Gradient Inversion Attacks in Federated Learning: Evaluating Privacy Risks and Differential Privacy Defenses in Cross-Silo Settings

University of Turku
Department of Computing
Master of Science (Tech) Thesis
Data Analytics
June 2025
Faiza Anan Noor

UNIVERSITY OF TURKU
Department of Computing

Faiza Anan Noor: Gradient Inversion Attacks in Federated Learning: Evaluating Privacy Risks and Differential Privacy Defenses in Cross-Silo Settings

Master of Science (Tech) Thesis, 74 p.
Data Analytics
June 2025

This thesis investigates the vulnerability of cross-silo Federated Learning (FL) systems to gradient-based privacy attacks, particularly focusing on the reconstruction of private image data from shared gradients. In cross-silo FL, a small number of relatively reliable and stable entities—such as hospitals, banks, or research institutions—collaborate to train a shared machine learning model without exchanging raw data. Each institution (or silo) computes local model updates based on its private dataset and periodically shares only gradients or model parameters with a central server, which aggregates them to improve the global model. While this setup is designed to protect data privacy, recent research has shown that gradients themselves can unintentionally leak information about the underlying data.

To study this risk, we implement a gradient inversion pipeline based on the Deep Leakage from Gradients (DLG) method, using a simplified linear reconstruction approach. Our aim is to determine whether private images can be recovered from aggregated gradients in a cross-silo setting. Uniquely, this work isolates Differential Privacy (DP) as the only defense mechanism under evaluation, allowing for a focused analysis of its protective effect. We assess the quality of reconstructed images using the Learned Perceptual Image Patch Similarity (LPIPS) score, which captures perceptual similarity as judged by deep neural networks.

To analyze the effectiveness of DP, we systematically vary the gradient clipping threshold and Gaussian noise multiplier applied during Differentially Private Stochastic Gradient Descent(DP-SGD), thereby controlling and reporting the privacy budget (epsilon) over multiple training rounds. The reconstructed images are evaluated using LPIPS, allowing us to quantify the extent of perceptual leakage under different privacy settings. In addition to evaluating leakage, we monitor classification accuracy across individual clients throughout the federated training and testing process with and without the presence of attacks, thereby highlighting the inherent trade-off between model utility and privacy preservation.

Our results reveal that effective visual degradation (i.e., high LPIPS) begins to occur only at extremely high noise levels, often resulting in ϵ values well below standard deployment thresholds. This suggests that although DP can mitigate perceptual reconstruction under aggressive noise conditions, achieving meaningful formal privacy guarantees remains difficult in practice without compromising model performance. By isolating DP as the defense mechanism and LPIPS as the evaluation tool, this study provides a focused empirical exploration of privacy–utility dynamics in FL under linear reconstruction attacks.

Keywords: Federated Learning, Differential Privacy, Secure Aggregation, Gradient Leakage, Deep Leakage from Gradients, Privacy Preservation, Model Performance, Federated Systems

Contents

List of Figures

List of Acronyms

CPU Central Processing Unit

DLG Deep Leakage from Gradients

DP Differential Privacy

DP-SGD Differentially Private Stochastic Gradient Descent

FC1 First Fully Connected Layer (Binning Layer)

FC2 Second Fully Connected Layer (Reconstruction Layer)

FL Federated Learning

FTL Federated Transfer Learning

GAN Generative Adversarial Network

GCN Gradient clipping threshold

GPU Graphics Processing Unit

HardTanh Hard Tangent Hyperbolic Activation

HE Homomorphic Encryption

HFL Horizontal Federated Learning

iDLG Improved Deep Leakage from Gradients

LDP Local Differential Privacy

LOKI Large-scale Optimization-based Knowledge Inference

MIA Membership Inference Attack

MPC Secure Multiparty Computation

NM Noise Multiplier

ReLU Rectified Linear Unit

SGD Stochastic Gradient Descent

SMC Secure Multiparty Computation

UI User Interface

VFL Vertical Federated Learning

List of Mathematical Notations

Symbol	Description
ϵ	Privacy budget in differential privacy; smaller values imply stronger privacy
δ	Probability of privacy guarantee failure in differential privacy
σ	Standard deviation of Gaussian noise added for differential privacy
C	Gradient clipping threshold
q	Sampling probability (i.e., client participation rate in each round)
T	Total number of training steps
LPIPS	Learned Perceptual Image Patch Similarity — a perceptual similarity metric
PSNR	Peak Signal-to-Noise Ratio — a reconstruction quality metric
SSIM	Structural Similarity Index Measure — a perceptual image quality metric

Table 1: List of notations used throughout the thesis

List of Tables

1 Introduction

FL [1] enables multiple clients such as mobile devices or medical institutions to collaboratively train a shared model without transmitting raw data to a central server. This decentralized framework is particularly attractive in sensitive domains such as healthcare and finance, where data confidentiality is critical. By ensuring that data remains local, FL provides an inherent layer of privacy while allowing collective intelligence. In cross-silo FL, it is typically assumed that the participating clients are a small number of stable, semi-trusted organizations (e.g., hospitals or banks) with reliable network connections and sufficient computational resources. These clients often hold non-overlapping but complementary datasets and are expected to be available throughout training.

However, FL is not immune to privacy risks. Recent studies show that gradient updates can leak sensitive information. Gradient-based reconstruction attacks—such as DLG [2] and its variants—can recover raw input data from shared gradients, especially those from shallow layers like imprint or fully connected layers. This raises serious concerns when the central server is untrusted or compromised.

To counter such threats, Differential Privacy (DP) [3] has emerged as a leading defense. In FL, DP is typically applied using DP-SGD [4], which involves clipping gradients and injecting calibrated Gaussian noise [3]. In cross-silo FL and general machine learning, Differential Privacy (DP) limits the risk of exposing individual data by adding noise to model updates. This ensures that the presence or absence of any single data point has minimal impact on the final model. DP is commonly implemented through gradient clipping and noise injection during training. While DP provides theoretical privacy guarantees, it often comes at the cost of degraded model performance, slower convergence,

and reduced representational power—especially in low-data or heterogeneous settings.

Despite its promise, the interplay between DP parameters [5] (clipping threshold, epsilon), model utility, and actual protection against gradient leakage remains underexplored. In particular, few studies have quantified how the level of Differential Privacy (DP) affects perceptual leakage metrics, which measure how visually similar reconstructed data is to the original. One such metric, Learned Perceptual Image Patch Similarity (LPIPS)[6], uses deep neural networks to evaluate image similarity in a way that aligns with human perception, providing a more realistic assessment of privacy leakage in image reconstruction attacks.

Our work highlights a critical concern in this context: if the central server coordinating the federated process is compromised or untrusted—as can often be the case with outsourced infrastructure or third-party coordination—gradient leakage attacks [7] can be launched to reconstruct sensitive medical data from client updates [8]. This risk is amplified by the fact that individual hospitals may not be aware of ongoing attacks or lack the technical capacity to implement rigorous defenses [9]. The imprint-layer-based architecture we investigate serves as a practical and interpretable attack surface, showing how even shallow gradients can be used to reconstruct detailed patient information [10]. In a gradient-based reconstruction attack, an adversary attempts to recover original client data by solving an optimization problem that finds inputs whose gradients match the shared gradients. This makes the attack not only feasible but also difficult to detect in distributed systems where each client has limited visibility into the global process.

Linear leakage refers to a simplified form of gradient-based attack where input data is reconstructed using a closed-form solution (i.e., an exact mathematical formula without iterative optimization), typically by leveraging linear layers in the model. While linear leakage or specific reconstruction attacks are intentionally designed and rely on architectural modifications such as the imprint layer, they highlight a critical insight: privacy leakage in FL is not merely a theoretical concern, but one that can manifest in practice—especially in scenarios where model components or training configurations have not been rigorously evaluated for privacy robustness. In domains like healthcare, where FL

is increasingly proposed for sensitive medical imaging and diagnostics, it is unrealistic to assume that practitioners or IT staff will rigorously inspect every architectural detail or detect subtle design choices that could facilitate leakage. Many real-world deployments rely on pretrained models or shared infrastructure where such components may be abstracted away or obscured within packaged code. Even if the presence of certain layers is known, the exact privacy implications—such as how a projection layer could enable closed-form input recovery may not be well understood [11]. A malicious or careless central coordinator could, for example, deploy a model with an imprint-like structure, enabling data reconstruction which is a huge threat to privacy in medical settings. This underscores the need for stronger default privacy mechanisms and architectural audits in safety-critical federated systems.

