POLYTECHNIQUE MONTRÉAL

affiliée à l'Université de Montréal

Privacy-Preserving Federated Learning Architecture for Secure Patient Data Sharing in Hospital Networks

MAHNAZ GHORBANIZAD

Département de génie informatique et génie logiciel

Mémoire présenté en vue de l'obtention du diplôme de *Maîtrise ès sciences appliquées*

Génie informatique et génie logiciel

Avril 2025

POLYTECHNIQUE MONTRÉAL

affiliée à l'Université de Montréal

Ce mémoire intitulé :

Privacy-Preserving Federated Learning Architecture for Secure Patient Data Sharing in Hospital Networks

présenté par **Mahnaz GHORBANIZAD**

en vue de l'obtention du diplôme de *Maîtrise ès sciences appliquées*

a été dûment accepté par le jury d'examen constitué de :

DEDICATION

This dissertation is dedicated to my mother, Shokat Asghari, whose boundless love and unwavering support have been the foundation of my success

To my father, Reza Ghorbanizad, whose wisdom and quiet strength continue to inspire me.

To my husband, Masoud Abedini, my greatest support and steadfast companion, whose patience and encouragement have uplifted me every step of the way.

I am forever grateful to you all for your love, strength and belief in me.

ACKNOWLEDGEMENTS

First, I would like to begin by expressing my deepest gratitude to Professor Samuel Pierre, distinguished professor at Polytechnique Montreal and director of LARIM, for providing me with the exceptional opportunity to conduct my master’s research in his esteemed laboratory. His outstanding guidance, unwavering support, and profound trust in my potential have been indispensable throughout this remarkable academic journey. It has been a privilege to learn from his expertise and to be part of such an inspiring research environment. I extend my sincere appreciation to the jury members for their time, thoughtful feedback, dedication in evaluating my work and participating in my dissertation presentation.

I am especially indebted to Dr. Franjieh El Khoury, LARIM's research associate, coordinator, and supervisor of the HYPERSEC project. Her exceptional mentorship, steadfast encouragement, and profound expertise have significantly shaped my research and personal growth. Her meticulous attention to detail, critical insights, and constructive feedback were pivotal in refining my dissertation. It has been an honor to work under the guidance of such a dedicated and accomplished researcher.

My heartfelt thanks also go to the Flex team and the LARIM team for their collaboration, assistance, and invaluable support. Their shared commitment to excellence made this experience all the more rewarding.

To my family, I owe my deepest appreciation for their unwavering love, resilience, and encouragement. My father, Reza Ghorbanizad, has been a steady source of strength and inspiration, and my mother, Shokat Asghari, whose unconditional love and endless sacrifices have been the cornerstone of my success, has been my greatest motivation. Finally, I am profoundly grateful to my husband and closest confidant, Masoud Abedini, for his boundless patience, steadfast support, and belief in my potential. His constant encouragement and companionship have been a source of strength through every challenge and triumph. To all those who have played a role in this journey, whether directly or indirectly, I extend my heartfelt thanks. Your contributions have left an enduring mark on this achievement.

RÉSUMÉ

La croissance rapide des systèmes de santé numériques a transformé la manière dont les données des patients sont collectées, traitées et analysées pour améliorer les résultats médicaux. Cependant, cette évolution a également soulevé des préoccupations majeures concernant la confidentialité et la sécurité des données, notamment dans les réseaux hospitaliers où les informations sensibles des patients doivent être protégées contre tout accès non autorisé. La recherche collaborative entre les hôpitaux est souvent freinée par les réglementations sur la confidentialité, empêchant ainsi le partage de données précieuses. L'apprentissage fédéré « Federated Learning » (FL) s'impose comme une solution innovante permettant aux institutions de former conjointement des modèles d'apprentissage automatique sans transférer de données brutes, répondant ainsi à de nombreux défis liés à la confidentialité. Toutefois, malgré son potentiel, l'apprentissage fédéré présente certaines limites, telles que les vulnérabilités aux fuites de données, le risque de compromission de l’entraînement local du modèle et une diminution des performances en raison de l’hétérogénéité des données entre les différents nœuds.

Cette dissertation propose une architecture d'apprentissage fédéré préservant la confidentialité pour améliorer le partage sécurisé des données des patients entre les hôpitaux, permettant ainsi une collaboration efficace sans enfreindre les réglementations sur la confidentialité. L'architecture proposée intègre le mécanisme de confidentialité différentielle « Differential Privacy » (DP) afin d'introduire un bruit contrôlé dans les mises à jour de l'entraînement du modèle local avant leur partage pour l'agrégation globale, protégeant ainsi les informations sensibles même en présence de menaces adversariales. De plus, l'algorithme d'agrégation « Federated Averaging» (FedAvg) est utilisé pour combiner les mises à jour des modèles provenant de diverses institutions tout en empêchant la divulgation des contributions individuelles, préservant ainsi la confidentialité des ensembles de données de chaque hôpital.

La recherche utilise la base de données « Oxford Parkinson's Disease Detection Dataset » pour simuler un scénario réaliste impliquant plusieurs nœuds hospitaliers. Chaque nœud traite ses données partitionnées localement, reflétant une configuration décentralisée similaire aux environnements hospitaliers réels. L'étude compare également l'apprentissage fédéré avec l'application de la confidentialité différentielle et l'apprentissage fédéré sans confidentialité différentielle dans une analyse comparative visant à mettre en évidence les avantages et les compromis de l'approche. Des métriques de performance clés, telles que la précision du modèle, l'efficacité computationnelle et la préservation de la confidentialité, sont utilisées pour évaluer l'efficacité du système proposé.

Un élément clé de l'évaluation consiste à examiner l'impact des différents multiplicateurs de bruit (σ) sur les budgets de confidentialité correspondants (ε) dans le mécanisme de confidentialité différentielle. Cette analyse quantifie les compromis entre la préservation de la confidentialité et les performances du modèle, offrant un aperçu de l'influence de l'injection de bruit sur la précision et l'efficacité du modèle fédéré. Les résultats indiquent que l'architecture proposée préserve efficacement la confidentialité des données tout en maintenant une précision compétitive et une efficacité d'entraînement stable par rapport à l'apprentissage fédéré sans mécanisme de confidentialité différentielle.

Les résultats expérimentaux démontrent que l'architecture proposée atteint un équilibre solide entre confidentialité et utilité. Le modèle intégrant la confidentialité différentielle avec un multiplicateur de bruit de σ = 1,0 a obtenu une précision de 92,50 %, avec un budget de confidentialité de ε = 4,80, offrant ainsi une protection robuste de la confidentialité tout en maintenant des performances compétitives.

Une analyse comparative entre l'apprentissage fédéré avec confidentialité différentielle et l'apprentissage fédéré sans confidentialité différentielle révèle que, bien que la confidentialité différentielle améliore la confidentialité des données, elle introduit des compromis en termes de précision et de temps d'entraînement. Le modèle, sans confidentialité différentielle, a atteint une précision de 98,70 % avec un temps d'entraînement total de 7,82 secondes ; cependant, il ne protège pas la confidentialité, le rendant vulnérable aux attaques de reconstruction et d'inférence des données.

En revanche, l'apprentissage fédéré avec (σ = 1,0) n'a nécessité que 5,76 secondes pour compléter l'entraînement tout en garantissant une forte protection de la confidentialité, empêchant tout accès non autorisé aux données sensibles des patients. Bien que l'apprentissage fédéré sans atteigne une précision plus élevée, la confidentialité reste une exigence essentielle dans les applications de santé, où la protection des informations des patients est primordiale.

Contrairement à certaines études précédentes ayant rapporté une surcharge computationnelle importante due à l’intégration de la confidentialité différentielle, l’efficacité de l’entraînement dans cette architecture est restée stable, confirmant ainsi sa faisabilité pour des collaborations hospitalières en conditions réelles.

ABSTRACT

The rapid growth of digital healthcare systems has transformed the way patient data is collected, processed, and analyzed to improve medical outcomes. However, this evolution has also introduced significant concerns regarding data privacy and security, particularly in hospital networks where sensitive patient information must be safeguarded against unauthorized access. Collaborative research between hospitals is often hindered by privacy regulations, preventing the sharing of valuable datasets. Federated Learning (FL) has emerged as an innovative solution that enables institutions to collaboratively train machine learning models without transferring raw data, thus addressing many privacy-related challenges. However, despite its potential, FL presents several limitations, such as vulnerabilities to data leakage, the risk of compromised local model training and decreased performance due to data heterogeneity across different nodes.

This dissertation proposes a Privacy-Preserving Federated Learning (PPFL) architecture to enhance secure patient data sharing among hospitals, enabling effective collaboration without breaching privacy regulations. The proposed architecture incorporates Differential Privacy (DP) mechanism to introduce controlled noise into local model training updates before they are shared for global aggregation, protecting sensitive information even in the presence of adversarial threats. Additionally, the Federated Averaging (FedAvg) aggregation algorithm is employed to combine model updates from various institutions while preventing the disclosure of individual contributions, thereby preserving the privacy of each hospital's dataset.

The research employs the Oxford Parkinson's Disease Detection Dataset to simulate a realistic healthcare scenario involving multiple hospital nodes. Each node processes its partitioned data locally, reflecting a decentralized setup that mirrors real-world hospital environments. Moreover, the research benchmarks Federated Learning (FL) with applying Differential Privacy (DP) and Federated Learning (FL) without Differential Privacy (DP) for comparative analysis to highlight the benefits and trade-offs of the approach. Key performance metrics, such as model accuracy, computational efficiency, and privacy preservation, are used to evaluate the effectiveness of the proposed architecture.

A critical component of the evaluation examines the impact of different noise multipliers (σ) on the corresponding privacy budgets (ε) within the Differential Privacy (DP) mechanism. This analysis quantifies the trade-offs between privacy preservation and model performance, offering insights into how noise injection influences the accuracy and efficiency of the federated model. The results indicate that the proposed architecture effectively preserves data confidentiality while maintaining competitive accuracy and stable training efficiency compared to Federated Learning (FL) without Differential Privacy (DP) mechanism.

Experimental results demonstrate that the proposed architecture achieves a strong balance between privacy and utility. The model, incorporating Differential Privacy (DP) with a noise multiplier of $\sigma = 1.0$ attained an accuracy of 92.50%, with a privacy budget of $\varepsilon = 4.80$, providing robust privacy protection while maintaining competitive performance. A comparative analysis between Federated Learning (FL) with DP and Federated Learning (FL) without DP reveals that while DP enhances data confidentiality, it introduces trade-offs in accuracy and training time. The model, without Differential Privacy (DP), attained an accuracy of 98.70% with a total training time of 7.82 seconds; however, it lacks privacy protection, making it vulnerable to data reconstruction and inference attacks. In contrast, the FL with DP ($\sigma = 1.0$) required only 5.76 seconds to complete training while providing strong privacy guarantees, preventing unauthorized access to sensitive patient data. Although FL without DP achieves higher accuracy, confidentiality is a critical requirement in healthcare applications, where protecting patient information is paramount. Unlike some prior research works that reported significant computational overhead due to DP integration, the training efficiency in this architecture remained stable, confirming its feasibility in real-world hospital collaborations.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF SYMBOLS AND ABBREVIATIONS

AI	Artificial Intelligence
DP	Differential Privacy
DFA	Detrended Fluctuation Analysis
FL	Federated Learning
GDPR	General Data Protection Regulation
HNR	Harmonic-to-Noise Ratio
HIPAA	Health Insurance Portability and Accountability
IoMT	Internet of Medical Things
ε (Epsilon)	Privacy Budget (in Differential Privacy)
IID	Independent and Identically Distributed
Non-IID	Non-Independent and Identically Distributed
FF	Fundamental Frequency
RPDE	Recurrence Period Density Entropy
SMPC	Secure Multi-Party Computation
HE	Homomorphic Encryption
DP-SGD	Differentially Private Stochastic Gradient Descent
FedAvg	Federated Averaging Algorithm
DP-FedAvg	Differentially Private Federated Averaging

SA	Secure Aggregation
LDP	Local Differential Privacy
FHE	Fully Homomorphic Encryption
PHE	Partially Homomorphic Encryption
SHE	Somewhat Homomorphic Encryption
σ	Noise Multiplier
AUC	Area Under the Curve
CNN	Convolutional Neural Network
RNN	Recurrent Neural Network
LSTM	Long Short-Term Memory
FNN	Feedforward Neural Network
EHR	Electronic Health Record
PPFL	Privacy-Preserving Federated Learning
ECG	Electrocardiogram
GRU	Gated Recurrent Unit
FLWR	Flower framework
1DCNN	One-Dimensional Convolutional Neural Network
ReLU	Rectified Linear Unit
Adam	Adaptive Moment Estimation

LIST OF APPENDICES

CHAPTER 1 INTRODUCTION

The rapid growth of data-driven technologies has revolutionized various fields, enabling organizations to leverage machine learning for improved decision-making and automation. From healthcare to finance, large-scale data analysis has led to groundbreaking advancements, enhancing efficiency and accuracy in critical applications. However, these innovations come with significant challenges, particularly concerning data privacy, security, and ethical considerations. As organizations collect and process vast amounts of sensitive information, protecting user privacy has become a pressing issue.

Traditional centralized machine learning approaches require data to be aggregated in a single location for model training, raising concerns about unauthorized access, data breaches, and regulatory compliance. In sectors like healthcare, where patient confidentiality is crucial, sharing data across institutions is often restricted due to legal and ethical constraints. These concerns have led to an increasing demand for privacy-preserving mechanisms that allow collaborative learning while maintaining data security.

Federated Learning (FL) addresses this challenge by enabling institutions to collaboratively train machine learning models without sharing raw data. Each institution performs local model training on its private dataset and sends only model updates to a central server for aggregation, ensuring that sensitive data remains local. While FL helps protect data privacy, it remains vulnerable to inference attacks and adversarial threats.

To mitigate these risks, privacy-preserving mechanisms such as Differential Privacy (DP) are used. DP applies controlled noise to model updates, ensuring that individual data points cannot be inferred while maintaining overall model utility. However, the addition of noise can impact model accuracy and training efficiency, posing challenges in balancing privacy and performance.

In this chapter, we will first present the basic definitions and concepts of Federated Learning (FL) and privacy-preserving mechanisms in healthcare. This will be followed by the problem statement, the research objectives, and an overview of the dissertation structure.

1.1 Basic Definitions and Concepts

In this section, we review the basic definitions and main concepts related to Federated Learning (FL) in healthcare and the privacy-preserving mechanisms employed in this research, specifically

Differential Privacy (DP). We also describe the healthcare dataset selected for this research and the proposed security solutions integrated into our architecture. This is followed by the problem statement, the research objectives, and an outline of the dissertation plan.

1.1.1 Federated Learning in Healthcare

Federated Learning (FL) is a decentralized approach to machine learning that allows multiple institutions, such as hospitals or research centers, to collaboratively train a shared model without exchanging raw data [1]. Unlike traditional centralized systems, which require all data to be transferred to a central repository for model training, *FL* keeps sensitive information remains within its source environment [2]. This is achieved by enabling each institution to train the model locally on its proprietary dataset and sending only the computed model updates to a central server for aggregation [3]. The central server does not have direct access to the raw data or individual updates. Instead, it combines the local updates received from all participating nodes to refine the global model, which is then redistributed to the nodes for further training. This iterative process continues until the global model reaches the desired performance level [4]. By keeping the data localized, *FL* preserves the privacy of sensitive information while facilitating collaborative learning across institutions [5].

In healthcare, the adoption of *FL* addresses one of the most pressing challenges: ensuring the privacy and security of patient data while leveraging the collective insights of distributed datasets [6]. Hospitals and healthcare providers generate vast amounts of data, including medical records, imaging data, diagnostic results, and biosensor outputs However, these data are often siloed within institutions due to privacy concerns and organizational boundaries, limiting the potential for collaborative research and the development of robust machine learning models [7]. *FL* enables institutions to collaborate on building predictive and diagnostic models without violating privacy laws or risking exposure of sensitive information [1][8].

The *FL* operates through a process of communication rounds. In each round, the global model is sent to participating nodes, where it is trained on local data. The locally updated models are then transmitted back to the central server, which aggregates these updates to create an improved global model. This process is repeated iteratively until the model converges [2] [4]. By leveraging the local data at each institution, *FL* allows the global model to learn from diverse datasets without centralizing sensitive information [3].

FL is particularly effective in handling data heterogeneity, which is common in healthcare. For example, hospitals serving different demographics may have datasets with distinct distributions of patient characteristics and disease patterns [6]. By aggregating updates from such diverse datasets, *FL* produces a global model that is more generalizable and robust compared to models trained on data from a single institution. However, this diversity also presents challenges, as the non-independent and identically distributed (non-IID) nature of the data can slow down model convergence and affect performance [7].

Despite its advantages, *FL* face several challenges. Communication between nodes and the central server requires frequent transmission of model updates, which can impose significant network and computational overhead [8]. Additionally, while *FL* keeps data local, the updates shared with the central server can still reveal sensitive information about the underlying data if not properly secured [5] [6]. To address these concerns, privacy-preserving mechanisms such as Differential Privacy (DP) are often integrated into FL. Differential Privacy (DP) adds noise to model updates, reducing the risk of exposing sensitive information while maintaining data confidentiality. These mechanisms enhance the security of FL, making it a practical and reliable solution for privacy-sensitive domains like healthcare [4] [7].

1.1.2 Privacy-Preserving Mechanisms

Privacy-Preserving Mechanisms are critical components of *Federated Learning (FL)*, particularly in privacy-sensitive domains such as *healthcare*. These mechanisms aim to protect sensitive data from being inferred or reconstructed during training and communication. Despite *Federated Learning's decentralized nature*, where raw data remains at local institutions, there is still a risk of *privacy leakage* through model updates, especially during aggregation at the central server.

- Key privacy-preserving mechanisms in FL

Differential Privacy (DP) is a *mechanisms* designed to ensure that the inclusion or exclusion of any single data point in a dataset does not significantly impact the output of an algorithm. In

Federated Learning (FL), DP is implemented by adding noise to the model updates before they are sent to the central server. This noise obfuscates the contribution of individual data points, making it challenging for adversaries to infer sensitive information [9] [10]. The strength of *privacy in DP* is controlled by a parameter known as the privacy budget (ε). The privacy budget (ε) in *DP* quantifies the trade-off between *privacy protection* and model utility. Smaller ε values offer *stronger privacy* but degrade model performance due to added noise, while larger ε values improve accuracy at the cost of *weaker privacy* guarantees. In *healthcare* applications, where patient confidentiality is paramount, *DP* ensures that adversaries cannot reconstruct patient-specific details even if model updates are intercepted [11]. The challenge lies in selecting an appropriate ε value, as excessive noise can hinder clinical predictions while insufficient noise risks privacy exposure. Striking a balance is crucial to maintaining both data security and model reliability in FL-based *healthcare* systems [12] [13].

In addition to *DP*, other *privacy-enhancing techniques* such as Secure Multi-Party Computation (SMPC) and Homomorphic Encryption (HE) provide cryptographic security by encrypting model updates during training. While these methods offer *strong privacy* guarantees, they often introduce significant computational overhead, making *DP* a more practical choice for large-scale FL-based *healthcare* applications.

1.1.3 Applications of Federated Learning in Healthcare

The Federated Learning (FL) has gained significant attention in the *healthcare* sector for its potential to enable collaborative research and model development without compromising patient privacy. The unique challenges and opportunities in *healthcare* make it an ideal domain for *FL* implementation. Below are some key applications of *FL in healthcare*:

- *Disease Prediction and Diagnosis: FL* facilitates the development of predictive models for various diseases by combining insights from distributed *healthcare datasets.* For instance, hospitals can collaboratively train models to predict diseases like cancer, diabetes, or Parkinson's disease while keeping sensitive patient data localized. This approach enhances model robustness by leveraging diverse datasets across institutions [14].

- *Personalized Medicine:* By enabling institutions to train models on their specific patient populations, *FL* supports the development of personalized treatment strategies. For example, models trained using *FL* can recommend tailored drug regimens or treatment plans based on a patient's medical history, genetic data, and local healthcare trends [15].

- *Medical Imaging and Diagnostics: FL* is widely applied in medical imaging tasks such as tumor detection, segmentation, and anomaly identification in MRI, CT scans, and X-rays. Collaborative training across multiple hospitals ensures models can generalize well to diverse imaging data, improving diagnostic accuracy [16].

- *Drug Discovery and Genomics:* Pharmaceutical companies and research labs use *FL* to collaborate on drug discovery and genomics research without exposing proprietary or sensitive data. This accelerates the identification of drug candidates and insights into genetic markers associated with diseases [17].

- *Wearable Devices and Remote Monitoring: FL* enables the development of machine learning models that analyze data from wearable health devices. By processing data locally on devices, *FL* preserves user privacy while enabling continuous monitoring of health metrics such as heart rate, activity levels, and sleep patterns [18].

- *Pandemic Response:* During global health crises, such as the COVID-19 pandemic, *FL* has been employed to develop predictive models for outbreak trends, vaccine distribution strategies, and treatment efficacy without centralizing sensitive epidemiological data [19].

These applications demonstrate the transformative potential of *FL* to drive innovation and improve patient outcomes while adhering to stringent privacy and regulatory requirements.

1.1.4 Healthcare Dataset Overview

In this research, we use the *Oxford Parkinson's Disease Detection Dataset*, a widely used biomedical dataset that contains 4,680 structured health-related features for *Parkinson's disease detection*. *The dataset* enables effective simulation of *Federated Learning (FL)* scenarios, where data remains distributed across multiple institutions. Each node in *the federated setup* processes its partitioned subset locally, reflecting a decentralized training approach similar to real-world *hospital networks.*

Feature normalization is applied to maintain consistency, and missing values are handled to improve robustness. *This dataset* is suitable for *Privacy-Preserving Federated Learning (PPFL)*, as it allows model training across decentralized institutions without exposing raw data. The structured nature of the dataset facilitates collaborative research while aligning with *privacy-preserving principles.*

1.1.5 Concepts of Proposed Security Solutions

The proposed architecture leverages *Differential Privacy (DP)* to address critical privacy challenges in *Federated Learning (FL) for healthcare. Differential Privacy (DP)* is carefully integrated to provide robust protection against data leakage, safeguarding sensitive healthcare data during local model training and while transmitting updates to the central server. *DP* enhances privacy by introducing controlled noise to model updates, ensuring that individual contributions remain indistinguishable, even in the presence of adversaries intercepting communication channels [20] [21]. In the proposed architecture, *DP* protects the integrity of local updates and minimizes the risk of sensitive information being inferred from shared model updates. This approach effectively reduces the risk of data leakage across both the training and aggregation phases, enabling collaborative model development across healthcare institutions without compromising patient privacy [20] [22].

However, integrating *Differential Privacy (DP) into Federated Learning (FL)* requires careful parameter tuning. Factors such as noise levels (σ) and communication frequency must be optimized to strike a balance between privacy preservation and computational efficiency [21] [22]. This challenge is particularly critical in *healthcare* settings, where achieving high model accuracy is essential for patient care and medical research. By addressing these trade-offs, this research seeks to advance the practicality and scalability of *Federated Learning (FL) frameworks in healthcare,* enabling effective and secure collaboration in medical research [20-22].

1.2 Problem Statement

Upon reviewing the current advancements in Federated Learning (FL) and privacy-preserving technologies in healthcare, several significant challenges and gaps have been identified. These limitations, which hinder the successful implementation of secure and efficient digital healthcare systems, include the following critical issues:

- *Lack of Security in Healthcare Systems:* The protection of sensitive patient data is a critical requirement in healthcare systems. Despite advancements in security, many systems remain vulnerable to data breaches, unauthorized access, and malicious attacks. With increasing reliance on digital technologies, addressing these risks has become essential [23] [24].

- *Challenges in Ensuring Privacy:* Privacy preservation is a significant concern, as healthcare systems often handle highly sensitive information. Many current approaches fail to adequately protect user privacy, particularly during data storage, transmission, and analysis [25].
- *Efficiency Trade-Offs with Security:* Implementing robust security and privacy mechanisms often comes with performance overhead. Balancing efficiency with the need for secure and privacy-preserving mechanisms remains a critical challenge [26].
- *Lack of Real-World Implementation Evidence:* While many emerging technologies propose solutions to these issues, their real-world efficiency and effectiveness in providing secure, privacy-preserving healthcare solutions have not been thoroughly investigated [27] [28].
- *Growing Security Risks with Technological Advancements:* As technologies such as IoMT and AI continue to evolve, the complexity of potential security threats also increases. Ensuring that technological progress does not compromise security is essential to the future of healthcare systems [24] [29].

Considering these challenges, it is essential to address the growing concerns surrounding security and privacy in healthcare data sharing. This leads us to our primary research question: How can we enhance the privacy preservation and data security in hospital data sharing to ensure effective and confidential collaborative healthcare research? To further explore this topic, we focus on two specific aspects:

1. What mechanism can be used to analyze the privacy risks in healthcare data sharing?
2. Which model can be adopted to enhance the privacy preservation and secure the exchange of data in the context of hospital data sharing?

1.3 Research Objectives

The main objective of this dissertation is to propose an architecture to enhance the privacy preservation and data security in hospital data sharing while enabling effective collaborative healthcare research.

To achieve this purpose, this research focuses on:

1. Design a mechanism to analyze the privacy risks in healthcare data sharing;
2. Propose a model to enhance the privacy preservation and secure the exchange of data in the context of hospital data sharing;
3. Implement the proposed mechanism and model;
4. Evaluate the performance of the proposed mechanism and model in terms of accuracy and security.

1.4 Dissertation Plan

The remaining of this dissertation is organized as follows. Chapter 2 provides the background and literature review, discussing Federated Learning (FL), its applications in healthcare, privacy-preserving mechanisms, and existing challenges. Chapter 3 introduces the proposed Privacy-Preserving Federated Learning (PPFL) architecture, detailing its design, the integration of the Differential Privacy (DP) mechanism, and the underlying rationale. Chapter 4 presents the implementation and results, covering dataset preparation, partitioning strategies, Federated Learning (FL) configurations, and experimental evaluations. Chapter 5 concludes the dissertation by summarizing key contributions, discussing limitations, and suggesting directions for future research in Privacy-Preserving Federated Learning (PPFL) for healthcare.

CHAPTER 2 LITERATURE REVIEW

This chapter presents a comprehensive literature review on Federated Learning (FL) in healthcare, focusing on its applications, challenges, and privacy-preserving techniques. The review examines key privacy-preserving mechanisms, including Secure Multi-Party Computation (SMPC), Homomorphic Encryption (HE), and Differential Privacy (DP), with a particular emphasis on DP as the primary privacy mechanism in our research work. Additionally, the chapter explores the privacy-utility trade-offs in FL, analyzing how different privacy settings impact model performance and convergence. By evaluating existing research, identifying limitations, and addressing open challenges such as data heterogeneity, computational overhead, and privacy budget selection, this review lays the groundwork for developing a more privacy-efficient FL framework in healthcare applications.

