes have the same distribution. The node distributions can be found in Figure 3.3.
- Gaussian distribution profile: 2/3 of the nodes will randomly sample from the Junior distribution and 1/3 of the nodes will randomly sample from the Old distribution. This



(a) Normal distribution representing the Junior distribution, a younger population age group. (b) Normal distribution representing the Senior distribution, an older population age group.



(c) Mixture of the two normal distributions, showing the combined age distribution.

Figure 3.2: Age distribution of two distinct population groups and their mixture.

66,6% en 33,3% split is intended to recreate the components of the mixture distribution across separate nodes. The node distributions can be found in Figure 3.4.

- Transition distribution profile (only available for 6 nodes): We transition the nodes from the Junior distribution to the Senior distribution. We start with a node that samples 80% from Junior and 20% from Senior and we transition each time by 20% until we have a node that samples 20% from Junior and 80% from Senior. To address the class imbalance between the Junior and Senior distributions, the predominant Junior nodes will be represented at twice the frequency of the predominant Senior nodes. The node distributions can be found in Figure 3.5.

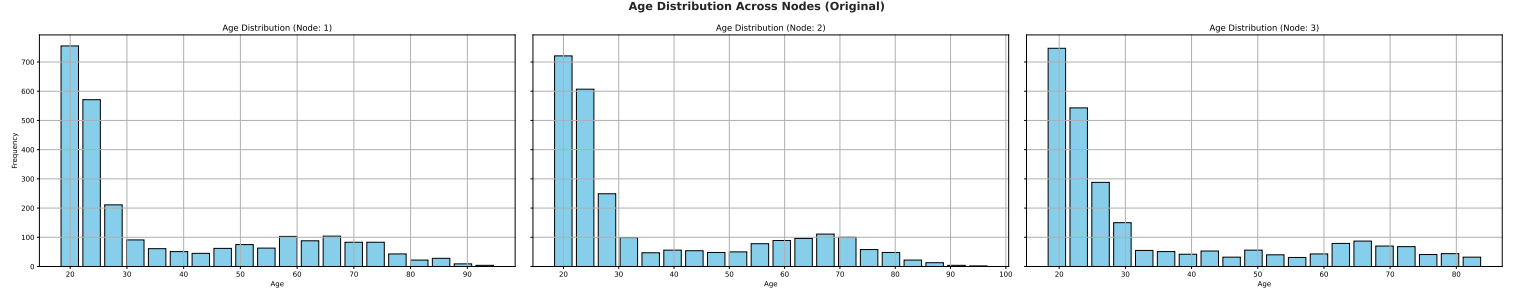


Figure 3.3: Age distributions of Original distributed nodes.

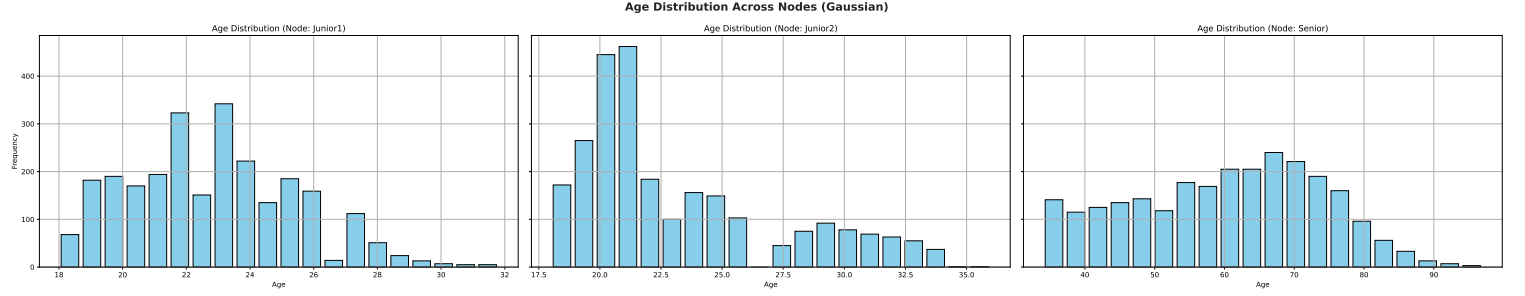


Figure 3.4: Age distributions of Gaussian distributed nodes.

3.2 Experimental Configuration

3.2.1 Model Starting Point

In Federated Learning, a server distributes the initial instance of the global model to the nodes. This model can be initialized either with random weights (RW) or with a pre-trained dataset. For our experiments, we have chosen to investigate two starting points: one using uninitialized random weights and the other using pre-trained models from David Wood (DWood) [165].

3.2.2 Federation

The training dataset is then evenly distributed across the nodes, sampled according to the aforementioned distribution profiles. We ensure that the same subject does not appear more than once in separate nodes or within the same node. Per node, we refine the previous global model by training it on the local data of the node for one epoch, which is a complete pass through the entire dataset of the node, before sending it back to the server. We combine these models using a Federated Learning aggregation algorithm (henceforth referred to as strategy) to obtain a new iteration of the global model. We evaluate this model with the test dataset and redistribute the global model. This process is repeated for 20 server rounds. We opted for 20 server rounds to prevent computational exhaustion and avoid excessive training time, while still ensuring sufficient rounds to observe improvements in the learning process.

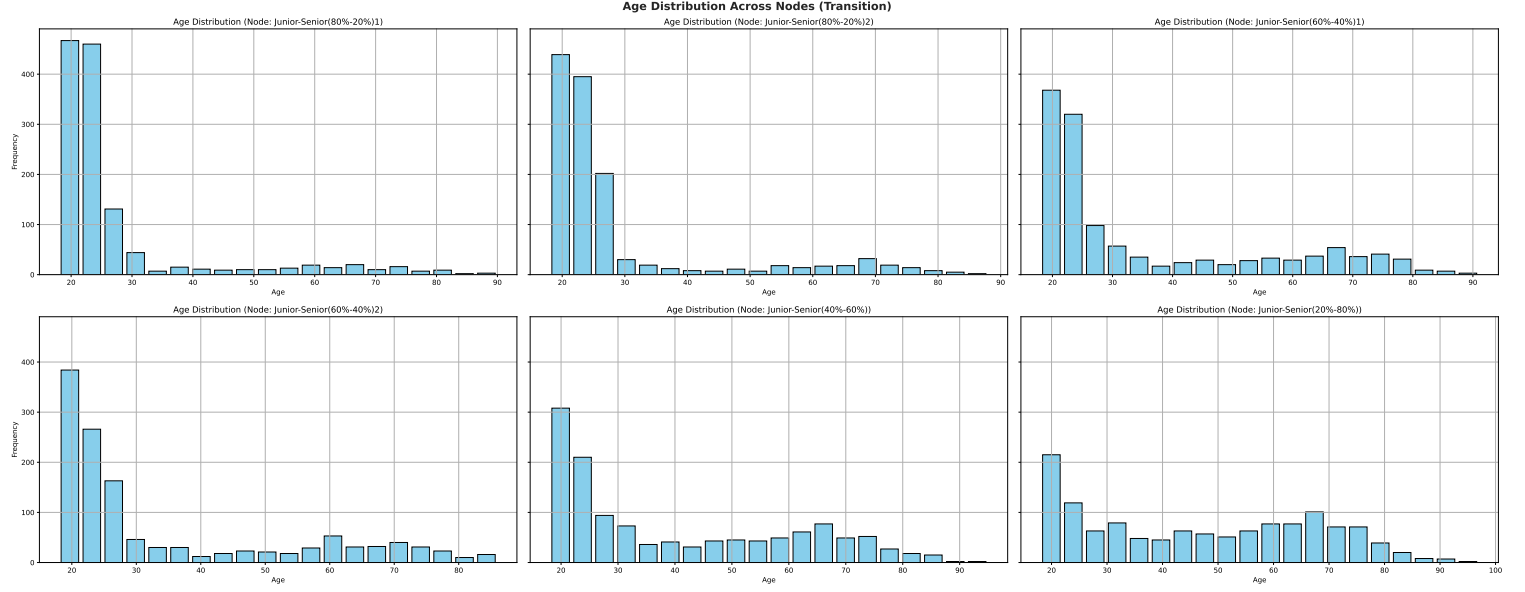


Figure 3.5: Age distributions of Transition distributed nodes.

3.2.3 Local Training

Each node locally divides their portion of the training dataset into 10 distinct data folds. This division facilitates the training of 10 separate models, yielding 10 trained models, each with its associated training loss and validation loss. All models are trained using a single epoch and optimized using the ADAM optimizer [117] with a learning rate of 1×10^{-4} . To evaluate local node performance, the average validation loss and the standard deviation of the validation losses are calculated. Subsequently, the weights of these 10 models are averaged to form the locally trained node model, which is then transmitted to the global server. The decision was made not to select the best-performing model from the individual folds. This approach avoids the potential bias introduced by "lucky splits", where particular divisions of the data might inadvertently favor certain models, thus not providing a true representation of the model's general performance. By averaging the models instead of selecting the best, we ensure a more robust and representative measure of the model's capabilities across a variety of data distributions.

3.2.4 Model Aggregation

For the aggregation of model weights from each node, two strategies are employed: Federated Averaging (FedAvg) and FedProx, the latter incorporating a proximal term (μ) set to 1. To benchmark the effectiveness of these federated learning algorithms, a theoretical baseline is established through the training of a model using 10-fold model training and cross-validation. This method mirrors the local retraining process, utilizing the complete training dataset. Similar to the local training protocol, this model is trained for an amount of epochs corresponding to the number of server rounds.

3.2.5 Plotting

The performance of the federated models, utilizing either FedAvg or FedProx, is assessed by plotting their respective losses alongside those of the centralized baseline model. This comparison aids in discerning the relative efficiencies and potential drawbacks of the federated learning strategies compared to traditional centralized methods. In addition to plotting centralized losses, the local training losses for each node are also visualized, accompanied by a 95% Confidence Interval to provide statistical significance to the observations. To account for the variability inherent in the different node distributions, the name of the distribution strategy employed is displayed above each subplot, clearly identifying the strategy associated with each set of results. These nodes have the following names:

- Original distribution (3 Nodes): Node 1, 2 and 3
- Gaussian distribution (3 Nodes): Node Junior1, Junior2 and Senior
- Original distribution (6 Nodes): Node 1, 2, 3, 4, 5 and 6
- Gaussian distribution (6 Nodes): Node Junior1, Junior2, Junior3, Junior4, Senior1, Senior2
- Transition distribution (6 Nodes): Node Junior-Senior(80%-20%)1, Junior-Senior(80%-20%)2, Junior-Senior(60%-40%)1, Junior-Senior(60%-40%)2, Junior-Senior(40%-60%) and Junior-Senior(20%-80%)

3.2.6 Hardware & Software Setup

These experiments were run on a computer from the AIMS laboratory, the full specifications of this computer can be found in Figure 3.6. We have used the Brain Age model architecture from Silva [169] and Wood et al. [165] and the Flower framework [150] to implement the Federated Learning aspect. We have used the Flower VCE, which utilizes multicore programming using Ray to simulate nodes that each independently hold part of the training dataset. These experiments were conducted in a Linux environment, though they should be supported on other operating systems as well.

Figure 3.6: Specifications of the Computer System

Component	Specification
CPU	Intel Core i9-12900F
RAM	128 GB DDR4 at 4800 MHz
GPU	NVIDIA GeForce RTX 4090
Storage	Primary: WD 1 TB SSD (Operating System) Secondary: 4 x WD 18 TB HDD in RAID configuration (54 TB total)
Network Connection	1 Gbps Ethernet

Chapter 4

Results

Before discussing the results, we need to clarify the legend names used for each line in the graphs. The attributes, listed in order of appearance and separated by underscores in the legend names, are as follows:

- Strategy: FedProx or FedAvg
- Starting Point: Random Weights (RW) or David Wood (DWood)
- Distribution: Original, Gaussian, or Transition
- Node Amount: 3 or 6 Nodes

In some graphs, the node amount might not be included in the legend but will be indicated in the title and caption, as we never directly compare the loss lines of 3 nodes with those of 6 nodes. If a model is centralized, it will only contain the starting point in its legend name, excluding strategy, distribution and node amount.

4.1 3 Nodes Experiment

From Figure 4.1, we can observe the following:

- The global models produced from the nodes which follow the Original distribution have a learning loss which is either similar or slightly worse than the centralized baseline. FedProx performs worse than FedAvg in this scenario.
- The global models produced from the nodes which follow the Gaussian distribution have the highest amount of learning loss. The Gaussian models end up performing worse than the initial David Wood starting point.
- In each server round, nearly all models show improved performance, with the exceptions being the FedProx Gaussian and the FedAvg Gaussian models starting from the David Wood model.
- Models aggregated using the FedProx strategy generally exhibit a higher learning loss compared to those using FedAvg in nearly every scenario, except for the case of Gaussian distributed nodes starting from the David Wood model.



Figure 4.1: Global model losses over server rounds of Federated Learning (across 3 nodes). Calculated using the test dataset.