Our study offers a practical pathway toward mitigating such threats. By analyzing how DP mechanisms—specifically, gradient clipping and calibrated noise addition—impact reconstruction quality and model accuracy, we equip practitioners with actionable insights on how to configure FL systems under specific privacy budgets. Hospitals and other data custodians can use these findings to set safe default privacy configurations, even in settings where the central coordinator is semi-trusted or potentially adversarial. In this way, our work contributes not only to the academic understanding of privacy leakage but also to the real-world deployment of safer, more robust federated systems in critical domains like healthcare. Since FL remains an active and evolving area of research, especially in privacy-critical domains like healthcare, this thesis addresses a timely and novel challenge by combining gradient-based attacks with differential privacy defenses in a federated setting. The approach demonstrates how data reconstruction can occur in FL and how it can be effectively mitigated using differential privacy mechanisms, making it highly relevant in the current context of cybersecurity threats targeting distributed machine learning systems in hospital environments.

To address this, we systematically study the effect of gradient-based reconstruction attacks on FL models and evaluate DP-SGD as a mitigation strategy. We simulate attacks under varied privacy settings and analyze their impact on both client model training

accuracy and reconstruction quality. We specifically examine how a malicious server can degrade client model learning by inducing gradient-based linear reconstruction attacks through targeted layer manipulations. The topic is socially, scientifically, and practically significant, as it directly addresses the protection of sensitive patient data in real-world healthcare systems using rigorous, scientifically grounded privacy techniques. Few existing works systematically analyze both attack and defense in such settings, making this an ongoing and underexplored research question in the field. Our key contributions are:

- We investigate the vulnerability of FL to gradient inversion attacks, leveraging imprint layers to improve interpretability of the leakage.
- We implement DP-SGD as a defense, systematically varying DP parameters, Clipping Thresholds C and Epsilon ϵ, and evaluate their impact on both model accuracy and privacy leakage.
- We use LPIPS to quantify reconstruction quality and provide a detailed characterization of the privacy–utility trade-off.
- We analyze how varying data volumes influence the privacy–utility trade-off in FL with DP applied, in both the presence and absence of attacks.
- We evaluate the baseline performance degradation introduced by DP-SGD, independent of attacks, to understand its direct impact on learning dynamics.
- We present a practical analysis demonstrating that, even after successful reconstruction, applying DP-SGD with varied C and privacy budgets(ϵ) can substantially mitigate leakage, offering actionable guidance for deploying privacy-preserving FL in sensitive domains such as healthcare.

This study aims to guide the deployment of safer, more robust federated systems in sensitive applications like healthcare, by providing practical insights on configuring privacy protections under the presence of adversaries.

1.1 Research Questions

This study investigates the privacy vulnerabilities of FL systems under gradient-based reconstruction attacks, and evaluates the effectiveness of DP as a defense mechanism. The following research questions guide our analysis:

1. **Can a malicious server reconstruct private client data from gradient updates in an FL setup using gradient inversion attacks (e.g., DLG with imprint layers)?**

   This question explores how vulnerable client data is in standard FL when no additional privacy protection is used. To test this, we apply a simplified version of the DLG (Deep Leakage from Gradients) attack by modifying the model architecture to include an imprint layer—a special linear layer designed to isolate gradient information for easier reconstruction of input images.

2. **How does the integration of DP affect the success of gradient-based reconstruction attacks?**

   We aim to understand whether applying DP mechanisms can mitigate such attacks and protect client-level data from being reconstructed.

3. **What is the impact of DP parameters—specifically the privacy budget ϵ and gradient clipping threshold—on both model accuracy and attack robustness?**

   This question focuses on quantifying the trade-offs between privacy and utility by systematically varying ϵ and clipping threshold values. We analyze their effects on model performance not only in the presence of gradient inversion attacks but also under standard training without attacks, allowing us to isolate the direct impact of DP on learning dynamics as well as its defensive capability against reconstruction.

4. **To what extent do privacy-preserving operations (e.g., noise addition and gradient clipping) affect the learning dynamics and final performance of the global model?**

We assess how the application of DP alone, even without considering an attack scenario, impacts the overall model convergence, stability, and accuracy.

5. **How do FL Learning and DP interact, even in the absence of adversaries, in shaping the learning outcomes and privacy guarantees?**
   This broader question examines the fundamental trade-offs introduced by the combination of FL and DP, serving as a baseline to distinguish the additional effects caused by reconstruction attacks.

Research Objective

The primary aim is not to improve the global model's accuracy, but rather to investigate whether—and to what extent—a malicious server can reconstruct client data in an FL setup using such an attack. The focus is on evaluating the privacy implications of this threat, analyzing its impact on model performance, and exploring whether integrating differential privacy (DP) mechanisms can effectively mitigate the attack and protect client data. Additionally, we aim to understand how the server-side attack influences client-side learning by observing how client training and testing accuracy varies under attack conditions, with and without DP applied. Beyond attack scenarios, we also study the privacy–utility trade-off introduced by DP-SGD in FL when no attack is present, to isolate and understand the direct effect of DP on model performance in federated settings.

1.2 Thesis Structure

This thesis is structured into eleven chapters. **Chapter 1: Introduction** outlines the motivation, research questions, and structure of the thesis. **Chapter 2: Literature Review** presents prior work on FL, DP, and adversarial attacks in FL systems. **Chapter 3: Preliminaries** introduces the foundational concepts of FL and DP, including why privacy guarantees are critical in collaborative learning settings. **Chapter 4: Threat Model** defines the attacker assumptions and details the imprint-layer-based reconstruc-

tion pipeline, including CNN manipulation, closed-form inversion, and leakage quantification.

Chapter 5: Methodology describes the experimental setup, including datasets, model design, DP-SGD implementation, attack integration, and how privacy budgets are computed and evaluated. **Chapter 6: Results and Analysis** presents client both training and test performance and reconstruction leakage across varying privacy parameters and data volumes. **Chapter 7: Discussion** interprets key findings on the impact of architectural choices, clipping thresholds, and noise on utility and privacy. **Chapter 8: Conclusion** summarizes core contributions. **Chapter 9: Challenges and Limitations** discusses practical constraints. **Chapter 10: Future Direction** proposes further research on scalability, utility enhancement, and stronger defenses. Finally, **Chapter 11: Supplementary Materials** presents extended visualizations and leakage results over additional datasets including BreastMNIST, DermaMNIST, RetinaMNIST, and CIFAR10.

2 Preliminaries

This section introduces the foundational concepts of FL, DP, and their intersection, along with an overview of privacy vulnerabilities arising from gradient and update sharing.

2.1 Federated Learning

FL is a decentralized machine learning paradigm that allows multiple clients (e.g., mobile devices, IoT devices, or organizations) to collaboratively train a shared model without exchanging raw data. This approach enhances privacy, reduces communication costs, and enables learning from heterogeneous data sources.

2.2 Federated Learning Paradigms

Based on the type and reliability of participating clients, FL can be broadly categorized into two paradigms: **cross-silo FL** and **cross-device FL**. Figure 2.1 provides a conceptual overview of a typical FL workflow in the cross-silo setting.

2.2.1 Cross-Silo Federated Learning

Cross-silo FL typically involves a small number of stable, semi-trusted organizations—such as hospitals, banks, or research labs—that collaborate over time to train a shared model. This setting is characterized by:

- A fixed client population with consistent participation throughout training,
- Reliable network connectivity and synchronized communication,

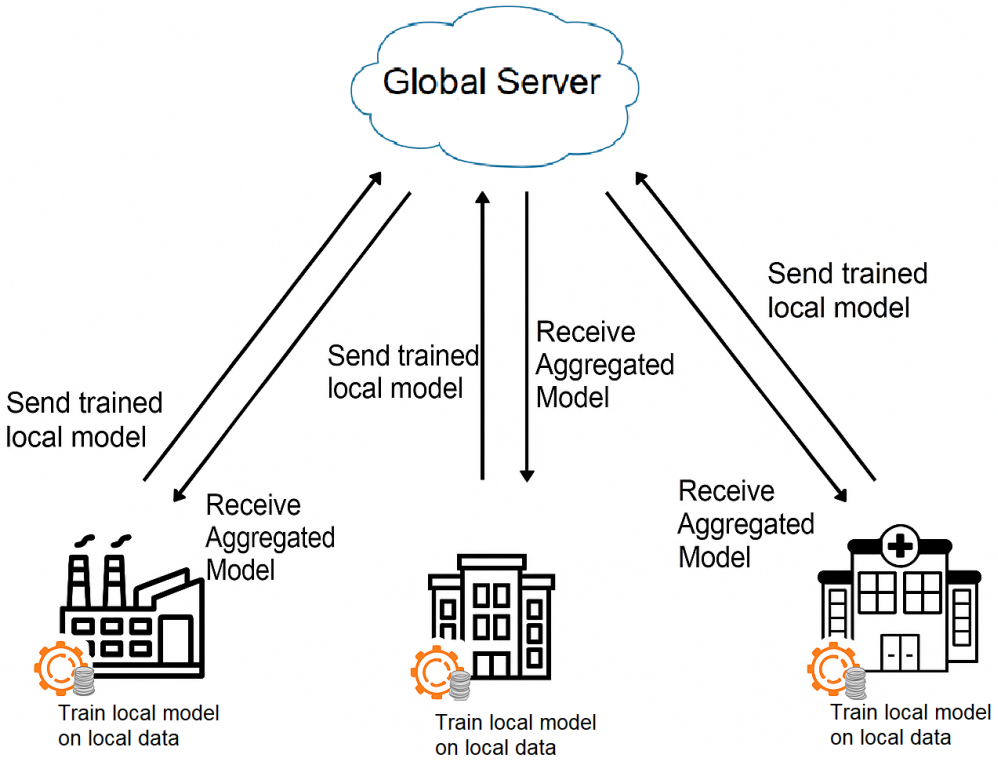


Figure 2.1: Illustration of a typical Federated Learning workflow in cross-silo settings.

