PALIHAWADANA, C. 2024. *Improving federated learning performance with similarity guided feature extraction and pruning*. Robert Gordon University, PhD thesis. Hosted on OpenAIR [online]. Available from: https://doi.org/10.48526/rgu-wt-2801100

Improving federated learning performance with similarity guided feature extraction and pruning.

PALIHAWADANA, C.

2024

The author of this thesis retains the right to be identified as such on any occasion in which content from this thesis is referenced or re-used. The licence under which this thesis is distributed applies to the text and any original images only – re-use of any third-party content must still be cleared with the original copyright holder.

OpenAIR @RGU

This document was downloaded from https://openair.rgu.ac.uk

CC BY NC

Improving Federated Learning Performance with Similarity Guided Feature Extraction and Pruning

Chamath Palihawadana

A thesis submitted in partial fulfilment
of the requirements of
Robert Gordon University
for the degree of Doctor of Philosophy

November 2024

Abstract

Federated Learning (FL) is a Machine Learning (ML) paradigm that learns from distributed clients to collaboratively train a global model in a privacy-preserved manner without sharing their private data. Traditional centralised ML approaches require aggregating data from various sources into a single location. This poses substantial risks regarding data privacy, security breaches and compliance with data protection regulations. The primary goal of FL is to ensure data privacy by keeping the raw data on clients' devices while sharing only model parameters with a central server. Each client updates its model locally using its data and then sends these updated parameters (weights) to the server. The server aggregates these updates to create an improved global model, which is then distributed back to the clients. This aggregation process is crucial in FL, as it combines knowledge learned across a diverse range of clients, enabling them to benefit from collective insights while preserving the privacy of their data. This iterative process continues, gradually refining the global model through multiple rounds of local training and aggregation.

However, the adoption of FL is not without its challenges. FL's decentralised nature introduces complexities such as the impact on model performance with aggregation methods, communication overhead and security threats. Existing aggregation methodologies often lack generalisability across different datasets and applications. Moreover, communication efficiency remains a significant bottleneck. The frequent exchange of model updates between clients and the server can be resource-intensive. Additionally, communicating model updates and the distributed nature of FL opens up more threat surfaces for attacks.

In this thesis, we present several methodologies to improve FL performance, with a focus on neural architectures for classification tasks. The models considered include Multinomial Logistic Regression (MLR), Convolutional Neural Networks (CNN), Recurrent Neural Networks (RNN) and Multi-layer Perceptrons (MLP) trained using Stochastic Gradient Descent (SGD). First, we introduce FedSim, a similarity-guided model aggregation algorithm designed to enhance FL accuracy by leveraging inter-client relationships. Instead of relying on client data, we extract similarity knowledge by comparing client gradients. FedSim algorithm decomposes into cluster aggregation and global aggregation steps. Cluster aggregation considers only the updated models within the cluster and then globally aggregates them to ensure better coverage and reduce variance. FedSim

prioritises gradient updates that are consistent across multiple clients to ensure that these aligned updates have a stronger influence on the global model. To evaluate the generalisability of FedSim, we conducted extensive experiments across various datasets and model architectures to demonstrate its effectiveness. Secondly, we introduce FedFT, a communication-efficient FL algorithm that leverages frequency space transformation to reduce communication overhead while maintaining model accuracy. Communicating in the frequency space enables efficient compression due to its compact representation. Its linear properties eliminate the need for multiple transformations during aggregation reducing additional computational overhead. We then address the security challenges in FL, recognising the potential risks posed by gradient inversion attacks. To mitigate these threats, we present pFGD, a defence mechanism that utilises FedFT to protect against such privacy attacks.

Finally, we validated our proposed methodologies through a real-world case study in the healthcare domain. Applying FedSim and FedFT in this context demonstrated their practical applicability and generalisability, highlighting that these methods enhance FL performance. This thesis contributes to the field of FL by introducing novel methods that address critical challenges while ensuring their applicability across diverse scenarios.

Keywords: Federated Learning, Privacy Preserved Machine Learning, Communication Efficiency, Frequency Space Transformation, Model Compression and Pruning

Declaration of Authorship

I declare that I am the sole author of this thesis and that all verbatim extracts contained in the thesis have been identified as such and all sources of information have been specifically acknowledged in the bibliography. Parts of the work presented in this thesis have appeared in the following publications.

- Palihawadana, C., Wiratunga, N., Wijekoon, A. and Kalutarage, H., 2022. FedSim: Similarity guided model aggregation for Federated Learning. Neurocomputing, 483, pp.432-445. (**Chapter 4**)
- Palihawadana, C., Wiratunga, N., Kalutarage, H. and Wijekoon, A., 2023, September. Mitigating Gradient Inversion Attacks in Federated Learning with Frequency Transformation. In European Symposium on Research in Computer Security (pp. 750-760). Cham: Springer Nature Switzerland. (**Chapter 6**)

Submitted/Under Review

- Palihawadana, C., Wiratunga, N., Kalutarage, H. and Wijekoon, A., 2024. FedFT: Improving Communication Performance for Federated Learning with Frequency Space Transformation. Journal of Parallel and Distributed Computing - Submitted/Under Review (**Chapter 5**)
- Palihawadana, C., Wiratunga, N., Kalutarage, H. and Wijekoon, A., 2024. Role of Similarity-Guided Federated Learning for Predicting ICU Mortality with Transformed Feature Representations. Expert Systems with Applications - Submitted/Under Review (**Chapter 7**)

Dissemination of Source Code and Datasets

Below, we list the contributed work and provide links to the corresponding source code repositories. These open-source repositories offer access to the implementation details, datasets introduced and support further research and development in the field. Each link is accompanied by the Chapter associated with the respective repository.

- *FedSim*: https://github.com/chamathpali/FedSim (Chapter 4)
- Fed-MEx Dataset: https://github.com/chamathpali/Fed-MEx (Chapter 4)

- Fed-Goodreads Dataset: https://github.com/chamathpali/Fed-Goodreads (Chapter 4)
- *FedFT*: https://github.com/chamathpali/FedFT (Chapter 5)
- *p*FGD: https://github.com/chamathpali/pFGD (Chapter 6)
- ICU Case Study: https://github.com/chamathpali/FL-eICU (Chapter 7)

Acknowledgements

I would like to express my most profound appreciation to my supervisory team, Prof. Nirmalie Wiratunga, Dr. Harsha Kalutarage and Dr. Anjana Wijekoon from Robert Gordon University (RGU). You have always provided the best support and advice whenever I needed it. I am thankful to the past and present members of the Artificial Intelligence and Reasoning Research Group (AIR) for the wonderful memories and insightful conversations. I would also like to thank the administrative and IT teams at the School of Computing for their invaluable support.

I am deeply grateful to my parents, sister, cousins, extended family, fiancée and friends for their continuous moral support throughout this journey. Your endless love and care made this achievement possible and helped me feel at home even when I was miles away. A special thanks to Guhanathan Poravi and Naomi Krishnarajah from the Informatics Institute of Technology, Sri Lanka for encouraging me to pursue my PhD journey.

Once again, I would like to express my sincere gratitude to Prof. Nirmalie Wiratunga for giving me this incredible opportunity, believing in me and constantly encouraging my academic and personal growth.

Contents

List of Tables

List of Figures

List of Algorithms

List of Acronyms

AI Artificial Intelligence.

APACHE Acute Physiology, Age, and Chronic Health Evaluation.

CCPA California Consumer Privacy Act.

CDP Central Differential Privacy.

CFL Clustered Federated Learning.

CNI Client-wise Non-IID Index.

CNN Convolutional Neural Network.

CPL Client Privacy Leakage.

DCT Discrete Cosine Transform.

DFT Discrete Fourier Transform.

DLG Deep Leakage from Gradient.

DP Differential Privacy.

EEG Electroencephalography.

FedAvg Federated Averaging.

FFT Fast Fourier Transform.

FL Federated Learning.

GAN Generative Adversarial Network.

GCS Glasgow Coma Scale.

GDPR General Data Protection Regulation.

HE Homomorphic Encryption.

ICU Intensive Care Unit.

iDLG Improved Deep Leakage from Gradient.

IID Independently and Identically Distributed.

IoT Internet of Things.

KS Kolmogorov-Smirnov.

LBFGS Limited-memory-Broyden–Fletcher–Goldfarb–Shanno.

LDP Local Differential Privacy.

MAE Mean Absolute Error.

MB Megabytes.

MIMIC Medical Information Mart for Intensive Care.

ML Machine Learning.

MLP Multi-layer Perceptron.

MLR Multinomial Logistic Regression.

MPM Mortality Probability Model.

MSE Mean Square Error.

NI Non-IID Index.

non-IID non-Independently and Identically Distributed.

PCA Principal Component Analysis.

RMSE Root Mean Square Error.

RNN Recurrent Neural Network.

SAPS Simplified Acute Physiology Score.

SGD Stochastic Gradient Descent.

SMOTE Synthetic Minority Oversampling Technique.

SOFA Sequential Organ Failure Assessment.

Chapter 1

Introduction

> "**Privacy is a fundamental human right**. No matter what country you live in, that right should be protected."
>
> *Tim Cook*

In the digital world, humans generate billions of pieces of data, much of which carries immense privacy implications. Data has become a fundamental element for driving innovation in various industries. With the growth of computing power and smart devices, analysing such data has become more accessible. This ease of analysis is primarily enabled by machine learning (ML) advances, where building a ML system only takes a few clicks. ML has become an integral part of our daily lives. We use it every time we interact with smart assistants like Alexa and Siri, and even when we use our mobile phones for various purposes. Every interaction we have with a smart assistant or every action we make on our smartphone can be collected and analysed as data. The data we generate and share daily can include sensitive health information, financial transactions, and social preferences. This information is tracked throughout our lives and can have profound implications for individuals, organisations, and society. Fortunately, in the modern world, many regulations and policies are in place to minimise the misuse of such data. Examples include the General Data Protection Regulation (GDPR) in Europe and the California Consumer Privacy Act (CCPA) in the United States, which imposes strict data handling and privacy guidelines.

With the privacy constraints that limit access to data, it is now a challenge for researchers

and enterprises to train models. Regulations like GDPR restrict the collection and processing of data with personal identifiers. If an individual is identifiable through a data point, GDPR recognises it as "personal" (Voigt and Von dem Bussche, 2017). Even using anonymised datasets has shown the risk of revealing personal information by combining multiple datasets (Sweeney, 2000). ML research in domains like healthcare has the challenge of deploying as real-world applications (Barlow et al., 2019; Mamoshina et al., 2016; Mohr et al., 2017; Shatte et al., 2019); such work can mostly be deployed if access to personal data is available. On the other hand, the data collected in large quantities on our devices can be used to improve intelligence and predictions (Zhao et al., 2018). Limited access and the risk of communicating personal data to a shared location have made it impossible to exploit such data sources to achieve improvements.

To overcome the challenge of data accessibility, distributed machine learning was introduced in early research (Park and Kargupta, 2002; Shokri and Shmatikov, 2015). However, despite its potential, distributed machine learning has not fully achieved the level of privacy preservation that was originally expected. In 2016, researchers at Google introduced a novel approach *Federated Learning* (FL) to learn from decentralised data while maintaining the privacy of users (clients) (McMahan et al., 2017). The primary goal of FL is to ensure that **client data never leaves the devices**, with only the model parameters being shared with a central server. Each client updates its model locally using its data and then sends these updated parameters (weights) to the server. The server then aggregates these updates to create an improved global model, which is subsequently distributed back to the clients. This iterative learning process allows clients to benefit from the collective knowledge without directly sharing their data.

1.1 Introduction to Federated Learning

FL is a machine learning paradigm that enables distributed clients to train a global model collaboratively while preserving privacy without transferring local data (McMahan et al., 2017). FL builds upon prior work in the distributed machine learning setting and shares common goals and challenges with these earlier models. FL can be categorised into 'cross-device' and 'cross-silo' based on the application (Kairouz et al., 2019). The cross-device method is considered when a large number of clients are in the system (e.g., mobile phone applications, Internet of Things (IoT) devices). Cross-silo is when the number of participating clients is very small but each client has a large quantity of data (e.g., on-premise banking systems, healthcare systems). The benefits of FL were first realised in a

cross-device setting where a large number of clients were involved in the system. Later, organisation-level applications such as geo-distributed data centres (e.g., NHS England and NHS Scotland patient records) reaped the benefits of FL. Using FL in either method will ensure privacy protection and communication efficiency for adopting applications.

FL typically involves the following six steps:

1. Initialisation of a global model and sharing with all clients.
2. The server selects a sample of clients in each round.
3. Clients update the global model locally using their data (typically through Stochastic Gradient Descent (SGD)).
4. Clients send their updated local models to the server.
5. The server aggregates these updates to compute a new global model.
6. All clients receive the updated global model.

Figure 1.1 visually demonstrates these steps. A typical communication round in FL consists of steps 2 through 6. This iterative process involves different samples of clients in each round and may continue for many rounds as required.

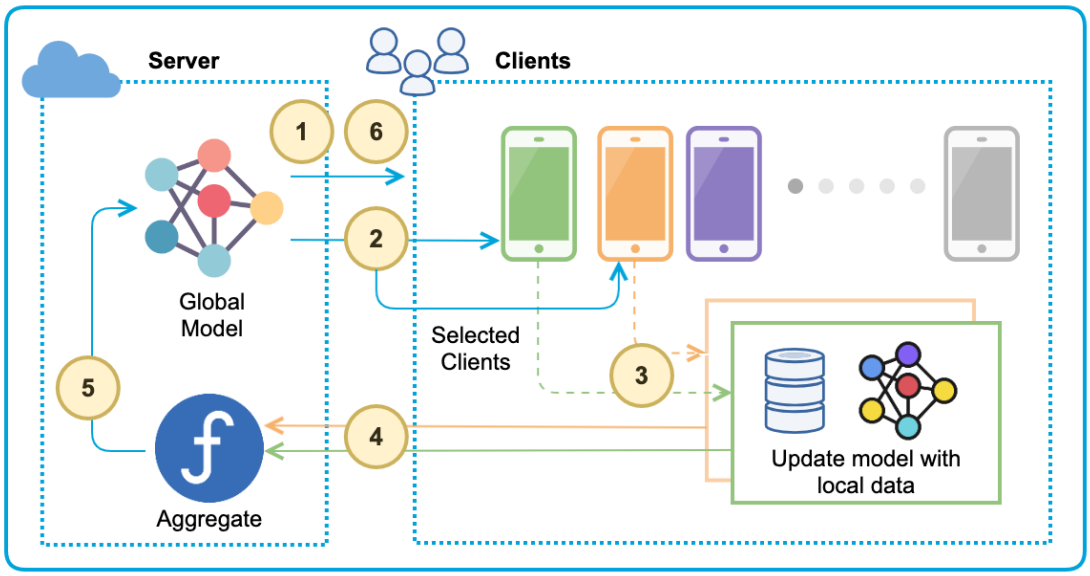

Figure 1.1: Federated Learning Process (FL setting)

The primary objective of a FL system is to train an efficient global model with the use of

decentralised data. With the decentralised nature there is an identified challenge in FL which is the statistical heterogeneity of client data (Smith et al., 2017). The implications of statistical heterogeneity are discussed in detail in Section 2.2. Such heterogeneity makes FL both a challenging and suitable approach for handling diverse data structures in real-world applications (Li et al., 2019; McMahan et al., 2017).

1.1.1 Importance and Challenges in Federated Learning

FL is a growing area in privacy-preserved machine learning, gaining significant attention from industry and academia. Below is a list of critical factors that make FL an exciting area of research and application:

Importance of Federated Learning

- **Privacy Preservation**: The primary objective of FL is to preserve the privacy of training data, which is a significant concern in real-world AI applications. FL keeps the data at the client level and training occurs locally. This is especially essential for domains such as healthcare, finance and smart homes.
- **Scalability**: FL is considered a distributed machine learning approach and scalability is a core feature. Thousands or even millions of clients can contribute towards training a shared model. This scalability is very beneficial for applications involving IoT and mobile devices.
- **Regulatory Compliance**: FL offers a way to comply with legal requirements by ensuring that client data is not communicated to a central server. Although the model is trained on client data, there is no need to enforce very strict rules for handling client data.

Challenges in Federated Learning

Many open research challenges in the FL domain include improving aggregation strategies, reducing communication costs, handling client dropout, supporting non-independently and identically distributed (non-IID) data, and tightening privacy (Aledhari et al., 2020; Zhang et al., 2021). We have identified three key challenges:

- **Model Aggregation**: The aggregation step in FL plays a crucial role in determining the overall system performance. Finding an improved method for model aggregation will help enhance FL effectiveness.

- **Communication Overhead**: The frequent exchange of local models with the server can consume substantial network bandwidth. This exchange occurs in every communication round and there can be many such rounds.
- **Privacy and Security**: Although FL is designed to preserve privacy, there has been evidence and research indicating data leakage through trained parameters. Since models are still exchanged over the network, there is a risk of attacks such as man-in-the-middle, which could compromise the security of the FL setting (Ma et al., 2020; Mothukuri et al., 2021).

1.1.2 Use Cases

There are many use cases where FL is applied; in this section, we will highlight a few applications that take full advantage of FL. Most privacy-sensitive systems, such as healthcare, finance, smart home and mobile applications, can benefit significantly from the FL setting.

Next Word Prediction

One of the largest real-world FL applications is next word prediction on mobile devices. With over 7.5 billion sentences used for training across the English-speaking population in the United States. Researchers Hard et al. (2018a) from Google have applied FL in keyboard next-word prediction in the Gboard[1] application. Figure 1.2 is a sample screenshot from the Gboard application, showing how next-word prediction is used. The input keystrokes are privacy-sensitive and cannot be shared with a central server. Users often enter passwords, credit card numbers, and personal information using the keyboard. FL was adapted to tackle this issue and improve the next-word prediction task. A similar application of FL has been implemented to predict emojis based on the input text, as demonstrated by Ramaswamy et al. (2019). Figure 1.3 shows how emoji prediction is used in keyboard applications.

FL in Healthcare

Recent work in the healthcare domain using FL includes many practical applications. For instance, Brisimi et al. (2018) applied FL to predict future hospitalisations due to heart disease based on existing electronic health records. Their proposed method utilised the cluster primal-dual splitting (cPDS) algorithm, which showed improvements

[1] https://play.google.com/store/apps/details?id=com.google.android.inputmethod.latin

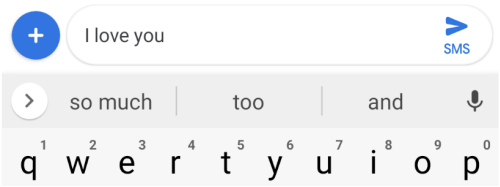

Figure 1.2: Gboard next word prediction. (Hard et al., 2018a)

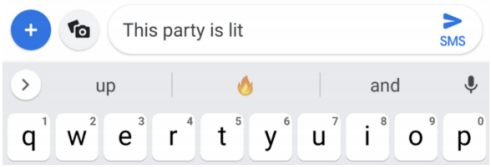

Figure 1.3: Emoji prediction based on text.(Ramaswamy et al., 2019)

in convergence rate without the need to share data. Similarly, Gao et al. (2019) applied FL to Electroencephalography (EEG) classification. EEG data is highly privacy-sensitive as it records the brain's electrical activity, which can reveal human behaviour patterns. The researchers demonstrated the impact of statistical heterogeneity among patients and underscored the importance of FL in managing such scenarios. Work done by Lee et al. (2018) proposed a method using FL to match similar patients based on medical events. Patient data is highly sensitive in this application, necessitating a decentralised learning process. Their method showed promising results, enabling patient matching across different hospitals while preserving data privacy.

Wake Word Detection for Voice Assistants and IoT Applications

The use of voice assistants has increased due to their easy accessibility on mobile phones and smart home devices (Hoy, 2018). Voice assistants are activated using predefined trigger words, known as wake words. Commonly used wake words include 'Hey Siri', 'Ok Google' and 'Alexa' on their respective devices (Hoy, 2018). Wake word detection needs to be highly accurate and efficient to provide the best user experience. Training such wake words involves collecting data from various devices and environments, as voice strength and background noise can vary significantly (Hard et al., 2020; Leroy et al., 2019). FL is ideal for such scenarios since it preserves privacy and allows for decentralised learning. Leroy et al. (2019) have crowd-sourced and created a dataset for the wake word 'Hey Snips', which includes data from 1,774 users and 69,592 utterances.

Another interesting application of FL is in the IoT domain. With more and more IoT devices being deployed in real-world activities like smart homes, smart meters and wearable devices, FL can be highly beneficial. Baucas et al. (2023) proposes an IoT platform using FL to preserve data privacy within the network and incorporate distributed learning with the human activity recognition predictive model. An application for smart homes is proposed by Nour et al. (2021), presenting an architecture that incorporates

federated edge learning to promote data privacy. Communication and computation costs are major barriers to the application of FL in the IoT domain. IoT devices often have limited processing power and energy resources, making it challenging to implement FL efficiently. However, ongoing research is focused on optimising these aspects to make FL more feasible for IoT applications.

1.2 Research Motivation

FL is a relatively new methodology in the comprehensive Artificial Intelligence (AI) domain, as it was first introduced by researchers at Google in 2016 and formally published in 2017 (McMahan et al., 2017). FL has gained much attraction from academics and the industry due to its privacy-preserved nature. This approach is gaining popularity, shown by its use in technologies like Google's keyboard predictions (Hard et al., 2018b) and Apple's speech recognition, because to its ability to protect user privacy. However, despite its promise, FL faces considerable challenges related to statistical heterogeneity, communication efficiency and privacy vulnerabilities. A primary challenge is statistical heterogeneity variability in data distributions across clients. Clients often have non-IID data, leading to conflicting gradients that slow convergence, reduce accuracy and create biased models. While foundational aggregation methods like Federated Averaging (*FedAvg*) (McMahan et al., 2017) and more adaptive approaches such as *FedProx* (Li et al., 2018) aim to address this heterogeneity in FL. However, they often lack the flexibility needed in highly diverse environments where dynamic adaptation to data variation is essential. This gap points to the need for more flexible aggregation methods that leverage inter-client similarities and dynamically adjust to improve accuracy and diversity in FL.

We believe that by understanding the relationships and commonalities among the data on different clients, we can improve FL performance. For example, if we can identify and use data similarities, we can potentially improve how FL models learn from each other. By harnessing unexplored knowledge among clients (e.g., similarity, geo-boundaries), we expect to help guide FL methods to comparable improvements.

Another key limitation in FL is communication efficiency. The need for frequent parameter exchanges between clients and the central server incurs high bandwidth costs, particularly problematic in resource-constrained environments. Some methods, such as model compression and quantisation aim to reduce the communication overhead in FL. While these approaches effectively lower bandwidth requirements, they often come with

trade-offs. Achieving efficient compression of model updates while preserving model performance remains an open challenge in FL, highlighting the need for novel methods that balance communication efficiency with comparable model performance.

Privacy risks presents another major challenge in FL. Although FL keeps data local on the client, model updates can still unintentionally leak sensitive information. For instance, gradient inversion attacks can reconstruct input data from client-shared gradients. Current defences, such as differential privacy and encryption-based methods offer some protection but are computationally intensive and may impact model performance. This highlights the need for secure and efficient methods that prevent data leakage without high computational costs or significantly reducing model quality.

Additionally, we have identified a significant need for more generalisability in the algorithms and methodologies discussed in the current literature. While there are numerous studies on FL, many focus narrowly on specific datasets or tasks. This specialisation limits the applicability of these studies to broader contexts and restricts the overall advancement of the FL field. Our research aims to address this gap by developing methodologies and solutions that are adaptable and extensible, thereby enhancing the generalisability of FL algorithms. Given these challenges, our research is motivated by the opportunity to make FL more adaptable to heterogeneous data, efficient in communication, and secure against privacy threats.

1.3 Research Questions and Objectives

To guide the research, the following research questions were formulated based on the research motivation above.

RQ1: To what extent does the identification and utilisation of similarity knowledge among clients influence model aggregation in FL?

RQ2: Building on the insights from RQ1, how does the chosen aggregation strategy impact communication efficiency in FL, and in what ways can model compression and pruning enhance this efficiency?

RQ3: How do the communication efficiency strategies identified in RQ2 affect the security of FL in terms of data privacy?

In order to answer the challenges and the these research questions, we establish the following objectives:

O1: Conduct a comprehensive literature review to identify and analyse existing FL aggregation methodologies.

This objective involves a systematic review of the current literature to map out and evaluate different aggregation methods used in FL. The review will help identify gaps in the current methodologies and guide the development of improved aggregation strategies.

O2: Develop a similarity-weighted aggregation method that harnesses commonalities in learning behaviour between client models to improve accuracy in FL.

This objective focuses on designing and implementing a novel aggregation method that leverages identified similarities in learning behaviours across different client models participating in FL. By exploiting these similarities, the proposed method aims to more effectively aggregate model updates, which should result in enhanced overall model accuracy and efficiency.

O3: Evaluate the generalisability of the similarity-weighted aggregation from Objective 2 on different model architectures and diverse datasets.

Evaluate the newly developed aggregation method's (Objective 2) effectiveness over various FL scenarios to ensure its robustness and effectiveness. This evaluation will assess the method's performance across diverse model architectures and datasets.

O4: Develop an FL algorithm to improve communication performance, ensuring adaptability across diverse FL scenarios

Develop an algorithm that optimises the communication from/to clients and server in FL, aiming to reduce bandwidth usage. The algorithm developed should integrate seamlessly with the similarity-weighted aggregation method developed in Objective 2. Similar to Objective 3, this algorithm will be tested across different model architectures and datasets to verify its robustness and generalisability.

O5: Evaluate the integrated security benefits within the communication optimised algorithm

This objective focuses on evaluating the security enhancements that have been integrated into the communication optimisation algorithm developed in Objective 4. The effectiveness of security measures in protecting client data during transmission will be evaluated.

O6: Conduct a case study to validate the proposed methodologies in a real-world context

Apply the developed algorithms in a practical setting to demonstrate their effectiveness and practicality. This case study will provide empirical evidence of the methodologies' utility in real-world applications.

1.4 Contributions

This thesis presents four contributions in the field of FL. These contributions are summarised below:

C1 FedSim: Similarity Guided Model Aggregation for Federated Learning

Our first contribution is a similarity-guided FL aggregation algorithm called *FedSim* (Chapter 4). The *FedSim* algorithm, introduced in this work, decomposes FL aggregation into local and global steps. Clients with similar gradients are clustered to provide local aggregations, which can be globally aggregated to ensure better coverage and reduce variance. Through comprehensive comparative studies involving real-world and synthetic datasets, *FedSim* has demonstrated significant performance improvements over state-of-the-art federated learning baselines. Our findings illustrate that *FedSim* not only outperforms these baselines in accuracy and stability but also effectively handles statistical heterogeneity, a common challenge in federated environments.

C2 FedFT: Improving Communication Performance for Federated Learning with Frequency Space Transformation

The second contribution proposes a novel methodology, Federated Frequency-Space Transformation (*FedFT*) (Chapter 5), which leverages Discrete Cosine Transform (DCT) to encode model parameters in the frequency space. *FedFT* enhances communication efficiency by representing the differences in model parameters between communication rounds in a compact frequency space. *FedFT* employs a linear property of the DCT, simplifying the federated aggregation process by eliminating the need for multiple transformations. This makes the methodology computationally efficient and reduces the potential for errors or data loss during transformation processes. A key feature of *FedFT* is its compatibility with a wide range of existing FL methodologies and neural architectures.

C3 Pruned Frequency-based Gradient Defence (*p*FGD) against Gradient Inversion Attacks

The third contribution of this thesis is developing a robust defence strategy against gradient inversion attacks in FL. Utilising *FedFT*, we developed the Pruned Frequency-based Gradient Defence, termed *p*FGD (Chapter 6). We are combining DCT and gradient pruning to obfuscate and minimise the transmission of sensitive information. Experimental validation on a real-world dataset confirms that *p*FGD offers robust defence, enhancing the privacy and security of FL systems against such attacks.

C4 Conducting a Case Study to Validate the Proposed Methodologies in a Real-World Context

The final contribution involves the application of the previously introduced methodologies, *FedSim* and *FedFT*, to a practical, real-world problem in the healthcare sector (Chapter 7). The efficacy of these methods are validated through a case study centred on a critical dataset in healthcare research. This case study not only tests the practical implementation of our approaches but also demonstrates their potential to tackle complex challenges within the healthcare domain, emphasising their broad applicability and transformative impact. Selecting a healthcare scenario highlights the relevance and urgency of advancing FL solutions in contexts where privacy and data sensitivity are critical.

1.5 Overview of Research Framework Relationships

Figure 1.4 presents a comprehensive visual overview of the research framework, illustrating the relationships between research questions, objectives, contributions and chapters. The diagram begins with the three research questions (RQ1, RQ2 and RQ3) that guide the study. Each research question is associated with multiple objectives (O1 through O6), forming the next tier of the diagram. The objectives are linked to specific contributions (C1 through C4) of the research.

1.6 Thesis Overview

The rest of the thesis is organised as follows;

Chapter 2 lays the groundwork for our exploration of FL by reviewing the key challenges and research questions that guide this study. We review standard aggregation methods,

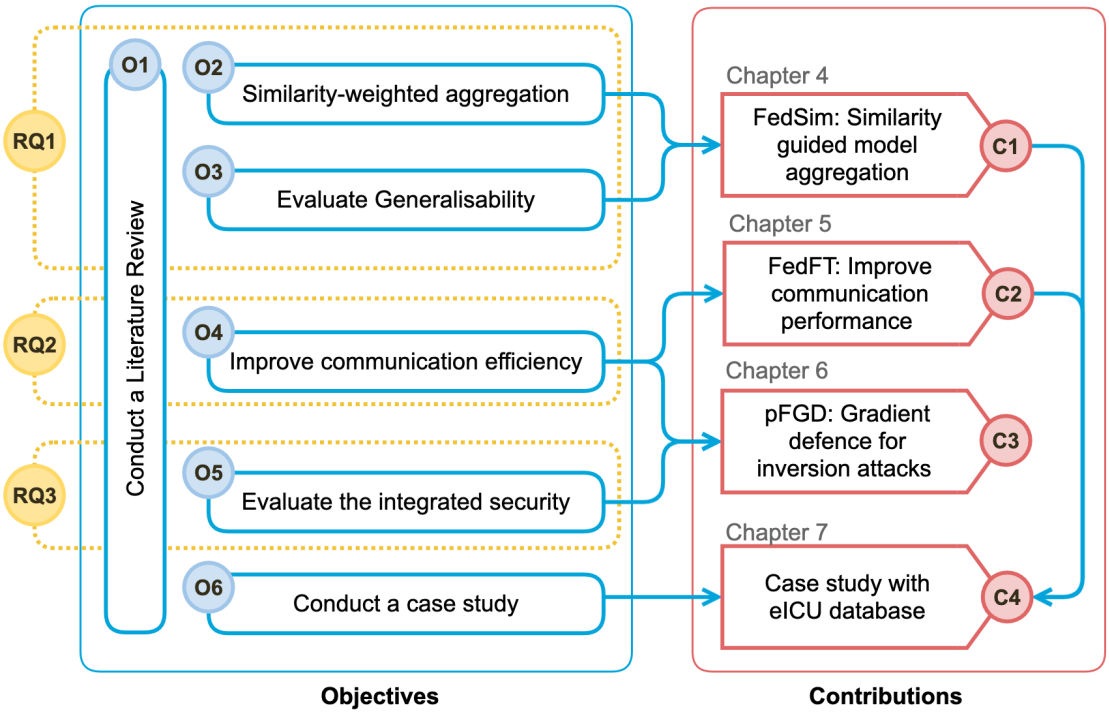

Figure 1.4: Overview of Research Questions, Objectives and Contributions

examine techniques for enhancing communication efficiency, and explore critical privacy and security concerns within FL. This chapter sets the stage for developing and evaluating the methods proposed in subsequent chapters.

Chapter 3 formalises the fundamental concepts of FL, introducing the key terminology that will be used throughout the thesis. This chapter also introduces the datasets that will be used to evaluate the proposed methods, along with a detailed description of the experimental setup. Additionally, we discuss the evaluation methodology and the rationale behind the selection of baselines.

In Chapter 4, we present a similarity-guided model aggregation strategy. This chapter explores how similarity knowledge can be inferred by comparing client gradients, rather than relying on client data, which would violate FL's privacy-preserving principles. A comprehensive evaluation of the proposed method is conducted to assess its performance.

Chapter 5 presents a novel FL methodology leveraging frequency space transformation to communicate and aggregate model parameters. The method is designed to improve communication performance in FL and this chapter evaluates its generalisability and

effectiveness in enhancing communication efficiency.

In Chapter 6, we utilise the proposed method from Chapter 5 to enhance security in FL. We evaluate its effectiveness in defending against privacy attacks, demonstrating its potential to safeguard client data.

The final contribution, Chapter 7, where we apply the proposed methods to a real-world application in the healthcare domain, chosen due to its significant privacy concerns and practical relevance. This case study explores the challenges inherent in this domain and evaluates the impact of the proposed methods.

We conclude the thesis in Chapter 8 by summarising the key contributions made throughout the thesis. We revisit the research objectives, discuss the limitations of the current work, and suggest potential directions for future research and development.

Chapter 2

Literature Review

In this chapter, we review the literature on FL, focusing on the identified challenges and research questions. We begin by introducing FL's standard aggregation methods to provide a comprehensive understanding of its internal mechanisms. Following this, we explore various aggregation techniques, comparing their performance, use cases, and limitations across specific applications. We then examine techniques to improve communication efficiency in FL, given that communication overhead is a significant concern. This review includes methods like model compression, quantisation and sparsification. Finally, we explore the privacy and security measures in FL, including an overview of attacks and defence methods.

2.1 Formalising FL Concepts

In a FL setting, the approach to handling datasets differs from traditional methods. The key difference between a dataset used in a typical ML training (i.e., centralised training) and a dataset used in FL is that in FL, the data needs to be distributed and decentralised among clients (i.e., edge devices/nodes). Each client's data is isolated for that client, giving FL the core benefit of privacy preservation. The evaluation of FL algorithms is also considered a bit different than the typical ML approaches. In this approach, each client needs to be evaluated individually on the collectively trained global model.

To describe the proposed methodologies and experimentation setup it is important to formalise the terms. The following terms are used consistently throughout our work:

- **Client**: This term refers to any device or user that participates in the FL process.

Typically, a client is an edge device, like a smartphone or a hospital's computer system that performs computations on its data and contributes to the model's training.

- **Local Training**: The process by which each client uses its own data to update the global model. This step ensures that the global model learns from diverse data sources without requiring data to be centralised.
- **Local Data**: The data possessed by a client that is utilised for local training purposes. This is also referred to as client data.
- **Local Model**: Upon completing the local training process, each client generates a local model that is then transmitted to the central server for aggregation.
- **Central Server**: A server that coordinates the FL process, receives model updates from clients, aggregates these updates to improve the global model, and then distributes the updated model back to the clients. The central server has the authority to select clients in each round and which updated local models to aggregate. A central server is also called the 'aggregator', a core server functionality to aggregate models.
- **Global Model**: The shared model that is being collaboratively trained by all participating clients. This model is updated iteratively through the aggregation of local model updates.
- **Model Aggregation**: A key step in FL where the central server combines updates received from clients to enhance the global model. This aggregation can take various forms, depending on the specific FL algorithm used.

2.2 Statistical Heterogeneity in FL

Statistical heterogeneity in FL occurs when the data distribution across different clients is inconsistent and does not follow the same sampling distribution. This situation is often referred to as non-IID data. From the early works of FL researchers have noted that FL is well-suited for the non-IID setting (McMahan et al., 2017). In practical situations non-IID data distributions are naturally expected. These differences can arise from diverse sources such as varying data collection methods, regional differences, and unique client behaviours. Statistical heterogeneity can result in biased updates, slower convergence,

and degraded overall model performance. It is crucial to develop effective methods to handle such variability.

2.2.1 Types of Statistical Heterogeneity

Recent surveys by Ye et al. (2023) and Kairouz et al. (2019) presents a comprehensive overview of the statistical heterogeneity in FL. They categorise statistical heterogeneity into four main types: label skew, feature skew, quality skew, and quantity skew. Understanding these types is crucial for developing effective strategies to mitigate their impacts. Table 2.1 presents an overview of the various types of statistical heterogeneity and provides practical examples for each type.

Table 2.1: Types of Statistical Heterogeneity in Federated Learning

Type	**Description**	**Examples**
Label Skew	When the label distributions across clients differ.	Client A has images labelled as cats and dogs, while Client B has images labelled as birds and fish. Client A labels a beach image as "relaxing," whereas Client B labels it as "boring."
Feature Skew	When the feature distributions across clients vary.	Client A's handwritten digits have thick strokes, while Client B's handwritten digits have thin strokes. Client A's dog images are indoor photos, Client B's are outdoor photos.
Quality Skew	Differences in the quality of data across clients.	Client A's X-ray images are accurately labelled, Client B's images have some incorrect labels. Client A's X-ray images are high-quality, while Client B's images are noisy and less detailed.
Quantity Skew	Imbalance in the amount of data held by different clients.	Client A has thousands of user data points, while Client B has only a few hundred.

2.2.2 Challenges Posed by Statistical Heterogeneity

Statistical heterogeneity presents several challenges in FL due to inconsistent data distributions across different clients. These challenges can significantly impact the effectiveness and efficiency of FL models. Below, we discuss some of the primary challenges:

- **Model Convergence:** In FL, the global model is updated by aggregating local models trained on diverse datasets. Non-IID data can lead to conflicting gradients, which in turn can slow down the convergence rate of the global model.

- **Model Performance and Fairness:** Since the model is trained on diverse data, it may not generalise well across all clients. This can lead to performance degradation and fairness issues, where clients with more abundant or higher-quality data unreasonably influence the global model.
- **Privacy Risks:** When a unique subset of clients shares similar data distribution, their privacy may be at risk, potentially revealing sensitive information.
- **Algorithm and Resource Complexity:** Designing FL algorithms that can effectively handle statistical heterogeneity is inherently complex. Techniques to address heterogeneity often involve additional computation and more complex coordination among clients, which can be particularly challenging in resource-constrained environments.

When creating new algorithms it is essential to consider the impacts of statistical heterogeneity and integrate strategies to address these variations. This enhances the overall performance and fairness of FL models.

2.3 Aggregation Methodologies in Federated Learning

In FL, the aggregation step is pivotal in computing the global model from the updated client models. This step is crucial for ensuring the FL algorithm's efficiency, privacy-preserving nature, and communication efficiency. However, a significant challenge in FL is statistical heterogeneity (as discussed in Section 2.2), where data distributions across clients vary. This non-IID nature of client data can cause client drift and slower convergence, negatively impacting overall model performance. To effectively address these challenges, robust aggregation methods are needed to manage statistical heterogeneity and ensure consistent training across diverse client datasets. The global model's performance largely depends on the effectiveness of the aggregation method used. Researchers have proposed various aggregation methods to achieve these goals. This section provides an overview of common aggregation techniques, reviews similarity-based approaches, evaluates these methods, and identifies gaps and opportunities for further improvement.

Figure 2.1 presents a typical aggregation process in a FL setting. The step labelled (A) in the figure represents the local update, where each client trains the global model using its local data. Step (B) in the figure represents the aggregation function, which can involve various operations such as simple averaging, weighted averaging, and median-based aggregation. Designing and developing a new aggregation function involves extensive testing

and experimentation across different scenarios.

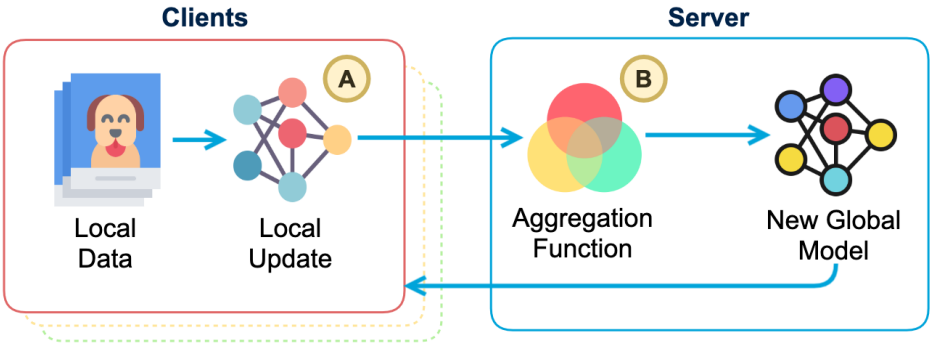

Figure 2.1: Federated Aggregation process

Although the aggregation occurs on the server, the local client updates play a crucial role in determining the effectiveness and performance of the aggregated model. Therefore, it is essential to consider approaches that modify the local training step to improve the overall aggregation.

Numerous aggregation methods tailored to specific datasets and problems can produce excellent results in their respective contexts. However, these specialised approaches often need more generalisability and may perform poorly across different scenarios or data distributions. In contrast, this research focuses on generalisable aggregation methods that can dynamically adapt to diverse and evolving data distributions across clients. The goal is to develop models that are robust and scalable in a wide range of FL environments and capable of dynamically learning and adjusting to varying levels of statistical heterogeneity.

2.3.1 Federated Averaging (FedAvg)

Federated Averaging (*FedAvg*) is the conventional FL algorithm introduced by McMahan et al. (2017). *FedAvg* marked the beginning of FL which enabled the creation of an effective global model by collaboratively training across decentralised clients. *FedAvg* aims to create an effective global model with broader coverage from the participating clients. The FL system initialises by randomly setting the weights of the global model w_0, which are then communicated to the clients. A communication round t in *FedAvg* executes the following steps:

1. A sample of K clients are selected for the round and the selected clients receive the most recent global model.

2. Each of the K client performs local updates using stochastic gradient descent (SGD) for E epochs with a batch size of B on its local training data.
3. Once the local update is completed, the updated weights will be communicated to the server.
4. The server aggregates these weights, weighted by the client k's sample size n_k (Aggregation step).

The aggregation is performed as follows:

$$w_{t+1} \leftarrow \sum_{k=1}^{K} \frac{n_k}{n} w_{t+1}^k \tag{2.1}$$

In this equation:

- K represents the clients selected in a round.
- w_{t+1}^k denotes the updated weights of client k after local training.
- n is the total sample size from all clients.
- n_k is the sample size of client k.
- w_{t+1} is the updated global model after aggregation.

Once the updated global model w_{t+1} is computed it is distributed to all the clients so they can benefit from the model refinements. *FedAvg* is evaluated by the number of communication rounds required to achieve a target accuracy. Several hyper-parameters can be adjusted to optimise performance:

1. Number of clients per round (K): varying the number of clients participating in a training round.
2. Number of local epochs (E): changing the computation capacity by varying the number of local epochs for a client
3. Local training hyper-parameters: Number of local epochs (E), batch size (B), learning rate and regularisation parameters.
4. Weight initialisation: The method used to initialise the global model weights.

Each hyper-parameter impacts the performance of *FedAvg* differently depending on the

specific application of FL. For instance, increasing the number of local epochs can make models more personalised to individual clients' data. This may potentially harm the generalisation of the global model. On the other hand, increasing the number of clients per round generally improves the global model's robustness and generalisations but can result in a longer time to convergence and higher computation costs. *FedAvg* serves as the baseline algorithm for FL. Numerous variations and enhancements are built upon it. These adaptations usually involve modifying various steps in the aggregation process to address specific challenges such as data heterogeneity, communication efficiency, and model personalisation. The complete *FedAvg* algorithm is presented in Algorithm 1.

Algorithm 1 *FedAvg* Algorithm

Require: w_0 initial global model, $\mathcal{K}$ clients
for t=1,2,.. **do**
  Select $\mathcal{S}_t$ clients where $\mathcal{S} \subset \mathcal{K}$
  for all $k \in S_t$ **do**
    $w_{t+1}^k \leftarrow$ client updates w_t using SGD
  end for
  $w_{t+1} \leftarrow \sum_{k=1}^{K} \frac{n_k}{n} w_{t+1}^k$
end for

2.3.2 FedProx

Li et al. (2018) introduced a framework called *FedProx* to address two major heterogeneity challenges in the FL setting:

1. System heterogeneity: In real-world applications, issues like clients unable to complete local training or clients might only sometimes be available.