2.1 Federated Learning in Healthcare

Federated Learning (FL) has gained significant attention in healthcare research as a privacy-preserving alternative to centralized machine learning models, enabling collaborative model training without direct data sharing. This section reviews the related works in two subdomains: applications of FL in healthcare and challenges associated with its implementation. At the end of this section, we summarize key findings and Outline research gaps that this dissertation aims to address.

The role of technology in modern healthcare is expanding rapidly, with the integration of Artificial Intelligence (AI) and Machine Learning (ML) enhancing diagnostics, treatment planning, and patient monitoring. One of the most critical advancements in this space is Federated Learning (FL), which enables hospitals and research institutions to collaborate on training AI models without sharing raw patient data [30]. This decentralized approach allows for multi-institutional model training while ensuring that patient privacy and regulatory compliance are maintained [31].

In recent years, FL has been increasingly applied in various healthcare domains, such as medical imaging, disease prediction, clinical decision support systems, and remote patient monitoring [32]. These applications demonstrate the potential of FL to improve healthcare services while overcoming data-sharing restrictions. However, alongside these advancements, new challenges have emerged, including concerns related to data heterogeneity, security vulnerabilities, model convergence, and communication efficiency.

Despite these challenges, Federated Learning (FL) presents a transformative opportunity for medical research and healthcare AI. By enabling collaborative model training without violating data privacy regulations, FL fosters a new era of secure, AI-driven healthcare solutions [30] [31].

2.1.1 Federated Learning Applications in Healthcare

The application of Federated Learning (FL) in healthcare has gained substantial attention in recent years, driven by the need for data privacy and multi-institutional collaboration. Unlike traditional machine learning models that require centralized data aggregation, FL enables local model training on sensitive medical data without data transfer between institutions. This makes FL particularly attractive for healthcare settings, where patient data privacy is paramount due to stringent regulations such as HIPAA (Health Insurance Portability and Accountability Act) and GDPR (General Data Protection Regulation) [33].

Over the past few years, various research works have demonstrated the practical implementation of FL in healthcare. Its applications span from disease prediction and medical imaging to remote patient monitoring and personalized medicine. These diverse use cases illustrate the versatility and potential of FL in improving healthcare outcomes while complying with data protection laws. This section reviews key research works in each of these areas, focusing on their methodologies, outcomes, and challenges.

2.1.2 Disease Prediction and Diagnosis

Li *et al.* [34] developed a Federated Learning (FL) framework for cardiovascular risk prediction using multi-site MRI data. They integrated imaging data from five institutions while maintaining privacy in data collaboration. The framework aimed to harmonize data across multiple sites through a domain adaptation strategy, enabling the model to generalize effectively on unseen data. Convolutional Neural Networks (CNNs) were used for feature extraction and prediction, achieving a 12% accuracy improvement compared to traditional models.

Data heterogeneity across sites created challenges due to variations in imaging protocols and MRI machine settings. The authors implemented a standardized pre-processing pipeline to normalize imaging data. Additionally, privacy-preserving techniques, including Differential Privacy (DP), were applied to protect individual patient information during model training. The results demonstrated that Federated Learning (FL) could enhance prediction accuracy without compromising patient confidentiality, establishing a new benchmark for future research in this field.

Xu *et al.* [35] designed a Federated Learning (FL) model to predict complications in diabetic patients, including nephropathy and retinopathy. The model was trained on clinical data from six healthcare institutions, utilizing time-series data such as blood glucose levels, blood pressure, and kidney function tests. Long Short-Term Memory (LSTM) networks were used to capture temporal patterns, resulting in more accurate predictions of diabetic complications.

Their federated model surpassed local model training, achieving a 10% improvement in prediction accuracy. A harmonization pipeline was developed to standardize data across institutions. However, training deep models in a federated environment introduced challenges related to computational cost and communication overhead. They applied selective model updates and compression techniques to reduce communication requirements while maintaining model performance. This research work demonstrated the need for scalable algorithms capable of handling high-dimensional time-series data in healthcare.

Dayan *et al.* [36] applied Federated Learning (FL) for early sepsis prediction in ICU patients across multiple hospitals. The model used vital signs such as heart rate, blood pressure, respiratory rate, and clinical notes as input features. A Gradient-Boosted Decision Tree (GBDT) algorithm was chosen for its interpretability and reliable performance in clinical prediction tasks.

Their model reduced false negatives by 15% compared to standard sepsis models. However, challenges included missing data and inconsistent documentation practices across hospitals. They implemented data imputation techniques to manage missing values effectively. This research work demonstrated that Federated Learning (FL) offers a scalable approach for early sepsis detection while overcoming data-sharing limitations between hospitals.

2.1.3 Medical Imaging and Radiology

Sheller *et al.* [37] applied Federated Learning (FL) for brain tumor segmentation using MRI datasets from multiple institutions. Their model was based on a U-Net architecture, which is commonly used for medical image segmentation. Federated Learning (FL) enabled collaborative training without requiring institutions to share sensitive patient images, preserving privacy and facilitating a large-scale multi-institutional research work that would have been infeasible due to data-sharing restrictions.

The results showed that the federated model achieved comparable performance to a centralized model, with only a 2% decrease in accuracy. The federated model demonstrated robustness when applied to diverse datasets, indicating its strong generalization capability. A secure aggregation protocol was implemented to protect model updates during transmission, preventing individual institutions from accessing others' updates. Morever, the research work emphasized the computational challenges of aggregating large-scale imaging data and proposed communication-efficient protocols to minimize network strain. Variations in MRI scanning protocols led to inconsistent segmentation results. The authors proposed domain adaptation techniques to improve the model's robustness across different imaging settings and introduced data harmonization strategies to address inter-institutional variability.

Kaissis *et al.* [38] applied a Federated Learning (FL) framework for pneumonia detection in chest X-rays using data from four hospitals. The model was based on a ResNet-50 architecture for feature extraction and classification. Federated Learning (FL) facilitated cross-institutional model training, achieving an AUC of 0.91, reflecting high prediction accuracy despite differences in imaging protocols and equipment across hospitals. One of the primary challenges was the inconsistency in image quality and resolution, which negatively impacted model performance. To overcome this, the researchers developed a preprocessing pipeline to standardize image dimensions and applied histogram equalization to enhance contrast. This preprocessing pipeline improved feature extraction and reduced the model's error rate. Furthermore, they proposed a secure communication protocol to protect sensitive information during the model update process. Kaissis *et al.* presented Federated Learning (FL) as a promising approach to improving diagnostic accuracy in radiology while safeguarding patient data. The authors concluded that for Federated Learning (FL) to be widely adopted in clinical practice, standardization of imaging protocols across institutions is essential to enhance model reliability and consistency.

Rieke *et al.* [39] focused on diagnosing retinal diseases such as diabetic retinopathy and macular degeneration using Federated Learning (FL). The model was trained on fundus images from multiple ophthalmology clinics using an ensemble of convolutional neural networks (CNNs) for feature extraction and classification. The federated model achieved an accuracy of 87%, demonstrating its potential to improve diagnostic accuracy in retinal imaging. A key contribution of this research work was the use of adaptive learning rates to address data heterogeneity across clinics. Differences in annotation standards and labeling practices introduced noise into the model, which affected the accuracy of predictions. The authors introduced consensus-based labeling strategies to reduce variability and improve model performance. Additionally, this research work emphasized the importance of continuous model updating to integrate new clinical data and adapt to evolving diagnostic criteria. Despite these challenges, it demonstrated how Federated Learning (FL) can expand access to high-quality diagnostic tools in underserved regions with limited annotated data.

Pati *et al.* [40] proposed a Federated Learning (FL) framework to improve the detection of tumor boundaries in rare cancer cases using multi-institutional imaging data. They aimed to address the lack of large datasets available for rare cancers by enabling multiple institutions to collaborate without sharing sensitive patient data. The model employed a U-Net architecture for image segmentation, achieving detection accuracy comparable to models trained on centralized data. The authors focused on the advantages of Federated Learning (FL) in enhancing the generalizability of the model due to the diversity of data from different institutions. A key contribution was the use of a secure aggregation protocol to protect the privacy and integrity of model updates. However, variations in imaging protocols and different labeling standards across institutions introduced noise into the training process. To tackle these challenges, the authors recommended incorporating domain adaptation techniques and standardized imaging protocols. Overall, the research demonstrated the potential of Federated Learning (FL) to fill the data gap in rare cancer research and facilitate more reliable diagnostic models.

Rajpurkar *et al.* [41] explored Federated Learning (FL) for radiology image classification in multi-institutional settings. They developed an AI model for identifying abnormalities in chest X-rays across hospitals without sharing raw imaging data. Federated Learning (FL) allowed the model to benefit from diverse patient populations, significantly improving generalizability compared to local model training.. A DenseNet-based architecture was used for feature extraction and classification, achieving an Area Under the Curve (AUC) of 0.93.

A key innovation was the development of an adaptive learning mechanism that adjusted model updates based on local data quality. Despite its success, variability in image acquisition protocols and differences in annotation practices affected model performance. To reduce these inconsistencies, a standardized image preprocessing pipeline was introduced. This research work demonstrated that Federated Learning (FL) offers a scalable approach for radiology diagnostics, contingent on improving interoperability between hospital systems.

2.1.4 Remote Patient Monitoring

Brisimi *et al.* [42] proposed a Federated Learning (FL) framework for monitoring chronic disease patients using wearable devices. The system detected anomalies in vital signs such as heart rate and blood pressure, triggering early warnings for potential health issues. By aggregating data from multiple healthcare providers, the model improved anomaly detection accuracy compared to traditional single-site models.

The framework supported real-time monitoring while preserving patient privacy and complying with data protection regulations. A key challenge was the variability in sampling rates and sensor accuracy across different wearable devices. The authors introduced calibration protocols and signal processing techniques to filter noise and ensure data consistency. Additionally, they proposed a multi-layered architecture for the model to prioritize alerts based on severity, reducing false alarms. This research work illustrates how Federated Learning (FL) can enhance chronic disease management by enabling personalized monitoring and early intervention.

Yang *et al.* [43] developed a Federated Learning (FL) framework for Continuous Glucose Monitoring (CGM) in diabetic patients. The model analyzed blood glucose trends and provided personalized recommendations for diet, physical activity, and medication adjustments. This real-time system reduced the risk of hyperglycemia and hypoglycemia by predicting sudden fluctuations in blood glucose levels. The researchers used a Recurrent Neural Network (RNN) architecture to capture temporal dependencies in glucose patterns, enhancing the model's ability to predict rapid changes. Federated Learning (FL) enabled data from multiple wearable devices to be utilized without compromising patient privacy, resulting in a more comprehensive prediction model. One significant challenge was the high communication frequency between devices and the central server, which caused network strain and delays in prediction updates. To mitigate this, the researchers adopted communication-efficient algorithms that selectively updated the model only when substantial changes were detected.

Additionally, they implemented data compression techniques to reduce bandwidth usage, ensuring the system remained scalable in real-world settings. Despite these improvements, this research work emphasized the need for standardized data formats across devices to minimize inconsistencies and improve the reliability of CGM-based predictions.

2.1.5 Clinical Decision Support Systems (CDSS)

Thwal *et al.* [44] designed a personalized Clinical Decision Support System (CDSS) using a Federated Learning (FL) framework to manage diverse clinical datasets while preserving patient privacy. This research work highlights the challenges of traditional centralized models, which require data aggregation and can compromise data security. To address these issues, they introduced a sequence-to-sequence neural network architecture enhanced with an attention mechanism. This framework allowed healthcare providers to perform local model training on-site while contributing to a global model without sharing raw data.

The CDSS model was applied to various clinical tasks, including diagnosis support and personalized treatment recommendations. By incorporating edge AI, the system reduced communication overhead and minimized data transfer between nodes. A notable feature was the model’s capacity to adapt to new clinical data without requiring full system retraining. However, computational constraints at the edge level posed challenges to scalability in resource-limited environments. The authors recommended integrating more efficient model update strategies to reduce these challenges. This work demonstrates the potential of Federated Learning (FL) to transform clinical decision-making by enabling personalized, privacy-preserving support for healthcare providers.

Mondrejevski *et al.* [45] developed FLICU, a Federated Learning (FL) workflow designed for predicting mortality in Intensive Care Unit (ICU) patients. They focused on overcoming the barriers of data-sharing restrictions in critical care by enabling institutions to collaboratively train predictive models without exposing sensitive patient information. The authors used multivariate time-series data, including lab results and vital signs, extracted from the MIMIC-III database.

Four deep learning models—Recurrent Neural Network (RNN), Long Short-Term Memory (LSTM), Gated Recurrent Unit (GRU), and 1D Convolutional Neural Network (1DCNN)—were benchmarked for performance across various patient history lengths. The federated models consistently outperformed local model training and achieved comparable results to centralized models in terms of Area Under the Precision-Recall Curve (AUPRC) and F1-score. A key contribution of this research work was the implementation of a data harmonization strategy to manage inconsistent data formats across participating institutions. The authors also emphasized the importance of efficient communication protocols to reduce network strain during model updates. This research provides strong evidence that Federated Learning (FL) can be a practical solution for developing predictive models in ICU settings, where privacy and data security are critical.

2.1.6 Drug Discovery and Genomics

Chen *et al.* [46] proposed FL-QSAR, a Federated Learning (FL) framework aimed at improving Quantitative Structure-Activity Relationship (QSAR) modeling in drug discovery. Traditional QSAR models rely on centralized data aggregation, raising concerns about To overcome these issues, the authors implemented a Horizontal Federated Learning (HFL) approach, allowing multiple pharmaceutical institutions to collaboratively train machine learning models without the need to sharing raw data. This approach kept data secure within each organization while contributing to a shared global model.

The performance of FL-QSAR was compared to traditional centralized models, showing that the federated approach achieved similar predictive accuracy while preserving data confidentiality. One of the key contributions of this work was the development of a standardized molecule encoding protocol to harmonize the chemical data across institutions, improving model compatibility and performance. To reduce network load during model updates, a communication-efficient optimization algorithm was integrated. This work illustrates the potential of Federated Learning (FL) to facilitate secure, large-scale collaborations in drug discovery, enabling faster identification of drug candidates while protecting sensitive data.

Teo *et al.* [47] explored the use of Federated Learning (FL) to accelerate drug discovery for rare diseases. The model aggregated data from pharmaceutical companies and academic research centers to identify potential drug candidates. By combining molecular structure data and high-throughput screening results, the model improved prediction accuracy for drug efficacy and toxicity.

The federated approach tackled the problem of data fragmentation in drug discovery, enabling collaboration across institutions without centralizing proprietary data. A deep neural network architecture was employed to capture complex relationships between chemical compounds and biological targets. Key challenges included data standardization and the high computational cost of training large models. The authors introduced adaptive learning rates and data compression techniques to optimize performance. Teo *et al.* concluded that Federated Learning (FL) could significantly reduce the time and cost associated with drug discovery, particularly for rare and understudied diseases.

2.1.7 Summary

This section provides an overview of the findings from the reviewed literature on Federated Learning (FL) in healthcare, categorizing them by key challenges and common limitations. As shown in Table 2.1, the main challenges in applying FL in healthcare include data security, heterogeneous data, interoperability and model convergence. Overcoming these challenges is essential for the successful deployment of FL in real-world healthcare settings.

Data Security and Privacy Risks: While the decentralized nature of Federated Learning (FL) minimizes the need for data sharing, security concerns remain. Model updates can inadvertently expose sensitive information about local datasets. Several research works, particularly in clinical decision support systems and remote patient monitoring, emphasize the importance of secure aggregation protocols and privacy-preserving mechanisms such as Differential Privacy (DP) to safeguard patient data (cf. §2.1.2, §2.1.4, and §2.1.5) [34] [36] [44].

Data Heterogeneity: Successful FL deployment depends on seamless interoperability between healthcare systems. However, many research works overlook this issue, resulting in fragmented models that cannot integrate with standard health platforms. This challenge is especially critical in remote monitoring systems and clinical decision support, where heterogeneous data sources must be combined to generate consistent outputs (cf. §2.1.2 and §2.1.3) [35-37] [40].

Interoperability Limitations: Interoperability between healthcare systems is essential for the effective implementation of Federated Learning (FL). However, many research works fall short in addressing this requirement, leading to fragmented models that struggle to integrate with standard health platforms. This challenge is especially critical in remote monitoring systems and Clinical Decision Support Systems (CDSS) (cf. §2.1.4 and §2.1.5) [41-43].

Model Convergence: Non-IID (non-independent and identically distributed) data distributions are a common challenge in FL, slowing down model convergence and reducing performance. Personalized Federated Learning (PFL) frameworks and adaptive learning rates have been proposed to address this issue by improving convergence and tailoring global models to local data distributions, especially in disease prediction and remote patient monitoring (cf. §2.1.2 and §2.1.4) [35] [43].

Table 2.1 Summary of Federated Learning Applications in Healthcare

Application	Reference	Contributions	Limitations
Disease Prediction and Diagnosis	Li *et al.* [34], Xu *et al.* [35], Dayan *et al.* [36]	Improved prediction accuracy for cardiovascular diseases, diabetes complications, and sepsis.	Data heterogeneity, inconsistent data formats, communication overhead, and data security risks in federated updates.
Medical Imaging and Radiology	Sheller *et al.* [37], Kaissis *et al.* [38], Rieke *et al.* [39]	Enhanced diagnostic accuracy for brain tumor segmentation, chest X-rays, and retinal diseases.	Imaging protocol variation, data inconsistency, lack of standardization, and secure aggregation protocol challenges.
Remote Patient Monitoring	Brisimi *et al.* [42], Yang *et al.* [43]	Real-time anomaly detection and personalized monitoring for chronic diseases and diabetes.	Inconsistent device data quality, vulnerability to data breaches in IoMT networks, and network load.
Clinical Decision Support Systems	Thwal *et al.* [44], Mondrejevski *et al.* [45]	Personalized support for clinical decision-making and ICU mortality prediction.	High computational costs, edge-level constraints, and privacy risks from data aggregation in distributed systems.
Drug Discovery and Genomics	Chen *et al.* [46], Teo *et al.* [47]	Accelerated drug discovery and personalized medicine predictions.	Data standardization, computational costs, and protection of proprietary data.

While these applications demonstrate FL’s potential in healthcare, challenges such as data standardization, non-IID distributions, and model convergence across heterogeneous institutions must be addressed to facilitate real-world deployment. Additionally, privacy risks persist despite decentralized learning, necessitating robust privacy-preserving mechanisms. The next section explores these key challenges in more detail.

2.2 Challenges in Federated Learning for Healthcare

Federated Learning (FL) presents a transformative opportunity for advancing healthcare while maintaining data privacy and security. However, its real-world implementation is not without challenges. These challenges can affect data quality, model accuracy, communication efficiency, and overall system security. This section identifies and discusses the main obstacles hindering the practical adoption of FL in healthcare, focusing on data heterogeneity, communication overhead and security and privacy risks.

2.2.1 Data Heterogeneity and Non-IID Data Distributions

One of the core challenges in Federated Learning (FL) for healthcare is data heterogeneity, where datasets from different institutions differ in format, quality, and representation. This heterogeneity arises from varying clinical protocols, imaging devices, and data collection practices, making it difficult for models to generalize across all datasets. Additionally, healthcare data is rarely IID (independent and identically distributed), which further complicates the learning process and leads to poor convergence.

Kairouz *et al*. [48] provided a comprehensive overview of these challenges, emphasizing data heterogeneity as a major barrier to Federated Learning (FL). They discussed how variations in patient demographics, imaging protocols, and data collection methods across institutions hinder model generalization and slow down convergence. When data is non-IID, the learning process becomes even more complex, leading to performance degradation compared to centralized models. Addressing these discrepancies requires strategies that facilitate better alignment of distributed datasets.

Zhao *et al*. [49] explored this issue in multi-institutional healthcare settings, focusing on the role of data harmonization in mitigating heterogeneity. Their research demonstrated that standardizing input data significantly improves both model performance and convergence. This effect is particularly evident in time-series data, such as glucose monitoring and vital signs, where variations in measurement intervals and device calibration introduce noise. By implementing consistent preprocessing techniques across sites, Federated Learning (FL) frameworks become more robust and adaptable, ultimately enhancing their predictive capabilities in real-world clinical applications.

2.2.2 Communication and Computational Overhead

In Federated Learning (FL), communication overhead and computational efficiency shape the feasibility of large-scale deployments, particularly in resource-constrained healthcare environments. Model updates travel between local clients, such as hospitals or medical devices, and the central server at frequent intervals. As these exchanges scale up, network congestion increases, leading to delays that disrupt real-time decision-making and strain system resources.

Bonawitz *et al*. [50] examined the challenges of communication efficiency in large-scale Federated Learning (FL) framework. They proposed secure aggregation protocols with communication-efficient algorithms to reduce bandwidth usage while maintaining model performance. These solutions minimize network strain but introduce new trade-offs in accuracy.

Sattler *et al*. [51] explored the computational cost of training deep learning models in federated environments, particularly when Differential Privacy (DP) is applied. Their research introduced gradient compression techniques and adaptive communication strategies to balance computational demands without compromising privacy. These approaches resonate with efforts to reduce model update frequency while maintaining strong privacy protections.

2.2.3 Security and Privacy Risks

While Federated Learning (FL) mitigates some data-sharing risks by keeping raw data local, it is not immune to privacy threats. Instead of attackers targeting centralized databases, they now focus on model updates as a potential source of sensitive information. This shift introduces new vulnerabilities that could expose underlying patient data, raising serious concerns about privacy and security.

Hitaj *et al.* [52] analyzed these risks, revealing how malicious actors could infer private patient information simply by analyzing subtle patterns in model updates. Their findings highlighted that even when institutions retain their data locally, certain techniques could still be used to reconstruct aspects of the training data. This underscores the urgent need for enhanced privacy-preserving mechanisms that go beyond simple data decentralization. One major concern is that model updates can inadvertently leak sensitive details about the data used for training. Shokri *et al.* [53] demonstrated how models trained on medical records could unintentionally encode identifiable patterns, making it possible to extract information that should remain private. To counter this, they emphasized the importance of Differential Privacy (DP), which introduces carefully calibrated noise to ensure that no single data point can be distinguished from the rest.

However, while Differential Privacy (DP) strengthens security, it often comes at the cost of model performance. Excessive noise can reduce accuracy, creating a difficult trade-off between privacy and utility. So *et al.* [54] proposed secure aggregation protocols as an additional safeguard. Their method ensures that model updates from multiple institutions are encrypted and combined before being shared, preventing any single party from accessing another institution's contributions. While Secure Aggregation (SA) ensures privacy by encrypting local updates, it introduces computational and communication overhead that makes it impractical for large-scale healthcare FL systems. The encryption and decryption process significantly increases resource consumption, which is especially problematic for real-time medical applications where efficiency is crucial. Instead, Differential Privacy (DP) is the privacy-preserving mechanism due to its ability to mitigate privacy risks without imposing excessive computational costs. DP provides strong privacy guarantees by introducing controlled noise to model updates before they are shared, reducing the risk of data reconstruction while maintaining practical model utility. However, DP comes with its own challenges, particularly in balancing privacy strength (ε) with model accuracy, which will be explored further in Section 2.3.

2.3 Privacy-Preserving Mechanisms in Federated Learning

In Federated Learning (FL), protecting sensitive data is crucial, especially in privacy-sensitive fields like healthcare. Although FL minimizes the risk of data exposure by enabling collaborative model training without sharing raw data, it does not eliminate privacy threats. Adversaries can exploit vulnerabilities in model updates through attacks like membership inference, which reveals whether specific data is part of the training set. Model inversion is another attack that attempts to reconstruct sensitive data from gradients. To mitigate these risks, various privacy-preserving mechanisms have been developed. This section reviews three key techniques—Secure Multi-Party Computation (SMPC), Homomorphic Encryption (HE), and Differential Privacy (DP), the primary focus of this research. The following sections will delve deeper into these privacy-preserving mechanisms in FL, focusing on their role, trade-offs between privacy and utility and how they contribute to a more secure Federated Learning (FL) environment.

2.3.1 Secure Multi-Party Computation (SMPC)

Secure Multi-Party Computation (SMPC) is a cryptographic mechanism that allows multiple parties to jointly compute a function over their private data without revealing their inputs to one another. This technique ensures that sensitive data remains confidential throughout the computation process, making it a widely used privacy-preserving mechanism in Federated Learning (FL) [55]. Secure Multi-Party Computation (SMPC) is particularly beneficial in settings where institutions, such as hospitals or research organizations, seek to collaborate on model training while preserving data privacy.

Goldreich *et al.* [56] introduced secure computation protocols that allow multiple parties to jointly perform computations while keeping their inputs secret. These protocols form the foundation of modern SMPC applications, including those in FL. In the context of FL, SMPC ensures that each participating client secret-shares its local model training updates using cryptographic mechanisms. The server can only access the aggregated result of all model updates, preventing any individual client's contribution from being reconstructed. This mechanism is particularly useful in healthcare applications, where patient data privacy is crucial.

Damgård *et al.* [57] introduced a multi-party replicated secret sharing scheme that enhances performance in Secure Multi-Party Computation (SMPC), particularly for privacy-preserving machine learning applications. Unlike traditional secret sharing methods, which rely on field-based operations, their approach operates over a ring structure, improving computational efficiency and making SMPC more scalable. The scheme ensures that data is split into multiple shares, which alone are meaningless, but when a sufficient number of shares are combined, the original data can be reconstructed securely. This technique has been widely adopted in privacy-preserving AI applications where maintaining strict confidentiality of model updates is essential.

Kolesnikov *et al.* [58] presented significant improvements in garbled circuits, a technique enabling secure function evaluation over encrypted inputs without revealing intermediate values. Their work focused on optimizing garbled circuits to reduce computational overhead and enhance efficiency, addressing key limitations of earlier implementations. By introducing methods to minimize the size of garbled tables and enhance the speed of secure computations, they made garbled circuits more practical for large-scale Federated Learning (FL) environments. These optimizations have led to the broader adoption of garbled circuits in secure machine learning frameworks, ensuring privacy without excessive computational costs.

Mohassel *et al.* [59] demonstrated that SMPC provides strong cryptographic privacy guarantees without relying on noise-based obfuscation like Differential Privacy (DP). This ensures that exact computation results are preserved, which is particularly beneficial in medical AI applications where precision is crucial. Similarly, Kairouz *et al.* [48] emphasized that SMPC is well-suited for multi-institutional FL applications, such as collaborations between hospitals and research labs, where parties may not fully trust each other. Compared to other privacy-preserving mechanisms, SMPC ensures that each party's local model training updates remain encrypted and inaccessible to others, mitigating membership inference attacks and model inversion attacks [53].

Despite its strong privacy guarantees, SMPC introduces significant computational and communication overhead. Konecny *et al.* [60] noted that SMPC-based Federated Learning (FL) requires multiple rounds of secure data exchanges, leading to increased latency and network congestion. This makes it impractical for large-scale FL applications where thousands of clients participate. Zhao *et al.* [49] further explored that SMPC scales poorly with an increasing number of participants, as each additional client increases both computation time and communication complexity. In a comparative research, Shokri *et al.* [53] found that while SMPC provides higher privacy protection than DP, it significantly increases training latency, making real-time FL applications difficult. They suggested that hybrid mechanisms combining SMPC with Differential Privacy (DP) could improve scalability while maintaining privacy guarantees.