- Access to high-quality, structured, and often sensitive data,
- Sufficient computational infrastructure on the client side.

2.2.2 Cross-Device Federated Learning

Cross-device FL, on the other hand, involves a large, diverse set of edge devices such as smartphones or IoT sensors. These devices are typically less reliable and highly variable in terms of participation. This setting is characterized by:

- A massive number of clients with intermittent availability,
- Unstable network conditions and asynchronous updates,
- Limited computational and battery resources,
- Non-IID (non-independent and identically distributed) and noisy data across devices.

FL follows an iterative training process consisting of the following steps:

1. **Client Selection:** A subset of clients is selected in each training round based on availability, network connectivity, and computational capacity.
2. **Local Model Training:** Each selected client trains a local model on its private dataset using optimization algorithms such as Stochastic Gradient Descent (SGD).
3. **Local Model Upload:** Instead of transmitting raw data, clients send their locally updated model parameters (e.g., gradients or weights) to a central server.
4. **Aggregation of Local Models:** The server aggregates the received model updates using an aggregation strategy such as Federated Averaging (FedAvg) [12].
5. **Global Model Update:** The aggregated model is updated and sent back to the clients.
6. **Iterative Training:** The above steps are repeated for multiple rounds until the model converges.

2.3 Differential Privacy

DP is a rigorous mathematical framework that ensures privacy preservation by limiting the influence of any single data point on the output of a computation. It provides formal guarantees against privacy leakage, even if an adversary has auxiliary knowledge.

A mechanism [5] M satisfies (ϵ, δ)-differential privacy if, for all datasets D_1 and D_2 differing by at most one element, and for all measurable subsets $S \subseteq \text{Range}(M)$, the following holds [3]:

$$P[M(D_1) \in S] \leq e^{\epsilon} P[M(D_2) \in S] + \delta. \tag{2.1}$$

Here:

- ϵ (privacy budget) controls the privacy-utility trade-off.
- δ is a small probability of violating privacy guarantees.

2.3.1 Importance of DP in FL

Although FL prevents direct data sharing, model updates still reveal private information through membership inference and model inversion attacks [13]. DP enhances FL privacy by introducing controlled noise and limiting the sensitivity of model updates.

- **Adding Noise to Model Updates:** Clients perturb their gradients before sending them to the server. For example, DP-SGD modifies gradients as follows:

$$\tilde{g} = g + \mathcal{N}(0, \sigma^2), \tag{2.2}$$

  where g is the computed gradient, and $\mathcal{N}(0, \sigma^2)$ represents Gaussian noise.

- **Gradient Clipping:** To limit individual influence, the gradient thresholds are clipped to a predefined threshold.

  In the original DP-SGD work [5], the amount of noise added is:

$$\textbf{Noise} \sim \mathcal{N}\big(0, \sigma^2 C^2 I\big) \tag{2.3}$$

  where:

  - σ = Noise Multiplier (sometimes called noise scale)
  - C = clipping threshold (L2 threshold bound)
  - I = Identity matrix (independent noise for each dimension)

  In DP-SGD, the privacy budget ϵ depends on several key factors: the noise multiplier σ, the sampling ratio $q = \frac{B}{N}$ (where B is the batch size and N is the total dataset size), the total number of training steps T, and the privacy failure probability δ. An approximate relationship under the subsampled Gaussian mechanism is:

$$\epsilon \approx \frac{\sqrt{2T \cdot \log(1/\delta)} \cdot q}{\sigma} \tag{2.4}$$

Here, T is the number of training steps, δ is the target failure probability, q is the sampling ratio, and σ is the noise multiplier that controls the scale of Gaussian noise added to the gradients.

As evident from this expression, increasing the noise multiplier σ or reducing the sampling ratio q results in a smaller privacy budget ϵ, thereby providing stronger differential privacy guarantees. While this formula offers an intuitive understanding, exact ϵ values are typically computed using advanced accounting techniques such as the Moments Accountant [5].

Gaussian noise, proportional to the clipping threshold, is added to the aggregated clipped gradients to sufficiently obscure the impact of any single data point [5], [14]. Thus, a larger clipping threshold results in a larger total noise — even if the noise multiplier is small. The analysis of noise scaling in DP-SGD reveals that the total noise added to the gradients is jointly determined by both the clipping threshold C and the noise multiplier σ. This dynamic impacts privacy guarantees: a smaller clipping threshold naturally reduces the influence of individual data points and thus enhances privacy, but it also requires careful tuning of the noise multiplier to maintain adequate noise levels. Conversely, a larger clipping threshold leaves gradients more intact but demands proportionally larger noise to protect privacy. Therefore, achieving strong DP requires balancing both the clipping threshold and the noise multiplier to ensure sufficient noise masking while preserving model utility.

- **Utility-Privacy Trade-off:** Stronger DP guarantees (lower ϵ values) provide better privacy but may reduce model accuracy. Adaptive noise calibration techniques can mitigate this trade-off as these techniques adjust the amount of noise added during training based on factors like gradient thresholds or training progress. This helps balance privacy protection with model performance more effectively than fixed-noise methods. We analyze the privacy-utility tradeoff for the MNIST dataset in both attack and non-attack scenarios across 5 clients in the federation, as presented in Table 6.2 in Section 6.

3 Literature Review

3.1 Federated Learning

FL has attracted substantial attention as a decentralized machine learning framework that enables collaborative model training without exposing raw user data. A growing body of research has explored the design principles, challenges, and diverse applications of FL.

Li et al. [15] systematically categorized the key challenges in FL into four areas: communication overhead, system heterogeneity, statistical heterogeneity, and privacy/security. To reduce communication costs, techniques such as local update strategies [16], model compression [17], and decentralized training protocols [18] have been proposed. Privacy-enhancing mechanisms have also been developed, including Secure Multiparty Computation (SMC) [19] and DP [20].

In terms of architecture, FL is commonly deployed in a client-server setup, where a central aggregator collects and updates the global model based on client contributions [21]. Alternative peer-to-peer architectures have been explored to increase robustness and eliminate reliance on a centralized coordinator [22]. Depending on data partitioning, FL is typically classified into Horizontal FL (HFL), Vertical FL (VFL), and Federated Transfer Learning (FTL) [23]. HFL involves clients with overlapping feature spaces but different samples, VFL applies to cases where clients share users but differ in features, and FTL is suited to scenarios where both features and samples are non-overlapping [24].

Various model aggregation strategies have been developed to improve FL's performance. The widely adopted FedAvg algorithm [16] suffers under non-IID data distri-

butions. To address this, algorithms such as Scaffold [25] introduce control variates to correct for client drift, while adaptive optimization techniques [26] dynamically adjust server-side learning rates to improve convergence.

Privacy and security remain ongoing concerns in FL deployments. Secure aggregation protocols [19] protect client updates from server inspection, and DP-based techniques [20] inject calibrated noise to mitigate leakage risks. Homomorphic Encryption (HE) has also been explored to enable encrypted computation on model parameters [27].

FL has found practical applications in numerous domains. In healthcare, it supports privacy-preserving collaborations among hospitals for predictive modeling using electronic health records [28]. In consumer settings, FL powers mobile keyboard prediction [29], personalized recommendation systems [30], and coordination in autonomous vehicles [31].

Despite substantial progress, challenges persist in ensuring communication efficiency, handling client heterogeneity, and achieving effective personalization. Future research should aim to design scalable communication protocols, enhance robustness under non-IID data, and develop adaptive personalization frameworks to meet user-specific needs.

3.2 Adversarial Threats in Federated Learning

Despite its decentralized nature, FL is vulnerable to privacy and security threats through gradients or model updates shared between clients and the server. As mentioned in Table 3.1, common attack types in FL include:

- **Model Inversion Attacks:** These attacks aim to reconstruct input data by inverting outputs or gradients of a shared model, often leveraging access to confidence scores or intermediate representations to approximate training samples [32], [2].

- **Membership Inference Attacks (MIA):** Conducted by a curious server or malicious client, MIAs aim to determine whether a specific data sample was part of the training set. They analyze model outputs or gradient patterns and can occur during model querying or after observing model updates [13], [33].