From Figure 4.2, we can observe the following:

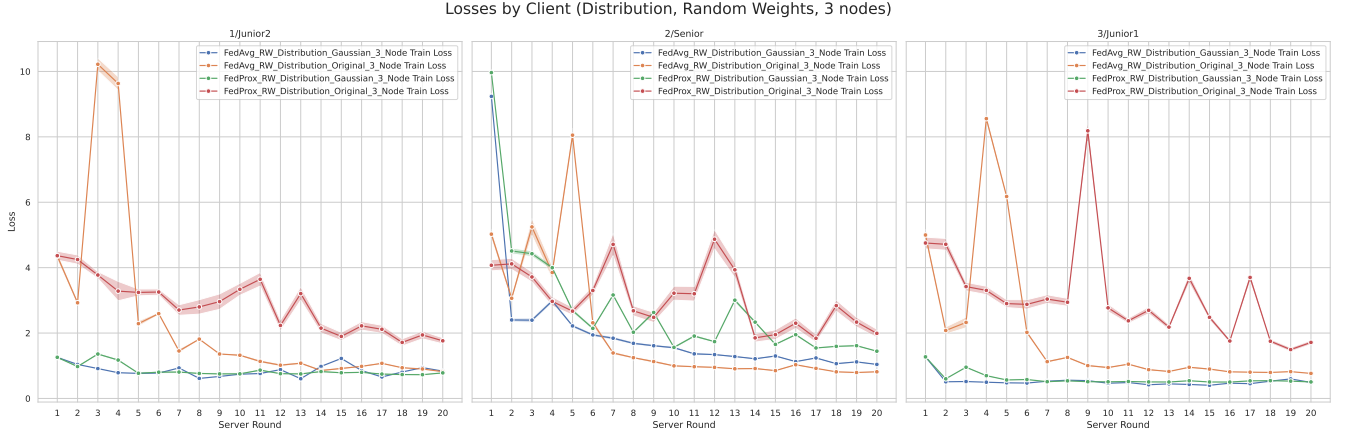
- FedProx has a higher local loss than FedAvg across every single possible configuration.
- In the nodes with a Junior distribution, the Gaussian local models have a better performance compared to their Original distributed node counterpart. However, when looking at the node with the Senior distribution, this is no longer the case.
- As shown by the 95% confidence intervals, the Junior Gaussian local models seem to have less variability in the losses compared to the Original local models. The Senior Gaussian local models appear to have at least the same, if not higher, level of variability between losses compared to the Original local models.

The previous experiment was done using 3 nodes and 2 distribution profiles. We then scaled the experiment to 6 nodes using Original, Gaussian and Transition distribution profiles.

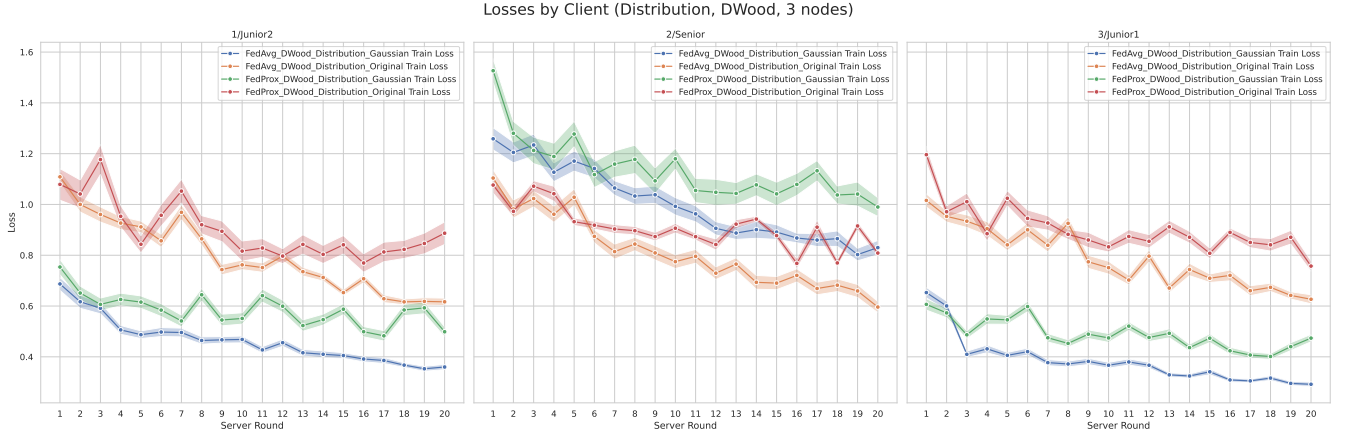
4.2 6 Nodes Experiment

From Figure 4.3, we can see some consistency with the previous results, enriched with some new observations:

- The nodes with Gaussian distribution continue to exhibit the poorest performance in terms of aggregated model outcomes.
- Every algorithm/distribution combination that has also been examined in the 3 nodes experiment, as shown in in Figure 4.1, performs slightly worse in the 6 nodes experiment.
- The Transition distributed nodes have a comparable performance to the Original distributed nodes despite having more variety in how the age distribution is handled. However, despite



(a) Starting from random weights.



(b) Starting from a David Wood pre-trained model.

Figure 4.2: Local model losses over server rounds of Federated Learning (across 3 nodes). Calculated using the average training losses in every node.

the near identical amount of loss, the Original distribution generally yields the least amount of loss.

- Starting with the David Wood model, FedProx outperforms FedAvg up until server round 11, after which FedAvg achieves a lower loss. A similar pattern of learning instability is observed when comparing FedAvg and FedProx in the Original distribution. When starting from randomly weighted models, Transition nodes trained with FedProx demonstrate the best performance among FedProx-trained models, but they do not generally outperform the FedAvg models, consistent with previous findings. In both scenarios, FedProx does not yield superior performance.

The previous depiction of individual node losses was somewhat challenging to interpret; therefore, for the sake of simplicity, nodes exhibiting similar distribution profiles have been consolidated into a single learning curve through the averaging of their losses. Consequently, nodes with an original distribution now generate a single learning curve, while nodes with a Gaussian distribution

yield two, and those with a transition distribution produce four. To enhance the clarity and presentation of the graph, it has been arranged to display each strategy separately. As a result, Figure 4.4 now includes four distinct subplots. The previous representation of the local node graph can be found in Figure A.1.

Based on the analysis presented in Figure 4.4, the following observations can be made:

- When initiated with the pre-trained David Wood model, FedProx exhibits a lower local loss in the initial rounds compared to FedAvg, though it does not optimize as efficiently in subsequent server rounds. Interestingly, when comparing the losses across graphs, we observe that the local loss trajectories remain largely unaffected by the choice of aggregation algorithm. Both FedAvg and FedProx demonstrate similar loss distributions across the rounds, with the notable exception of FedProx on the Junior-Senior (40%-60%) distributed node, where stability is notably lacking, and loss metrics fluctuate significantly.
- Beginning with models initialized with random weights, both FedAvg and FedProx display nearly identical learning curves after 20 server rounds. However, an exception is observed with FedProx on an Original distributed node.
- In terms of local performance, Junior nodes consistently outperform Senior nodes. This pattern aligns with results from the three-node experiment.

4.3 Model Performance

In order to measure the effectiveness of the brain age prediction models produced here, we have provided the Mean Absolute Error (MAE) of the Federated Models after 20 rounds of federation. To facilitate a fair comparison with a theoretical best algorithm, we also included the MAE of the centralized baseline models after 20 epochs. These MAE values can be found in Table 4.1.

It is expected that most models will underperform relative to these benchmarks, as the baseline models have full access to the entire training dataset, resulting in more accurate models. This has been observed in practice. The results from Section 4.1 and Section 4.2 reveal that nodes with the Original distribution yield the most effective federated models, except in scenarios involving six nodes starting from random weights, where the Transition distribution profile proves more advantageous.

We have included scatterplots of the federated models after 20 server rounds. In these plots, the x-axis represents the actual age, while the y-axis represents the predicted age. Additionally, the identity line is shown to indicate the ideal scenario where the points should align. These scatterplots are presented in Figure 4.5, Figure 4.6 and Figure 4.7.

Strategy	Starting Point	Distribution Profile	Amount of Nodes	MAE
FedAvg	David Wood	Gaussian	6	10.88
FedAvg	David Wood	Original	6	3.37
FedAvg	David Wood	Transition	6	3.58
FedProx	David Wood	Gaussian	6	8.04
FedProx	David Wood	Original	6	3.62
FedProx	David Wood	Transition	6	3.93
FedAvg	Random Weights	Gaussian	6	14.73
FedAvg	Random Weights	Original	6	3.99
FedAvg	Random Weights	Transition	6	3.61
FedProx	Random Weights	Gaussian	6	15.67
FedProx	Random Weights	Original	6	9.53
FedProx	Random Weights	Transition	6	5.37
FedAvg	David Wood	Gaussian	3	10.31
FedAvg	David Wood	Original	3	3.04
FedProx	David Wood	Gaussian	3	7.21
FedProx	David Wood	Original	3	3.41
FedAvg	Random Weights	Gaussian	3	10.32
FedAvg	Random Weights	Original	3	3.46
FedProx	Random Weights	Gaussian	3	14.21
FedProx	Random Weights	Original	3	7.00
Centralized	Random Weights	N/A	N/A	3.40
Centralized	David Wood	N/A	N/A	2.94

Table 4.1: Summary of MAE values based on Strategy, Starting Point, Distribution Profile, and Node Amount after 20 server rounds/epochs. Bold values indicate lowest MAE in the respective group.

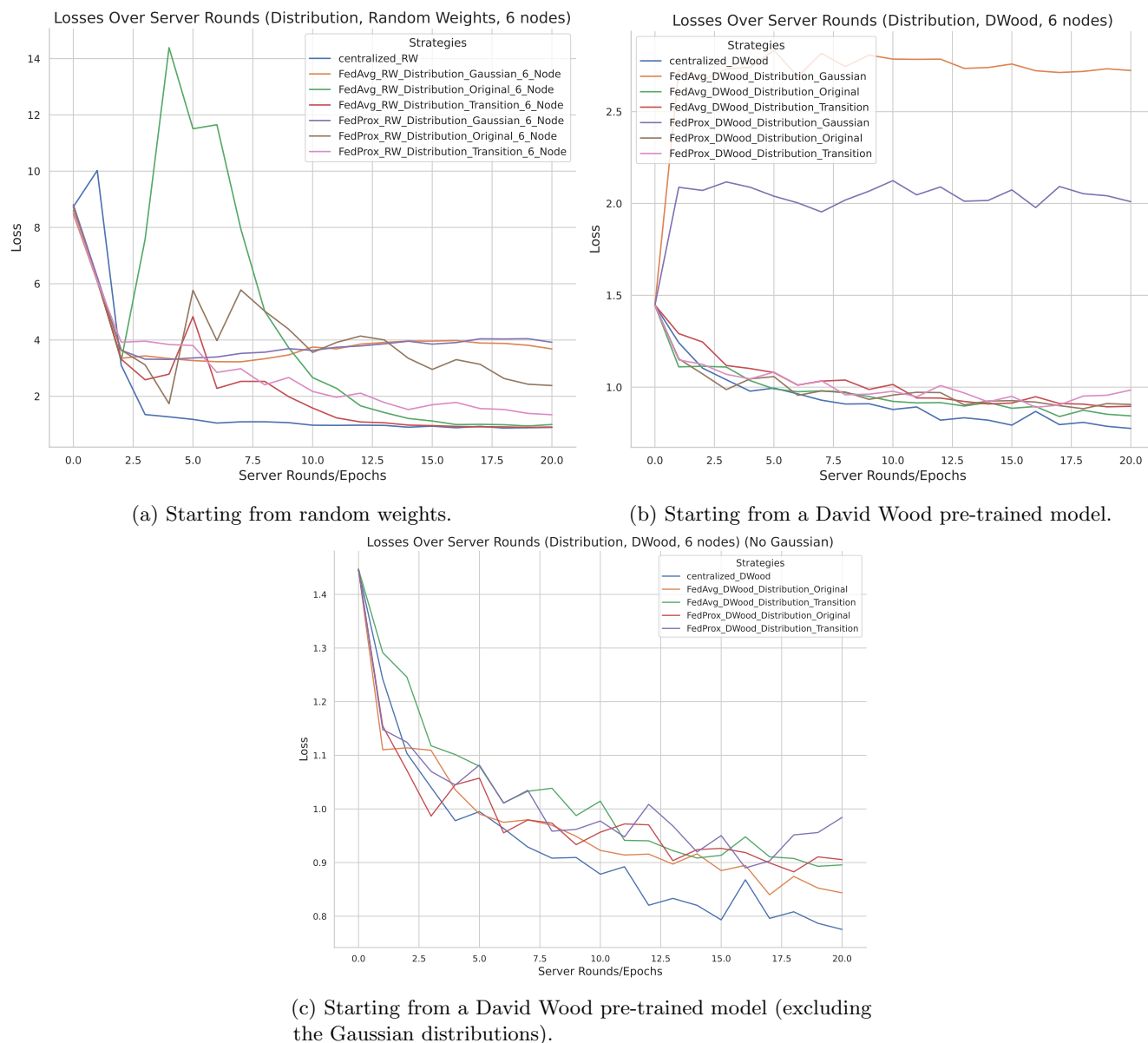
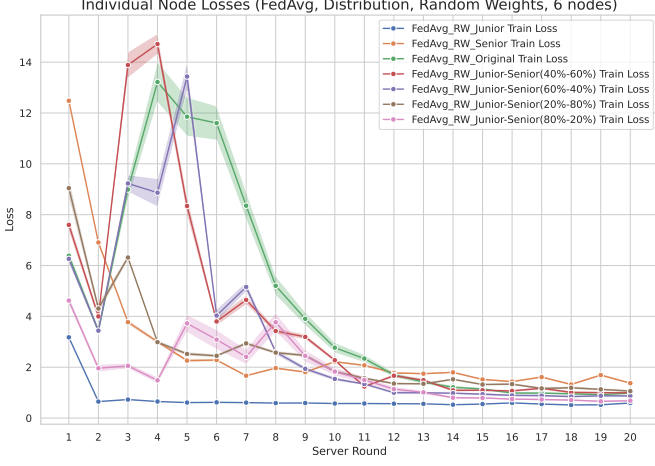
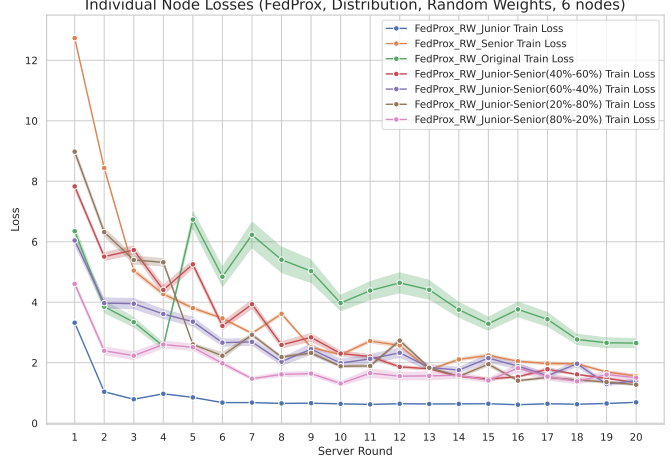