Several FL frameworks have adopted SMPC to enhance privacy. Mohassel *et al.* [59] implemented a Privacy-Preserving FL (PPFL) using SMPC for collaborative machine learning, demonstrating its feasibility in securing sensitive model updates. Similarly, Aono *et al.* [61] explored SMPC in healthcare FL, showing that it effectively prevents data leakage in hospital collaborations but incurs high computational costs when applied to deep learning models. While SMPC provides strong cryptographic privacy, its computational and communication costs limit its scalability in real-world FL applications. Homomorphic Encryption (HE) offers an alternative mechanism, allowing computations on encrypted data without decryption. The following section explores HE in detail, comparing its benefits and limitations to SMPC.

2.3.2 Homomorphic Encryption (HE)

Homomorphic Encryption (HE) is a cryptographic mechanism that enables computations to be performed directly on encrypted data without requiring decryption. This property is particularly useful in Federated Learning (FL), where model updates need to remain confidential while still allowing for aggregation and training across multiple institutions. HE ensures that even if a central server is compromised, sensitive data remains protected, making it a strong candidate for Privacy-Preserving FL (PPFL) applications [62].

Gentry *et al.* [62] introduced the first practical Fully Homomorphic Encryption (FHE) scheme, which allows arbitrary computations to be performed on encrypted data. This breakthrough made it possible to conduct privacy-preserving computations without exposing raw inputs. In FL, HE is applied to encrypt local model training updates before they are sent to the aggregation server. The server then performs aggregation operations on the encrypted values, producing an encrypted result that can only be decrypted by the original data owners. There are three main types of HE:

- Partially Homomorphic Encryption (PHE): Supports either addition or multiplication but not both (e.g., RSA, ElGamal encryption).
- Somewhat Homomorphic Encryption (SHE): Allows a limited number of both addition and multiplication operations.
- Fully Homomorphic Encryption (FHE): Supports an unlimited number of operations on encrypted data but at a high computational cost.

Gentry *et al*. [62] demonstrated that FHE provides strong data confidentiality since encrypted values remain secure throughout the computation process. Similarly, Brakerski *et al.* [63] highlighted that HE eliminates the risk of data exposure at the server level, making it suitable for scenarios where institutions cannot fully trust a central aggregator. Compared to Secure Multi-Party Computation (SMPC), HE requires less communication overhead because encrypted model updates can be aggregated directly without multi-party interactions. This makes HE a more efficient privacy-preserving solution in resource-constrained environments where communication latency is a concern.

Despite its strong security guarantees, HE introduces significant computational overhead. The encryption and decryption processes in Fully Homomorphic Encryption (FHE) are computationally expensive, making them impractical for large-scale deep learning applications. Acar *et al.* [64] noted that while Partially Homomorphic Encryption (PHE) schemes such as RSA and ElGamal are computationally more efficient, they do not support arbitrary computations, limiting their applicability in complex FL models. Additionally, HE-based FL frameworks require trusted key management systems to prevent unauthorized access to decryption keys, adding another layer of complexity.

In a comparative research, Chillotti *et al.* [65] found that HE-based FL models suffer from a 10x–100x increase in computational cost compared to non-encrypted training. This performance bottleneck has hindered the adoption of HE in real-world FL systems, leading researchers to explore hybrid approaches that combine HE with Differential Privacy (DP) to balance privacy and efficiency. Several Privacy-Preserving FL (PPFL) architectures have incorporated HE for secure model aggregation. Paillier *et al.* [66] developed an additive homomorphic encryption scheme that has been widely adopted in privacy-sensitive applications such as medical AI and financial risk analysis. Chillotti *et al*. [65] implemented an optimized BFV-based HE scheme for encrypted machine learning, demonstrating its feasibility in securing federated healthcare models while maintaining practical performance.

While HE offers strong data security, its high computational cost limits its practicality in large-scale FL applications. Differential Privacy (DP) offers a software-based alternative, ensuring privacy by adding statistical noise to model updates. The next section explores how DP enhances privacy in FL and compares its efficiency to other privacy-preserving mechanisms.

2.3.3 Differential Privacy (DP)

While cryptographic mechanisms such as Secure Multi-Party Computation (SMPC) and Homomorphic Encryption (HE) focus on securing computations, Differential Privacy (DP) takes a fundamentally different approach by ensuring that individual contributions to a dataset remain mathematically untraceable. Unlike encryption-based methods, which require complex cryptographic operations and significant computational resources, DP provides privacy guarantees with lower computational overhead, making it a highly practical solution for Federated Learning (FL).

FL involves multiple decentralized clients collaboratively training a shared model without exchanging raw data. However, the iterative communication of model updates exposes privacy risks, as adversaries can exploit updates to infer sensitive information. DP mitigates these risks by adding carefully calibrated noise to data or model updates, ensuring that an attacker cannot determine whether a specific data point was included in the training process. Given its balance between privacy, efficiency, and scalability, DP has become a widely adopted privacy-preserving mechanism in FL, offering a viable alternative to computationally expensive cryptographic mechanisms. A fundamental concept in DP is the privacy budget (ε), which quantifies the level of privacy protection. A smaller ε value provides stronger privacy guarantees but may reduce model accuracy, whereas a larger ε improves utility at the cost of weaker privacy protection.

Dwork *et al.* [67] introduced ε-Differential Privacy, a mathematical mechanism that ensures an attacker cannot determine whether a particular data point was included in a dataset. This is achieved by adding noise to computations, making statistical outputs indistinguishable regardless of an individual's presence in the dataset. Their research introduced key concepts such as the privacy budget (ε), a parameter that quantifies the trade-off between privacy and utility. A lower ε provides stronger privacy but reduces accuracy, while a higher ε preserves utility at the cost of weaker privacy guarantees. Another essential concept is global sensitivity, which measures how much a single record can influence the output of a function, guiding noise calibration. DP also employs mechanisms such as the Laplace and Gaussian distributions to introduce noise, ensuring a balance between privacy protection and data utility. This framework has been widely adopted in real-world applications, including Google's data systems, the U.S. Census Bureau's data collection practices, and Apple's iOS analytics.

In FL, DP enhances privacy during the model training process by protecting data at the gradient or model parameter level. Unlike centralized machine learning, which often exposes raw data to a central server, FL integrates DP by adding noise to local model training updates before they are shared with the central aggregator [68]. There are two main approaches to applying DP in FL. Global Differential Privacy (Centralized DP) adds noise at the server level after receiving aggregated updates from clients [69]. This method ensures that the final model does not reveal sensitive information from any individual participant. In contrast, Local Differential Privacy (LDP) requires each client to add noise before transmitting updates, ensuring that the central server never sees raw gradients [80]. While LDP provides stronger privacy guarantees, it introduces higher noise levels that can impact accuracy.

One of the most widely used DP mechanisms in FL is Differentially Private Stochastic Gradient Descent (DP-SGD). Abadi *et al.* [68] proposed DP-SGD as a privacy-preserving variant of stochastic gradient descent, where Gaussian noise is added to gradients at each training step. To limit the sensitivity of individual updates before adding noise, gradient clipping is applied. This approach prevents adversaries from reconstructing training data through model inversion attacks while ensuring strong privacy guarantees. However, DP-SGD introduces challenges in balancing privacy and model performance. Excessive noise can lead to unstable model convergence and reduced accuracy, requiring careful tuning of the privacy budget (ε).

McMahan *et al.* [69] introduced Differentially Private Federated Averaging (DP-FedAvg), an extension of the Federated Averaging (FedAvg) algorithm that applies global DP to aggregated model updates. Instead of adding noise at the client level, as in DP-SGD, noise is applied only after aggregation at the server. This reduces overall noise accumulation and maintains higher model utility while still enforcing DP guarantees. However, DP-FedAvg offers weaker privacy protection compared to LDP-based methods and remains vulnerable to gradient leakage before aggregation.

Xie *et al.* [71] proposed an LDP-based FL framework where each client applies DP locally before sending updates. This method ensures that even a compromised aggregator cannot access sensitive information. While LDP provides the strongest level of privacy in FL, it significantly impacts model accuracy due to the accumulation of noise. Moreover, it increases communication overhead, as clients must adjust updates to compensate for the noise.

While DP provides strong privacy guarantees, it introduces several trade-offs that must be carefully managed. McMahan *et al.* [69] observed that lower ε values provide stronger privacy but can significantly degrade model accuracy, particularly in deep learning applications. Nasr *et al.* [72] noted that DP does not protect against side-channel attacks, which exploit auxiliary information to infer private data. Additionally, DP-SGD requires gradient clipping to limit sensitivity before adding noise, which can slow down convergence and prevent models from learning complex patterns, especially in non-IID federated datasets [70]. Local DP, while providing the highest privacy guarantees, increases communication overhead, making it impractical for large-scale FL deployments.

Several innovations have been proposed to optimize DP in FL. Wei *et al.* [73] introduced Noising before Aggregation Federated Learning (NbAFL), a technique where noise is added before transmission, reducing privacy loss while improving training convergence. Guo *et al.* [74] proposed a Fast Fourier Transform (FFT)-based DP architecture where noise is applied in the frequency domain, reducing its impact on model accuracy while preserving privacy.

Nasr *et al.* [72] explored hybrid DP mechanisms that combine DP with Secure Aggregation (SA) and TEE-based FL to mitigate accuracy trade-offs while maintaining strong privacy guarantees. These advancements highlight ongoing efforts to optimize DP for practical FL applications, ensuring privacy without compromising model utility.

Overall, Differential Privacy (DP) has emerged as one of the most scalable and computationally efficient privacy mechanisms for Federated Learning (FL). Unlike cryptographic approaches such as Secure Multi-Party Computation (SMPC) and Homomorphic Encryption (HE), DP ensures privacy through noise addition rather than resource-intensive encryption, making it more suitable for large-scale FL deployments.

However, its impact on model accuracy must be carefully managed. While Local Differential Privacy (LDP) offers the strongest privacy guarantees by obfuscating individual client updates before transmission, it significantly degrades model utility due to the accumulation of excessive noise. In contrast, DP-FedAvg introduces noise at the aggregated level, achieving a better trade-off between privacy protection and model performance. As a result, DP-FedAvg has become one of the most practical and widely adopted privacy-preserving mechanisms in Federated Learning (FL) applications, particularly in healthcare environments.

2.3.4 Summary of Privacy-Preserving Mechanisms in Federated Learning

In Section 2.3, we reviewed various privacy-preserving mechanisms in Federated Learning (FL), focusing on their mechanisms, strengths, and limitations. Based on the literature, privacy-preserving mechanisms in FL can be categorized into three primary methods: Secure Multi-Party Computation (SMPC), Homomorphic Encryption (HE), and Differential Privacy (DP).

Secure Multi-Party Computation (SMPC) ensures privacy through cryptographic mechanisms that allow multiple participants to collaborate on computations without exposing their individual data. While SMPC provides strong security guarantees, its high computational and communication overhead makes it less scalable for large-scale FL applications.

Homomorphic Encryption (HE) enables computations to be performed on encrypted data, eliminating the need to decrypt model updates during training. Despite its strong privacy guarantees, HE introduces significant computational costs, making it impractical for real-time FL deployments due to slow processing times.

Differential Privacy (DP) takes a different approach by adding controlled noise to model updates before they are shared. This ensures that individual contributions remain untraceable, providing strong privacy while maintaining computational efficiency. Although DP introduces a privacy-accuracy trade-off, it remains the most widely adopted privacy-preserving mechanism in FL due to its low overhead and scalability.

Table 2.2 presents a comparison of these privacy-preserving mechanisms in FL, evaluating them based on privacy guarantees, computational cost, scalability, and overall practicality.

Table 2.2 Comparison of Privacy-Preserving Techniques in Federated Learning

Privacy Technique	Privacy Guarantees	Computational Cost	Scalability	Practicality in FL
Secure Multi-Party Computation (SMPC)	Strong cryptographic privacy	High	Low	Limited due to high overhead
Homomorphic Encryption (HE)	Strong encryption-based privacy	Very High	Low	Limited due to encryption complexity
Differential Privacy (DP)	Privacy preserved by adding noise to data	Low	High	Most widely used due to efficiency

2.4 Privacy-Utility Trade-offs in Federated Learning

Federated Learning (FL) enables decentralized model training while maintaining data privacy. However, integrating privacy-preserving mechanisms introduces a fundamental trade-off between privacy and model utility. While privacy ensures data confidentiality, stronger privacy protections often come at the cost of reduced model accuracy, slower convergence, and increased training complexity. Striking the right balance between privacy strength and model performance remains a critical challenge in Privacy-Preserving FL (PPFL). Among privacy-preserving mechanisms, Differential Privacy (DP) has emerged as a widely used mechanism due to its strong privacy guarantees and computational efficiency. DP achieves privacy by adding noise to model updates, preventing adversaries from extracting sensitive information. However, increasing noise reduces the model's ability to learn meaningful patterns, leading to a decrease in accuracy and slower convergence.

Dwork *et al.* [67] first introduced DP with the privacy budget (ε), a parameter that defines the trade-off between privacy and accuracy:

- Lower ε (Stronger Privacy) → More noise, greater privacy protection, but lower model accuracy.
- Higher ε (Weaker Privacy) → Less noise, better model utility, but increased privacy risks.

McMahan *et al.* [69] studied the effects of DP on FL models and found that lower ε values (strong DP settings) led to a 5–15% accuracy drop compared to non-private models. Similarly, Abadi *et al.* [68] proposed Differentially Private Stochastic Gradient Descent (DP-SGD), demonstrating that excessive noise disrupts training dynamics, causing slower convergence and decreased model generalization.

In FL, privacy strength is controlled by adjusting the noise multiplier in DP mechanisms. The noise multiplier determines how much noise is added to model updates—higher noise levels strengthen privacy but degrade accuracy.

Several research works have examined how different noise multipliers impact privacy budgets (ε) and model performance:

- Zhu *et al.* [75] found that high noise multipliers (≥1.0) in DP-FedAvg resulted in a 15–20% drop in model accuracy, whereas moderate noise multipliers (0.1–0.5) achieved a better balance between privacy and accuracy.
- Nasr *et al.* [72] emphasized that DP-based FL models with $\varepsilon < 1$ provide strong privacy guarantees but significantly degrade model utility, making them impractical for high-performance applications.

Kim *et al.* [76] analyzed the trade-offs between user privacy, global utility, and transmission rates in FL systems employing Local Differential Privacy (LDP). They demonstrated that enhancing privacy protection often leads to a decrease in model utility and an increase in communication costs. Their work emphasizes the importance of carefully selecting privacy parameters to balance these competing factors effectively.

Zhang *et al.* [77], in their research work *Trading Off Privacy, Utility, and Efficiency in Federated Learning (FL)*, formulate and quantify the inherent trade-offs in FL systems. They introduce the No-

Free-Lunch (NFL) theorem, which asserts that it is unrealistic to expect FL to simultaneously provide excellent privacy, utility, and efficiency in certain scenarios. This theorem underscores the necessity of making informed compromises based on specific application requirements. Zhang *et al.* further expanded on this idea by analyzing the impact of different privacy-preserving mechanisms on model convergence and communication efficiency. Their findings demonstrate that while stronger privacy measures enhance data security, they often result in slower convergence rates and increased computational costs, particularly in resource-constrained environments.

A systematic review by Fu *et al.* [78] shows that while DP offers strong privacy guarantees in FL, it often comes at the cost of reduced model accuracy. The review suggests that optimizing the privacy-utility trade-off requires innovative approaches, such as adaptive noise addition and personalized privacy budgets, to mitigate the adverse effects on model performance.

The privacy-utility trade-off in FL is particularly critical for healthcare applications, where patient confidentiality must be preserved while ensuring model reliability. research works show that:

- Values of ε between 1 and 5 provide strong privacy but can significantly reduce model accuracy if not optimized. Research has shown that $\varepsilon = 1.0$ leads to a model accuracy drop to approximately 61.33%, demonstrating strong privacy but notable accuracy loss [79].
- Values of ε between 5 and 10 are often considered a good balance between privacy protection and data utility in federated healthcare AI models. Research shows that $\varepsilon = 8.0$ allows models to maintain an accuracy of 64.50%, suggesting that moderate ε values can provide both privacy and acceptable model performance [79].
- Values of $\varepsilon > 10$ may not provide sufficient privacy protection for sensitive healthcare applications, as adversaries could potentially reconstruct patient data from model updates. While research works on $\varepsilon > 10$ in healthcare FL are limited, Differential Privacy (DP) literature consistently suggests that higher ε values correspond to weaker privacy guarantees.

Recent research has examined different DP configurations to mitigate this trade-off while preserving model utility. research works have shown that selecting an appropriate noise multiplier is crucial for balancing privacy and accuracy. Moderate noise multipliers (e.g., 0.1–0.5) in DP-FedAvg have been found to achieve a reasonable balance, preventing excessive accuracy degradation while maintaining strong privacy guarantees [80].

2.4.1 Summary of Privacy-Utility Trade-offs

Balancing privacy and model utility is essential in federated healthcare AI. Differential Privacy (DP) protects patient data but reduces accuracy as noise increases [69]. Low privacy budgets ($\varepsilon < 1$) provide strong privacy but severely degrade accuracy, while high values ($\varepsilon > 10$) weaken privacy protections [72]. Moderate budgets ($5 \leq \varepsilon \leq 10$) achieve the best balance and reliable model performance [75].

Selecting the right noise multiplier further minimizes accuracy loss while maintaining privacy [69]. The privacy-utility trade-off in FL must be carefully managed, especially in healthcare, where both privacy protection and model reliability are crucial. The following table summarizes these findings.

Table 2.3 Privacy-Utility Trade-offs in Federated Learning

Reference	**Privacy Budget (ε)**	**Privacy Level**	**Model Utility Impact**	**Effectiveness in Healthcare**
Nasr *et al.* [72]	$\varepsilon < 1$	Very Strong	Severe degradation in model performance	Not practical for real-world FL
Zhu *et al.* [75]	$1 \leq \varepsilon \leq 5$	Strong	Moderate decrease in model utility	Suitable for privacy-sensitive settings
Abadi *et al.* [68]	$5 \leq \varepsilon \leq 10$	Balanced	Minimal impact on model performance	Widely adopted in healthcare FL
Nasr *et al.* [72]	$\varepsilon > 10$	Weak	Low impact, but weaker privacy protection	High privacy risk

2.5 Limitations and Future Directions in Privacy-Preserving FL

Federated Learning (FL) has emerged as a promising solution for privacy-preserving collaborative model training, particularly in healthcare. However, several challenges hinder its large-scale adoption. One of the most pressing issues is achieving an optimal balance between privacy protection and model accuracy, especially when employing Differential Privacy (DP). While DP safeguards patient data by adding noise to model updates, this often results in a privacy-utility trade-off, where excessive noise degrades model accuracy and slows down convergence. Nasr *et al.* [72] emphasized that while FL reduces the need for raw data sharing, it remains vulnerable to privacy attacks, such as inference attacks, particularly when insufficient noise is applied. Moreover, research works indicate that strong privacy settings ($\varepsilon < 1$) can severely impair model accuracy, making them impractical for high-performance healthcare applications [69].

Another significant challenge in Privacy-Preserving FL (PPFL) for healthcare is data heterogeneity. Medical datasets are inherently non-IID, as hospitals and clinics collect data under varying conditions, using different imaging techniques, sensor types, and patient demographics. This variation complicates model training, leading to poor generalization and slow convergence [49]. DP exacerbates this issue, as the added noise disproportionately affects small, non-IID datasets, leading to biased model updates. A promising research direction is privacy-aware aggregation techniques that adaptively adjust noise levels based on the statistical properties of local datasets rather than applying uniform noise across all participants.

Computational and communication overhead further limits FL's scalability in healthcare environments. DP-based FL mechanisms require frequent model updates with added noise, significantly increasing computational demands on client devices [51]. This is particularly problematic for resource-constrained healthcare institutions, such as small clinics with limited computational resources. Additionally, the communication overhead from frequent DP-enforced updates can strain networks, leading to slower model convergence. Future research should explore efficient noise mechanisms that reduce DP's computational cost while maintaining privacy.

Finally, privacy budget selection remains an open research challenge in Privacy-Preserving FL (PPFL). Choosing an appropriate ε value is crucial, as smaller values provide stronger privacy but lead to substantial accuracy loss, whereas larger values risk exposing sensitive patient information [67]. While existing research works suggest that ε values between 5 and 10 offer a reasonable balance between privacy and model utility [75], the optimal selection of ε remains application-dependent, particularly in privacy-sensitive fields like healthcare. McMahan *et al.* [69] demonstrated that selecting an appropriate noise multiplier is crucial to maintaining model accuracy while ensuring privacy protection. However, current research lacks standardized guidelines for determining the best ε values across different FL applications in healthcare. Future research works should further investigate the impact of varying privacy budgets on clinical AI models and develop empirical benchmarks to optimize privacy-utility trade-offs in FL-based healthcare applications.

In summary, While DP-based FL provides strong privacy guarantees, several challenges must be addressed for practical deployment in healthcare. Ensuring an optimal balance between privacy and accuracy, improving model convergence under non-IID conditions, reducing computational and communication costs, and refining privacy budget selection are key areas requiring further investigation. Future advancements should focus on fine-tuning DP parameters and developing scalable Privacy-Preserving Federated Learning (PPFL) architectures that protect patient data while maintaining high model utility in real-world healthcare settings.

CHAPTER 3 PROPOSED ARCHITECTURE

In this chapter, we present the proposed Privacy-Preserving Federated Learning (PPFL) architecture, which enhances privacy and security in collaborative machine learning for healthcare applications. This section provides an overview of the PPFL architecture, enabling multiple hospitals to collaboratively train a machine learning model without sharing raw patient data. The architecture is designed to mitigate privacy risks and maintain data confidentiality. The proposed architecture consists of three primary components, as illustrated in Figure 3.1:

1. Local Model Training at Hospitals;
2. Privacy-Preserving Mechanism using Differential Privacy (DP);
3. Federated Model Aggregation with Global Model Distribution.

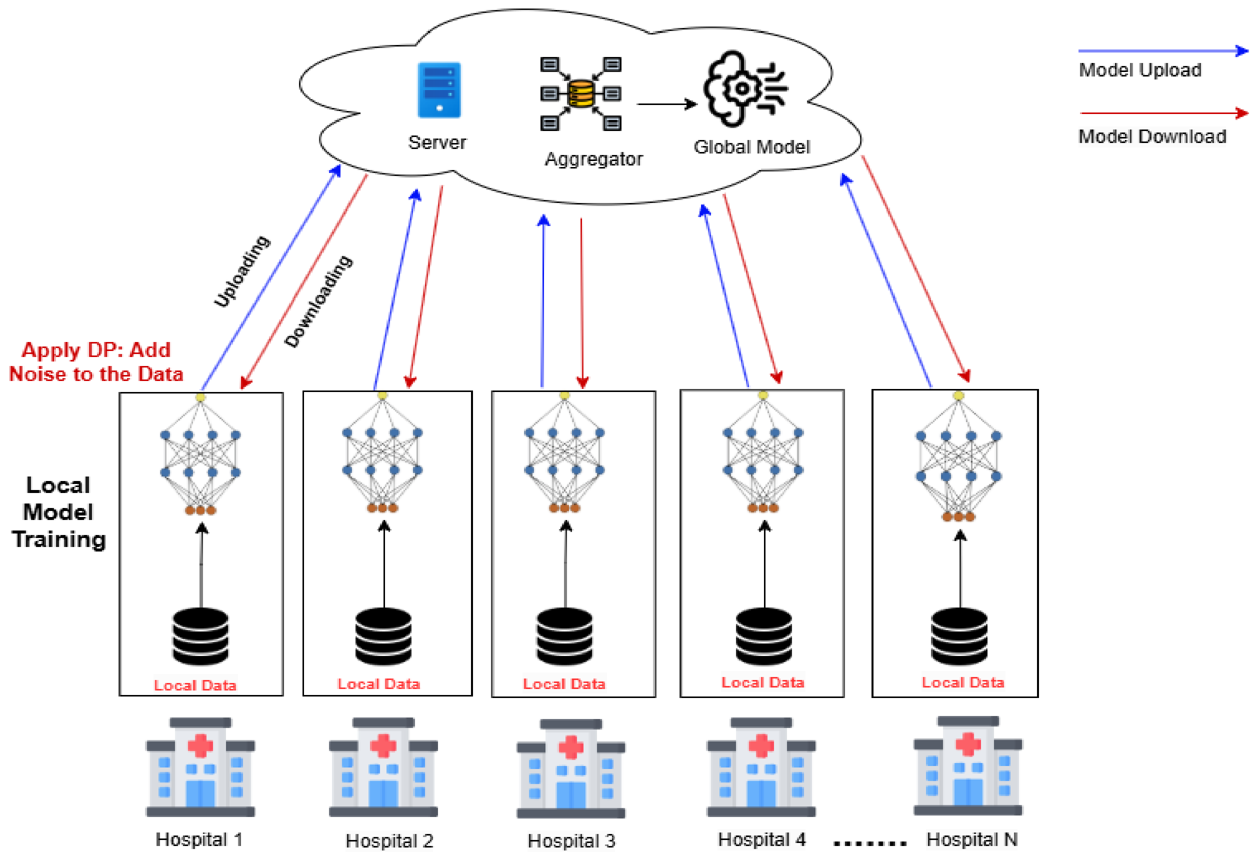


Figure 3.1 Privacy-Preserving Federated Learning (PPFL) Architecture

The overall architecture of the PPFL is illustrated in Figure 3.1, where each participating hospital locally trains a machine learning model using its private dataset. Instead of sharing raw patient data, hospitals compute model updates, which are then perturbed with noise using DP mechanism before being sent to a central aggregator for global model updating. This prevents an adversary from reconstructing individual patient records with high confidence, even if they gain access to transmitted updates.

The architecture consists of three interconnected layers, each playing a crucial role in preserving data security, model performance, and privacy preservation.

Each participating hospital maintains its local dataset and trains a machine learning model without transferring raw patient data. This decentralized approach minimizes privacy risks but remains vulnerable to inference attacks, where adversaries analyze model updates to reconstruct private data [72]. To mitigate this issue, Differential Privacy (DP) mechanism is applied at the local model training stage, so that any updates shared with the central server include controlled noise, thereby preventing the leakage of sensitive patient information [67].

Once local model training is complete, the differentially private model updates are transmitted to a central aggregator, which combines them to create a global model. The aggregation process follows the Federated Averaging (FedAvg) algorithm, which has been widely used in federated healthcare applications [69]. This algorithm facilitates the incorporation of contributions from all hospitals while maintaining privacy. However, existing research works have highlighted limitations in FL model aggregation, particularly in handling non-IID (non-independent and identically distributed) data distributions [35] [43]. Therefore, our approach optimizes the selection of privacy budgets (ε) based on the statistical properties of each hospital's dataset, instead of applying a uniform noise level across all institutions [69] [75].