2. Statistical heterogeneity: Occurs when clients' local training samples come from highly non-IID data distributions.

In the standard *FedAvg* algorithm, a client is required to complete the full computation round (local epochs), or else they are considered dropouts. However, the *FedProx* algorithm considers them as 'stragglers' and allow them to communicate partial updates based on the new approach. This modification helps better accommodate client variability and ensures more robust and inclusive model aggregation.

In order to handle the partial updates, *FedProx* uses an inexactness measure (γ) for each client in each round. This measure quantifies the extent of computation completed to optimise the local model. For addressing statistical heterogeneity, *FedProx* introduces a

proximal term (μ) to the local update objective. This modifies the local update objective h_k as shown in Equation 2.2, where F_k is the local objective function of client k.

$$h_k = F_k(w) + \frac{\mu}{2}\|w - w^t\|^2 \tag{2.2}$$

FedAvg is a special case of *FedProx* when $\mu = 0$, the local update is performed using SGD, and γ is a constant for all clients. *FedProx* has demonstrated significant stabilisation in training relative to *FedAvg* and an accuracy improvement by 22% when the system is highly heterogeneous. The effectiveness of *FedProx* was empirically validated using four real federated datasets and four synthetic datasets, showcasing its improvements over *FedAvg* in heterogeneous settings. The drawbacks of *FedProx* include its sensitivity to the proximal term choice, which can be challenging to optimise. If the proximal term is not set well, *FedProx* may struggle to reduce client drift, leading to suboptimal convergence. *FedProx* is more beneficial when data distributions across clients are significantly different, but in cases with mild heterogeneity, it may not offer substantial improvements over *FedAvg*. In summary, *FedAvg* is suitable for relatively homogeneous and stable environments. *FedProx* extends its applicability to more complex and realistic scenarios by addressing critical heterogeneity and client variability issues.

2.3.3 Adaptive Federated Optimisation (FedAdam)

Research by Reddi et al. (2020) propose the federated implementation of adaptive optimisation techniques to address the lack of adaptivity in FL. They introduce an approach that differentiates between client and server optimisation techniques, unlike the uniform use of SGD in *FedAvg*. This approach includes using adaptive optimisers such as Adagrad (Duchi et al., 2011), Adam (Kingma and Ba, 2014), and Yogi (Zaheer et al., 2018), which have shown success in non-federated settings by effectively managing gradient updates. This approach is commonly referred to as FedAdam by the research community, but it also includes variations such as FedAdagrad and FedYogi.

FedAdam specialise FEDOPT (framework for federated optimisation) for settings where SERVEROPT (server-side optimisation) is an adaptive optimisation method (one of Adagrad, Yogi, or Adam) and CLIENTOPT (client-side optimisation) is SGD. By using adaptive methods on the server to aggregate client updates and SGD on the clients for local training, they ensure that their methods maintain the exact communication cost as *FedAvg* and work efficiently in cross-device settings. This approach is more complex

because it needs to manage the server's adaptive states, which adds extra computational overhead. While this approach benefits certain implementations like sparse-gradient tasks such as Natural Language Processing, it presents a challenge in making it an accessible solution in FL. The increased server-side complexity and resource requirements may limit its practicality, especially in environments where computational resources are limited.

2.3.4 Federated Normalised Averaging (FedNova)

FedNova is an advanced FL algorithm designed to address the challenges of client heterogeneity in FL environments. This method was introduced by Wang et al. (2020) to ensure fair contributions from clients and improve the robustness and stability of the global model. The core idea is to adjust the weight of each client's update according to their local computational effort, ensuring that all clients contribute fairly to the global model, regardless of their computational capabilities. After local training, each client's update is normalised by the number of local steps performed. This is done by dividing the client's gradient by the number of local steps, ensuring that updates are proportional to the computational effort. Then the server aggregates the normalised updates to form the new global model. This weighted averaging ensures that each client's contribution is balanced, improving the robustness of the global model.

Figure 2.2 compares the global model updates between FedNova and *FedAvg*. Here, x represents the model parameter vectors at different stages of the training process. The green dot represents FedNova, and the blue dot represents *FedAvg*. As observed, *FedAvg* has a natural bias towards client updates that performed more training iterations. In contrast, FedNova ensures a more balanced aggregation by normalising the updates based on the number of local steps performed.

FedNova introduces additional complexity and computational overhead with the normalisation process. Precise tracking of the local updates can be challenging. Furthermore, the proposed approach requires careful optimisation of a wide range of hyper-parameters, which can add extra effort when applying it. The generalisability of FedNova is also a concern, as the experiments presented in the paper are relatively limited in scope, primarily focusing on synthetic datasets with controlled heterogeneity and the non-IID partitioned real-world dataset CIFAR-10.

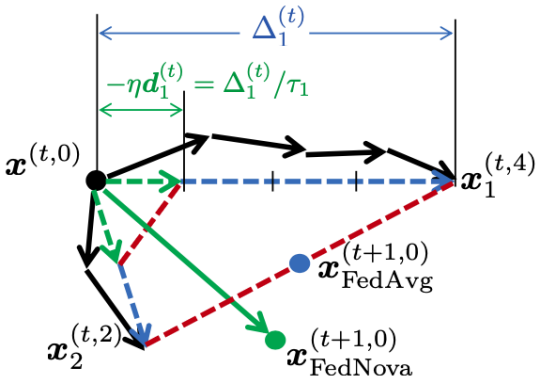

Figure 2.2: Comparison between FedNova (Green dot) and *FedAvg* (Blue dot) in the parameter space. (Figure taken form Wang et al. (2020))

2.3.5 SCAFFOLD

SCAFFOLD (Stochastic Controlled Averaging for Federated Learning) introduced by Karimireddy et al. (2020) specifically addresses the 'client-drift' problem when the data is non-IID. Client-drift occurs when local model updates from different clients diverge significantly due to differences in their local data distributions. Figure 2.3 demonstrates the mechanism of client-drift in a FL scenario with two clients on *FedAvg*. In Figure 2.3, local updates have drifted towards their optima (x_1^* and x_2^*), and the aggregated global model moves towards the average of the client optima rather than the true global optimum (x^*). This divergence due to client-drift leads to slower and more unstable convergence of the global model.

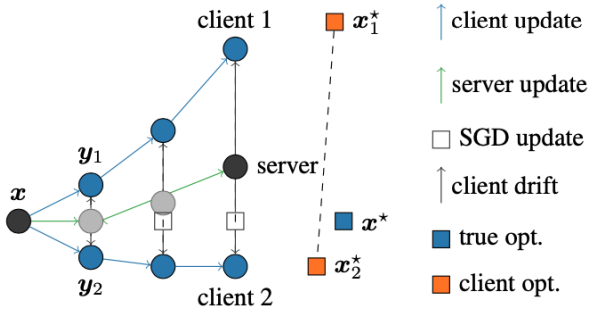

Figure 2.3: Client-drift in FL (Figure taken form Karimireddy et al. (2020))

The SCAFFOLD algorithm estimates the global model's update direction and each client

using control variates. These control variates estimate the drift and adjust the local gradient updates accordingly. The difference between the global model's control variate and each client's control variate is used to correct the local updates. In each communication round, the server sends the global model parameters and control variates to a subset of clients. These clients perform local updates using their data while adjusting for the estimated drift via their control variates. After local training, clients update their control variates and return their model updates to the server. The server then aggregates these updates to refine the global model and updates its control variate accordingly. By reducing the variance caused by client-drift, SCAFFOLD improves convergence rates and requires fewer communication rounds, making it efficient and robust against data heterogeneity.

Compared with FedProx (discussed in Section 2.3.2), SCAFFOLD is generally better in extremely non-IID scenarios. However, FedProx is considered more scalable and simpler when handling heterogeneous data. For extremely heterogeneous data, SCAFFOLD can be particularly beneficial.

2.3.6 FedGroup

Duan et al. (2020) presents two novel FL algorithms, FedGroup and FedGroupProx, which group clients based on the similarity of their parameter updates. Clients are clustered into groups where each group contains a model tailored to the clients in that group. The FedGroup algorithm includes three levels of model aggregation: intra-group, inter-group, and global aggregation. Intra-group aggregation occurs within a single group, where the group model is broadcast to all clients in that group. This aggregation uses *FedAvg* to combine the client updates within the group. After the training round, group-level models are aggregated using a defined weight called inter-group aggregation. Although FedGroup maintains a global model similar to typical FL algorithms, this global model is primarily used to initialise models for new clients. A visual demonstration of the complete FedGroup process is shown in Figure 2.4.

The second proposed approach is FedGrouProx, which incorporates a proximal term as discussed in the works of *FedProx* (Li et al., 2018). Using the proximal term, the training has shown more stability, although it did not significantly improve accuracy. FedGroup and FedGrouProx have improved FL by dividing the global optimisation problem into groups of sub-optimisation tasks.

It is worth noting that there are several challenges with the proposed approach. There

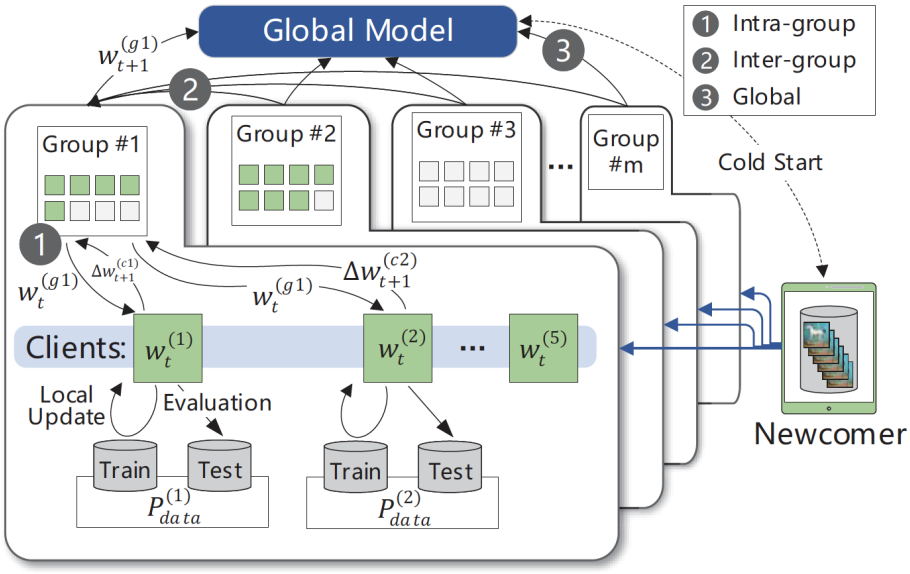

Figure 2.4: An overview of the FedGroup Algorithm (Figure taken form Duan et al. (2020))

is a substantial computational overhead in maintaining the groups and their states, and the framework involves multiple levels of aggregation based on group distributions, which can add complexity. Another critical weakness is the cold start problem. At the beginning of the training, new clients must be assigned to a group based on their optimisation direction. This initial assignment is static, meaning clients remain in the same group throughout training. This static grouping can limit the system's flexibility and adaptability, particularly when new clients join the training later or when the data distribution changes over time.

2.3.7 Summary of Aggregation Methods

Table 2.2 presents a brief overview of the algorithms discussed in this section. We focus on dynamic and generalisable training methods rather than specialised or personalised solutions. Unlike approaches that tailor the model to individual clients, the selected baselines aim to address the challenges of adapting to diverse data distributions and evolving conditions across clients. The selection of aggregation methods for the review *FedAvg*, *FedProx*, FedAdam, FedNova, SCAFFOLD and FedGroup was based on their broad applicability and coverage of key challenges in FL.

Table 2.2: Comparison of FL aggregation methods

FL Algorithm	**Aggregation Method**	**Objective**	**Review**
FedAvg (McMahan et al., 2017)	Weighted aggregation by sample size	Develop a straightforward, efficient global model that generalises well across various client data	Advantages: Simple, effective for homogeneous data. Disadvantages: Struggles with non-IID data, slower convergence.
FedProx (Li et al., 2018)	Weighted aggregation with a proximal term	Address system and statistical heterogeneity	Advantages: Better stability, handles partial updates, handles non-IID data well. Disadvantages: Increased complexity, additional computational overhead.
FedNova (Wang et al., 2020)	Normalised aggregation based on local steps	Ensure fair contributions from all clients	Advantages: Improves robustness and fairness. Disadvantages: Adds complexity due to normalisation overhead, additional hyper-parameters to optimise
SCAFFOLD (Karimireddy et al., 2020)	Variance reduction using control variates	Correct client drift and improve convergence	Advantages: High stability and performance with non-IID data. Disadvantages: More complex to implement, computationally intensive.
FedGroup (Duan et al., 2020)	Multi-level aggregation (intra-group, inter-group, global)	Improve group models by creating sub-optimisation problems	Advantages: Utilises client similarity knowledge. Disadvantages: Inefficient client handling with static groups, less flexibility to extend.

These methods were chosen because they represent a range of approaches: traditional (*FedAvg*), adaptive (FedAdam), non-IID handling (*FedProx*, SCAFFOLD), more advanced

techniques such as normalised averaging (FedNova) and similarity-based (FedGroup). This combination ensures that the evaluation covers various aspects of FL, including general aggregation techniques and approaches designed to address non-IID data. Each algorithm brings unique strengths and addresses fundamental challenges in FL, providing insights into the trade-offs and considerations necessary for effective implementation.

2.4 Similarity-Based Aggregation in FL

Different FL algorithms focus on achieving different goals. The FL literature is limited in its work on similarity-guided aggregation techniques. We have identified two methods that harness the effect of similarity among clients. Each of them has identified a different goal, and they need the goal to guide global model aggregation.

2.4.1 Clustered Federated Learning (CFL)

CFL proposed by Sattler et al. (2020a) is a Federated Multi-Task Learning framework that groups clients into clusters with similar data distributions. CFL is a post-processing algorithm that begins after the training phase of FL is completed and the global model is converged. CFL focuses on creating specialised models for a set of clients that can benefit better from the similarity of the data distribution. To identify the similarity among client data distributions, they propose the cosine similarity of weight updates. The algorithm computes bi-partitions (branching) of the global model until there is a specialised model for each branch, which may contain one or more clients.

The CFL method involves the following steps:

- Perform standard FL to obtain a stationary solution (i.e. converged).
- Check Stopping Criteria: Evaluate if all clients are sufficiently close to a stationary solution of their local risk functions. If they are, the process terminates here.
- Cluster Clients: Calculate the pairwise cosine similarities between the clients' gradient updates. Use these similarities to cluster the clients into groups with similar data distributions.
- Recursive Clustering: Repeat the above steps for each of the newly formed clusters, including performing standard FL within each cluster, until all clients in each cluster are close to their local stationary solutions.

While CFL can significantly improve model performance for specific tasks, it is a post-processing algorithm. It only begins after the standard FL process is completed and the global model has converged. This can limit its applicability in real-world scenarios where adaptations to new data patterns and continuous improvement of the global model are needed. Additionally, requiring a fully trained global model before initiating clustering can result in higher computational costs and longer training times. Furthermore, there can be many tasks where the global model does not converge to a satisfactory level, making CFL less effective.

2.4.2 FedGroup

FedGroup discussed in Section 2.3.6 considers the similarity knowledge of clients to create client groups. The grouping of clients are based on the similarities between the local optimisation directions. To measure the similarity between gradient updates, the authors have presented an approach named *ternary cosine similarity*. The similarity algorithm starts by pre-training clients for one epoch on the initial global model. Then, they are assigned to three directional vectors, and the local optimisation direction of clients is measured using the normalised ternary cosine similarity function for the pre-training updated model.

A clustering approach is proposed to initialise the groups. The K-Means++ algorithm clusters the clients based on the similarity matrix generated by the pre-training process. In this approach, each client is assigned to only one group, and the grouping occurs only once, making the strategy static and unchanging across rounds. This static grouping strategy can lead to scenarios where some groups have no clients (group cold start) and some clients are not assigned to any group (client cold start). All calculations and clustering occur in a single communication round. After addressing the cold-start problem, training in FedGroup begins by initialising all group models with the initial global model weights. A random subset of clients is then selected, and each group trains the model in a FL manner, resulting in a temporary model aggregated within the group (intra-group aggregation). Finally, the global model is updated by aggregating all the group models.

2.4.3 Summary of Similarity-Based Approaches

Sattler et al. (2020a) present Clustered Federated Learning (CFL), a post-processing step for FL that clusters similar clients. CFL's primary objective is to develop specialised models for individual client clusters rather than enhance the global model or the

aggregation process. CFL utilises the learned global model to create models tailored to specific client clusters, thereby improving performance for those clusters. However, CFL does not focus on improving the global model's performance. As a post-processing step, it inherently introduces computational overhead, making it less suitable for unknown or dynamic systems where adaptability and efficiency are crucial. On the other hand, the FedGroup algorithm (Duan et al., 2020) proposes dividing the optimisation problem into multiple sub-optimisation problems to enhance each group's model. This method aims to improve the performance of group models instead of the global model. In FedGroup, clustering is performed once at the beginning, with clients locked into groups for the system's duration. This static clustering approach can be problematic in dynamic environments where client data can vary over time.

CFL and FedGroup highlight the potential benefits of utilising client similarity in FL. However, they also share limitations, particularly in their static approaches to clustering and specialisation, which do not account for the evolving nature of client data distributions. The static assignment of clients in FedGroup and the post-processing nature of CFL indicate a significant gap in the literature: the lack of dynamic, generalisable, similarity-guided FL approaches that continuously adapt to the clients and data. CFL and FedGroup demonstrate the benefits of leveraging client similarity in FL, inspiring our work. However, they were not included in the baseline comparisons because their approaches emphasise specialisation and personalisation of FL models rather than generalisation, which is the focus of our methods. There is a clear need for further research into dynamic similarity-guided aggregation methods in FL. This gap presents an opportunity for developing novel algorithms that leverage real-time similarity metrics to guide both local training and global aggregation processes, potentially leading to significant advancements in FL.

2.5 Measuring Statistical Heterogeneity

We have identified several metrics for measuring datasets' non-IIDness, but only a few are applicable in the FL setting where data is distributed among clients.

2.5.1 Non-IID Index (NI)

He et al. (2019) introduced the Non-IID Index, which uses a feature extractor to compare data across each class of the dataset. Data related to each class is processed through the feature extractor and the results are used to calculate an overall index for the dataset

indicating the degree of distribution shift. However, due to the limitations of the FL setting and privacy concerns, the Non-IID Index does not apply to datasets used in FL.

2.5.2 Client-wise non-IID Index (CNI)

Adopting the Non-IID Index (He et al., 2019) introduced a new measure to capture the degree of non-IID in the FL setting by Li et al. (2020). The new measure named Client-wise Non-IID Index (CNI) examines the degree of distribution shift for each client, unlike the Non-IID Index (NI) which focuses on each class. CNI captures three factors: feature distribution skew, label distribution skew, and quantity skew. CNI uses a trained encoder (VGG16 trained on ImageNet) to get an encoding for each client class. CNI for each client is calculated at the client class level with respect to other clients, and an overall CNI value is calculated by adding up all clients. While CNI is is particularly designed for visual data, this approach reveals a gap when addressing non-image data or more diverse data types in FL.

2.5.3 One-Pass Distribution Sketch

A more recent approach introduced by Liu et al. (2024) is the One-Pass Distribution Sketch, which aims to measure data heterogeneity across clients efficiently. This method uses a single pass over the client data to create a distribution sketch that efficiently represents the client data distribution in terms of time and memory. The distance between the two distribution sketches reflects the divergence between their corresponding distributions. This method improves client selection in FL training and addresses the cold start problem for new clients with unlabelled data.

In summary, while the Client-wise Non-IID Index (CNI) remains a valuable tool for measuring non-IIDness in FL settings, the One-Pass Distribution Sketch offers a promising new approach that is efficient and scalable for diverse data types and large-scale FL applications. However, two challenges have been identified with this approach. Firstly, there is an impact on privacy when class-level knowledge of a client is shared with the server. Secondly, using pre-trained encoders can present challenges when comparing different data types (e.g., image and text), as the encoders may perform inconsistently across these varied data types.

2.6 Communication Efficient FL

Communication cost is a primary bottleneck for FL systems (Aledhari et al., 2020; Zhang et al., 2021; Zhao et al., 2023). This is due to the high-frequency communication of model parameters between clients and the server, with the number of clients potentially reaching into the millions (Bonawitz et al., 2019; Niu et al., 2020). Depending on their complexity, these neural models can vary significantly in size, ranging from a few kilobytes to several hundred megabytes. The communication bottleneck in FL systems can lead to unreliability and limit their ability to scale up and meet increasing demands. In a realistic setting, clients with poor network connections or resource limitations can further hinder the FL system's performance. Methodologies like *FedProx* (Li et al., 2018) have addressed this issue by handling partial updates from clients with limited connectivity.

Approaches in the literature aimed at mitigating this communication bottleneck can be broadly categorised into two groups (Konečnỳ et al., 2016):

Structured updates: Restrict local training and communication to a reduced space (e.g., focusing on fewer model parameters). This method can be particularly beneficial in scenarios where bandwidth is limited or the number of participating clients is very high.

Sketched updates: Perform local updates on the entire model and compress it for communication. This method allows the complete model to be trained locally while significantly reducing the communication overhead by compressing the data that must be transmitted.

Both approaches have their advantages and disadvantages. However, communication efficiency in updates performed through sketches is often more adaptable and can be easily integrated with existing FL techniques. In this thesis we explore the sketched update approach to enable local client updates to be performed in either tensor or compressed frequency space, with communication and federated aggregation performed in compressed frequency space.

2.6.1 Compression for Communication Efficiency

Compression techniques for FL models in tensor space include sub-sampling, which reduces spatial resolution, and probabilistic quantisation, which reduces precision (Konečnỳ et al., 2016). The work presented by Sattler et al. (2020b) compresses model parameters using Golomb Coding and reduces model complexity through quantisation. Golomb

Coding is a compression method that doesn't lose any data. However, because of its non-linear nature, extra transformations are needed at the server. This means there are additional steps to rebuild the data, because the compressed space doesn't allow for combined gathering. Additionally, the quantisation is tightly coupled with the client's local update. Which makes the method by Sattler et al. (2020b) less adaptable to existing FL methodologies.

Research by Dai et al. (2019), also aim for communication efficiency through lossless compression and quantisation, but they apply compression on the gradients rather than model parameters. Dai et al. (2019) share the same limitation of adaptability for FL as discussed for Sattler et al. (2020b). Figure 2.5 illustrates applying a standard compression algorithm in FL. This approach involves multiple stages of model compression and decompression to reduce the communication overhead between the server and the clients. The increased computational demand can make the approach less practical, especially for resource-constrained environments or large-scale deployments with millions of clients. Next, we explore alternative solutions using frequency space transformations as a compression mode.

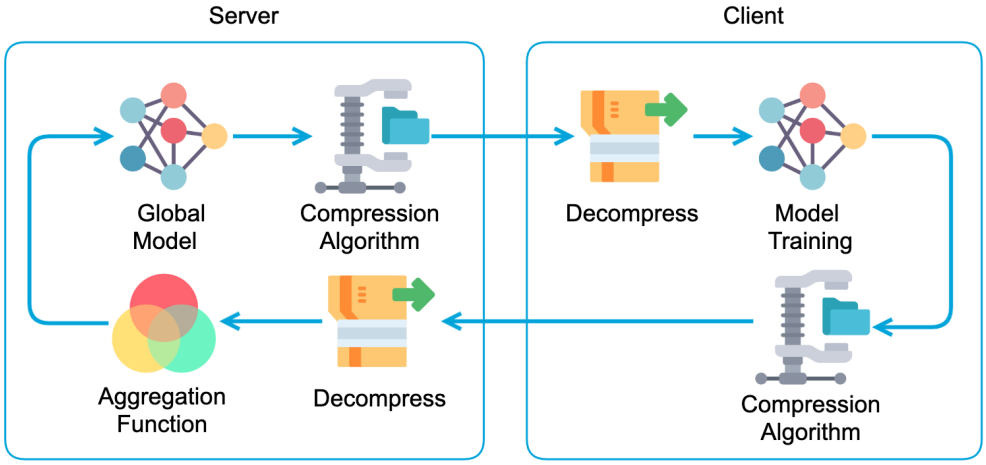

Figure 2.5: Process of applying standard compression algorithms

Frequency space transformation techniques have been employed for data compression for many years; examples include Discrete Cosine Transformation (DCT) (Ahmed et al., 1974), Discrete Fourier Transform (DFT) (Winograd, 1978), Fast Fourier Transform (FFT) (Cooley and Tukey, 1965), and Principal Component Analysis (PCA) (Pearson, 1901). However, DCT is one of the most widely used techniques due to its favourable properties such as computational efficiency and the ability to compactly represent the

energy content of a signal (Rao and Yip, 2014; Strang, 1999). As a result, DCT has been widely adopted for image and video compression applications. Data compression in the frequency space is accomplished by pruning (i.e. trimming) or quantising the least significant information (detailed in Section 2.6.2). In the field of ML, researchers have explored using the DCT for compressing models by transforming data into the frequency space (Dimililer, 2022; Robinson and Kecman, 2003).

To the best of our knowledge, the full potential of DCT has yet to be fully exploited in the context of FL to improve communication efficiency through model compression.

2.6.2 Pruning and Quantisation for Communication Efficiency

Pruning and quantisation are two effective methods to optimise and simplify ML models (Jiang et al., 2022; Liu et al., 2018; Zhao et al., 2023). Pruning reduces model size by removing less significant model parameters. A pruning mask can be learned as part of model optimisation (Liu et al., 2018) or determined using prior knowledge in static pruning. Quantisation decreases the precision of model parameters by using fewer bits, resulting in a smaller model size. Both techniques are crucial for deploying ML models in resource-constrained environments.

In FL, some quantisation techniques include FedPAQ (Reisizadeh et al., 2020), which uses periodic averaging, and FedPara (Hyeon-Woo et al., 2021), which employs low-rank Hadamard product parameterisation to reduce the precision of model weights. Prakash et al. (2022) adopt a similar method to Liu et al. (2018) in FL by learning a pruning mask during client training to reduce upstream communication costs. In contrast, Jiang et al. (2022) apply global and client model pruning using static pruning masks to reduce overall communication costs, requiring an extra step at the server where a single gradient descent step is taken on the global model.

2.7 Security in FL

Security is a well-discussed area in ML and FL introduces additional complexities, particularly in preserving privacy. Due to the decentralised nature of data storage and processing, ensuring the security of client data and model updates is essential to maintaining the integrity and trustworthiness of FL systems. This section discusses various security issues and techniques to improve security in FL.

2.7.1 Security Challenges and Threat Actors

ML systems have been identified to be at risk from various attacks, both non-ML-specific and ML-specific (Al-Rubaie and Chang, 2019). These attacks mainly target personal data, learning models and communication protocols. Security measures are required to prevent such attacks. In a distributed system like FL, there are numerous data access points (i.e. attack surfaces) that need to be secured. With multiple actors involved (e.g. clients, server), there is a high security risk. A survey by Kairouz et al. (2019) discusses the threat actors and models related to an FL setting as follows:

- Clients - Malicious clients can inspect the communication data and tamper with the local training process, as they have full control over their devices.
- Server - The server orchestrates the entire FL system and communicates with clients. A malicious server can tamper with the aggregation step and compute a malicious model. The server can read the weights/gradients sent from clients, so security needs to be tightened (i.e. curios server).
- Deployed models - Once a model is trained and deployed to clients, it becomes vulnerable to inversion attacks that extract original data from the model's outputs. Such attacks can be executed from outside the FL setting at the network level by an eavesdropper. These attacks can reveal sensitive information that the model has learned from the training data.

2.7.2 Threat Surfaces

The threat surfaces are more exposed in an FL setting than in traditional ML settings. The network of clients and the communication layer in FL represent significant threat surfaces (Jere et al., 2020). Table 2.3 presents the potential threat surfaces in an FL system. The table describes the threat surface, threat level and likelihood of attack. Organisations can tailor security measures to each identified threat surface by detailing specific vulnerabilities. Using a threat matrix such as "Adversarial Threat Landscape for Artificial-Intelligence Systems" (ATLAS)[1], these threats can be explored depending on the impact for each FL application.

Figure 2.6 illustrates the threat surfaces in a typical FL setting. This visual representation highlights the various challenges and potential threat actors that can compromise system performance.

[1]https://github.com/mitre/advmlthreatmatrix

Table 2.3: Overview of Attack Surfaces in FL

Threat Surface	**Description**	**Threat Level**	**Attack Likelihood**
Local Data	Vulnerable to data tampering/poisoning	High	High
Local Update	Model manipulation	High	High
Model Inference	Expose client training data using model inference	High	Medium
Network Layer	Susceptible to eavesdropping, man-in-the-middle attacks and data tampering during communication	High	High
Server	A curios server can alter global model and local updates	High	Low
API Endpoints	Exposed interfaces that can be exploited for unauthorised access	Medium	Medium
Monitoring	Having a system monitoring/logging system could expose user behaviour	Low	Low
Firmware	Due to FL's distributed nature, outdated/-malicious firmware on clients can be vulnerable	Medium	Low

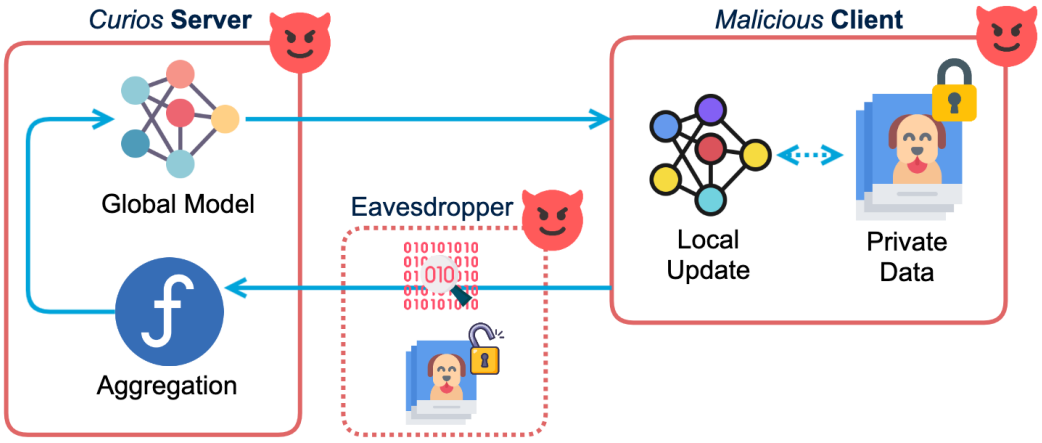

Figure 2.6: Illustration of attack surfaces and potential threats in FL

2.8 Attacks in FL Setting

As described in Section 2.7.2, the communication layer and the network of participating clients are particularly vulnerable. Attackers can exploit these vulnerabilities to compromise the integrity, confidentiality, and availability of the FL system. A taxonomy presented by Jere et al. (2020) organises attacks in FL settings into two types:

- Model Performance Attacks: These attacks occur during the training phase using poisoning techniques. By manipulating the model or local data, it is possible to degrade the model's overall performance.
- Privacy Attacks: These attacks exploit the model parameters to extract sensitive information from the training data. Privacy attacks in FL are a widely researched area due to data exposure's significant impact and risk.

To visually represent the types of attacks, Figure 2.7 provides an overview of the taxonomy of attacks. In this thesis, we explore privacy attacks and gradient inversion attacks, which are highlighted in yellow in Figure 2.7. Privacy attacks present significant risks that could undermine the fundamental purpose of FL to protect client data privacy. Among these, gradient inversion attacks are particularly concerning due to their high risk and likelihood of occurrence, making them a critical area of focus for ensuring the security of FL systems.

2.8.1 Privacy Attacks

Privacy attacks in FL target the confidentiality of client data. These attacks exploit the private information in model updates to infer sensitive data or compromise clients' privacy. FL's decentralised nature and the need for frequent communication between clients and the central server make it easy for various privacy attacks. Privacy attacks can be further categorised as gradient inversion, membership inference, and generative adversarial network (GAN) reconstruction attacks. Gradient inversion attacks have demonstrated the capability to reconstruct the classes and individual data instances using the communicated client gradients Wei et al. (2020); Zhao et al. (2020); Zhu et al. (2019). Membership inference attacks aim to find if a data instance was included in the training data and determine its class Melis et al. (2019). GAN reconstruction attacks are similar to gradient inversion attacks but capable of generative synthetic data instances representative of the client training data Hitaj et al. (2017).

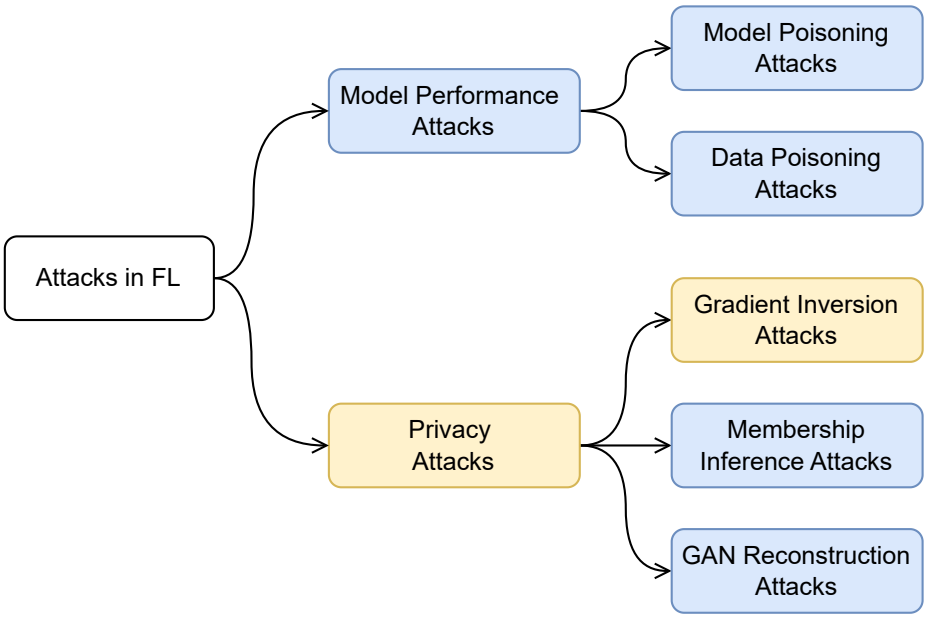

Figure 2.7: Taxonomy of attacks in Federated Learning

Gradient Inversion Attacks in FL

A recent survey Zhang et al. (2022a) proposed a taxonomy for gradient inversion attacks characterising into two paradigms: iteration-based and recursion-based attacks.

Iteration-Based Attacks start by creating random (dummy) data and labels. The gradients can be optimised for data recovery by performing forward and backwards propagation iteratively. The reconstruction of private data is viewed as an iterative process using gradient descent. Private data can be retrieved by minimising the difference between the original and generated gradients. Figure 2.8 presents an illustration of a typical iteration-based gradient inversion attack.

There is an increasing number of gradient inversion attacks that use iteration-based techniques. Some of the commonly used and studied methods include:

- Deep Leakage from Gradient (DLG) (Zhu et al., 2019): Recovers training data by iteratively updating dummy data to minimize the difference between the computed gradients of the dummy data and the true gradients.
- Improved-DLG (iDLG) (Zhao et al., 2020): An enhanced version of DLG, it improves the efficiency and accuracy of data recovery. Unlike DLG, iDLG consistently

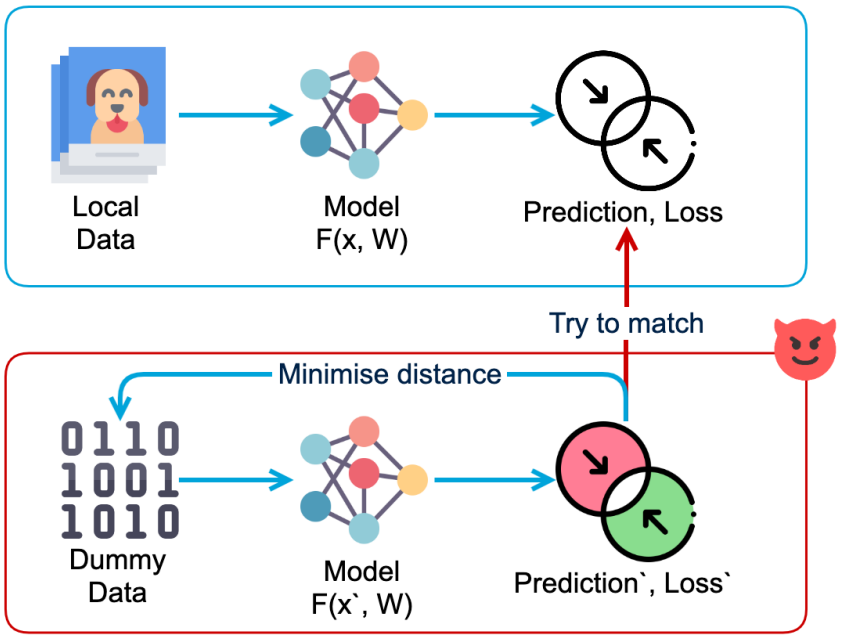

Figure 2.8: Illustration of an iteration-based gradient inversion attack.

discovers the ground truth labels.

- Client Privacy Leakage (CPL) (Wei et al., 2020): Feature reconstruction attack using a reconstruction learning algorithm.
- Inverting Gradients (Geiping et al., 2020): A broader approach that applies similar iterative optimisation techniques to invert gradients and reconstruct data.

Techniques such as differential privacy, secure multi-party computation, and encryption mechanisms are being actively researched to defend against these threats (Yang et al., 2023). Section 2.8.2 provides more details on these defence mechanisms.

The ease of implementation and the ability to generalise iteration-based gradient inversion attacks pose a significant threat to FL's privacy-preserved nature. These attacks are particularly concerning due to their adaptability and higher risk of exposing client privacy.

Recursion-Based Attacks recover the input of each layer recursively by solving an optimisation problem that minimises the error layer by layer. These attacks do not

require the initialisation or generation of dummy inputs and leverage the linear relationships between inputs, model parameters, and gradients. There are several approaches to recursion-based attacks in FL systems. One notable method is R-GAP (Recursive Gradient Attack on Privacy) (Zhu and Blaschko, 2020), which combines forward and backward propagation to formulate the recovery problem as solving a system of linear equations. R-GAP exploits the intrinsic relationships between feature maps, their gradients, and model weights to reconstruct the original data. This method is particularly effective in scenarios where the structure and parameters of the global model are known, allowing for precise calculation and recovery of the input data. The R-GAP attack can fully recover private data under specific conditions, outperforming traditional optimisation-based attack methods in certain scenarios. The COPA (Combined Optimisation Privacy Attack) method was proposed based on R-GAP principles (Chen and Campbell, 2021). COPA uses a fully connected and convolutional layer to reconstruct the input data.

Unlike iteration-based attacks, recursion-based attacks require precise knowledge of the model's structure and parameters. Such knowledge may be limited in real-world scenarios. Recursion-based attacks are often tailored to specific model architectures and configurations limiting their generalisability. This specificity and limited applicability reduce the appeal of recursion-based attacks for broad research exploration compared to iteration-based gradient inversion attacks.

2.8.2 Common Defence Mechanisms

Building defence mechanisms to mitigate the previously discussed attacks is an open problem in the FL domain. Various proposed methods aim to make FL more secure and private. Generally, when a defence mechanism is applied to any threat surfaces or actors, its effect cascades to other components. Surveys conducted by Jere et al. (2020), Lyu et al. (2022) and Rodríguez-Barroso et al. (2023) highlight the most developed and practical defence mechanisms in FL. This section will discuss related defence mechanisms for gradient inversion attacks.

Differential Privacy

Differential Privacy (DP) (Dwork, 2006) is an extensively studied defence mechanism due to its generalisability and strong privacy guarantees. Initially designed for single database scenarios, DP ensures that the results of queries have enough noise to protect individual data privacy. This improves upon traditional data anonymisation techniques. DP introduces controlled randomness into the results of queries, making it challenging to infer

information about any specific individual. DP's privacy guarantees are mathematically rigorous and can be adjusted to provide the desired level of protection.

DP can be applied at different stages of the FL process. Generally, DP is applied to gradient updates in FL as they pose a high risk when communicated (El Ouadrhiri and Abdelhadi, 2022; Sabater et al., 2020). DP adds random noise to the gradient updates to obscure the details of individual data points. The noise is typically drawn from a Gaussian or Laplacian distribution. This noise makes it challenging for attackers to reconstruct the original data or infer membership, providing a robust defence against various attacks. This approach is known as Local Differential Privacy (LDP), where the client does not trust the server or the network layer (Kasiviswanathan et al., 2011; Warner, 1965). Another approach applies DP to the global model called Central Differential Privacy (CDP) (Dwork et al., 2006). In FL, this approach is referred to as Distributed Differential Privacy (Cheu et al., 2019). CDP ensures that the server only has access to a differentially private model, minimising the risk of exposure.

One practical implication of applying DP in an FL setting is its trade-off with model accuracy. Adding random noise can degrade accuracy and too much noise can significantly impact the model's performance. Another challenge is finding the right balance for the privacy budget and managing the additional complexity.

Homomorphic Encryption

Homomorphic Encryption (HE) is a powerful cryptographic technique that enables computations to be performed directly on encrypted data without decrypting it. This allows operations such as addition, multiplication, and other mathematical functions to be carried out on encrypted space with the results remaining in encrypted form. Once decrypted, the results are identical to those that would have been obtained if the operations had been performed on the original, unencrypted data.

HE is a promising and practical defence mechanism for FL and has been applied in various FL settings (Fang and Qian, 2021; Jin et al., 2024; Zhang et al., 2020). The primary advantage of HE is its ability to preserve data privacy while allowing complex computations, making it highly suitable for privacy-sensitive applications in FL. However, a significant drawback of HE is its computational intensity. Encrypting and performing operations on encrypted data requires considerably more computational resources compared to standard encryption methods. Despite its potential, the computational overhead associated with HE can restrict its widespread adoption in FL.

The concept of HE is particularly appealing for performing mathematical operations in the encrypted domain, which aligns well with the privacy-preserving goals of FL. Our research will further explore this approach to assess its practicality and effectiveness.

Gradient Compression

As an alternative to DP, introducing noise to vulnerable components in the FL setting, such as the locally updated model or the global model, can enhance privacy. Gradient compression prunes small (insignificant) gradients to zeros, making it more difficult for adversaries to match gradients during gradient inversion attacks (Lin et al., 2017; Tagliavini et al., 2017). Similarly, Zhang and Wang (2021) proposes a technique called random sketching, which is applied to client gradients to defend against privacy attacks.

Using gradient compression techniques to defend against gradient inversion attacks is both efficient and practical (Zhu et al., 2019). Unlike methods that involve substantial computation and performance loss, compression techniques offer a better balance in the FL setting. This thesis further explores the potential of these techniques for enhancing security in FL.

2.9 Conclusions from the Literature

We have gathered essential findings from the literature, focusing on the key research elements that have influenced the methods discussed in Chapters 4, 5 and 6. We began by examining various aggregation methods in FL, tracing the growth from foundational approaches like federated averaging to more advanced techniques such as *FedProx*. The key observation was the need for more generalisability across existing aggregation methods, many of which are tailored to specific datasets, domains or architectures. This limitation highlighted the need for more adaptable and reusable aggregation strategies, which motivated the development of the generalisable methods presented in this thesis. Furthermore, our literature review identified a research gap in similarity-based aggregation methods. Despite their potential, there is a limited exploration of how inter-client similarities can be leveraged to improve model aggregation in FL. Chapter 4 addresses this gap by introducing a novel similarity-guided model aggregation method, contributing a fresh perspective to the field.

Another critical finding was the significant impact of statistical heterogeneity in FL. We explored various types of heterogeneity and methods for its measurement, identifying a

lack of privacy-preserving techniques for quantifying non-IIDness. Drawing inspiration from the non-IID index (He et al., 2019), we developed a privacy-preserving measure for non-IIDness, which is presented in Chapter 4.

In addition, we reviewed methods aimed at improving FL communication efficiency. We identified compression and pruning techniques as promising solutions to the communication bottleneck in FL. Chapter 5 introduces *FedFT*, a novel approach that leverages frequency space transformation and pruning to enhance communication performance while maintaining model accuracy.

Finally, our review of potential security threats in FL underscored the vulnerability of FL systems to privacy attacks. In particular, gradient inversion attacks pose a high risk and are relatively easy to execute. This concern drove us to integrate security considerations into our proposed methods. Chapter 6 evaluates the effectiveness of *FedFT* in defending against gradient inversion attacks and provides a comprehensive assessment of its security benefits.