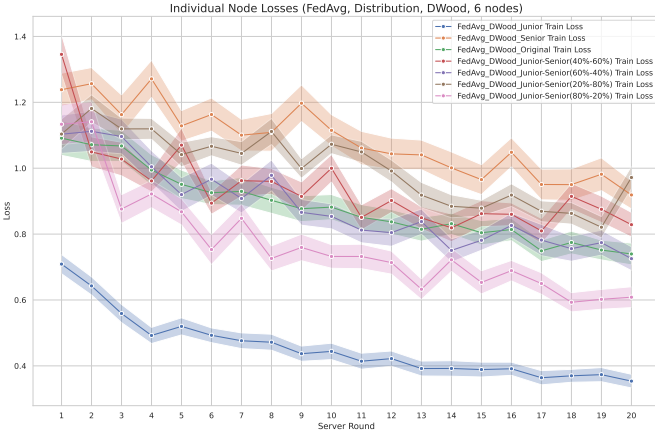
Figure 4.3: Global model losses over server rounds of Federated Learning (across 6 nodes). Calculated using the test dataset.



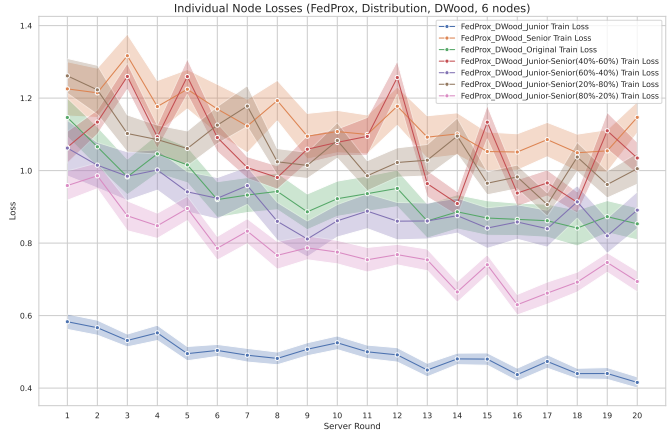
(a) Starting from random weights - FedAvg Algorithm.



(b) Starting from random weights - FedProx Algorithm.



(c) Starting from a David Wood pre-trained model - FedAvg



(d) Starting from a David Wood pre-trained model - FedProx Algorithm.

Figure 4.4: Individual local node losses based on aggregation algorithm and initialization. Calculated using the average training losses in every node and then averaging over the nodes with the same distribution.

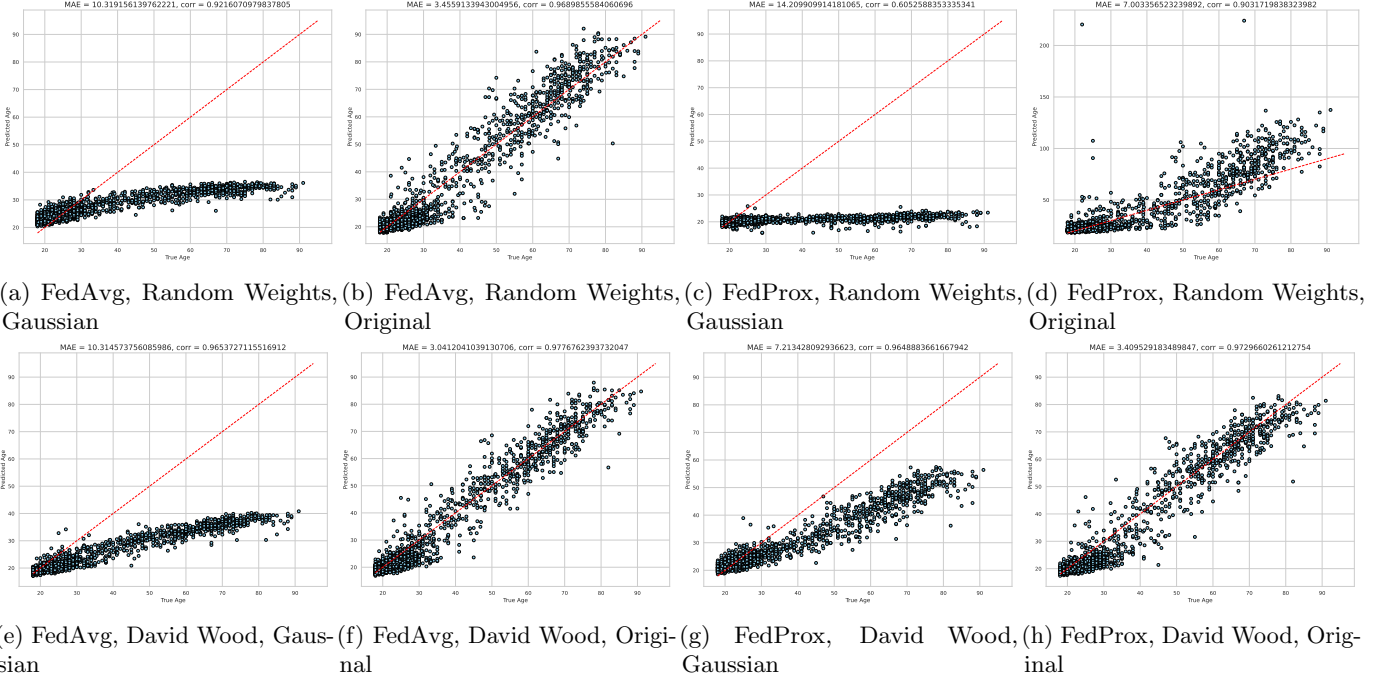


Figure 4.5: Scatterplots of age predictions for models with 3 nodes.

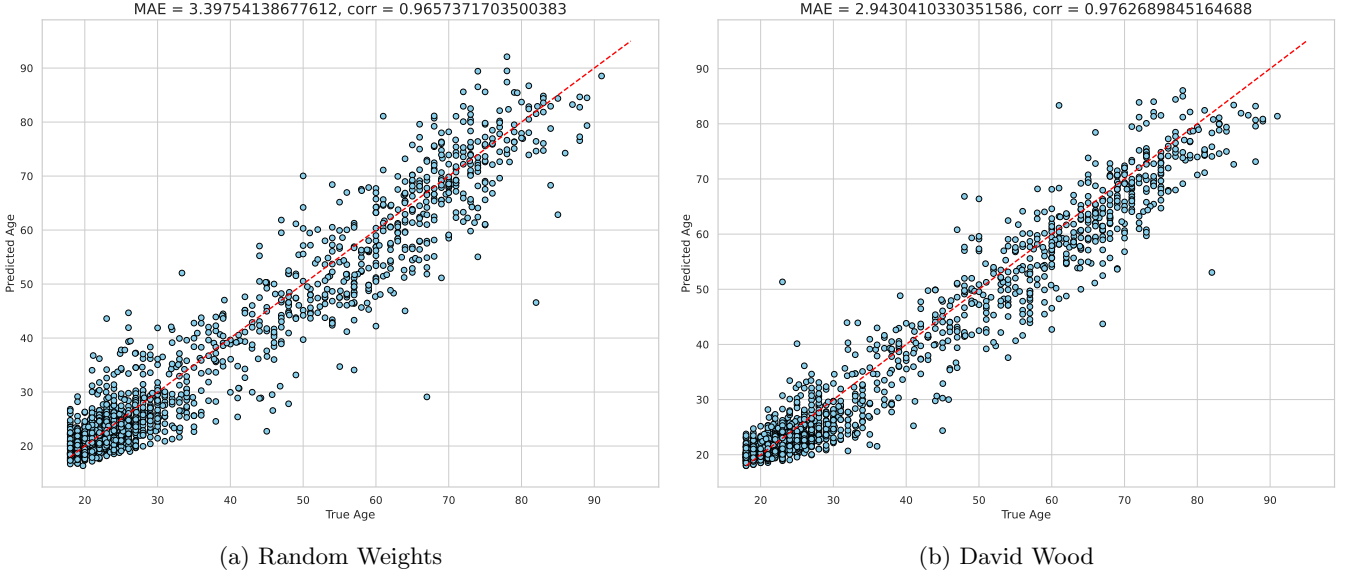


Figure 4.6: Scatterplots of age predictions for the centralized models.

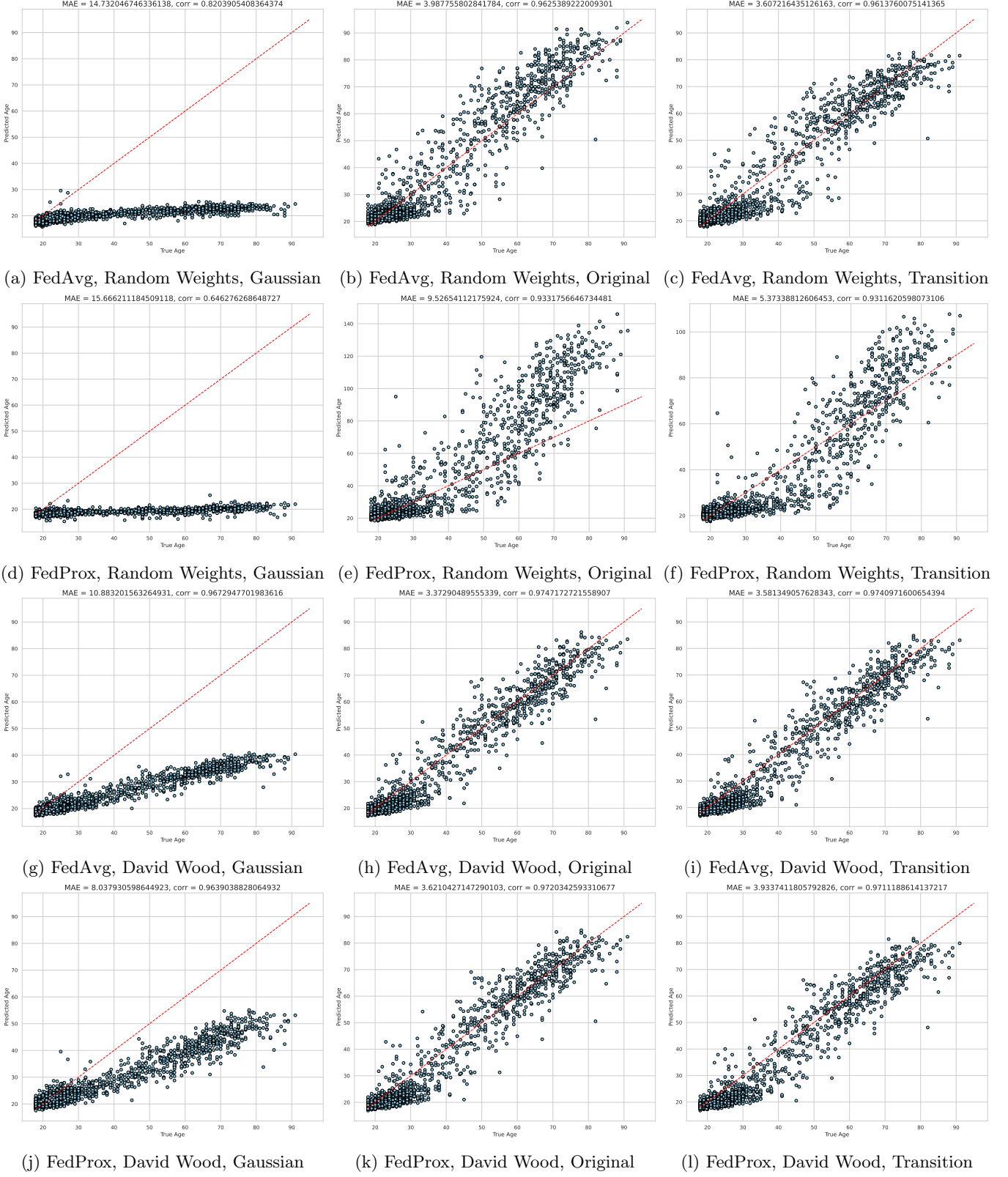


Figure 4.7: Scatterplots of age predictions for models with 6 nodes.

Chapter 5

Discussion

5.1 Overview of Experimental Findings

5.1.1 Summary of Key Results

Our study extensively evaluated the impact of age distribution on the effectiveness of federated learning models, particularly within nodes with varied demographic profiles. We observed that nodes with Transition distributions, which incorporate a mix of younger and older data subjects, generally mirrored the performance of the nodes with Original distributions, where the data was evenly spread across age groups. This was in stark contrast to Gaussian distributed nodes, characterized by skewed age distributions and underrepresentation of age groups and subsequently higher model learning losses. These findings are critical as they highlight the practical challenges and potential of using balanced versus unbalanced age distributions in federated learning applications.