One of the major challenges in Privacy-Preserving Federated Learning (PPFL) is balancing model utility and privacy. While smaller privacy budgets ($\varepsilon < 1$) offer stronger privacy protection, they significantly degrade model accuracy, making them impractical for high-performance healthcare applications [72]. Conversely, larger ε values improve accuracy but may expose sensitive information. To manage this trade-off, our model evaluates various noise multipliers and calculates their corresponding privacy budgets (ε) to establish an optimal balance between privacy protection and model accuracy [67].

Another critical aspect of PPFL architecture is managing communication overhead effectively. Frequent transmission of differentially private model updates can increase network load, particularly in resource-constrained healthcare environments [51]. Therefore, our approach addresses communication efficiency while maintaining model convergence, drawing on mechanisms from prior work in communication-efficient Federated Learning (FL) [69].

By integrating Differential Privacy (DP), federated model aggregation, and communication efficiency mechanisms, the proposed PPFL architecture enhances the feasibility of Privacy-Preserving FL (PPFL) in healthcare applications. This addresses key limitations in existing approaches, such as Secure Aggregation (SA) challenges in multi-institutional FL architecture [38] and privacy risks in model updates [72]. The next section delves deeper into the privacy-preserving mechanisms employed in our architecture, detailing how Differential Privacy (DP) provides strong privacy guarantees while maintaining model performance.

3.1 Local Model Training at Hospitals

In the proposed Privacy-Preserving Federated Learning (PPFL) Architecture, each participating hospital independently trains a local machine learning model using its private dataset. This decentralized approach eliminates the need for direct data sharing, keeping patient records within institutional boundaries. Instead of transmitting raw data to a central server, hospitals compute model updates based on their local data distribution and share these updates with the global model aggregator [72]. The primary objective of this phase is to perform local model training on hospital-specific datasets and generate model updates, which are later aggregated at the central server.

During local model training, each hospital uses its dataset to update the model parameters. The training process follows a standard iterative learning approach, where the model learns to minimize a predefined loss function. Each hospital trains a Feedforward Neural Network (FNN) on its private dataset. The training process involves loading and preprocessing the dataset, defining the model and performing multiple training iterations to refine model parameters. The hospital nodes do not communicate raw data but send their model updates to the aggregator [69].

To coordinate the Federated Learning (FL) process efficiently, the Flower (flwr) framework was used to manage communication between hospital nodes and the central aggregator. Flower provides a lightweight and scalable solution for Federated Learning (FL), enabling hospitals to train models locally and send only model updates to the server while preserving privacy.

To mitigate privacy concerns, Differential Privacy (DP) is integrated into the training process. Before transmitting model updates, hospitals add controlled noise to prevent the exposure of sensitive information. As a result, even if an adversary intercepts the updates, they cannot infer individual patient details [67]. DP helps protect privacy while still allowing effective model convergence, balancing privacy preservation with utility [72].

3.1.1 Data Partitioning and Preprocessing

The dataset used in the PPFL is partitioned among participating hospitals to simulate a real-world scenario where each institution retains control over its patient data. The partitioning follows a non-IID (non-independent and identically distributed) distribution, reflecting the natural variation in patient demographics, disease patterns, and treatment histories across different hospitals. This data heterogeneity presents a challenge in Federated Learning (FL), as it can impact model convergence and overall performance [69] [75].

To prepare the data for training, hospitals apply preprocessing procedures to achieve consistency and minimize biases. These steps include:

- Feature Normalization: Since medical data often contains features with different scales (e.g., age, lab results, vital signs), Min-Max normalization is applied to achieve uniformity.
- Handling Missing Data: Missing values are handled using mean or median imputation, leading to consistent feature distributions across hospitals.

The dataset is randomly partitioned among hospitals, with each institution receiving a distinct subset of the data. While the partitioning is random, it follows a non-IID structure, where each hospital receives slightly different class distributions. This approach reflects real-world scenarios where different institutions may have more or fewer instances of certain medical conditions based on natural variations in collected data.

Unlike traditional centralized machine learning, where all training data is uniformly distributed, Federated Learning (FL) must account for statistical heterogeneity across hospitals. The use of random non-IID partitioning results in:

- A dataset representing only a portion of the overall distribution is used at each hospital node;
- The global model must generalize across imbalanced class distributions when updates are aggregated;
- The FedAvg algorithm effectively combines knowledge from diverse data partitions without directly sharing patient records.

By applying random non-IID partitioning, the PPFL preserves data privacy while maintaining the real-world complexity of decentralized healthcare data.

Figure 3.2 illustrates the local model training process in our PPFL architecture. The process begins with data partitioning and preprocessing, followed by local model training at each hospital. Before sending updates to the central server, Differential Privacy (DP) mechanisms are applied to protect data security while preserving model utility.

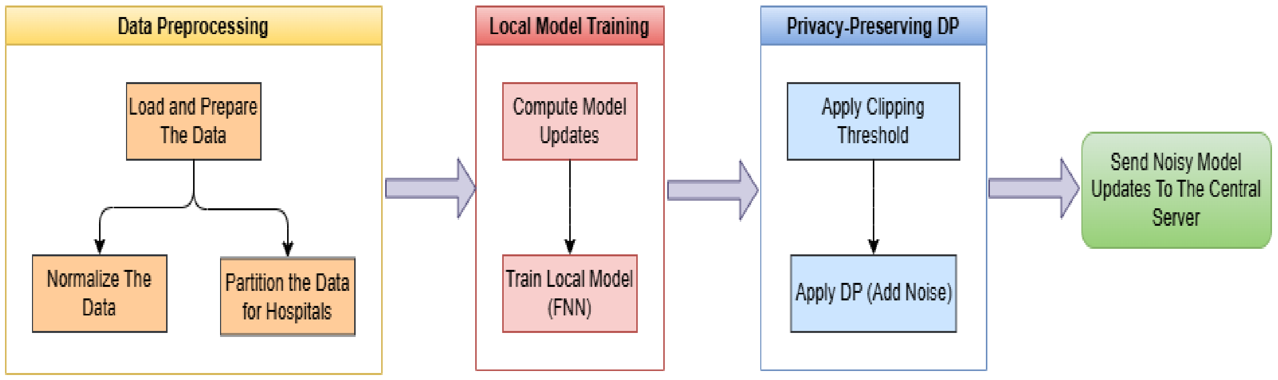


Figure 3.2 Local Model Training and Privacy-Preserving Process

As shown in Figure 3.2, data preprocessing involves normalizing hospital datasets and partitioning them to reflect real-world distributions. Once preprocessing is complete, models are trained independently at each hospital before applying Differential Privacy (DP) mechanisms to the model updates. The differentially private updates are then transmitted to the central server for aggregation.

3.1.2 Model Definition and Training

Each hospital's system independently trains a Feedforward Neural Network (FNN) on its local dataset. The model consists of an input layer, hidden layers, and an output layer, designed to effectively process structured medical data. This model structure is chosen for its ability to extract meaningful representations from diverse hospital datasets in a Federated Learning (FL) framework.

The selection of a Feedforward Neural Network (FNN) over alternative machine learning models.is influenced by the characteristics of healthcare datasets and the requirements of Federated Learning (FL). This section provides a detailed explanation of why FNN is the preferred choice in this context.

Suitability for Structured Medical Data: Healthcare datasets, such as Electronic Health Records (EHRs), laboratory test results, and structured clinical data, primarily consist of numerical and categorical variables. Unlike Convolutional Neural Networks (CNNs), which are specialized for image processing, or Long Short-Term Memory networks (LSTMs), which are optimized for sequential time-series data, FNNs are well-suited for handling structured tabular data. The ability of FNNs to model relationships between diverse medical features makes them a natural fit for diagnostic predictions, patient risk assessments, and clinical decision support systems. Since Federated Learning (FL) in healthcare often focuses on structured datasets rather than images or continuous time-series data, FNNs provide an effective and computationally efficient solution.

Lower Computational Overhead in Federated Learning (FL): Federated Learning (FL) requires models to be trained across multiple decentralized nodes (hospitals), often on resource-constrained devices. Unlike deep CNNs or RNNs, which demand significant computational power and memory, FNNs have a relatively simple structure with fewer parameters, making them lightweight and computationally efficient. This is particularly important in Federated Learning (FL), where hospitals frequently exchange model updates with the central aggregator. Using FNNs reduces communication costs and speeds up the federated training process, ensuring that institutions with varying hardware capabilities can participate without performance bottlenecks. Additionally, since hospitals may have different computing capacities, FNNs provide a more inclusive approach, allowing even smaller institutions with limited resources to contribute to the global model.

Comparison with Alternative Machine Learning Models: Alternative deep learning models, such as CNNs and RNNs (including LSTMs), may be effective for specific healthcare tasks but are not optimal for structured Federated Learning (FL) scenarios. CNNs, for example, are widely used in radiology and pathology, where medical imaging is the primary data source. However, CNNs require significantly more processing power and are not designed for handling numerical patient records or tabular data. On the other hand, RNNs and LSTMs are ideal for time-series applications, such as continuous patient monitoring through Electrocardiogram (ECG) readings or wearable health devices. These models rely on sequential dependencies, making them less practical for Federated Learning (FL) scenarios where patient records and clinical datasets are mostly static. In contrast, FNNs provide a general-purpose solution that is both computationally feasible and well-suited to the structured nature of federated healthcare data.

Robustness in Non-IID Data Distribution: One of the fundamental challenges in Federated Learning (FL) is data heterogeneity, where different hospitals have varying patient demographics, disease prevalence, and medical practices. This results in non-IID (non-independent and identically distributed) data, making it difficult for a global model to generalize effectively. FNNs demonstrate strong adaptability across diverse datasets because they do not rely on highly specific spatial or sequential relationships, unlike CNNs and RNNs. Their ability to generalize across multiple institutions without overfitting to a single hospital's dataset makes them particularly effective for Federated Learning (FL) in healthcare. Additionally, FNNs are less sensitive to minor differences in feature distributions, making them more resilient to the data inconsistencies commonly found in multi-institutional collaborations.

In FNN model training, each hospital follows a structured process;

- Loads its dataset and initializes the local model training;
- Computes model predictions based on input features;
- Evaluates prediction errors using a loss function suitable for the classification task;
- Updates model parameters using an optimization algorithm to minimize the loss.

The training process follows an iterative learning approach, where the model undergoes multiple local updates before sharing its parameters with the central server. Instead of transmitting raw patient data, each hospital only shares its locally computed model updates. These updates are then aggregated at the server to refine the global model.

To prevent overfitting, a regularization technique is applied during training. Additionally, an activation function is used in hidden layers to introduce non-linearity, enabling the model to capture complex relationships in the medical dataset.

This decentralized training approach guarantees that:

- Hospitals retain control over their data while contributing to the global model;
- Local model training uses patient records without exposing sensitive information;
- The global model benefits from diverse institutional knowledge without requiring direct data sharing.

Once local model training is complete, the updated model parameters are prepared for aggregation at the central server, as described in the next section.

3.1.3 Differential Privacy in Local Model Training

To protect sensitive patient data, Differential Privacy (DP) is incorporated into the local model training phase before transmitting model updates to the central server. The DP mechanism introduces controlled noise to the model updates, reducing the risk of individual data points being inferred from shared updates. This mechanism strengthens privacy guarantees while allowing meaningful learning [67] [72].

Before applying noise, the clipping threshold limits the impact of individual updates. This step prevents any single hospital's update from disproportionately influencing the global model. The clipping threshold restricts extreme values in the model updates, maintaining stability in the training process.

Once the model updates are restricted by the clipping threshold, noise is added based on a predefined noise multiplier. The level of noise directly affects the privacy budget (ε), where higher noise levels strengthen privacy but may reduce model accuracy. The choice of the noise multiplier is critical in balancing privacy protection and model utility.

A challenge in applying DP is selecting an appropriate privacy budget. A lower ε value offers stronger privacy but can reduce model performance, while a higher ε improves accuracy but weakens privacy protection. Multiple noise multipliers are evaluated to determine an optimal balance between privacy and model effectiveness [69].

By integrating DP at the local model training stage, hospitals can participate in Federated Learning (FL) without exposing sensitive patient information. The added noise protects individual contributions while still allowing the global model to learn from decentralized data sources [75].

To evaluate the impact of privacy noise on model performance, different noise multipliers were tested. The selection of noise levels was informed by prior research, which suggests that higher noise improves privacy but may degrade model accuracy. By testing multiple levels, we aimed to identify an optimal balance between privacy protection and utility. The specific effects of different noise multipliers on model performance are analyzed in Chapter 4.

3.2 Privacy-Preserving Mechanism Using Differential Privacy

Differential Privacy (DP) is incorporated into the proposed Privacy-Preserving Federated Learning (PPFL) architecture to safeguard sensitive patient data during training while allowing hospitals to collaboratively improve model performance. Since FL enables model training across multiple institutions without sharing raw data, there remains a risk that adversaries could infer sensitive information by analyzing model updates. DP mitigates this risk by introducing controlled noise into the model updates before they are transmitted to the central aggregator. This section explores the role of DP in securing Federated Learning (FL), the methodology for applying it, and the challenges involved in balancing privacy protection with model performance.

3.2.1 Role of Differential Privacy in Federated Learning (FL)

In Federated Learning (FL), each participating hospital performs local model training using its private dataset and shares only model updates rather than raw data. However, these updates may still carry patterns that could potentially be exploited to extract sensitive information. DP prevents this by ensuring that the inclusion or exclusion of any patient's data does not significantly alter the output, making it difficult for an attacker to infer details about individual records [67]. By injecting noise into model updates before they are uploaded, DP obscures the contribution of individual data points, thereby reducing privacy risks while still allowing meaningful learning.

The key element in DP is the privacy budget (ε), which quantifies the trade-off between privacy protection and model accuracy. Smaller values of ε provide stronger privacy guarantees but introduce more noise, potentially degrading the model's performance. Conversely, larger ε values enhance accuracy but weaken privacy. Selecting an appropriate ε is essential in federated healthcare applications, where both privacy and predictive reliability are critical. [69] [75].

By applying DP, the PPFL architecture prevents direct exposure of sensitive patient data, allowing hospitals to participate in Federated Learning (FL) without compromising confidentiality.

3.2.2 Implementation of DP in the PPFL

The proposed architecture applies Local Differential Privacy (LDP) at each hospital before transmitting model updates to the central aggregator. This process guarantees that sensitive patient information remains protected throughout the Federated Learning (FL) workflow. The implementation of DP follows these key steps:

First, each hospital trains a local model on its private dataset, generating model updates after multiple training iterations. Instead of sharing raw data, hospitals compute model updates that reflect their local model training progress. Before these updates are transmitted, a Differential Privacy (DP) mechanism is applied to obscure individual contributions. This is achieved by introducing noise to the model updates. The amount of noise added depends on the chosen privacy budget (ε), where smaller values of ε enhance privacy at the cost of reduced model accuracy. To systematically assess this trade-off, different noise multipliers were tested, and their respective privacy budgets were calculated to select an optimal configuration for Federated Learning (FL) in healthcare [75].

Once the differentially private updates are generated, they are transmitted to the central aggregator, which performs global model aggregation using the Federated Averaging (FedAvg) algorithm [69]. The aggregator combines the received updates from multiple hospitals to create an improved global model while maintaining that no raw patient data is exposed. Finally, the updated global model is distributed back to participating hospitals, allowing them to benefit from collective learning in a privacy-preserving mechanism.

By implementing DP at the local level, the proposed architecture protects patient data privacy while enabling institutions to collaborate on AI-driven healthcare solutions. However, the addition of noise poses challenges in maintaining model accuracy, particularly in healthcare applications where high precision is crucial for clinical decision-making [72].

3.2.3 Challenges and Trade-offs in Applying DP

While Differential Privacy (DP) strengthens security in Federated Learning (FL), integrating it into the workflow introduces several challenges that require careful management.

One primary concern is the degradation of model accuracy due to noise injection. Smaller datasets, which are common in federated healthcare applications, can suffer more from added noise, making it difficult to maintain reliable model performance [72]. The challenge lies in identifying an appropriate privacy budget (ε) that balances strong privacy protection with acceptable accuracy levels. Models trained with very low ε values may provide strong privacy guarantees but at the cost of reduced predictive performance. On the other hand, larger ε values can enhance model accuracy but increase the risk of data leakage.

Another significant challenge involves both computational overhead and communication costs. The integration of Differential Privacy (DP) mechanisms increases the complexity of local model training, which can strain the computational resources of smaller healthcare institutions, leading to slower training times and making DP-based Federated Learning (FL) less practical [51]. Additionally, Federated Learning (FL) already requires frequent exchanges of model updates, and the addition of noise and privacy mechanisms further increases data transmission, potentially causing network congestion and delays in global model updates, especially in bandwidth-limited environments. To address these challenges, efficient DP implementations, hardware optimization strategies, and techniques such as reducing update frequency or compressing differentially private model updates can help balance computational efficiency with privacy protection [69].

Finally, non-IID data distributions remain a fundamental challenge in Federated Learning (FL). Since hospital datasets often vary in patient demographics, disease prevalence, and medical imaging standards, DP's noise addition can have disproportionate effects on different institutions. This variability complicates the training of an accurate global model across heterogeneous datasets [35].

To address these challenges, the proposed architecture optimizes privacy budget selection and reduces update frequency to balance privacy protection and model performance. The evaluation phase will compare two models: one trained without DP and another with DP at different noise levels. This comparison will provide insights into how privacy mechanisms impact accuracy, training efficiency, and privacy budgets (ε) in Federated Learning (FL).

The next section explores the federated model aggregation process, detailing how differentially private updates are incorporated to maintain security and performance in Federated Learning (FL).

3.3 Federated Model Aggregation and Global Model Distribution

The federated model aggregation phase plays a crucial role in integrating locally trained updates from multiple hospitals while maintaining privacy and ensuring that no raw patient data is shared. In the Privacy-Preserving Federated Learning (PPFL) architecture, each hospital conducts local model training independently and sends differentially private model updates to a central server. These updates are then aggregated using a secure and privacy-preserving algorithm, such as Federated Averaging (FedAvg), to refine the global model without exposing sensitive patient information. The aggregation process helps improve the overall model's generalization by combining knowledge from diverse healthcare institutions, addressing the challenges posed by non-IID data distributions. This section provides an in-depth explanation of how model updates are securely integrated, the role of privacy mechanisms in aggregation, and the subsequent distribution of the refined global model back to hospitals for continuous training and improvement.

Figure 3.3 illustrates the federated model aggregation process in the PPFL architecture. Each hospital trains its local model independently using private patient data and generates model updates. These updates are then sent to a central aggregator, where they are combined to create a refined global model. The aggregated model is subsequently distributed back to hospitals, enabling them to benefit from collective learning without sharing raw data.

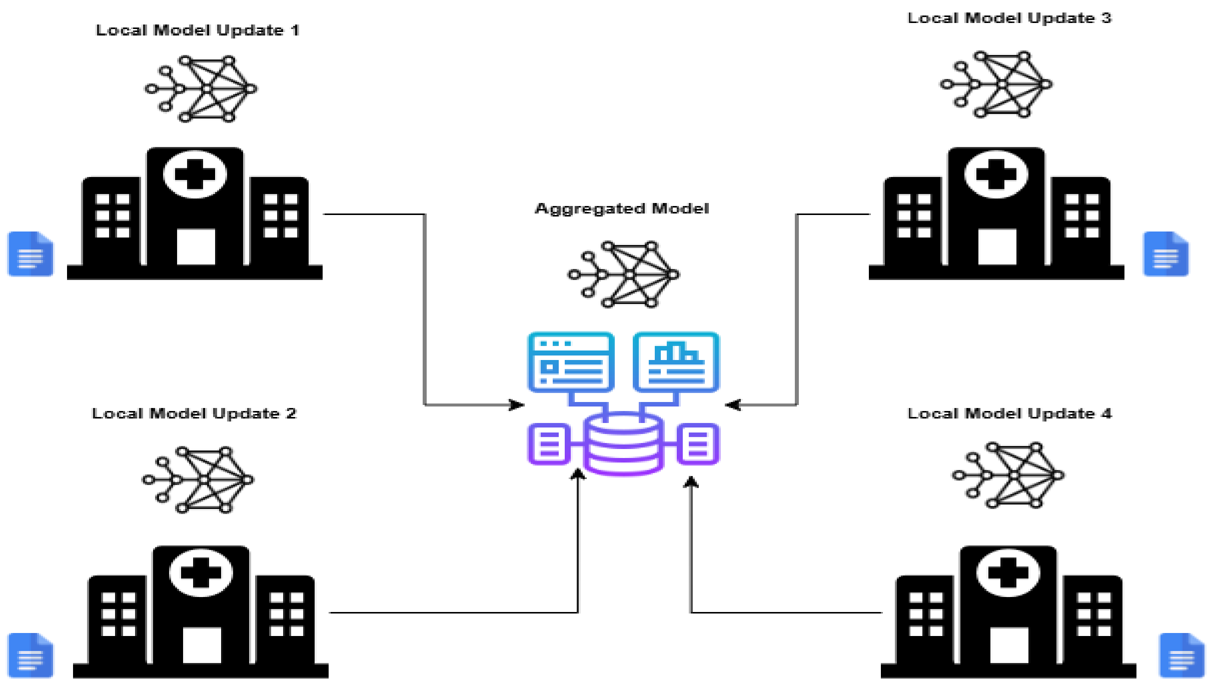


Figure 3.3 Aggregation of Local Model Training Updates

As depicted in Figure 3.3, Federated Learning (FL) enables multiple hospitals to collaboratively improve a shared global model while maintaining data privacy. Each institution contributes to the global model by submitting locally trained updates instead of sharing raw patient data. The central aggregator combines these updates using the Federated Averaging (FedAvg) algorithm, producing a global model that captures knowledge from all participating hospitals. This approach ensures that valuable insights are leveraged across institutions while preserving patient confidentiality and addressing privacy concerns in healthcare AI applications.

3.3.1 Federated Model Aggregation Process

After local model training rounds, each hospital transmits its differentially private model updates to the central aggregator instead of sharing raw patient data. The aggregation process follows the Federated Averaging (FedAvg) algorithm [69], which is widely used in Federated Learning (FL) due to its efficiency in handling multiple model updates and combines model updates from various institutions to refine the global model. The Federated Learning (FL) setup was implemented using the Flower (flwr) framework, which facilitates communication between the federated server and hospital nodes. Flower allows seamless orchestration of the training process by handling model updates, aggregations, and secure communication while maintaining scalability.

The FedAvg algorithm aggregates model updates by computing a weighted average of the received updates. The weights are determined based on the number of data records in each hospital's dataset. This approach prevents bias in global model training, where hospitals with smaller datasets might otherwise have an equal influence as those with larger datasets. The aggregation process follows these key steps:

1. Receives model updates from all participating hospitals. Each update consists of differentially private weight adjustments derived from local datasets;
2. Computes a weighted average of all received updates. Since hospitals may have different dataset sizes, their contributions to the global model are weighted accordingly to prevent bias from smaller datasets;
3. Applies the aggregated updates to the global model, refining its parameters based on collective learning across institutions.

One of the primary challenges in federated aggregation is handling non-IID data distributions across hospitals. Since each institution collects patient data under different conditions, local model Training updates may diverge significantly, affecting global model convergence. To address this, the FedAvg algorithm aggregates model updates by averaging them across hospitals, helping to smooth out variations caused by heterogeneous data distributions. While FedAvg does not explicitly correct non-IID imbalances, increasing the number of training rounds can improve convergence and model stability across diverse hospital datasets [69][75].

3.3.2 Global Model Distribution

Once the aggregation phase is complete, the refined global model is distributed back to each participating hospital. The updated model serves as a new starting point for subsequent training rounds, enabling hospitals to continue improving their local model training while benefiting from collective knowledge.

A key consideration during global model distribution is ensuring that the aggregated model performs well across diverse hospital datasets. Since Federated Learning (FL) operates in a decentralized setup, each hospital conducts local model training on private patient data before contributing updates to the global model. This approach helps mitigate performance disparities that arise due to variations in patient demographics, medical imaging standards, and disease prevalence across institutions [35].

Figure 3.4 presents the federated model aggregation process in the PPFL architecture. After receiving differentially private model updates from hospitals, the central server aggregates them using the Federated Averaging (FedAvg) algorithm. This step combines the contributions of all participating hospitals into a single global model, which is then distributed back to facilitate continuous collaborative learning.

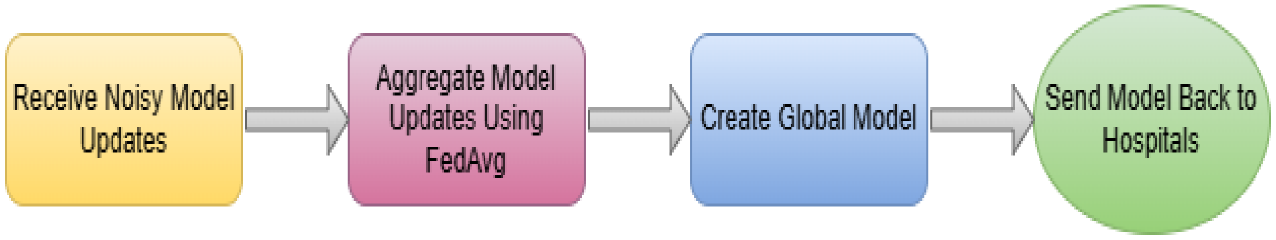

Receive Noisy Model Updates
Aggregate Model Updates Using FedAvg
Create Global Model
Send Model Back to Hospitals

Figure 3.4 Federated Model Aggregation and Global Model Distribution

As depicted in Figure 3.4, the aggregation process plays a critical role in Federated Learning (FL) by integrating updates from multiple decentralized models into a unified global model. This allows hospitals to benefit from collective learning without exposing sensitive data. The refined global model is then shared with hospitals, enabling them to build upon prior knowledge while maintaining strong privacy guarantees.

3.4 Summary

In this chapter, we introduced the Privacy-Preserving Federated Learning (PPFL) architecture, designed to facilitate secure and privacy-aware collaborative learning across multiple hospitals. The proposed architecture integrates Federated Learning (FL) with Differential Privacy (DP) to protect sensitive patient data while enabling institutions to develop a robust global machine learning model.

The architecture of the PPFL was described in detail, outlining its three primary components: local model training at hospitals, privacy-preserving mechanisms using Differential Privacy (DP), and federated model aggregation with global model distribution. The methodology enables hospitals to contribute to a shared learning process without transferring raw patient data, mitigating privacy risks and regulatory concerns.

The local model training process was presented as a decentralized approach where each hospital trains its machine learning model using institution-specific patient data. Given the presence of non-IID data distributions, preprocessing techniques such as feature normalization and missing value imputation were applied to improve data consistency. Additionally, a Feedforward Neural Network (FNN) model was selected for its suitability in handling structured medical data in Federated Learning (FL).