Chapter 3

Datasets and Evaluation

This chapter provides a detailed analysis of the datasets selected for this study and the evaluation methodology used. We have carefully examined each dataset to highlight its importance to our research and to explain the rigorous analytical approach we have employed. By focusing exclusively on classification tasks, we aim to ensure that our findings are valid, reliable, and directly applicable to scenarios where classification is essential.

3.1 Datasets

Traditional ML datasets are abundant, but datasets for FL training are relatively scarce. However, the FL domain is currently experiencing a growing number of contributions. In our work, we use a combination of real-world datasets and synthetic datasets. We source real-world datasets from public platforms, such as the LEAF benchmark for FL (Caldas et al., 2018), alongside our adaptations of centralised datasets. We also incorporated the extensive eICU database, which was carefully processed for the FL application. The datasets were selected to support a variety of experiments designed to evaluate the proposed methodologies.

3.1.1 Real-world Datasets

Real-world datasets exhibit a mixture of feature distribution skew, class distribution skew and quantity shift to characterise statistical heterogeneity (see Section 2.2). The core experiment (i.e. for evaluating proposed methodologies) setup considers four real-world

datasets, while Chapter 7 focuses on the healthcare dataset for the case study. The four real-world datasets considered in the core experiment setup are MNIST, FEMNIST, Fed-MEx and Fed-Goodreads. For the final real-world case study, we use the eICU database (Pollard et al., 2018). In this section, we will concentrate only on the core datasets. The eICU database, given its significance, will be discussed in detail in Section 7.1. Fed-MEx and Fed-Goodreads datasets are FL versions contributed from our work based on their original datasets.

MNIST is a handwritten digit recognition dataset adapted in the FL setting. We reuse the FL setting proposed by Li et al. (2018) where there are 69,035 data samples of 10 classes distributed among 1000 clients and each client has samples for only 2 classes. A data sample is an image of size 28×28 and the number of samples per client follows a power law.

FEMNIST (Federated-Extended-MNIST) is a handwritten character recognition dataset from Caldas et al. (2018). The subsample consists of 10 lowercase characters (a-j) and is used for a 10-class character classification task. This dataset is distributed among 200 clients, with each client having samples for only 3 classes. Each data instance in this dataset is an image with dimensions of 28x28.

Fed-MEx is a novel FL dataset produced with this work. MEx is a publicly available exercise recognition dataset collected with 30 subjects performing 7 different physiotherapy exercises Wijekoon et al. (2020)[1]. The Fed-MEx dataset has 934 data samples from the pressure mat subset of the MEx dataset. Each client has a random amount of samples for only 2 exercise classes. A pressure mat data sample contains a sequence of heat maps (size $5 \times 16 \times 16$) recorded for 5 seconds with $1Hz$ frequency. MEx has previously been used for personalised activity recognition research (Wiratunga et al., 2020) and forms an interesting contrast to the other image and text datasets. The federated version of this dataset and the generation code is published on GitHub[2].

Fed-Goodreads is a novel FL dataset produced in this work. Goodreads[3] is a publicly available dataset and commonly used for text classification with DL due to its large volume (Maity et al., 2018; Thelwall, 2019; Wan et al., 2019; Zuccala et al., 2015). Fed-Goodreads contains the book reviews subset with parsed spoiler tags is used

[1] https://archive.ics.uci.edu/ml/datasets/MEx
[2] https://github.com/chamathpali/Fed-MEx
[3] https://mengtingwan.github.io/data/goodreads.html

(1.38m reviews) to perform a binary classification task of predicting if a review sentence contains a spoiler or not. This dataset provides an ideal peronsalised setting for a federated dataset as the data is organised by individual users, where a user has different quantities of data and different users have different patterns of writing sentences. Fed-Goodreads contains 100 unique clients. The number of samples per client is limited to 2-10 to enforce statistical heterogeneity and each data sample contains 2517 features. The 2,517 features in the Fed-Goodreads dataset represent the unique words (tokens) used to describe the text data. These features were selected using a count vectoriser with a minimum document frequency parameter set to 1,000, which included only words appearing in at least 1,000 reviews. Each feature corresponds to a specific word from the vocabulary, with the value for a given feature in a data sample indicating the count of that word in the corresponding review. The variation in text vocabulary in relation to the binary classification makes this a challenging text classification task. The curated FL version of the dataset and the generation code is published on GitHub[4].

The four datasets were selected due to their variety and also to be comparable with FL state-of-the-art algorithms. Table 3.1 provides a comprehensive summary of the four datasets utilised in the core experiments, detailing their specifications. The table provides a detailed overview of the chapters where each dataset is used for evaluation purposes.

Table 3.1: Summary of Real-World Datasets Utilised in Core Experiments and Evaluations

	MNIST	FEMNIST	Fed-MEx	Fed-Goodreads
Type	Image	Image	Sensor	Text
Application	Digit recognition	Character recognition	Human activity recognition	Text classification (binary)
Classes	10	10	7	2
Train Size	61,664	17,840	934	355
Test Size	7,371	2,083	250	130
Clients	1000	200	30	100
Features	28x28	28x28	5 x 16 x 16	2517
Chapters	4, 5, 6	4, 5, 6	4, 5	4, 5

[4]https://github.com/chamathpali/Fed-Goodreads

3.1.2 Synthetic Datasets

The real-world datasets we used fall within the non-IID spectrum, but there can still be extreme non-IID scenarios we need to explore. We use six variants of synthetic datasets to study the effect of varying levels of non-IID and IID datasets. Synthetic datasets are generated using the approach described in Shamir et al. (2014) and widely used in FL experiments (Duan et al., 2020; Li et al., 2018). Two parameters α and β control the statistical heterogeneity of the generated data: increasing α increases class distribution skew by controlling the class generator model; Furthermore, increasing β increases feature distribution skew. Together, both parameters will impact levels of concept shift, where the relationship between features and classes can change from client to client. Data samples, each with 60 features, are generated using different α and β values to obtain six datasets with varying degrees of statistical heterogeneity. The dataset includes ten classes and 30 clients, and the distribution of the number of samples per client follows a controlled power law. The notation $synthetic(\alpha, \beta)$ denotes a synthetic dataset. We can generate a *synthetic IID* dataset by using identical distributions for features and classes across clients. All statistical heterogeneity factors (in Section 2.2) feature in synthetic non-IID datasets. In Chapter 4, we use synthetic datasets primarily to evaluate the proposed *FedSim* method.

Measuring Statistical Heterogeneity

To study the relationship between similarity and non-IIDness, we propose a novel privacy-preserving data characterisation measure called Privacy-preserving Non-IID Index (PNI) as a contribution of this research. PNI is used to analyse the impact of statistical heterogeneity on FL performance. Such a measure of non-IIDness should increase with increasing statistical heterogeneity. Model error captures the extent to which a model fails to generalise to its underlying data, which can be due to the heterogeneity of the feature distributions. A non-IIDness measure can be defined as a function of error terms. Unlike raw prediction data, using derived information like model error is privacy-preserving and preferable over measures that rely on access to raw data.

Root mean square error (RMSE) is selected as the error function for the proposed *PNI* measure. RMSE is preferred for capturing the magnitude of prediction errors and providing insights into deviations from actual values. However, if a different measure better aligns with the analysis goals, it can be replaced with another metric, such as Mean Absolute Error (MAE) or accuracy. RMSE calculations require access to raw data (actual

and predicted class data), which should not be communicated to the server for privacy reasons. In order to overcome the privacy concern, a model, w_{rnd}, is initialised at the server with random weights and communicated to all clients. Here w_{rnd} is considered to be a neural model suitable for the reasoning task of clients. Each client i will predict labels $\hat{y}_i$ for all its training data D_i (Equation 3.1).

$$\hat{y}_i \leftarrow predict(w_{rnd}, D_i) \tag{3.1}$$

Each client will then use the predicted labels to calculate RMSE concerning the actual label y and for n number of local samples (Equation 3.2).

Once computed locally, $RMSE_i$ will be communicated to the server.

$$RMSE_i = \sqrt{\sum_{n}^{D_i} \frac{(\hat{y}_n - y_n)^2}{n}} \tag{3.2}$$

Here, we can use paired differences between model errors to measure the statistical heterogeneity of a client, i:

$$PNI_i = \left\| RMSE_i - RMSE_{j \neq i} \right\|_2 \tag{3.3}$$

where differences in RMSE between client, i, and all other clients, $j \neq i$, are computed using the Euclidean norm (L2 distance). The PNI, for the federation of all clients, is calculated as an average over client PNI values:

$$PNI = \frac{1}{|C|} \sum_{i}^{C} PNI_i \tag{3.4}$$

PNI is applied explicitly to synthetic data because synthetic datasets allow for controlled variation in data characteristics (e.g., different α and β values), enabling accurate calculation of PNI values across various levels of heterogeneity.

3.2 Federated Learning Setup

For the foundation of our FL setup, we use the open-source implementation of *FedProx*[5] as a starting point. Subsequently, we integrate our proposed methodologies into this framework to conduct our experiments. All experiments are implemented using Python, explicitly leveraging the TensorFlow libraries (Abadi et al., 2015) for ML functionalities. Incorporating existing frameworks, such as the open-source implementation of *FedProx*, into our research offers numerous distinct advantages. Firstly, it provides a foundation of reliability and robustness, having been widely tested within the research community. This reliability is crucial for the integrity of our experiments. Secondly, by building on these frameworks, we significantly expedite our development process, concentrating our efforts on innovating within our proposed methodologies rather than constructing the infrastructure from scratch. Leveraging established libraries like TensorFlow further enhances this benefit by offering a wide array of ML tools and functionalities, all optimised for performance and scalability. Another benefit of using the existing FL setup is that the federated versions of the datasets, like MNIST and FEMNIST, are already compatible and allow a good comparison with the existing baselines.

3.3 Evaluation Methodology

Evaluating methodologies for FL presents unique challenges compared to traditional centralised model training. This is due to the distributed architecture of FL, which involves a central server and numerous client devices. Consequently, the evaluation criteria and methods must be adapted to address the different components of the FL system. For example, evaluating the performance of individual clients requires a client-specific assessment approach, while measuring the performance of the central server might focus on aspects such as aggregation efficiency or resource utilisation. Similarly, evaluating communication performance requires metrics that capture the efficiency and reliability of data exchange between clients and the server. In this section, we describe the selected evaluation metrics and measures.

[5] https://github.com/litian96/FedProx

3.3.1 Performance Metrics

Test Accuracy

As the field of FL continues to expand, researchers are using a variety of metrics to evaluate FL systems. However, one particular measure has gained widespread acceptance and standardisation within the FL research community: the test accuracy of the global model, evaluated against the test data of every client. This metric has proven to be an effective way to assess the performance of FL systems and is now widely used for research and development in the field.

The test accuracy is calculated in each round and plotted against the communication rounds. To calculate the test accuracy of a FL algorithm: Once the global model is updated using the global aggregation step at the end of a communication round, it is communicated to all clients to evaluate using their test data. At any given round, an algorithm's test accuracy is calculated using the formula:

$$\text{Test Accuracy} = \frac{\sum_{k=1}^{K} \text{Correct Predictions}_k}{\sum_{k=1}^{K} \text{Test Samples}_k}$$

where K represents all the clients in the system. This value is plotted against the number of communication rounds to evaluate the performance of a baseline model. In comparative evaluations, a mean performance improvement is calculated as a quantitative measure. The cumulative difference of test accuracy measures between two algorithms are averaged over the number of rounds to obtain the mean performance improvement as a percentage.

Test accuracy serves as the primary measure for evaluating the proposed methodologies, including *FedSim*, *FedFT*, and *p*FGD, as well as in the case study chapter throughout this work.

Communication Cost

In evaluations related to capturing communication performance, we have introduced an additional performance metric that quantifies these costs in megabytes (MB). This metric is based on the storage size of a model, denoted as w, and measured in MB using the notation $\Theta(.)$. We measure the upstream communication cost accumulated over t communication rounds per client as $t \cdot \Theta(.)$.

Analysing Error in Attack Reconstructions

When evaluating the security enhancements work with pFGD, We log the reconstructed instance's Mean Squared Error (MSE) and the original image at each iteration. These MSE metrics serve a pivotal role in evaluating and interpreting the efficacy of the proposed method. By counting the number of successful bypasses at each threshold, we gain insights into the effectiveness of the different variants in defending against the respective attacks. Considering the minimum MSE value from each experiment ensures that we capture the reconstruction's performance under various conditions and iterations.

3.3.2 Robustness and Statistical Significance

Randomness is significant in many experiments, especially ML and statistical analysis. Running experiments with different random seeds multiple times can help understand the robustness of algorithms or models being tested by observing the variability in the results. The core experiments performed on all the datasets were carried out with 35 random seeds (from 0 to 34 incremented by 1) to empirically demonstrate the significance. Repetition of the same experiment with different random seeds helps to reduce the sampling error of our experiments.

Statistical significance helps quantify whether an outcome of an experiment is random or likely due to the factor of interest. Therefore, a one-tailed hypothesis test with a significance level of 0.05 was carried out to determine if *FedSim* performed better than *FedAvg* and *FedProx*. The null hypothesis for these tests stated that the performance of *FedSim* is not significantly higher than that of *FedAvg* and *FedProx*. One-tailed t-tests were chosen because the primary goal was to determine whether *FedSim* outperformed the other methods. The tests were specifically designed to detect if the mean performance of *FedSim* was significantly higher than that of *FedAvg* and *FedProx* at each communication round.

3.3.3 Baselines

To effectively evaluate and determine the effectiveness of the proposed methodologies, we select the following two baselines.

- ***FedAvg***: The general FL methodology as described in McMahan et al. (2017).
- ***FedProx***: variant of *FedAvg* focused on improving stability and performance in non-IID settings using regularisation in client update from Li et al. (2018).

These baseline selections ensure a comprehensive evaluation covering both traditional FL approaches (*FedAvg*) and more advanced techniques tailored for specific challenges (*FedProx*). The comparison with *FedAvg* and *FedProx* was selected because they were the standard baselines in FL at the time of this research, effectively addressing both general and non-IID scenarios. It is important for the baselines to be generalisable and not specific to any particular domain, dataset or model type. By comparing our methodologies to established baselines, we aim to provide insights on their strengths, weaknesses, and potential areas of improvement.

3.3.4 Basic Experiment Model and Hyper-parameter Selection

All datasets present classification tasks which we initially model using Multinomial Logistic Regression (MLR). A flattened feature vector is used as the input. Input sizes for image data, Fed-MEx, Fed-Goodreads and *synthetic*(α, β) are 784, 1280, 2517 and 60. Table 3.2 summarises the training-test split strategies and handling clients with low sample counts across the different datasets used in this study. Each dataset employs a specific approach to data splitting tailored to its characteristics and requirements. The split ratios vary, with most datasets using either 90%-10% or 80%-20% splits, while the eICU dataset utilises a larger 30% test set to account for sample volume.

Table 3.2: Training-Test split across different datasets

Dataset	Train size	Test size	Sample details
MNIST	90%	10%	Minimum of 10 samples per client.
FEMNIST	90%	10%	Minimum of 2 samples per class for each client.
Fed-MEx	80%	20%	Each client is assigned data from 2 randomly chosen exercises. Clients with very few samples may have small test sets.
Fed-Goodreads	80%	20%	Clients need at least 3 spoiler and 3 non-spoiler reviews. Random limits balance class sizes. The first 100 qualifying clients are selected.
synthetic(any)	90%	10%	Minimum of 50 samples per client.
eICU Case Study	70%	30%	Minimum of 200 samples per client with at least one record per class.

We also select following hyper-parameters: number of epochs for local update as 20; and batch size for local update as 10. Learning rates used for local update are 0.03, 0.003, 0.01, 0.3 and 0.01 for MNIST, FEMNIST, Fed-MEx, Fed-Goodreads and *synthetic(any)*

respectively. Number of communication rounds are limited after convergence or at maximum 500 rounds. Selected communications rounds for MNIST, FEMNIST, Fed-MEx, Fed-Goodreads and *synthetic*(α, β) are 30, 500, 200, 250 and 100 respectively. Hyper-parameters mentioned above for MNIST, FEMNIST and *synthetic*(α, β) were adapted from Li et al. (2018) to ensure comparability and reproducibility.

Additionally, we explore two hyper-parameters specifically for *FedSim*: the number of clients per round and the number of clusters. These are two key factors to successfully discover latent similarity properties among clients. We explored following values: 10, 20, and 30 clients per round while keeping cluster size constant at 5; and 3, 5, 7, 9, 11 cluster sizes while keeping clients per round at 10 and 20. We find 20, 20, 10, 20, 10 as the most optimal number of clients per round and 5, 9, 3, 11, 5 are the most optimal cluster sizes for MNIST, FEMNIST, Fed-MEx, Fed-Goodreads and *synthetic(any)* datasets respectively. Hyper-parameter are summarised in Table 3.3.

Table 3.3: Hyper-parameter details

Dataset	Features	Learning rate	Total clients	Com. rounds	Clients per round	Number of clusters (*FedSim*)
MNIST	784	0.03	1,000	30	20	5
FEMNIST	784	0.003	200	500	20	9
Fed-MEx	1280	0.01	30	200	10	3
Fed-Goodreads	2517	0.3	100	250	20	11
synthetic(any)	60	0.01	30	100	10	5

3.3.5 Generalisability to neural architectures

Applicability of the proposed methods to different neural architectures that are of different dimensions is key to generalisability. We evaluate this with the three most commonly used neural architectures: Multi-layer Perceptrons (MLP); Convolutional Neural Networks (CNN); and Recurrent Neural Networks (RNN). Table 3.4 summarises the model parameters' dimensions. $|w|$ is the dimensions of each layer, and the params column presents the total parameters in the model.

It is important to highlight that each architecture makes use of a unique multi-dimensional tensor. For the FEMNIST and MNIST datasets, a CNN-2D architecture with 6.49 million parameters is employed. In the case of Fed-MEx, a deep neural network called MLP-3 is utilised with 2.53 million parameters. Lastly, the Fed-Goodreads dataset employs an RNN architecture. The hyperparameters are kept same with the

Table 3.4: Alternative neural architectures

Dataset	Model	Architecture	$\lvert w \rvert$	Params
FEMNIST MNIST	CNN-2D	$conv2d(5,5)64 \rightarrow$ $maxpool(2,2) \rightarrow$ $conv2d(5,5)64 \rightarrow$ $maxpool(2,2) \rightarrow$ $dense(2048) \rightarrow dense(10)$	[[5, 5, 32], [32], [5, 5, 64, 32], [64],[3136, 2048], [2048], [2048, 10], [10]]	6.49M
Fed-MEx	MLP-3	$dense(1280) \rightarrow$ $dense(640) \rightarrow$ $dense(120) \rightarrow dense(7)$	[[1280, 1280], [1280],[1280, 640],[640], [640, 120], [120],[120, 7],[7]]	2.53M
Fed-Goodreads	RNN	$embedding(25) \rightarrow$ $rnn(128) \rightarrow dense(2)$	[[25, 25], [25, 128], [128, 128], [128], [25, 128], [128, 2],[2]]	20K

exception of reducing the number of local epochs to 10 and the learning rate to 0.0001, to prevent over-fitting on the Fed-Goodreads dataset.

3.3.6 Reproducibility

In order to ensure that our research findings can be replicated by others, we took several measures. Firstly, we made sure that all of our methods and procedures are reproducible by setting seeds for random states and proper documentation. We have made the source code for our proposed methodologies and datasets publicly available on GitHub.

3.4 Chapter Summary

In this chapter, we formalise the concepts of FL and describe the experiment setup that will be used throughout this thesis. We select four real-world datasets and six variants of synthetic datasets to simulate a wide range of statistical heterogeneity. Additionally, we discussed the FL setup and its implementation details, including hyper-parameters. We presented the evaluation methodology, detailing the multiple performance metrics used. We also selected two baseline algorithms, *FedAvg* and *FedProx*, to evaluate the proposed methodologies.

Chapter 4

FedSim: Similarity Guided Model Aggregation for Federated Learning

> "Our ability to **reach unity in diversity** will be the beauty and the test of our civilization."
>
> *Mahatma Gandhi*

In the Literature Review chapter, we discussed various aggregation methodologies in FL for different purposes. We identified that some aggregation methods are focused on specific applications and need more methods focused on similarity. Leveraging the untapped potential of similarity knowledge in an FL setting can achieve significant benefits. Emphasising the need for more generalisability in most aggregation methods and preserving privacy is also essential. In this chapter, we introduce a novel FL algorithm, *FedSim*, designed to harness inter-client similarities for effective model aggregation (Objective 2). Specifically, we aim to address the first research question (RQ1): *To what extent does the identification and utilisation of similarity knowledge among clients influence model aggregation in FL?* To address RQ1, we focus on Objective 2 and Objective 3.

4.1 Use Case: Similarity of Clients in FL

As introduced previously, FL enables multiple clients to train a global model collaboratively without sharing their private data. However, it is important to note that all clients

share the same goal but can come from various backgrounds such as geographical locations, age groups, genders and literacy rates. To illustrate this concept further, consider the goal of training a model to recognise handwritten digits. Imagine 1000 clients in the FL system, each contributing 100 training samples. When training a shared model to achieve this goal, client parameters such as handwriting style, refinement of writing, image quality, and stroke thickness can be diverse. This diversity occurs due to various factors, primarily the wide distribution of clients. Table 4.1 presents examples of diverse client characteristics that can be present in handwriting data.

Table 4.1: Examples of Diverse Client Characteristics in Handwriting Data

Diversity Factor	**Characteristic**	**Description**	**Example**
Geographical Distribution	Urban vs. Rural	Difference in handwriting due to regional lifestyle and education	Client A: Modern handwriting. Client B: Traditional handwriting
	Variations	Regional writing styles	Client A: Digit '1' in serif Client B: Digit '1' as simple line
Demographic Distribution	Age	Difference in handwriting due to age-related motor skills	Client A: Younger clients. Client B: Older clients with experience.
	Occupation	Professional experience influencing handwriting	Client A: Teacher with neat and consistent digits. Client B: Software Engineer with less day-to-day writing.
	Cultural	Influence of cultural practices and norms	Client A: Calligraphy style writing. Client B: Informal writing.
Equipment Conditions	Pen	Difference in handwriting due to the pen used to write	Client A: Ball point pen. Client B: Ink pen.
	Quality	Quality of the image due to equipment used for capturing	Client A: Smartphone with 5MP camera. Client B: Smartphone with 25MP camera and flash.
	Lighting	Difference lighting condition due to environment condition	Client A: Sunny with natural lighting. Client B: Captured at night with bulb lighting.

As stated in the chapter quote by Mahathma Gandhi, it is essential to harness this

diversity to benefit everyone. Traditional FL aggregation methodologies does not consider this diversity among clients. We aim to use this client diversity as a method of capturing similarities among each other to improve the aggregation step in FL.

4.2 Do Clients Share Similar Characteristics?

Before diving into similarity-guided aggregation methods, it is essential to explore whether clients share similar characteristics. We first examine the raw data to identify if any similarity knowledge is present. Figure 4.1 illustrates the similarities using the Kolmogorov-Smirnov (KS) statistic for distribution comparison between clients using three datasets commonly used in FL research (Li et al., 2018; McMahan et al., 2017; Sattler et al., 2020a). The experiment uses a two-sample KS test, which iterates through each client selecting ten random samples from its local data. For each iteration, ten random samples are selected from every other client, including the client itself and the KS value is calculated by comparing the samples from each client pair. This approach efficiently captures representative differences without comparing all possible data samples.

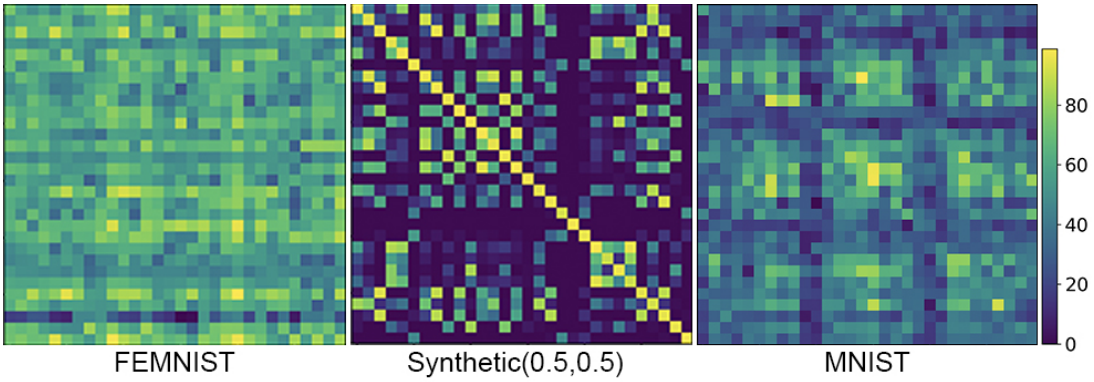

Figure 4.1: Pairwise similarity between clients in FL datasets

In the resulting plots from Figure 4.1, yellow indicates high similarity between the clients' data distributions, while dark blue signifies significant differences. In the synthetic dataset experiment, the diagonal of the plot mainly displays yellow, indicating a high degree of similarity. This is because the synthetic data for a client is generated from the same distribution, resulting in consistent matches across the local data distributions. However, for the MNIST and FEMNIST datasets, the diagonal does not consistently show yellow for each client's data. This occurs because the data distributions across different clients are not strictly identical, even when sampling from the same client.

With this initial observation, it is evident that distributed datasets still demonstrate characteristics of similarity among clients. This similarity varies from dataset to dataset and for each application. For instance, applications such as image classification can have much similarity knowledge, whereas a healthcare application might have less similarity. Exploiting these pairwise client similarities could help reduce computational costs, such as the number of FL communication rounds needed to achieve comparable convergence. Similarity knowledge can also be used to identify divergent clients, thereby tempering their influence on the global aggregation (i.e., reducing variation).

Accessing raw data should be minimised in an FL setting, so we take one step further to analyse if this similarity knowledge is passed down with locally updated gradients. Generally, gradients can be used to represent the underlying data. We use the four real-world datasets described in Section 3.1.1 and cluster the clients using the two most significant PCA components derived from the gradients. This approach represents and clusters the clients using K-Means with $n\_clusters = 10$.

Figure 4.2 presents a two-dimensional mapping of the clustering, showing the clients and their cluster memberships (using 10 colours for the different clusters). FEMNIST has well-defined clusters, exhibiting higher intra-cluster similarity (density) and greater inter-cluster distance (separability) than the other datasets. Similar observations can be made for the MNIST clustering, where although clusters are densely formed their separability is less prominent than FEMINIST's. In contrast, Fed-MEx lacks well-defined clusters. Like MNIST, in Fed-Goodreads, we can observe reasonable clustering but with weaker separability between clusters. Our clustering analysis suggests that clients in an FL setting have similar characteristics and the proposed method should be able to exploit this similarity.

4.3 Utilising Client Similarities to Enhance Model Aggregation

In this section we introduce *FedSim*, a novel approach to leverage inter-client similarities for more effective model aggregation in FL. As demonstrated in Section 4.2, similarity knowledge is present in client gradients. The goal of *FedSim* is to utilise this knowledge to provide an improved aggregation strategy while preserving privacy.

FedSim favours similar clients when formulating locally specialised models while also creating a generalised global model that can accommodate practical differences among these

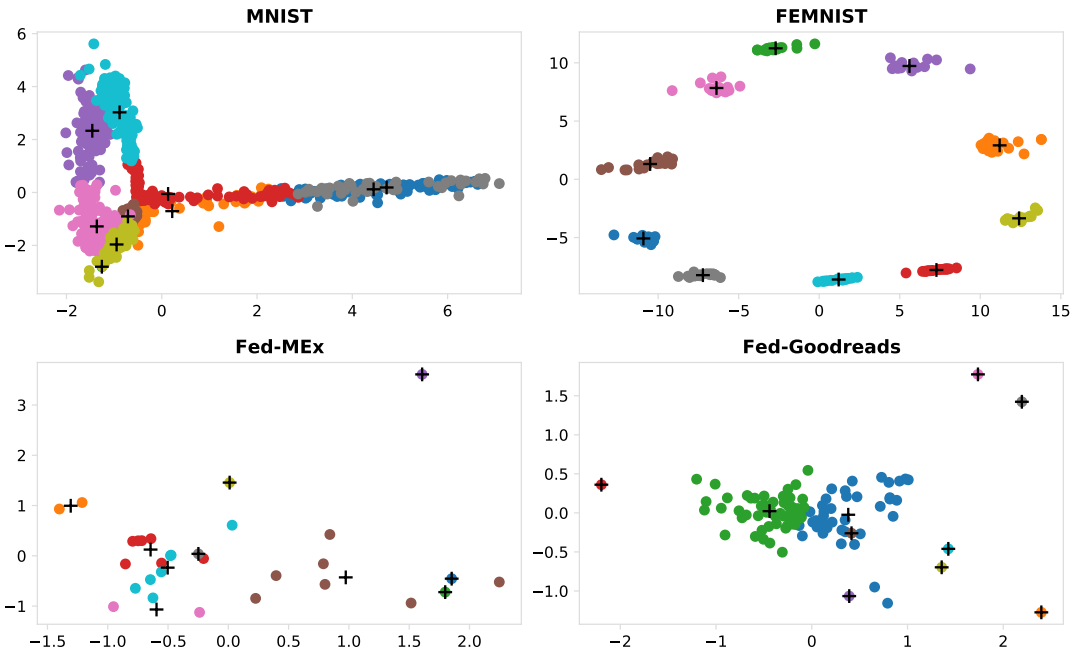

Figure 4.2: An example clustering of clients to visualise similarity

specialised models. Using similarity knowledge in clustering helps improve coverage and identify representative clients. In non-IID settings, improving client coverage enhances the generalisability of the global model. At the same time the use of similarity helps identify and mitigate potentially harmful influences on global aggregations by divergent clients.

4.3.1 Federated Learning with *FedSim*

A high-level view of the proposed *FedSim* algorithm is illustrated in Figure 4.3.

Following the distribution of a randomly initialised global model to clients, a *FedSim* communication round has the following steps: client sampling, clustering, local updates cluster aggregation and global aggregation.

Step 1 Due to communication constraints and intermittent client availability, a subset of clients are randomly sampled to participate in each FL round.

Step 2 Selected clients are clustered based on their local gradients without sharing each other's privately held data. These gradients are calculated based on errors from the most recently distributed global model.

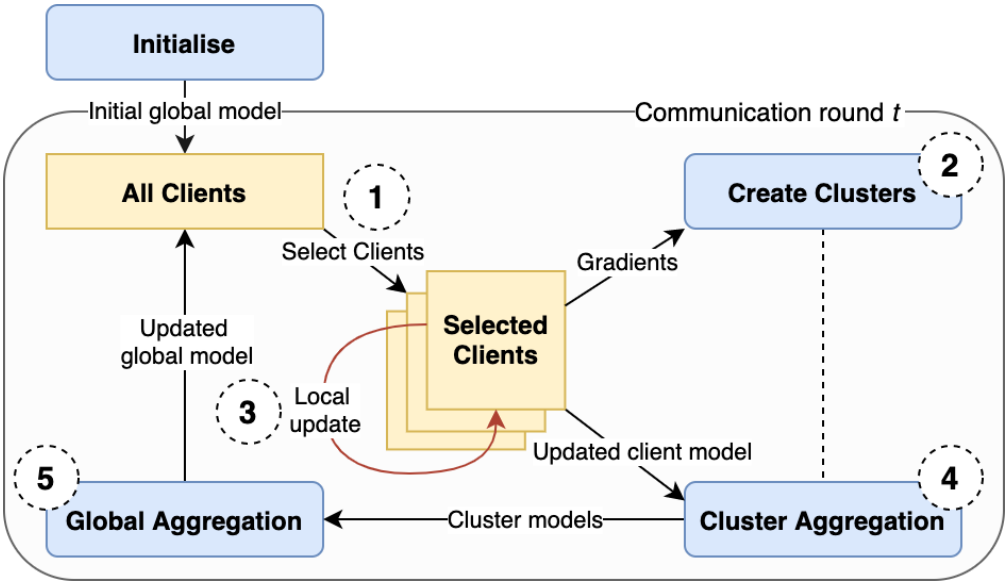

Figure 4.3: High-level approach of the proposed *FedSim* algorithm

Step 3 Selected clients continue to update their model weights (using an optimisation method such as SGD) to produce their local models.

Step 4 Locally updated client weights are combined within each cluster using a weighted average to form a specialised, representative model for the cluster. Here, weights are a function of a client's sample size, given the cluster.

Step 5 a global aggregation step is used to combine the specialised models to generate the new global model

Follow-up sections discuss these *FedSim* steps in detail, providing a theoretical underpinning and with reference to Algorithm 2.

4.3.2 Federated Optimisation with Clusters

Mathematically, an ML optimisation problem aims to minimise an objective function, f, which is defined as follows for an instance, i, with weights, w:

$$\min_{w} f(w^*) \quad \text{where} \quad f(w) = \mathbb{E}(f_i(w)) \tag{4.1}$$

Algorithm 2 *FedSim* Algorithm

Require: w_0 initial global model, $\mathcal{K}$ clients, *n_ clusters*

for t=0,1,2,.. **do**
  Broadcast w_t to all clients
  Select $\mathcal{S}$ clients where $\mathcal{S} \subset \mathcal{K}$
  $\mathcal{C} \leftarrow$ Clustering(S, *n_ clusters*) *(Algorithm 2)*
  for all $c \in \mathcal{C}$ **do**
    for all $k \in c$ **do**
      $w_t^k \leftarrow$ updates w_t using SGD
    end for
    $\bar{w}_t^c \leftarrow$ ClusterAggregation$(w_t^1, w_t^2, ..., w_t^{|c|})$ (Eq.4.11)
  end for
  $w_{t+1} \leftarrow$ GlobalAggregation$(\bar{w}_t^1, \bar{w}_t^2, ..., \bar{w}_t^{|\mathcal{C}|})$ (Eq. 4.12)
end for

Here f is a function of the model error (e.g. $f_i(w) = (\hat{y}_i - y_i)^2$) and $\mathbb{E}$ is the expected value. In a FL system with K clients indexed by k, each with n_k data instances, the objective function for a client k is:

$$F_k = \mathbb{E}(f_i(w); \forall i \in k) = \frac{1}{n_k} \sum_{i=1}^{n_k} f_i(w) \tag{4.2}$$

Let the objective function of *FedAvg*, for a set of selected clients, K, in a given round be:

$$F_O = \mathbb{E}(F_k; \forall k \in K) \tag{4.3}$$

Suppose we used a clustering algorithm (such as k-means) to create a client cluster, c, containing a set of S clients, then we can define an objective function for that cluster as:

$$F_c = \mathbb{E}(F_k; \forall k \in S) \tag{4.4}$$

Accordingly, for a FL system with *FedSim*, having a set of clusters, C, let the objective function be:

$$F_G = \mathbb{E}(F_c; \forall c \in C) \tag{4.5}$$

Furthermore, to prove the impact of similarity-based clustering, let F_R be the objective function of a randomly formed cluster (R) of the same size as that of cluster c (as in Equation 4.4). Then:

$$F_R = \mathbb{E}(F_k; \forall k \in R) \tag{4.6}$$

Equation 4.2 is a function of model error, which increases with increasing variance in data points. When clients are tightly clustered using similarity-based measures as in Equation 4.4, it is expected that the error variance in a single cluster will be lower than that of a cluster that is formed randomly. Hence, when the weight parameters tend towards optimal values, we expect:

$$F_c \leq F_R \tag{4.7}$$

$$\mathbb{E}(F_c) \leq \mathbb{E}(F_R) \tag{4.8}$$

Note that in *FedAvg*, clients are randomly included within a single cluster, therefore from Equation 4.3 and 4.5, we expect:

$$F_G \leq F_O; \text{ where } F_G = \mathbb{E}(F_c; \forall c \in C) \text{ and } F_O = \mathbb{E}(F_k; \forall k \in K) \tag{4.9}$$

Now, suppose we introduce an arbitrary constant λ as follows:

$$F_G + \lambda = F_O \tag{4.10}$$

such that in Equation 4.10, λ is a regularisation term that varies with cluster settings (e.g. number of clusters, clustering method, similarity metric, dimensionality). This would suggest that we can obtain a more regularised objective function using an informed clustered approach, producing improved performance compared to an FL method with a single cluster. Indeed, when all clients belong to a single cluster, then Equation 4.5 is identical to *FedAvg* (F_O) where $|C| = 1, n_c = n$ and λ becomes 0. When $|C| > 1$, we expect clustering to better represent the federated problem space by ensuring that the influence of similar clients does not dominate the aggregation of parameters in federated learning. This helps reduce potential variance in the weight aggregation (averaging) step. Our idea is to cluster clients according to the similarity of client parameters, which acts as a proxy for similarity based on client data.

Initialisation

The initialisation step of *FedSim* is identical to *FedAvg* where a global model is initialised with random weights, w_0. Here, the global model (and corresponding local models) is selected to meet the requirements of the reasoning task (i.e. next-word suggestion or character recognition). Commonly, it is a neural architecture where w_0 is its model parameter.

Clustering

In communication round, t, a set of clusters, $\mathcal{C}$, are created with a subset of clients, $\mathcal{S}$, randomly sampled from the set of all clients, $\mathcal{K}$, where $\mathcal{S} \subset \mathcal{K}$. This clustering process (see Algorithm 3) is triggered in each *FedSim* round as shown in Algorithm 2, line 4. A client, k, is represented using the gradient vector, g_k, obtained using the client's training data error of the recently distributed global model, w_t. Once client gradients are communicated to the server, we use a clustering algorithm, specifically *kmeans++* (Arthur and Vassilvitskii, 2007), to form $|\mathcal{C}|$ number of clusters (i.e. $n\_clusters$) based on the similarity of client gradient vectors. Clustered clients perform SGD to create locally updated models (w_t^k for client k).

Clients are sampled without replacement at each round (as in *FedAvg*), which helps with applications having intermittent clients. Accordingly, each round's clustering step in *FedSim* must be repeated. The computational cost of each communication round can increase exponentially due to pairwise similarity calculations for clustering. We alleviate this cost by sampling a few clients in each round, and applying dimensionality reduction (in this work, we use PCA) to the gradient vectors.

Algorithm 3 *FedSim* Clustering Method

Require: $\mathcal{S}$ clients, $n\_clusters$, w_t model
1: **for all** $k \in \mathcal{S}$ (selected clients in round t) **do**
2: $g_k \leftarrow$ compute gradients for w_t using SGD on local data
3: **end for**
4: $G' \leftarrow g_1 ... g_{|\mathcal{S}|}$, where gradients g_k received from each client k $\in S$
5: $G \leftarrow$ dimensionality_reduction(G') ; e.g. PCA
6: $\mathcal{C} \leftarrow$ client_clustering($n\_clusters$, G') ; e.g. K-Means++
7: **return** $\mathcal{C}$

The use of gradient vectors ensures that semantically meaningful private data is not communicated when similarity is computed. We also expect that for similarity, gradients will capture latent patterns of a client's data w.r.t. model error. Accordingly, using

gradient vectors to derive similarity between clients ensures that *FedSim* can reason about client similarity without exposing client data or meta-data to the server or other clients. Other locality-sensitive hashing methods can also be adopted to secure the communication of similar information further.

As stated in Section 4.3.2, when $n\_clusters = 1$, Algorithm 2 is equivalent to the baseline *FedAvg*; where cluster aggregation in step 9 is applied to all clients involved in a given FL round using the aggregation in Equation 2.1; and step 11 becomes redundant (average for a single cluster). In addition to $n\_clusters = 1$, the *FedProx* algorithm combines a proximal regularisation term in step 7.

Cluster Aggregation

The purpose of cluster aggregation is to combine local models to create a representative model for each cluster. For a given cluster, c, at round, t, a cluster model is formed as:

$$\bar{w}_t^c \leftarrow \sum_{k \in c} \frac{n_k}{n} w_t^k \tag{4.11}$$

Here n_k is the sample size of client k, n the total number of samples and w_t^k the updated local model of client k (i.e. after locally updating the global model w_t). The weighting by sample size is borrowed from the original *FedAvg* aggregation. At the end of cluster aggregation, we obtain $|\mathcal{C}|$ clusters (i.e. $|\mathcal{C}| < |S|$), with each representing a specialisation over a distinct set of similar clients. From Equation 4.10, the cluster settings in terms of the number of clusters and the metric space for similarity computations all contribute to λ regularisation term.

Global Aggregation

The main objective of FL settings is to learn a global model that is generalisable to clients. In *FedSim* the new global model for distribution in a round, $t + 1$, is created as:

$$w_{t+1} \leftarrow \frac{1}{|\mathcal{C}|} \sum_{c \in \mathcal{C}} \bar{w}_t^c \tag{4.12}$$

which is an average over all cluster models $\bar{w}_t^c$. Here, all clusters are considered equally important to ensure equal coverage of the federated cluster space $\mathcal{C}$.

4.4 Experiment Setup

To evaluate the proposed *FedSim* methodology, we use the following datasets as described in Section 3.1. MNIST, FEMNIST, Fed-Goodreads, and Fed-MEx are used for real-world datasets. For synthetic data, we use the six variants described in Section 3.1.2. *FedAvg* and *FedProx* are used as baseline methods for comparison. For evaluation, we use the test accuracy as described in Section 3.3.1. Additionally, the experiments are carried out with 35 random seeds (ranging from 0 to 34) to demonstrate significance and robustness. The setup and hyper-parameters are as described in Section 3.2. The source code for the experiment setup is available on GitHub [1].

As *FedSim* is a novel aggregation methodology, we conduct diverse experiments to study its performance and generalisability. Each experiment assesses various aspects of the *FedSim* algorithm to ensure its effectiveness and robustness in different scenarios. Table 4.2 summarises the experiments conducted, their objectives, and the setup.

Table 4.2: Overview of experiments conducted with its objective and setup

Experiment	**Objective**	**Setup**
Evaluation with real-world data	FedSim's robustness and performance improvements in real-world scenarios and data types	Datasets: MNIST, FEMNIST, Fed-Goodreads and Fed-MEx Model: MLR
Evaluation with different learning models	Generalisability with different neural architectures	Datasets: MNIST, FEMNIST, Fed-Goodreads and Fed-MEx Models: CNN-2D, MLP-3 and RNN
Impact of similarity based clustering vs random	Does similarity based clustering perform better	Dataset: MNIST, FEMNIST, Fed-Goodreads and Fed-MEx Model: MLR
Impact of PCA	Minimise computation costs	Datasets: MNIST, FEMNIST, Fed-Goodreads and Fed-MEx Model: MLR
Evaluation with synthetic data	Investigate impact on FedSim with varying levels of statistical heterogeneity	Datasets: 5 non-IID, 1 IID Model: MLR
Analysing statistical heterogeneity	Using the PNI method study the relationship between similarity and non-IIDness	Datasets: 33 variants of synthetic data

[1] https://github.com/chamathpali/FedSim

4.5 Results and Discussion

In this section, we discuss the results of the experiments conducted. We begin by analysing experiments using real-world data and then those utilising synthetic data.