5.1.2 Contextual Comparison

Comparatively, our results highlight the critical need for broad demographic representation in training datasets to mitigate bias and enhance the robustness of predictive models. This perspective aligns with Section 2.5, which advocates for inclusive data practices to improve algorithmic fairness and effectiveness across various applications. Furthermore, our analysis of FedProx versus FedAvg offers insights into how different federated learning algorithms handle age diversity, revealing inherent strengths and limitations within each approach that are crucial for their application in real-world scenarios.

5.2 Interpretation of Results

5.2.1 Comparative Analysis of Brain Age Models

We will undertake a comparison of the final federated models with several established examples within the Brain Age domain. It is important to note that the Mean Absolute Error metric, which we will use for this comparison, is not a universal measure. This limitation arises because the MAE is calculated based on different test sets, making direct comparisons between models somewhat problematic. Therefore, while we will provide a comparative analysis of the MAE

values, the results should be interpreted with caution due to the inherent variability in the test datasets. This approach will help highlight the relative performance of our federated models while acknowledging the contextual differences in evaluation metrics.

The prediction model from Silva [169] produces a MAE of 3.38 starting from random weights and 3.40 when starting from the David Wood pre-trained model. When compared directly to our centralized baseline models, the model initialized with random weights shows a negligible improvement, with a MAE reduction of only 0.01. In contrast, the model starting from the David Wood pre-trained model demonstrates a substantial improvement: our model achieves a 0.46 decrease in loss compared to Silva’s model. Numerous factors influence these outcomes; notably, Silva employs a smaller dataset encompassing 6,326 subjects, which includes gender data utilized for bias correction. Additionally, Silva adopts a variable training duration across numerous epochs, with a decaying learning rate, unlike our fixed duration of 20 epochs.

Wood et al. [165] evaluates their brain age prediction model using multiple datasets, thus facilitating an assessment of performance variability across different datasets and enhancing the robustness of the findings concerning the model’s adaptability to unknown data. Nonetheless, to align with our study’s focus on a single composite dataset, we will limit our discussion to the MAE measurements from these conditions. Wood et al. [165] further investigates the impact of different brain scanners, a factor that extends beyond the scope of this thesis. David Wood’s research utilizes two datasets: one from King’s College Hospital NHS Foundation Trust (KCH) with 8,072 subjects, and another from Guy’s and St Thomas’ NHS Foundation Trust (GSTT) comprising 11,735 subjects. The MAEs from these datasets are as follows:

Dataset	MAE
KCH + GSTT	2.97
KCH	3.12
GSTT	3.05

Table 5.1: Mean Absolute Errors for different datasets in the Brain Age prediction model study of Wood et al. [165]

We also examined insights from the review study by Kumari and Sundarrajan [183]. While the accuracy of these brain age models is variable, it is noted that most brain age models utilizing large datasets achieve a MAE ranging between 3 and 5. This indicates that the Federated models developed using Original and Transition distribution profiles are sufficiently accurate to merit consideration within the research domain of brain age prediction. For a comprehensive analysis, we direct readers to the detailed review provided by Kumari and Sundarrajan [183].

5.2.2 Age Distribution

The experiments demonstrated that non-IID data distributions, particularly concerning age, significantly affect the training dynamics and effectiveness of federated learning models. Transition nodes, which blend data from diverse age groups, effectively counter the negative impacts observed in Gaussian nodes, which are limited to specific age groups. This balance enabled Transition nodes to approach the performance of nodes following the Original distribution, underscoring the importance of representative demographic sampling in training datasets. These results suggest that integrating a variety of age groups within each node could be a crucial strategy for developing robust and accurate federated learning models, particularly in fields sensitive to demographic factors such as healthcare and social research.

5.2.3 Aggregation Algorithm

In the direct comparison between FedProx and FedAvg, it was evident that FedAvg tends to be more flexible and adaptive, capable of leveraging the rich, diverse data provided by Transition nodes. Conversely, FedProx, designed to constrain updates to remain close to the global model, proved less effective in capturing the full statistical diversity of the population, as evidenced by its performance in most scenarios. However, it excelled in environments with significant age disparities as seen in its performance in Gaussian distributions starting from the David Wood model. This is because FedProx’s conservative update mechanism can prevent overfitting to outlier data distributions, such as those seen in Gaussian nodes, thereby maintaining a more stable global model across training rounds. However, this strength also manifests as a limitation when it hinders the model’s ability to fully adapt to diverse data inputs, leading to potentially suboptimal performance in less controlled environments. From these results, FedProx appears to be the logical choice in scenarios where the initial model is already well-optimized, and where data disparity could potentially degrade the model’s performance, as evidenced in Figure 4.1b and Figure 4.3b.

5.2.4 Model Starting Point

The initial conditions of the models did not produce particularly notable results. Models that began with random weights achieved competitive performance only when the data distribution across nodes was relatively uniform and when utilizing the FedAvg algorithm. This is because FedProx tends to resist significant changes to the global model, even when the starting point is not a strong reference point. In all scenarios, starting from a pre-trained model proved advantageous, as it provided a strong initial performance baseline and facilitated further improvements with additional iterations.

5.2.5 Amount of Nodes

In Figure 5.1, we observe that increasing the number of nodes directly impacts model performance, with more nodes leading to a noticeable decrease in effectiveness. This trend highlights the challenges of federated learning, where data fragmentation across multiple nodes hinders the model’s learning ability. As nodes increase, data distribution variance can increase, resulting in higher MAE values and reduced accuracy. It is important to note that we are distributing the existing training dataset across more nodes, not adding new data. In practice, welcoming new institutions typically brings additional data, which can enhance the federated model’s theoretical best performance. A more in-depth examination of future experiments addressing this issue will be provided in Section 5.4.

5.3 Challenges & Limitations

The first challenge we encountered was the sheer volume of publications on the subject of Federated Learning. This is reflected in the extensive coverage in Chapter 2, as there are many promising yet unfocused papers on FL. Given that the concept of FL was not formally introduced until 2016, there has been only eight years of research at the time of writing, placing FL in its pioneering stage. Additionally, there is little consensus on which algorithms are essential for research. FedAvg is a mandatory algorithm in every relevant FL paper due to its precedence and ease of implementation. However, the second most commonly used algorithm could be either SCAFFOLD or FedProx, as both effectively mitigate client drift. Other techniques and algorithms show promise, but a

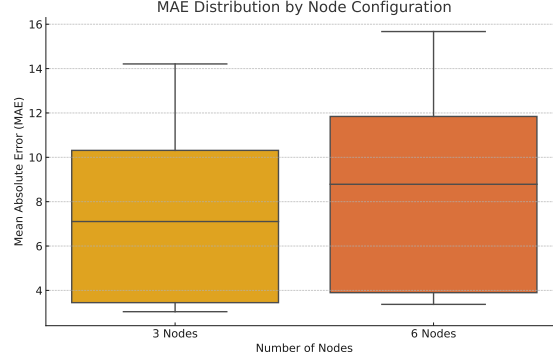


Figure 5.1: Comparison of the MAE of the models across different node configurations, including Gaussian and Original models. Transition models are excluded due to the unavailability of the Transition distribution profile for the 3-node setting.

comprehensive review paper comparing their performance would be a valuable addition to the field and could help guide new researchers in adapting FL to their specific domains.

In these experiments, we used only one dataset across a single run. This approach may limit the thoroughness of our comparisons, especially when contrasted with previous brain age studies. The single run of experiments also precluded the inclusion of confidence intervals in the global loss graphs, which could have explained and averaged out any results influenced by random behavior. Several factors contributed to the structure of the experiment:

- There was a mistaken belief that the dataset with 6,326 subjects was a subset of the larger dataset with 9,573 subjects. This confusion arose from the absence of the age column in the larger dataset, leading to the incorrect assumption that the smaller dataset was a filtered version of the larger one. Upon investigating the original dataset, this assumption was disproven. A table detailing the origin of the 6,326 subjects is provided in Table 5.2. As previously mentioned, Silva [169] used this dataset for their experiments.
- Due to the lack of background in MRI scans and the preprocessing of such data, it was unfeasible to compile and arrange an additional dataset within the given timeframe.
- Running all the experiments on the computer required approximately two weeks of computation. These computations had to be repeated once due to a bug that caused the same patients to appear in multiple nodes and even within the same nodes. Parallelizing these experiments was not feasible because the GPU was required to power the experiments, and PyTorch cannot utilize a GPU that is already in use by another process.

During the later stages of writing this thesis, we discovered that the dataset included not only healthy controls but also subjects with pathological conditions and subjects without a pathological diagnosis. Had this been known earlier, it might have influenced the number of data points used in the experiments. Specifically, we could have either retained only the confirmed controls or removed the positively diagnosed subjects from the dataset. This adjustment would have been particularly relevant as the focus of this research was to create an accurate brain age prediction model that could reflect the actual age of healthy controls and identify any learning disparities.

The original plan was to test the federated learning process not only on a single computer but also

Table 5.2: Number of images per open-source dataset in the 6,326 subject dataset

Source	Name/Description	Images
ABIDE	Autism Brain Imaging Data Exchange	185
ABIDE II	Autism Brain Imaging Data Exchange II	108
ADHD200	Attention-Deficit/Hyperactivity Disorder 200	44
CamCAN	Cambridge Centre for Ageing and Neuroscience	653
COBRE	Center for Biomedical Research Excellence	74
CORR	Consortium for Reliability and Reproducibility	1193
Openneuro	Open platform for sharing and analyzing neuroimaging datasets	324
INDI	International Neuroimaging Data-Sharing Initiative	1062
IXI	Information eXtraction from Images	584
OASIS_3	Open Access Series of Imaging Studies version 3	1085
PDD	Pediatric Developmental Disorder	16
RocklandSample	Sample from 1000 Functional Connectomes Project	998

across multiple computers. The AIMS lab generously provided us with access to three computers for this research. After evaluating Flower and the available computers, we opted to use Flower VCE and a single computer to emulate the nodes working independently. The reasons for this decision are as follows:

- The first computer is an older build that is heavily utilized by the researchers of the AIMS group. It has a powerful but dated Nvidia Titan X GPU. We decided not to use this computer to avoid interfering with other experiments conducted by the AIMS lab, as these federated learning experiments would take two weeks and render the GPU unavailable for other machine learning tasks.
- The second computer lacks a GPU. Since these experiments involve training deep neural networks, a GPU is mandatory to minimize the experiment duration.
- The third computer (specifications detailed in Figure 3.6) was chosen to run all the aforementioned experiments due to its suitable hardware configuration.

To simplify our approach, we chose to use only the David Wood model constructed from random seed 2. Although we had the option to select from seeds 2, 31, 35, 60, and 67, we decided to use a single seed for consistency. Utilizing multiple seeds would enable the plotting of confidence intervals on the global model losses, but this would only be applicable when starting from the David Wood models.

5.4 Future Research Directions

Future research should explore several key areas to enhance the understanding and application of federated learning in brain age prediction models. One promising direction is to conduct experiments multiple times without a predetermined random seed, allowing for the calculation and plotting of confidence intervals on the global model loss. This approach would provide a more robust assessment of model performance and variability. If future researchers choose to abandon models starting from random weights, they could plot confidence intervals by running the experiments on all David Wood models. This approach would maintain the predetermined random seed for the rest of the federated process, ensuring consistency in the experimental setup.

FedProx was tested using a single proximal term of 1. Observing the effect of varying the proximal term on the performance of the federated learning models could provide valuable insights into how deviation from the global model affects the prediction performance. Additionally, integrating new federated learning strategies, such as Scaffold, FedOpt and FedCurv, could yield further understanding. These strategies might offer improvements over existing methods by addressing specific challenges associated with federated learning, such as client drift and model convergence.

Another important area for investigation is the effect of the number of nodes on model performance. While the current study examined configurations with 3 and 6 nodes, future experiments should consider a wider range of node configurations. This could include increasing the number of nodes and incorporating additional data, which may enhance the model's ability to generalize across diverse populations.

Moreover, running experiments with separate datasets would provide a more comprehensive understanding of model performance across different data distributions. For instance, utilizing the smaller dataset of 6,326 subjects, which has been pre-processed and is available, could offer insights due to its differing age distribution compared to the larger dataset used in this study. Adapting other datasets that comply with the BIDS standard [182] and include patient age information would further enrich the analysis, though this would require expertise in MRI data preprocessing.

In these experiments, we investigated the effects of age distribution and how the heterogeneity of this variable impacts model performance, along with potential mitigation strategies. It would be interesting to extend these experiments to other variables such as gender, education, race, and more.

Exploring the use of synthetic data to enhance age representation within nodes is another promising avenue. As demonstrated in Section 4.2, sufficient age distribution can mitigate the effects of non-IID data, suggesting that synthetic data could help balance node distributions and improve model performance.

The initial aim of this thesis was to investigate blockchain-based systems for brain age federated learning. These systems could potentially offer benefits such as incentive mechanisms, enhanced security, and decentralization. However, given the early stage of this subdomain within federated learning, it was considered too extensive for this project. Future researchers interested in pursuing blockchain-based federated learning can refer to the pull request on the Flower GitHub ¹, which implements a blockchain-based federated learning solution using Ethereum.