A key focus of this chapter was the implementation of Differential Privacy (DP) within the PPFL architecture. DP was applied at the local model training phase to prevent the exposure of sensitive patient information through model updates To determine an optimal privacy-utility trade-off, different noise multipliers were tested. The corresponding privacy budgets (ε) were computed, allowing for an informed selection of noise levels that preserved model accuracy while maintaining strong privacy guarantees.

The federated model aggregation process was then described, with updates from participating hospitals being aggregated using the Federated Averaging (FedAvg) algorithm.

While DP enhances privacy, trade-offs between privacy and model performance remain a critical consideration. The addition of noise can degrade accuracy, particularly in healthcare applications where precise predictions are essential. To overcome this challenge, privacy budget selection was fine-tuned to balance accuracy preservation with strong privacy guarantees.

Having established the methodology and proposed architecture in Chapter 3, the next chapter focuses on the implementation and evaluation of the PPFL framework.

CHAPTER 4 IMPLIMENTATION AND RESULTS

This chapter presents the implementation and evaluation of the Privacy-Preserving Federated Learning (PPFL) architecture proposed in Chapter 3 (cf. Chap. 3, §3.1). Firstly, the implementation of the Privacy-Preserving Federated Learning (PPFL) are presented, including data processing and partitioning, local model training, integration of the Differential Privacy (DP) mechanism and aggregation of local model training updates at the central server. Secondly, a key focus is on integrating Differential Privacy (DP) into Federated Learning (FL) to strengthen privacy protection while maintaining local model training performance. Thirdly, the evaluation results of the proposed architecture are presented based on accuracy, loss, privacy budget (ε), and training time. Fourthly, a comparative analysis is conducted to assess the impact of different noise multipliers (σ) on utility and privacy protection. Finally, key findings are summarized, providing insights into the privacy-utility trade-off and demonstrating the potential of integrating the Federated Learning (FL) framework with Differential Privacy (DP) in real-world healthcare scenarios.

4.1 Implementation of the Privacy-Preserving Federated Learning

This section details the implementation of the Privacy-Preserving Federated Learning (PPFL) architecture, focusing on the local model training process at hospital nodes, Differential Privacy (DP) integration and aggregation of local model training updates at the central server. The implementation of the Privacy-Preserving Federated Learning (PPFL) architecture was conducted using Python, a versatile programming language widely used for machine learning and Federated Learning (FL) research. Python was chosen due to its extensive library support for data processing, machine learning, and privacy-preserving mechanisms.

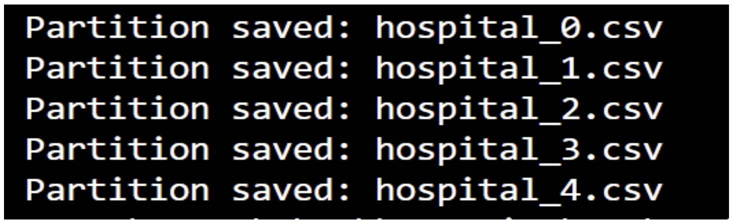
4.1.1 Data Processing and Partitioning

In this dissertation, the Oxford Parkinson's Disease Detection Dataset is used, a widely recognized biomedical dataset containing 4,680 structured health-related data points for Parkinson's disease detection. Its structured format enables effective simulation of Federated Learning (FL) scenarios, where data remains distributed across multiple institutions.

Data preprocessing is conducted using Pandas and NumPy for dataset manipulation and structuring. These libraries facilitate loading, cleaning, and organizing the dataset to ensure compatibility with the FL. Additionally, Scikit-learn’s MinMaxScaler is applied to normalize feature values between 0 and 1, maintaining a consistent scale for all input features and improving local model training convergence and stability (Appendix A, pp. 88–90).

For data partitionimg feature selection is performed to remove non-relevant attributes, including dropping the 'name' column, which does not contribute to classification and may introduce bias. To ensure data consistency, missing values are replaced with the mean of their respective feature, preventing incomplete records from affecting local model training. The dataset is then structured by separating input features (X) from the target label (Y), where X contains biomedical attributes, and Y represents a binary classification indicating whether a patient has Parkinson’s disease (1) or not (0).

Within the Federated Learning (FL) framework, the dataset is divided into five partitions, each assigned to a different hospital using a non-IID partitioning strategy. This reflects real-world healthcare environments, where patient data is inherently imbalanced across institutions. Some hospitals may have a higher proportion of Parkinson’s patients, while others may include a more diverse mix of early-stage cases and healthy individuals. These variations arise due to differences in demographics, specialization, or geographic location. Each partition remains private to its respective hospital, providing data confidentiality while enabling collaborative learning (Appendix B, pp. 90-91). Once partitioning is complete, each partition is saved as a separate CSV file, ensuring that each hospital has access to its local dataset. These files are named sequentially as "hospital_0.csv" to "hospital_4.csv", corresponding to the five simulated hospital nodes. The successful saving of the partitioned datasets is confirmed in Figure 4.1.



```
Partition saved: hospital_0.csv
Partition saved: hospital_1.csv
Partition saved: hospital_2.csv
Partition saved: hospital_3.csv
Partition saved: hospital_4.csv
```

Figure 4.1 Dataset Partitioning

The summary of dataset partitioning is provided in Table 4.1.

Table 4.1 Dataset Partitioning Summary

Hospital	No. of Records	Data Distribution
Hospital 1	39 Records	Non-IID
Hospital 2	39 Records	Non-IID
Hospital 3	39 Records	Non-IID
Hospital 4	39 Records	Non-IID
Hospital 5	39 Records	Non-IID

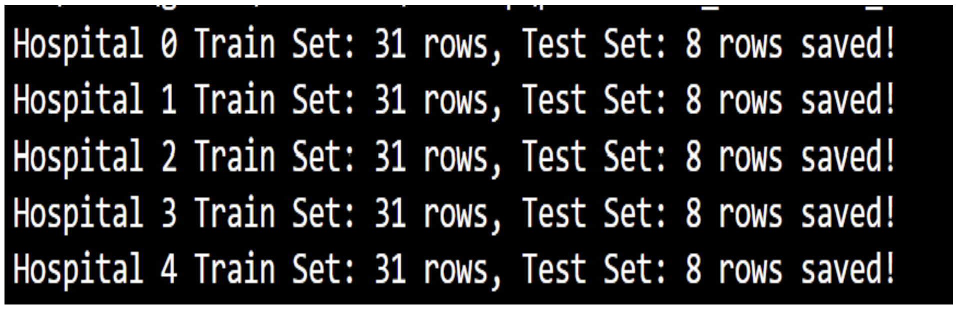
In real-world healthcare applications, patient records are not evenly distributed across hospitals. Some hospitals may have more patients with advanced Parkinson’s, while others may have more cases of early-stage diagnoses or healthy controls. A non-IID distribution helps train the Federated Learning (FL) in a way that reflects real-world heterogeneity across medical institutions. This partitioning supports data decentralization, allowing hospitals to perform local model training on private data before contributing to the global model.

4.1.2 Local Model Training

A Feedforward Neural Network (FNN) is used for local model training at each hospital node. The input layer receives numerical features from the preprocessed Parkinson’s dataset. Hidden layers consist of fully connected dense layers with ReLU activation to capture complex patterns. The output layer includes a single neuron with a sigmoid activation function, producing a probability score for binary classification. The local model is trained using the binary cross-entropy loss function, suitable for classification tasks. During local model training, each hospital updates its model weights by computing binary cross-entropy loss and optimizing parameters using Adam Optimizer. Before transmitting updates, a clipping threshold is applied, followed by noise addition to ensure privacy protection. Each hospital evaluates its locally trained model using its local test set, computing accuracy and loss metrics before contributing to global aggregation via FedAvg.

Each hospital splits its local dataset into training and testing sets, where 80% of the data is used for training (x_train, y_train), while 20% is reserved for testing (x_test, y_test). This enables the evaluation of local model training on unseen data at each hospital before updates are sent to the FL server (Appendix C, pp. 93–96).

Figure 4.2 illustrates the dataset partitioning used in the Federated Learning (FL).



```
Hospital 0 Train Set: 31 rows, Test Set: 8 rows saved!
Hospital 1 Train Set: 31 rows, Test Set: 8 rows saved!
Hospital 2 Train Set: 31 rows, Test Set: 8 rows saved!
Hospital 3 Train Set: 31 rows, Test Set: 8 rows saved!
Hospital 4 Train Set: 31 rows, Test Set: 8 rows saved!
```

Figure 4.2 Train-Test Split for Each Hospital Node

Each hospital performs local model training using its partitioned dataset, optimizing the neural network parameters based on local data patterns. The local model training process includes multiple local epochs, where this model iteratively updates its weights using the hospital's dataset before communicating with the central server. After completing the training phase, each hospital sends only its model updates to the central server. The server aggregates the updates received from all hospitals using the Federated Averaging (FedAvg) algorithm (cf. Chap. 3, §3.3.1). Once the aggregation is complete, the updated global model is distributed back to all hospital nodes. Each hospital then continues local model training using the updated model parameters, repeating the process for multiple communication rounds until the model converges (Appendix D, pp. 97-98).

4.1.3 Integration of Differential Privacy Mechanism

The Differential Privacy (DP) mechanism is implemented at the hospital level before transmission. Each hospital applies a clipping threshold to limit the influence of individual updates on the locally trained model. After clipping, noise is added to the updates of the trained local model, making it statistically difficult to infer any single patient's contribution. The noise level is determined by the noise multiplier (σ), which influences the privacy budget (ε). A smaller ε provides stronger privacy by introducing more noise, whereas a larger ε allows for better performance with reduced privacy guarantees (Appendix E, pp. 99-102). The differentially private updates are then sent to the central server for aggregation using FedAvg, ensuring that the global model remains privacy-preserving while capturing collective knowledge from all hospitals. The global model is redistributed, and the next federated training round begins.

To evaluate the impact of Differential Privacy (DP), multiple noise multipliers (σ) were tested, including 0.01, 0.05, 0.1, 0.5, and 1.0, with corresponding privacy budgets (ε) calculated (Appendix G, pp. 106-107). The final noise multiplier was selected based on empirical performance, balancing privacy protection and model accuracy.

4.1.4 Aggregation of Local Model Training Updates at the Central Server

Once local model training is completed, hospitals send their model updates to the central server for aggregation using the Federated Averaging (FedAvg) algorithm (cf. Chap. 3, §3.3.1). FedAvg combines updates from multiple hospitals without requiring data sharing, ensuring privacy preservation. The aggregation process begins when the central server receives local model training updates from all participating hospitals. Since different hospitals have varying amounts of data, FedAvg applies a weighted averaging approach, allowing hospitals with larger datasets to contribute more significantly (Appendix F, pp. 103-105).

Once the aggregation is performed, the updated global model is generated and redistributed to all hospital nodes. Each hospital receives the new global model, which serves as the starting point for the next round of local model training. This iterative process continues across multiple communication rounds until the model stabilizes and performance converges.

4.2 Evaluation and Results

This section presents the evaluation of the Privacy-Preserving Federated Learning (PPFL) architecture, analyzing model performance under different configurations. The evaluation considers two scenarios: local model training without Differential Privacy (DP) and local model training with DP, enabling a comparative analysis of the impact of noise injection on performance.

The following subsections provide a detailed analysis of the experimental results, starting with model performance without Differential Privacy (DP), followed by an assessment of DP's impact on accuracy, an evaluation of the privacy-utility trade-off across different noise multipliers, and concluding with an analysis of the privacy budget (ε) impact on model performance.

4.2.1 Model Performance Without Differential Privacy

This subsection presents the baseline performance of the Privacy-Preserving Federated Learning (PPFL) architecture without applying Differential Privacy (DP). The evaluation focuses on accuracy and loss to observe the learning behavior of the locally trained model at individual hospital nodes. Each hospital performs local model training on its dataset and evaluates performance at the end of each training round before transmitting updated model weights to the central server.

Figure 4.3 shows the accuracy and loss trends over 100 training rounds, illustrating the learning progression during local model training (Appendix H, pp. 108-109).

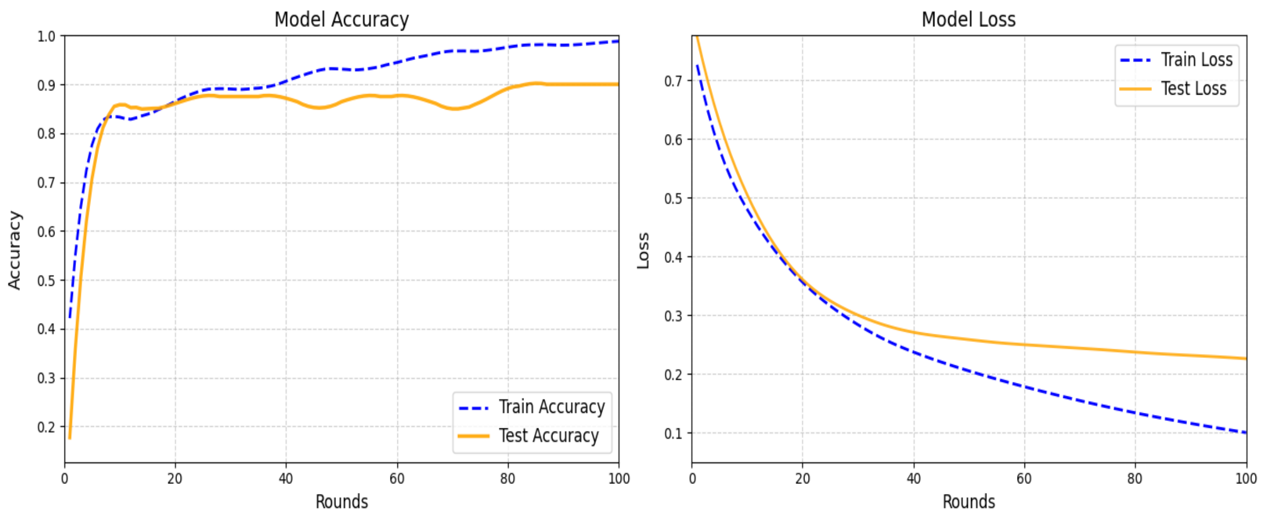


Figure 4.3 Model Accuracy and Loss Over Federated Training Rounds

Figure 4.3 illustrates the local model's accuracy and loss trends over 100 federated training rounds. The left plot shows the progression of training and test accuracy, where both curves improve and stabilize as training advances. The right plot presents the training and test loss, demonstrating a consistent decrease and convergence, indicating effective local model learning and generalization. The results are recorded for each training round and visualized through accuracy and loss plots, showing trends over the training process.

The next section examines how Differential Privacy (DP) affects accuracy when different noise multipliers are applied to the local model training updates.

4.2.2 Impact of Differential Privacy on Model Performance

This subsection analyzes the impact of integrating Differential Privacy (DP) into the local model training update process on overall model performance. To assess this impact, hospitals applied different noise multipliers (σ) during local model training before transmitting model updates to the central server. The results are compared to the baseline local model training without DP to observe how varying noise levels influence learning outcomes. Additionally, the final accuracy after 100 training rounds was recorded for each noise multiplier (σ) to evaluate the effect of DP on model performance.

Figure 4.4 provides a comparison of accuracy across different noise multipliers (Appendix L, pp. 115-116).

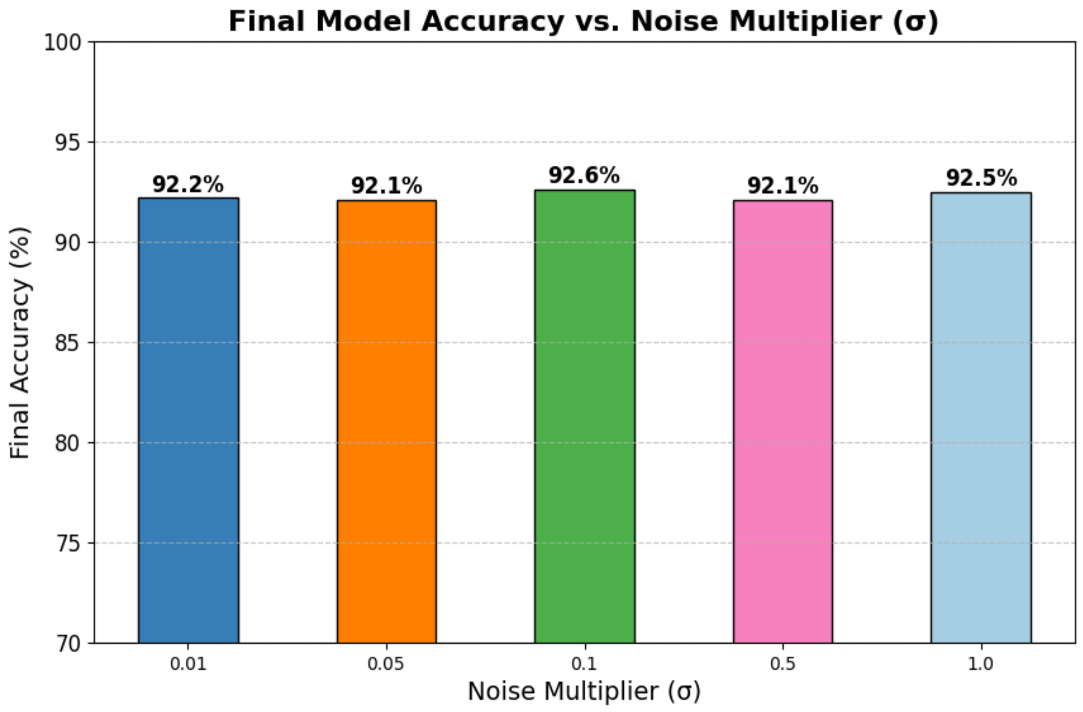


Figure 4.4 Final Model Accuracy vs. Noise Multiplier (σ)

Figure 4.4 presents the final accuracy values for different noise multipliers (σ). The accuracy remains relatively stable, ranging between 92.1% and 92.6% across various noise levels. Notably, the locally trained model with σ = 0.1 achieved the highest accuracy (92.6%), while those trained with σ = 0.05 and 0.5 had slightly lower accuracy (92.1%). These findings suggest that moderate noise levels provide a reasonable balance between privacy and model performance.

The next section further examines the privacy-utility trade-off, illustrating how different noise levels impact both accuracy and privacy budgets (ε).

4.2.3 Privacy-Utility Trade-Off Analysis

The privacy-utility trade-off in Federated Learning (FL) represents the balance between privacy protection and model performance. Differential Privacy (DP) enhances privacy by adding noise to the local model training updates; however, excessive noise can reduce model accuracy and increase loss, ultimately diminishing learning effectiveness.

A fundamental principle governing this trade-off is captured in the No-Free-Lunch theorem, which states that stronger privacy inevitably comes at the cost of reduced model utility [77]. Our results align with this principle, demonstrating that as the noise multiplier (σ) increases, the accuracy slightly decreases, particularly at σ = 1.0. However, privacy protection is significantly enhanced under such conditions, making it a crucial factor for applications requiring strong confidentiality guarantees. While prior research works have explored moderate noise levels such as σ = 0.1 for balancing privacy and utility, our work extends this analysis by further evaluating higher noise levels. Figure 4.5 illustrates these trends, showing how accuracy and loss evolve over training rounds under different noise multipliers (σ) (Appendix I, pp. 110-111).

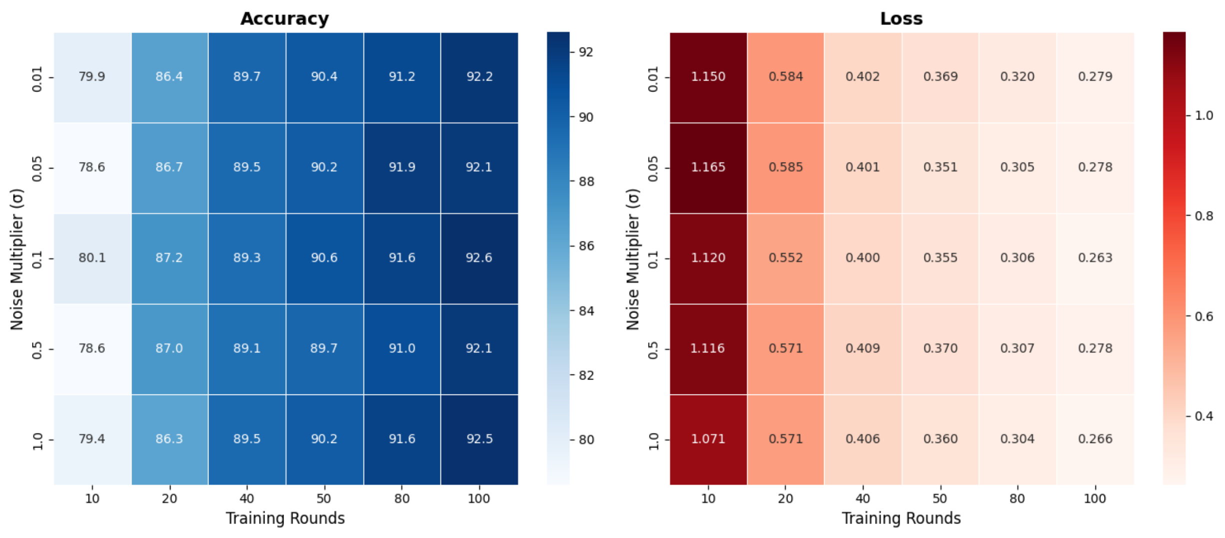


Figure 4.5 Accuracy and Loss Heatmaps Across Training Rounds

Figure 4.5 presents two heatmaps:

- Left: Accuracy across training rounds for different noise multipliers (σ).
- Right: Loss values across training rounds for different noise multipliers (σ).

The accuracy heatmap shows that higher noise multipliers (σ) slightly delay accuracy convergence, although local model trained with σ = 0.1 and 0.01 achieve the highest accuracy in the later rounds. In contrast, the loss heatmap shows that all configurations start with higher loss values in the early training rounds, with no consistent pattern of higher noise causing significantly higher loss. As training progresses, loss values gradually decrease and stabilize across all noise levels. Although early rounds exhibit slight fluctuations in convergence due to added noise, the accuracy drop remains within an acceptable range. Overall, this model stabilizes in later rounds, demonstrating that Differential Privacy (DP) can be effectively integrated into Federated Learning (FL) without causing significant accuracy trade-offs.

4.2.4 Evaluation of Privacy Budget Impact

The relationship between privacy budgets (ε) and model accuracy is a key consideration in Differential Privacy (DP). A smaller privacy budget (ε) provides stronger privacy guarantees but often reduces the accuracy of local model training due to increased noise. Conversely, a larger privacy budget (ε) allows better accuracy but may weaken privacy protection. To analyze this trade-off, privacy budgets were computed for different noise multipliers (σ), and their corresponding final accuracy values were recorded.

A key challenge in Privacy-Preserving Federated Learning (PPFL) is determining the optimal privacy budget (ε) that balances privacy and accuracy. Our results indicate that a higher noise multiplier (σ = 1.0) leads to a lower privacy budget (ε = 4.8), providing stronger privacy protection but slightly reducing accuracy. Previous research on privacy, model utility, and communication trade-offs in Local Differential Privacy (LDP) has demonstrated similar findings, emphasizing the need for selecting an appropriate privacy budget based on application requirements [76]. Our research work extends this analysis to Federated Learning (FL), showing that σ = 1.0 provides the strongest privacy guarantee while maintaining acceptable accuracy (92.5%).

Figure 4.6 provides an overview of how accuracy changes with varying privacy budgets (Appendix J, p. 112).

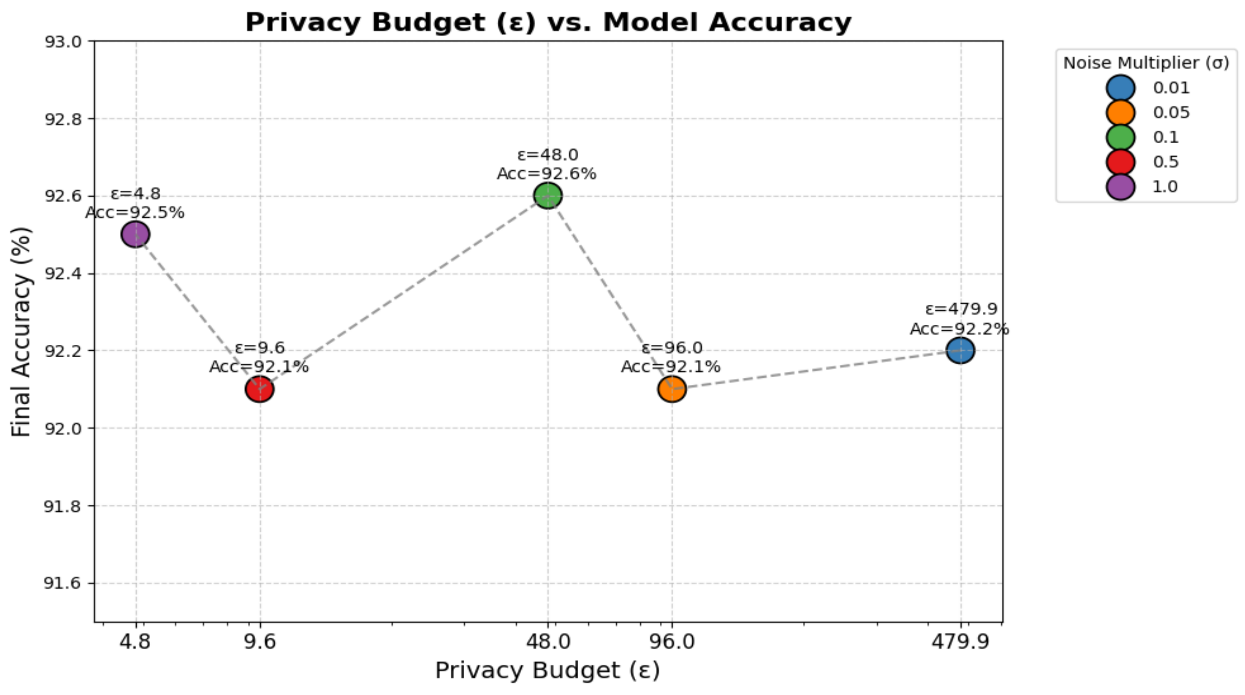


Figure 4.6 Privacy Budget (ε) vs. Model Accuracy

Figure 4.6 shows that accuracy remains relatively stable across different privacy budgets, fluctuating between around 92.1% and 92.6%. As expected, higher noise multipliers (σ = 1.0) correspond to lower privacy budgets (ε = 4.8), whereas lower noise multipliers (σ = 0.01) result in a significantly larger privacy budget (ε = 479.9).

These findings indicate that while strong privacy (small ε values) slightly reduce accuracy, moderate noise levels (σ = 0.1) maintain a balance between privacy and model utility. The next section explores the impact of Differential Privacy (DP) on total training time.