4.5.1 Evaluation with Real-world Datasets

Figure 4.4 presents performance results for the three algorithms with increasing rounds on four real-world datasets. *FedSim* achieves higher performance on all datasets with noticeable improvements in FEMNIST, Fed-MEx and Fed-Goodreads. All algorithms show stable convergence with increasing rounds on three datasets (MNIST, FEMNIST and Fed-MEx), with *FedSim* demonstrating earlier convergence and improved stability on FEMNIST and Fed-MEx. *FedAvg* and *FedProx* have very similar convergence graphs on

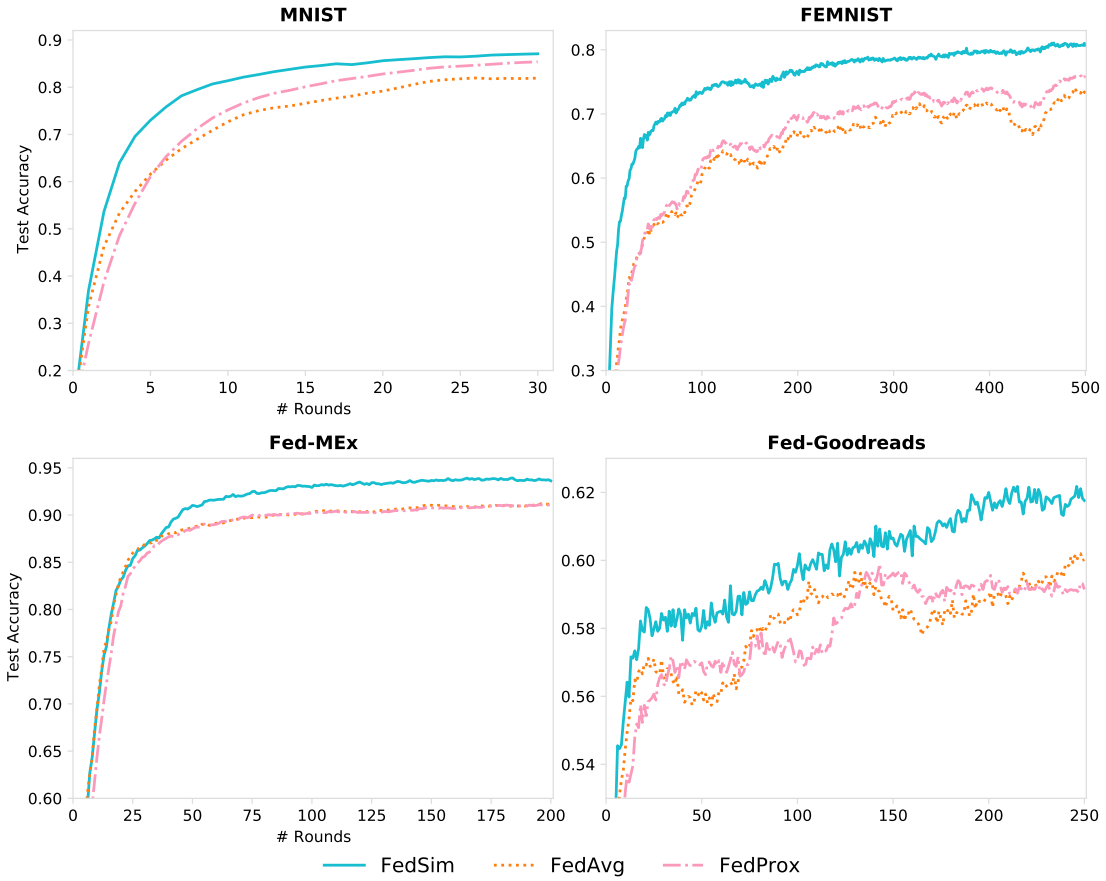

Figure 4.4: Comparison of performances over communication rounds

Fed-MEx but *FedSim* outperforms these baselines from approximately round 30 onwards.

Convergence on Fed-Goodreads was observed at round 250 when changes to loss did not exceed 8e-4 for the last 50 rounds. Although this dataset's results were less stable than the others, *FedSim* consistently outperformed the baselines. This can be expected in a dataset like Fed-Goodreads where statistical heterogeneity is high (e.g., clients with one training instance versus those with over five and low vocabulary overlap due to significant variation in word usage). However, despite reduced stability, *FedSim* maintains relatively higher accuracy with a more stable performance graph compared to *FedAvg* and *FedProx*. However it was surprising that *FedProx* which was introduced with the aim to improve stability over *FedAvg*, had performed poorly (notably on FEMNIST).

Table 4.3 lists the averaged accuracy percentage improvement gains achieved by *FedSim* over each of the two baselines. *FedSim* has significantly outperformed both baselines on all four datasets (MNIST, FEMNIST, Fed-MEx, and Fed-Goodreads). The improvement is particularly pronounced with FEMNIST. We use the cluster analysis visualised in Figure 4.2 to explore this further. Our analysis of clustering suggests that *FedSim* can exploit similarities in client learning and effectively capture these similarities by comparing their gradients.

Table 4.3: Comparison of overall performance improvements of *FedSim* over baselines

	FedSim improvement over	
Dataset	*FedAvg* (%)	*FedProx* (%)
FEMNIST	**11.11**±2.88	**9.08**±3.63
MNIST	**7.32**±2.69	**5.65**±4.35
Fed-MEx	**2.08**±1.26	**2.68**±0.78
Fed-Goodreads	**1.86**±0.73	**1.98**±0.64

In Figure 4.5, we highlight in grey any communication rounds in which *FedSim* failed to significantly outperform at least one of the baselines (significance level=0.05). Values below zero indicate negative performance against a baseline and grey vertical lines denote areas of no statistical significance. Significance testing results show that with a majority of increasing rounds, *FedSim*'s performance is superior to the baselines on all of the datasets. For instance, significance was observed after the first communication round with MNIST and FEMNIST. Whilst with Fed-MEx significance was achieved after 37 rounds and maintaining significance until the 200th round. Similar observations were noted with Fed-Goodreads where in most of the rounds (i.e. 94%), *FedSim* had achieved significant improvements. Compared to FEMNIST, Fed-MEx and Fed-Goodreads datasets where cluster separability is not as pronounced (see Figure 4.2), *FedSim* achieves only minor

improvements while maintaining statistical significance.

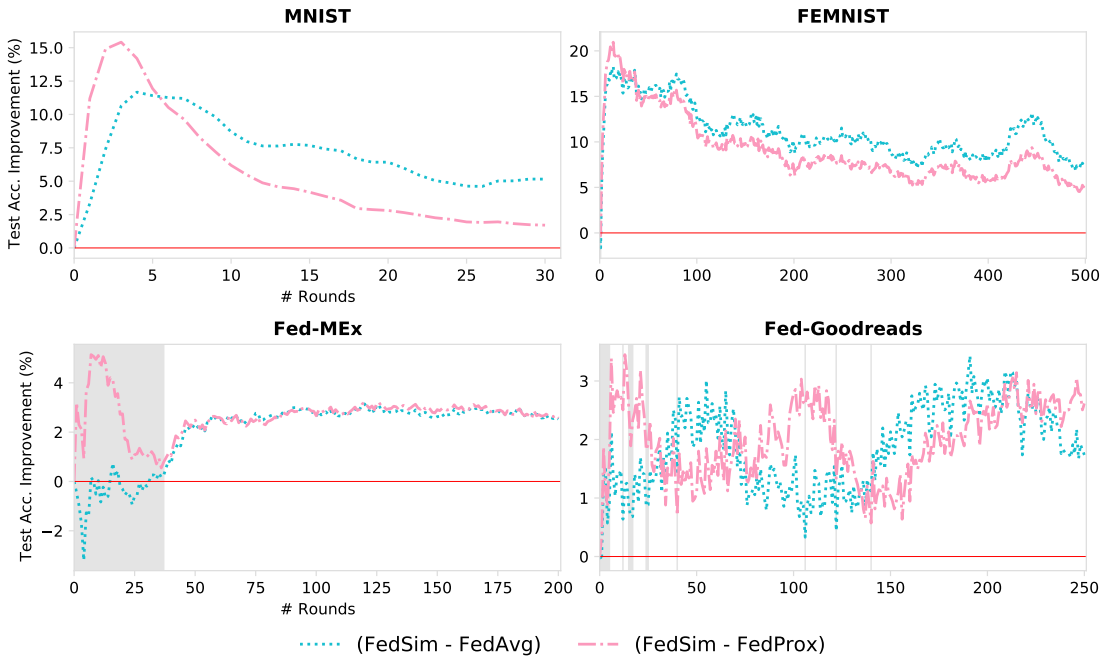

Figure 4.5: Analysis of accuracy improvements of *FedSim* compared to *FedAvg* and *FedProx* of experiments in Figure 4.4

Average time elapsed for a communication round with MNIST, FEMNIST, Fed-MEx and Fed-Goodreads in milliseconds are as follows: 2929.3, 4433.5, 759.9 and 541.8 for *FedAvg*; 1423.7, 5855.8, 949.9 and 716.7 for *FedProx* and; 2858.3, 4443.4, 1049.5 and 646.8 for *FedSim*. The comparison of average time taken per communication round is presented in Table 4.4. The average time elapsed for a communication round of *FedSim* is nearly comparable on all datasets to the other two methods. Each dataset's time taken for a round varies due to its data size and experiment configuration (e.g., local epochs, number of clients selected per round).

Table 4.4: Comparison of average time taken per communication round in milliseconds

Dataset	*FedAvg*	*FedProx*	*FedSim*
MNIST	2929.3	1423.7	2858.3
FEMNIST	4433.5	5855.8	4443.4
Fed-MEx	759.9	949.9	1049.5
Fed-Goodreads	541.8	716.7	646.8

Generalisability Over Different Learning Models

In order to analyse model generalisability of *FedSim*, further experiments were conducted with alternative neural classifiers. A 2-D CNN was used for MNIST and FEMNIST handwritten digit classification tasks, a MLP with 3 hidden layers for the Fed-MEx dataset and a single layer RNN for the Fed-Goodreads dataset. Details of these architectures appear in Table 3.4. We use the hyper-parameter configuration as discussed in Section 3.3.5.

Figure 4.6 plots the test accuracy over FL rounds using the CNNs, MLP and RNN models. It is also evident that *FedSim* maintains similar accuracy improvements and learning stability with most of the neural models, suggesting that *FedSim* is model agnostic. *FedSim*'s accuracy improvements with FEMNIST are minor because the CNN model

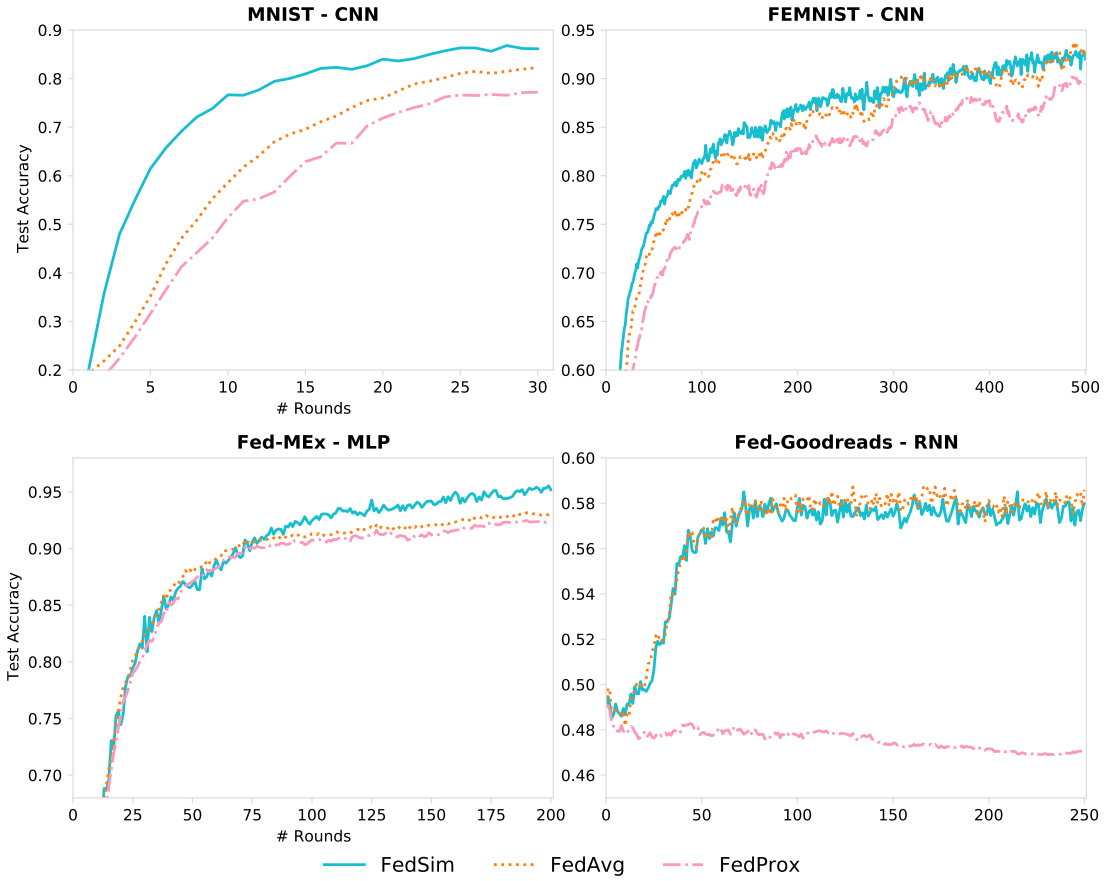

Figure 4.6: Comparison of performances over communication rounds with real-world datasets for different neural classifiers

converged quickly. However, the model maintains better stability with *FedSim* compared to *FedAvg* and *FedProx*. The results with Fed-Goodreads indicate that none of the RNN models achieved the accuracy levels previously obtained with the more straightforward logistic regression (shown in Figure 4.4).

We found that anything other than a simpler regression model led to over-fitting with this dataset. Changing the optimiser to Adam (Kingma and Ba, 2014) (from SGD) helped somewhat. While it is clear from these experiments that the logistic regression model is best for the Fed-Goodreads dataset, we can still demonstrate that *FedSim*'s performance is comparable to *FedAvg*. Note that *FedProx* was severely impacted due to its inability to use the Adam optimiser (due to its use of partial updates from straggler clients), which explains its poor performance. We used the original optimiser recommended by the authors.

Results in Table 4.5 show *FedSim* having an overall accuracy improvement on most of the datasets (except Fed-Goodreads) compared to the results seen with the logistic regression model (in Figure 4.4). These results show that the highest improvement with *FedSim* is achieved on the MNIST dataset, with an improvement of 11.69% over *FedAvg* and 17.25% over *FedProx*. The visual presentation in Figure 4.6 supports this improvement on MNIST. Both FEMNIST and Fed-MEx also show overall accuracy improvements with *FedSim*. These results empirically demonstrate that the proposed method is model-agnostic and can be used with different model architectures in practical use cases.

Table 4.5: Comparison of overall performance improvements of *FedSim* over baselines with real-world datasets with different model architectures

		FedSim improvement over	
Dataset	Model	*FedAvg* (%)	*FedProx* (%)
FEMNIST	CNN-2D	**1.42**±1.52	**5.05**±2.75
MNIST	CNN-2D	**11.69**±7.58	**17.25**±8.26
Fed-MEx	MLP-3	**0.89**±1.29	**1.79**±1.28
Fed-Goodreads	RNN	-0.32 ±0.42	**8.88**±2.87

Like Figure 4.5, we plot the accuracy improvements and analyse their significance in Figure 4.7. The grey highlights represent cases where *FedSim* did not significantly outperform at least one of the baselines (significance level = 0.05). Values below zero indicate negative performance compared to a baseline.

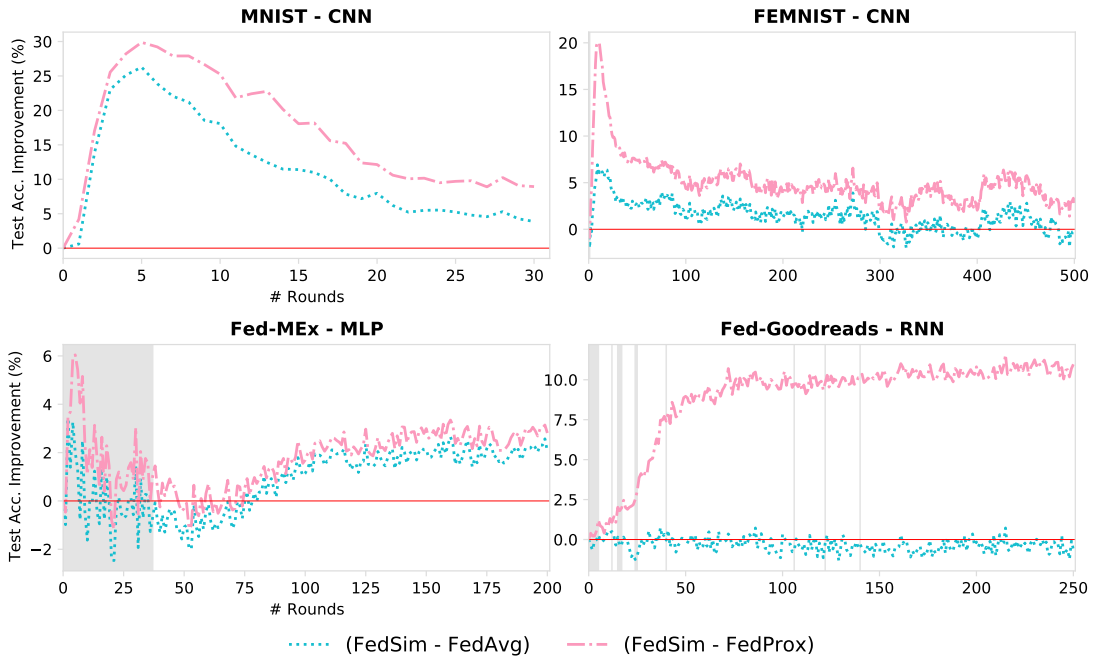

Figure 4.7: Analysis of accuracy improvements of *FedSim* compared to *FedAvg* and *FedProx* of experiments in Figure 4.6

It can be observed that *FedSim* on Fed-MEx with the MLP model initially struggles to outperform the two baselines significantly. However, after around 40 rounds, *FedSim* can surpass the performance of both baselines. This behaviour is reflected in the test accuracy plots in Figure 4.6. Once the model achieves a good level of convergence, *FedSim* further improves its performance.

Similarity Guided vs. Random Clustering

To investigate the effect of exploiting similarity knowledge, a closer examination of *FedSim* was carried out by comparing a random cluster creation approach, which assigns clients in a round-robin manner. The significant gains observed on FEMNIST with *FedSim* can be explained by this comparison in Figure 4.8, demonstrating the benefit of using similarity knowledge for model aggregation. Figure 4.8 is plotted from a single run to present the impact on random and similarity-guided clustering visually. This empirically proves our expectation in Equation 4.8. Interestingly (but not surprisingly) we also found that random clustering outperforms similarity clustering with extreme non-IID datasets (*synthetic* (0.75,0.75) and (1,1)) where there is likely to be no helpful similarity knowledge to exploit.

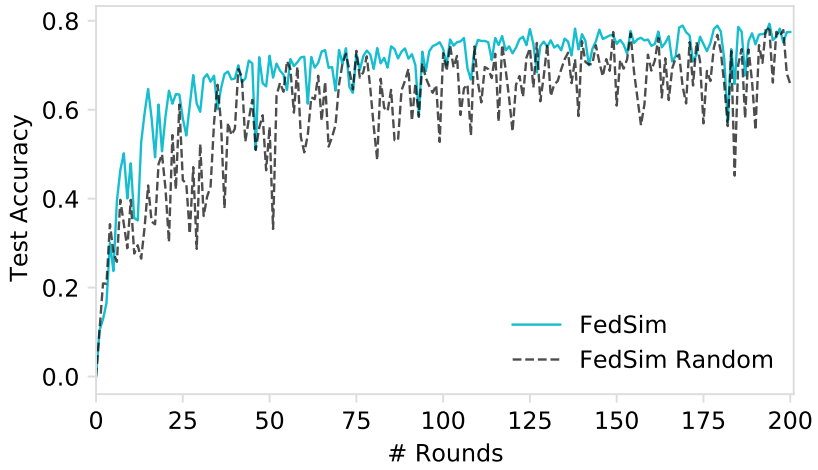

Figure 4.8: Comparison of similarity guided clustering vs random clustering on FEMNIST

While Figure 4.8 focuses on the FEMNIST dataset, similar experiments on MNIST, Fed-MEx, and Fed-Goodreads are presented in Appendix A, confirming the same findings.

Dimensionality Reduction with PCA

As discussed in section 4.3.1, dimensionality reduction with PCA minimises computation costs when computing similarity-based clusters. Figure 4.9 explains the level of compression that can be achieved with PCA applied to each real-world dataset.

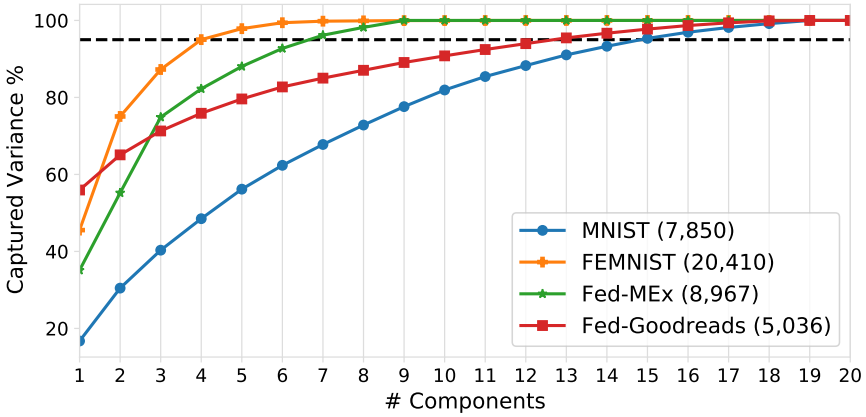

Figure 4.9: Analysis of variance captured by the number components when using PCA

The dotted line indicates the number of PCA coefficients that captured 95% of the variance of the original gradient and were selected for clustering. PCA reduces the gradient vector size to 4 and 15 with all datasets. The impact of dimensionality reduction to minimise computational cost is quantified by comparing the time elapsed for a communication round in *FedSim* with and without PCA. We found that a *FedSim* communication round with MNIST, FEMNIST, Fed-MEx and Fed-Goodreads is 230.0, 469.1, 19.3 and 92.9 milliseconds, faster than without PCA. This improvement will have a significant impact on performance in production environments.

4.5.2 Comparative Study with Synthetic Datasets

A further investigation was conducted to understand the performance of *FedSim* and the baselines under different levels of controlled IID-ness. For this purpose, we used five synthetic datasets with varying levels of statistical heterogeneity. By increasing α, we increased the class distribution shift by varying the standard deviation for sampling the weights that control the class label generation model. Similarly, by increasing β, we shifted the feature distributions between clients, thereby increasing the levels of non-IIDness through features. In practice, real-world FL datasets are unlikely to be completely non-IID (i.e., α and β equal to one) in terms of having clients with both unique feature and class distributions. In contrast, an IID dataset will have no feature distribution shift and use the same class label generation model to generate client data, resulting in highly similar clients.

Figure 4.10 plots the test accuracy measures over the communication rounds to investigate the performance stability of *FedSim* compared to *FedProx* and *FedAvg*. The similarity guided, *FedSim*, has achieved increased performance stability on *synthetic* datasets (0,0), (0.25,0.25) and (0.5,0.5) which are considered to be moderately non-IID. For datasets (0.75,0.75), (1,1) that are highly non-IID, *FedSim* fails to outperform *FedProx*; however, *FedSim* is significantly stable compared to *FedAvg*. Similarly, in the IID setting, *FedSim* fails to outperform *FedAvg*; however, *FedSim* significantly outperforms *FedProx*.

A summary of results is presented in Table 4.6, showing the mean test accuracy improvement (as a percentage) achieved by *FedSim*, over *FedAvg* and *FedProx* over 35 trials involving 100 communication rounds. *FedProx*, optimised for non-IID settings, has significantly poor performance in IID settings (-20.92% with *synthetic IID* dataset). In comparison, *FedSim* has only a 8.92% drop in performance over *FedAvg* compared to

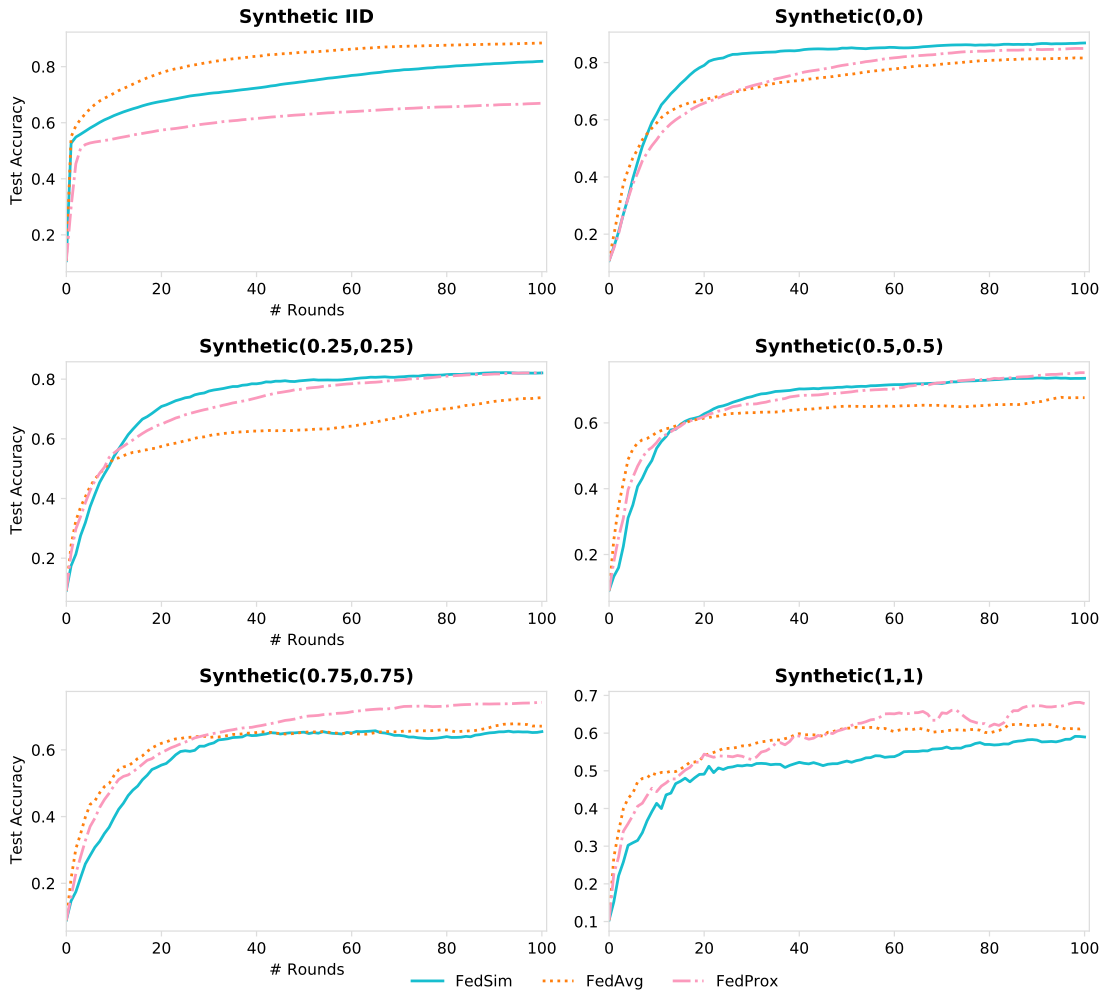

Figure 4.10: Comparison of performances over communication rounds with *synthetic* datasets to study the effect of statistical heterogeneity and similarity

that of 11.99% with *FedProx*. We expect that the IID situation will not benefit from clustering since all clients are likely to be similar and could instead be treated as members of one and the same cluster. *FedProx* achieves performance improvements with all five non-IID datasets compared to *FedAvg*. Similarly, three of the five non-IID datasets record performance improvements with *FedSim*. Notably, with the most extreme non-IID synthetic dataset, *FedSim* has failed to outperform *FedAvg* and records a performance reduction of 6.18%.

Overall, moderate non-IID settings benefit from a similarity-guided approach to FL.

Table 4.6: Comparison of overall performance improvements of *FedSim* over baselines

Synthetic Dataset	*FedSim* improvement over *FedAvg* (%)	*FedSim* improvement over *FedProx* (%)
Synthetic IID	-8.92 ±2.04	**11.99**±2.90
Synthetic(0,0)	**6.93**±4.78	**5.83**±4.17
Synthetic(0.25,0.25)	**11.21**±6.39	**1.87**±3.01
Synthetic(0.5,0.5)	**3.61**±6.23	-0.06±2.47
Synthetic(0.75,0.75)	-3.23±4.16	-6.22±2.74
Synthetic(1,1)	-6.18±2.83	-6.98±2.83

Additionally, *FedSim* performs comparably well in both IID and extreme non-IID settings. In moderate non-IID settings, *FedSim* exploits latent client similarities to improve performance. However, in the IID setting, *FedAvg*'s improvement over *FedSim* can be explained by observing that *FedSim* with $|C| = 1$ is equivalent to *FedAvg*. This suggests that when inter-client similarities are high, forming fewer clusters for aggregation is better. Currently, *FedSim* maintains a fixed cluster size and would need to reduce the number of clusters in an IID setting to achieve performance comparable to *FedAvg*. In an extreme non-IID setting where similarities are minimal to non-existent, *FedProx* uses proximity regularisation in its weight update step, enabling better adaptation to the setting and achieving superior performance.

Measuring Statistical Heterogeneity

Finally, to study if *PNI* correlates with known factors that cause statistical heterogeneity, we use the methods described in Shamir et al. (2014) to create 11 synthetic non-IID datasets by changing α incrementally from 0 to 1 (α controls the variation in class distributions among clients). Then, three variants of each dataset are created for β values 0, 0.5 and 1 (β controls the variation in feature distributions).

Figure 4.11 plots the mean *PNI* values obtained for the 11 datasets and their variants from 100 repeated experiments (with 100 random seeds). Overall, *PNI* values consistently increased with β, demonstrating that *PNI* is capturing the heterogeneity in feature distributions among clients.

It is reassuring that the *PNI* measure, validated in this controlled setting with synthetic data, shows increasing values with increasingly heterogeneous feature distributions. This

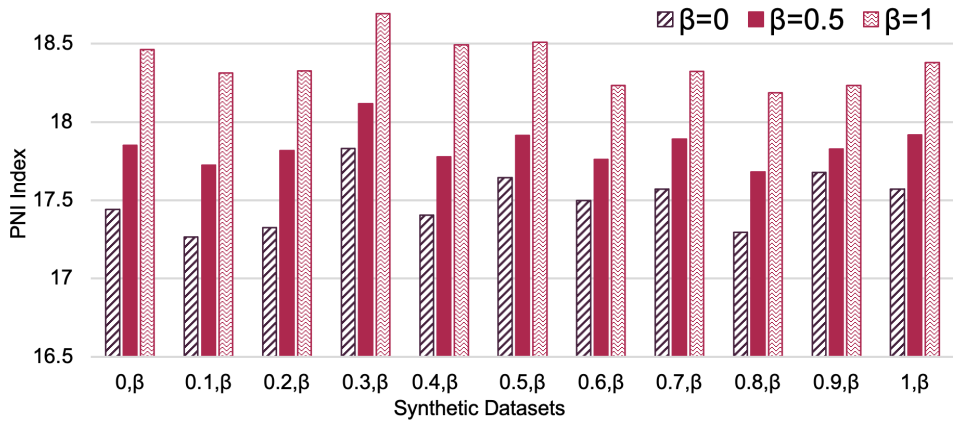

Figure 4.11: Comparison of *PNI* values across different feature distributions

suggests that *PNI* could be used to characterise real-world datasets where feature distributions are not explicitly controlled.

4.6 Chapter Summary

In this chapter, we presented the *FedSim* aggregation strategy for FL. We aimed to leverage inter-client relationships (i.e., similarity knowledge) to enhance model aggregation. A comprehensive evaluation across multiple application domains using real-world datasets (contributing two new FL datasets) and synthetic datasets demonstrated that *FedSim* outperforms the *FedAvg* and *FedProx* baselines when similarity knowledge is utilised.

Results with real-world datasets confirmed that *FedSim* effectively captures similarity knowledge among clients, leading to significantly better performance in model aggregation. Our findings also confirm the generalisability of *FedSim* with alternative neural models and optimisation algorithms. To explore which settings are best suited for *FedSim*, we conducted experiments with six synthetic datasets featuring IID and multiple variants of non-IID distributions. Significant performance improvements were observed with *FedSim* on various synthetic dataset variants, except in the IID and extreme non-IID settings. It is reassuring that the *PNI* measure, validated in this controlled setting with synthetic data, shows increasing values with increasingly heterogeneous feature distributions. This suggests that *PNI* could be used to characterise real-world datasets where feature distributions are not explicitly controlled.

FedSim faced performance limitations in extreme cases of non-IIDness where insufficient

similarity knowledge is found and in highly IID settings where every client is similar. Additionally, the method's reliance on fixed cluster sizes may hinder adaptability across varying datasets, as selecting an optimal cluster size requires domain-specific knowledge. Another limitation is the lack of a mechanism to dynamically switch between aggregation methods, which could potentially enhance performance depending on dataset characteristics. To address these limitations, future work could focus on developing adaptive clustering techniques to adjust cluster sizes dynamically and implementing flexible aggregation methods that can switch based on client data.

Chapter 5

FedFT: Improving Communication Performance for Federated Learning with Frequency Space Transformation

"**Out of clutter, find simplicity.**
From discord, find harmony. In the middle of difficulty lies opportunity."

Albert Einstein

In the Section 2, we discussed several major challenges in FL. One of the primary challenges is the cost of communication. Due to FL's distributed nature, there is a significant communication bandwidth requirement that must be managed across the network. This demand affects both client-side and server-side capacities. Implementing FL systems at a larger scale can lead to significant communication overhead reaching hundreds of gigabytes. This can create challenges for practical and efficient implementation. Addressing this issue is crucial because even minor communication efficiency improvements can significantly impact. This chapter focuses on addressing the second research question (RQ2): *Building on the insights from RQ1, how does the chosen aggregation strategy impact communication efficiency in FL, and in what ways can model compression and pruning enhance this efficiency?*. To address RQ2, we focus on Objective O4: *Develop an FL*

algorithm to improve communication performance, ensuring adaptability across diverse FL scenarios. This chapter introduces *FedFT* (Federated Frequency-space Transformation), a simple yet effective methodology for communicating model parameters in an FL setting. *FedFT* uses DCT to represent model parameters in frequency space, enabling efficient compression and reducing communication overhead. *FedFT* is compatible with various existing FL methodologies and neural architectures.

5.1 Background and Use Case

Central to FL is its decentralised training of a shared global model with many communication exchanges between the server and its clients. At each communication round, the server updates the shared model as an aggregation of client models received. The FL communication layer needs to handle many requirements due to its iterative nature, which involves frequent and large model exchanges. Inefficient communication can slow training, increase computational cost, decrease accuracy, raise energy consumption, and limit scalability (Konečnỳ et al., 2016). Compression can be used to improve communication performance by reducing the amount of data being transmitted. For instance, ML applications can optimise storage and inference speed using transformation methods like DCT, as demonstrated in Liu et al. (2018). DCT operates by converting model parameters into the frequency domain, after which quantisation and pruning can be applied to discard less significant coefficients. This results in a more compact representation of the model optimising both storage requirements and computational efficiency during inference.

In FL research, although the use of DCT has been acknowledged, the main focus has been on using it to compress training data for better local representation on client devices (Chen and Koushanfar, 2022; Han et al., 2021). We use the term "tensor space" to distinguish the space in which model parameters are represented from that in which training data is represented. The example in Figure 5.1 compares model weight representation in tensor and frequency spaces.

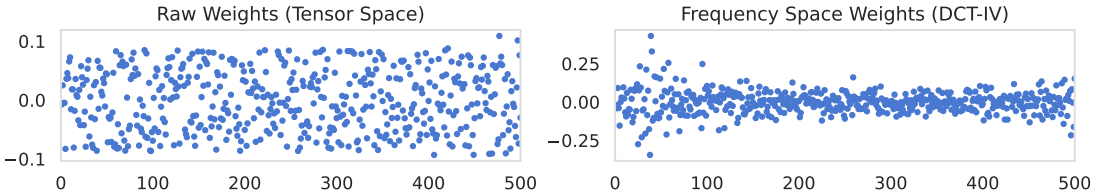

Figure 5.1: Model parameters represented in tensor space and frequency space.

In the frequency space, the weights decomposed into their constituent frequencies are more spread out and concentrated to a few dominant frequency components, allowing for efficient representation and storage. One of the challenges facing FL using tensor space compression is ensuring that compression techniques do not obstruct server-side aggregation operations. Most FL methods address this challenge by using lossless compression techniques and incorporate an extra step of reconstructing the tensor space at the server for model aggregation prior to compressing it again for transmission back to the clients (Dai et al., 2019; Sattler et al., 2020b). In this Chapter, we propose a methodology, *FedFT*, that enables server aggregation in the same compressed space. To achieve this, we investigate the feasibility of using DCT-transformed model parameters in the communication layer of FL to enhance communication performance without sacrificing model accuracy.

The direct advantage of *FedFT* is that it enables the sharing of model parameters in the frequency domain, and local client updates can be done in either the frequency space or tensor space, making it adaptable across different methodologies. Furthermore, the compact representations in frequency space make it simple for clients to identify sparse areas that could be easily pruned before communicating them to the central server.

Accordingly, we make three contributions.

- Introducing *FedFT*, a novel FL methodology that utilises frequency space transformation to improve communication efficiency while preserving performance.
- Conducting a comparative study with state-of-the-art FL methodologies to demonstrate the generalisability of the frequency space transformation and assess the trade-off between model performance and communication efficiency.
- Demonstrating the generalisability of *FedFT* by analysing evaluation results from various neural architectures for image, text, and sensor data.

5.2 *FedFT* Methodology

FedFT aims to improve communication efficiency in FL by utilising the frequency space transformations on the model parameters. The overall *FedFT*, client and server communication setting is presented in Figure 5.2. The original *FedAvg* phases are shown as grey-filled rectangles, i.e., the initialisation, local update, and federated aggregation.

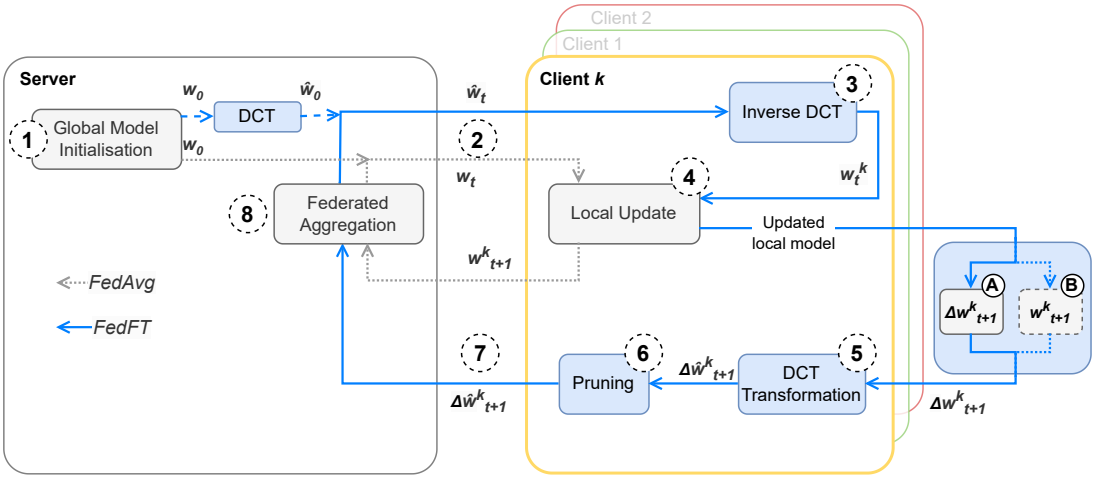

Figure 5.2: Proposed *FedFT* methodology

Here, Steps 2 and 7 refer to downstream and upstream communications forms. Contributions of *FedFT* are the blue-filled rectangles. The blue and grey communication lines differentiate steps in relation to *FedFT* and *FedAvg* respectively.

Next, we explain how to use the frequency space for communicating with *FedFT*, including its integration into the FL methodology. In the remainder of this section we explore the additional steps needed to use the frequency space for communication with *FedFT* and suggest how it can be smoothly integrated into the general FL methodology.

5.2.1 Global Model Initialisation

The first step in Figure 5.2 is the initialisation of the global model, w_0, at $t = 0$, which is familiar to both *FedFT* and *FedAvg*. Additionally, *FedFT* converts w_0 into the frequency space using a transformation function, T, to obtain $\hat{w}_0$, which is communicated to all clients. *FedFT* can be applied even if the initial global model is pre-trained, such as a language model (Tian et al., 2022) or transferred from another domain (Florescu et al., 2022), by converting the pre-trained weights into the frequency space.

5.2.2 Communication

The communication of model parameters in FL happens in two directions: from server to client (downstream) and from client to server (upstream). In both cases, *FedFT* communicates model parameters in the frequency space using DCT-IV, a linear lossy function further discussed in Section 5.3.

5.2.3 Client Local Update

Local update for a supervised task typically employs stochastic gradient descent (SGD) over several epochs using local training data (Step 4 in Figure 5.2). In *FedAvg* this local update is applied to the model received through downstream communication from the server. With *FedFT*, the downstream communications of the initial and follow-on models, w_0 and w_t; are communicated in the frequency space, as transformed models, $\hat{w}_0$ and $\hat{w}_t$; accordingly, an additional step of inverse transform, $\hat{T}$, is required, where $\hat{T}$ reconstructs the model parameters from the frequency space to tensor space where local model updates can occur. We acknowledge the possibility of performing these updates in the frequency space (Liu et al., 2018).

However, we have chosen to maintain our approach, which helps to evaluate communication efficiency in isolation and enables us to assess *FedFT* on a diverse set of federated methodologies (*FedAvg*, *FedProx* and *FedSim*) and neural models, all of which commonly operate in tensor space. Our research examines two methods for representing locally updated models before transforming them into the frequency space (using T) for upstream communication to the server in Step 5 of Figure 5.2. In the figure, these alternative routes are labelled as (A) and (B) and refer to the following:

Difference model (A)

This method captures only the net changes from local training. The server can update the global model by adding these differences to its existing version, thus efficiently reconstructing the complete model. Where updated local model parameters w_{t+1}^k are compared against the received global model w_t^k and the differences ($\Delta w_{t+1}^k = w_{t+1}^k - w_t^k$) are transformed into the frequency space and communicated to the server. This is similar to the FL methodologies where client model update differences are communicated to the server Sattler et al. (2020b), except we do so in the frequency space.

Complete model (B)

If the objective is to conserve computational resources on the server when handling incoming updated models, opting to send the complete model is advantageous. However, this involves sending more parameters from each client which restricts the potential for compacting the models for efficient communication. w_{t+1}^k is transformed into the frequency space and communicated to the server. This is simply the general FL methodology from McMahan et al. (2017).

We present the case for why $\Delta\hat{w}_{t+1}^k$ (difference model) is a more favourable choice compared to $\hat{w}_{t+1}^k$ (complete model) in Section 5.3.

5.2.4 Pruning of Model Parameters

Pruning allows FL to operate at varying compression levels, thereby improving the efficiency of upstream communication. With *FedFT*, we can implement pruning at Step 6 in Figure 5.2, i.e. after performing the DCT transformation but before the upstream communication (Step 7). The parameters pruned are the least significant coefficients of the updated client model in the frequency space (either $\hat{w}_{t+1}^k$ or $\Delta\hat{w}_{t+1}^k$). In the case of *FedFT*, pruning on DCT coefficients results in lossy compression where it approximates and discards some of the less significant frequency coefficients. Optimised compression with DCT is possible when many model parameters are captured within low-frequency coefficients.

Pruning becomes an effective technique when the magnitudes of a specified percentage of high-frequency coefficients are set to 0 while minimising the reconstruction error. This is because the high-frequency coefficients often correspond to features with high variance, i.e., noisy information. The pruning function and percentage are called $P(.)$ and α. Pruning can be applied once convergence is close, at this point, most of the model will be contained within a low-variance. The implications of pruning in the frequency space are discussed in Section 5.3 with empirical findings in Section 5.5.7.

Figure 5.3 provides a visual representation of the pruned frequency space weights using DCT-IV. In this process, low-frequency weights are set to zero, transmitting a condensed model to the server and enhancing communication efficiency.