Although we have explored the concepts of further protecting model weights using the techniques listed in Section 2.3.4, we have not applied any of these techniques in our experiments. It could be insightful to examine the impact on model performance when these security protocols are implemented. However, as noted in Section 2.6.3, Flower lacks these security measures and may require additional implementation to incorporate them. We have also forgone any deployment features by choosing Flower, but if future work aims to deploy a production FL model for brain age prediction, it could be interesting to implement these models in NVFlare instead of Flower due to NVFlare's increased focus on healthcare and its broader range of options for deployment, monitoring, and privacy-preserving methods.

Finally, re-running the experiments using only healthy controls could refine the model's accuracy in predicting brain age for healthy populations. This approach would eliminate potential confounding

¹<https://github.com/adap/flower/pull/1835>

factors introduced by pathological conditions and provide a clearer understanding of the model's capabilities in a controlled setting.

By pursuing these future research directions, the field can advance towards developing more accurate and generalizable federated learning models for brain age prediction and other applications sensitive to demographic factors.

5.5 Theoretical and Practical Implications

The findings from this study have significant theoretical and practical implications for the development and deployment of federated learning models in the domain of brain age prediction. The necessity of employing domain adaptation techniques to ensure that all nodes possess sufficient age representation, regardless of the skewness in data distribution for specific age groups, is paramount. This approach mitigates the adverse effects of non-IID data and enhances the robustness of federated learning models. The study underscores the importance of selecting appropriate algorithms to manage data skewness and client drift. Algorithms like FedProx, which are designed to correct client drift, show particular effectiveness when starting from a robust pre-trained model such as David Wood's brain age model. This suggests that initializing with a pre-trained model can significantly enhance performance, especially in skewed data scenarios. Ensuring varied age representation across nodes is also crucial; FedAvg performs optimally with diverse age distribution, while FedProx offers stability in scenarios with heavily skewed data by maintaining a consistent global model. Depending on the starting point of the federation, this stability can be a desirable trait.

From a practical standpoint, starting with a well-established pre-trained model like David Wood's brain age model offers a strong baseline and facilitates further improvements, making it more efficient than starting from scratch. Researchers should consider the specific characteristics of their data when selecting a federated learning strategy. For datasets with varied age representation, FedAvg is recommended due to its adaptability, whereas in environments with significant data skewness, FedProx offers more reliable performance. Additionally, the use of new data to enhance age representation within nodes is a practical strategy to counteract the negative impacts of non-IID data and improve overall model performance. In conclusion, this study highlights the critical role of domain adaptation techniques and strategic algorithm selection in federated learning for brain age prediction, providing a solid foundation for future research and practical implementations in the field.

Chapter 6

Conclusion

This thesis has explored the integration of Federated Learning (FL) within the domain of brain age prediction, focusing on structural MRI scans. The investigation has demonstrated that by leveraging FL, medical institutions can collaboratively develop predictive models while ensuring data privacy and compliance with stringent regulations such as GDPR and the California Consumer Privacy Act. By adopting a Federated Learning framework (Flower), this research marks a pioneering step in utilizing FL for brain age estimation, addressing the intrinsic challenges of non-IID data distributions. Our findings reveal that implementing domain adaptation techniques to ensure age representation across all nodes mitigates the non-IID challenges, enhancing model performance. Additionally, the study highlights the effectiveness of using pre-trained models, such as those developed by Wood et al. [165], particularly when employing algorithms like FedProx, which are designed to correct client drift. This approach proves advantageous in scenarios with highly skewed data distributions. Furthermore, the research underscores the potential of Federated Learning to deliver models competitive enough for practical application, demonstrating that federated frameworks can indeed be successfully applied to neuroimaging tasks. The implications of this work extend beyond brain age prediction, suggesting broader applications of FL in healthcare, where data privacy and collaborative learning are paramount. This thesis serves as a foundational effort in merging theoretical advancements with practical applications in medical imaging, encouraging further exploration and development in these domains. Future researchers are encouraged to build on this groundwork by further exploring FL strategies and expanding the scope of brain age prediction models to improve their robustness and utility in clinical settings.

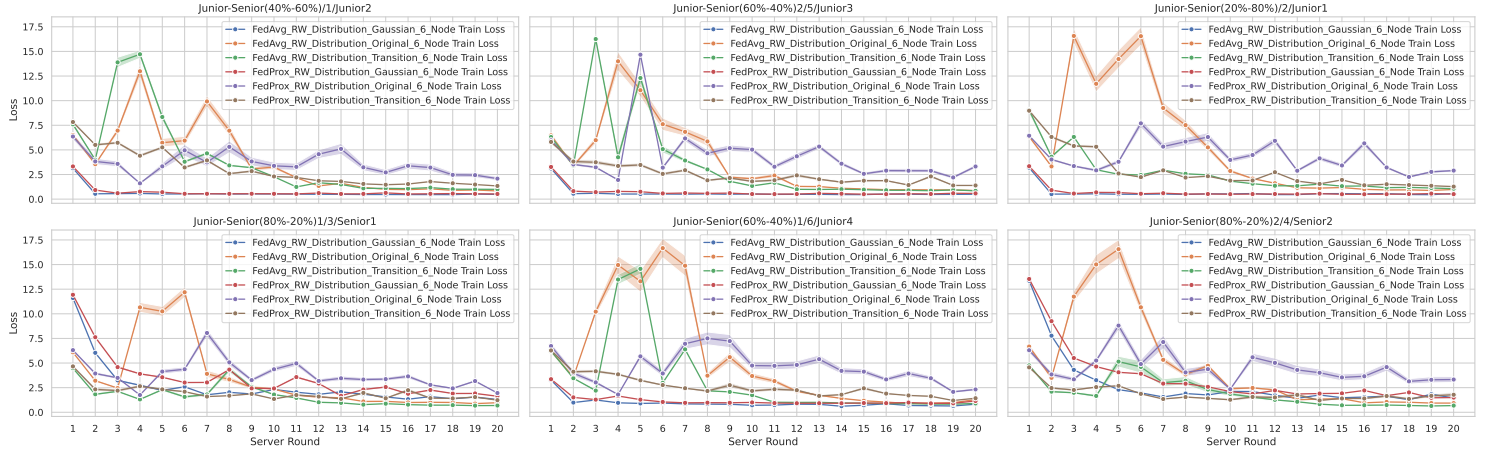
In a broader context, Federated Learning has shown significant promise as a tool for training sophisticated models using data previously inaccessible due to regulatory compliance. However, FL is still nascent and ripe for refinement. This research has discussed numerous challenges associated with FL, including heterogeneity, computational and communication overhead, security, and fairness. The primary focus of this thesis has been addressing data heterogeneity by exploring methods to standardize data distributions across nodes or account for client drift, examining various distributions and analyzing how performance changes based on these distributions. Despite these challenges, various accessible frameworks for FL and its numerous applications have been highlighted. Future researchers are encouraged to delve deeper into solving these issues while discovering innovative ways to apply FL to other domains or enhance existing solutions. This thesis provides a comprehensive understanding of FL, demonstrating its potential for various use cases and advocating for continued research to address current challenges and bolster the case for

utilizing Federated Learning.

Appendix A

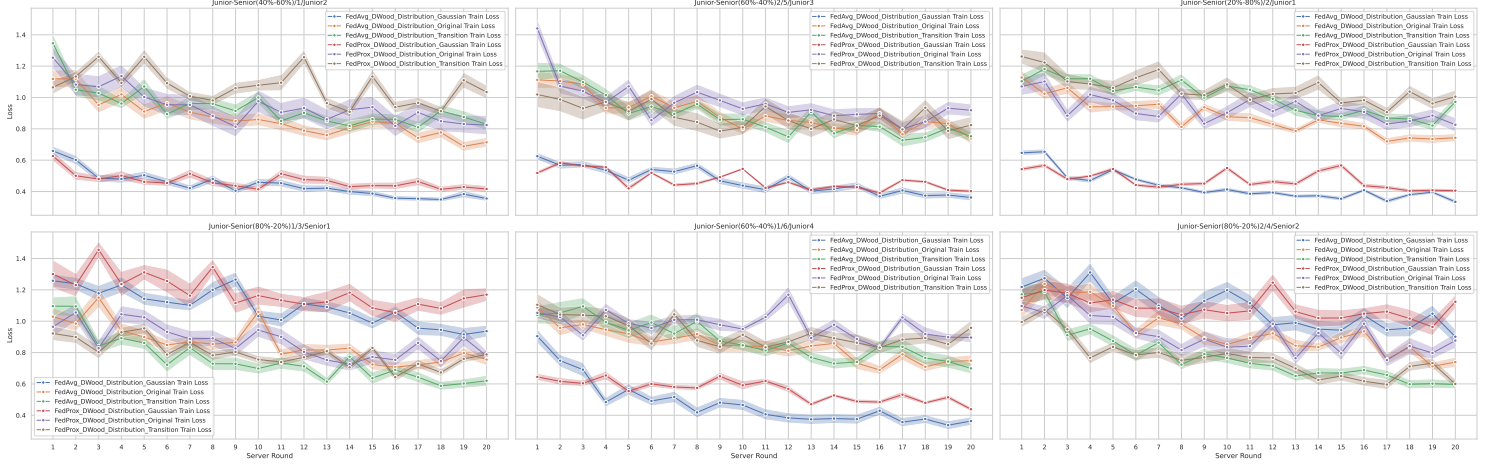
Additional Plots

Losses by Client (Distribution, Random Weights, 6 nodes)



(a) Starting from random weights.

Losses by Client (Distribution, DWood, 6 nodes)



(b) Starting from a David Wood pre-trained model.

Figure A.1: Local model losses over server rounds of Federated Learning (across 6 nodes). Calculated using the average training losses in every node.

Bibliography

- [1] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. “Deep learning”. In: *Nature* 521.7553 (May 2015), pp. 436–444. ISSN: 1476-4687. DOI: 10.1038/nature14539. URL: <https://doi.org/10.1038/nature14539>.
- [2] Pramila P. Shinde and Seema Shah. “A Review of Machine Learning and Deep Learning Applications”. In: *2018 Fourth International Conference on Computing Communication Control and Automation (ICCUBEA)*. 2018, pp. 1–6. DOI: 10.1109/ICCUBEA.2018.8697857.
- [3] Eunsuk Chong, Chulwoo Han, and Frank C. Park. “Deep learning networks for stock market analysis and prediction: Methodology, data representations, and case studies”. In: *Expert Systems with Applications* 83 (2017), pp. 187–205. ISSN: 0957-4174. DOI: <https://doi.org/10.1016/j.eswa.2017.04.030>. URL: <https://www.sciencedirect.com/science/article/pii/S0957417417302750>.
- [4] M. G. Schultz et al. “Can deep learning beat numerical weather prediction?” In: *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences* 379.2194 (Feb. 2021), p. 20200097. ISSN: 1471-2962. DOI: 10.1098/rsta.2020.0097. URL: <http://dx.doi.org/10.1098/rsta.2020.0097>.
- [5] He Li, Kaoru Ota, and Mianxiong Dong. “Learning IoT in Edge: Deep Learning for the Internet of Things with Edge Computing”. In: *IEEE Network* 32.1 (2018), pp. 96–101. DOI: 10.1109/MNET.2018.1700202.
- [6] Tobias Gentner et al. “Data Lakes in Healthcare: Applications and Benefits from the Perspective of Data Sources and Players”. In: *Procedia Computer Science* 225 (2023). 27th International Conference on Knowledge Based and Intelligent Information and Engineering Systems (KES 2023), pp. 1302–1311. ISSN: 1877-0509. DOI: <https://doi.org/10.1016/j.procs.2023.10.118>. URL: <https://www.sciencedirect.com/science/article/pii/S1877050923012760>.
- [7] Paul Voigt and Axel Von dem Bussche. “The eu general data protection regulation (gdpr)”. In: *A Practical Guide, 1st Ed., Cham: Springer International Publishing* 10.3152676 (2017), pp. 10–5555.
- [8] Eric Goldman. “An introduction to the california consumer privacy act (ccpa)”. In: *Santa Clara Univ. Legal Studies Research Paper* (2020).
- [9] Katja Franke and Christian Gaser. “Ten Years of BrainAGE as a Neuroimaging Biomarker of Brain Aging: What Insights Have We Gained?” In: *Frontiers in Neurology* 10 (Aug. 2019). DOI: 10.3389/fneur.2019.00789.
- [10] Seyyed Saeid Cheshmi et al. “Brain Age Estimation Using Structural MRI: A Clustered Federated Learning Approach”. In: *2023 IEEE International Conference on Omni-layer Intelligent Systems (COINS)*. IEEE. 2023, pp. 1–6.