The impact of privacy budgets (ε) on training time is a significant factor when applying Differential Privacy (DP) in Federated Learning (FL). Higher noise multipliers (σ) increase privacy protection but also introduce additional computational overhead, affecting the total training duration. To analyze this, total training time was recorded for different privacy budgets (ε) and noise multipliers (σ). The relationship between these factors is illustrated in Figure 4.7, which presents a 3D bar chart depicting total training time across different privacy budgets and noise multipliers (Appendix K, pp. 113-114).

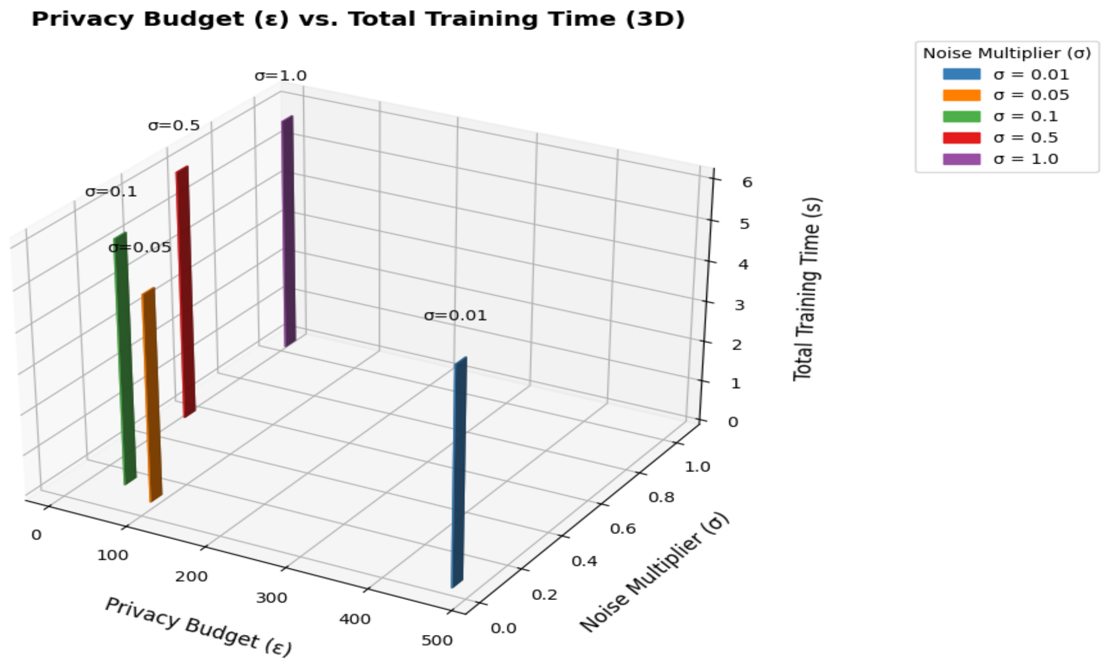


Figure 4.7 Privacy Budget (ε) vs. Total Training Time

Figure 4.7 demonstrates that training time does not increase strictly with higher noise multipliers (σ). While it is expected that larger noise multipliers require additional computations due to the clipping threshold, the observed results indicate a more nuanced trend. The training time fluctuates slightly across different noise multipliers, with σ = 0.1 taking the longest (6.00s), while σ = 1.0 achieves a slightly shorter training time (5.76s). Lower noise multipliers (σ = 0.05 and 0.01) show minimal differences in training duration, suggesting that noise injection may sometimes stabilize learning rather than significantly prolong it.

These results demonstrate a key trade-off:

- Smaller privacy budgets (stronger privacy) do not always correlate with increased training time.
- Larger privacy budgets (weaker privacy) allow for more predictable training durations.

Thus, achieving an optimal privacy-utility balance requires considering both accuracy and computational efficiency, particularly in real-world healthcare applications where resource constraints may impact Federated Learning (FL) feasibility.

To further analyze the privacy-utility trade-off, the final accuracy, privacy budget (ε), and total training time were recorded for each noise multiplier (σ). Table 4.2 presents a comparison of these factors, illustrating how different noise multipliers influence both accuracy and computational cost.

Table 4.2 Impact of Noise Multipliers (σ) on Accuracy, Privacy Budget (ε), and Training Time (s)

Noise(σ)	Accuracy (%)	Privacy Budget (ε)	Training Time (s)
1.0	92.50	4.80	5.76
0.5	92.10	9.60	5.77
0.1	92.58	48.00	6.00
0.05	92.11	96.00	5.07
0.01	92.23	479.90	5.31

As observed in Table 4.2, accuracy remains relatively stable across different noise multipliers, with a slight decrease at higher noise multipliers. The local model trained with σ = 0.1 achieves the highest accuracy (92.58%), while those trained with σ = 0.05 and 0.5 achieve slightly lower accuracy values. In contrast, the privacy budget (ε) varies significantly, with stronger privacy guarantees (lower ε) observed at higher noise multipliers. Training time also fluctuates slightly but remains within a reasonable range. These findings emphasize the importance of carefully selecting the noise multiplier to maintain a balance between privacy, accuracy, and computational efficiency.

4.3 Comparative Analysis

This section provides a detailed comparison of local model trained within the Federated Learning (FL) framework, both with and without Differential Privacy (DP), evaluating their accuracy, training efficiency, and privacy protection levels. The comparative analysis is based on the results obtained from different noise multipliers applied during training. The key aspects analyzed include accuracy progression, training time variations, and the overall impact of DP on privacy and performance.

The comparison demonstrates the impact of adding noise to local model updates, showcasing the trade-offs between privacy preservation and model utility. While the introduction of DP enhances privacy by obscuring individual data contributions, it also introduces minor changes in accuracy and training time. The results allow for a quantitative assessment of privacy-aware Federated Learning (FL) and guide the selection of an optimal noise multiplier that achieves both privacy protection and strong predictive performance.

Based on the previous table (Table 4.2), the noise multipliers σ = 1.0 and σ = 0.1 were selected for further comparison, as they provided the most balanced trade-off between accuracy, privacy, and training efficiency. The locally trained model with σ = 0.1 achieved the highest accuracy (92.58%) while maintaining a moderate privacy budget (ε = 48.0), making it a strong candidate for applications requiring both high utility and privacy. Meanwhile, σ = 1.0 provided the strongest privacy protection (ε = 4.80) while maintaining acceptable accuracy (92.50%) and the fastest training time (5.76s), making it ideal for scenarios prioritizing privacy preservation. These two configurations offer different advantages, making them the most suitable candidates for comparison in the following sections.

The following subsections discuss the observed trends in accuracy across training rounds, the effect of noise on training efficiency, and the final performance metrics of the locally trained model.

4.3.1 Accuracy Progression Across Training Rounds

Table 4.3 presents the accuracy across different training rounds for local model trained with and without the Differential Privacy (DP) mechanism, comparing the baseline model to those with DP applied using noise multipliers of σ = 1.0 and σ = 0.1. The accuracy values illustrate how the local trained model learns over time and demonstrate the impact of Differential Privacy (DP) on the learning process.

Table 4.3 Accuracy Across Training Rounds

Round	Accuracy Model 1 Without DP (%)	Accuracy Model 2 With DP (σ=0.1) (%)	Accuracy Model 2 With DP (σ=1.0) (%)
10	83.87	80.11	79.44
20	89.03	87.23	86.29
40	90.96	89.27	89.45
50	94.83	90.57	90.17
80	97.41	91.60	91.61
100	98.70	92.58	92.50

As expected, the local trained model without DP achieves the highest accuracy, reaching 98.70% after 100 rounds. However, the differentially private models show a slight reduction in accuracy due to the noise added for privacy protection. The model with σ = 0.1 achieves 92.58% accuracy, while the model with σ = 1.0 achieves 92.50%. These results indicate that DP introduces a slight trade-off in learning performance but still allows for meaningful model improvements over time.

Despite the accuracy gap, the local model with DP maintains a stable learning trajectory, with accuracy gradually increasing as training progresses. The differences in accuracy among the local trained model remains relatively small, demonstrating that privacy-preserving training can still achieve competitive performance. These findings reinforce that while DP limits the ability of attackers to extract sensitive data, it does not significantly hinder model utility, particularly when an appropriate noise multiplier is chosen.

The next section analyzes the impact of Differential Privacy (DP) on training time.

4.3.2 Training Time Comparison with and without Differential Privacy

Table 4.4 compares the training time per round for local model trained without Differential Privacy (DP) and with DP at noise multipliers σ = 0.1 and σ = 1.0. Training time is a crucial factor when implementing DP, as privacy mechanisms can introduce computational overhead.

Table 4.4 Training Time Across Training Rounds

Round	Training Time Model 1 Without DP (S)	Training Time Model 2 With DP (σ=0.1) (S)	Training Time Model 2 With DP (σ=1.0) (S)
10	0.068	0.072	0.062
20	0.065	0.065	0.047
40	0.067	0.052	0.061
50	0.066	0.079	0.078
80	0.073	0.065	0.061
100	0.076	0.047	0.047

The training time remains comparable across local trained model, with minimal additional overhead introduced by Differential Privacy (DP). Interestingly, the differentially private models (σ = 0.1 and σ = 1.0) exhibit slightly lower training time in some rounds. This could be attributed to the noise influencing convergence behavior, reducing sensitivity to small updates and leading to faster stabilization. During the early training rounds, the non-private model and the DP models exhibit similar training times, but as training progresses, the DP models complete some rounds slightly faster. This suggests that while DP introduces mathematical transformations such as noise addition and applying a clipping threshold, these operations do not significantly slow down Federated Learning (FL) framework when optimized properly. The results indicate that integrating DP does not introduce a significant computational burden, making it feasible for real-world Privacy-Preserving Federated Learning (PPFL) architectures. The next section provides a comprehensive comparison of the final accuracy, training time, and privacy budget for local trained model with and without DP.

4.3.3 Overall Model Performance Summary

Table 4.5 provides a comparative summary of the final accuracy, total training time, noise multipliers (σ = 0.1 and σ = 1.0), and privacy budget (ε) for local trained model with and without Differential Privacy (DP), highlighting its impact on performance, computational efficiency, and privacy protection.

Table 4.5 Summary of Final Model Performance Metrics

Model	Final Accuracy (%)	Total Training Time (s)	Noise Multiplier (σ)	Privacy Budget (ε)
Model 1 (Without DP)	98.70	7.82	N/A	N/A
Model 2 With DP (σ = 0.1)	92.58	6.00	0.1	48.00
Model 2 With DP (σ =1.0)	92.50	5.76	1.0	4.80

This table provides a high-level comparison of local trained model with and without DP, illustrating the trade-offs between privacy, accuracy, and computational efficiency. The results indicate that while applying DP slightly reduces accuracy, it significantly enhances privacy protection. The local trained model without DP achieves the highest accuracy (98.70%) but lacks any privacy guarantees. In contrast, the local trained model with DP maintains competitive accuracy, with Model 2 (σ = 0.1) achieving the highest DP accuracy (92.58%) and Model 2 (σ = 1.0) offering the strongest privacy protection (ε = 4.80) while maintaining 92.50% accuracy. Training time varies slightly between the models, with the differentially private models demonstrating shorter training times. Notably, Model 2 (σ = 1.0) achieves the fastest total training time (5.76 seconds), suggesting that higher noise levels may accelerate convergence by reducing the sensitivity of updates, leading to more stable learning.

Based on these findings, Model 2 (σ = 1.0) is the most suitable configuration, as it provides the best balance between privacy (ε = 4.80), training time (5.76s), and accuracy (92.50%). This setup ensures strong privacy protection while maintaining efficient learning, making it the recommended choice for Privacy-Preserving Federated Learning (PPFL) architectures.

The results confirm the privacy-utility trade-off in the Federated Learning (FL), demonstrated by the local trained model with Differential Privacy (DP). While DP introduces minor accuracy reductions, it significantly enhances data privacy. The choice of noise multiplier directly influences this balance:

- Model 2 ($\sigma = 0.1$) strikes a balance between accuracy (92.58%) and privacy ($\varepsilon = 48.00$), making it a reasonable choice for applications needing both privacy and utility.
- Model 2 ($\sigma = 1.0$) provides the strongest privacy ($\varepsilon = 4.80$) while maintaining acceptable accuracy (92.50%) and the best training efficiency. Given the importance of privacy in healthcare applications, this configuration offers a promising balance between security and model performance.

With this, the section on comparative analysis is complete. The final section will summarize key findings and discuss future optimizations for Privacy-Preserving Federated Learning (PPFL).

4.4 Summary of Findings

This chapter detailed the implementation and evaluation of the Privacy-Preserving Federated Learning (PPFL) architecture, focusing on its effectiveness in balancing privacy, accuracy, and computational efficiency. The experimental results reveal the impact of DP on accuracy, training efficiency, and privacy protection, illustrating the trade-offs between data confidentiality and predictive performance.

Our results confirm that integrating DP into FL is computationally feasible and does not introduce significant overhead, even at higher noise levels. Moreover, our research supports previous findings on DP-FedAvg [69], demonstrating that Differential Privacy (DP) can provide strong privacy protection while maintaining a good level of model accuracy. A key aspect of DP-FL is determining an appropriate privacy budget (ε), as improper tuning can either compromise privacy or unnecessarily degrade model performance. Our findings align with previous research [72, 75, 76, 77], reinforcing the importance of carefully optimizing DP parameters to achieve a practical balance between privacy and utility in FL-based healthcare systems. Our research work demonstrates that appropriate privacy budget choices ensure strong patient data protection while preserving learning performance, making DP-FL a scalable and secure solution for collaborative hospital networks.

In our experiments, local trained model without DP achieved the highest accuracy, reaching 98.70% after 100 training rounds. However, this model provided no privacy protection, making it unsuitable for sensitive healthcare applications. Introducing DP resulted in a slight accuracy decrease, with local trained model using noise multipliers of 0.1 and 1.0 achieving 92.58% and 92.50% accuracy, respectively. Despite this reduction, the trade-off was acceptable given the substantial privacy benefits.

Surprisingly, the impact of DP on training time was minimal, with the local trained model with DP exhibiting slightly shorter training durations compared to the non-private model. This suggests that adding noise does not significantly increase computational costs and may, in some cases, stabilize the learning process.

A crucial finding was the relationship between noise multipliers and privacy budgets. A noise multiplier of 1.0 provided the strongest privacy guarantee with a privacy budget of $\varepsilon = 4.80$, offering robust protection against potential data reconstruction attacks. In contrast, a lower noise multiplier of 0.1 resulted in a weaker privacy budget ($\varepsilon = 48.00$), indicating reduced privacy preservation. Despite the trade-off in accuracy, the model with $\sigma = 1.0$ demonstrated strong privacy protection, acceptable accuracy, and the shortest training time, making it the preferred configuration for Privacy-Preserving Federated Learning (PPFL).

The findings confirm that Privacy-Preserving Federated Learning (PPFL) architecture is a viable solution for privacy-sensitive environments such as healthcare. Hospitals can collaboratively train machine learning models without sharing raw patient data. The optimal DP configuration must balance privacy, accuracy, and computational efficiency, with $\sigma = 1.0$ emerging as the best choice in our research work. While DP introduces a minor reduction in model performance, its privacy benefits outweigh the drawbacks, demonstrating that Privacy-Preserving Federated Learning (PPFL) architecture can be effectively deployed in real-world healthcare applications.

The final chapter will provide a conclusion to this research, discuss its contributions, and outline future directions for enhancing Privacy-Preserving Federated Learning (PPFL) in healthcare applications.

CHAPTER 5 CONCLUSION

This dissertation presented a Privacy-Preserving Federated Learning (PPFL) architecture designed to enhance security and privacy in hospital data sharing while supporting collaboration among hospitals. This research work explored the integration of Differential Privacy (DP) into Federated Learning (FL) to mitigate privacy risks associated with shared local model training updates Additionally, the impact of DP on model utility and computational efficiency was examined, analyzing the trade-offs between privacy preservation and performance.

This chapter provides a comprehensive conclusion to the dissertation. The first section summarizes the research contributions, emphasizing the key findings from the proposed PPFL architecture and its evaluation. The second section discusses the limitations of this research, addressing challenges encountered during implementation and areas that require further exploration. Finally, the third section outlines future research directions, suggesting potential improvements and extensions to Privacy-Preserving Federated Learning (PPFL) in healthcare applications.

5.1 Summary of works

Privacy and security concerns in hospital data sharing have become increasingly significant as healthcare institutions embrace digital transformation. The need for collaborative healthcare research has led to the development of frameworks that enable multiple institutions to train machine learning models on patient data. However, traditional centralized machine learning approaches require aggregating raw data in a central repository, exposing sensitive patient records to potential security breaches and privacy violations. This limitation makes centralized approaches unsuitable for privacy-sensitive applications, particularly in healthcare, where patient confidentiality is protected by strict regulations.

Federated Learning (FL) framework has emerged as a promising solution, allowing decentralized model training across multiple institutions while keeping patient data within hospital boundaries. By keeping raw data at its local source, FL mitigates many privacy concerns associated with centralized machine learning. However, despite its decentralized nature, FL alone does not guarantee full privacy protection, as shared local model updates may still reveal sensitive information. Various attacks, such as gradient leakage and membership inference attacks, have demonstrated that adversaries can reconstruct private data from model updates, posing a significant risk in privacy-sensitive domains [68]. To address these vulnerabilities, Differential Privacy (DP) has been proposed as a privacy-preserving mechanism that introduces controlled noise into local trained model updates to prevent adversarial inference. However, integrating DP into FL introduces a fundamental challenge: balancing privacy preservation with model utility. Higher noise levels provide stronger privacy guarantees but may lead to significant accuracy degradation, impacting the overall effectiveness of the learning process [69].

This dissertation sought to address these challenges by developing a Privacy-Preserving Federated Learning (PPFL) architecture that enhances privacy protection in hospital data sharing while maintaining efficient model training. The motivation for this research stemmed from growing concerns regarding patient data confidentiality and the limitations of existing FL frameworks in healthcare. Furthermore, applying DP to FL often results in accuracy degradation, making it crucial to analyze the privacy-utility trade-off and determine an optimal balance [72]. Given these challenges, this dissertation aimed to explore how privacy preservation in hospital data sharing can be improved to support effective and confidential collaborative healthcare research.

To achieve this, the research focused on two key aspects: proposing a Privacy-Preserving Federated Learning (PPFL) architecture that integrates Differential Privacy (DP) while maintaining utility and evaluating its performance under different noise multipliers (σ). Since FL allows hospitals to retain their data while collaboratively training machine learning models, it is essential to implement a mechanism that prevents information leakage through shared local trained model updates. This motivated the design of a Privacy-Preserving Federated Learning (PPFL) architecture that incorporates Differential Privacy (DP) mechanisms at the hospital level before transmitting local trained model updates to the central server. By introducing differentially private noise at the local level, this approach prevents attackers from reconstructing sensitive patient information while still allowing meaningful collaborative learning.

To accomplish these objectives, the dissertation was structured around three key phases. First, a Privacy-Preserving Federated Learning (PPFL) architecture was designed by integrating Differential Privacy (DP) at the local hospital level to enhance privacy protection before local trained model updates were shared for global aggregation. By applying a clipping threshold to the local model updates before adding noise, this approach aimed to mitigate potential privacy risks while maintaining model utility. Second, the implemented Feedforward Neural Network (FNN) model was deployed in a simulated hospital network, where data was partitioned into five non-IID subsets representing five hospitals to reflect the heterogeneous nature of real-world healthcare data. Third, the effectiveness of the proposed architecture was evaluated by conducting extensive experiments. Local trained model with and without DP were compared across multiple training rounds, and different noise multipliers (σ) were tested to analyze their impact on model accuracy, privacy budget (ε), and training efficiency. The findings provided insights into the privacy-utility trade-off in Privacy-Preserving Federated Learning (PPFL) for healthcare applications.

The implementation was carried out using the Flower Federated Learning (FL) framework, where each hospital node trained a Feedforward Neural Network (FNN) on its local dataset. The FedAvg algorithm was applied at the central server to aggregate local trained model updates while preserving privacy. To analyze the impact of Differential Privacy (DP) on model performance, different noise multipliers (σ) ranging from 0.01 to 1.0 were tested, with corresponding privacy budgets (ε) calculated for each setting. Supporting prior research on DP-FedAvg [69] and DP-SGD [68], our results demonstrate that a noise multiplier of $\sigma = 1.0$ provides strong privacy protection with an acceptable trade-off in accuracy. Moreover, the results confirm that privacy budgets (ε) must be carefully chosen to balance privacy preservation and model performance, consistent with prior research works [72, 75, 76, 77].

These results align with previous research, which suggests that moderate noise levels can effectively enhance privacy without significantly compromising accuracy. However, this research work extends previous work by evaluating higher noise levels in a Federated Learning (FL), demonstrating that privacy can be strengthened with minimal accuracy loss when an optimal noise multiplier is carefully chosen [72].

By addressing the privacy-utility trade-off, this dissertation contributes to the growing field of Privacy-Preserving Federated Learning (PPFL) for healthcare applications. The proposed PPFL architecture provides a practical approach for hospitals to collaborate on training a global model without exposing sensitive patient data, making it a viable solution for real-world hospital networks. The research work also provides a detailed empirical evaluation of DP's impact on FL performance, offering insights that can guide future implementations in privacy-sensitive domains.

In summary, this dissertation demonstrates that Privacy-Preserving Federated Learning (PPFL) is a feasible and effective architecture for secure hospital data sharing. By integrating Differential Privacy (DP) with Federated Learning (FL), the research work validates that patient confidentiality can be preserved while maintaining accuracy and efficiency. The findings confirm that FL alone is insufficient to provide strong privacy protection and that DP mechanisms must be carefully optimized to achieve the best balance between privacy preservation and model utility. These contributions establish a strong foundation for future research in privacy-aware federated healthcare learning, paving the way for improved security, efficiency, and scalability in collaborative medical AI applications.

5.2 Limitations

This research demonstrates the feasibility of Privacy-Preserving Federated Learning (PPFL) in healthcare, but several limitations must be acknowledged.

One key challenge is the privacy-utility trade-off, as higher noise levels improve privacy but slightly reduce accuracy. Our results showed that local trained model with Differential Privacy achieved lower accuracy (92.50%) compared to the non-private model (98.70%), underscoring the need for careful parameter tuning to balance privacy and utility. Although training time remained stable in our experiments, real-world hospital deployments may require higher computational resources due to DP’s noise injection, applying a clipping threshold to local model updates, and secure aggregation overhead. Further testing in real hospital environments is needed to assess practical performance at scale. Another limitation is that our PPFL architecture was evaluated in a simulated setup, meaning real-world factors such as network latencies, hospital-to-hospital communication, and data inconsistencies were not considered. Implementing the architecture in a real hospital network would help assess practical scalability and deployment challenges.

The security evaluation in this research focused on privacy risks; however, the architecture was not tested against advanced adversarial attacks, such as model inversion, membership inference, or data poisoning. Future work should explore integrating Privacy-Preserving Federated Learning (PPFL) with additional security techniques to enhance overall robustness.

5.3 Future Work

Future research can explore several enhancements to improve the Privacy-Preserving Federated Learning (PPFL) architecture and address the challenges identified in this dissertation. One possible improvement is implementing adaptive Differential Privacy (DP) mechanisms, where the noise multiplier (σ) is dynamically adjusted based on training progress and data sensitivity. Instead of using a fixed noise level, an adaptive approach could optimize the privacy-utility trade-off by applying minimal noise when privacy risks are lower and increasing noise when necessary.

Another important direction is the real-world deployment of PPFL in actual hospital networks to evaluate the impact of network latencies, communication overhead, and hospital-to-hospital coordination on privacy and model performance. While this research was conducted in a simulated environment, deploying it in real healthcare applications would provide deeper insights into its practical feasibility, scalability, and security challenges.

Further improvements can also be made by combining Differential Privacy (DP) with other security mechanisms to enhance model protection. While DP effectively limits the risk of data exposure, it does not defend against all types of attacks. Future research works could integrate secure aggregation mechanisms such as Secure Multi-Party Computation (SMPC), Homomorphic Encryption (HE), or Trusted Execution Environments (TEE) to further protect model updates while maintaining accuracy.

Additionally, assessing the resilience of PPFL against adversarial threats is a crucial area for future exploration. While this research primarily focused on privacy risks, future work should evaluate PPFL's robustness against sophisticated attacks, such as model inversion, membership inference, and data poisoning. Strengthening defenses against these threats would further enhance its applicability in privacy-sensitive environments like healthcare.

REFERENCES

[1] M. H. Brendan, E. Moore, D. Ramage, S. Hampson, and Arcas, Blaise Agüera y, “Communication-Efficient Learning of Deep Networks from Decentralized Data,” *arXiv.org*, 2016. https://arxiv.org/abs/1602.05629

[2] K. Bonawitz *et al.*, “Practical Secure Aggregation for Privacy-Preserving Machine Learning,” *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, Oct. 2017, doi: https://doi.org/10.1145/3133956.3133982.

[3] C. Dwork and A. Roth, “The Algorithmic Foundations of Differential Privacy,” *Foundations and Trends® in Theoretical Computer Science*, vol. 9, no. 3–4, pp. 211–407, 2014, doi: https://doi.org/10.1561/0400000042.

[4] S. Zapechnikov, “Secure multi-party computations for privacy-preserving machine learning,” *Procedia Computer Science*, vol. 213, pp. 523–527, 2022, doi: https://doi.org/10.1016/j.procs.2022.11.100.

[5] B. Raju Cherukuri, “Federated Learning: Privacy-Preserving Machine Learning in Cloud Environments,” *International Journal of Science and Research (IJSR)*, vol. 13, no. 10, pp. 1539–1549, Oct. 2024, doi: https://doi.org/10.21275/ms241022095645

[6] J. Liu, Y. Li, M. Zhao, L. Liu, and N. Kumar, “EPFFL: Enhancing Privacy and Fairness in Federated Learning for Distributed e-Healthcare Data Sharing Services,” *IEEE Transactions on Dependable and Secure Computing*, pp. 1–14, 2024, doi: https://doi.org/10.1109/tdsc.2024.3431542

[7] A. El Ouadrhiri and A. Abdelhadi, “Differential Privacy for Deep and Federated Learning: A Survey,” *IEEE Access*, pp. 1–1, 2022, doi: https://doi.org/10.1109/access.2022.3151670.

[8] B. Zhang, Y. Mao, Z. Tu, X. He, P. Ping, and J. Wu, “Optimizing Privacy-Accuracy Trade-off in DP-FL via Significant Gradient Perturbation,” *2021 17th International Conference on Mobility, Sensing and Networking (MSN)*, pp. 423–430, Dec. 2023, doi: https://doi.org/10.1109/msn60784.2023.00068.

[9] K. Wei *et al.*, “Federated Learning With Differential Privacy: Algorithms and Performance Analysis,” *IEEE Transactions on Information Forensics and Security*, vol. 15, pp. 3454–3469, 2020, doi: https://doi.org/10.1109/TIFS.2020.2988575.