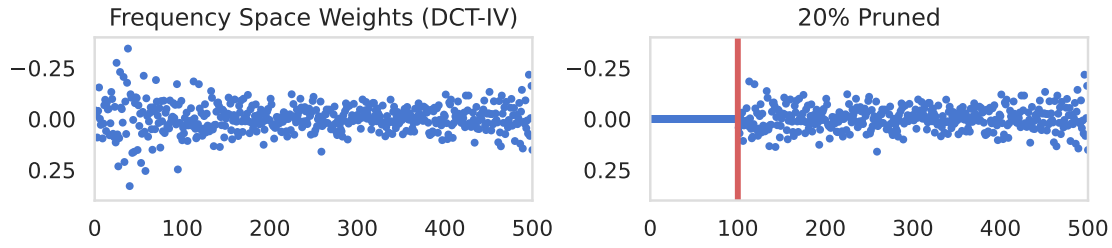

Figure 5.3: Visual representation of pruning

5.2.5 Federated Aggregation

A linear transformation function such as DCT-*IV* is useful for performing federated aggregation in the frequency space. If the transformation was non-linear this would require additional inverse transformations at the server to reconstruct the models in tensor space before federated aggregation can be performed and transformed for downstream communication. Using DCT as the T function enables *FedFT* to aggregate in the frequency space. It can do so with either the Difference client models (see Equation 5.1 with $\Delta\hat{w}^k_{t+1}$) or Complete client models (see Equation 5.2 with $\hat{w}^k_{t+1}$) based on the selected approach for the local update step.

$$\hat{w}_{t+1} \leftarrow \sum_{k \in K} \frac{n_k}{n} (\hat{w}_t + \Delta\hat{w}^k_{t+1}) \tag{5.1}$$

$$\hat{w}_{t+1} \leftarrow \sum_{k \in K} \frac{n_k}{n} \hat{w}^k_{t+1} \tag{5.2}$$

In Equation 5.1, the calculation of the weighted average includes the addition of the model changes to the previously maintained model on the server, which distinguishes it from the other (Equation 5.2).

5.2.6 *FedFT* algorithm

Algorithm 4 brings together the extensions proposed by the *FedFT* methodology. The algorithm text highlighted in blue text signifies the modifications we have implemented to adapt our proposed method to the vanilla *FedAvg* methodology. Line 4 performs the initial global model transformation into the frequency space, once received by clients each performs the inverse transformation in Line 9, prior to carrying out the local update. Once completed, the client calculates the Δw^k_{t+1} (Line 11), and performs the frequency space transformation and pruning with the percentage of pruning controlled by α (Line 12). Once the client models in the frequency space $\Delta\hat{w}^k_{t+1}$ are communicated to the server, it performs federated aggregation on the updated local models in the frequency space.

Algorithm 4 *FedFT*

Require: w_0: initial global model, α: Pruning Rate, K: number of selected clients per round
Require: $T(.)$ DCT Function, $\hat{T}(.)$ Inverse DCT Function, $P(.)$ Pruning Function
$\hat{w_0} = T(w_0) \leftarrow$ DCT transformation
for t=0,1,2, ... **do**
  Broadcast $\hat{w}_t$ to all clients
  Select K clients
  for all $k \in K$ **do**
    $w_t^k = \hat{T}(\hat{w}_t) \leftarrow$ inverse DCT transform
    $w_{t+1}^k \leftarrow$ update w_t^k using SGD on client data
    $\Delta w_{t+1}^k = w_{t+1}^k - w_t^k \leftarrow$ update differences
    $\Delta \hat{w}_{t+1}^k = P(T(\Delta w_{t+1}^k), \alpha) \leftarrow$ DCT transform and prune
    Send $\Delta \hat{w}_{t+1}^k$ to the server
  end for
  $\hat{w}_{t+1} \leftarrow \sum_{k \in K} \frac{n_k}{n} (\hat{w}_t + \Delta \hat{w}_{t+1}^k) \leftarrow$ Federated Aggregation on update differences
end for

5.3 Role of Model Variance for Transformed Communication

Based on an literature analysis (see Section 5.1), we select DCT as the transformation technique to convert w into the frequency space. Where a given set of model parameters, w is a multi-dimensional array (i.e. a tensor) where the number of dimensions depends on the model architecture. Out of the DCT variants, DCT-IV (Britanak, 2003) is selected due to its linear, orthogonal and symmetric properties required for inverse transformations and necessary for federated aggregation.

Equation 5.3 presents the DCT-IV transformation function $T(.)$ for w represented in a tensor space of $\mathbb{R}^{N \times M}$, where $k \in \{0, \ldots, N-1\}$ and $l \in \{0, \ldots, M-1\}$ respectively.

$$\hat{w}_{k,l} = \sum_{n=0}^{N-1} \sum_{m=0}^{M-1} w_{n,m} \cos\left(\frac{\pi(2m+1)(2k+1)}{4N}\right) \cos\left(\frac{\pi(2n+1)(2l+1)}{4M}\right) \tag{5.3}$$

Without loss of generalisability, w represents a set of model parameters between two fully connected layers of a neural architecture. With multi-dimensional tensors, beyond just 2-dimensions, the summations can be extended over the additional dimensions.

Equation 5.4 is the inverse transformation function $\hat{T}(.)$, where $n \in \{0, \ldots, N-1\}$ and

$m \in \{0, \ldots, M-1\}$.

$$w'_{n,m} = \frac{2}{N} \sum_{k=0}^{N-1} \sum_{l=0}^{M-1} \hat{w}_{k,l} \cos\left(\frac{\pi(2m+1)(2k+1)}{4N}\right) \cos\left(\frac{\pi(2n+1)(2l+1)}{4M}\right) \tag{5.4}$$

Accordingly, the reconstruction loss is calculated as $|\hat{T}(T(w)) - w|$.

The distribution of the tensor space directly impacts the magnitude of the DCT coefficients and how they are distributed. This in turn affects the level of pruning possible to manage reconstruction error after the inverse transform (Lam and Goodman, 2000). We observe the distribution of the tensor space conforms to a Gaussian distribution which can be expressed using mean and variance (Figure 5.4). Accordingly, the variance of model parameters, $w^k \in \mathbb{R}^{N \times M}$, for any given round is calculated as in Equation 5.5, where $\bar{w}^k$ indicates the mean of model parameters.

$$Var(w^k) = \frac{1}{N \times M} \sum_{n=0}^{N-1} \sum_{m=0}^{M-1} (w^k_{n,m} - \bar{w}^k)^2 \tag{5.5}$$

Similarly, the variance of the difference model, $\Delta w^k \in \mathbb{R}^{N \times M}$, can be calculated as in Equation 5.6.

$$Var(\Delta w^k) = \frac{1}{N \times M} \sum_{n=0}^{N-1} \sum_{m=0}^{M-1} (\Delta w^k_{n,m} - \bar{\Delta w}^k)^2 \tag{5.6}$$

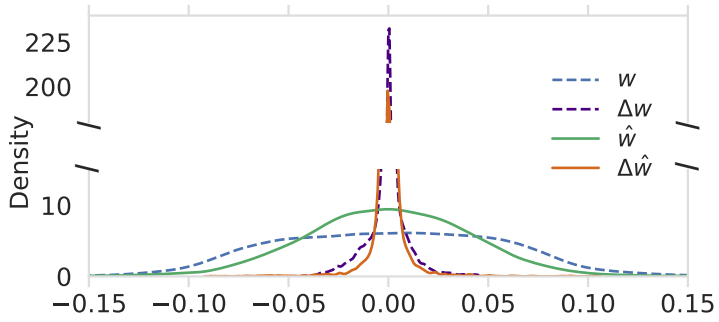

Figure 5.4: Density in tensor and frequency spaces

We conducted an empirical study to better understand these distributional relationships between tensors and how it transforms into the frequency space in the context of variance. The following observations were made: the variance of $Var(\Delta w^k)$ remains consistently below that of $Var(w^k)$ throughout the communication rounds; tensor space (for both

w and Δw) conform to a Gaussian distribution (Figure 5.4); Further the corresponding frequency space in the form of DCT coefficients (for both $\hat{w}$ and $\Delta\hat{w}$) also conform to a Gaussian distribution (Figure 5.4); frequency space has lower variances, compared to tensor space under the strict constraint that w is a set of model parameters that are optimised using SGD. Accordingly, at any given round, it is reasonable to assume that the inequalities between the variances in the tensor space are also likely to hold in the frequency space (Equation 5.7).

$$Var(\Delta w^k) < Var(w^k) \iff Var(\Delta\hat{w}^k) < Var(\hat{w}^k) \tag{5.7}$$

We study the link between variance and reconstruction error in Figure 5.5. The plots show five synthetic Gaussian distributions, each with 0 mean and 10,000 samples for increasing variances (a) and their corresponding reconstruction errors (b), the x-axis is the variance, and the y-axis is the reconstruction error. It is clear from these plots that there is a direct relationship between increasing variance in distributions and increasing reconstruction errors. This confirms the benefits of using the difference model over the complete model and highlights the advantages of reduced variance in the frequency space for optimising compression in communication.

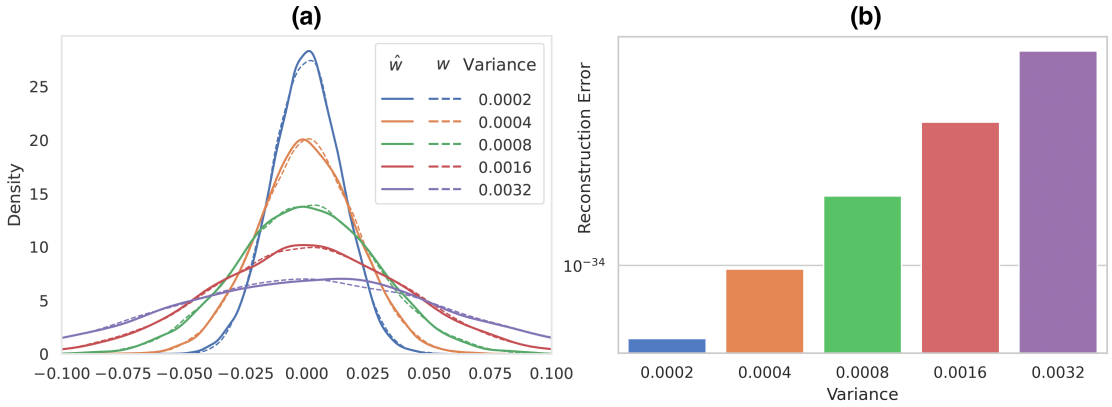

Figure 5.5: Variance and reconstruction error relationship

Finally in Figure 5.6 we analyse how pruning affects model parameters in the frequency space. Here the variances in the y-axis are in log scale and the x-axis is communication rounds. This plot further verifies the assertion made in Equation 5.7 that in the frequency space, the variance of the difference model is less than that of the complete model (blue

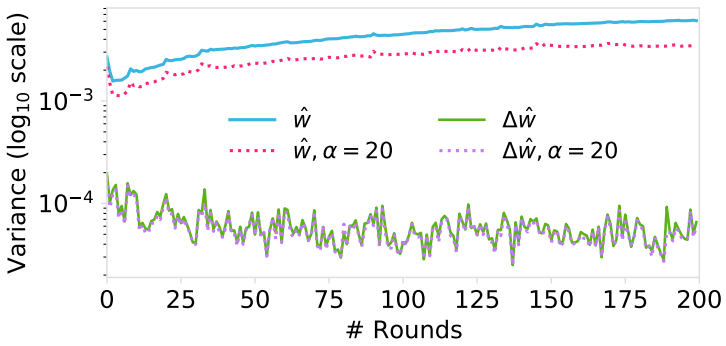

Figure 5.6: Variance in frequency space and pruning

and green lines). We use variance here as a proxy for reconstruction error, where increasing variance (and so increasing reconstruction error) indicates the diminishing utility of pruning.

Accordingly, we use a pruning rate, $\alpha = 20\%$, to study the impact of pruning less significant coefficients in the frequency space. We can see that pruning the difference ($\Delta\hat{w}$) results in hardly any drop in variance. In contrast, a noticeable drop in variance is observed when using the complete model ($\hat{w}$). We can conclude from these empirical observations, that utilising the Difference model in the frequency space, $\Delta\hat{w}$, for *FedFT* will yield better results as compared to using the complete model, as stated in Equation 5.1 vs Equation 5.2 and in Algorithm 4. We will use this version of *FedFT* in our comparative studies next.

5.4 Experiment Setup

We evaluate the performance of *FedFT*, with respect to three important aspects. First, its generalisability to existing Federated Learning baseline methodologies. Second, we investigate its applicability to various complex neural architectures. Finally, we analyse the impact of pruning with *FedFT* on performance and communication efficiency. The generalisability of *FedFT* is evaluated with four real-world datasets as described in Section 3.1. MNIST, FEMNIST, Fed-Goodreads, and Fed-MEx are used. *FedAvg*, *FedProx*, and *FedSim* are used as baseline methods for comparison. Similar to the approach in Algorithm 4, where *FedFT* was implemented with FedAvg, Algorithms 5 and 6 detail the application of *FedFT* within the *FedSim* and *FedProx* methods, respectively.

Algorithm 5 *FedFT* adaptation of *FedSim* (Introduced in Chapter 4)

Require: w_0: initial global model, α: Pruning Rate, K: num. of selected clients
Require: $T(.)$ DCT Function, $\hat{T}(.)$ Inverse DCT Function, $P(.)$ Pruning Function
$\hat{w}_0 = T(w_0) \leftarrow$ DCT transformation
for t=1,2,.. **do**
  Broadcast $\hat{w}_t$ to all clients
  Select $\mathcal{S}$ clients where $\mathcal{S} \subset \mathcal{K}$
  $\mathcal{C} \leftarrow$ Clustering(S, $n\_clusters$)
  for all $c \in \mathcal{C}$ **do**
    for all $k \in c$ **do**
      $w_t^k = \hat{T}(\hat{w}_t) \leftarrow$ inverse DCT transform
      $w_{t+1}^k \leftarrow$ updates w_t^k using SGD
      $\Delta w_{t+1}^k = w_{t+1}^k - w_t^k \leftarrow$ *update differences*
      $\Delta \hat{w}_{t+1}^k = P(T(\Delta w_{t+1}^k), \alpha) \leftarrow$DCT transform and prune
      Send $\Delta \hat{w}_{t+1}^k$ to the server
    end for
    $\hat{\bar{w}}_{t+1}^c \leftarrow$ ClusterAggregation($\{\hat{w}_t + \Delta \hat{w}_{t+1}^k\}\ \forall\ k\ \in\ c$)
  end for
  $\hat{w}_{t+1} \leftarrow$ GlobalAggregation($\{\hat{\bar{w}}_{t+1}^c\}\ \forall\ c\ \in\ \mathcal{C}$)
end for

Algorithm 6 *FedFT* adaptation of *FedProx*

Require: w_0: initial global model, α: Pruning Rate, K: num. of selected clients
Require: $T(.)$ DCT Function, $\hat{T}(.)$ Inverse DCT Function, $P(.)$ Pruning Function
$\hat{w}_0 = T(w_0) \leftarrow$ DCT transformation
for t=0,1,2, ... **do**
  Broadcast $\hat{w}_t$ to all clients
  Select K clients with probability p_k
  for all $k \in K$ **do**
    $w_t^k = \hat{T}(\hat{w}_t) \leftarrow$ inverse DCT transform
    $w_{t+1}^k \leftarrow$ update w_t^k using $F_k(w) + \frac{\mu}{2}\|w - w^t\|^2$ (Li et al. (2018))
    $\Delta w_{t+1}^k = w_{t+1}^k - w_t^k \leftarrow$ *update differences*
    $\Delta \hat{w}_{t+1}^k = P(T(\Delta w_{t+1}^k), \alpha) \leftarrow$ DCT transform and prune
    Send $\Delta \hat{w}_{t+1}^k$ to the server
  end for
  $\hat{w}_{t+1} \leftarrow \sum_{k=1}^{K} \frac{n_k}{n}(\hat{w}_t + \Delta \hat{w}_{t+1}^k)$ *Federated Aggregation on update differences*
  Set PGD Parameters $\leftarrow \hat{T}(\hat{w}_{t+1})$
end for

We use the test accuracy described in Section 3.3.1 for evaluation. Additionally, the experiments are carried out with 35 random seeds (ranging from 0 to 34) to demonstrate significance and robustness. We measure the upstream communication cost accumulated

over t communication rounds per client as $t \cdot \Theta(P(T(w), \alpha))$. The setup and hyper-parameters are described in Section 3.2. The source code for the experiment setup is available on GitHub [1].

5.4.1 Overview of Experiments

In Table 5.1, we present a comprehensive summary of the experiments carried out in this study. Detailed descriptions of each experiment are provided in the corresponding subsections. The table showcases the range of datasets, baseline methodologies, and neural architectures employed in our experiments.

Impact of *FedFT* Pruning

FedFT applies pruning to improve communication efficiency which is lossy and can impact overall performance. Accordingly, we explore the performance impact of pruning with MLR models trained on four datasets with increasing α rates. We explore two variants of pruning: one applied from the start of communication (round=0) and the other applied after the model has converged (round$\sim$50). In each case, we compare pruning rates where α varies from 0% (no pruning) to $\sim 50\%$ in increments of $\sim 10\%$. The actual percentages for MLR models depend on the output layer size; for example, on Fed-MEx, where $|\hat{w}| = [1280, 7]$, $\alpha =\sim 14\%, \sim 29\%, \sim 43\%$ *and* $\sim 57\%$ for when 1,2,3, and 4 weights are set to 0 in each of 7 weights. Each experiment plots the test accuracy over communication rounds. Furthermore, we evaluate how pruning impacts communication efficiency by plotting the cumulative communication cost in Megabytes (MB) over 200 rounds for each dataset. This is repeated for all values of α to determine the optimal value that can maintain test accuracy (as close to accuracy with no pruning, i.e., when $\alpha = 0$) while minimising the cost in MB.

Analysing the Impact of non-IID on *FedFT*

This experiment evaluates the influence of non-IIDness on the effectiveness of the proposed *FedFT* method. The datasets utilised in the *FedFT* experiments are carefully selected to reflect their realistic non-IID nature. These datasets are chosen based on previous research in FL, as discussed in Section 3.1.1. In this analysis, we employ the FEMNIST dataset as our core dataset. To evaluate the impact of *FedFT* across varying degrees of non-IID, we purpose three variants of the FEMNIST dataset: FEMNIST(1) with one class per client, FEMNIST(2) with two classes per client, and FEMNIST(3)

[1] https://github.com/chamathpali/FedFT

Table 5.1: Comprehensive summary of *FedFT* experiments across diverse datasets, baselines and model architectures

Experiment	**Objective**	**Setup**
Comparing frequency transformation methods	To select which frequency transformation method is suitable	Baseline: *FedAvg* Datasets: MNIST, FEMNIST, Fed-Goodreads and Fed-MEx Model: MLR
Comparison of different variants of DCT	To select which variant of DCT is most applicable	Baseline: *FedAvg* Datasets: MNIST, FEMNIST, Fed-Goodreads and Fed-MEx Model: MLR
Generalisability of *FedFT*	Study the applicability of *FedFT*	Datasets: MNIST, FEMNIST, Fed-Goodreads and Fed-MEx Model: MLR
Evaluation with different learning models	Study generalisability with different neural architectures	Datasets: MNIST, FEMNIST, Fed-Goodreads and Fed-MEx Models: CNN-2D, MLP-3 and RNN
Evaluation of communication cost	Effect on computation overheads	Datasets: MNIST, FEMNIST, Fed-Goodreads and Fed-MEx Model: MLR
Analysing statistical heterogeneity	Investigate the impact on *FedFT* with varying levels of statistical heterogeneity	Datasets: FEMNIST(1-3) Model: MLR
Impact of *FedFT* pruning and communication efficiency	Investigate the communication performance of *FedFT*	Datasets: MNIST, FEMNIST, Fed-Goodreads and Fed-MEx Model: MLR
Impact of *FedFT* pruning on *FedSim*	Investigate the impact of using *FedFT* on the *FedSim* method	Datasets: MNIST, FEMNIST, Fed-Goodreads and Fed-MEx Model: MLR
Impact of pruning post-convergence	Investigate the impact of using *FedFT* as a post-convergence method	Datasets: MNIST and Fed-MEx Model: MLR

with three classes per client. The default configuration of the FEMNIST dataset used for primary experiments typically consists of three classes per client.

5.5 Results and Discussion

In this section, we conduct a detailed analysis of the experimental results and discuss the findings.

5.5.1 Comparing Frequency Transformation Methods

We aim to optimise the function $T(.)$ for efficient communication in FL. To do this, we compare two well-known frequency transformation methods: DCT and FFT. These methods are vital for transforming model parameters into frequency space for *FedFT* as discussed in Section 5.2. We designed an experiment to test how well DCT, particularly DCT-IV, works compared to FFT in FL settings. Our comparison looks at essential factors for FL, including compression efficiency, information retention, and impact on the convergence rate of the learning process. This experiment is conducted using *FedAvg* baseline across 200 communication rounds. To ensure statistical robustness, we averaged the results over 35 separate runs; each initialised with a unique random seed.

As illustrated in Figure 5.7, our results demonstrate a notable performance differential between the two methodologies. DCT-IV emerges as a superior choice, offering significant advantages over FFT. While Figure 5.7 focuses on the MNIST dataset, equivalent experiments on FEMNIST, Fed-MEx and Fed-Goodreads are presented in Appendix B.1, confirming the same findings. These advantages are quantified regarding reduced communication overhead and enhanced model accuracy post-transformation. DCT-IV's superiority can be attributed to its inherent properties, which align well with the sparsity and locality of model parameters in FL scenarios.

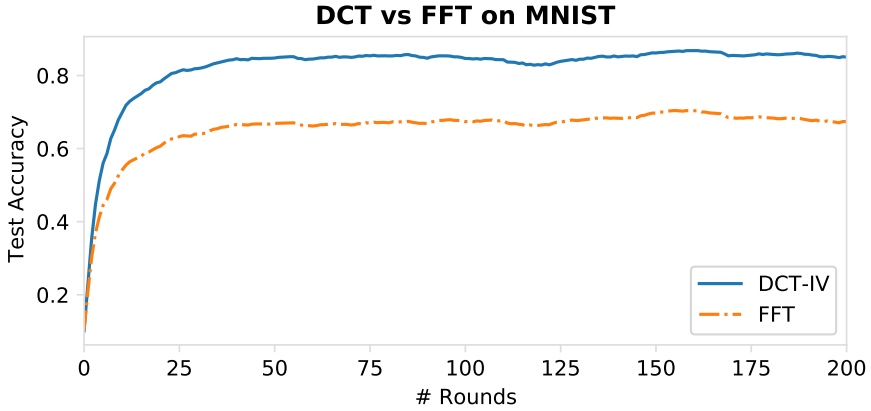

Figure 5.7: Comparison of DCT and FFT on MNIST dataset

5.5.2 Comparison of Different Variants of DCT

Evaluating the impact of various DCT variants is crucial as each variant has distinct characteristics and applications. This experiment is designed to discover which DCT variant is most suitable for our specific needs with FL. In Figure 5.8, we present a comparative analysis of four DCT variants, identified as DCT-I through DCT-IV. This experiment compares the *FedAvg* baseline with our proposed *FedFT* algorithm across 200 communication rounds, averaging the results across 35 unique runs.

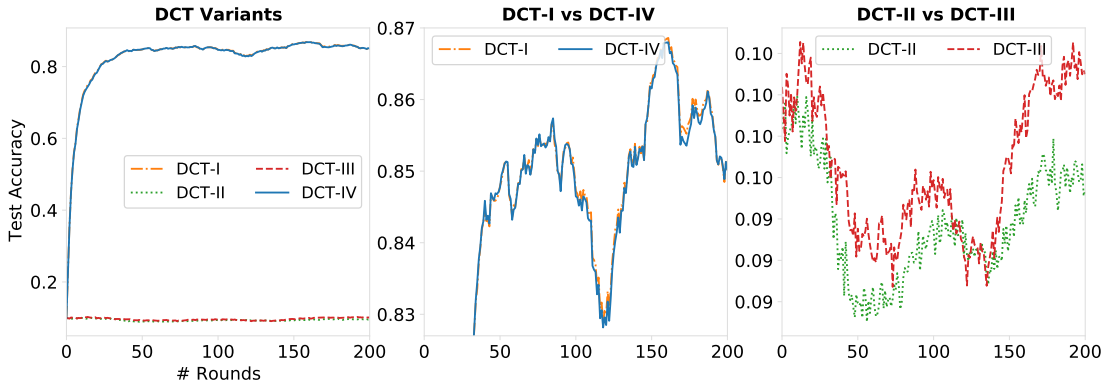

Figure 5.8: Comparison of DCT variants (I to IV) on MNIST with *FedFT* with *FedAvg*

This comparison is crucial in understanding how each variant handles the transformation and compression of model parameters. The results reveal a notable divergence in performance among these variants. Specifically, DCT-I and DCT-IV stand out for their efficiency, with lower reconstruction errors and accurate representations of the original model parameters. In contrast, DCT-II and DCT-III, while effective in their respective applications, show less favourable results in our context. Their performance is characterised by higher reconstruction errors, which suggests that they are not suitable for handling FL model parameters. While Figure 5.8 focuses on the MNIST dataset, equivalent experiments on FEMNIST, Fed-MEx and Fed-Goodreads are presented in Appendix B.2, confirming the same findings.

We have selected the DCT-IV variant for our transformation function $T(.)$, primarily due to its lower computational demands and proficiency in managing large data structures. This makes DCT-IV particularly well-suited for the diverse and computationally varied landscape of FL applications. All subsequent experiments in this study will utilise DCT-IV as the transformation function.

5.5.3 Generalisability of *FedFT*

Our primary experiments focus on assessing the generalisability of the proposed *FedFT* method. We evaluate the efficacy of *FedFT* across four datasets, comparing it with three state-of-the-art FL baselines. Figure 5.9 presents test accuracy results for increasing communication rounds with three FL methodologies, both with *FedFT* (solid line) and without *FedFT* (dotted line), across four datasets. Overall, *FedFT* adaptations match the performance of baseline counterparts at convergence, demonstrating that efficient communication of model parameters in frequency space does not compromise performance. The noticeable performance difference in the rounds prior to convergence across all methodologies on the Fed-MEx dataset is attributed to the small number of participating clients and their data sizes (30 total clients and ten selected per round). With fewer clients with fewer samples, each local update makes more extensive weight adjustments (high variance) resulting in significant changes to the global model.

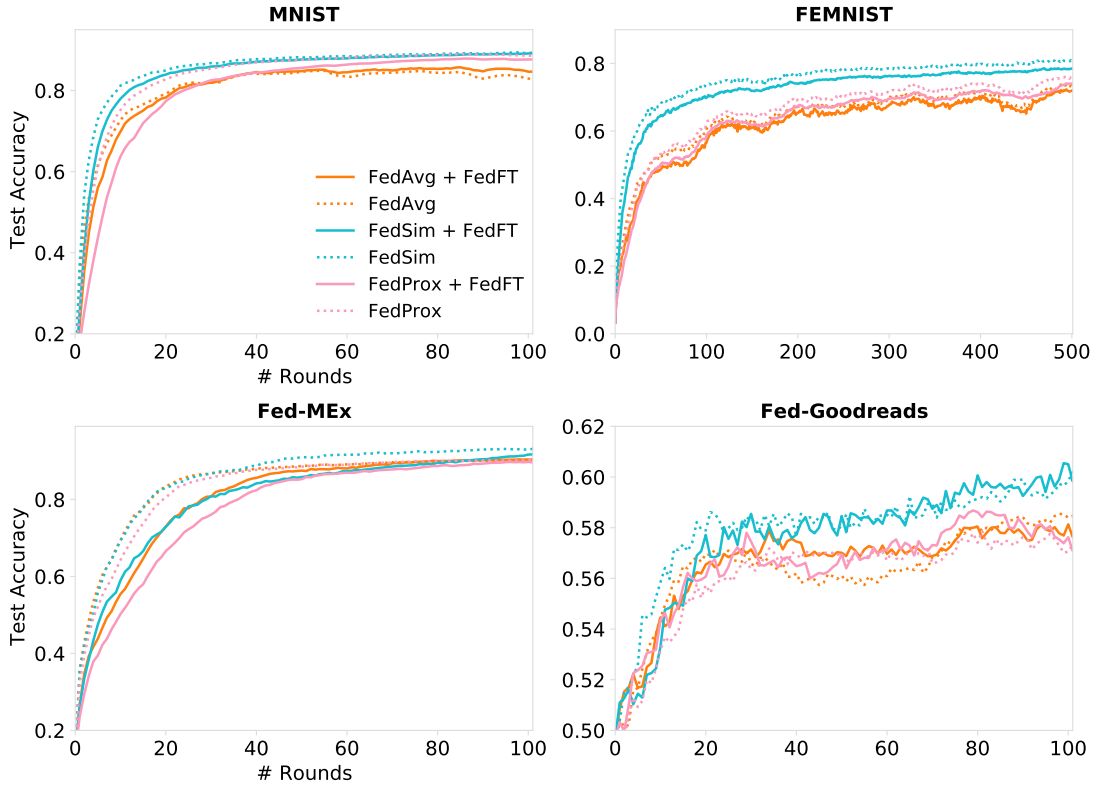

Figure 5.9: Comparison of *FedFT* with baselines FL methodologies

As discussed in Section 5.3, high variance results in high reconstruction error and affects

the model performance before convergence. The only significant performance loss post-convergence is observed with *FedProx* on Fed-Goodreads dataset where *FedFT* adaptation of *FedProx* fails to converge. We attribute this to the MLR classifier not being a suitable architecture; we recover this performance loss when using a recurrent neural model, which is better suited to textual content as shown in Section 5.5.4.

This study demonstrates the practicality of integrating *FedFT* into various FL methodologies and highlights its minimal impact on overall performance. This finding is significant as it highlights the adaptability and compatibility of *FedFT* with a wide array of FL methodologies and datasets. The results suggest that *FedFT* could be a valuable tool in improving communication efficiency and privacy in FL systems, offering a balance between efficiency and performance.

5.5.4 *FedFT* with Different Neural Architectures

To further understand the adaptability of *FedFT*, we explore its impact on different neural architectures. In Figure 5.10, we present a comparative analysis of *FedFT* and *FedAvg* when applied to different neural architectures.

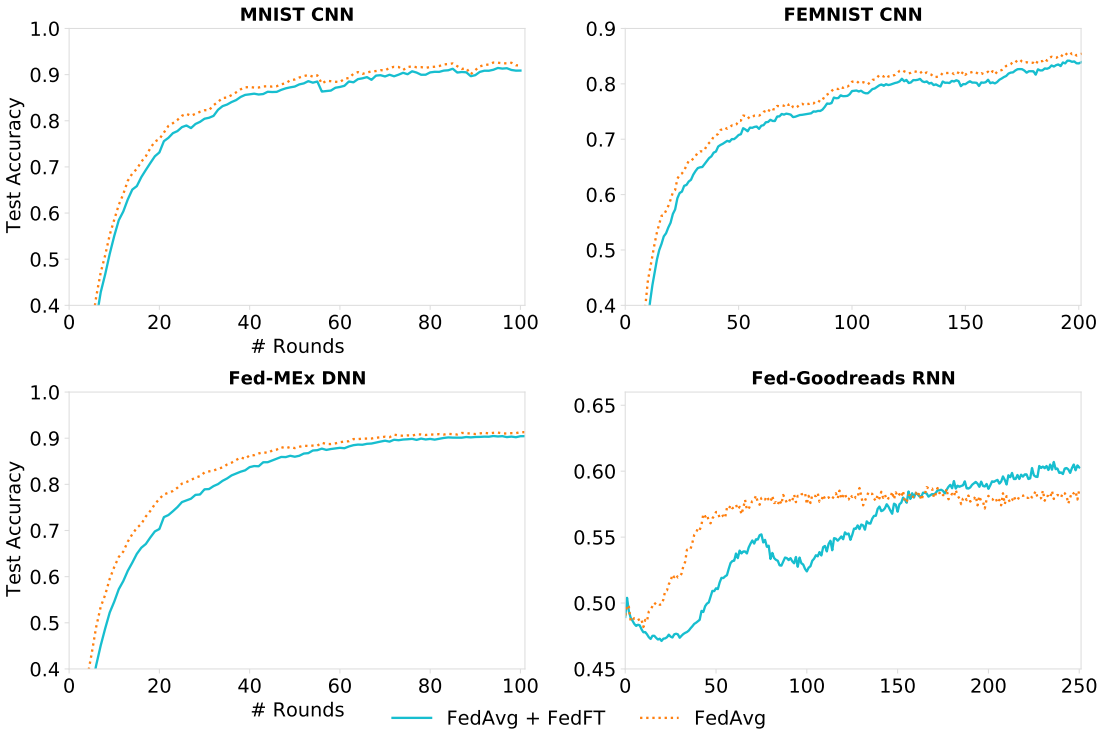

Figure 5.10: *FedFT* using different neural architectures

As illustrated in Figure 5.10, *FedFT* shows a slight performance drop across all datasets except for Fed-Goodreads, where it surprisingly outperforms *FedAvg*. In the MNIST dataset, the CNN model processes approximately 6.5 million parameters transformed between tensor space and the frequency space in each communication round, resulting in only a marginal drop in performance. This slight decline is similarly observed with the Fed-MEx dataset, which employs a DNN architecture. The RNN model trained on Fed-Goodreads with *FedFT* shows a drop in performance in early communication rounds. However, it improves and surpasses *FedAvg* performance after round ~ 150. We attribute this improved performance to the imperceptible reconstruction error in DCT-IV that is present even at 0% pruning, reducing noise for the federated aggregation. These results empirically support the selection of multi-dimensional DCT-IV for the transformation.

5.5.5 Effect of *FedFT* on Computation Overheads

Understanding the computation overheads is essential for FL methods, particularly in environments with limited computing resources. To assess the impact of *FedFT*, we compared it against baseline methods over 100 communication rounds across all the datasets. The results showed that *FedFT* requires up to a 6% increase in resources compared to *FedSim* and *FedAvg*. However, this overhead is less than 5% when compared to *FedProx*. In our setup with a 1.7 GHz Quad-Core CPU, a 6% increase amounted to an additional 0.03 seconds of computation time. This increase is relatively insubstantial when weighed against the benefits that *FedFT* offers. Therefore, the slight increase in computation can be neglected when weighed against the enhanced communication efficiency it provides, saving network resources and overall efficiency.

5.5.6 Analysing the Effect of Non-IID on *FedFT*

FL environments inherently support and often require the handling of non-IID data due to their distributed nature. This experiment is focused on evaluating how *FedFT* performs under different levels of non-IID data. Figure 5.11 illustrates the outcomes obtained from the three FEMNIST variants, representing varying levels of non-IID, when applied to the *FedAvg* baseline. In the figure, two types of lines are used to represent the results. The solid lines show how *FedAvg*, combined with *FedFT*, performs. In contrast, the dashed lines show the performance of the standard *FedAvg* method.

The presented plots depict the average results obtained from 35 independent runs with random seeds conducted over 500 communication rounds. Our observations indicate that, in the experiment, *FedFT* consistently maintains comparable performance across

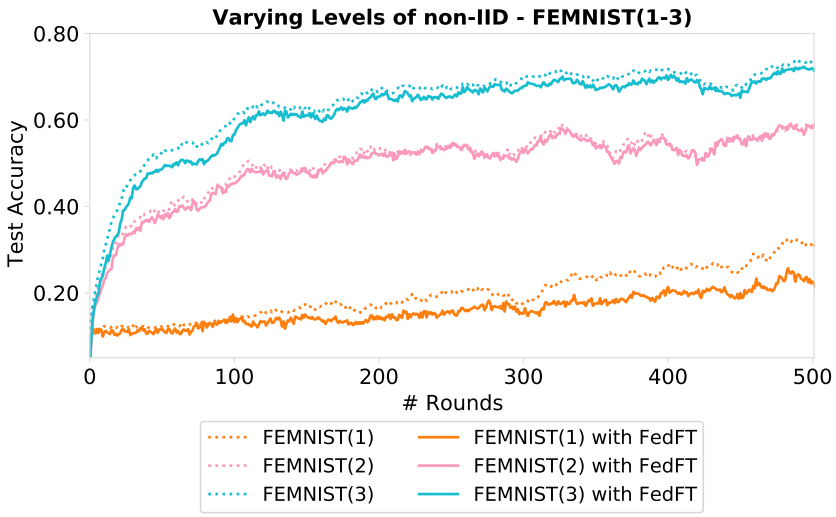

Figure 5.11: Varying levels of non-IID with three versions of the FEMNIST dataset

all levels of non-IID. Additionally, we note that any initial decrease in performance seen in FEMINST(2) and FEMNIST(3) gradually recovers in the later rounds. Additionally, it is essential to highlight that the overall performance of *FedAvg* in the FEMINST(1) dataset is comparatively weaker, requiring more communication rounds for convergence compared to the baseline *FedAvg*.

However, we observe that *FedFT* can catch up and follow a similar convergence trend in this extreme non-IID scenario. *FedFT* still shows a consistent trend, even in these varied non-IID conditions. This detailed investigation and generalisability studies confidently suggest that *FedFT* is appropriately suited for non-IID settings in Federated Learning.

5.5.7 Impact of *FedFT* Pruning

The ability to compress model parameters using pruning or quantisation (such as with JPEG images and video streaming) is a crucial aspect of communication in the frequency space. We examined the extent to which pruning can compress while preserving performance. Figure 5.12 presents test accuracy with increasing values of the pruning rate α for each dataset. As expected, accuracy suffers with higher values of α. This poor performance is mostly evident for pruning with $\alpha > 20\%$. Note that the model's inability to overcome the negative impact of high pruning on its performance prior to convergence results in a sub-optimal test accuracy post-convergence. However, it is encouraging to observe that at lower levels of pruning, comparable performance to no-pruning is achieved.

This suggests that there is a sweet-spot where pruning can achieve comparable or in some cases better accuracy than no-pruning. For instance, test accuracy with $\alpha = 10\%, 10\%$ and 14% is comparable to $\alpha = 0\%$ with MNIST, FEMNIST and Fed-MEx datasets respectively.

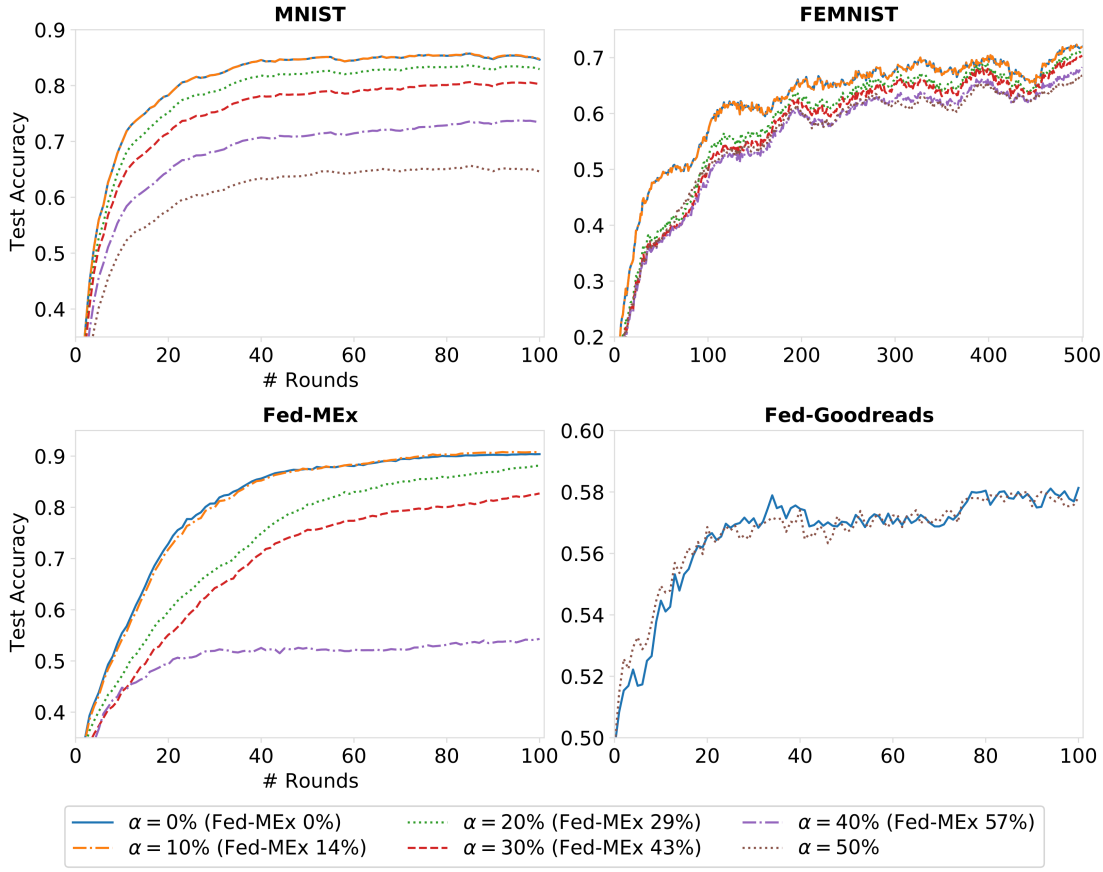

Figure 5.12: *FedFT* with pruning

The most favourable outcomes with pruning are observed in Fed-Goodreads, where $\alpha = 50\%$ yields performance comparable to that of no-pruning across communication rounds. This finding suggests that the magnitudes of high-frequency coefficients (i.e., those preserved without pruning) unintentionally carried noisy information, which initially hindered the federated aggregation.

5.5.8 Communication Efficiency with *FedFT*

To study the communication efficiency, we plot upstream communication costs in Figure 5.13. Here, a single trend line of a plot shows the test accuracy values measured

at a particular communication round (coloured lines). The x-axis is the accumulated upstream communication cost per client in MB measured on different α indicated by the markers.

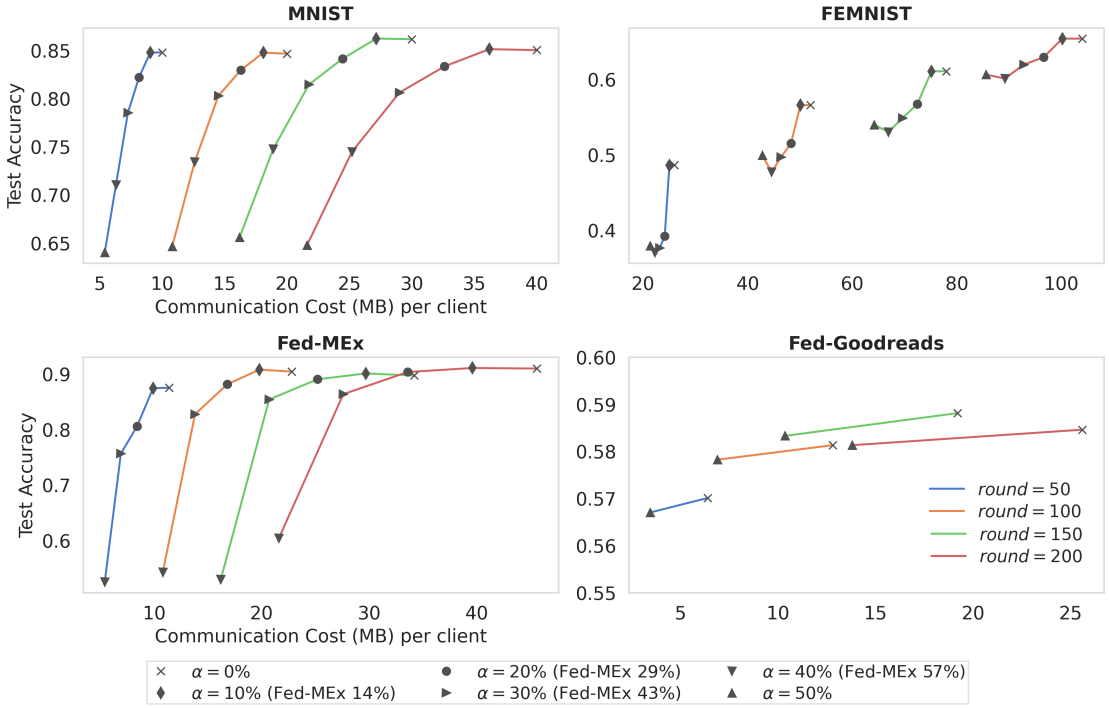

Figure 5.13: Optimising the upstream communication cost with *FedFT*

Firstly, Figure 5.13 confirms the general finding in Figure 5.12 that accuracy with pruning in the range, $0 < \alpha < 20\%$, is comparable to no pruning ($\alpha = 0$). Secondly, we can observe how *FedFT* pruning can optimise communication cost when given thresholds for test accuracy and communication rounds. The values shown in Figure 5.13 are outlined in Table 5.2 at the 200th round (i.e. red color line). The **bolded** figures highlight the optimal balance between accuracy and communication efficiency.