- [11] H. Brendan McMahan et al. *Communication-Efficient Learning of Deep Networks from Decentralized Data*. 2016. arXiv: 1602.05629 [cs.LG].
- [12] Jakub Konečný et al. *Federated Optimization: Distributed Machine Learning for On-Device Intelligence*. 2016. arXiv: 1610.02527 [cs.LG].
- [13] Reza Shokri and Vitaly Shmatikov. “Privacy-preserving deep learning”. In: Sept. 2015, pp. 909–910. ISBN: 9781509018246. DOI: 10.1109/ALLERTON.2015.7447103.
- [14] Jakub Konečný et al. *Federated Learning: Strategies for Improving Communication Efficiency*. 2017. arXiv: 1610.05492 [cs.LG].
- [15] Ashish Rauniar et al. *Federated Learning for Medical Applications: A Taxonomy, Current Trends, Challenges, and Future Research Directions*. 2023. arXiv: 2208.03392 [cs.LG].
- [16] Qiang Yang, Yang Liu, and Yong Cheng. *Federated Learning*. Morgan & Claypool Publishers, 2019. ISBN: 1681736977.
- [17] WeBank AI Department. *Federated Learning White Paper V1.0*. Technical Report. Available online: <https://aisp-1251170195.cos.ap-hongkong.myqcloud.com/fedweb/1552917186945.pdf>. WeBank, Shenzhen, China, Sept. 2018.
- [18] Qiang Yang et al. *Federated Machine Learning: Concept and Applications*. 2019. arXiv: 1902.04885 [cs.AI].
- [19] Priyanka Mary Mammen. *Federated Learning: Opportunities and Challenges*. 2021. arXiv: 2101.05428 [cs.LG].
- [20] Enrique Tomás Martínez Beltrán et al. “Decentralized Federated Learning: Fundamentals, State of the Art, Frameworks, Trends, and Challenges”. In: *IEEE Communications Surveys & Tutorials* 25.4 (2023), pp. 2983–3013. DOI: 10.1109/COMST.2023.3315746.
- [21] Li Li et al. “A review of applications in federated learning”. In: *Computers & Industrial Engineering* 149 (2020), p. 106854. ISSN: 0360-8352. DOI: <https://doi.org/10.1016/j.cie.2020.106854>. URL: <https://www.sciencedirect.com/science/article/pii/S0360835220305532>.
- [22] Andrew Hard et al. *Federated Learning for Mobile Keyboard Prediction*. 2019.
- [23] Yong Song et al. “Federated Learning Application on Telecommunication-Joint Healthcare Recommendation”. In: *2021 IEEE 21st International Conference on Communication Technology (ICCT)*. 2021, pp. 1443–1448. DOI: 10.1109/ICCT52962.2021.9657870.
- [24] Adrià Gascón et al. “Privacy-Preserving Distributed Linear Regression on High-Dimensional Data”. In: *Proceedings on Privacy Enhancing Technologies* 2017 (2017), pp. 345–364. URL: <https://api.semanticscholar.org/CorpusID:717272>.
- [25] Yang Liu et al. *Vertical Federated Learning: Concepts, Advances and Challenges*. 2023. arXiv: 2211.12814 [cs.LG].
- [26] Sinno Jialin Pan and Qiang Yang. “A Survey on Transfer Learning”. In: *IEEE Transactions on Knowledge and Data Engineering* 22.10 (2010), pp. 1345–1359. DOI: 10.1109/TKDE.2009.191.
- [27] Yang Liu et al. “A Secure Federated Transfer Learning Framework”. In: *IEEE Intelligent Systems* 35.4 (2020), pp. 70–82. DOI: 10.1109/MIS.2020.2988525.
- [28] Tuo Zhang et al. “Federated Learning for the Internet of Things: Applications, Challenges, and Opportunities”. In: *IEEE Internet of Things Magazine* 5.1 (2022), pp. 24–29. DOI: 10.1109/IOTM.004.2100182.
- [29] Peter Kairouz et al. *Advances and Open Problems in Federated Learning*. 2021. arXiv: 1912.04977 [cs.LG].
- [30] Tian Li et al. “Federated Learning: Challenges, Methods, and Future Directions”. In: *IEEE Signal Processing Magazine* 37.3 (May 2020), pp. 50–60. ISSN: 1558-0792. DOI: 10.1109/msp.2020.2975749. URL: <http://dx.doi.org/10.1109/MSP.2020.2975749>.

- [31] Brendan McMahan and Daniel Ramage. *Federated Learning: Collaborative Machine Learning without Centralized Training Data* — *blog.research.google*. <https://blog.research.google/2017/04/federated-learning-collaborative.html>. [Accessed 24-01-2024]. 2017.
- [32] Yujing Chen et al. “Asynchronous Online Federated Learning for Edge Devices with Non-IID Data”. In: *2020 IEEE International Conference on Big Data (Big Data)*. 2020, pp. 15–24. DOI: 10.1109/BigData50022.2020.9378161.
- [33] Andrew C. Yao. “Protocols for secure computations”. In: *23rd Annual Symposium on Foundations of Computer Science (sfcs 1982)*. 1982, pp. 160–164. DOI: 10.1109/SFCS.1982.38.
- [34] Chenhao Xu et al. “Asynchronous federated learning on heterogeneous devices: A survey”. In: *Computer Science Review* 50 (2023), p. 100595.
- [35] Holger R. Roth et al. “Federated Learning for Breast Density Classification: A Real-World Implementation”. In: *Lecture Notes in Computer Science*. Springer International Publishing, 2020, pp. 181–191. ISBN: 9783030605483. DOI: 10.1007/978-3-030-60548-3_18. URL: http://dx.doi.org/10.1007/978-3-030-60548-3_18.
- [36] Michal Yemini et al. “Semi-Decentralized Federated Learning with Collaborative Relaying”. In: *2022 IEEE International Symposium on Information Theory (ISIT)*. 2022, pp. 1471–1476. DOI: 10.1109/ISIT50566.2022.9834707.
- [37] Frank Po Chen Lin et al. “Semi-Decentralized Federated Learning with Cooperative D2D Local Model Aggregations”. English (US). In: *IEEE Journal on Selected Areas in Communications* 39.12 (Dec. 2021). Publisher Copyright: © 1983-2012 IEEE., pp. 3851–3869. ISSN: 0733-8716. DOI: 10.1109/JSAC.2021.3118344.
- [38] Lingjuan Lyu et al. *Towards Fair and Privacy-Preserving Federated Deep Models*. 2020. arXiv: 1906.01167 [cs.CR].
- [39] Youyang Qu et al. “Blockchain-enabled Federated Learning: A Survey”. In: *ACM Comput. Surv.* 55.4 (Nov. 2022). ISSN: 0360-0300. DOI: 10.1145/3524104. URL: <https://doi.org/10.1145/3524104>.
- [40] Christopher Natoli et al. “Deconstructing Blockchains: A Comprehensive Survey on Consensus, Membership and Structure”. In: (Aug. 2019).
- [41] Satoshi Nakamoto. “Bitcoin: A Peer-to-Peer Electronic Cash System”. In: *Cryptography Mailing list at https://metzdowd.com* (Mar. 2009).
- [42] Hamed Taherdoost. “Smart Contracts in Blockchain Technology: A Critical Review”. In: *Information* 14.2 (2023). ISSN: 2078-2489. URL: <https://www.mdpi.com/2078-2489/14/2/117>.
- [43] Tien Tuan Anh Dinh et al. *Untangling Blockchain: A Data Processing View of Blockchain Systems*. 2017. arXiv: 1708.05665 [cs.DB].
- [44] Guy Zyskind, Oz Nathan, and Alex ‘Sandy’ Pentland. “Decentralizing Privacy: Using Blockchain to Protect Personal Data”. In: *2015 IEEE Security and Privacy Workshops* (2015), pp. 180–184. URL: <https://api.semanticscholar.org/CorpusID:10184767>.
- [45] Zhilin Wang and Qin Hu. *Blockchain-based Federated Learning: A Comprehensive Survey*. 2021. arXiv: 2110.02182 [cs.CR].
- [46] Arthur Gervais et al. “On the security and performance of proof of work blockchains”. In: *Proceedings of the 2016 ACM SIGSAC conference on computer and communications security*. 2016, pp. 3–16.
- [47] Harsh Bimal Desai, Mustafa Safa Ozdayi, and Murat Kantarcioglu. *BlockFLA: Accountable Federated Learning via Hybrid Blockchain Architecture*. 2020. arXiv: 2010.07427 [cs.CR].
- [48] Xidi Qu et al. *Proof of Federated Learning: A Novel Energy-recycling Consensus Algorithm*. 2019. arXiv: 1912.11745 [cs.CR].

- [49] Joost Verbraeken et al. “A Survey on Distributed Machine Learning”. In: *ACM Comput. Surv.* 53.2 (Mar. 2020). ISSN: 0360-0300. DOI: 10.1145/3377454. URL: <https://doi.org/10.1145/3377454>.
- [50] Ji Liu et al. “From distributed machine learning to federated learning: A survey”. In: *Knowledge and Information Systems* 64.4 (2022), pp. 885–917.
- [51] Tommy van der Vorst et al. *Fast forward : How the speed of the internet will develop between now and 2020*. English. Publication 2013.048-1262. - Report commissioned by NLkabel & Cable Europe. Netherlands: Dialogic innovatie & interactie, 2014.
- [52] Yujun Lin et al. *Deep Gradient Compression: Reducing the Communication Bandwidth for Distributed Training*. 2020. arXiv: 1712.01887 [cs.CV].
- [53] Sai Praneeth Karimireddy et al. *Error Feedback Fixes SignSGD and other Gradient Compression Schemes*. 2019. arXiv: 1901.09847 [cs.LG].
- [54] Thijs Vogels, Sai Praneeth Karimireddy, and Martin Jaggi. “PowerSGD: Practical Low-Rank Gradient Compression for Distributed Optimization”. In: *Advances in Neural Information Processing Systems*. Ed. by H. Wallach et al. Vol. 32. Curran Associates, Inc., 2019. URL: https://proceedings.neurips.cc/paper_files/paper/2019/file/d9fbed9da256e344c1fa46bb46c34c5f-Paper.pdf.
- [55] Hanlin Tang et al. *DeepSqueeze: Decentralization Meets Error-Compensated Compression*. 2019. arXiv: 1907.07346 [cs.DC].
- [56] Sebastian U. Stich and Sai Praneeth Karimireddy. *The Error-Feedback Framework: Better Rates for SGD with Delayed Gradients and Compressed Communication*. 2021. arXiv: 1909.05350 [cs.LG].
- [57] Qinbin Li, Bingsheng He, and Dawn Song. *Practical One-Shot Federated Learning for Cross-Silo Setting*. 2021. arXiv: 2010.01017 [cs.LG].
- [58] Nguyen Truong et al. “Privacy preservation in federated learning: An insightful survey from the GDPR perspective”. In: *Computers & Security* 110 (2021), p. 102402. ISSN: 0167-4048. DOI: <https://doi.org/10.1016/j.cose.2021.102402>. URL: <https://www.sciencedirect.com/science/article/pii/S0167404821002261>.
- [59] Alissa Brauneck et al. “Federated Machine Learning, Privacy-Enhancing Technologies, and Data Protection Laws in Medical Research: Scoping Review”. In: *J Med Internet Res* 25 (Mar. 2023), e41588. ISSN: 1438-8871. DOI: 10.2196/41588. URL: <http://www.ncbi.nlm.nih.gov/pubmed/36995759>.
- [60] Bharat Bhushan, G. Sahoo, and Amit Kumar Rai. “Man-in-the-middle attack in wireless and computer networking — A review”. In: *2017 3rd International Conference on Advances in Computing, Communication & Automation (ICACCA) (Fall)*. 2017, pp. 1–6. DOI: 10.1109/ICACCAF.2017.8344724.
- [61] Ligeng Zhu, Zhijian Liu, and Song Han. “Deep Leakage from Gradients”. In: *Advances in Neural Information Processing Systems*. Ed. by H. Wallach et al. Vol. 32. Curran Associates, Inc., 2019. URL: https://proceedings.neurips.cc/paper_files/paper/2019/file/60a6c4002cc7b29142def8871531281a-Paper.pdf.
- [62] Yehuda Lindell. “Secure multiparty computation”. In: *Commun. ACM* 64.1 (Dec. 2020), pp. 86–96. ISSN: 0001-0782. DOI: 10.1145/3387108. URL: <https://doi.org/10.1145/3387108>.
- [63] F. Lau et al. “Distributed denial of service attacks”. In: *Smc 2000 conference proceedings. 2000 ieee international conference on systems, man and cybernetics. 'cybernetics evolving to systems, humans, organizations, and their complex interactions' (cat. no.0)*. Vol. 3. 2000, 2275–2280 vol.3. DOI: 10.1109/ICSMC.2000.886455.