Attack Type	Adversary Role	Attack Mechanism	When It Occurs	Works
Model Inversion (Gradient Leakage)	Malicious Client or Server	Reconstructs training data from shared gradients during FL rounds, exploiting early-layer representations.	During Gradient Exchange (Training Phase)	[32], [2]
Membership Inference Attack (MIA)	Curious Server or Malicious Client	Determines if a specific sample was in training set by analyzing model outputs or gradients.	During Model Querying or Update Analysis	[13], [33]
Model Poisoning / Byzantine Attack	Malicious Clients	Sends adversarial or crafted updates to either disrupt training (e.g., random/erroneous gradients) or bias the global model (e.g., backdoors).	During Aggregation (Server-Side) or Local Training	[11], [34], [35], [36], [37], [38]
Model Extraction Attack	External Adversary (Black-box)	Uses repeated queries to copy the behavior or parameters of a deployed model.	After Deployment (Inference Phase)	[39], [40]
Evasion Attack	Malicious Client or External Adversary	Creates adversarial examples that fool the model at inference without altering training.	After Deployment (Inference Phase)	[41], [42], [43]

Table 3.1: Common FL attack types, attack mechanisms, and associated works

- **Model Poisoning / Byzantine Attacks:** Malicious clients submit crafted or erroneous updates to corrupt or bias the global model. These attacks may introduce backdoors, degrade performance, or destabilize training. They occur either during local training or server-side aggregation [11], [34]–[38].

- **Model Extraction Attacks:** External adversaries, without internal access, can perform black-box attacks by repeatedly querying the model after deployment. The goal is to replicate the model's parameters or behavior [39], [40].

- **Evasion Attacks:** These attacks are performed either by external adversaries or malicious clients who craft adversarial inputs that cause incorrect predictions at inference time, without modifying the training process [41]–[43].

In this work, we focus on model inversion (gradient leakage) attacks, where an adversary attempts to reconstruct client training data from shared gradients. Our evaluation specifically targets this vulnerability under different conditions, emphasizing how leakage severity varies with privacy mechanisms and training configurations. Among inversion techniques, we investigate linear leakage, a simplified attack strategy that reconstructs input data using a closed-form solution (i.e., a direct mathematical computation with-

out iterative optimization), typically enabled by inserting a linear imprint layer in the model architecture. This approach highlights how even minimal gradient information, if exploited effectively, can lead to significant privacy breaches in FL systems.

3.3 FL with DP

DP is a widely used mechanism in FL to ensure privacy by adding calibrated noise to gradients or model updates before they are shared. Geyer et al. [20] introduced DP-FedAvg, which adds Gaussian noise to aggregated updates, demonstrating strong privacy guarantees but highlighting the trade-off between noise magnitude and model accuracy. Recent advancements have explored adaptive noise mechanisms to balance privacy and utility. Despite its effectiveness, DP often degrades model performance when privacy budgets are tight, necessitating complementary techniques to maintain accuracy.

Recent research in privacy-preserving FL has proposed various approaches leveraging DP to protect sensitive user data during decentralized training. Truex et al. [44] introduced LDP-Fed, which applies local DP (LDP) for high-dimensional parameter updates using selective perturbation and filtering. Triastcyn and Faltings [45] propose a Bayesian DP approach offering tighter privacy bounds and improved model utility in FL. Choudhury et al. [46] present a FL framework with two levels of privacy for healthcare data, maintaining performance on large-scale medical datasets.

Naseri et al. [47] investigate the use of both local and central DP to defend against robustness and privacy attacks in FL, revealing nuanced trade-offs in protection and utility. Sun et al. [48] design a practical LDP mechanism that reduces variance and avoids high-dimensionality issues in model updates. Hu et al. [49] focus on personalized FL with DP, ensuring robustness to user heterogeneity and formal convergence guarantees. Finally, Geyer et al. [50] propose a client-level DP approach to obscure individual contributions in FL while preserving model performance with many clients.

These approaches can be broadly categorized based on where in the learning process DP is applied (e.g., on the data, during training, or to the model itself) and how DP

is implemented (e.g., locally at the client level, centrally at the server, or using more advanced variants like Rényi DP [51]). Table 3.2 categorizes existing DP approaches in FL based on the stage of application (data, training, or model level), while Table 3.3 classifies them according to the underlying DP setting (local, central, or hybrid/advanced).

Application Stage	**Focus Area**	**Representative Works**
Data-Level Privacy Protection	Noise is added to raw data before training to generate privacy-preserving datasets.	[52], [53], [54], [55]
Training-Time Privacy Mechanisms	DP is applied during model training, typically on gradients or model updates.	[56], [57], [58], [59], [60], [61]
Model-Level Privacy Preservation	Ensures the final trained model resists inference and reconstruction attacks.	[62], [63], [64], [65], [66]

Table 3.2: Differential Privacy Applications in Federated Learning

DP Setting	**Focus and Approach**	**Representative Works**
Local Differential Privacy (LDP)	Noise is added on-device before data/gradient sharing. No trust in central server. Common in mobile/private FL.	[56], [57], [58], [59], [60], [61]
Central Differential Privacy (CDP)	Noise is added at the server or trusted aggregator after collecting exact client data/gradients. Assumes trusted curator.	[52], [53], [62], [65], [64]
Hybrid / Advanced DP Variants	Uses advanced DP accounting or variants like Rényi DP (RDP), f-DP, or compositions to balance utility/privacy.	[66], [54], [63]

Table 3.3: Classification of DP Techniques in Federated Learning by Privacy Setting

We adopt techniques from both Training-Time Privacy Mechanisms and the LDP setting. Specifically, our approach adds calibrated noise to client-side model updates, balancing privacy and utility without relying on a trusted aggregator.

4 Threat model

In our work, we consider a **server-side linear leakage attack**, inspired by the work of LOKI [67], where an **imprint layer** is used to map the final layer activations to client-specific representations. By analyzing these activations—available from shared model updates—we estimate input reconstructions using a closed-form linear inversion, without iterative optimization or access to individual gradients. We primarily test whether DP can mitigate the gradient leakage attack introduced by LOKI [67], rather than proposing a new attack. While the approach for LOKI [67] claims to break privacy even under FedAvg and secure aggregation by avoiding the limitations of gradient mixing, our results show that such leakage is significantly reduced when DP is applied. Although the use of DP in FL is not new, we position it as a practical and essential extension for evaluating the real-world robustness of this attack. In future chapters, we will demonstrate that adjusting the privacy budget (ϵ) and C substantially degrades reconstruction quality, effectively mitigating the attack even under this strong threat model.

The attack used falls under the broader category of model inversion attacks, specifically focusing on a linear leakage setting. We exploit the imprint layer—a linear projection from deep representations to class prototypes—to reconstruct inputs from aggregated updates. Unlike optimization-based attacks like DLG or iDLG, our method uses a closed-form solution to recover input estimates, making it computationally efficient and practical for FL. This work investigates the effectiveness of such linear reconstruction attacks and their mitigation through DP. Figure 4.1 shows an overview of server-side privacy attacks in FL.

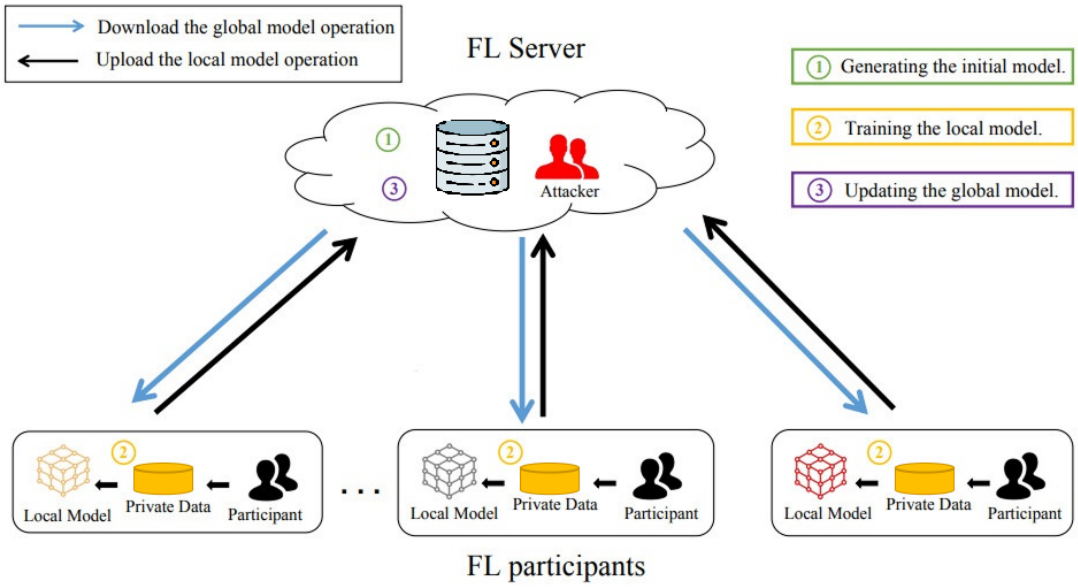


Figure 4.1: FL under Server-Side privacy attacks

4.1 Threat Model and Adversarial Assumptions

In this work, we assume a threat model in which the central server in an FL setup acts as the adversary. While the server orchestrates the FL protocol—coordinating model distribution and aggregation—it is also capable of modifying the model architecture before distributing it to clients. The server's objective is to reconstruct private client data by exploiting gradients observed from model updates, without requiring direct access to raw training data.