Finally, we address the issue where pruning at early stages of model training can lead to sub-optimal test accuracy. To mitigate the risk of losing information about clients at the early stages of training, we propose applying pruning after some communication rounds, preferably post-convergence. Post-convergence pruning can enhance communication efficiency by allowing the fine-tuning of a model after convergence. We choose these two datasets (MNIST, Fed-MEx) due to their apparent convergence, enabling us to establish the pruning threshold. When applying pruning, MNIST performances across all α

Table 5.2: Communication costs and accuracy for each pruning α Percentage at the 200th round

	MNIST		**FEMNIST**		**Fed-MEx**		**Fed-Goodreads**	
α	Cost (MB)	Acc.	Cost (MB)	Acc.	Cost (MB)	Acc.	Cost (MB)	Acc.
0%	40	85%	104	65%	45.6	90%	25.6	58%
10%	36.2	85%	**100.2**	**65%**				
14%					39.6	91%		
20%	**32.6**	**83%**	96.6	62%				
29%					**33.6**	**90%**		
30%	29	80%	92.8	61%				
40%	25.2	74%	89.2	60%				
43%					27.6	86%		
50%	21.6	64%	85.6	60%			**13.8**	**58%**
57%					21.6	60%		

values are comparable to no pruning ($\alpha = 0\%$). Fed-MEx also maintains comparable performances up to $\alpha = 43\%$. We attribute these improved pruning performances to the reduced magnitudes of weight adjustments made by client models after the global model converges.

5.5.9 Impact of *FedFT* pruning on *FedSim*

Building upon the analysis in Figure 5.12, we further explore the balance between pruning and performance retention, specifically within the *FedSim* (Introduced in Chapter 4) aggregation methodology. Figure 5.14 presents test accuracy with increasing values of the pruning rate α for each dataset with *FedFT* applied on *FedSim*. We note that at a pruning rate of $\alpha = 10\%$ (Fed-MEx: $\alpha = 14\%$), *FedFT* achieves comparable performance, enhancing communication efficiency.

As expected, the accuracy declines with higher values of α. However, it is significant to observe that, despite this reduction in accuracy, the core performance benefits of the *FedSim* method remain largely intact. This resilience highlights the robustness of the *FedFT* pruning approach, particularly in synergy with *FedSim* advanced aggregation strategy. We specifically chose to test *FedFT* with the *FedSim* method to explore its adaptability and performance in personalised/clustered FL algorithms. This approach is

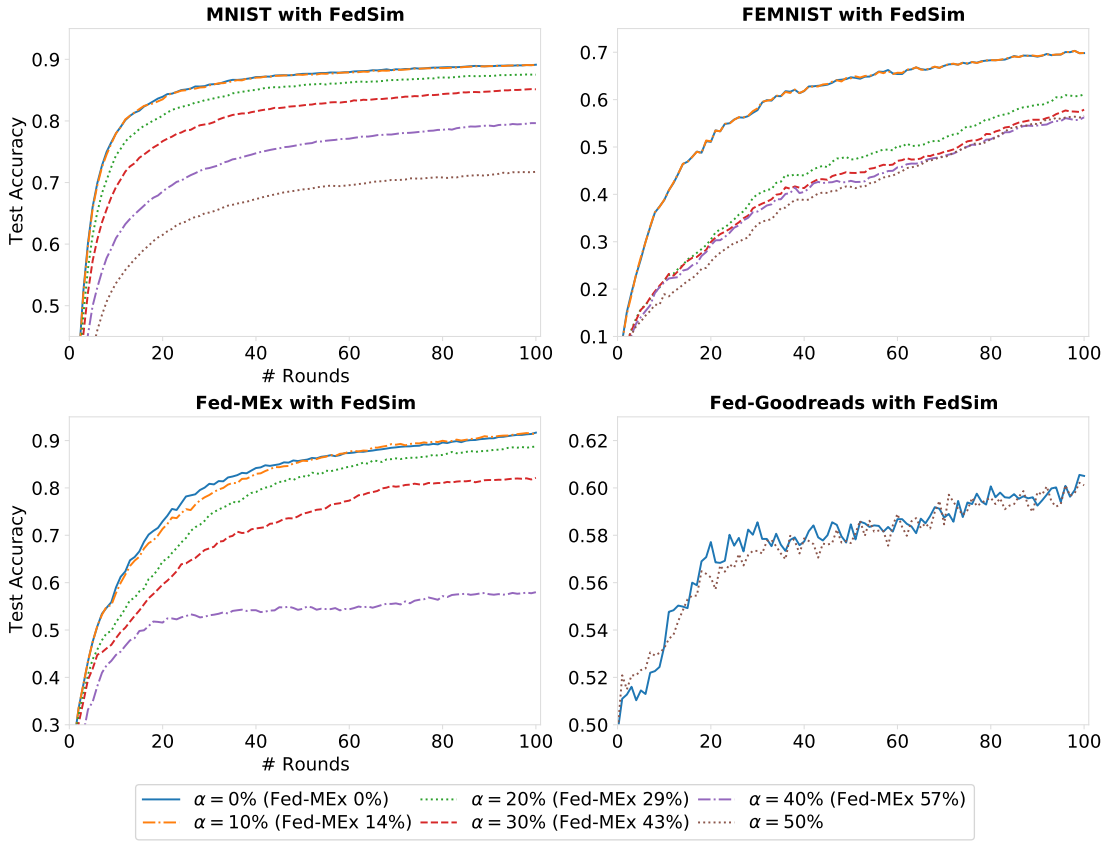

Figure 5.14: *FedFT* with pruning on *FedSim*

particularly relevant for real-world applications, where similarities among clients play a crucial role in enhancing the efficiency and effectiveness of the learning process.

5.5.10 Impact of *FedFT* pruning post-convergence

In FL environments, learning often occurs in incremental steps involving a substantial number of clients and rounds of communication. This process can continue to improve model performance even after initial convergence. We study post-convergence pruning in Figure 5.15, where we plot the results with a pruning threshold set at 50 communication rounds (represented by the blue vertical line) for MNIST and Fed-MEx. We chose these two datasets due to their apparent convergence, which enabled us to establish the pruning threshold.

When applying pruning, MNIST performances across all α values are comparable to no pruning ($\alpha = 0\%$). Fed-MEx also maintains comparable performances up to $\alpha =$

43%. We attribute these improved pruning performances to the reduced magnitudes of weight adjustments made by client models after the convergence of the global model. These findings suggest that post-convergence pruning can effectively maintain model performance while optimising communication efficiency in FL settings.

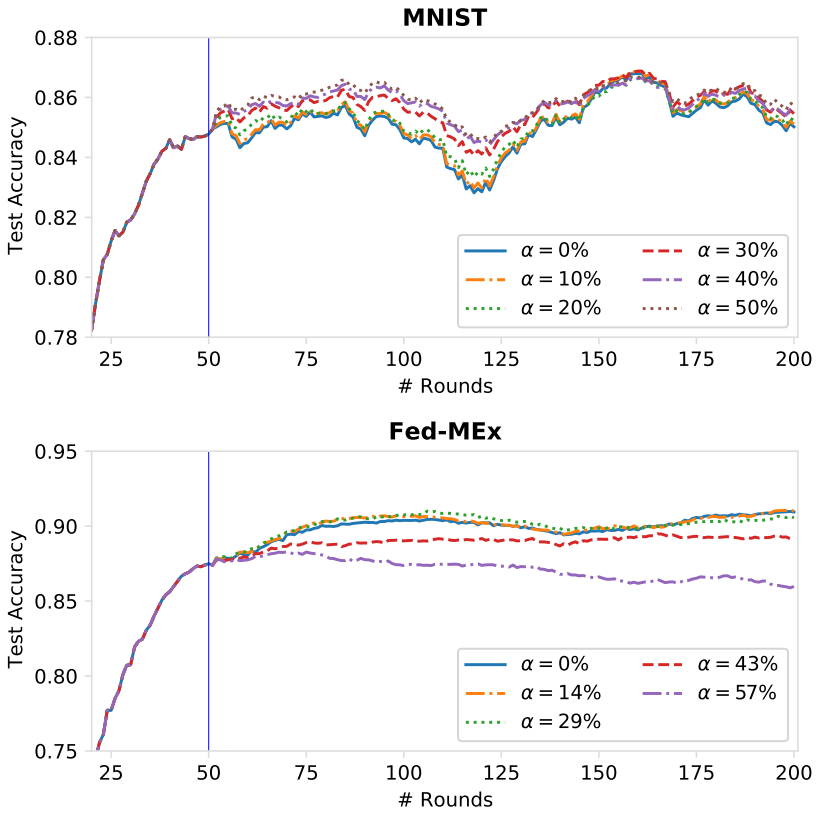

Figure 5.15: *FedFT* post-convergence pruning (round> 50)

5.6 Conclusion

FedFT introduced a novel FL methodology that communicates model parameters in the frequency space and performs federated aggregation in that same space. DCT-*IV* transformed and pruned model parameters of *FedFT* achieved reduced communication costs while maintaining model accuracy. Extensive experiments conducted on four FL datasets and employing three state-of-the-art FL methodologies demonstrate the generalisability of *FedFT* across diverse neural model architectures and FL methodologies. *FedFT* proves to be a generalisable solution, achieving communication savings of $5\% - 30\%$ while maintaining comparable accuracy. While *FedFT* demonstrated strong generalisability and

robustness across various scenarios, some limitations were identified. One notable limitation is determining the optimal pruning percentage to maximize communication efficiency without compromising information quality. Another limitation, similar to what was observed with *FedSim*, is the difficulty in handling extreme cases of IIDness and non-IIDness. Addressing these limitations opens several promising research directions. For instance, future work could explore dynamic adjustment methods that respond to varying degrees of statistical heterogeneity.

Chapter 6

Mitigating Gradient Inversion Attacks in Federated Learning with Frequency Transformation

"An ounce of **prevention** is worth a pound of cure."

Benjamin Franklin

This chapter explores and establishes a novel research direction that utilises the frequency space to defend against gradient inversion attacks in FL (Discussed in Section 2.8.1). By investigating and positioning the potential of utilising frequency space in this context, we aim to provide valuable insights and propose an effective strategy to counter the vulnerabilities posed by gradient inversion attacks within the FL setting. In this chapter, we explore the implications of gradient inversion attacks in FL and propose a novel defence mechanism, Pruned Frequency-based Gradient Defence (pFGD), to mitigate these risks. This chapter addresses the third research question (RQ3): *How do the communication efficiency strategies identified in RQ2 affect the security of FL in terms of data privacy?* To address RQ3, we focus on Objective 5: *Evaluate the integrated security benefits within the communication optimised algorithm.*

6.1 Use Case

Literature review in Chapter 2 discussed common defence strategies for gradient inversion attacks, including adding noise, gradient compression, training with large batch sizes and using complex models. We noted that methods like compression can be advantageous for the FL setting overall and for defending against such attacks. A key challenge in recent literature is the trade-off between model performance and communication efficiency, with security concerns further complicating this balance (Zhang et al., 2022b).

Recent work on gradient inversion attacks like DLG (Zhu et al., 2019) and iDLG (Zhao et al., 2020) has demonstrated the risk to privacy by exposing client private data (Discussed in Section 2.8.1). Both attacks attempt to reconstruct client data instances and labels using a gradient-matching objective. In a typical FL setting, clients share gradients with the server after a local training step. If an attacker obtains such gradients they can reconstruct training instances (there are assumptions on these methods as discussed in their methods). Gradient inversion attacks can be performed at any round in the FL process, even before model convergence.

Figure 6.1 presents the potential threat surfaces where gradient inversion attacks can occur. If an attacker in the network layer accesses the gradient information and obtains a view of client data, it undermines the entire purpose of FL. The previous chapter introduced *FedFT*, which uses the frequency space to improve communication efficiency in FL. In this chapter, we utilise the *FedFT* method to explore a practical defence against gradient inversion attacks in FL. Defending from such attacks ensure that the proposed methods in Chapter 4 and Chapter 5 are safe to use in practical applications.

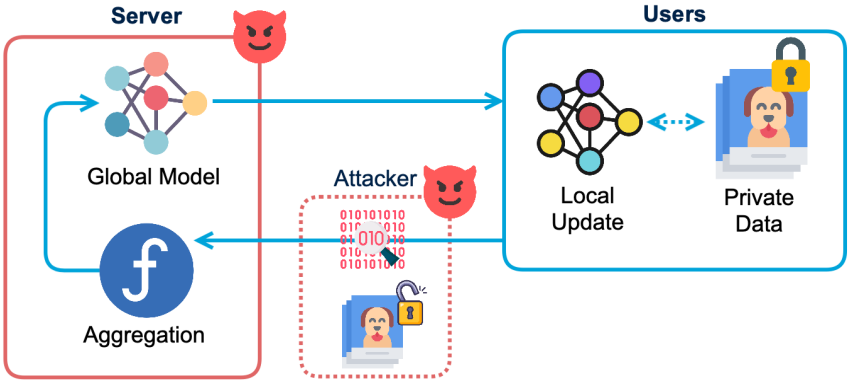

Figure 6.1: Potential risks of gradient inversion attacks in FL

6.1.1 Attack Methods

In this work, we use two iteration-based attack methods from the literature: DLG and iDLG. These attacks aim to reconstruct (steal) an FL client's local data instances using the communicated ΔW gradients. The attacker generates a pair of dummy data x' and dummy labels y', which are used to generate dummy gradients $\Delta W'$. By optimising the dummy gradients to closely match the client gradients, the dummy data becomes close to the actual data. Equation 6.1 demonstrates the objective of the selected gradient inversion attacks, where W is the shared global model, $F(.)$ is the shared optimisation function, and x'^*, y'^* are the optimised results (i.e., reconstructed data).

$$x'^*, y'^* = \arg\min_{x',y'} ||\Delta W' - \Delta W||^2 = \arg\min_{x',y'} ||\frac{\partial l(F(x', W), y')}{\partial W} - \Delta W||^2 \tag{6.1}$$

The key difference between DLG and its improved version iDLG, is how they extract the ground truth labels. The results presented by the iDLG authors suggest a 100% accuracy rate in generating the label from the gradients, unlike the DLG, which is around 79%-90% in the same experiments.

6.2 *p*FGD Methodology

We propose Pruned Frequency-based Gradient Defence (*p*FGD) which can act as a defence mechanism to such attacks while preserving model performance for FL setting. *p*FGD is a client-side frequency space based defence mechanism against DLG and iDLG. Once the local training is performed the updated gradients are transformed into the frequency space $\Delta\widehat{W}$ using transformation function $T(.)$. Then pruned by a pruning function $P(.)$ controlled by α percentage. The method assumes that the attacker possesses knowledge of the transformation function and can invert it using $\hat{T}(.)$. Client communication of pruned frequency gradients prevents gradient inversion through noise and parameter reduction. The *p*FGD transmission from the client mitigates risks from curious servers and network eavesdroppers. Figure 6.2 visually illustrates the workflow taking place on the client side, providing a clear representation of the various steps involved in *p*FGD.

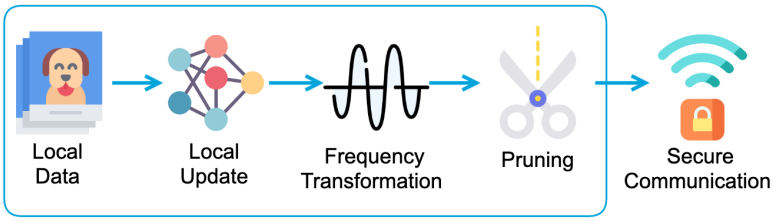

Figure 6.2: Client-side workflow in *p*FGD

6.2.1 Frequency Space Transformation

Based on *FedFT* and results from Section 5.5.2, we have selected DCT-IV as the transformation function, denoted as $T(.)$. DCT-IV has been found to balance preserving model performance and enhancing communication efficiency through pruning in the frequency space. After the gradients transform to the frequency space the resulting coefficients are structured to preserve the necessary information for model aggregation. Using the frequency space enables efficient pruning, identifying and discarding coefficients with lower magnitudes without significantly compromising model performance.

6.2.2 Parameter Pruning

Incorporating noisy gradients can be beneficial for defending against gradient inversion attacks such as DLG. However, determining an appropriate threshold for pruning gradients is a significant challenge. The objective is to strike a balance where the pruned gradients introduce sufficient noise to thwart such attacks while maintaining comparable performance. In the pruning function, $P(.)$, we adopted a straightforward approach within our proposed method. We set the coefficients with the most minor frequency (corresponding to small magnitudes) obtained from the DCT transformation to zero. By zeroing out these coefficients, we effectively prune the model to reduce its size while aiming to retain the essential information contained in the remaining coefficients.

6.2.3 Improving Resilience in FL

This work aims to introduce a method that strengthens the resilience of FL approaches against gradient inversion attacks. These attacks could compromise the fundamental benefits of FL, which is the preservation of client privacy. By incorporating the proposed method, *p*FGD, resilience can be achieved by utilising a generalisable technique such as the frequency domain (the frequency space) and pruning. The *p*FGD method addresses the vulnerability to gradient inversion attacks by leveraging the inherent properties of the

frequency space and pruning. Overall, this work aims to establish a resilient FL methodology that effectively combats gradient inversion attacks, thus enabling the continued protection of client privacy, a core principle of FL.

Figure 6.3 illustrates the adaptation of pFGD to existing FL methodologies. This adaptation introduces Steps 5 and 6, specifically designed to enhance resilience against gradient inversion attacks in the FL setting. Step 8 is used to inverse the frequency space model to raw space before model aggregation.

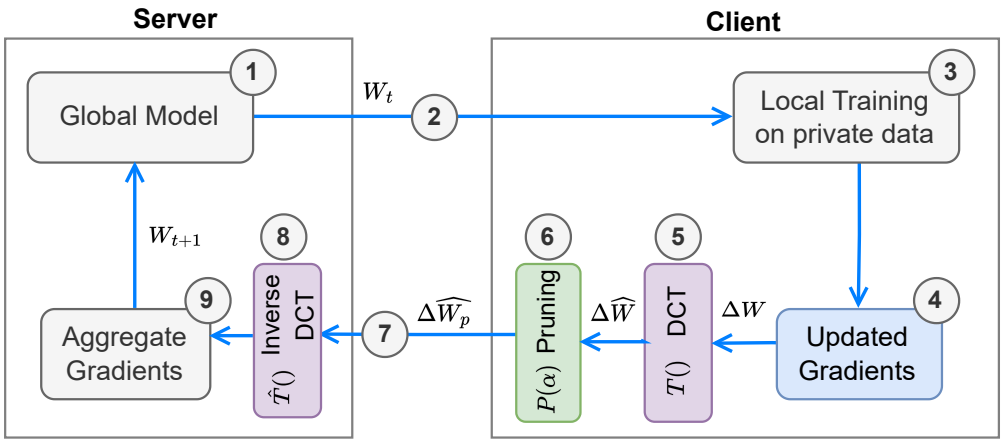

Figure 6.3: Adapting pFGD to existing FL methodologies

In Figure 6.3, Steps 2 and 7 represent the communication between the client and server, highlighting the potential vulnerability where an attacker can intercept and compromise the system's privacy.

6.2.4 *p*FGD Algorithm

Based on the considerations in previous sections, Algorithm 7 outlines the workflow required to implement pFGD. Note that the algorithm references the attacker method which assumes the attacker possesses knowledge of inverting the DCT through the inverse transformation function $\hat{T}(.)$.

6.3 Experiment Setup

To evaluate the pFGD, first, we study the impact of privacy on communicating client gradients in the frequency space. We explore to what extent parameter pruning in the frequency space can defend gradient inversion attacks. To evaluate the impact on privacy

Algorithm 7 Pruned Frequency-based Gradient Defence

Require: W: global model, α: Pruning Rate
Require: $T(.)$ DCT Function, $P(.)$ Pruning Function
1: $\Delta W \leftarrow$ update W using SGD on local data
2: **procedure** pFGD (ΔW, α)
3: $\Delta\widehat{W} = T(\Delta W) \leftarrow$ DCT transformation
4: $\Delta\widehat{W}_p = P(\Delta\widehat{W}, \alpha) \leftarrow$ Transformed Space Pruning
5: **return** $\Delta\widehat{W}_p$
6: **end procedure**
7: **procedure** ATTACKER($\Delta\widehat{W}_p$)
8: $\Delta W \leftarrow \hat{T}(\Delta\widehat{W}_p) \leftarrow$ Inverse DCT Transformation
9: DLG(ΔW) or iDLG(ΔW) $\leftarrow$ Perform Attack Scenario
10: **end Procedure**

by communicating model parameters in the frequency space we use two attack methods and one image dataset. DLG and iDLG are selected to study the performance of pFGD. The two methods are compared with and without the DCT transformation during the communication phase.

Unlike the previous experiments with *FedSim* and *FedFT* evaluating pFGD requires a specialised experimental setup and a specific use case. The experiment setup is publicly accessible on GitHub[1] for reproducibility. The experiment setup includes:

Dataset We select the MNIST (LeCun et al., 1998) dataset, a 10-class handwritten digit recognition image dataset. A single image's dimensions are 28x28 and have one channel. MNIST is commonly used in FL and security benchmarks as it provides a realistic setting. MNIST's single-channel images aid performance assessment due to sensitivity to variations. Selecting MNIST for comparison with prior works enhances understanding of the approach against gradient inversion attacks.

Configuration We adopt the experimental settings from Zhao et al. (2020); Zhu et al. (2019) to ensure consistency and comparability. We utilise LBFGS (Liu and Nocedal, 1989) with a learning rate of 1, batch size of 1 and 100 attack iterations for the attack scenarios. To mitigate the influence of random bias, we conduct 1000 runs of the experiments on LeNet models randomly initialised (i.e. 1000 randomly initiated models on a unique data instance). Experiments will terminate at the 100th iteration or if the loss is below 0.000001.

[1] https://github.com/chamathpali/pFGD

Pruning As highlighted in Section 6.2, pruning plays a significant role in introducing noise to the gradients, thereby diminishing the effectiveness of the attacks. In our experiments, we ensure consistency by using a fixed pruning rate of $\alpha = 1\%$, resulting in the pruning of 133 parameters. Additionally, we performed secondary experiments with a 0.1% pruning rate (11 parameters pruned) to ensure fair comparison and assess pruning's impact on pFGD's defence against gradient inversion attacks.

Evaluation Metrics We log the MSE of the reconstructed instance and the original image at each iteration. These MSE values are used to analyse and evaluate the behaviour of the proposed method. By counting the number of successful bypasses at each threshold, we gain insights into the effectiveness of the different variants in defending against the respective attacks. Considering the minimum MSE value from each experiment ensures that we capture the reconstruction's performance under various conditions and iterations.

6.3.1 Comparative Study

We explore multiple variants of the selected baselines to understand the impact of the pFGD technique. We are explicitly considering the following four variants for the DLG and iDLG attack methods:

1. Vanilla (original method without modifications)
2. Vanilla with pruning (pruning applied to the vanilla method)
3. DCT (applying only DCT transformation)
4. DCT with pruning (pruning applied to the DCT transformed gradients)

By examining these different variants, we can assess the effectiveness and comparative performance of pFGD in various configurations and scenarios.

6.4 Results and Discussion

We first study the reconstructed images to gain insight and visually understand the process of reconstruction attacks. Figure 6.4 visually presents the reconstructed images at different MSE threshold points, allowing for an assessment of their readability. By observing these visual representations, we can assess the success of the reconstructions

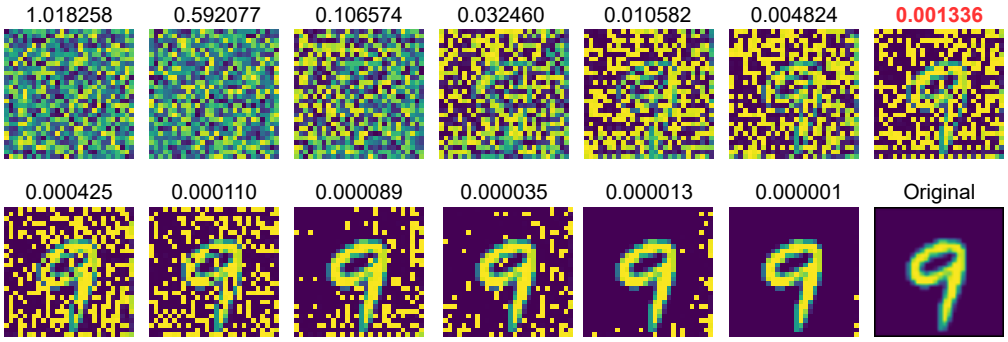

Figure 6.4: Reconstructions of digit 9 are displayed at various MSE points, indicated above each image, ranging from higher to lower values. The final image presents the original digit 9 for comparison.

and identify any potential leakage of private information. At MSE= 0.001 (highlighted with red text in Figure 6.4), the digit 9 becomes noticeable upon closer examination.

Next, we study the impact of gradient inversion attacks on the four variants described in Section 6.3.1. The results are presented in Figure 6.5, which illustrates the number of experiments that could surpass different MSE thresholds.

Figure 6.5 presents bar plots with eight colours, representing DLG and iDLG experiments in two groups. Plots with the squared pattern represent the pruned variants, while those with diagonal patterns represent the DCT variants. The graph legend's notation '_P' represents the pruned variants. Specifically the blue and orange bars represent DCT with pruning for DLG and iDLG experiments respectively.

We observe that when MSE= 1, only 24 and 12 experiments surpass the threshold for DLG and iDLG, respectively, when applying DCT with pruning. Additionally, we found that no reconstructions of DCT with pruning were found when the MSE was less than 0.9. In contrast, reconstructions were identified even when the MSE reached a low value of 0.005 for pruning on the vanilla methods.

When pruning on vanilla gradients without DCT, there is still a high risk of leaking privacy-sensitive information. In our experiments, we could visually identify these reconstructions as the original images. For the MNIST dataset, having a reconstructed image with an MSE value of approximately 0.001 is sufficient for accurate digit identification. This cutoff point may differ from dataset to dataset for individuals with different eyesight. However, our key observation is that even with pruning at $\alpha = 1\%$, reconstructions are still possible on vanilla gradients.

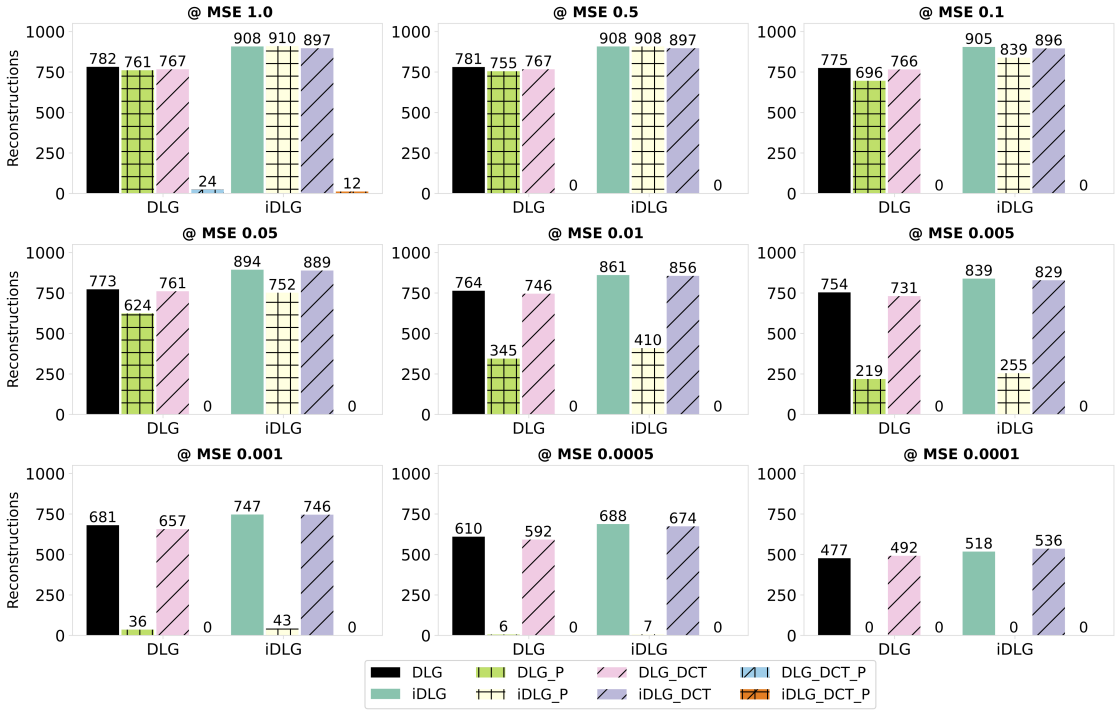

Figure 6.5: Number of reconstructions at different MSE thresholds on MNIST dataset with $\alpha = 1\%$ with 4 variants on DLG and iDLG

Similarly, we performed experiments with a pruning rate of $\alpha = 0.1\%$, and the corresponding results are depicted in Figure 6.6. In this particular set of experiments, we observed a notable increase in the number of reconstructions in the case of vanilla with pruning when the MSE reached 0.001. Specifically, we observed 393 reconstructions with DLG and 454 reconstructions with iDLG.

Notably, maintaining a low pruning percentage in the vanilla variants improves the readability of the digits. Even at MSE = 0.0001, 99 reconstructions with DLG and 109 reconstructions with iDLG using the vanilla pruned method. In contrast, no reconstructions are observed for DCT with pruning when MSE is less than 0.02. These findings demonstrate the resilience of the proposed pFGD against gradient inversion attacks.

The results obtained in our study provide compelling evidence that combining DCT with pruning techniques significantly enhances defence against gradient inversion attacks. Throughout the 1000 experiment runs, we did not observe any reconstructions when applying DCT with pruning (pFGD) with an MSE below 0.9. These reconstructions lacked readability, rendering them essentially non-existent. In contrast, our research

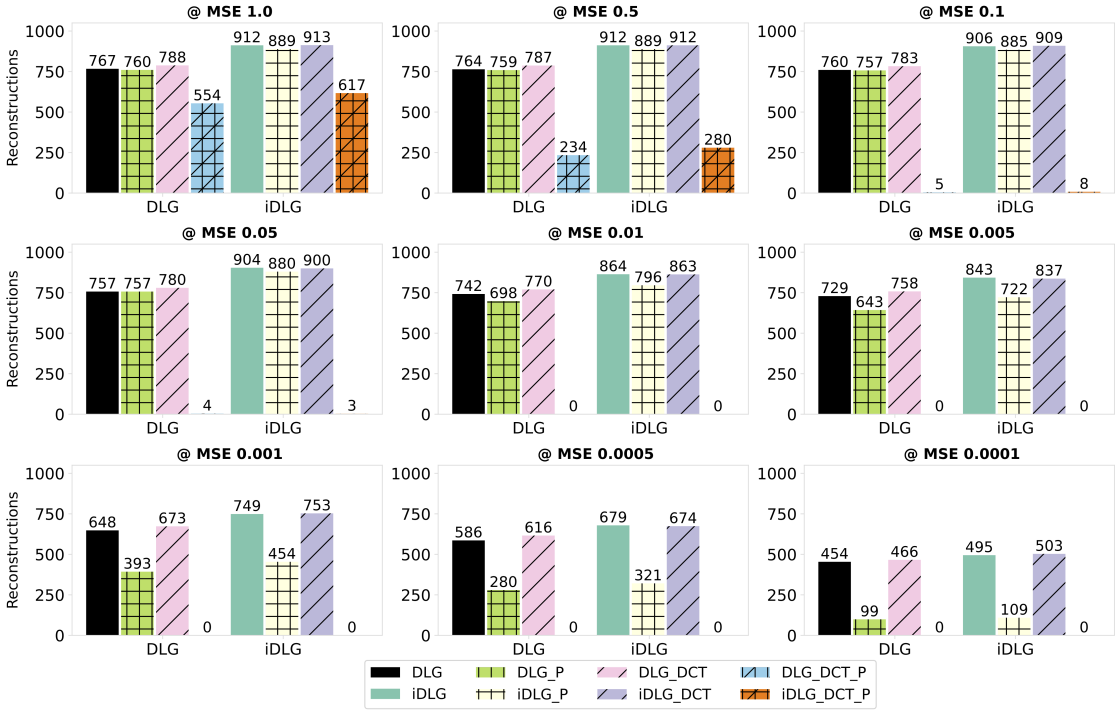

Figure 6.6: Number of reconstructions at different MSE thresholds on MNIST dataset with $\alpha = 0.1\%$ with 4 variants on DLG and iDLG

findings reveal that applying pruning alone to the vanilla gradients, in the absence of employing the DCT, still poses a considerable risk of privacy breaches. For the MNIST dataset, our findings indicate that achieving a reconstructed image with an MSE of approximately 0.001 is sufficient for accurate digit identification. Around the MSE value of 0.005, we noticed a significant indication of a digit with potential lines emerging in the reconstructions. However, it is essential to note that this threshold may vary across datasets and individual visual strengths. Our experiments visually demonstrated the identification of reconstructed images as the original ones in such cases. Together, these results highlight the resilience and efficacy of the proposed pFGD in countering gradient inversion attacks. However, to further strengthen the evaluation, exploring other frequency transformation approaches and testing against different attack methods is important. This could help ensure that pFGD remains robust across various scenarios and adversarial strategies.

6.5 Conclusion

In this chapter, we introduced pFGD, a defence mechanism designed to mitigate gradient inversion attacks in FL. By applying *FedFT* and incorporating pruning before communication, pFGD effectively enhances the resilience of FL models against such attacks. We conducted a comparative study involving two attack methods and four variants for each method on the MNIST dataset. Our experimental results prove that utilising pFGD offers superior protection against gradient inversion attacks compared to pruning with raw gradients alone. Additionally, we observed that implementing pFGD using the frequency space does not lead to significant performance degradation (based on findings from Section 5.5.7). One of the notable advantages of pFGD is its practicality, as it can be easily applied to different FL methodologies with minimal modifications. Our findings highlight the effectiveness and potential of pFGD as a defence mechanism against gradient inversion attacks in FL.

A limitation of this study is the scope of the evaluation on gradient inversion attacks with DLG and iDLG. pFGD 's effectiveness against other privacy attacks, such as membership inference and GAN reconstruction attacks, remains unexplored. Additionally, a comparative analysis with established privacy-preserving methods like Differential Privacy and Homomorphic Encryption would provide deeper insights into pFGD 's effectiveness. Future work could expand the evaluation of pFGD across a broader range of privacy attacks and benchmark it against other approaches.

Chapter 7

Application to Real-world Data: The eICU Database Case Study

> "The whole is more than the sum of its parts."
>
> *Aristotle*

In this chapter, we apply the methodologies introduced earlier, *FedSim* and *FedFT*, to a practical problem encountered in the real world. This chapter focuses on Objective 6: *Conduct a case study to validate the proposed methodologies in a real-world context.* We selected a healthcare case study to validate the efficacy of our proposed methods and demonstrate their applicability in addressing complex challenges within the healthcare domain. This choice is motivated by the unique issues associated with real-world healthcare data, such as class imbalance and statistical heterogeneity. We have chosen the eICU Collaborative Research Database (Pollard et al., 2018) as our focal dataset for this case study.

7.1 Background on the eICU Dataset

The eICU Collaborative Research Database [1] (eICU) is a ground-breaking initiative that brings together comprehensive clinical data from a large number of intensive care units (ICUs) throughout the continental United States. The creation of this dataset was a joint

[1] `https://eicu-crd.mit.edu/`

effort between Philips Healthcare and the MIT Laboratory for Computational Physiology. The eICU database is an extensive and comprehensive collection of data comprising information from more than 200 hospitals. The database consists of patients who were admitted to critical care units during the years 2014 and 2015. With detailed information from over 200,000 ICU admissions, researchers can analyse various aspects of ICU care, including basic demographic details, intricate physiological metrics, treatment interventions and clinical outcomes. These aspects include patient mortality, disease progression, treatment efficacy and healthcare resource utilisation. Due to its distribution, the eICU database was selected as the preferred option over the Medical Information Mart for Intensive Care (MIMIC) (Johnson et al., 2020). Unlike the MIMIC database, which gathers data solely from one medical centre, the eICU database comprises data from more than 200 hospitals. This extensive range of data offers a more comprehensive perspective, which is particularly advantageous for the FL setting.

7.1.1 Relevance to Healthcare Research

The eICU database has proven to be an indispensable resource for healthcare researchers. Previous studies have utilised this dataset to develop predictive models for patient outcomes (Patel et al., 2021), characterise patient behaviour (O'Halloran et al., 2020; Sheikhalishahi et al., 2020), and identify critical illnesses (Beyer et al., 2021). The extensive collection of patient data in the dataset enabled researchers to gain a deeper understanding of critical care dynamics and patient recovery patterns. One critical field of study that leverages the eICU database concerns mortality rates (Safaei et al., 2022; Xu et al., 2022). Examining these rates has yielded valuable knowledge regarding the factors that influence patient outcomes in ICUs. This area of investigation is essential in identifying risk factors associated with higher mortality rates. As a result, healthcare practitioners can develop and apply specific interventions to improve patient care.

7.1.2 Problem Statement and Objective

Unlike a patient admitted to a regular hospital ward, when admitted to the ICU, more complex tests, interventions and scores are calculated (e.g., advanced monitoring, mechanical ventilation). These tests and scores are to support the healthcare workers and act rapidly in a critical time for a patient between life and death. Scoring methods are split into two types (Vincent and Moreno, 2010):

1. For an organ or disease: Such as Glasgow Coma Scale (GCS) and Sequential Organ Failure Assessment (SOFA).

2. Generic for all patients: Such as Acute Physiology, Age, and Chronic Health Evaluation (APACHE), Simplified Acute Physiology Score (SAPS) and Mortality Probability Model (MPM).

The eICU database utilises the APACHE, which measures the severity of ICU patients. This includes predictions for the probability that a patient will die (mortality) and performance benchmarking for an ICU unit. Calculate the probability of mortality using data collected in the first 24 hours, including physiologic measurements, comorbid burden, treatments given and admission diagnosis. Then, these parameters are used in a logistic regression model to predict the mortality. The eICU database contains explicitly all the parameters required for the APACHE IV model (Zimmerman et al., 2006). The *apacheapsvar* table contains physiologic parameters and other parameters in the *apachePredVar* table. *apachePatientResult* table stores the results from APACHE IV and APACHE IVa.

Out of the different types of research possible with the eICU dataset, we select forecasting the mortality of an ICU patient. This selection is due to its impact on the actions taken by healthcare professionals and the APACHE scoring parameters already computed into two tables. Further details of the feature selection and data preprocessing are detailed in Section 7.2 and 7.3, respectively.

7.2 Feature Selection

The selection of parameters for the mortality prediction model can be organised into eight categories, each representing a diverse aspect of patient health and hospital care. These categories are designed to capture specific dimensions of patient data, ranging from physiological indicators to treatment interventions. The features are captured from the two tables *apacheapsvar* and *apachePredVar*, which computed the APACHE scores by capturing the features. For the case study, we selected 40 features from the two tables out of 74 features to predict a patient's mortality. The columns removed are either with all the data is the same (e.g. saps3yesterday, teachtype), minimal patients (e.g. aids, only in 189 patients) or system-related features (e.g. managementsystem, sicuday, region). Table 7.1 presents the selected variables categorised into eight key areas and their descriptions.

Table 7.1: Selected variables from the eICU Database

Category	**Name**	**Description**
Demographic	age	Age of the patient in years (full)
	gender	Female =1, Male = 0, Not available =-1
Lab	wbc	White blood count
	temperature	Celsius temperature value
	bun	Blood urea nitrogen
	creatinine	Level of creatinine in the blood
	glucose	Level of glucose in the blood
	hematocrit	Proportion of red blood cells in the blood
	bilirubin	Bilirubin level
Treatment	dialysis	Patient is on dialysis = 1, Not on dialysis/not available = 0
History	metastaticcancer	Patient has metastatic cancer = 1, Patient doesn't have metastatic cancer = 0
	immunosuppression	Patient has immunosuppression = 1, Patient doesn't have immunosuppression = 0
Respiratory	vent	Patient is on mechanical ventilation = 1, Not on ventilation/not available = 0
	pao2	Partial pressure of oxygen in arterial blood
	fio2	Fraction of inspired oxygen
	respiratoryrate	Respiratory rate measured in breaths per minute
	pco2	Partial pressure of carbon dioxide in arterial blood
	urine	Urine output value when present (24 hours)
	sodium	Level of sodium in the blood
	heartrate	Heart rate measured in beats per minute
	meanbp	Mean arterial blood pressure
	ph	Arterial blood pH value
	albumin	Level of albumin in the blood
Clinical Assessments and Interventions	motor	Motor response component of the GCS (Glasgow Coma Scale)
	eyes	Eye opening response component of the GCS
	verbal	Verbal response component of the GCS

	activetx	Active treatment being administered
	ima	Internal Mammary Artery Graft
	ventday1	Patient was on mechanical ventilation on day 1 of ICU stay
	oobventday1	Patient was ventilated at anytime for the APACHE day
	oobintubday1	Patient was intubated at anytime for the APACHE day
	intubated	Patient is intubated = 1, Not intubated/not available = 0
Other Conditions and History	diabetes	Patient has diabetes = 1, Patient doesn't have diabetes = 0
	readmit	Patient readmitted = 1, not readmitted = 0
	admitdiagnosis	Primary diagnosis at admission (APACHE admission diagnosis code)
	day1verbal	Verbal response component of the GCS on day 1
	day1motor	Motor response component of the GCS on day 1
	day1eyes	Eye opening response component of the GCS on day 1
	day1pao2	Partial pressure of oxygen in arterial blood on day 1
	day1fio2	Fraction of inspired oxygen on day 1

The eICU case study aims to enhance the predictive accuracy for ICU patient mortality in an FL setting, identified through the *diedinhospital* variable. This goal highlights our commitment to applying and validating FL approaches in critical healthcare scenarios.

7.3 Data Preprocessing

In this section, we discuss the data pre-processing of the eICU database and analysis of the selected data partition. This analysis is essential to ensure that the data we consider is of the highest quality and suitable for our research.

7.3.1 Data Extraction

The access to the database was obtained using the PhysioNet data repository (Goldberger et al., 2000) and its platform. As the data was extensive and for efficiently retrieve and query the database, we used Google BigQuery[2], with the specific query shown in Listing 7.1.

[2] `https://cloud.google.com/bigquery/`

```
SELECT
  `physionet-data.eicu_crd.apachepredvar`.*,
  `physionet-data.eicu_crd.apacheapsvar`.*,
  `physionet-data.eicu_crd.patient`.hospitalid
FROM
  `physionet-data.eicu_crd.apachepredvar`
LEFT JOIN
  `physionet-data.eicu_crd.apacheapsvar`
ON
  `physionet-data.eicu_crd.apacheapsvar`.patientunitstayid =
      `physionet-data.eicu_crd.apachepredvar`.patientunitstayid
LEFT JOIN
  `physionet-data.eicu_crd.patient`
ON
  `physionet-data.eicu_crd.patient`.patientunitstayid
   = `physionet-data.eicu_crd.apachepredvar`.patientunitstayid
WHERE
  `physionet-data.eicu_crd.apachepredvar`.age > 0 AND
  `physionet-data.eicu_crd.apachepredvar`.admitdiagnosis != ""
```

Listing 7.1: SQL query to extract the raw data from eICU database

7.3.2 Data Analysis

After exporting the raw data from BigQuery, we analysed it statistically. Figure 7.1 presents the hospital mortality rate distribution, showing how they vary across different institutions. From the query in Listing 7.1, there are 207 unique hospitals selected. Out of the 207, there are 11 hospitals which have no mortality (i.e. 5.3%). We consider three thresholds:

- Low: Less than 5%
- Medium: More than 5% and less than 10%
- High: More than 10%

Most hospitals fell within the Medium threshold category, with 96 hospitals fitting this criterion. This suggests that a significant proportion of hospitals experience a moderate level of mortality rates, indicating a standard range of outcomes within the broader critical care landscape. In contrast, 67 hospitals were categorised within the High threshold, signalling a concerning level of mortality that prompts questions regarding care quality

or patient case severity. Meanwhile, the Low mortality category comprised 33 hospitals, reflecting a subset of institutions achieving notably favourable patient outcomes.