- [64] Lucas Davi et al. “Privilege escalation attacks on android”. In: *Information Security: 13th International Conference, ISC 2010, Boca Raton, FL, USA, October 25-28, 2010, Revised Selected Papers 13*. Springer. 2011, pp. 346–360.
- [65] Guo Jianluan and Wang Xiaoyan. “Computer Vision Operating System of Bank Economic Management Security under 5G Wireless Communication Technology”. In: *Wireless Communications and Mobile Computing 2022* (Mar. 2022), pp. 1–7. DOI: 10.1155/2022/6233870.
- [66] *What is a DDoS Attack: Types, Prevention & Remediation — OneLogin — onelogin.com*. <https://www.onelogin.com/learn/ddos-attack>. [Accessed 06-06-2024].
- [67] Lingjuan Lyu, Han Yu, and Qiang Yang. *Threats to Federated Learning: A Survey*. 2020. arXiv: 2003.02133 [cs.CR].
- [68] Arjun Nitin Bhagoji et al. *Analyzing Federated Learning through an Adversarial Lens*. 2019. arXiv: 1811.12470 [cs.LG].
- [69] Eugene Bagdasaryan et al. *How To Backdoor Federated Learning*. 2019. arXiv: 1807.00459 [cs.CR].
- [70] Jinhyun So, Başak Güler, and A. Salman Avestimehr. “Byzantine-Resilient Secure Federated Learning”. In: *IEEE Journal on Selected Areas in Communications* 39.7 (2021), pp. 2168–2181. DOI: 10.1109/JSAC.2020.3041404.
- [71] Clement Fung, Chris J. M. Yoon, and Ivan Beschastnikh. *Mitigating Sybils in Federated Learning Poisoning*. 2020. arXiv: 1808.04866 [cs.LG].
- [72] Thijs Veugen et al. “Secure comparison protocols in the semi-honest model”. In: *IEEE Journal of Selected Topics in Signal Processing* 9.7 (2015), pp. 1217–1228.
- [73] Battista Biggio, Blaine Nelson, and Pavel Laskov. “Support Vector Machines Under Adversarial Label Noise.” In: *Journal of Machine Learning Research - Proceedings Track* 20 (Jan. 2011), pp. 97–112.
- [74] Marco Barreno et al. “Can machine learning be secure?” In: vol. 2006. Mar. 2006, pp. 16–25. DOI: 10.1145/1128817.1128824.
- [75] Ling Huang et al. “Adversarial machine learning”. In: *Proceedings of the 4th ACM Workshop on Security and Artificial Intelligence*. AISec ’11. Chicago, Illinois, USA: Association for Computing Machinery, 2011, pp. 43–58. ISBN: 9781450310031. DOI: 10.1145/2046684.2046692. URL: <https://doi.org/10.1145/2046684.2046692>.
- [76] Xinyun Chen et al. *Targeted Backdoor Attacks on Deep Learning Systems Using Data Poisoning*. 2017. arXiv: 1712.05526 [cs.CR].
- [77] Yingqi Liu et al. “Trojaning Attack on Neural Networks”. In: *Network and Distributed System Security Symposium*. 2018. URL: <https://api.semanticscholar.org/CorpusID:31806516>.
- [78] Tianyu Gu, Brendan Dolan-Gavitt, and Siddharth Garg. *BadNets: Identifying Vulnerabilities in the Machine Learning Model Supply Chain*. 2019. arXiv: 1708.06733 [cs.CR].
- [79] Ali Shafahi et al. *Poison Frogs! Targeted Clean-Label Poisoning Attacks on Neural Networks*. 2018. arXiv: 1804.00792 [cs.LG].
- [80] Le Trieu Phong et al. “Privacy-Preserving Deep Learning via Additively Homomorphic Encryption”. In: *IEEE Transactions on Information Forensics and Security* 13.5 (2018), pp. 1333–1345. DOI: 10.1109/TIFS.2017.2787987.
- [81] Lili Su and Jiaming Xu. *Securing Distributed Gradient Descent in High Dimensional Statistical Learning*. 2019. arXiv: 1804.10140 [cs.DC].
- [82] Luca Melis et al. *Exploiting Unintended Feature Leakage in Collaborative Learning*. 2018. arXiv: 1805.04049 [cs.CR].

- [83] Briland Hitaj, Giuseppe Ateniese, and Fernando Perez-Cruz. “Deep Models Under the GAN: Information Leakage from Collaborative Deep Learning”. In: Oct. 2017, pp. 603–618. DOI: 10.1145/3133956.3134012.
- [84] Milad Nasr, Reza Shokri, and Amir Houmansadr. “Comprehensive Privacy Analysis of Deep Learning: Passive and Active White-box Inference Attacks against Centralized and Federated Learning”. In: *2019 IEEE Symposium on Security and Privacy (SP)*. 2019, pp. 739–753. DOI: 10.1109/SP.2019.00065.
- [85] David Evans, Vladimir Kolesnikov, Mike Rosulek, et al. “A pragmatic introduction to secure multi-party computation”. In: *Foundations and Trends® in Privacy and Security* 2.2-3 (2018), pp. 70–246.
- [86] J. Ayo Akinyele. *Intro to Secure Multi-Party Computation — medium.com*. <https://medium.com/boltlabs/intro-to-secure-multi-party-computation-15875dc5a24a>. [Accessed 25-01-2024]. 2020.
- [87] Claudio Orlandi. “Is multiparty computation any good in practice?” In: *2011 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. 2011, pp. 5848–5851. DOI: 10.1109/ICASSP.2011.5947691.
- [88] Peter Bogetoft et al. “Secure Multiparty Computation Goes Live”. In: vol. 5628. Feb. 2009, pp. 325–343. ISBN: 978-3-642-03548-7. DOI: 10.1007/978-3-642-03549-4_20.
- [89] Dahlia Malkhi et al. “Fairplay—A Secure Two-Party Computation System”. In: *13th USENIX Security Symposium (USENIX Security 04)*. San Diego, CA: USENIX Association, Aug. 2004. URL: <https://www.usenix.org/conference/13th-usenix-security-symposium/fairplay%7B%5Ctextendash%7D-secure-two-party-computation-system>.
- [90] Dan Bogdanov, Sven Laur, and Jan Willemson. “Sharemind: A Framework for Fast Privacy-Preserving Computations”. In: vol. 2008. Oct. 2008, p. 289. ISBN: 978-3-540-88312-8. DOI: 10.1007/978-3-540-88313-5_13.
- [91] Haris Smajlović et al. “Sequare: a high-performance framework for secure multiparty computation enables biomedical data sharing”. In: *Genome Biology* 24.1 (Jan. 2023), p. 5. ISSN: 1474-760X. DOI: 10.1186/s13059-022-02841-5. URL: <https://doi.org/10.1186/s13059-022-02841-5>.
- [92] Nipuna Senanayake et al. “NeuroCrypt: Machine Learning Over Encrypted Distributed Neuroimaging Data”. In: *Neuroinformatics* 20.1 (May 2021), pp. 91–108. ISSN: 1559-0089. DOI: 10.1007/s12021-021-09525-8. URL: <http://dx.doi.org/10.1007/s12021-021-09525-8>.
- [93] Cynthia Dwork. “Differential privacy”. In: *International colloquium on automata, languages, and programming*. Springer. 2006, pp. 1–12.
- [94] Cynthia Dwork. “Differential privacy: A survey of results”. In: *International conference on theory and applications of models of computation*. Springer. 2008, pp. 1–19.
- [95] Alberto Blanco-Justicia et al. “A critical review on the use (and misuse) of differential privacy in machine learning”. In: *ACM Computing Surveys* 55.8 (2022), pp. 1–16.
- [96] Elise Devaux. *What is Differential Privacy: definition, mechanisms, and examples - Statice — statice.ai*. <https://www.statice.ai/post/what-is-differential-privacy-definition-mechanisms-examples>. [Accessed 05-04-2024]. 2022.
- [97] Ronald L Rivest, Len Adleman, Michael L Dertouzos, et al. “On data banks and privacy homomorphisms”. In: *Foundations of secure computation* 4.11 (1978), pp. 169–180.
- [98] Shafi Goldwasser, Silvio Micali, and Po Tong. “Why and how to establish a private code on a public network”. In: *23rd Annual Symposium on Foundations of Computer Science (sfcs 1982)*. IEEE. 1982, pp. 134–144.

- [99] Pascal Paillier. “Public-Key Cryptosystems Based on Composite Degree Residuosity Classes”. In: vol. 5. May 1999, pp. 223–238. ISBN: 978-3-540-65889-4. DOI: 10.1007/3-540-48910-X_16.
- [100] Dan Boneh, Eu-Jin Goh, and Kobbi Nissim. “Evaluating 2-DNF Formulas on Ciphertexts”. In: vol. 3378. Feb. 2005, pp. 325–341. ISBN: 978-3-540-24573-5. DOI: 10.1007/978-3-540-30576-7_18.
- [101] Craig Gentry. “Fully Homomorphic Encryption Using Ideal Lattices”. In: vol. 9. May 2009, pp. 169–178. DOI: 10.1145/1536414.1536440.
- [102] Michael Naehrig, Kristin Lauter, and Vinod Vaikuntanathan. “Can homomorphic encryption be practical?” In: *Proceedings of the 3rd ACM workshop on Cloud computing security workshop*. 2011, pp. 113–124.
- [103] Haokun Fang and Quan Qian. “Privacy Preserving Machine Learning with Homomorphic Encryption and Federated Learning”. In: *Future Internet* 13.4 (2021). ISSN: 1999-5903. URL: <https://www.mdpi.com/1999-5903/13/4/94>.
- [104] Chengliang Zhang et al. “BatchCrypt: Efficient Homomorphic Encryption for Cross-Silo Federated Learning”. In: *2020 USENIX Annual Technical Conference (USENIX ATC 20)*. USENIX Association, July 2020, pp. 493–506. ISBN: 978-1-939133-14-4. URL: <https://www.usenix.org/conference/atc20/presentation/zhang-chengliang>.
- [105] Mohamed Sabt, Mohammed Achemlal, and Abdelmadjid Bouabdallah. “Trusted Execution Environment: What It is, and What It is Not”. In: *2015 IEEE Trustcom/BigDataSE/ISPA*. Vol. 1. 2015, pp. 57–64. DOI: 10.1109/Trustcom.2015.357.
- [106] Jan-Erik Ekberg, Kari Kostiaainen, and N. Asokan. “Trusted execution environments on mobile devices”. In: *Proceedings of the 2013 ACM SIGSAC Conference on Computer & Communications Security*. CCS ’13. Berlin, Germany: Association for Computing Machinery, 2013, pp. 1497–1498. ISBN: 9781450324779. DOI: 10.1145/2508859.2516758. URL: <https://doi.org/10.1145/2508859.2516758>.
- [107] Victor Costan and Srinivas Devadas. *Intel SGX Explained*. Cryptology ePrint Archive, Paper 2016/086. <https://eprint.iacr.org/2016/086>. 2016. URL: <https://eprint.iacr.org/2016/086>.
- [108] *AMD SEV-TIO Whitepaper*. Tech. rep. Accessed: yyyy-mm-dd. Advanced Micro Devices, Inc., 2021. URL: <https://www.amd.com/content/dam/amd/en/documents/epyc-business-docs/white-papers/sev-tio-whitepaper.pdf>.
- [109] Hang Su and Haoyu Chen. *Experiments on Parallel Training of Deep Neural Network using Model Averaging*. 2018. arXiv: 1507.01239 [cs.LG].
- [110] Chen Yu et al. *Distributed Learning over Unreliable Networks*. 2019. arXiv: 1810.07766 [cs.DC].
- [111] Hao Yu, Sen Yang, and Shenghuo Zhu. *Parallel Restarted SGD with Faster Convergence and Less Communication: Demystifying Why Model Averaging Works for Deep Learning*. 2018. arXiv: 1807.06629 [math.OA].
- [112] Kele Xu et al. *Collaborative Deep Learning Across Multiple Data Centers*. 2018. arXiv: 1810.06877 [cs.LG].
- [113] Jianmin Chen et al. *Revisiting Distributed Synchronous SGD*. 2017. arXiv: 1604.00981 [cs.LG].
- [114] Sai Praneeth Karimireddy et al. “SCAFFOLD: Stochastic Controlled Averaging for Federated Learning”. In: *Proceedings of the 37th International Conference on Machine Learning*. Ed. by Hal Daumé III and Aarti Singh. Vol. 119. Proceedings of Machine Learning Research. PMLR, 13–18 Jul 2020, pp. 5132–5143. URL: <https://proceedings.mlr.press/v119/karimireddy20a.html>.

- [115] Sai Praneeth Karimireddy et al. “Breaking the centralized barrier for cross-device federated learning”. In: *Advances in Neural Information Processing Systems*. Ed. by M. Ranzato et al. Vol. 34. Curran Associates, Inc., 2021, pp. 28663–28676. URL: https://proceedings.neurips.cc/paper_files/paper/2021/file/f0e6be4ce76ccfa73c5a540d992d0756-Paper.pdf.
- [116] Tian Li et al. *Federated Optimization in Heterogeneous Networks*. 2020. arXiv: 1812.06127 [cs.LG].
- [117] Diederik P. Kingma and Jimmy Ba. *Adam: A Method for Stochastic Optimization*. 2017. arXiv: 1412.6980 [cs.LG].
- [118] John Duchi, Elad Hazan, and Yoram Singer. “Adaptive Subgradient Methods for Online Learning and Stochastic Optimization”. In: *Journal of Machine Learning Research* 12.61 (2011), pp. 2121–2159. URL: <http://jmlr.org/papers/v12/duchi11a.html>.
- [119] Manzil Zaheer et al. “Adaptive Methods for Nonconvex Optimization”. In: *Advances in Neural Information Processing Systems*. Ed. by S. Bengio et al. Vol. 31. Curran Associates, Inc., 2018. URL: https://proceedings.neurips.cc/paper_files/paper/2018/file/90365351ccc7437a1309dc64e4db32a3-Paper.pdf.
- [120] Dami Choi et al. “On empirical comparisons of optimizers for deep learning”. In: *arXiv preprint arXiv:1910.05446* (2019).
- [121] Sashank Reddi et al. *Adaptive Federated Optimization*. 2021. arXiv: 2003.00295 [cs.LG].
- [122] Xingchao Peng et al. *Federated Adversarial Domain Adaptation*. 2019. arXiv: 1911.02054 [cs.CV].
- [123] Daniel Peterson, Pallika Kanani, and Virendra J. Marathe. *Private Federated Learning with Domain Adaptation*. 2019. arXiv: 1912.06733 [cs.LG].
- [124] Xiaoxiao Li et al. *Multi-site fMRI Analysis Using Privacy-preserving Federated Learning and Domain Adaptation: ABIDE Results*. 2020. arXiv: 2001.05647 [cs.LG].
- [125] Yan Kang et al. “Privacy-preserving Federated Adversarial Domain Adaptation over Feature Groups for Interpretability”. In: *IEEE Transactions on Big Data* (2022), pp. 1–12. DOI: 10.1109/TBDATA.2022.3188292.
- [126] Yue Zhao et al. “Federated Learning with Non-IID Data”. In: (2018). DOI: 10.48550/ARXIV.1806.00582. URL: <https://arxiv.org/abs/1806.00582>.
- [127] Neta Shoham et al. *Overcoming Forgetting in Federated Learning on Non-IID Data*. 2019. arXiv: 1910.07796 [cs.LG].
- [128] James Kirkpatrick et al. “Overcoming catastrophic forgetting in neural networks”. In: *Proceedings of the National Academy of Sciences* 114.13 (Mar. 2017), pp. 3521–3526. ISSN: 1091-6490. DOI: 10.1073/pnas.1611835114. URL: <http://dx.doi.org/10.1073/pnas.1611835114>.
- [129] Virginia Smith et al. *Federated Multi-Task Learning*. 2018. arXiv: 1705.10467 [cs.LG].
- [130] Xibin Dong et al. “A survey on ensemble learning”. In: *Frontiers of Computer Science* 14 (2020), pp. 241–258.
- [131] Alhassan Mabrouk et al. “Ensemble Federated Learning: An approach for collaborative pneumonia diagnosis”. In: *Applied Soft Computing* 144 (2023), p. 110500. ISSN: 1568-4946. DOI: <https://doi.org/10.1016/j.asoc.2023.110500>. URL: <https://www.sciencedirect.com/science/article/pii/S1568494623005185>.
- [132] Meisam Hejazinia et al. *FEL: High Capacity Learning for Recommendation and Ranking via Federated Ensemble Learning*. 2022. arXiv: 2206.03852 [cs.IR].
- [133] Alireza Fallah, Aryan Mokhtari, and Asuman Ozdaglar. *Personalized Federated Learning: A Meta-Learning Approach*. 2020. arXiv: 2002.07948 [cs.LG].
- [134] Kangkang Wang et al. *Federated Evaluation of On-device Personalization*. 2019. arXiv: 1910.10252 [cs.LG].