Attacker Role: The attacker is the central FL server, assumed to be semi-trusted or potentially adversarial. Before training begins, the server injects a malicious **imprint layer** into the client model. This layer is specifically designed to embed structured, interpretable activation patterns that leak information during backpropagation. Once inserted, the modified model is distributed to all participating clients for local training.

Capabilities: The server can freely alter model architectures, observe pre- and post-training parameters, and collect aggregated model updates after each training round.

Although the server does not interact with raw data, it can analyze these updates to extract signal patterns resulting from specific inputs processed during local training.

Limitations: The attacker does not have access to the private data stored on client devices. There is no visibility into internal computations, intermediate activations, or individual sample processing. The attack is limited to observing weight updates or gradients derived from completed training rounds.

Timing of the Attack: The attack unfolds during the local training phase on the client. As mini-batches are processed, the imprint layer silently produces sparse, structured activations. These activations influence the model's gradients in a way that is both predictable and recoverable. After training, the server receives the updated model and computes the difference from its pre-training version, revealing gradient information tied to specific input patterns.

Why the Attack Works: This attack is made effective by the *linear leakage* enabled by the imprint layer's architecture and initialization. Each client receives a slightly customized convolutional front-end, allowing the server to disentangle updates after aggregation. The use of bounded activations ensures that only a few units (or bins) activate per input, producing rank-1 gradient updates. These sparse, directional gradients leave a unique and traceable signature, enabling accurate reconstruction without requiring iterative optimization.

Simply put, this attack works because the model is intentionally designed to leave behind clues about the input data in its training updates. By using a special layer (the **imprint layer**) and carefully chosen settings, each input activates only a small part of the model called **bins**, which are fixed output slots that respond to different input patterns. Because only one or a few bins activate for each image, the resulting update is clean and easy to interpret. These patterns show up in the gradients shared with the server, which can then be traced back to reconstruct the original input using a direct mathematical method (**closed-form reconstruction**), meaning no guessing or

optimization is needed—just solving an equation based on known values. The rest of this section explains how each part of this design makes such leakage possible.

4.2 Attack Design: Model Manipulation and Closed-Form Reconstruction

Building on the threat model defined earlier, we now describe how the server-side attacker designs and executes the attack in practice. The core idea is to modify the client model architecture in a way that intentionally produces structured, interpretable gradients during local training. These gradients leak sufficient information for the attacker to reconstruct private training data from model updates, even when these updates are aggregated across clients.

Our approach combines architectural manipulation with deterministic initialization and sparse activation to enable efficient, closed-form gradient inversion. The attack pipeline consists of the following components: (1) manipulating the CNN architecture to simplify and linearize gradient flow; (2) introducing a custom imprint layer to project features into sparse, reconstructible representations; (3) customizing the model per client to allow post-aggregation disentanglement; and (4) reconstructing inputs algebraically based on sparse bin activations. We detail each of these components below.

4.2.1 Manipulating CNNs for Gradient-Based Leakage

Convolutional Neural Networks (CNNs) are widely used in image classification due to their ability to extract hierarchical feature representations from input images. In the context of FL, CNNs are trained locally on private data, and clients share only model updates (typically gradients or parameter differences) with a central server. While this is designed to preserve privacy, it does not prevent all forms of information leakage. In particular, gradients carry information about how the model output changes with respect to input features, and under certain architectural conditions, these gradients can be inverted to reveal input content.

In our work, we deliberately manipulate the CNN architecture to make this leakage interpretable and easily decodable. We simplify the model by using only a shallow convolutional layer followed by a fully connected projection layer. This reduces non-linearity and creates a more transparent path from input to gradient. By applying a bounded activation function such as HardTanh, we ensure that only a few output units (or "bins") activate for a given input. This sparsity produces clean and structured gradients that can be linked back to specific input characteristics.

This controlled design enables us to trace which parts of the input were responsible for activating specific units, making it feasible to reconstruct the original input from shared model updates. The next section introduces the imprint layer—a customized projection mechanism built on top of this architecture—which plays a central role in enabling efficient, closed-form input reconstruction from aggregated gradients.

4.2.2 Imprint Layer: Sparse, Reconstructible Feature Projection

The imprint layer is a linear projection that maps deep feature representations to a fixed number of class prototypes or "bins." In this work, it is implemented using a small convolutional layer followed by two fully connected layers, with a hardtanh activation in between. This design enables our attack to exploit linear relationships between input features and output gradients for reconstruction. In summary, the term "imprint layer" refers to this interpretable projection layer, inspired [68] by its role in storing input-specific activation bins that can be reconstructed from gradients. This term is crucial for designing the custom binning architecture designed for leakage analysis.

4.2.3 Closed-Form Linear Reconstruction & Bin Guided Reconstruction

Instead of using iterative optimization methods like DLG or iDLG [2], we recover input estimates directly from aggregated gradients using closed-form [69] linear algebra. This is enabled by the imprint layer's deterministic initialization and sparse activation behavior. Specifically, the first fully connected layer (FC1) projects flattened convolutional features

into a fixed number of "bins," where each bin corresponds to a linear combination of input features. The activation function (HardTanh) constrains bin outputs between 0 and 1.

During training, the model is constructed such that, for a given input, only one or a few bins activate (i.e., their pre-activation values fall within the (0,1) range). These "active bins" directly indicate which FC1 weight vectors contributed to the observed gradient update. Since FC1 weights are pre-initialized using quantiles of the training data distribution and scaled to produce interpretable gradients, the activation of a single bin results in a rank-1 gradient that can be algebraically inverted to recover the corresponding input. This sparsity in activation ensures that individual data points leave a distinct and decodable imprint in the gradient, enabling efficient and accurate reconstruction from aggregated updates.

When only one or a few bins activate for a given input, the corresponding gradient update becomes simple and well-structured. Each activated bin directly reflects how a specific portion of the input contributed to the model's output. Since the imprint model is carefully initialized and the weights are known, the gradient associated with an active bin effectively points in the direction of the original input.

Because only one bin is typically active, the resulting update is not mixed with other signals, making it easy to isolate and reshape into an image. This allows the attacker to reconstruct the original input accurately by directly interpreting the gradient, without needing iterative optimization. The approach is fast, effective, and works even when updates are aggregated across multiple clients.

4.2.4 Client-Specific Model Customization

To maintain attack viability under aggregation, each client receives a customized version of the imprint model. Specifically, the convolutional kernel is altered for each client to ensure that their gradient updates activate disjoint channels. This customization helps preserve client-specific leakage patterns even when updates are averaged during aggregation, enabling reconstruction of inputs on a per-client basis.

4.2.5 Gradient Aggregation and Attack Robustness

In typical FL, client updates are aggregated at the server—usually via simple averaging. While aggregation is often seen as a defense against leakage, our model customization and sparse bin activation mitigate this defense. Because each client activates unique convolutional channels and uses disjoint imprint bins, their individual contributions remain separable in the global update. Thus, even aggregated gradients can be deconstructed into identifiable components for reconstruction.

4.2.6 Leakage Quantification

To measure the fidelity of reconstructed images, we use LPIPS (Learned Perceptual Image Patch Similarity), a perceptual similarity metric that captures high-level visual similarity based on deep features. LPIPS is more aligned with human perception than traditional pixel-wise metrics like PSNR (Peak Signal-to-Noise Ratio) or SSIM (Structural Similarity Index Measure). In our evaluation, we report LPIPS scores under various DP settings to analyze how noise impacts the attack's effectiveness. The underlying principles and implementation details of LPIPS are discussed in Section 5.6, while experimental results across different ϵ values and clipping thresholds are presented in Section 6.3 for the MNIST and PathMNIST dataset. Additional results for other MedMNIST datasets are included in the Appendix.

5 Methodology

5.1 Experimental Setup, Dataset Selection and Configuration

We evaluate the vulnerability of FL systems to gradient leakage attacks using a variety of image classification datasets. Our main experiments focus on MNIST [70] and PathMNIST [71], while CIFAR-10 [72], RetinaMNIST [71], BreastMNIST [71], BloodMNIST [71], and PneumoniaMNIST [71] are included in the supplementary materials. Each dataset is preprocessed to ensure compatibility with our model architecture and training setup.