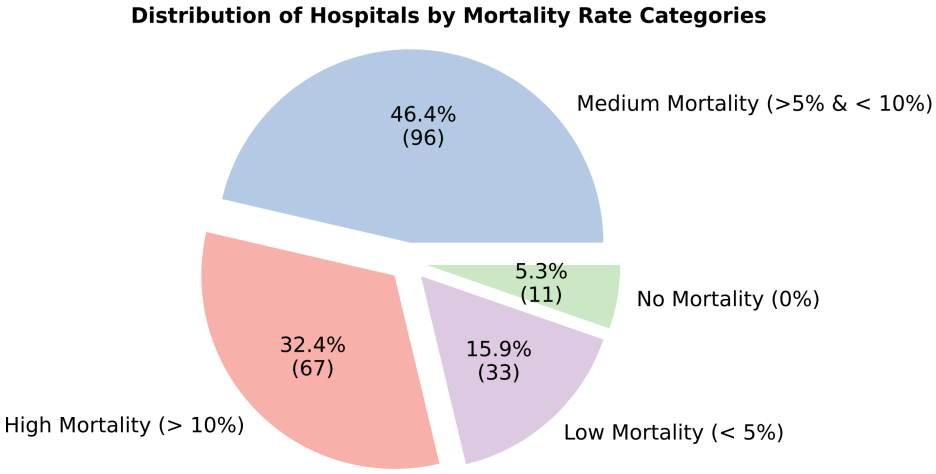

Figure 7.1: Distribution of mortality rate categories

To further analyse the distribution of mortality rates by hospitals, Figure 7.2 plots the hospitals with high mortality rates. The red line presents the threshold point (i.e. 10%), and the green dotted line shows an at 20%. We observe that there are only four hospitals with over 20% mortality rates.

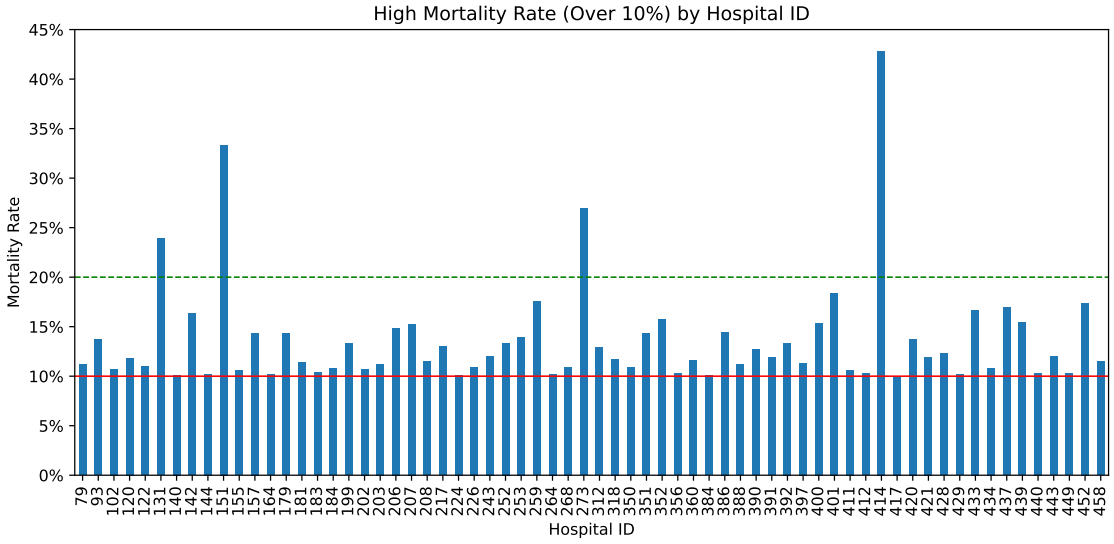

Figure 7.2: High mortality rates by hospital ID

This data distribution provides insights into the characteristics of realistic healthcare

databases. Furthermore, these databases often exhibit a majority class, leading to data imbalance. This imbalance implies that models may exhibit high accuracy due to the dominant majority class while failing to detect the minority class accurately, resulting in misleading and fluctuating performance metrics. Such limitations affect local ML algorithms and FL setting, as both struggle with imbalanced data. In a real-world FL scenario with this type of data imbalance, applying techniques like SMOTE (Synthetic Minority Over-sampling Technique) before federated aggregation could improve the model's ability to recognise minority classes. Using SMOTE within each client and applying FL methodologies can enhance the detection of rare events, such as unusually high or very low hospital mortality rates. This approach helps create a more balanced and effective healthcare model, promoting fairer and more robust healthcare insights across institutions. The following sections explore both techniques, showing their application and impact on model performance with imbalanced datasets.

7.3.3 Fundamental Data Handling

This section outlines the required steps to prepare the dataset for further analysis, beginning with feature transformation and partitioning the data into training and testing splits. These fundamental steps are essential and utilised in the procedures described in Section 7.3.4 and Section 7.3.5.

Data Transformation

In our study, as detailed in Table 7.1, most features collected from the eICU database are numeric, providing a straightforward path for statistical analysis and model input. Numeric data types facilitate a range of computational analyses without extensive preprocessing. However, a notable exception within our dataset is the *admitdiagnosis* feature, which is categorical and presented in a string format. When presented as strings, categorical data cannot be directly used in most mathematical models that require numerical input. We employed a label encoding technique to address this challenge and ensure that *admitdiagnosis* contributes meaningfully to our predictive models. This transformation assigns an integer to each possible category of the *admitdiagnosis* variable.

Data Split

Uniquely in an FL setting, this partitioning happens at the client level, which corresponds to individual hospitals in our study. We adhere to a conventional 70:30 ratio for this split,

allocating approximately 70% of each hospital's data for training and the remaining 30% for testing.

7.3.4 Preliminary Analysis with Raw Data

We begin by analysing the behaviour of the raw data obtained from the eICU database. Figure 7.3 presents the preliminary experiment results, with the parameter and model details described in Section 7.5.

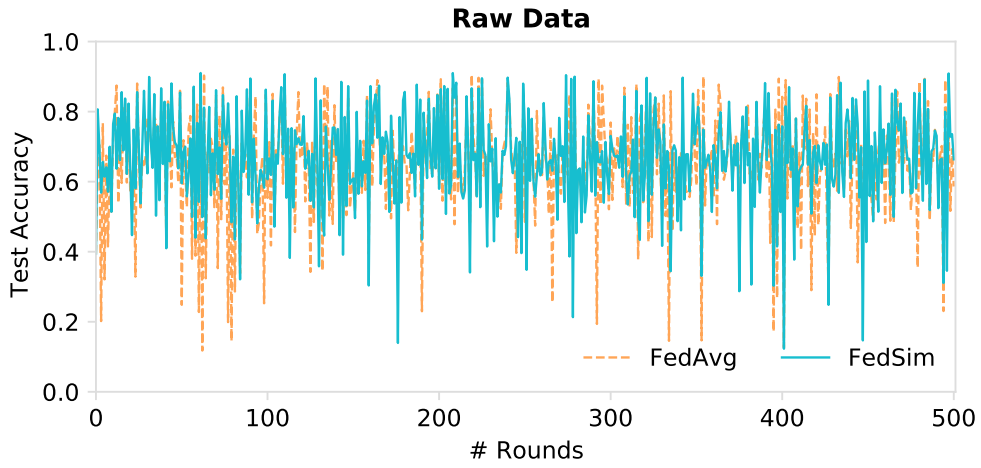

Figure 7.3: Preliminary analysis with raw data from the eICU database for *FedAvg* vs *FedSim*

The results from Figure 7.3 show significant fluctuations and low accuracies, indicating that the model struggles to generalise well with the raw data. This suggests that further preprocessing is necessary to enhance model performance and stability.

7.3.5 Data Pre-processing

The pre-processing of data is a critical step in effectively preparing datasets for complex analysis and training. It involves a series of carefully executed procedures, each aimed at refining the dataset and ensuring it is optimally helpful for further analysis. Below are the pre-processing steps we take:

- **Data Filtering:** This initial step involves excluding records that fail to meet established inclusion criteria.
- **Addressing Imbalanced Classes:** It is identified from Section 7.3.2 that the

eICU database exhibits a class imbalance. Steps are required to enhance the representativeness of minority classes (i.e., diedinhospital).

The sequence of steps taken in the data pre-processing is illustrated in Figure 7.4. The following sections will discuss each step in detail. The final phase in our data preprocess-

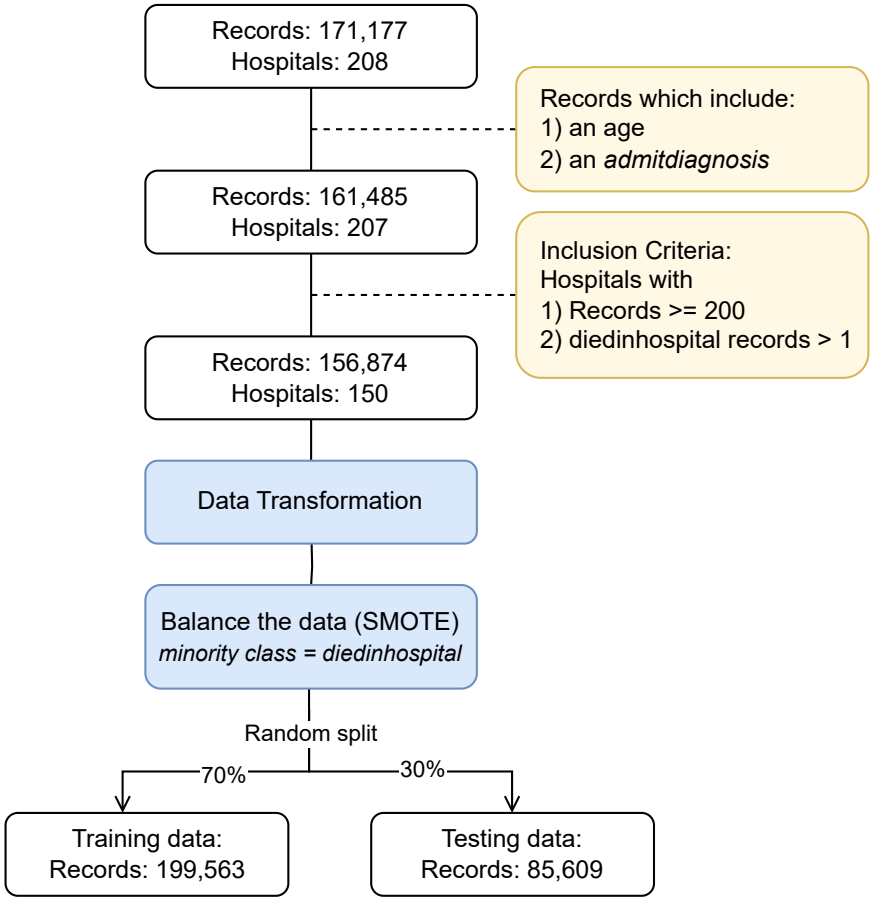

Figure 7.4: Dataset Compilation Process

ing pipeline is partitioning the dataset into distinct sets for training and testing purposes. Following 70:30 split ratio across all participating hospitals, we gather 199,563 records for training and 85,609 records for testing. This enables a structured and thorough analysis throughout the FL training rounds.

Data Filtering

Data filtering is a crucial initial step in refining datasets/databases to ensure that only the most relevant and accurate data points are included in further analyses. We carefully

carried out this process for the eICU database by setting specific filtering criteria. Records not meeting these criteria were excluded, ensuring the dataset's integrity and reliability. From the data extraction step described in Section 7.3.1, we retrieve all the records which have an *admitdiagnosis* and patients having an *age*, which is 161,485 records. Out of the filtered records, we have the following inclusion criteria for the hospitals: with 200 or more records available, more than one mortality record (i.e. $diedinhospital > 1$). The filtering step refined the database to 150 hospitals and 156,874. Figure 7.4 highlights the filtering criteria in yellow.

Addressing Imbalanced Classes

Based on the analysis of the eICU database in Section 7.3.2, the number of samples with $diedinhospital = true$ vary and imbalanced across hospitals, we faced a significant issue of data imbalance. In order to address this issue and to ensure that our analysis is statistically robust and meaningful, we employed a state-of-the-art technique called Synthetic Minority Oversampling Technique (SMOTE) (Chawla et al., 2002). This approach generates synthetic instances for the underrepresented class, thereby mitigating the data imbalance and ensuring the resulting dataset is balanced and representative. SMOTE is applied within each client (i.e., on a hospital-wise basis), preserving the data's decentralised nature. As a result of employing SMOTE, our refined dataset comprises a total of 285,172 samples, which is a substantial increase from the original dataset.

7.3.6 Impact of Data Pre-processing

To illustrate the necessity of both preprocessing steps described in Section 7.3.5, we present results from experiments evaluating different combinations of these steps. We conduct two experiments:

- Raw data with SMOTE
- Filtered data without SMOTE

The experimental setup, parameters, and model details are the same as described in Section 7.5. First, we apply the SMOTE approach to address the issue of data imbalance. Figure 7.5 presents the results for raw data with SMOTE applied. Here, we observe reduced accuracy drops and more stable performance, ranging around 45% to 65%. These results are much more stable than the raw data experiment in Figure 7.3. While SMOTE helps to address the data imbalance, it does not fully resolve the underlying issues affecting model performance. This suggests that the additional filtering step could further

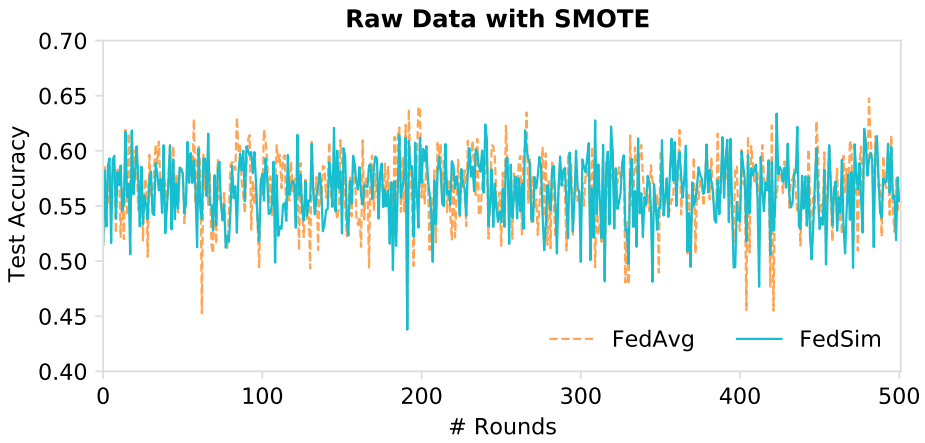

Figure 7.5: Analysis with raw data with SMOTE for *FedAvg* vs *FedSim*

optimise the training process and improve efficiency.

Next, we apply filtering to the raw data as described in Section 7.3.5. Figure 7.6 presents the results for filtered data without SMOTE. While the results show slight stabilisation compared to the raw data, there are still fluctuations, and the model does not exhibit the desired behaviour. This suggests that both filtering and applying SMOTE to address class imbalance are necessary to achieve better and more stable performance.

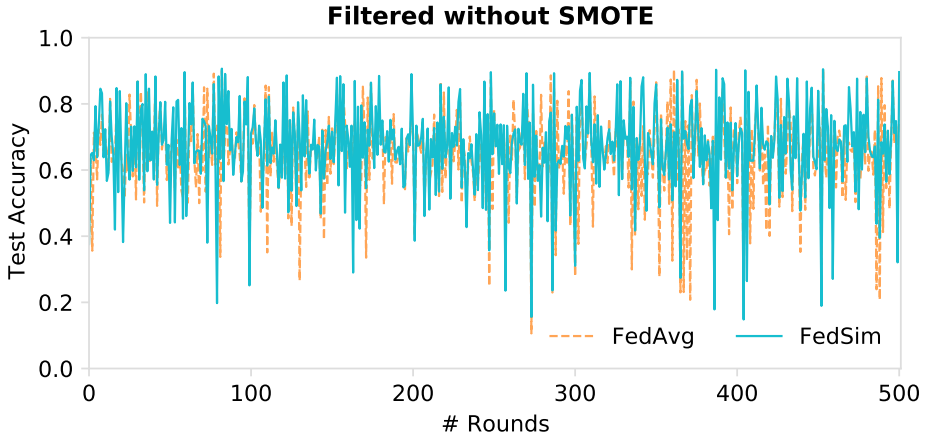

Figure 7.6: Analysis with filtered data without SMOTE for *FedAvg* vs *FedSim*

These preliminary results demonstrate the challenges associated with realistic healthcare data. Raw medical datasets often contain issues such as class imbalance, which can significantly affect model performance. In real-world applications, healthcare data is typically

collected from multiple sources with varying protocols, making it highly heterogeneous and complex to standardise. This further strengthens the rationale for using healthcare data as a case study to evaluate the proposed methodologies.

7.4 Applying Federated Learning to the eICU Database

As previously discussed, FL has many applications across different domains. However, special consideration needs to be given to the healthcare domain. One of the main factors to consider is when it consists of only a few clients in the hundreds scale (i.e. 150 hospitals in this case), unlike the typical FL setting (i.e. cross-device) where thousands of clients participate in an FL training. Scenarios with fewer clients but where each client has massive data are called cross-silo FL. There is a lack of research and case studies in the cross-silo setting in FL due to its closed nature of application (Kairouz et al., 2019). A cross-silo setting has many advantages compared to a cross-device setting, as there is more data security and communication layer control.

In order to apply the FL setting to the eICU database, we can consider each hospital as a client. Each hospital would have its isolated dataset and a local model, which trains and optimises with the FL rounds. Figure 7.7 presents a high-level system architecture of applying FL in a healthcare domain.

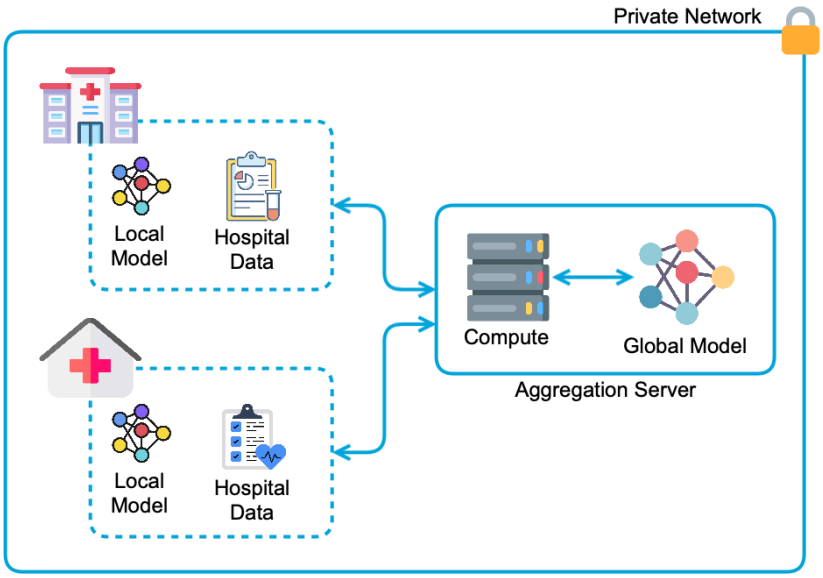

Figure 7.7: Using FL across multiple hospitals

Typically, a cross-silo FL setting will be in a private network with restricted access connecting with the hospital governing bodies (e.g. NHS England and NHS Scotland). Each hospital will have its own private data, which any participating body will not share. When the FL training starts, each hospital will receive the shared global model and train with its data. Once the training step is completed, they will share the updated local model with the aggregation server to aggregate and update the global model. The global mode will be trained with many federated rounds until the global model is converged. For the eICU database case study, there are 150 hospitals (silos), and each hospital has its private patient records.

7.5 Experiment Setup

The experiment setup follows the same methodology described in Chapter 3, with a few exceptions. The baselines used are *FedAvg*, *FedSim* (Chapter 4) and *FedFT* (Chapter 5). Model details and hyper-parameters are as follows:

Model A classification approach has been employed to predict the mortality outcome of a patient in the eICU database. Specifically, a multinomial logistic regression model was selected, consistent with previous experiment setups that utilised *FedSim* and *FedFT*. The model considers 40 identified features and incorporates an L2 regulariser with a value of 0.001 to mitigate overfitting and enhance model generalisation.

Hyper-parameters The number of clients selected per round is 20 as there are 150 hospitals in the dataset. Due to its closed nature, more clients can typically be selected per round in a cross-silo setting. The number of communication rounds is 500 to ensure the models achieve good performance. For local training, we select a batch size of 10 to balance the learning steps and computational efficiency; the number of training epochs is set to 20 to avoid significant overfitting, and a learning rate of 0.1 to optimise stability. Cluster size set to 3 for *FedSim*. The aim is to split the 20 clients selected per round into 3 clusters based on local updates.

7.5.1 Summary of Experiments and Evaluation Metrics

Multiple experiments are carried out to compare the performance of the baselines. This experiment's objective was to analyse the extent to which a similarity-guided aggregation method could enhance the performance of the eICU case study. Later, the *FedFT*

algorithm was utilised to evaluate the influence of the frequency transformation approach on both *FedAvg* and *FedSim*. The primary evaluation is the test accuracy of the global model against individual client test data (same metric used to evaluate *FedSim* and *FedFT*). To ensure the accuracy of the results, we will run the experiments 35 times with unique random seeds.

7.5.2 Ethical Considerations

It is important to note that eICU involves the ICU records of actual patients, but the data has been completely anonymised and has already gone through an ethical approval process. The ethical statement below is from the eICU website.

> "The study is exempt from institutional review board approval due to the retrospective design, lack of direct patient intervention, and the security schema, for which the re-identification risk was certified as meeting safe harbor standards by an independent privacy expert (Privacert, Cambridge, MA) (Health Insurance Portability and Accountability Act Certification no. 1031219-2)."

7.6 Results and Discussion

This section will analyse each experiment to ensure a complete understanding of the impact of *FedSim* and *FedFT* on the eICU healthcare case study. Our research has confirmed that these solutions are highly effective in improving FL model performance. First, we explore the performance analysis of *FedAvg* and *FedSim*, then we investigate the effect of *FedFT*, and finally, we compare the significance and reliability of the results.

7.6.1 Performance Analysis of *FedAvg* and *FedSim*

We compare the performance of the widely recognised *FedAvg* methodology against our proposed *FedSim* approach. Figure 7.8 plots the average test accuracy over communications rounds.

The orange dotted line represents *FedAvg*, and the blue solid line is *FedSim*. The first thing to observe is that the accuracy of the model reaches above 70% within the first few rounds. Reaching a test accuracy of such is understandable as most of the data is biased towards non-mortality, which is the majority class. Before the SMOTE process, the models started to overfit towards the majority class and training was stationery from the early stages. Using SMOTE has led to a more balanced and challenging dataset for

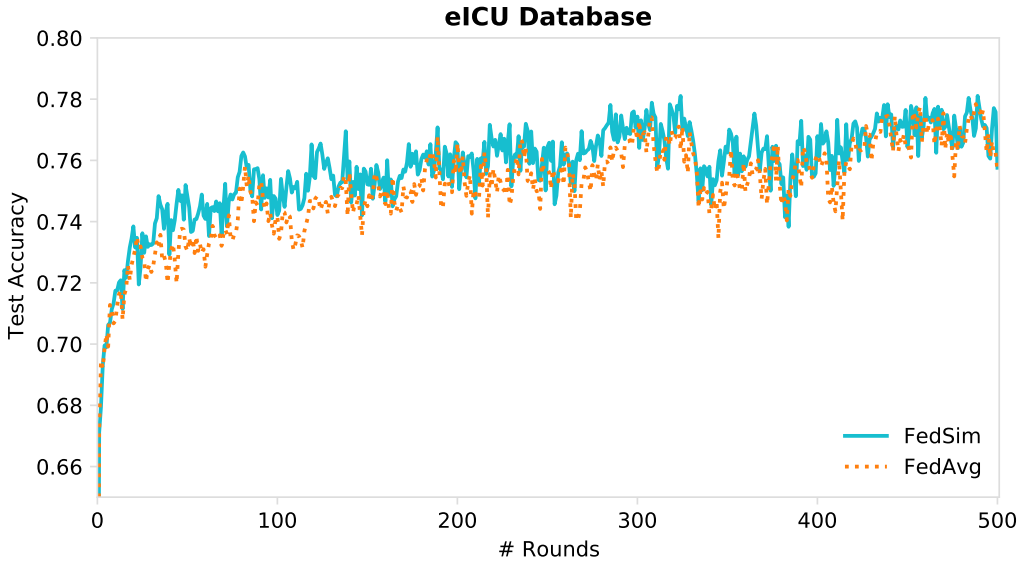

Figure 7.8: *FedAvg* vs *FedSim* on the eICU database

training. As a result of these pre-processing efforts the models were able to achieve test accuracies of 77% with *FedAvg* and 78% with *FedSim*.

FedSim works best when there is similarity among clients, in this case among hospitals. We see improvements in test accuracies throughout the communication rounds for the *FedSim* algorithm. From the 20 hospitals selected in each round, 3 clusters are created based on their model parameters. We decided to use a cluster count of three after carefully analysing the database and considering domain-specific knowledge. For example, if the cluster size is seven *FedSim* would not benefit from this scale as there is not enough dissimilarity among clients to divide them into seven clusters. In rounds 100-150, *FedSim* showcases stability and higher performance than *FedAvg*, and it maintains overall better performance throughout the communication rounds. Another notable observation is that the highest accuracy achieved by *FedSim* is at the 324th round, which is 78.1%, and *FedAvg* achieves its peak accuracy at the 488th round, which is 77.8%. There is a gap over 100 communication rounds where the *FedAvg* achieves its peak performance. This improvement with *FedSim* could significantly reduce computation and communication costs in an FL setting.

Next, we analyse the statistical significance between *FedSim* and *FedAvg* using a T-test with a one-sided alternative hypothesis that *FedSim* performs better than *FedAvg*. Figure

7.9 plots the test accuracy improvements as a percentage between *FedAvg* and *FedSim* over communication rounds.

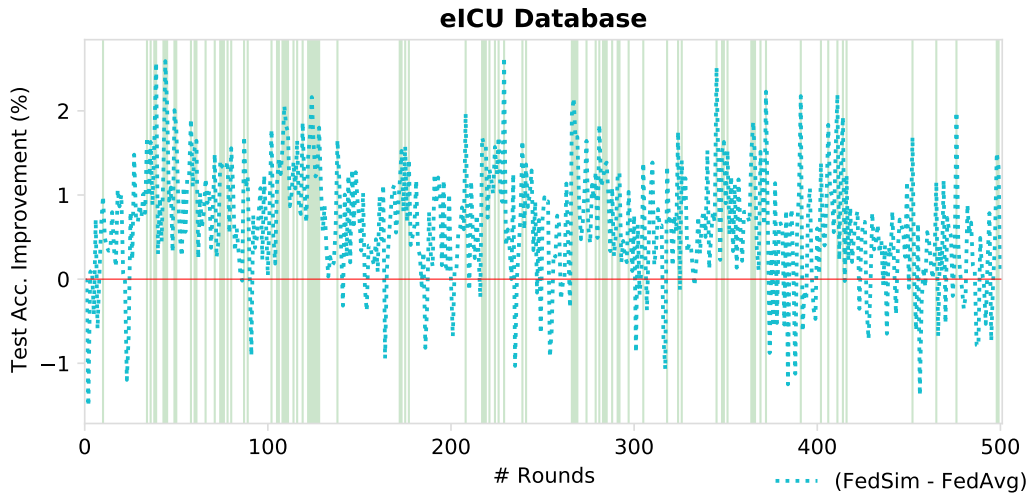

Figure 7.9: Statistical significance between *FedAvg* vs *FedSim* on the eICU database

Values above the zero line indicate that *FedSim* performs better than *FedAvg*. The shaded portions (green colour) in the graphs represent the rounds where *FedSim* has performed significantly better (i.e. p-value is less than 0.05), as determined by the T-test with the `scipy.stats.ttest_ind` library.

In Figure 7.9, the accuracy improvements are the majority on the positive side and some over 2% significant performance gains. It is worth noting that the experiment's accuracy peaks up to 78%, but the model's improvements are observed around 70%. Therefore, a 2% gain in performance is a good acceptance criterion in this case study. The performance drop of over 1% is observed only in very few rounds, suggesting that the proposed *FedSim* algorithm can maintain its performance and stability in large-scale experiments like the eICU case study.

These results demonstrate the positive effect of incorporating *FedSim* even in a large-scale cross-silo setting like the eICU case study. The results we analysed further strengthen the generalisability aspect of *FedSim*.

7.6.2 Evaluating the Effect of *FedFT* with *FedAvg* and *FedSim*

We apply the proposed *FedFT* method to the eICU case study to evaluate the effect of the frequency space for improving communication performance and model efficiency.

Figure 7.10 plots the average test accuracy across the 500 communication rounds.

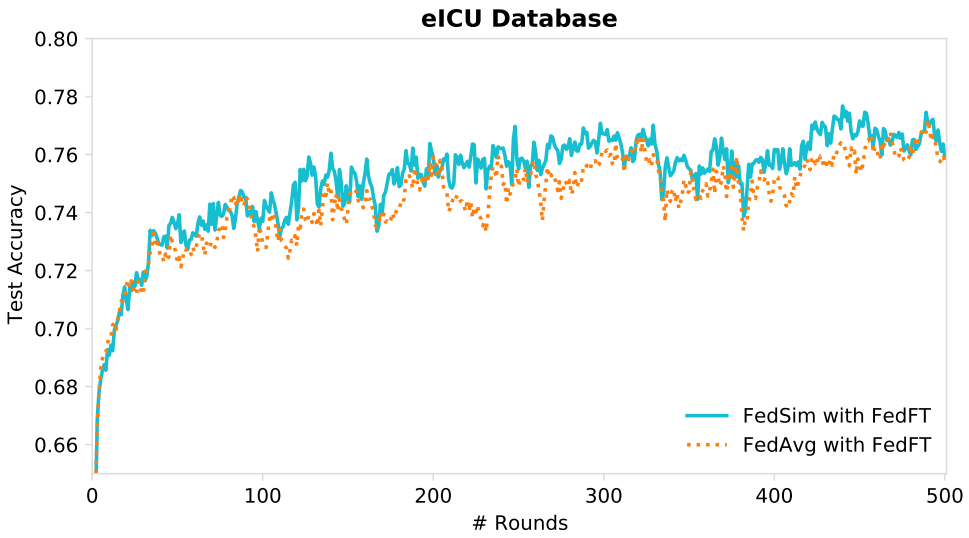

Figure 7.10: *FedAvg* vs *FedSim* with *FedFT* on eICU database

The first observation is that both *FedAvg* and *FedSim* have improved their stability across the training rounds. The previous experiment in Section 7.6.1 showed visibly high fluctuations across the communication rounds. However, when *FedFT* is applied there is a noticeable improvement in stability.

The stability improvement is due to the transformation to frequency space, where minor model parameters are ignored. This effect is due to the reduced variance in the frequency space transformation. The impact on variance with *FedFT* was discussed in detail in Chapter 5. The negative impact of the reduced variance is that performance is reduced in each round. However, performance is still on an upward trend with each round. The peak performance achieved in this experiment by *FedAvg* is 77.1% at round 490, and *FedSim* achieves 77.6% at round 440. It is evident that *FedSim* still maintains its performance benefits even with *FedFT*.

Similar to Section 7.6.1, we perform statistical analysis on the results with *FedFT* as well. Figure 7.11 plots the average test accuracy improvements over communication rounds. The green highlights represent the rounds where *FedSim* performs significantly better than *FedAvg* when utilised with *FedFT*.

We observe that the performance improvements demonstrated fewer variations than the results without *FedFT* in Figure 7.9. Also, we observe that the negative performance

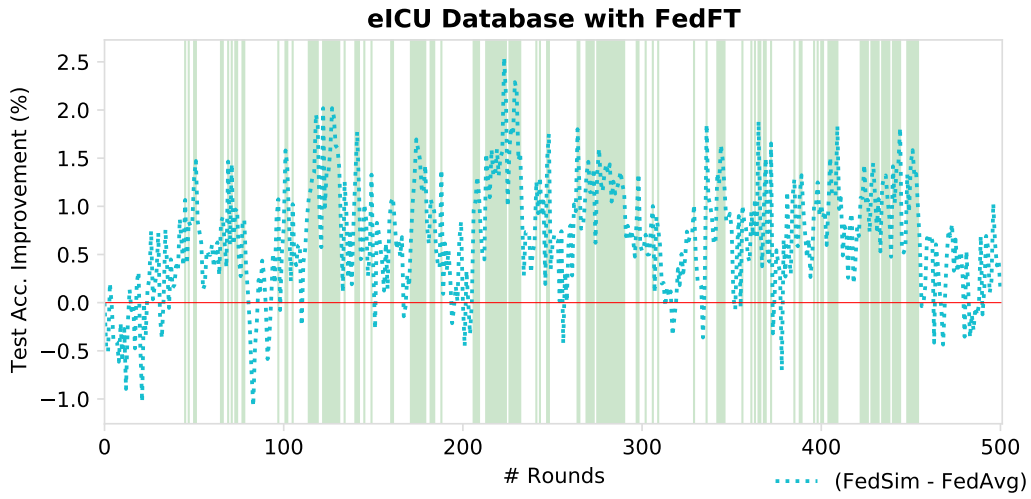

Figure 7.11: Statistical significance between *FedAvg* vs *FedSim* on the eICU database with *FedFT*

points are lesser than running without *FedFT* for *FedSim*.

The study's results suggest that both *FedSim* and *FedFT* display distinct advantages, and their combined use leads to even better performance and stability. Furthermore, it is worth noting that both methodologies demonstrate a high level of generalisability without any modifications to their underlying algorithms.

7.6.3 Results Summary

Table 7.2 summarises the results obtained from the eICU case study. The highest performance with the lowest number of communication rounds required was achieved by *FedSim*. Using *FedSim* saves 164 rounds before reaching its peak accuracy in the vanilla setting, which can be considered a significant saving in compute and communication resources. In the *FedFT* experiments, *FedSim* performed better, achieving peak performance with 50 rounds fewer than its counterpart, *FedAvg*.

These results suggest the effectiveness of the proposed methods and their generalisability to real-world problems. Appendix C provides a consolidated set of results illustrating the impact of different pruning percentages in *FedFT*.

Table 7.2: Peak accuracy and communication rounds for different methods

Method	**Highest Accuracy**	**Com. Round**
FedAvg	77.8%	488
FedSim	**78.1%**	**324**
FedAvg with *FedFT*	77.1%	490
FedSim with *FedFT*	77.6%	440

7.7 Conclusion

In this chapter, we applied the *FedSim* and *FedFT* methods to address a real-world problem in the healthcare domain with the eICU database. We demonstrated their practical applicability in this context by leveraging the similarity-guided aggregation of *FedSim* and the enhanced communication performance of *FedFT*. Our experiments demonstrated that these methods effectively handle the unique challenges posed by healthcare datasets. This successful application underscores the potential of *FedSim* and *FedFT* to enhance FL systems, particularly in domains where data privacy and communication costs are critical concerns.

Chapter 8

Conclusion

This thesis has investigated approaches to improving FL, a similarity-guided aggregation method and enhancing communication performance using the frequency space. We identified a lack of aggregation methods that leverage client similarity knowledge and those that are generalisable across various applications. Additionally, there was a gap in the use of the frequency space in FL, which could potentially benefit both communication performance and security. In this thesis, we addressed the following three research questions:

RQ1: To what extent does the identification and utilisation of similarity knowledge among clients influence model aggregation in FL?

RQ2: Building on the insights from RQ1, how does the chosen aggregation strategy impact communication efficiency in FL, and in what ways can model compression and pruning enhance this efficiency?

RQ3: How do the communication efficiency strategies identified in RQ2 affect the security of FL in terms of data privacy?

Six objectives were identified to address these research questions. This chapter discusses the contributions of this thesis by revisiting the initial objectives and summarising the key contributions from each chapter. Additionally, we present the limitations of our proposed methods and outline directions for future work.

8.1 Objectives Revisited

This section revisits all the objectives identified in Chapter 1 and discusses how these objectives were met. By reflecting on each objective, we highlight the key contributions made in this thesis, summarising the progress and findings from each chapter.

O1: Conduct a comprehensive literature review to identify and analyse existing FL aggregation methodologies.

This objective focuses on conducting a comprehensive review of various aggregation methods, similarity-based approaches, statistical heterogeneity, and security in FL. We examined several essential aggregation methods foundational to FL and domain-specific methodologies. We also discussed the effect of statistical heterogeneity in FL, followed by the communication and security challenges. This objective was addressed in the literature review chapter (Chapter 2). The key findings include that, for aggregation methods in FL, there are only a limited number of methods generalisable across different problems and datasets. This lack of generalisability in FL aggregation methods has motivated us to design algorithms considering a wide range of applications in FL.

Next, we reviewed the impact of non-IIDness in FL and highlighted the importance of developing methods that support its non-IID nature. We then explored the aspect of communication efficiency in FL and identified a gap that needs to be addressed to help lower the entry barrier to FL adoption. Lastly, we examined the security implications in FL and the various types of attacks that can occur. A common type of attack identified is the gradient inversion attack, which can recreate client data.

O2: Develop a similarity-weighted aggregation method that harnesses commonalities in learning behaviour between client models to improve accuracy in FL.

The literature showed a notable absence of similarity-based aggregation methods for FL. Furthermore, most existing aggregation methods were tailored to specific applications with fixed conditions (e.g., specific neural models and data types), limiting their generalisability. To address this gap, we developed *FedSim*, an innovative aggregation strategy that leverages inter-client relationships modelled as pairwise similarity in gradients without sharing client data (Chapter 4).

FedSim employs a cluster-based approach where there are two rounds of aggregation. First, all clients within a cluster are aggregated based on their contributions, forming cluster-level models. Then, these cluster-level models are aggregated through a global aggregation step, which performs an average aggregation to create the final global model. Once the selected clients in a round perform a single forward pass on the latest global model, we create clusters based on the similarity of weights. To enhance the efficiency of the clustering process, we employ the PCA technique to reduce dimensionality. The primary aim of this objective was to enhance accuracy in FL by utilising similarity knowledge.

We conducted an extensive study containing multiple scenarios and datasets to achieve this. This comprehensive evaluation helped us understand the applicability and effectiveness of the proposed *FedSim* method in improving FL performance. In our evaluation, we compared *FedSim* with two state-of-the-art baselines, *FedAvg* and *FedProx*. The results confirmed that *FedSim* effectively captures the similarity knowledge among clients and significantly improves performance. Our findings also suggest that different datasets and applications can have varying levels of embedded similarity knowledge.

O3: Evaluate the generalisability of the similarity-weighted aggregation from Objective 2 on different model architectures and diverse datasets.

To assess the generalisability of the proposed *FedSim* method, as outlined in Chapter 4, we conducted a series of comprehensive experiments. Initially, we evaluated the method using four real-world datasets: two image datasets, one sensor data dataset and one text data dataset. This diverse selection allowed us to investigate the applicability of *FedSim* across various domains.

Subsequently, we tested *FedSim* with three complex neural architectures: a 2D CNN, a three-layer MLP and an RNN. This experimentation was crucial in understanding the impact of *FedSim* on different neural architectures and its adaptability to various model structures. Additionally, we conducted a study using six synthetic datasets to examine the effects of non-IIDness on *FedSim*. This step was essential to evaluate the method's robustness in handling statistical heterogeneity among clients. To measure the level of statistical heterogeneity among clients, we introduced a novel metric called the *PNI*. This metric utilizes the model error, explicitly using the root mean square error, to provide a non-IID score. The *PNI* measure

offers a quantifiable assessment of the degree of non-IIDness in the data.

The comprehensive evaluation across multiple datasets and neural architectures demonstrated the generalisability and effectiveness of the proposed *FedSim* method, thereby addressing the objective thoroughly.

O4: Develop an FL algorithm to improve communication performance, ensuring adaptability across diverse FL scenarios

Improving communication efficiency in FL is a highly challenging research area. FL's distributed nature requires significant bandwidth to scale effectively. This communication demand often poses a considerable entry barrier for applications to adopt FL as a secure method for training ML models. Various methods in the literature aim to enhance communication performance, yet they often impact model performance and are limited by different aggregation methods and datasets.

Through this objective, we aimed to develop an FL algorithm that reduces communication costs while remaining generalisable. In Chapter 5, we introduced *FedFT*, a novel approach that uses frequency space (the frequency space) to communicate and aggregate models efficiently. DCT-IV was employed as the transformation function, and a simple pruning technique effectively reduced communication costs while maintaining model accuracy. A fundamental consideration was ensuring that the proposed method was generalisable and supported the *FedSim* method introduced in Chapter 4.

Experiments conducted on four FL datasets across three FL methodologies demonstrated the generalisability of *FedFT*. With comparable model accuracy, communication savings of $5\% - 30\%$ were achieved, depending on the dataset and configuration.

O5: Evaluate the integrated security benefits within the communication optimised algorithm

Based on the literature review, we identified several potential security threats in FL. These threats can undermine FL's core principle of preserving the privacy of client data. One particularly common and risky type of attack is the gradient inversion attack, which attempts to recreate the client data that the model was trained on. Given the significant threat this poses to privacy in FL, we explored

existing defence mechanisms and found that some require special changes or additional compute resources.

This objective evaluates the security enhancements integrated into the communication optimisation algorithm developed in Objective 4 (i.e. *FedFT*). The effectiveness of these security measures in protecting client data during transmission will be evaluated. Specifically, we assess how the proposed *FedFT* method can help improve security in FL. In Chapter 6, we introduced pFGD, a defence mechanism designed to mitigate gradient inversion attacks in FL. By applying *FedFT* and incorporating pruning before communication, pFGD effectively enhances the resilience of FL models against such attacks.

O6: Conduct a case study to validate the proposed methodologies in a real-world context

The final objective was to evaluate the proposed methodologies, *FedSim* and *FedFT*, on a complex real-world application. We selected the healthcare domain and chose the eICU database to assess the efficacy of our methods. The eICU database includes over 200,000 ICU admissions from more than 200 hospitals, providing substantial data that closely mirrors real-world FL applications. Mortality prediction was considered a classification problem and we trained a model to classify patient mortality based on 40 features from the database.

In Chapter 7, we evaluate the eICU database using the proposed methods. First we evaluated the effect of *FedSim* compared to *FedAvg* in this case study. The results indicated that *FedSim* achieved peak accuracy and reduced the number of communication rounds by 164 to reach this peak. The second experiment compared the effectiveness and applicability of *FedFT* in the case study. The *FedAvg* and *FedSim* methods showed that *FedSim* with *FedFT* performed better, achieving significant savings in compute and communication resources.

Our experiments demonstrated that *FedSim* and *FedFT* methods effectively address the unique challenges posed by healthcare datasets. This successful application highlights the potential of these methods to significantly enhance FL systems, especially in domains where data privacy and communication efficiency are essential. The ability of *FedSim* and *FedFT* to maintain high performance while optimising communication costs highlights their value in practical, privacy-sensitive environments such as healthcare.

8.2 Limitations and Future Work

In this section, we discuss some of the limitations of the work presented in this thesis and highlight areas for future research.

8.2.1 *FedSim*

Chapter 4 introduced the similarity-guided model aggregation method, *FedSim*. While evaluating its impact on statistical heterogeneity using synthetic datasets, we observed performance degradation in extreme cases of non-IID and IID distributions. In scenarios of extreme non-IIDness, there was insufficient similarity knowledge to exploit, limiting *FedSim*'s effectiveness. On the other hand, in highly IID settings, where client data is overly homogeneous, *FedSim* faced challenges during the clustering step, as the similarity among clients was too high to benefit from clustering. However, our real-world experiments demonstrated that *FedSim* performs well in moderately non-IID environments. In practical applications, it is rare to encounter extreme cases of either non-IIDness or IIDness, suggesting that *FedSim* is well-suited for most real-world scenarios.

Another limitation of *FedSim* lies in its fixed cluster size selection. The choice of cluster size is crucial for the method's success, requiring domain knowledge and technical expertise. This fixed nature may lead to suboptimal performance across different datasets or applications where varying cluster sizes might be more appropriate.

We identified the absence of a dynamic switching mechanism between different aggregation methods. Depending on the dataset characteristics, various base aggregation methods could benefit from adopting this approach. For example, some datasets may perform significantly better when *FedSim* is combined with *FedAvg*, while others might excel when paired with *FedProx*.

Future research could explore the following areas to address these limitations:

Dynamic Clustering Developing adaptive clustering mechanisms that can adjust cluster sizes based on real-time data distribution could enhance *FedSim* 's performance across a broader range of non-IID and IID settings.