- [135] Tian Li et al. *Ditto: Fair and Robust Federated Learning Through Personalization*. 2021. arXiv: 2012.04221 [cs.LG].
- [136] Ninareh Mehrabi et al. “A survey on bias and fairness in machine learning”. In: *ACM computing surveys (CSUR)* 54.6 (2021), pp. 1–35.
- [137] Sean Vucinic and Qiang Zhu. “The Current State and Challenges of Fairness in Federated Learning”. In: *IEEE Access* 11 (2023), pp. 80903–80914. DOI: 10.1109/ACCESS.2023.3295412.
- [138] Yufeng Zhan et al. “A Survey of Incentive Mechanism Design for Federated Learning”. In: *IEEE Transactions on Emerging Topics in Computing* 10.2 (2022), pp. 1035–1044. DOI: 10.1109/TETC.2021.3063517.
- [139] Rongfei Zeng et al. *A Comprehensive Survey of Incentive Mechanism for Federated Learning*. 2021. arXiv: 2106.15406 [cs.LG].
- [140] Xuezhen Tu et al. “Incentive Mechanisms for Federated Learning: From Economic and Game Theoretic Perspective”. In: *IEEE Transactions on Cognitive Communications and Networking* 8.3 (2022), pp. 1566–1593. DOI: 10.1109/TCCN.2022.3177522.
- [141] Mehryar Mohri, Gary Sivek, and Ananda Theertha Suresh. “Agnostic Federated Learning”. In: *Proceedings of the 36th International Conference on Machine Learning*. Ed. by Kamalika Chaudhuri and Ruslan Salakhutdinov. Vol. 97. Proceedings of Machine Learning Research. PMLR, Sept. 2019, pp. 4615–4625. URL: <https://proceedings.mlr.press/v97/mohri19a.html>.
- [142] Cynthia Dwork et al. “Fairness through awareness”. In: *Proceedings of the 3rd Innovations in Theoretical Computer Science Conference (ITCS '12)*. ACM, 2012, pp. 214–226.
- [143] Alex Braungardt. “FATE, Flower, PySyft & Co. — Federated Learning Frameworks in Python”. In: *Medium* (Jan. 2023). Published in ELCAL IT. URL: <https://medium.com/elca-it/flower-pysyft-co-federated-learning-frameworks-in-python-b1a8eda68b0d>.
- [144] Alexander Ziller et al. “Pysyft: A library for easy federated learning”. In: *Federated Learning Systems: Towards Next-Generation AI* (2021), pp. 111–139.
- [145] Theo Ryffel et al. *A generic framework for privacy preserving deep learning*. 2018. arXiv: 1811.04017 [cs.LG].
- [146] OpenMined. *PySyft*. <https://github.com/OpenMined/PySyft>. Perform data science on data that remains in someone else’s server. 2024.
- [147] Ivan Damgard et al. *Practical Covertly Secure MPC for Dishonest Majority – or: Breaking the SPDZ Limits*. Cryptology ePrint Archive, Paper 2012/642. <https://eprint.iacr.org/2012/642>. 2012. DOI: http://dx.doi.org/10.1007/978-3-642-40203-6_1. URL: <https://eprint.iacr.org/2012/642>.
- [148] Sameer Wagh, Divya Gupta, and Nishanth Chandran. “SecureNN: 3-Party Secure Computation for Neural Network Training”. In: *Proceedings on Privacy Enhancing Technologies* 2019 (July 2019), pp. 26–49. DOI: 10.2478/popets-2019-0035.
- [149] Alexander J. Titus et al. *PySEAL: A Python wrapper implementation of the SEAL homomorphic encryption library*. 2018. arXiv: 1803.01891 [q-bio.QM].
- [150] Daniel J Beutel et al. “Flower: A Friendly Federated Learning Research Framework”. In: *arXiv preprint arXiv:2007.14390* (2020).
- [151] Maximilian Kapsecker et al. “Federated Learning with Swift: An Extension of Flower and Performance Evaluation”. In: *SoftwareX* 24 (2023), p. 101533.
- [152] Juncen Zhu et al. “Blockchain-empowered Federated Learning: Challenges, Solutions, and Future Directions”. In: *ACM Comput. Surv.* 55.11 (Feb. 2023). ISSN: 0360-0300. DOI: 10.1145/3570953. URL: <https://doi.org/10.1145/3570953>.

- [153] Philipp Moritz et al. *Ray: A Distributed Framework for Emerging AI Applications*. 2018. arXiv: 1712.05889 [cs.DC].
- [154] James Bell et al. *Secure Single-Server Aggregation with (Poly)Logarithmic Overhead*. Cryptology ePrint Archive, Paper 2020/704. <https://eprint.iacr.org/2020/704>. 2020. DOI: 10.1145/3372297.3417885. URL: <https://eprint.iacr.org/2020/704>.
- [155] Keith Bonawitz et al. *Practical Secure Aggregation for Federated Learning on User-Held Data*. 2016. arXiv: 1611.04482 [cs.CR].
- [156] Patrick Foley et al. “OpenFL: the open federated learning library”. In: *Physics in Medicine & Biology* (2022). DOI: 10.1088/1361-6560/ac97d9. URL: <http://iopscience.iop.org/article/10.1088/1361-6560/ac97d9>.
- [157] Sarthak Pati et al. *The Federated Tumor Segmentation (FeTS) Challenge*. 2021. arXiv: 2105.05874 [eess.IV].
- [158] Google. *TensorFlow Federated: Machine Learning on Decentralized Data*. <https://www.tensorflow.org/federated>. Accessed: 9/02/2024. 2019.
- [159] Holger R. Roth et al. “NVIDIA FLARE: Federated Learning from Simulation to Real-World”. In: *IEEE Data Eng. Bull.*, Vol. 46, No. 1 (Mar. 2023). DOI: <https://doi.org/10.48550/arXiv.2210.13291>.
- [160] NVIDIA. *NVIDIA Clara: AI-powered Solutions for Healthcare*. <https://www.nvidia.com/en-us/clara/>. Accessed: 9/02/2024. 2018.
- [161] 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. URL: <http://jmlr.org/papers/v22/20-815.html>.
- [162] Sebastian Caldas et al. *LEAF: A Benchmark for Federated Settings*. 2019. arXiv: 1812.01097 [cs.LG].
- [163] Iman Beheshti et al. “Predicting Brain Age Using Machine Learning Algorithms: A Comprehensive Evaluation”. In: *IEEE Journal of Biomedical and Health Informatics* 26.4 (2022), pp. 1432–1440. DOI: 10.1109/JBHI.2021.3083187.
- [164] M. Tanveer et al. “Deep learning for brain age estimation: A systematic review”. In: *Information Fusion* 96 (Aug. 2023), pp. 130–143. ISSN: 1566-2535. DOI: 10.1016/j.inffus.2023.03.007. URL: <http://dx.doi.org/10.1016/j.inffus.2023.03.007>.
- [165] David A Wood et al. “Accurate brain-age models for routine clinical MRI examinations”. English. In: *NeuroImage* 249 (Apr. 2022). Funding Information: This work was supported by the Royal College of Radiologists , King’s College Hospital Research and Innovation , King’s Health Partners Challenge Fund , NVIDIA (through the unrestricted use of a GPU obtained in a competition), and the Wellcome/Engineering and Physical Sciences Research Council Centre for Medical Engineering (WT 203148/Z/16/Z). Publisher Copyright: © 2022 The Author(s), p. 118871. ISSN: 1053-8119. DOI: 10.1016/j.neuroimage.2022.118871.
- [166] Han Peng et al. “Accurate brain age prediction with lightweight deep neural networks”. In: *Medical Image Analysis* 68 (2021), p. 101871. ISSN: 1361-8415. DOI: <https://doi.org/10.1016/j.media.2020.101871>. URL: <https://www.sciencedirect.com/science/article/pii/S1361841520302358>.
- [167] Esten H. Leonardsen et al. “Deep neural networks learn general and clinically relevant representations of the ageing brain”. In: *NeuroImage* 256 (2022), p. 119210. ISSN: 1053-8119. DOI: <https://doi.org/10.1016/j.neuroimage.2022.119210>. URL: <https://www.sciencedirect.com/science/article/pii/S1053811922003342>.
- [168] Dimitris Stripelis et al. *Secure & Private Federated Neuroimaging*. 2023. arXiv: 2205.05249 [cs.LG].

- [169] André Vital Serafim Silva. “Deep Learning for Brain Age Prediction from T1-Weighted Magnetic Resonance Images”. Master of Science in Applied Computer Science. Master’s Thesis. Boulevard de la Plaine 2, 1050 Ixelles: Vrije Universiteit Brussel (VUB), June 2023.
- [170] Gao Huang et al. *Densely Connected Convolutional Networks*. 2018. arXiv: 1608.06993 [cs.CV].
- [171] Trang T. Le et al. “A Nonlinear Simulation Framework Supports Adjusting for Age When Analyzing BrainAGE”. In: *Frontiers in Aging Neuroscience* 10 (2018). ISSN: 1663-4365. DOI: 10.3389/fnagi.2018.00317. URL: <https://www.frontiersin.org/articles/10.3389/fnagi.2018.00317>.
- [172] N. S. Brencic et al. “Intuitive and Intelligent Solutions for Elderly Care”. In: *Digital Health in Focus of Predictive, Preventive and Personalised Medicine*. Springer International Publishing, 2020, pp. 101–108.
- [173] Dongchul Cha, MinDong Sung, and Yu-Rang Park. “Implementing Vertical Federated Learning Using Autoencoders: Practical Application, Generalizability, and Utility Study”. In: *JMIR Medical Informatics* 9 (June 2021), e26598. DOI: 10.2196/26598.
- [174] Yiqiang Chen et al. *FedHealth: A Federated Transfer Learning Framework for Wearable Healthcare*. 2021. arXiv: 1907.09173 [cs.LG].
- [175] Burcin Ikiz. *A Pandemic AI Engine Without Borders* — hai.stanford.edu. <https://hai.stanford.edu/news/pandemic-ai-engine-without-borders>. [Accessed 23-01-2024]. 2020.
- [176] I. Dayan et al. “Federated learning for predicting clinical outcomes in patients with COVID-19”. In: *Nature Medicine* 27.10 (2021), pp. 1735–1743.
- [177] A. 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.
- [178] Q. Dou et al. “Federated deep learning for detecting COVID-19 lung abnormalities in CT: a privacy-preserving multinational validation study”. In: *NPJ Digital Medicine* 4.1 (2021), pp. 1–11.
- [179] Ashish Rauniyar et al. “Federated Learning for Medical Applications: A Taxonomy, Current Trends, Challenges, and Future Research Directions”. In: *IEEE Internet of Things Journal* (2023), pp. 1–1. DOI: 10.1109/JIOT.2023.3329061.
- [180] Nicola Rieke et al. “The future of digital health with federated learning”. In: *npj Digital Medicine* 3.1 (Sept. 2020), p. 119. ISSN: 2398-6352. DOI: 10.1038/s41746-020-00323-1. URL: <https://doi.org/10.1038/s41746-020-00323-1>.
- [181] Kritika Prakash et al. *Private Deep Learning of Medical Data for Hospitals using Federated Learning and Differential Privacy* — blog.openmined.org. <https://blog.openmined.org/untitled-3/>. [Accessed 25-01-2024]. 2021.
- [182] Krzysztof J Gorgolewski et al. “The brain imaging data structure, a format for organizing and describing outputs of neuroimaging experiments”. In: *Scientific data* 3 (June 2016), p. 160044. ISSN: 2052-4463. DOI: 10.1038/sdata.2016.44. URL: <https://europepmc.org/articles/PMC4978148>.
- [183] LK Soumya Kumari and R Sundarajan. “A review on brain age prediction models”. In: *Brain Research* (2023), p. 148668.