Data Sampling and Client Partitioning: A fixed number of samples is randomly selected from each dataset's training set and evenly distributed across five simulated clients. In attack-based experiments, each client receives exactly 64 images, yielding 320 training samples per run. While standard FL scales well with full datasets, such a configuration becomes computationally infeasible under reconstruction attacks, which require tracking and storing per-step gradients, activations, and model states. To keep the attack tractable on limited hardware, we restrict the number of samples per client, while still preserving the key features of decentralized learning.

Class Balance and Client Organization: To ensure that client datasets are representative of the overall label distribution, we sample from all available classes and enforce approximate class balance. Each client trains independently on its local subset without accessing other clients' data, preserving the decentralization property of FL. We also analyze per-image pixel intensity statistics to establish dataset-specific baselines. These

statistics are later used for initializing the activation binning mechanism within our model architecture.

Dataset Selection Rationale: The datasets chosen for our experiments are widely used benchmarks in the FL literature, representing both natural and medical imaging domains. MNIST and CIFAR-10 are canonical datasets in computer vision, frequently used in federated settings due to their simplicity and ubiquity in evaluating baseline methods. The MedMNIST suite, which includes PathMNIST, RetinaMNIST, BreastMNIST, BloodMNIST, and PneumoniaMNIST, is specifically curated for lightweight biomedical image classification tasks and provides a practical testbed for simulating privacy risks in healthcare applications. These datasets offer diverse image characteristics (e.g., grayscale vs. RGB, small vs. complex structures) and task types (e.g., digit recognition, tissue classification, disease detection), making them ideal for studying gradient leakage under varying data modalities and client configurations.

These datasets were chosen not only for their prominence in FL research but also for their suitability in simulating both cross-device and cross-silo federated settings as demonstrated by many works. While they are not inherently partitioned by institution or user, their structural and semantic diversity allows us to model decentralized learning environments where data is non-overlapping across clients. In particular, the medical datasets from MedMNIST reflect real-world applications where sensitive data is distributed across healthcare providers, even though we simulate this scenario by manually partitioning the data. Meanwhile, MNIST and CIFAR-10 provide controlled settings for analyzing the effects of reconstruction attacks and mitigation strategies under different levels of data complexity. This combination enables us to systematically study attack effectiveness, privacy-utility trade-offs, and the broader applicability of defense mechanisms in practical federated deployments.

5.2 Image Preprocessing

All images across datasets undergo a standardized preprocessing pipeline to ensure compatibility with the model architecture and to support consistent evaluation across modalities. The steps are as follows:

- **Resizing:** Images are resized to a fixed square resolution that matches the expected input size of the model (e.g., 28×28 for MNIST, 32×32 for CIFAR-10).
- **Center Cropping:** When needed, images are cropped at the center to maintain a uniform spatial dimension, preserving the most relevant features in the middle of the image.
- **Tensor Conversion:** Images are converted into PyTorch tensors, scaling pixel values into the range $[0, 1]$.
- **Normalization:** To improve activation balance and support symmetric learning behavior, all images are further scaled into the range $[-1, 1]$ using the transformation:

$$\text{image}_{\text{norm}} = \text{image} \times 2 - 1$$

- **Channel Handling:** For grayscale datasets like MNIST, the single channel is retained. For similarity-based metrics like LPIPS which require 3-channel inputs, grayscale images are duplicated across three channels during evaluation only.

This preprocessing ensures uniformity across datasets with varying characteristics and maintains consistency for both model input and visual quality assessments.

5.3 Model Architecture and Training Procedure

We employ a lightweight and interpretable neural network architecture, inspired by imprint layers, designed to enable gradient-based reconstruction analysis in a FL context. The model consists of three key components: a convolutional layer, a binning-based fully connected layer (FC1), and a final reconstruction layer (FC2). The initial convolutional

layer uses client-specific filters, scaled by a factor of 500 to control the signal magnitude. This is followed by a bounded Hardtanh activation in the range $[0, 1]$. The first fully connected layer projects the flattened feature map into a discretized bin space, and the second reconstructs the output back into a vector matching the original image dimensions. The number of bins per client is fixed at 256, corresponding to four times the local batch size. This compact architecture allows for precise control over gradient behavior and supports interpretability in privacy-focused evaluations.

Training Configuration: Each client trains locally for 15 communication rounds (epochs), with 10 local iterations per round. For the attack setup, each client is assigned only 64 images due to computational constraints during reconstruction. We use the SGD optimizer with a learning rate of 0.1. All training is performed using PyTorch [73], and models are run on GPU when available, defaulting to CPU otherwise. The architecture is kept shallow to balance computational feasibility and reconstruction observability. In one FL experiment without attack, we use the entire dataset per client to compare global model performance against the 64-sample setup.

DP Integration: To simulate privacy-preserving FL, we apply DP-SGD at each update step. Gradients are clipped using a configurable ℓ_2 threshold and perturbed with Gaussian noise, scaled by a user-defined noise multiplier. These hyperparameters are systematically varied to examine their effects on both privacy leakage and model utility.

Staged Experimental Design: We conduct a series of four core experiments to evaluate the interplay between model performance and privacy vulnerability:

1. **Standard FL (Full Data):** We first train the model using the full dataset per client, with no privacy constraints, to establish a performance baseline under ideal conditions.

2. **Standard FL (Small Batch):** Next, we limit each client to only 64 images and observe the resulting drop in accuracy due to reduced data diversity and underfitting.

3. **FL with DP and No Attack:** We also assess the predictive performance of the

model under DP without applying any attack, highlighting the utility loss caused purely by the privacy mechanism using varying data volumes.

4. **FL with Gradient Leakage Attack:** We then apply a gradient-based reconstruction attack in the small-batch setting, quantifying the extent of image leakage from client updates in a non-private setup.

5. **FL with DP and Attack:** Finally, we introduce DP into the training pipeline and repeat the attack to evaluate how clipping and noise injection impact both image reconstruction and training accuracy/learning of the clients.

Privacy-Utility Trade-off: To study the effect of varying privacy budgets, we systematically adjust the DP parameters, C and ϵ. This allows us to measure how changes in the effective privacy level (ϵ) influence both the quality of reconstructed images and model utility. These evaluations are performed across multiple datasets, including natural and medical image domains, to ensure generalizability of our findings.

5.4 Calculation of ϵ and DP-SGD Mechanism

To measure the formal privacy guarantee of the noise injection process during training, we compute the DP budget ϵ using a widely used analytical approximation proposed by [5]. This formulation estimates the cumulative privacy loss when training with mini-batch SGD combined with Gaussian noise — known as **DP-SGD**.

5.4.1 Epsilon Computation via Analytical Approximation

In our experiments, we estimate ϵ using the simplified analytical formula [5]:

$$\epsilon = \frac{\sqrt{2T \log(1/\delta)} \cdot q}{\sigma} \tag{5.1}$$

Here,

- T is the total number of training steps, computed as $T = \frac{\text{dataset size}}{\text{batch size}} \times \text{epochs}$,

- q is the sampling ratio per iteration, defined as $q = \frac{\text{batch size}}{\text{dataset size}}$,
- δ is a small failure probability, typically set as $\delta < \frac{1}{n}$, where n is the training set size.
- σ is the noise multiplier.

5.4.2 Illustrative Example

In our case, we train on a dataset of size $n = 320$ (5 clients $\times$ 64 images each) due to computational complexity of implementing the attack, with batch size $B = 64$, and epochs $E = 15$. In this way, the total number of steps becomes:

$$T = \left\lfloor \frac{n}{B} \right\rfloor \times E = \left\lfloor \frac{320}{64} \right\rfloor \times 15 = 5 \times 15 = 75$$

Sampling probability:

$$q = \frac{64}{320} = 0.2$$

Now, with a noise multiplier $\sigma = 10^{-9}$ and $\delta = 10^{-5}$, the privacy budget is:

$$\epsilon = \frac{\sqrt{2 \cdot 75 \cdot \log(1/10^{-5})} \cdot 0.2}{10^{-9}} \approx \frac{\sqrt{2 \cdot 75 \cdot 11.51} \cdot 0.2}{10^{-9}} \approx \frac{18.53 \cdot 0.2}{10^{-9}} \approx 8.31 \times 10^{9}$$

This very high ϵ value indicates that essentially no meaningful differential privacy guarantee is provided at such a low noise level. This aligns with the near-perfect image reconstructions observed in this regime.

5.4.3 DP-SGD Gradient Perturbation

During training, each client's gradient is first clipped to the fixed threshold C:

$$g \leftarrow g \cdot \min\left(1, \frac{C}{\|g\|_2}\right)$$

Then Gaussian noise is added:

$$g \leftarrow g + \mathcal{N}(0, \sigma^2 C^2)$$

To enforce differential privacy, we apply noise and gradient clipping locally at each client before model updates are shared. This ensures that all gradients are privatized prior to aggregation, aligning with the DP-SGD protocol.