Extending Similarity Measures The similarity measures used in *FedSim* could be further refined or replaced with more sophisticated techniques to capture better very high-dimensional data, particularly in scenarios involving large language models (LLMs).

Switching Aggregation Methods Developing adaptive aggregation methods that can switch between different approaches based on the real-time nature of clusters or client data could further optimise the performance and flexibility of *FedSim*.

8.2.2 *FedFT*

In Chapter 5, *FedFT* was introduced as a method to improve communication performance in FL by leveraging the frequency space. The proposed approach demonstrated significant robustness and generalisability across various scenarios, with relatively few limitations identified due to its foundation in the frequency space.

One notable limitation of *FedFT* is the challenge of determining the optimal pruning percentage that maximises communication efficiency. A promising direction for future research involves dynamically identifying the 'sweet spot' for pruning by analysing real-time training patterns. Another limitation, similar to what was observed with *FedSim*, is handling extreme cases of IIDness and non-IIDness. In an extremely IID setting, all clients may train similarly, leading to minimal gradient differences, which could impact the effectiveness of the frequency space transformation and pruning. On the other hand, in an extreme non-IID scenario the vast differences in gradients across clients may result in significant information loss during pruning.

Addressing these limitations opens up several promising research directions. For instance, future work could explore dynamic adjustment methods that respond to varying degrees of statistical heterogeneity. By continuing to refine and expand upon *FedFT*, it is possible to enhance further its applicability and effectiveness in improving communication efficiency in diverse and challenging FL environments.

8.2.3 *p*FGD

In Chapter 6, *p*FGD was introduced as a method to defend against gradient inversion attacks by leveraging the *FedFT* method. This method was specifically evaluated using DLG and iDLG, two widely recognised gradient inversion attack methods. The results demonstrated *p*FGD's effectiveness in mitigating these attacks, showcasing its potential as a robust defence mechanism within FL

However, a key limitation of the current work is the narrow scope of evaluation. While *p*FGD has proven effective against DLG and iDLG, its performance against other types of privacy attacks remains unexplored. Privacy attacks in FL are diverse, including but not limited to membership inference attacks, gradient inversion attacks and reconstruction

attacks. Evaluating pFGD against a broader range of these attack types would provide a more comprehensive understanding of its robustness and limitations.

Another important consideration is how pFGD compares to more complex and established privacy-preserving techniques, such as Differential Privacy and Homomorphic Encryption. A comparative analysis between pFGD and these techniques would help understand its effectiveness, highlighting scenarios where pFGD might offer advantages or benefit from integration with other methods.

Future research should focus on expanding the evaluation of pFGD to include a variety of privacy attacks and benchmarking its performance against more complex and established methods. This would strengthen the validation of pFGD and provide valuable insights into how it can be optimised or combined with other privacy-preserving techniques to enhance security in FL systems.

Bibliography

Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., Corrado, G. S., Davis, A., Dean, J., Devin, M., Ghemawat, S., Goodfellow, I., Harp, A., Irving, G., Isard, M., Jia, Y., Jozefowicz, R., Kaiser, L., Kudlur, M., Levenberg, J., Mané, D., Monga, R., Moore, S., Murray, D., Olah, C., Schuster, M., Shlens, J., Steiner, B., Sutskever, I., Talwar, K., Tucker, P., Vanhoucke, V., Vasudevan, V., Viégas, F., Vinyals, O., Warden, P., Wattenberg, M., Wicke, M., Yu, Y., and Zheng, X. (2015). TensorFlow: Large-scale machine learning on heterogeneous systems. Software available from tensorflow.org.

Ahmed, N., Natarajan, T., and Rao, K. R. (1974). Discrete cosine transform. *IEEE transactions on Computers*, 100(1):90–93.

Al-Rubaie, M. and Chang, J. M. (2019). Privacy-preserving machine learning: Threats and solutions. *IEEE Security & Privacy*, 17(2):49–58.

Aledhari, M., Razzak, R., Parizi, R. M., and Saeed, F. (2020). Federated learning: A survey on enabling technologies, protocols, and applications. *IEEE Access*, 8:140699–140725.

Arthur, D. and Vassilvitskii, S. (2007). k-means++: the advantages of careful seeding. In *Proceedings of the Eighteenth Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '07, page 1027–1035, USA. Society for Industrial and Applied Mathematics.

Barlow, H., Mao, S., and Khushi, M. (2019). Predicting high-risk prostate cancer using machine learning methods. *Data*, 4(3):129.

Baucas, M. J., Spachos, P., and Plataniotis, K. N. (2023). Federated learning and blockchain-enabled fog-iot platform for wearables in predictive healthcare. *IEEE Transactions on Computational Social Systems*.

Beyer, S. E., Salgado, C., Garçao, I., Celi, L. A., and Vieira, S. (2021). Circadian rhythm in critically ill patients: insights from the eicu database. *Cardiovascular Digital Health Journal*, 2(2):118–125.

Bonawitz, K., Eichner, H., Grieskamp, W., Huba, D., Ingerman, A., Ivanov, V., Kiddon, C., Konecny, J., Mazzocchi, S., McMahan, H. B., et al. (2019). Towards federated learning at scale: System design. *arXiv preprint arXiv:1902.01046*.

Brisimi, T. S., Chen, R., Mela, T., Olshevsky, A., Paschalidis, I. C., and Shi, W. (2018). Federated

learning of predictive models from federated electronic health records. *International journal of medical informatics*, 112:59–67.

Britanak, V. (2003). The fast dct-iv/dst-iv computation via the mdct. *Signal Processing*, 83(8):1803–1813.

Caldas, S., Duddu, S. M. K., Wu, P., Li, T., Konečnỳ, J., McMahan, H. B., Smith, V., and Talwalkar, A. (2018). Leaf: A benchmark for federated settings. *arXiv preprint arXiv:1812.01097*.

Chawla, N. V., Bowyer, K. W., Hall, L. O., and Kegelmeyer, W. P. (2002). Smote: synthetic minority over-sampling technique. *Journal of artificial intelligence research*, 16:321–357.

Chen, C. and Campbell, N. D. (2021). Understanding training-data leakage from gradients in neural networks for image classification. *arXiv preprint arXiv:2111.10178*.

Chen, H. and Koushanfar, F. (2022). Fl-talk: Covert communication in federated learning via spectral steganography. In *Workshop on Trustworthy and Socially Responsible Machine Learning, NeurIPS 2022*.

Cheu, A., Smith, A., Ullman, J., Zeber, D., and Zhilyaev, M. (2019). Distributed differential privacy via shuffling. In *Advances in Cryptology–EUROCRYPT 2019: 38th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Darmstadt, Germany, May 19–23, 2019, Proceedings, Part I 38*, pages 375–403. Springer.

Cooley, J. W. and Tukey, J. W. (1965). An algorithm for the machine calculation of complex fourier series. *Mathematics of computation*, 19(90):297–301.

Dai, X., Yan, X., Zhou, K., Yang, H., Ng, K. K., Cheng, J., and Fan, Y. (2019). Hyper-sphere quantization: Communication-efficient sgd for federated learning. *arXiv preprint arXiv:1911.04655*.

Dimililer, K. (2022). Dct-based medical image compression using machine learning. *Signal, Image and Video Processing*, 16(1):55–62.

Duan, M., Liu, D., Ji, X., Liu, R., Liang, L., Chen, X., and Tan, Y. (2020). Fedgroup: Ternary cosine similarity-based clustered federated learning framework toward high accuracy in heterogeneity data. *arXiv preprint arXiv:2010.06870*.

Duchi, J., Hazan, E., and Singer, Y. (2011). Adaptive subgradient methods for online learning and stochastic optimization. *Journal of machine learning research*, 12(7).

Dwork, C. (2006). Differential privacy. In *International colloquium on automata, languages, and programming*, pages 1–12. Springer.

Dwork, C., McSherry, F., Nissim, K., and Smith, A. (2006). Calibrating noise to sensitivity in private data analysis. In *Theory of Cryptography: Third Theory of Cryptography Conference, TCC 2006, New York, NY, USA, March 4-7, 2006. Proceedings 3*, pages 265–284. Springer.

El Ouadrhiri, A. and Abdelhadi, A. (2022). Differential privacy for deep and federated learning: A survey. *IEEE access*, 10:22359–22380.

Fang, H. and Qian, Q. (2021). Privacy preserving machine learning with homomorphic encryption and federated learning. *Future Internet*, 13(4).

Florescu, L. M., Streba, C. T., Şerbănescu, M.-S., Mămuleanu, M., Florescu, D. N., Teică, R. V., Nica, R. E., and Gheonea, I. A. (2022). Federated learning approach with pre-trained deep learning models for covid-19 detection from unsegmented ct images. *Life*, 12(7):958.

Gao, D., Ju, C., Wei, X., Liu, Y., Chen, T., and Yang, Q. (2019). Hhhfl: Hierarchical heterogeneous horizontal federated learning for electroencephalography. *arXiv preprint arXiv:1909.05784*.

Geiping, J., Bauermeister, H., Dröge, H., and Moeller, M. (2020). Inverting gradients-how easy is it to break privacy in federated learning? *Advances in Neural Information Processing Systems*, 33:16937–16947.

Goldberger, A. L., Amaral, L. A., Glass, L., Hausdorff, J. M., Ivanov, P. C., Mark, R. G., Mietus, J. E., Moody, G. B., Peng, C.-K., and Stanley, H. E. (2000). Physiobank, physiotoolkit, and physionet: components of a new research resource for complex physiologic signals. *circulation*, 101(23):e215–e220.

Han, B., Jhaveri, R., Wang, H., Qiao, D., and Du, J. (2021). Application of robust zero-watermarking scheme based on federated learning for securing the healthcare data. *IEEE journal of biomedical and health informatics*.

Hard, A., Partridge, K., Nguyen, C., Subrahmanya, N., Shah, A., Zhu, P., Moreno, I. L., and Mathews, R. (2020). Training keyword spotting models on non-iid data with federated learning. *arXiv preprint arXiv:2005.10406*.

Hard, A., Rao, K., Mathews, R., Ramaswamy, S., Beaufays, F., Augenstein, S., Eichner, H., Kiddon, C., and Ramage, D. (2018a). Federated learning for mobile keyboard prediction. *arXiv preprint arXiv:1811.03604*.

Hard, A., Rao, K., Mathews, R., Ramaswamy, S., Beaufays, F., Augenstein, S., Eichner, H., Kiddon, C., and Ramage, D. (2018b). Federated learning for mobile keyboard prediction. *arXiv preprint arXiv:1811.03604*.

He, Y., Shen, Z., and Cui, P. (2019). Towards non-i.i.d. image classification: A dataset and baselines.

Hitaj, B., Ateniese, G., and Perez-Cruz, F. (2017). Deep models under the gan: information leakage from collaborative deep learning. In *Proceedings of the 2017 ACM SIGSAC conference on computer and communications security*, pages 603–618.

Hoy, M. B. (2018). Alexa, siri, cortana, and more: an introduction to voice assistants. *Medical reference services quarterly*, 37(1):81–88.

Hyeon-Woo, N., Ye-Bin, M., and Oh, T.-H. (2021). Fedpara: Low-rank hadamard product for communication-efficient federated learning. *arXiv preprint arXiv:2108.06098*.

Jere, M. S., Farnan, T., and Koushanfar, F. (2020). A taxonomy of attacks on federated learning. *IEEE Security & Privacy*, 19(2):20–28.

Jiang, Y., Wang, S., Valls, V., Ko, B. J., Lee, W.-H., Leung, K. K., and Tassiulas, L. (2022). Model pruning enables efficient federated learning on edge devices. *IEEE Transactions on Neural Networks and Learning Systems*.

Jin, W., Yao, Y., Han, S., Gu, J., Joe-Wong, C., Ravi, S., Avestimehr, S., and He, C. (2024). Fedml-he: An efficient homomorphic-encryption-based privacy-preserving federated learning system.

Johnson, A., Bulgarelli, L., Pollard, T., Horng, S., Celi, L., and Mark, R. (2020). Mimic-iv (version 1.0).

Kairouz, P., McMahan, H. B., Avent, B., Bellet, A., Bennis, M., Bhagoji, A. N., Bonawitz, K., Charles, Z., Cormode, G., Cummings, R., et al. (2019). Advances and open problems in federated learning. *arXiv preprint arXiv:1912.04977*.

Karimireddy, S. P., Kale, S., Mohri, M., Reddi, S., Stich, S., and Suresh, A. T. (2020). Scaffold: Stochastic controlled averaging for federated learning. In *International Conference on Machine Learning*, pages 5132–5143. PMLR.

Kasiviswanathan, S. P., Lee, H. K., Nissim, K., Raskhodnikova, S., and Smith, A. (2011). What can we learn privately? *SIAM Journal on Computing*, 40(3):793–826.

Kingma, D. P. and Ba, J. (2014). Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Konečnỳ, J., McMahan, H. B., Yu, F. X., Richtárik, P., Suresh, A. T., and Bacon, D. (2016). Federated learning: Strategies for improving communication efficiency. *arXiv preprint arXiv:1610.05492*.

Lam, E. Y. and Goodman, J. W. (2000). A mathematical analysis of the dct coefficient distributions for images. *IEEE transactions on image processing*, 9(10):1661–1666.

LeCun, Y., Bottou, L., Bengio, Y., and Haffner, P. (1998). Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324.

Lee, J., Sun, J., Wang, F., Wang, S., Jun, C.-H., and Jiang, X. (2018). Privacy-preserving patient similarity learning in a federated environment: development and analysis. *JMIR medical informatics*, 6(2):e20.

Leroy, D., Coucke, A., Lavril, T., Gisselbrecht, T., and Dureau, J. (2019). Federated learning for keyword spotting. In *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6341–6345. IEEE.

Li, A., Sun, J., Wang, B., Duan, L., Li, S., Chen, Y., and Li, H. (2020). Lotteryfl: Personalized and communication-efficient federated learning with lottery ticket hypothesis on non-iid datasets.

Li, Q., Wen, Z., and He, B. (2019). Federated learning systems: Vision, hype and reality for data privacy and protection. *arXiv preprint arXiv:1907.09693*.

Li, T., Sahu, A. K., Zaheer, M., Sanjabi, M., Talwalkar, A., and Smith, V. (2018). Federated optimization in heterogeneous networks. *arXiv preprint arXiv:1812.06127*.

Lin, Y., Han, S., Mao, H., Wang, Y., and Dally, W. J. (2017). Deep gradient compression: Reducing the communication bandwidth for distributed training. *CoRR*, abs/1712.01887.

Liu, D. C. and Nocedal, J. (1989). On the limited memory bfgs method for large scale optimization. *Mathematical programming*, 45(1-3):503–528.

Liu, Z., Xu, J., Peng, X., and Xiong, R. (2018). Frequency-domain dynamic pruning for convolutional neural networks. *Advances in neural information processing systems*, 31.

Liu, Z., Xu, Z., Coleman, B., and Shrivastava, A. (2024). One-pass distribution sketch for measuring data heterogeneity in federated learning. *Advances in Neural Information Processing Systems*, 36.

Lyu, L., Yu, H., Ma, X., Chen, C., Sun, L., Zhao, J., Yang, Q., and Philip, S. Y. (2022). Privacy and robustness in federated learning: Attacks and defenses. *IEEE transactions on neural networks and learning systems*.

Ma, C., Li, J., Ding, M., Yang, H. H., Shu, F., Quek, T. Q., and Poor, H. V. (2020). On safeguarding privacy and security in the framework of federated learning. *IEEE network*, 34(4):242–248.

Maity, S. K., Panigrahi, A., and Mukherjee, A. (2018). Analyzing social book reading behavior on goodreads and how it predicts amazon best sellers. In *IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining*, pages 211–235. Springer.

Mamoshina, P., Vieira, A., Putin, E., and Zhavoronkov, A. (2016). Applications of deep learning in biomedicine. *Molecular pharmaceutics*, 13(5):1445–1454.

McMahan, B., Moore, E., Ramage, D., Hampson, S., and y Arcas, B. A. (2017). Communication-efficient learning of deep networks from decentralized data. In *Artificial Intelligence and Statistics*, pages 1273–1282. PMLR.

Melis, L., Song, C., De Cristofaro, E., and Shmatikov, V. (2019). Exploiting unintended feature leakage in collaborative learning. In *2019 IEEE symposium on security and privacy (SP)*, pages 691–706. IEEE.

Mohr, D. C., Zhang, M., and Schueller, S. M. (2017). Personal sensing: understanding mental health using ubiquitous sensors and machine learning. *Annual review of clinical psychology*, 13:23–47.

Mothukuri, V., Parizi, R. M., Pouriyeh, S., Huang, Y., Dehghantanha, A., and Srivastava, G. (2021). A survey on security and privacy of federated learning. *Future Generation Computer Systems*, 115:619–640.

Niu, C., Wu, F., Tang, S., Hua, L., Jia, R., Lv, C., Wu, Z., and Chen, G. (2020). Billion-scale federated learning on mobile clients: A submodel design with tunable privacy. In *Proceedings of the 26th Annual International Conference on Mobile Computing and Networking*, pages 1–14.

Nour, B., Cherkaoui, S., and Mlika, Z. (2021). Federated learning and proactive computation reuse at the edge of smart homes. *IEEE Transactions on Network Science and Engineering*, 9(5):3045–3056.

O'Halloran, H. M., Kwong, K., Veldhoen, R. A., and Maslove, D. M. (2020). Characterizing the patients, hospitals, and data quality of the eicu collaborative research database. *Critical care medicine*, 48(12):1737–1743.

Park, B.-h. and Kargupta, H. (2002). Distributed data mining: Algorithms, systems, and applications. *Data Mining Handbook*, 341-358.

Patel, S., Singh, G., Zarbiv, S., Ghiassi, K., Rachoin, J.-S., et al. (2021). Mortality prediction using sao 2/fio 2 ratio based on eicu database analysis. *Critical care research and practice*, 2021.

Pearson, K. (1901). Liii. on lines and planes of closest fit to systems of points in space. *The London, Edinburgh, and Dublin philosophical magazine and journal of science*, 2(11):559–572.

Pollard, T. J., Johnson, A. E. W., Raffa, J. D., Celi, L. A., Mark, R. G., and Badawi, O. (2018). The eICU Collaborative Research Database, a freely available multi-center database for critical care research. *Scientific data*, 5(1):1–13.

Prakash, P., Ding, J., Chen, R., Qin, X., Shu, M., Cui, Q., Guo, Y., and Pan, M. (2022). Iot device friendly and communication-efficient federated learning via joint model pruning and quantization. *IEEE Internet of Things Journal*, 9(15):13638–13650.

Ramaswamy, S., Mathews, R., Rao, K., and Beaufays, F. (2019). Federated learning for emoji prediction in a mobile keyboard. *arXiv preprint arXiv:1906.04329*.

Rao, K. R. and Yip, P. (2014). *Discrete cosine transform: algorithms, advantages, applications*. Academic press.

Reddi, S., Charles, Z., Zaheer, M., Garrett, Z., Rush, K., Konečnỳ, J., Kumar, S., and McMahan, H. B. (2020). Adaptive federated optimization. *arXiv preprint arXiv:2003.00295*.

Reisizadeh, A., Mokhtari, A., Hassani, H., Jadbabaie, A., and Pedarsani, R. (2020). Fedpaq: A communication-efficient federated learning method with periodic averaging and quantization. In *International Conference on Artificial Intelligence and Statistics*, pages 2021–2031. PMLR.

Robinson, J. and Kecman, V. (2003). Combining support vector machine learning with the discrete cosine transform in image compression. *IEEE Transactions on Neural Networks*, 14(4):950–958.

Rodríguez-Barroso, N., Jiménez-López, D., Luzón, M. V., Herrera, F., and Martínez-Cámara, E. (2023). Survey on federated learning threats: Concepts, taxonomy on attacks and defences, experimental study and challenges. *Information Fusion*, 90:148–173.

Sabater, C., Bellet, A., and Ramon, J. (2020). Distributed differentially private averaging with improved utility and robustness to malicious parties. *ArXiv*.

Safaei, N., Safaei, B., Seyedekrami, S., Talafidaryani, M., Masoud, A., Wang, S., Li, Q., and Moqri, M. (2022). E-catboost: An efficient machine learning framework for predicting icu mortality using the eicu collaborative research database. *Plos one*, 17(5):e0262895.

Sattler, F., Müller, K.-R., and Samek, W. (2020a). Clustered federated learning: Model-agnostic distributed multitask optimization under privacy constraints. *IEEE Transactions on Neural Networks and Learning Systems*.

Sattler, F., Wiedemann, S., Müller, K.-R., and Samek, W. (2020b). Robust and communication-efficient federated learning from non-i.i.d. data. *IEEE Transactions on Neural Networks and Learning Systems*, 31(9):3400–3413.

Shamir, O., Srebro, N., and Zhang, T. (2014). Communication-efficient distributed optimization using an approximate newton-type method. In *International conference on machine learning*, pages 1000–1008.

Shatte, A. B., Hutchinson, D. M., and Teague, S. J. (2019). Machine learning in mental health: a scoping review of methods and applications. *Psychological medicine*, 49(9):1426–1448.

Sheikhalishahi, S., Balaraman, V., and Osmani, V. (2020). Benchmarking machine learning models on multi-centre eicu critical care dataset. *Plos one*, 15(7):e0235424.

Shokri, R. and Shmatikov, V. (2015). Privacy-preserving deep learning. In *Proceedings of the 22nd ACM SIGSAC conference on computer and communications security*, pages 1310–1321.

Smith, V., Chiang, C.-K., Sanjabi, M., and Talwalkar, A. S. (2017). Federated multi-task learning. In *Advances in Neural Information Processing Systems*, pages 4424–4434.

Strang, G. (1999). The discrete cosine transform. *SIAM review*, 41(1):135–147.

Sweeney, L. (2000). Simple demographics often identify people uniquely. *Health (San Francisco)*, 671:1–34.

Tagliavini, G., Mach, S., Rossi, D., Marongiu, A., and Benini, L. (2017). A transprecision floating-point platform for ultra-low power computing.

Thelwall, M. (2019). Reader and author gender and genre in goodreads. *Journal of Librarianship and Information Science*, 51(2):403–430.

Tian, Y., Wan, Y., Lyu, L., Yao, D., Jin, H., and Sun, L. (2022). Fedbert: When federated learning meets pre-training. *ACM Transactions on Intelligent Systems and Technology (TIST)*.

Vincent, J.-L. and Moreno, R. (2010). Clinical review: scoring systems in the critically ill. *Critical care*, 14:1–9.

Voigt, P. and Von dem Bussche, A. (2017). The eu general data protection regulation (gdpr). *A Practical Guide, 1st Ed., Cham: Springer International Publishing*.

Wan, M., Misra, R., Nakashole, N., and McAuley, J. J. (2019). Fine-grained spoiler detection from large-scale review corpora. In Korhonen, A., Traum, D. R., and Màrquez, L., editors, *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, pages 2605–2610. Association for Computational Linguistics.

Wang, J., Liu, Q., Liang, H., Joshi, G., and Poor, H. V. (2020). Tackling the objective inconsistency problem in heterogeneous federated optimization. *Advances in neural information processing systems*, 33:7611–7623.

Warner, S. L. (1965). Randomized response: A survey technique for eliminating evasive answer bias. *Journal of the American Statistical Association*, 60(309):63–69.

Wei, W., Liu, L., Loper, M., Chow, K.-H., Gursoy, M. E., Truex, S., and Wu, Y. (2020). A framework for evaluating gradient leakage attacks in federated learning. *arXiv preprint arXiv:2004.10397*.

Wijekoon, A., Wiratunga, N., Cooper, K., and Bach, K. (2020). Learning to recognise exercises in the self-management of low back pain. In *The Thirty-Third International Flairs Conference*.

Winograd, S. (1978). On computing the discrete fourier transform. *Mathematics of computation*, 32(141):175–199.

Wiratunga, N., Wijekoon, A., and Cooper, K. (2020). Learning to compare with few data for personalised human activity recognition. In *International Conference on Case-Based Reasoning*, pages 3–14. Springer.

Xu, Y., Chao, S., Niu, Y., et al. (2022). Association between the predicted value of apache iv scores and intensive care unit mortality: A secondary analysis based on eicu dataset. *Computational and Mathematical Methods in Medicine*, 2022.

Yang, H., Ge, M., Xue, D., Xiang, K., Li, H., and Lu, R. (2023). Gradient leakage attacks in federated learning: Research frontiers, taxonomy and future directions. *IEEE Network*.

Ye, M., Fang, X., Du, B., Yuen, P. C., and Tao, D. (2023). Heterogeneous federated learning: State-of-the-art and research challenges. *ACM Computing Surveys*, 56(3):1–44.

Zaheer, M., Reddi, S., Sachan, D., Kale, S., and Kumar, S. (2018). Adaptive methods for nonconvex optimization. *Advances in neural information processing systems*, 31.

Zhang, C., Li, S., Xia, J., Wang, W., Yan, F., and Liu, Y. (2020). Batchcrypt: efficient homomorphic encryption for cross-silo federated learning. In *Proceedings of the 2020 USENIX Conference on Usenix Annual Technical Conference*, USENIX ATC'20, USA. USENIX Association.

Zhang, C., Xie, Y., Bai, H., Yu, B., Li, W., and Gao, Y. (2021). A survey on federated learning. *Knowledge-Based Systems*, 216:106775.

Zhang, M. and Wang, S. (2021). Matrix sketching for secure collaborative machine learning. *Proceedings of Machine Learning Research*, 139:12589 – 12599. Cited by: 5.

Zhang, R., Guo, S., Wang, J., Xie, X., and Tao, D. (2022a). A survey on gradient inversion: Attacks, defenses and future directions. *arXiv preprint arXiv:2206.07284*.

Zhang, T., Gao, L., He, C., Zhang, M., Krishnamachari, B., and Avestimehr, A. S. (2022b). Federated learning for the internet of things: Applications, challenges, and opportunities. *IEEE Internet of Things Magazine*, 5(1):24–29.

Zhao, B., Mopuri, K. R., and Bilen, H. (2020). idlg: Improved deep leakage from gradients. *arXiv preprint arXiv:2001.02610*.

Zhao, Y., Li, M., Lai, L., Suda, N., Civin, D., and Chandra, V. (2018). Federated learning with non-iid data. *arXiv preprint arXiv:1806.00582*.

Zhao, Z., Mao, Y., Liu, Y., Song, L., Ouyang, Y., Chen, X., and Ding, W. (2023). Towards efficient communications in federated learning: A contemporary survey. *Journal of the Franklin Institute*.

Zhu, J. and Blaschko, M. (2020). R-gap: Recursive gradient attack on privacy. *arXiv preprint arXiv:2010.07733*.

Zhu, L., Liu, Z., and Han, S. (2019). Deep leakage from gradients. *Advances in neural information processing systems*, 32.

Zimmerman, J. E., Kramer, A. A., McNair, D. S., and Malila, F. M. (2006). Acute physiology and chronic health evaluation (apache) iv: hospital mortality assessment for today's critically ill patients. *Critical care medicine*, 34(5):1297–1310.

Zuccala, A. A., Verleysen, F. T., Cornacchia, R., and Engels, T. C. (2015). Altmetrics for the humanities: Comparing goodreads reader ratings with citations to history books. *Aslib Journal of Information Management*.

Appendix A

Chapter 4: Additional Experiments

Similarity Guided vs. Random Clustering

This section presents the experiments conducted on additional datasets, similar to the results shown in Figure 4.8. The findings demonstrate that consistent with the FEMNIST dataset for similarity guided vs random clustering.

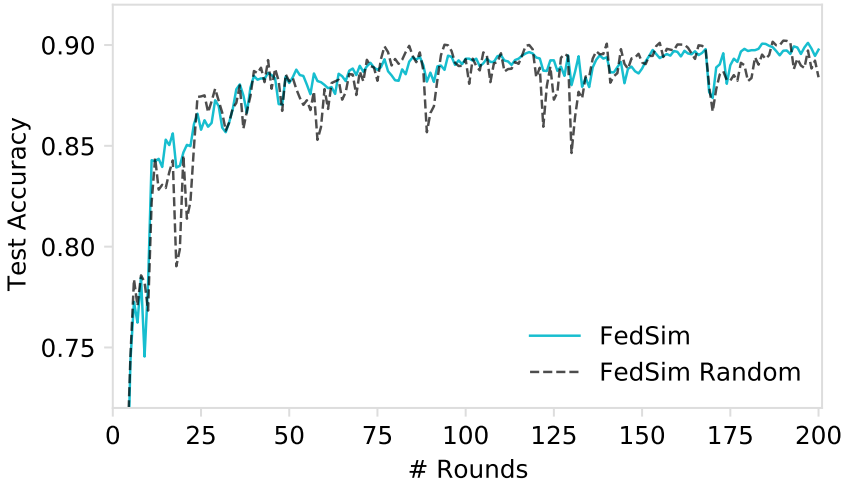

Figure A.1: Comparison of similarity guided clustering vs random clustering on MNIST

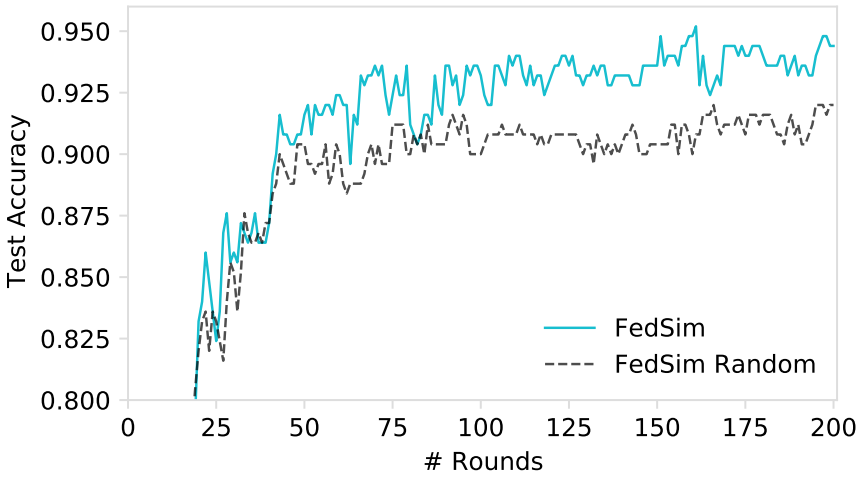

Figure A.2: Comparison of similarity guided clustering vs random clustering on Fed-MEx

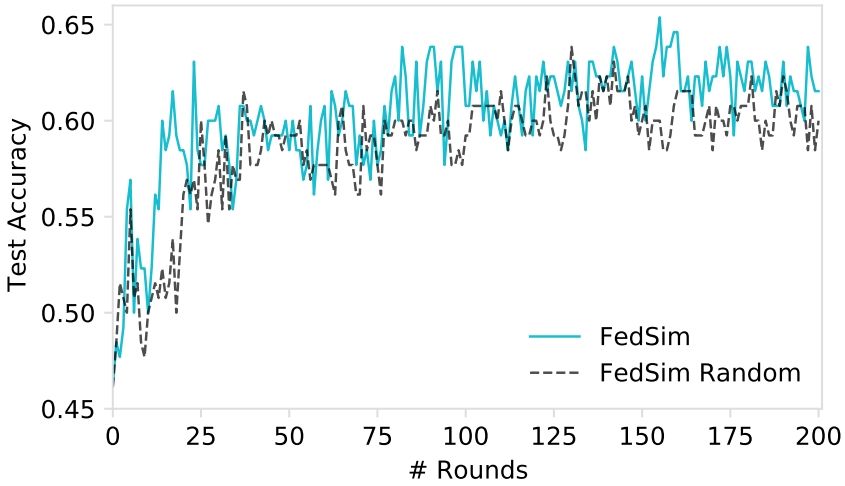

Figure A.3: Comparison of similarity guided clustering vs random clustering on Fed-Goodreads

Appendix B

Chapter 5: Additional Experiments

B.1 Comparison of Different DCT Variants (I to IV)

This section presents the experiments conducted on additional datasets, similar to the results shown in Figure 5.7. The findings demonstrate that consistent with the MNIST dataset, DCT-IV outperforms FFT in the other datasets.

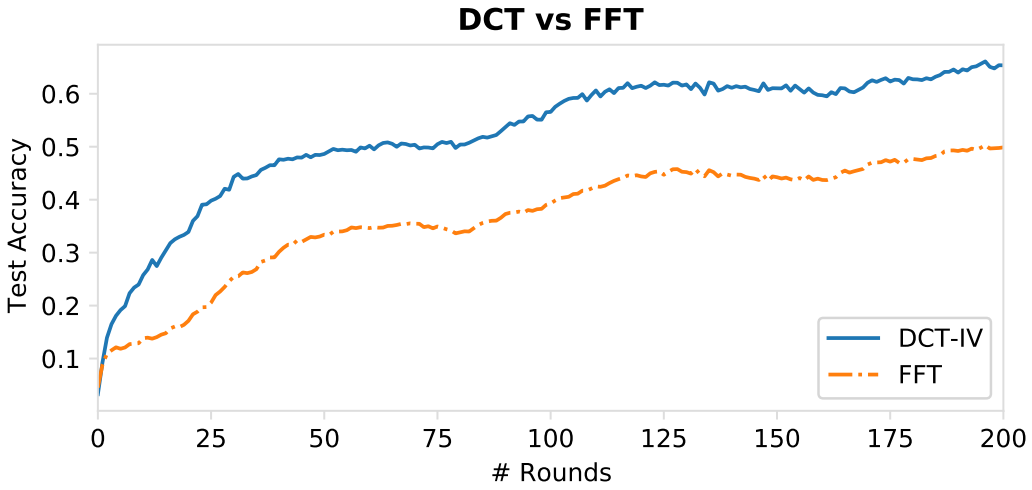

Figure B.1: Comparison of DCT and FFT on FEMNIST dataset

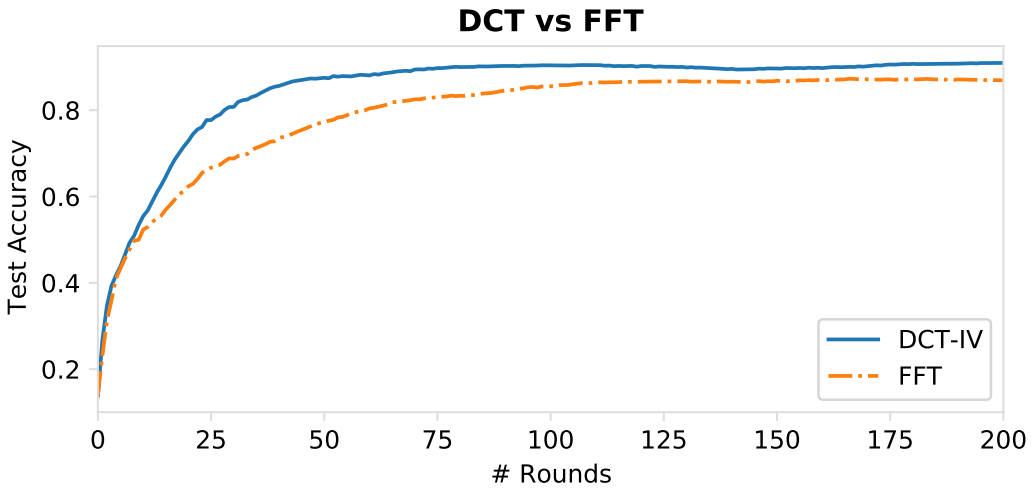

Figure B.2: Comparison of DCT and FFT on Fed-MEx dataset

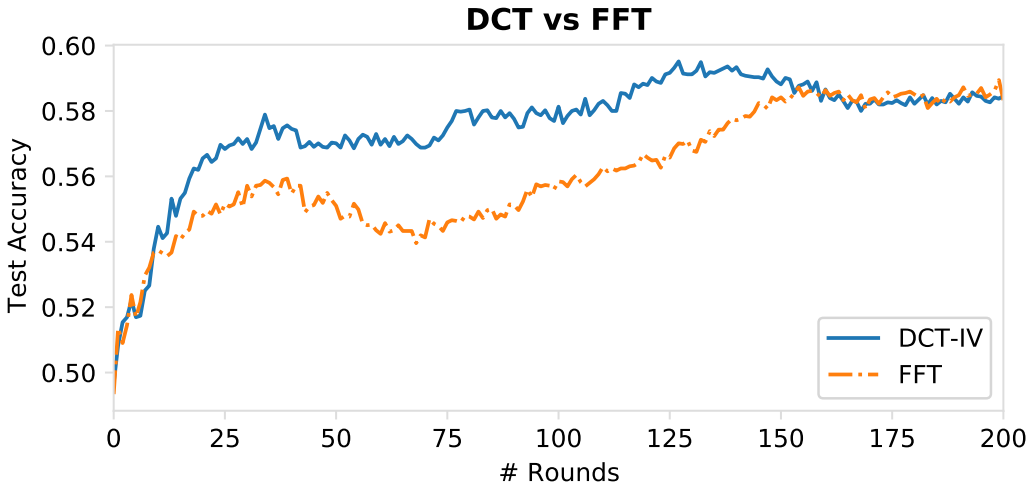

Figure B.3: Comparison of DCT and FFT on Fed-Goodreads dataset

B.2 Comparison of DCT Variants (I to IV) with *FedFT* and *FedAvg*

This section presents the experiments conducted on additional datasets, similar to the results shown in Figure 5.8. The findings indicate that the datasets exhibit similar behaviour, with DCT-IV being the most effective for *FedFT*.

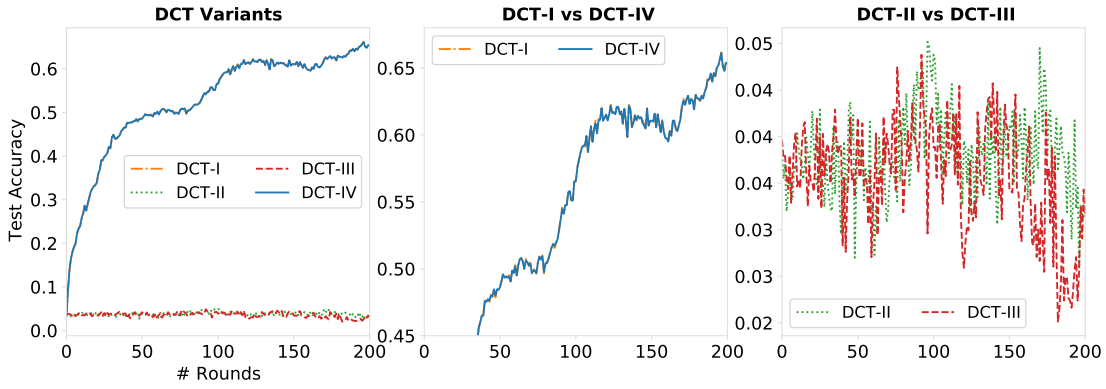

Figure B.4: Comparison of DCT variants (I to IV) on FEMNIST with *FedFT* with *FedAvg*

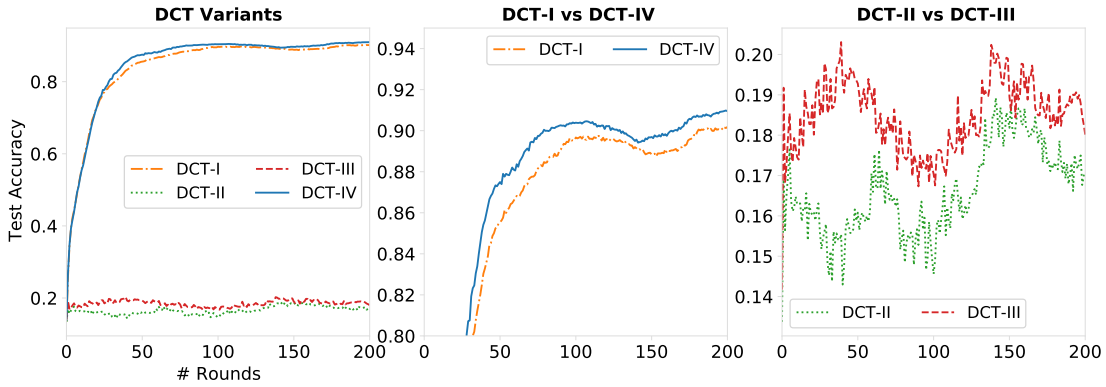

Figure B.5: Comparison of DCT variants (I to IV) on Fed-MEx with *FedFT* with *FedAvg*

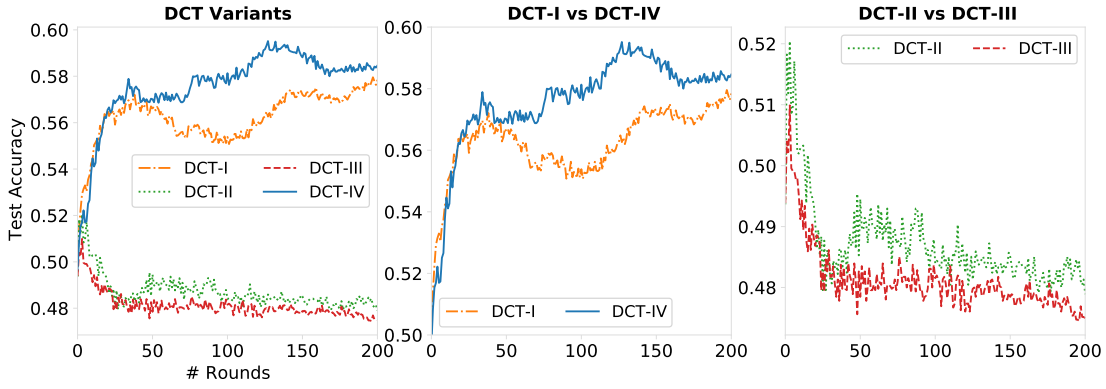

Figure B.6: Comparison of DCT variants (I to IV) on Fed-Goodreads with *FedFT* with *FedAvg*

Appendix C

Chapter 7: Additional Experiments

Figure C.1 presents summarized results for the eICU Database using *FedAvg*, *FedSim*, and *FedFT*. Additionally, it includes an experiment with a pruning percentage of $\alpha = 50\%$. This plot consolidates multiple experiments for comparative analysis.

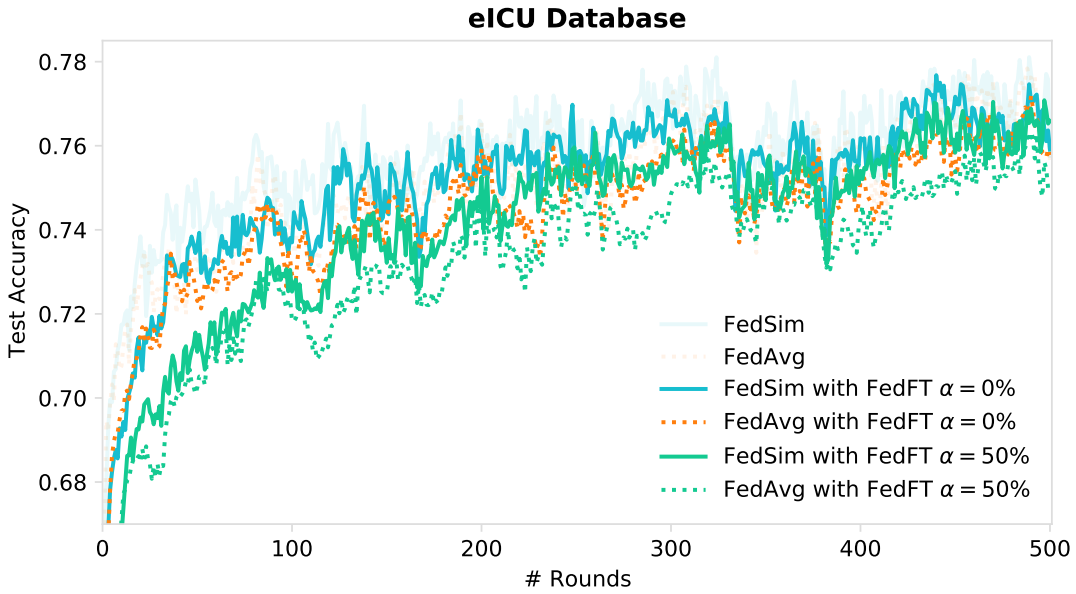

Figure C.1: Overall eICU database experiments including *FedSim* and *FedFT*.