[10] R. C. Geyer, T. Klein, and M. Nabi, “Differentially Private Federated Learning: A Client Level Perspective,” *arXiv:1712.07557 [cs, stat]*, Mar. 2018, Available: https://arxiv.org/abs/1712.07557

[11] A. Chouhan and B. R. Purushothama, “A Survey on Secure Aggregation for Privacy-Preserving Federated Learning,” *Communications in Computer and Information Science*, pp. 13–26, 2024, doi: https://doi.org/10.1007/978-3-031-59100-6_2.

[12] X. Li, G. Wu, L. Yao, and S. Geng, “Hybrid aggregation for Federated Learning (FL) under blockchain framework,” *Computer Communications*, vol. 225, pp. 311–323, Jul. 2024, doi: https://doi.org/10.1016/j.comcom.2024.06.009.

[13] J. Li *et al.*, “A Federated Learning Based Privacy-Preserving Smart Healthcare System,” *IEEE Transactions on Industrial Informatics*, vol. 18, no. 3, pp. 2021–2031, Mar. 2022, doi: https://doi.org/10.1109/TII.2021.3098010.

[14] M. Joshi, A. Pal, and M. Sankarasubbu, “Federated Learning for Healthcare Domain - Pipeline, Applications and Challenges,” *ACM Transactions on Computing for Healthcare*, May 2022, doi: https://doi.org/10.1145/3533708.

[15] M. Al-Rubaie and J. M. Chang, “Privacy-Preserving Machine Learning: Threats and Solutions,” *IEEE Security & Privacy*, vol. 17, no. 2, pp. 49–58, Mar. 2019, doi: https://doi.org/10.1109/msec.2018.2888775.

[16] M. Akgün, A. O. Bayrak, B. Ozer, and M. Ş. Sağıroğlu, “Privacy preserving processing of genomic data: A survey,” *Journal of Biomedical Informatics*, vol. 56, pp. 103–111, Aug. 2015, doi: https://doi.org/10.1016/j.jbi.2015.05.022.

[17] O. Choudhury, Y. Park, Theodoros Salonidis, Aris Gkoulalas-Divanis, I. Sylla, and A. K. Das, “Predicting Adverse Drug Reactions on Distributed Health Data using Federated Learning.,” *PubMed*, vol. 2019, pp. 313–322, Jan. 2019.

[18] Q. Wu, X. Chen, Z. Zhou, and J. Zhang, “FedHome: Cloud-Edge based Personalized Federated Learning for In-Home Health Monitoring,” *IEEE Transactions on Mobile Computing*, pp. 1–1, 2020, doi: https://doi.org/10.1109/tmc.2020.3045266.

[19] M. Abdul Salam, S. Taha, and M. Ramadan, “COVID-19 detection using federated machine learning,” *PLOS ONE*, vol. 16, no. 6, p. e0252573, Jun. 2021, doi: https://doi.org/10.1371/journal.pone.0252573.

[20] Z. Li, V. Sharma, and S. P. Mohanty, “Preserving Data Privacy via Federated Learning: Challenges and Solutions,” IEEE Consumer Electronics Magazine, vol. 9, no. 3, pp. 8–16, May 2020, doi: https://doi.org/10.1109/mce.2019.2959108.

[21] B. Jeon, S. M. Ferdous, Muntasir Raihan Rahman, and Anwar Walid, “Privacy-Preserving Decentralized Aggregation for Federated Learning,” *arXiv (Cornell University)*, May 2021, doi: https://doi.org/10.1109/infocomwkshps51825.2021.9484437.

[22] M. AbaOud, M. A. Almuqrin, and Mohammad Faisal Khan, “Advancing Federated Learning through Novel Mechanism for Privacy Preservation in Healthcare Applications,” *IEEE Access*, pp. 1–1, Jan. 2023, doi: https://doi.org/10.1109/access.2023.3301162.

[23] S. Chenthara, K. Ahmed, H. Wang, and F. Whittaker, “Security and Privacy-Preserving Challenges of e-Health Solutions in Cloud Computing,” *IEEE Access*, vol. 7, pp. 74361–74382, 2019, doi: https://doi.org/10.1109/access.2019.2919982.

[24] M. Nankya, A. Mugisa, Y. Usman, A. Upadhyay, and R. Chataut, “Security and Privacy in E-Health Systems: A Review of AI and Machine Learning Techniques,” *IEEE Access*, vol. 12, pp. 148796–148816, 2024, doi: https://doi.org/10.1109/access.2024.3469215.

[25] X. Zhang, J. Ding, M. Wu, Stephen, H. V. Nguyen, and M. Pan, “Adaptive Privacy Preserving Deep Learning Algorithms for Medical Data,” pp. 1168–1177, Jan. 2021, doi: https://doi.org/10.1109/wacv48630.2021.00121.

[26] S. Mishra, Ritu Tondon, and N. Pal, “Revolutionizing Healthcare with Federated Learning: A Comprehensive Review,” pp. 1–5, Sep. 2024, doi: https://doi.org/10.1109/acroset62108.2024.10743932.

[27] D. Ganesh and O.B.V. Ramanaiah, “Edge Federated Learning for Smart HealthCare Systems: Applications and Challenges,” pp. 1727–1735, Oct. 2024, doi: https://doi.org/10.1109/icses63445.2024.10763213.

[28] S. Moon and Won Hee Lee, “Privacy-Preserving Federated Learning in Healthcare,” Feb. 2023, doi: https://doi.org/10.1109/iceic57457.2023.10049966.

[29] M. Ali, F. Naeem, M. Tariq, and G. Kaddoum, “Federated Learning for Privacy Preservation in Smart Healthcare Systems: A Comprehensive Survey,” *IEEE Journal of Biomedical and Health Informatics*, pp. 1–14, 2022, doi: https://doi.org/10.1109/jbhi.2022.3181823.

[30] X. Yin, Y. Zhu, and J. Hu, “A Comprehensive Survey of Privacy-Preserving Federated Learning,” *ACM Computing Surveys*, vol. 54, no. 6, pp. 1–36, Jul. 2021, doi: https://doi.org/10.1145/3460427.

[31] M. J. Sheller *et al.*, “Federated Learning in medicine: facilitating multi-institutional collaborations without sharing patient data,” *Scientific Reports*, vol. 10, no. 1, p. 12598, Jul. 2020, doi: https://doi.org/10.1038/s41598-020-69250-1.

[32] J. Hong, Z. Wang, and J. Zhou, “Dynamic Privacy Budget Allocation Improves Data Efficiency of Differentially Private Gradient Descent,” *2022 ACM Conference on Fairness, Accountability, and Transparency*, Jun. 2022, doi: https://doi.org/10.1145/3531146.3533070.

[33] R. S. Antunes, C. André da Costa, A. Küderle, I. A. Yari, and B. Eskofier, “Federated Learning for Healthcare: Systematic Review and Architecture Proposal,” *ACM Transactions on Intelligent Systems and Technology*, vol. 13, no. 4, pp. 1–23, Aug. 2022, doi: https://doi.org/10.1145/3501813.

[34] X. Li, Y. Gu, N. Dvornek, L. H. Staib, P. Ventola, and J. S. Duncan, “Multi-site fMRI analysis using Privacy-Preserving Federated Learning and domain adaptation: ABIDE results,” *Medical Image Analysis*, vol. 65, p. 101765, Oct. 2020, doi: https://doi.org/10.1016/j.media.2020.101765.

[35] J. Xu, B. S. Glicksberg, C. Su, P. Walker, J. Bian, and F. Wang, “Federated Learning for Healthcare Informatics,” *Journal of Healthcare Informatics Research*, vol. 5, Nov. 2020, doi: https://doi.org/10.1007/s41666-020-00082-4.

[36] I. Dayan *et al.*, “Federated Learning for predicting clinical outcomes in patients with COVID-19,” *Nature Medicine*, pp. 1–9, Sep. 2021, doi: https://doi.org/10.1038/s41591-021-01506-3.

[37] M. J. Sheller, G. A. Reina, B. Edwards, J. Martin, and S. Bakas, “Multi-institutional Deep Learning Modeling Without Sharing Patient Data: A Feasibility Study on Brain Tumor Segmentation,” *Brainlesion: Glioma, Multiple Sclerosis, Stroke and Traumatic Brain Injuries*, pp. 92–104, 2019, doi: https://doi.org/10.1007/978-3-030-11723-8_9.

[38] G. A. Kaissis, M. R. Makowski, D. Rückert, and R. F. Braren, “Secure, privacy-preserving and federated machine learning in medical imaging,” Nature Machine Intelligence, vol. 2, no. 6, pp. 305–311, Jun. 2020, doi: https://doi.org/10.1038/s42256-020-0186-1.

[39] N. Rieke *et al.*, “The future of digital health with Federated Learning,” *npj Digital Medicine*, vol. 3, no. 1, pp. 1–7, Sep. 2020, doi: https://doi.org/10.1038/s41746-020-00323-1.

[40] S. Pati *et al.*, “Federated Learning enables big data for rare cancer boundary detection,” *Nature Communications*, vol. 13, no. 1, p. 7346, Dec. 2022, doi: https://doi.org/10.1038/s41467-022-33407-5.

[41] P. Rajpurkar, E. Chen, O. Banerjee, and E. J. Topol, “AI in Health and Medicine,” *Nature Medicine*, vol. 28, no. 1, pp. 31–38, Jan. 2022, doi: https://doi.org/10.1038/s41591-021-01614-0.

[42] T. S. Brisimi, R. Chen, T. Mela, A. Olshevsky, I. Ch. Paschalidis, and W. Shi, “Federated Learning of predictive models from federated Electronic Health Records,” *International Journal of Medical Informatics*, vol. 112, pp. 59–67, Apr. 2018, doi: https://doi.org/10.1016/j.ijmedinf.2018.01.007.

[43] Q. Yang, Y. Liu, T. Chen, and Y. Tong, “Federated Machine Learning,” *ACM Transactions on Intelligent Systems and Technology*, vol. 10, no. 2, pp. 1–19, Feb. 2019, doi: https://doi.org/10.1145/3298981.

[44] C. M. Thwal, K. Thar, Y. L. Tun, and C. S. Hong, “Attention on Personalized Clinical Decision Support System: Federated Learning Approach,” *2021 IEEE International Conference on Big Data and Smart Computing (BigComp)*, pp. 141–147, Jan. 2021, doi: https://doi.org/10.1109/bigcomp51126.2021.00035.

[45] L. Mondrejevski, I. Miliou, A. Montanino, D. Pitts, J. Hollmén, and P. Papapetrou, “FLICU: A Federated Learning Workflow for Intensive Care Unit Mortality Prediction,” *arXiv.org*, 2022. https://arxiv.org/abs/2205.15104 (accessed Feb. 11, 2025).

[46] S. Chen, D. Xue, G. Chuai, Q. Yang, and Q. Liu, “FL-QSAR: a federated Learning-based QSAR prototype for collaborative drug discovery,” *Bioinformatics*, vol. 36, no. 22–23, pp. 5492–5498, Dec. 2020, doi: https://doi.org/10.1093/bioinformatics/btaa1006.

[47] Z. L. Teo *et al.*, “Federated machine learning in healthcare: A systematic review on clinical applications and technical architecture,” *Cell Reports Medicine*, p. 101419, Feb. 2024, doi: https://doi.org/10.1016/j.xcrm.2024.101419.

[48] P. Kairouz *et al.*, “Advances and Open Problems in Federated Learning,” *arXiv:1912.04977 [cs, stat]*, Dec. 2019, Available: https://arxiv.org/abs/1912.04977.

[49] Y. Zhao, M. Li, L. Lai, N. Suda, D. Civin, and V. Chandra, “Federated Learning with Non-IID Data,” *arXiv.org*, 2018. https://arxiv.org/abs/1806.00582

[50] K. Bonawitz *et al.*, “Towards Federated Learning at Scale: System Design,” *Proceedings of Machine Learning and Systems*, vol. 1, pp. 374–388, Apr. 2019, Available: https://proceedings.mlsys.org/paper_files/paper/2019/hash/7b770da633baf74895be22a88

07f1a8f-Abstract.html

[51] F. Sattler, S. Wiedemann, K.-R. Muller, and W. Samek, "Robust and Communication-Efficient Federated Learning From Non-i.i.d. Data," *IEEE Transactions on Neural Networks and Learning Systems*, pp. 1–14, 2019, doi: https://doi.org/10.1109/tnnls.2019.2944481.

[52] B. Hitaj, G. Ateniese, and F. Perez-Cruz, "Deep Models Under the GAN," *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, Oct. 2017, doi: https://doi.org/10.1145/3133956.3134012.

[53] R. Shokri and V. Shmatikov, "Privacy-Preserving Deep Learning," *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security - CCS '15*, 2015, doi: https://doi.org/10.1145/2810103.2813687.

[54] J. So *et al.*, "LightSecAgg: a Lightweight and Versatile Design for Secure Aggregation in Federated Learning," *arXiv.org*, 2021. https://arxiv.org/abs/2109.14236.

[55] Y. Lindell, "Secure multiparty computation," *Communications of the ACM*, vol. 64, no. 1, pp. 86–96, Jan. 2021, doi: https://doi.org/10.1145/3387108.

[56] Oded Goldreich, "Foundations of Cryptography," Jan. 2004, doi: https://doi.org/10.1017/cbo9780511721656.

[57] I. Damgård, M. Fitzi, E. Kiltz, J. B. Nielsen, and T. Toft, "Unconditionally Secure Constant-Rounds Multi-party Computation for Equality, Comparison, Bits and Exponentiation," *Lecture Notes in Computer Science*, pp. 285–304, 2006, doi: https://doi.org/10.1007/11681878_15.

[58] V. Kolesnikov and T. Schneider, "Improved Garbled Circuit: Free XOR Gates and Applications," *Lecture notes in computer science*, pp. 486–498, Aug. 2008, doi: https://doi.org/10.1007/978-3-540-70583-3_40.

[59] Payman Mohassel and Y. Zhang, "SecureML: A System for Scalable Privacy-Preserving Machine Learning," May 2017, doi: https://doi.org/10.1109/sp.2017.12.

[60] J. Konečný, Hugh Brendan Mcmahan, F. X. Yu, P. Richtárik, A. T. Suresh, and D. Bacon, "Federated Learning: Strategies for Improving Communication Efficiency," *arXiv (Cornell University)*, Oct. 2016, doi: https://doi.org/10.48550/arxiv.1610.05492.

[61] L. T. Phong, Y. Aono, T. Hayashi, L. Wang, and S. Moriai, "Privacy-Preserving Deep Learning via Additively Homomorphic Encryption," *IEEE Transactions on Information Forensics and Security*, vol. 13, no. 5, pp. 1333–1345, May 2018, doi: https://doi.org/10.1109/tifs.2017.2787987.

[62] C. Gentry, "Fully homomorphic encryption using ideal lattices," *Proceedings of the 41st annual ACM symposium on Symposium on theory of computing - STOC '09*, 2009, doi: https://doi.org/10.1145/1536414.1536440.

[63] Z. Brakerski, C. Gentry, and V. Vaikuntanathan, "(Leveled) fully homomorphic encryption without bootstrapping," *Proceedings of the 3rd Innovations in Theoretical Computer Science Conference on - ITCS '12*, 2012, doi: https://doi.org/10.1145/2090236.2090262.

[64] A. Acar, H. Aksu, A. S. Uluagac, and M. Conti, "A Survey on Homomorphic Encryption Schemes," *ACM Computing Surveys*, vol. 51, no. 4, pp. 1–35, Sep. 2018, doi: https://doi.org/10.1145/3214303.

[65] I. Chillotti, N. Gama, M. Georgieva, and M. Izabachène, "TFHE: Fast Fully Homomorphic Encryption Over the Torus," *Journal of Cryptology*, Apr. 2019, doi: https://doi.org/10.1007/s00145-019-09319-x.

[66] P. Paillier, "Public-Key Cryptosystems Based on Composite Degree Residuosity Classes," *Advances in Cryptology — EUROCRYPT '99*, pp. 223–238, 2021, doi: https://doi.org/10.1007/3-540-48910-x_16.

[67] C. Dwork, F. McSherry, K. Nissim, and A. Smith, "Calibrating Noise to Sensitivity in Private Data Analysis," *Theory of Cryptography*, pp. 265–284, 2006, doi: https://doi.org/10.1007/11681878_14.

[68] M. Abadi *et al.*, "Deep Learning with Differential Privacy," *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security - CCS'16*, 2016, doi: https://doi.org/10.1145/2976749.2978318.

[69] H. B. McMahan, D. Ramage, K. Talwar, and L. Zhang, "Learning Differentially Private Recurrent Language Models," *arXiv:1710.06963 [cs]*, Feb. 2018, Available: https://arxiv.org/abs/1710.06963

[70] M. Kim, Onur Günlü, and R. F. Schaefer, "Federated Learning with Local Differential Privacy: Trade-offs between Privacy, Utility, and Communication.," *IACR Cryptology ePrint Archive*, vol. 2021, p. 142, Jan. 2021.

[71] Y. Xie, P. Li, C. Wu, and Q. Wu, "Differential Privacy Stochastic Gradient Descent with Adaptive Privacy Budget Allocation," Jan. 2021, doi: https://doi.org/10.1109/iccece51280.2021.9342525.

[72] M. Nasr, R. Shokri, and A. Houmansadr, "Comprehensive Privacy Analysis of Deep Learning: Passive and Active White-box Inference Attacks against Centralized and Federated Learning," *2019 IEEE Symposium on Security and Privacy (SP)*, May 2019, doi: https://doi.org/10.1109/sp.2019.00065.

[73] K. Wei *et al.*, "Federated Learning with Differential Privacy: Algorithms and Performance Analysis," *arXiv:1911.00222 [cs]*, Nov. 2019, Available: https://arxiv.org/abs/1911.00222

[74] S. Guo, J. Yang, S. Long, X. Wang, and G. Liu, "Federated Learning with Differential Privacy via fast Fourier transform for tighter-efficient combining," *Scientific Reports*, vol. 14, no. 1, Nov. 2024, doi: https://doi.org/10.1038/s41598-024-77428-0.

[75] W. Zhu, P. Kairouz, H. Sun, H. B. McMahan, and W. Li, "Federated Heavy Hitters Discovery with Differential Privacy," *International Conference on Artificial Intelligence and Statistics*, 2019.

[76] M. Kim, Onur Gunlu, and R. F. Schaefer, "Federated Learning with Local Differential Privacy: Trade-Offs Between Privacy, Utility, and Communication," *arXiv (Cornell University)*, Jun. 2021, doi: https://doi.org/10.1109/icassp39728.2021.9413764.

[77] X. Zhang, Y. Kang, K. Chen, L. Fan, and Q. Yang, “Trading Off Privacy, Utility and Efficiency in Federated Learning,” *arXiv.org*, 2022. https://arxiv.org/abs/2209.00230

[78] J. Fu *et al.*, “Differentially Private Federated Learning: A Systematic Review,” *arXiv.org*, 2024. https://arxiv.org/abs/2405.08299

[79] D. Commey, S. Hounsinou, and G. V. Crosby, “Securing Health Data on the Blockchain: A Differential Privacy and Federated Learning Framework,” *arXiv (Cornell University)*, May 2024, doi: https://doi.org/10.48550/arxiv.2405.11580.

[80] R. Ahmed, P. Kumar, Thippa Reddy Gadekallu, Naif Khalaf Alshammari, and Fatma Ali Hendaoui, “Efficient Differential Privacy enabled Federated Learning model for detecting COVID-19 disease using chest X-ray images,” *Frontiers in Medicine*, vol. 11, Jun. 2024, doi: https://doi.org/10.3389/fmed.2024.1409314

APPENDIX A DATA PREPROCESSING

```
# ==========================================================
# This script performs data preprocessing by:
# - Handling missing values
# - Normalizing feature values
# - Removing unnecessary columns
# - Preparing the dataset for federated learning
# ==========================================================

# ==========================
# Import Necessary Libraries
# ==========================
import pandas as pd  # Data handling
from sklearn.preprocessing import MinMaxScaler  # Feature normalization

# ==========================
# Load and Explore the Dataset
# ==========================
# Define the dataset file path
file_path = "parkinsons.data"

# Attempt to load the dataset
try:
    dataset = pd.read_csv(file_path)
    print(" Dataset successfully loaded!")
except FileNotFoundError:
    print(f" Error: File '{file_path}' not found. Please check the file path.")
    exit()

# Display a preview of the dataset
print("\n Dataset Preview:")
print(dataset.head())

# ==========================
# Data Exploration
# ==========================
# Check for missing values in each column
print("\n Checking for Missing Values:")
missing_values = dataset.isnull().sum()
print(missing_values)

# Display dataset structure (column names, data types, and non-null counts)
print("\n Dataset Information:")
dataset.info()

# Generate summary statistics for numerical columns
print("\n Summary Statistics:")
print(dataset.describe())

# ==========================
# Data Cleaning: Remove Unnecessary Columns
# ==========================
# Certain columns do not contribute to model learning
columns_to_drop = ['name']
if 'name' in dataset.columns:
    dataset = dataset.drop(columns=columns_to_drop, axis=1)
    print("\n Unnecessary columns removed.")

# ==========================
# Separate Features (X) and Labels (Y)
# ==========================
# 'status' is the target variable
X = dataset.drop(columns=['status'])  # Features (independent variables)
Y = dataset['status']  # Labels (dependent variable: 0 = healthy, 1 =
Parkinson's patient)

print("\n Features and labels separated.")

# ==========================
# Handle Missing Values
# ==========================
# If any missing values exist, replace them with the column mean
if missing_values.sum() > 0:
    X = X.fillna(X.mean())
    print("\n Missing values handled (replaced with column mean).")
else:
    print("\n No missing values detected.")

# ==========================
# Normalize Feature Values
# ==========================
# Normalize feature values between 0 and 1 using Min-Max Scaling
scaler = MinMaxScaler()
X_normalized = pd.DataFrame(scaler.fit_transform(X), columns=X.columns)

print("\n Feature normalization completed.")

# ==========================
# Merge Features and Labels into the Final Dataset
# ==========================
# Reconstruct the dataset with normalized features and the target label
prepared_data = pd.concat([X_normalized, Y], axis=1)

# ==========================
# Save the Cleaned Dataset
# ==========================
# Save the processed dataset as a CSV file for future use
output_file = "processed_dataset.csv"
prepared_data.to_csv(output_file, index=False)
```

```
print(f"\n Data preprocessing complete. Cleaned dataset saved as
'{output_file}'.")
```

APPENDIX B DATA PARTITIONING

```
# ========================================================
# This script partitions the preprocessed dataset into multiple partitions
# to simulate federated learning, where each partition represents a different
hospital.
# ========================================================

# ==========================
# Import Necessary Libraries
# ==========================
import pandas as pd  # For data manipulation
from sklearn.model_selection import train_test_split  # For data partitioning
import os  # For directory handling

# ==========================
# Ensure Output Directory Exists
# ==========================
# Federated learning requires decentralized data stored in separate files.
# We create a dedicated "datasets" folder to store partitioned files.
output_dir = "./datasets"
os.makedirs(output_dir, exist_ok=True)  # Create the folder if it doesn't exist

print("\n Output directory for partitioned datasets created.")

# ==========================
# Load the Preprocessed Dataset
# ==========================
# The dataset have already been preprocessed and saved as
'processed_dataset.csv'
file_path = "processed_dataset.csv"

# Try loading the dataset and handle errors if the file is missing
try:
    data = pd.read_csv(file_path)
    print("\n Preprocessed dataset successfully loaded!")
except FileNotFoundError:
    print(f"\n Error: File '{file_path}' not found. Ensure preprocessing is
completed before partitioning.")
    exit()

# ==========================
# Split Dataset into Multiple Partitions
# ==========================
# To simulate multiple hospitals in federated learning, we partition the data
into 5 partitions.
num_partitions = 5  # Number of federated clients (hospitals)

# Loop through and create partitions
for i in range(num_partitions):
    # Use stratified sampling to maintain class distribution (80% data is
excluded per partition)
    partition, _ = train_test_split(data, test_size=0.8, random_state=i,
stratify=data["status"])

    # Save partitioned dataset to a CSV file
    partition_file = f"{output_dir}/hospital_{i}.csv"
    partition.to_csv(partition_file, index=False)

    print(f" Partition saved: {partition_file} (Rows: {len(partition)})")

# ==========================
# Final Status Message
# ==========================
print("\n Partitioning completed! The datasets are saved in the './datasets'
directory.")
```

APPENDIX C FEDERATED LEARNING CLIENT WITHOUT DIFFERENTIAL PRIVACY (DP)

```python
# ========================================================
# This script implements a Federated Learning (FL) client that:
# - Loads local hospital data
# - Trains a machine learning model locally
# - Sends updated model parameters to the FL server
# - Evaluates model performance on hospital test data
# ========================================================

# ==========================
# Import Necessary Libraries
# ==========================
import sys
import flwr as fl  # Flower FL framework
import tensorflow as tf  # Deep learning library
import pandas as pd  # Data handling
from sklearn.model_selection import train_test_split  # Data splitting
from sklearn.preprocessing import StandardScaler  # Feature normalization

# ==========================
# Define the Federated Learning Client
# ==========================
class HospitalClient(fl.client.NumPyClient):
    """
    This class represents a federated learning client for a hospital.
    Each hospital trains its model locally and communicates updates with the
server.
    """

    def __init__(self, model, x_train, y_train, x_test, y_test):
        """
        Initializes the FL client with a neural network model and dataset.

        :param model: The local machine learning model
        :param x_train: Training features
        :param y_train: Training labels
        :param x_test: Testing features
        :param y_test: Testing labels
        """
        self.model = model
        self.x_train = x_train
        self.y_train = y_train
        self.x_test = x_test
        self.y_test = y_test

    def get_parameters(self, config):
        """Returns the current model parameters (weights)."""
        return self.model.get_weights()
    def fit(self, parameters, config):
        """
        Trains the model on the local dataset for one round of FL.

        :param parameters: Global model parameters received from the FL server
        :param config: Additional configuration
        :return: Updated model parameters, number of training samples, and
accuracy
        """
        self.model.set_weights(parameters)  # Load global model parameters

        # Train the model on hospital data
        history = self.model.fit(self.x_train, self.y_train, epochs=5,
batch_size=32, verbose=0)

        # Return updated model parameters and latest accuracy
        return self.model.get_weights(), len(self.x_train), {
            "accuracy": history.history["accuracy"][-1]
        }

    def evaluate(self, parameters, config):
        """
        Evaluates the local model training on the hospital test dataset.

        :param parameters: Global model parameters received from the FL server
        :param config: Additional configuration
        :return: Model loss, number of test samples, and accuracy
        """
        self.model.set_weights(parameters)  # Load global model parameters
        loss, accuracy = self.model.evaluate(self.x_test, self.y_test,
verbose=0)
        return loss, len(self.x_test), {"accuracy": accuracy}

# ==========================
# Load and Preprocess Local Hospital Data
# ==========================
def load_data(hospital_id):
    """
    Loads the dataset for a specific hospital and preprocesses it.

    :param hospital_id: ID of the hospital
    :return: Normalized training and testing data
    """
    filename = f"./datasets/hospital_{hospital_id}.csv"

    try:
        data = pd.read_csv(filename)
        print(f" Successfully loaded dataset for Hospital {hospital_id}")
    except FileNotFoundError:
        print(f" Error: {filename} not found. Ensure partitioning is completed
before running this script.")
        exit()

    # Ensure the label column ('status') is correctly positioned
    label_col = "status"
    X = data.drop(columns=[label_col])
    y = data[label_col].values

    # Split dataset into training and testing sets (80%-20% split)
    x_train, x_test, y_train, y_test = train_test_split(X, y, test_size=0.2,
random_state=42, stratify=y)

    # Normalize feature values to ensure balanced training
    scaler = StandardScaler()
    x_train = scaler.fit_transform(x_train)
    x_test = scaler.transform(x_test)

    return x_train, x_test, y_train, y_test

# ==========================
# Define the Neural Network Model
# ==========================
def create_model(input_shape):
    """
    Builds a feedforward neural network for binary classification.

    :param input_shape: Shape of input features
    :return: Compiled TensorFlow model
    """
    model = tf.keras.Sequential([
        tf.keras.layers.Dense(64, activation="relu", input_shape=input_shape),
        tf.keras.layers.Dense(32, activation="relu"),
        tf.keras.layers.Dense(1, activation="sigmoid")  # Output layer for
binary classification
    ])

    # Compile model using Adam optimizer and binary cross-entropy loss
    model.compile(optimizer="adam", loss="binary_crossentropy",
metrics=["accuracy"])
    return model

# ==========================
# Start the Federated Learning Client
# ==========================
if __name__ == "__main__":
    if len(sys.argv) != 2:
        print("Usage: python client.py <hospital_id>")
        sys.exit(1)

    # Read the hospital ID from the command-line argument
    hospital_id = int(sys.argv[1])

    # Load the hospital-specific dataset
    x_train, x_test, y_train, y_test = load_data(hospital_id)

    # Initialize the local model
    model = create_model((x_train.shape[1],))

    # Create the FL client
    client = HospitalClient(model, x_train, y_train, x_test, y_test)

    # Start the FL client and connect it to the central FL server
    fl.client.start_numpy_client(server_address="127.0.0.1:8081",
client=client)
```