5.4.4 Epsilon in Context

In our experiments, we deliberately explore a wide range of privacy budgets (ϵ) — spanning from strong to weak privacy settings — to identify at what point perceptual privacy begins to emerge. Our results show that LPIPS [6] scores remain low under weaker privacy, indicating that reconstructed images closely resemble the originals and substantial leakage is possible. As privacy strengthens (i.e., as ϵ decreases), LPIPS scores increase, reflecting more distorted and less accurate reconstructions. This highlights that weak privacy guarantees can still permit effective image reconstruction, revealing a gap between formal privacy metrics and actual visual protection. We also observe corresponding fluctuations in model accuracy, where stronger privacy often reduces utility while weaker privacy preserves or even improves predictive performance. We further explore this privacy-utility tradeoff in Section 6.

5.5 Concurrent Attack Execution

The attack operates in tandem with the training process. By examining discrepancies between the original and updated model parameters, the attacker identifies which neurons are activated by specific inputs. These identified activations are subsequently exploited to reconstruct the input images. This methodology highlights the vulnerability of FL to privacy breaches, even when raw data remains unseen.

To simulate the attack, we store the model weights before and after training. The aggregated gradients for each client are inferred from the weight deltas. Using knowledge

of the network structure and the client's data allocation, we isolate the relevant subset of gradients and reconstruct input images using a linear inversion strategy. This is done by mapping the weights of the FC1 layer back into image space and interpreting them as image approximations.

5.6 Image Reconstruction and Evaluation

In this work, the primary perceptual metric used to assess reconstruction quality is the LPIPS [6] metric. LPIPS is a deep feature-based metric designed to measure perceptual similarity between image pairs in a manner that better aligns with human visual judgment than traditional metrics such as PSNR or SSIM. Unlike pixel-wise losses, LPIPS computes distances in the feature space of a pretrained convolutional neural network (e.g., AlexNet or VGG), where perceptually meaningful structures are more robustly captured. In our experiments, we use the VGG-based variant of LPIPS, which relies on fixed, pretrained weights learned from human-annotated image similarity judgments, as introduced by Zhang et al. [74]. This makes it particularly suited for evaluating how "recognizable" or visually faithful a reconstructed image is to its original, even in the presence of adversarial noise or blurring. LPIPS scores are collected for every image successfully reconstructed (i.e., those with unique activation patterns), and averaged to assess leakage per client and per configuration.

Formally, given two images x and x', the LPIPS score is defined as:

$$\text{LPIPS}(x, x') = \sum_{l} w_l \cdot \|f_l(x) - f_l(x')\|_2^2 \tag{5.2}$$

where $f_l(x)$ represents the activation of image x at the l-th layer of a fixed feature extractor (typically a pretrained deep network), and w_l are scalar weights learned to align the feature differences with human perceptual preferences. The LPIPS metric therefore captures semantic-level distortions rather than raw pixel-wise deviations. A lower LPIPS score indicates that the reconstructed image is more perceptually similar to the original, and conversely, a higher score reflects stronger distortions or privacy-preserving degrada-

tion.

In the context of privacy-preserving FL, LPIPS provides an effective quantitative signal for visual leakage. When attackers attempt gradient inversion or reconstruction attacks, lower LPIPS scores imply that the attack was successful in recovering perceptually accurate images from shared model updates, posing a privacy risk. Conversely, high LPIPS values suggest that privacy mechanisms (such as DP noise or aggressive gradient clipping) have successfully disrupted reconstructive fidelity. In this study, we leverage LPIPS to evaluate and compare privacy leakage across datasets, ϵ and C, revealing how design choices in privacy budgets directly impact perceptual exposure and client accuracies.

While LPIPS is used to capture perceptual similarity, we also concurrently analyze client-level classification accuracy on both original and reconstructed images. This provides an additional axis of evaluation for privacy-utility trade-off: for instance, a low LPIPS score might indicate strong visual leakage, but if reconstructed images still yield poor classification accuracy, the practical risk to user-level inference may be mitigated. Together, LPIPS and classification accuracy form a comprehensive framework for measuring both visual fidelity and functional leakage in federated systems under adversarial reconstruction threats.

Both Algorithm 1 and Algorithm 2 begin with a standard FL setup, where the global dataset $\mathcal{D}$ is partitioned among N clients. Each client independently trains a shallow, interpretable neural network composed of a convolutional layer, a binning-based fully connected layer (FC1), and a final FC2 layer responsible for reconstructing pixel-level image data. This architecture—referred to as the **imprint layer**—is specifically designed to facilitate reconstruction-based leakage analysis by retaining informative gradients. Clients perform a fixed number of local iterations using mini-batches, compute gradients based on cross-entropy loss, and update model parameters using the SGD optimizer. The model weights are stored both before and after training to support reconstruction analysis.

Algorithm 1 outlines the training and attack process in a non-private setting. After training, each client's model weight delta ΔW_i—defined as the change in parameters

Algorithm 1 Federated Training with Gradient-Based Reconstruction (No DP)

Input: Dataset $\mathcal{D}$, number of clients N, batch size B, epochs E, steps T
Output: Reconstructed images $\hat{x}$, LPIPS scores, accuracy logs

1. Split dataset $\mathcal{D}$ into $\mathcal{D}_1, \ldots, \mathcal{D}_N$
2. Initialize client models $f_1, \ldots, f_N$ with imprint structure
3. **for** each client $i = 1$ to N:
  for each epoch $e = 1$ to E:
    Store model weights W_i^{before}
    for each local step $t = 1$ to T:
      Sample batch x from $\mathcal{D}_i$
      Compute gradients $\nabla\mathcal{L}$
      Update weights via optimizer
    Store model weights W_i^{after}
  Send updated model W_i^{after} to the server
4. Server aggregates all client updates into a global model
5. Server sends the updated global model back to all clients
6. **for** each client i:
  Compute weight delta $\Delta W_i = (W_i^{\text{after}} - W_i^{\text{before}})$
  for each bin b:
    If bin b is uniquely activated:
      Reconstruct image $\hat{x}_b$ from $\Delta W_i[b]$
      Evaluate LPIPS$(\hat{x}_b, x_b)$
  Log client accuracy and LPIPS

Algorithm 2 Federated Training with DP and Gradient-Based Reconstruction

Input: Dataset $\mathcal{D}$, number of clients N, batch size B, epochs E, steps T, noise multiplier σ, clipping threshold C
Output: Reconstructed images $\hat{x}$, LPIPS scores, accuracy logs

1. Split dataset $\mathcal{D}$ into $\mathcal{D}_1, \ldots, \mathcal{D}_N$
2. Initialize client models $f_1, \ldots, f_N$ with imprint structure
3. **for** each client $i = 1$ to N:
  for each epoch $e = 1$ to E:
    Store model weights W_i^{before}
    for each local step $t = 1$ to T:
      Sample batch x from $\mathcal{D}_i$
      Compute gradients $\nabla\mathcal{L}$
      Clip: $\nabla \leftarrow \nabla \cdot \min(1, C/\|\nabla\|)$
      Add noise: $\nabla \leftarrow \nabla + \mathcal{N}(0, \sigma^2 C^2)$
      Update weights via optimizer
    Store model weights W_i^{after}
  Send updated model W_i^{after} to the server
4. Server aggregates all client updates into a global model
5. Server sends the updated global model back to all clients
6. **for** each client i:
  Compute weight delta $\Delta W_i = (W_i^{\text{after}} - W_i^{\text{before}})$
  for each bin b:
    If bin b is uniquely activated:
      Reconstruct image $\hat{x}_b$ from $\Delta W_i[b]$
      Evaluate LPIPS$(\hat{x}_b, x_b)$
  Log client accuracy and LPIPS

before and after training—is computed. This delta is analyzed to extract information from uniquely activated bins in the FC1 layer, which correlate with input-specific activations. For each such bin, a corresponding image $\hat{x}_b$ is reconstructed from the relevant portion of the weight delta. The perceptual similarity between reconstructed and original inputs is evaluated using LPIPS, and model accuracy is logged. This procedure quantifies the extent of visual leakage possible in a FL scenario without any privacy-preserving defense.

In contrast, Algorithm 2 extends the training loop by incorporating DP through DP-SGD. At each local update step, the per-sample gradients are clipped using a configurable L2 clipping threshold bound C and perturbed with Gaussian noise scaled by a noise multiplier σ. These operations are designed to limit the influence of individual data samples on the model update, thereby reducing the risk of leakage. The attack procedure is repeated identically, using the post-training weight deltas from differentially private models. By comparing reconstruction quality and LPIPS scores between Algorithm 1 and Algorithm 2, we evaluate the effectiveness of DP as a defense mechanism. Additionally, by systematically varying the clipping threshold bound and noise multiplier (and thus the privacy budget ϵ), we explore the trade-offs between privacy and utility across datasets.