APPENDIX D FEDERATED LEARNING SERVER WITHOUT DIFFERENTIAL PRIVACY (DP)

```
# ==========================================================
# This script implements the federated learning server using the Flower
framework.
# The server:
# - Coordinates model training across multiple hospitals
# - Aggregates local client updates
# - Evaluates global model performance
# ==========================================================

# ===========================
# Import Necessary Libraries
# ===========================
import flwr as fl  # Flower framework for federated learning
import json  # Used for saving training results

# ===========================
# Define Aggregation Function
# ===========================
def weighted_average(metrics):
    """
    Aggregates accuracy and loss metrics across all participating hospital
clients.

    :param metrics: List of tuples (num_samples, metrics_dict) from each
client.
    :return: Dictionary with aggregated loss and accuracy.
    """
    total_samples = sum([num_examples for num_examples, _ in metrics])

    # Compute weighted average for all reported metrics
    aggregated_metrics = {
        key: sum([num_examples * metric[key] for num_examples, metric in
metrics]) / total_samples
        for key in metrics[0][1].keys()
    }
    return aggregated_metrics

# ===========================
# Set Up Different Training Round Configurations
# ===========================
# The server will test multiple configurations with varying training rounds.
round_configs = [10, 20, 30, 50, 80, 100]  # Different training rounds
results = {}  # Dictionary to store results for each configuration

# ===========================
# Start Federated Learning Process
# ===========================
for num_rounds in round_configs:
    print(f"\n Starting federated training for {num_rounds} rounds...")

    # Define the FedAvg aggregation strategy
    strategy = fl.server.strategy.FedAvg(
        fraction_fit=1.0,  # Require all clients to participate in training
        min_fit_clients=5,  # Minimum required clients for training
        min_available_clients=5,  # Minimum available clients needed to start
training
        evaluate_metrics_aggregation_fn=weighted_average,  # Aggregate
evaluation metrics
        fit_metrics_aggregation_fn=weighted_average,  # Aggregate training
metrics
    )

    try:
        # Start the federated learning server
        history = fl.server.start_server(
            server_address="127.0.0.1:8081",
            config=fl.server.ServerConfig(num_rounds=num_rounds),
            strategy=strategy,
        )

        # Store results for the current training round configuration
        results[num_rounds] = {
            "losses_distributed": history.losses_distributed,
            "metrics_distributed": history.metrics_distributed,
        }

        print(f" Federated training completed for {num_rounds} rounds.")

    except Exception as e:
        print(f" Error during training for {num_rounds} rounds: {str(e)}")

# ==========================
# Save Training Results to JSON
# ==========================
output_file = "results_without_DP.json"
try:
    with open(output_file, "w") as f:
        json.dump(results, f)
    print(f"\n All training results saved in '{output_file}'.")
except IOError:
    print("\n Failed to save training results. Check file permissions or
available storage.")
```

APPENDIX E FEDERATED LEARNING CLIENT WITH DIFFERENTIAL PRIVACY (DP)

```
# ========================================================
# This script implements a federated learning (FL) client with DP.
# DP is applied by clipping model updates and adding Gaussian noise before
sending them to the server.
# ========================================================

# ==========================
# Import Necessary Libraries
# ==========================
import sys
import flwr as fl  # Flower framework for federated learning
import tensorflow as tf  # Deep learning library
import pandas as pd  # Data handling
import time  # Track training time
import numpy as np  # Math operations
from sklearn.model_selection import train_test_split  # Train-test splitting
from sklearn.preprocessing import StandardScaler  # Feature normalization

# ==========================
# Define DP Clipping and Noise Addition
# ==========================
clipping_threshold = 1.0  # Clipping threshold (C)
noise_multiplier = 1.0  # Noise level for DP

def clip_and_noise(weights):
    """
    Clips and adds Gaussian noise to model updates before sending them to the
server.
    Uses the formula:
    Clipped Gradient = g / max(1, ||g|| / C)

    :param weights: Model weight updates
    :return: Clipped and noisy updates
    """
    norm = np.linalg.norm([np.linalg.norm(w) for w in weights])  # Compute
global norm
    scaling_factor = max(1, norm / clipping_threshold)  # Compute scaling
factor

    # Apply clipping
    clipped_weights = [w / scaling_factor for w in weights]

    # Add Gaussian noise
    noise = [np.random.normal(0, clipping_threshold * noise_multiplier,
w.shape) for w in clipped_weights]
    noisy_weights = [w + n for w, n in zip(clipped_weights, noise)]
    return noisy_weights

# ==========================
# Define the Federated Learning Client
# ==========================
class HospitalClient(fl.client.NumPyClient):
    def __init__(self, model, x_train, y_train, x_test, y_test):
        """
        Initializes the federated learning client with differential privacy
(DP).
        :param model: The neural network model for training
        :param x_train, y_train: Training dataset
        :param x_test, y_test: Test dataset
        """
        self.model = model
        self.x_train = x_train
        self.y_train = y_train
        self.x_test = x_test
        self.y_test = y_test

    def get_parameters(self, config):
        """Returns the local model parameters."""
        return self.model.get_weights()

    def fit(self, parameters, config):
        """
        Trains the local model with DP (clipping + noise addition).
        """
        self.model.set_weights(parameters)

        # Start tracking training time
        start_time = time.time()

        # Train model
        history = self.model.fit(self.x_train, self.y_train, epochs=5,
batch_size=32, verbose=0)

        # End tracking time
        training_time = time.time() - start_time

        # Apply clipping + noise before sending updates
        updated_weights = clip_and_noise(self.model.get_weights())

        return updated_weights, len(self.x_train), {
            "accuracy": history.history["accuracy"][-1],
            "loss": history.history["loss"][-1],
            "training_time": training_time,  # Track training time per round
        }

    def evaluate(self, parameters, config):
        """Evaluates the local model and returns evaluation metrics."""
        self.model.set_weights(parameters)
        loss, accuracy = self.model.evaluate(self.x_test, self.y_test,
verbose=0)
        return loss, len(self.x_test), {"accuracy": accuracy}

# ==========================
# Load and Preprocess Local Hospital Data
# ==========================
def load_data(hospital_id):
    """
    Loads the partitioned dataset for a specific hospital.
    :param hospital_id: The hospital ID
    :return: Normalized training and test datasets
    """
    filename = f"./datasets/hospital_{hospital_id}.csv"
    data = pd.read_csv(filename)

    # Drop non-numeric columns
    if not pd.api.types.is_numeric_dtype(data.iloc[: , 0]):
        data = data.iloc[: , 1:]

    # Extract features and labels
    x = data.iloc[: , : -1].apply(pd.to_numeric,
errors='coerce').fillna(0).values
    y = data.iloc[: , -1].values

    # Split into training and test sets (80% train, 20% test)
    x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=0.2,
random_state=42)

    # Normalize features
    scaler = StandardScaler()
    x_train = scaler.fit_transform(x_train)
    x_test = scaler.transform(x_test)

    return x_train, x_test, y_train, y_test

# ==========================
# Define the Neural Network Model
# ==========================
def create_model(input_shape):
    """
    Builds a neural network for federated learning
    :param input_shape: The shape of the input feature vector
    :return: Compiled Keras model
    """
    model = tf.keras.Sequential([
        tf.keras.layers.Dense(64, activation="relu", input_shape=input_shape),
        tf.keras.layers.Dense(32, activation="relu"),
        tf.keras.layers.Dense(1, activation="sigmoid")  # Binary classification
(Parkinson's detection)
```

```
    ])

    # Use Adam optimizer
    optimizer = tf.keras.optimizers.Adam(learning_rate=0.001)

    # Compile model
    model.compile(optimizer=optimizer, loss="binary_crossentropy",
metrics=["accuracy"])
    return model

# ==========================
# Start the Federated Learning Client with DP
# ==========================
if __name__ == "__main__":
    if len(sys.argv) != 2:
        print("Usage: python client.py <hospital_id>")
        sys.exit(1)

    # Parse hospital ID from command-line argument
    hospital_id = int(sys.argv[1])

    # Load hospital-specific dataset
    x_train, x_test, y_train, y_test = load_data(hospital_id)

    # Create model
    model = create_model((x_train.shape[1],))

    # Initialize the federated learning client
    client = HospitalClient(model, x_train, y_train, x_test, y_test)

    # Start the FL client with DP and connect to the FL server
    fl.client.start_numpy_client(server_address="127.0.0.1:8081",
client=client)
```

APPENDIX F FEDERATED LEARNING SERVER WITH DIFFERENTIAL PRIVACY (DP)

```
# ==========================================================
# This script implements a federated learning (FL) server that aggregates model
updates
# while ensuring privacy protection. Differential Privacy (DP) is applied at
the
# client level using local DP mechanisms.
# ==========================================================

# ==========================
# Import Necessary Libraries
# ==========================
import flwr as fl  # Flower framework for federated learning
import json  # Store training results
import numpy as np  # Mathematical calculations
import pandas as pd  # Save results in CSV format
import time  # Track training time

# ==========================
# Define Standard Aggregation Function
# ==========================
def weighted_average(metrics):
    """
    Aggregates accuracy and loss metrics across multiple hospital clients.
    :param metrics: List of tuples (num_samples, metrics_dict) from each client
    :return: Aggregated dictionary with average loss and accuracy
    """
    total_samples = sum([num_examples for num_examples, _ in metrics])

    # Compute weighted averages for all reported metrics
    aggregated_metrics = {
        key: sum([num_examples * metric[key] for num_examples, metric in
metrics]) / total_samples
        for key in metrics[0][1].keys()
    }
    return aggregated_metrics

# ==========================
# Compute Privacy Budget (ε)
# ==========================
def compute_epsilon(clipping_threshold, noise_multiplier, num_rounds, delta=1e-
5):
    """
    Computes the privacy budget (ε) using the Gaussian DP formula.
    :param clipping_threshold: The clipping threshold C
    :param noise_multiplier: The DP noise level σ
    :param num_rounds: Total number of training rounds (T)
    :param delta: Privacy parameter (default 1e-5)
    :return: privacy budget ε
    """
    return (clipping_threshold / noise_multiplier) * np.sqrt(2 * num_rounds *
np.log(1 / delta))

# ==========================
# Federated Learning Process with DP
# ==========================
# Define different training round configurations for experiments
round_configs = [10, 20, 30, 50, 80, 100]
results = {}  # Dictionary to store results for each configuration

# Define DP parameters
clipping_threshold = 1.0
noise_multiplier = 1.0
delta = 1e-5  # Small probability of failure

for num_rounds in round_configs:
    print(f"\n Starting DP-enabled federated training for {num_rounds}
rounds...")

    # Compute privacy budget
    privacy_budget = compute_epsilon(clipping_threshold, noise_multiplier,
num_rounds, delta)

    # Define standard federated learning strategy
    strategy = fl.server.strategy.FedAvg(
        fraction_fit=1.0,  # Require all clients to participate in training
        min_fit_clients=5,  # Minimum number of clients for training
        min_available_clients=5,  # Minimum number of clients that must be
available
        evaluate_metrics_aggregation_fn=weighted_average,
        fit_metrics_aggregation_fn=weighted_average,
    )

    # Start the federated learning server
    history = fl.server.start_server(
        server_address="127.0.0.1:8081",
        config=fl.server.ServerConfig(num_rounds=num_rounds),
        strategy=strategy,
    )

    # Store results for this training round configuration
    results[num_rounds] = {
        "losses_distributed": history.losses_distributed,
        "metrics_distributed": history.metrics_distributed,
        "privacy_budget_ε": privacy_budget,  # Store computed ε
    }

    print(f" Training for {num_rounds} rounds completed with ε =
{privacy_budget:.2f}")
```

```
# ==========================
# Save Training Results
# ==========================
output_file = "results_with_DP.json"
with open(output_file, "w") as f:
    json.dump(results, f)

print(f"\n DP-enabled training completed. Results saved in '{output_file}'.")
```

APPENDIX G PRIVACY BUDGET (ε) CALCULATION

```
# ==========================================================
# This script calculates the privacy budget (ε) for different
# training rounds in a Federated Learning system.
# ==========================================================

# ===========================
# Import Required Python Libraries
# ===========================
import numpy as np  # Numerical operations
import pandas as pd  # Data handling

# ===========================
# Define Privacy Budget Calculation Function
# ===========================
def compute_privacy_budget(noise_multiplier, num_rounds,
clipping_threshold=1.0, delta=1e-5):
    """
    Computes the privacy budget (ε) for given training rounds and DP noise
multiplier.

    :param noise_multiplier: The DP noise multiplier (σ).
    :param num_rounds: The number of federated learning rounds (T).
    :param clipping_threshold: The clipping threshold (C)is 1.0.
    :param delta: Privacy parameter delta, default is 1e-5.
    :return: Calculated privacy budget (ε).
    """
    return (clipping_threshold / noise_multiplier) * np.sqrt(2 * num_rounds *
np.log(1 / delta))

# ===========================
# Define Parameters for Privacy Budget Calculation
# ===========================
noise_levels = [0.01, 0.05, 0.1, 0.5, 1.0]  # noise multipliers (σ values)
rounds_list = [10, 20, 50, 80, 100]  # Different FL training rounds (T values)

# ===========================
# Compute Privacy Budgets for Various Configurations
# ===========================
results = []
for noise in noise_levels:
    for rounds in rounds_list:
        epsilon = compute_privacy_budget(noise_multiplier=noise,
num_rounds=rounds)
        results.append({"Noise Multiplier (σ)": noise, "Training Rounds (T)":
rounds, "Privacy Budget (ε)": round(epsilon, 3)})

# Convert results into a DataFrame
df_results = pd.DataFrame(results)
```

```
# ==========================
# Save Privacy Budget Results
# ==========================
df_results.to_csv("epsilon_results.csv", index=False)

# Display the privacy budget table
print("\n Privacy Budget Calculation Table:\n")
print(df_results.to_string(index=False))

print(" Privacy Budget results saved as 'epsilon_results.csv'")
```

APPENDIX H LINE CHART – MODEL ACCURACY AND LOSS WITHOUT DIFFERENTIAL PRIVACY (DP)

```python
# ========================================================
# This script visualizes the accuracy and loss curves for the FL model trained
# without Differential Privacy (DP).
# ========================================================

# ==========================
# Import Necessary Libraries
# ==========================
import pandas as pd  # Data handling
import matplotlib.pyplot as plt  # Plotting library
import numpy as np  # Mathematical operations

# ==========================
# Load Federated Learning Results
# ==========================
# Read FL training results from CSV file
df = pd.read_csv("results_without_DP.csv")

# ==========================
# Define Smoothing Function
# ==========================
def moving_average(data, window_size=5):
    """
    Applies a moving average filter to smooth curves.
    :param data: Input list or array
    :param window_size: Number of points for averaging
    :return: Smoothed data
    """
    return np.convolve(data, np.ones(window_size)/window_size, mode='valid')

# ==========================
# Extract Training Metrics
# ==========================
rounds = df["Round"]
train_accuracy = df["Train_Accuracy"]
test_accuracy = df["Test_Accuracy"]
train_loss = df["Train_Loss"]
test_loss = df["Test_Loss"]

# Apply smoothing
window_size = 5  # Adjust for smoothing
train_accuracy_smooth = moving_average(train_accuracy, window_size)
test_accuracy_smooth = moving_average(test_accuracy, window_size)
train_loss_smooth = moving_average(train_loss, window_size)
test_loss_smooth = moving_average(test_loss, window_size)
rounds_smooth = rounds[:len(train_accuracy_smooth)]
```

```
# ==========================
# Plot Model Accuracy and Loss
# ==========================
fig, axes = plt.subplots(1, 2, figsize=(14, 5))

# Plot Accuracy (Left)
axes[0].plot(rounds_smooth, train_accuracy_smooth, label="Train Accuracy",
color="blue")
axes[0].plot(rounds_smooth, test_accuracy_smooth, label="Test Accuracy",
color="orange")
axes[0].set_title("Model Accuracy")
axes[0].set_xlabel("Rounds")
axes[0].set_ylabel("Accuracy")
axes[0].legend()

# Plot Loss (Right)
axes[1].plot(rounds_smooth, train_loss_smooth, label="Train Loss",
color="blue")
axes[1].plot(rounds_smooth, test_loss_smooth, label="Test Loss",
color="orange")
axes[1].set_title("Model Loss")
axes[1].set_xlabel("Rounds")
axes[1].set_ylabel("Loss")
axes[1].legend()

plt.tight_layout()
plt.show()
```

APPENDIX I HEATMAP – IMPACT OF NOISE & ROUNDS ON ACCURACY & LOSS

```
# ========================================================
# This script generates a heatmap to visualize how different noise multipliers
(σ)
# and training rounds impact model accuracy and loss in Federated Learning with
DP.
# ========================================================

# ==========================
# Import Necessary Libraries
# ==========================
import pandas as pd  # Data handling
import numpy as np  # Mathematical operations
import seaborn as sns  # Visualization library
import matplotlib.pyplot as plt  # Plotting

# ==========================
# Load Federated Learning Results
# ==========================
# Read results from CSV file containing accuracy and loss values for different
σ and rounds
df = pd.read_csv("heatmap_results_DP.csv")

# Ensure columns include " noise_multiplier ", "Rounds", "Accuracy", and "Loss"
if not all(col in df.columns for col in [" noise_multiplier ", "Rounds",
"Accuracy", "Loss"]):
    raise ValueError("CSV file must contain columns: ' noise_multiplier ',
'Rounds', 'Accuracy', 'Loss'")

# ==========================
# Prepare Data for Heatmap
# ==========================
# Pivot tables to format data for heatmap visualization
accuracy_matrix = df.pivot(" noise_multiplier ", "Rounds", "Accuracy")
loss_matrix = df.pivot(" noise_multiplier ", "Rounds", "Loss")

# ==========================
# Generate Heatmap for Accuracy
# ==========================
plt.figure(figsize=(10, 6))
sns.heatmap(accuracy_matrix, annot=True, fmt=".2f", cmap="coolwarm",
linewidths=0.5)
plt.title("Impact of Noise Multiplier (σ) and Rounds on Model Accuracy")
plt.xlabel("Training Rounds")
plt.ylabel("Noise Multiplier (σ)")
plt.show()

# ==========================
# Generate Heatmap for Loss
# ==========================
plt.figure(figsize=(10, 6))
sns.heatmap(loss_matrix, annot=True, fmt=".2f", cmap="coolwarm",
linewidths=0.5)
plt.title("Impact of Noise Multiplier (σ) and Rounds on Model Loss")
plt.xlabel("Training Rounds")
plt.ylabel("Noise Multiplier (σ)")
plt.show()
```

APPENDIX J SCATTER PLOT – PRIVACY BUDGET (ε) VS. MODEL ACCURACY

```
# ========================================================
# This script generates a scatter plot to show the relationship between the
privacy budget (ε) and model accuracy. It helps analyze the trade-off between
privacy and model performance.
# ========================================================

# ==========================
# Import Python Libraries
# ==========================
import pandas as pd  # Data handling
import numpy as np  # Numerical operations
import matplotlib.pyplot as plt  # Plotting library

# ==========================
# Load Privacy Budget and Accuracy Data
# ==========================
# Read results from a CSV file containing privacy budget (ε) and accuracy
values
df = pd.read_csv("privacy_vs_accuracy.csv")

# Ensure the required columns exist
if not all(col in df.columns for col in ["Privacy_Budget_ε", "Accuracy"]):
    raise ValueError("CSV file must contain columns: 'Privacy_Budget_ε' and
'Accuracy'")

# Extract data for plotting
epsilon_values = df["Privacy_Budget_ε"]
accuracy_values = df["Accuracy"]

# ==========================
# Generate Scatter Plot
# ==========================
plt.figure(figsize=(8, 6))
plt.scatter(epsilon_values, accuracy_values, color="blue", alpha=0.7,
edgecolors="black")

# Add labels and title
plt.xlabel("Privacy Budget (ε)")
plt.ylabel("Model Accuracy")
plt.title("Privacy Budget (ε) vs. Model Accuracy")

# Display grid
plt.grid(True, linestyle="--", alpha=0.6)

# Show the scatter plot
plt.show()
```

APPENDIX K 3D BAR CHART – PRIVACY BUDGET (ε) VS. TOTAL TRAINING TIME

```
# ========================================================
# This script generates a 3D plot to visualize the relationship between privacy
budget (ε),
# noise multiplier (σ), and total training time in Federated Learning with DP.
# ========================================================

# ==========================
# Import Required Python Libraries
# ==========================
import pandas as pd  # Data handling
import numpy as np  # Numerical computations
import matplotlib.pyplot as plt  # Plotting library
from mpl_toolkits.mplot3d import Axes3D  # 3D plotting

# ==========================
# Load Privacy Budget and Training Time Data
# ==========================
# Read the dataset containing privacy budget (ε), noise multiplier (σ), and
total training time
df = pd.read_csv("privacy_vs_training_time.csv")

# Ensure the required columns exist
if not all(col in df.columns for col in ["Privacy_Budget_ε", " noise_multiplier
_σ", "Total_Training_Time"]):
    raise ValueError("CSV file must contain columns: 'Privacy_Budget_ε', '
noise_multiplier _σ', and 'Total_Training_Time'")

# Extract values for plotting
epsilon_values = df["Privacy_Budget_ε"]
noise_values = df[" noise_multiplier _σ"]
training_time_values = df["Total_Training_Time"]

# ==========================
# Generate 3D Plot
# ==========================
fig = plt.figure(figsize=(10, 7))
ax = fig.add_subplot(111, projection="3d")

# Create 3D scatter plot
scatter = ax.scatter(epsilon_values, noise_values, training_time_values,
                     c=training_time_values, cmap="viridis",
edgecolors="black")

# Label axes
ax.set_xlabel("Privacy Budget (ε)")
ax.set_ylabel("Noise Multiplier (σ)")
ax.set_zlabel("Total Training Time (seconds)")
ax.set_title("3D Plot: Privacy Budget (ε) vs. Training Time")

# Add color bar
cbar = fig.colorbar(scatter, ax=ax, shrink=0.5, aspect=10)
cbar.set_label("Training Time (seconds)")

# Show plot
plt.show()
```

APPENDIX L BAR CHART– FINAL MODEL ACCURACY VS. NOISE MULTIPLIER (σ)

```
# ========================================================
# This script generates a bar chart showing how different noise multipliers (σ)
# impact the final accuracy of the federated learning model with DP.
# ========================================================

# ==========================
# Import Required Python Libraries
# ==========================
import numpy as np  # Numerical computations
import pandas as pd  # Data handling
import json  # Loading JSON files
import os  # File path handling
import matplotlib.pyplot as plt  # Plotting library

# ==========================
# Define Parameters
# ==========================
file_path_template =
"C:/Users/ghorb/OneDrive/Desktop/parkinsons_federated_learning/training_results
_noise_{:.2f}_rounds_{}.json"

# DP parameters
delta = 1e-5
clipping_threshold = 1.0
noise_levels = [0.01, 0.05, 0.1, 0.5, 1.0]
final_rounds = 100

# Collect results
results = []

# ==========================
# Extract Final Accuracy from Training Results
# ==========================
for sigma in noise_levels:
    file_path = file_path_template.format(sigma, final_rounds)

    if os.path.exists(file_path):
        with open(file_path, 'r') as file:
            data = json.load(file)
            if "accuracy" in data and data["accuracy"]:
                final_accuracy = round(data["accuracy"][-1][1] * 100, 1)  #
Convert accuracy to percentage
                results.append({"σ": sigma, "Accuracy (%)": final_accuracy})

# Convert to DataFrame
df_results = pd.DataFrame(results)
```

```
# ==========================
# Generate Bar Chart
# ==========================
# Define colorblind-friendly colors
colors = ['#377eb8', '#ff7f00', '#4daf4a', '#f781bf', '#a6cee3']

if not df_results.empty:
    plt.figure(figsize=(10, 6))

    # Create bar plot
    bars = plt.bar(df_results["σ"].astype(str), df_results["Accuracy (%)"],
                   color=colors, edgecolor='black', width=0.5)

    # Annotate bars with accuracy values
    for bar, acc in zip(bars, df_results["Accuracy (%)"]):
        plt.text(bar.get_x() + bar.get_width() / 2, bar.get_height() + 0.3,
                 f"{acc:.1f}%", ha='center', fontsize=12, fontweight='bold')

    # Label axes
    plt.xlabel("Noise Multiplier (σ)", fontsize=14)
    plt.ylabel("Final Accuracy (%)", fontsize=14)
    plt.title("Final Model Accuracy vs. Noise Multiplier (σ)", fontsize=16,
weight='bold')

    # Adjust the y-axis
    plt.ylim(85, 100)
    plt.yticks([85, 90, 95, 100], fontsize=12)

    # Add grid lines
    plt.grid(axis='y', linestyle='--', alpha=0.7)

    # Show plot
    plt.show()
else:
    print("\nNo valid data to plot.")
```