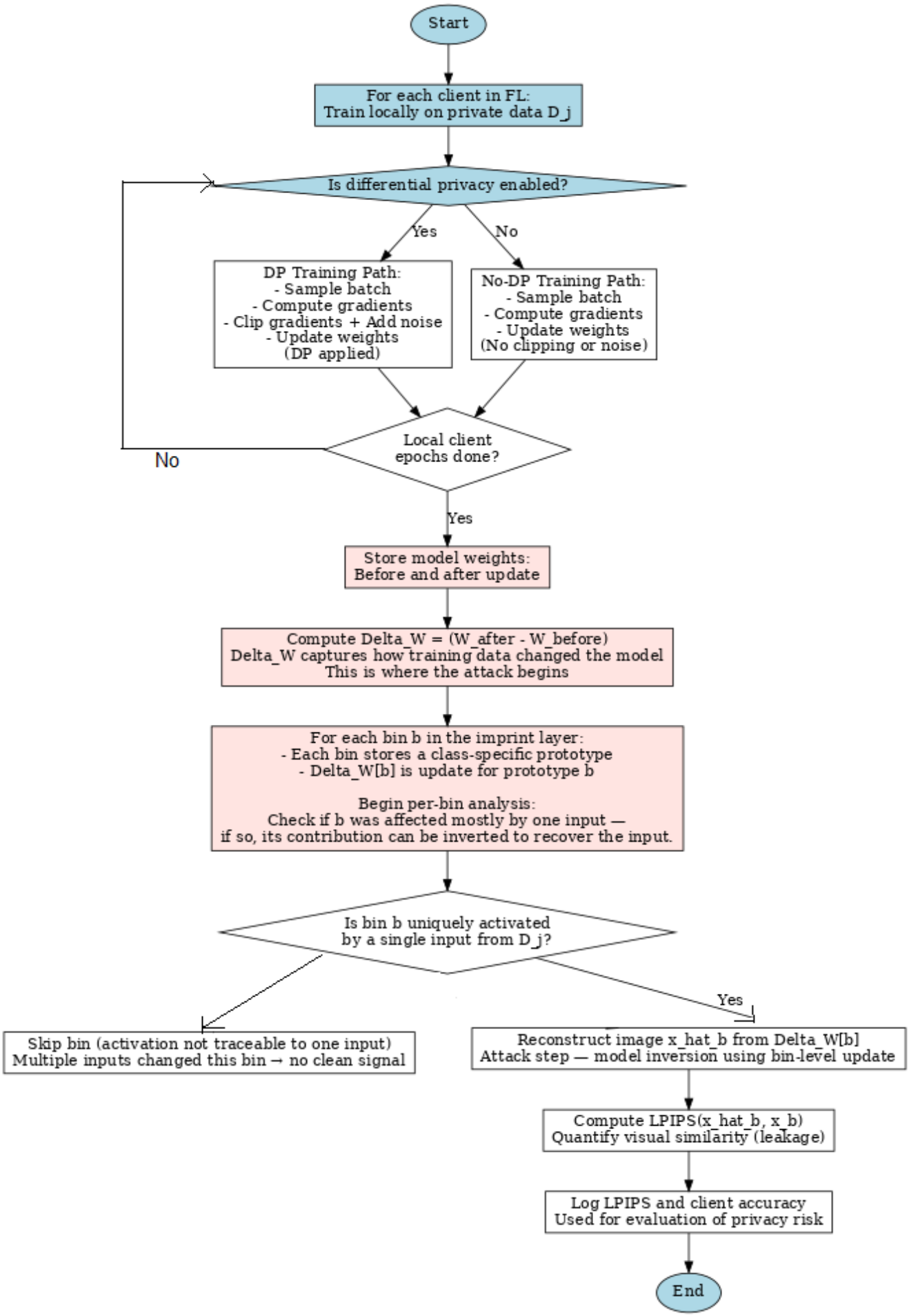


Figure 5.1: Client-Side workflow with optional DP and attack pipeline

Figure 5.1 presents the client-side training and attack workflow in FL. Each client trains its model using local data and updates the weights accordingly. A key branching point in the workflow determines whether DP is applied. If DP is enabled, the client clips the gradient to a fixed threshold and adds random Gaussian noise before updating the model. If not, the model is updated using the raw gradients.

Regardless of whether DP is applied, the model weights before and after local training are stored. By computing the weight difference, an attacker can analyze how specific parts of the model—such as the imprint layer bins—are updated. If a bin is significantly changed by only one input sample, it may be possible to reconstruct that sample using a model inversion attack.

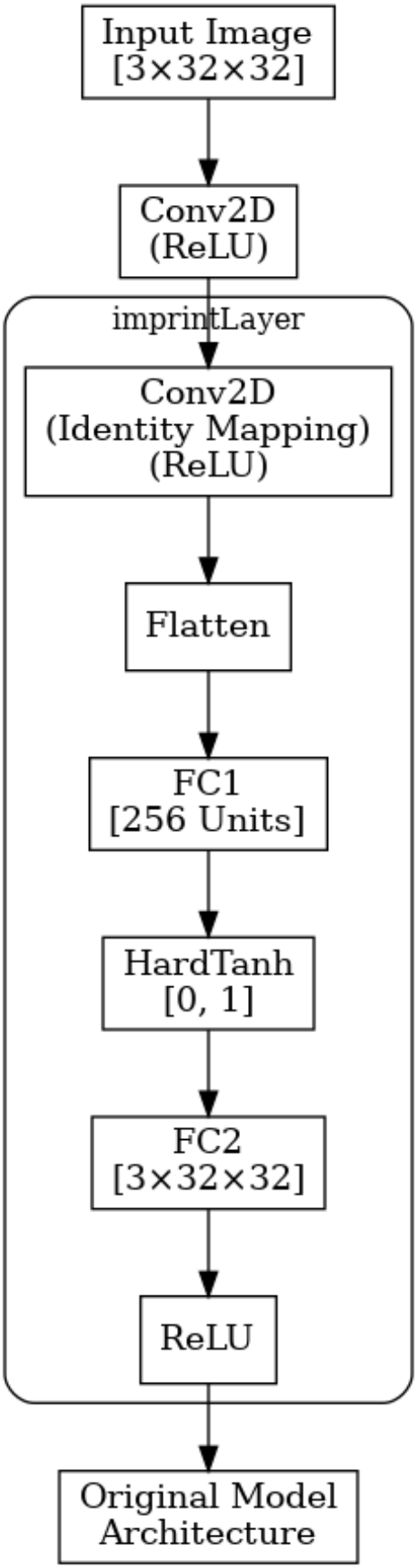


Figure 5.2: Attack module for Reconstruction process

Figure 5.2 illustrates the architecture of the attack module inserted before the original model in the FL setup. This module is responsible for enabling gradient-based input reconstruction attacks. The pipeline begins with a convolutional layer that employs identity mapping kernels, allowing the raw input image channels to be preserved and pushed forward without distortion. This output is then flattened into a vector and passed through a fully connected layer (FC1), which compresses the input into a lower-dimensional representation. A HardTanh activation bounds these activations between 0 and 1, facilitating interpretable activation patterns.

Following this, the FC2 layer projects the compact features back into the original image shape, and a ReLU ensures non-negative output suitable for further processing. The key insight behind the attack is that during training, gradients of FC1 weights are sensitive to the input data and client-specific. By capturing these gradients, an adversary can infer which activation bins were triggered and, in combination with the fixed identity convolution, reconstruct approximate input images. The final output of the attack module seamlessly connects to the original model, preserving the expected input shape and ensuring compatibility during training.

5.7 Measuring Utility and Privacy

In addition to LPIPS-based leakage evaluation, we monitor the classification accuracy of each client over training epochs. This allows us to assess how various DP configurations (C, ϵ) impact learning performance. For each configuration, the privacy budget ϵ is computed using an analytical formula based on the Gaussian DP model, taking into account the number of training steps, batch size, sampling rate, and noise multiplier.

5.8 Visualization and Analysis

The final reconstructed images are visualized in grid layouts alongside their corresponding ground-truth images to provide a qualitative understanding of leakage. Accuracy curves are plotted for all clients to highlight convergence patterns. We also report key metrics

such as the number of leaked images, average LPIPS scores, and computed ϵ values to facilitate an end-to-end evaluation of both attack strength and defense effectiveness.

5.9 Application of DLG-Based Attacks and Mitigation

The DLG attack was introduced to demonstrate how training data could be reconstructed from gradients in a collaborative FL setting. In its original form, DLG relies on iterative gradient matching: an adversary initializes dummy inputs and continuously updates them until the gradients they produce match the observed gradients shared by a client. While powerful, this approach is computationally intensive and assumes access to the model architecture.

In this work, we adapt the spirit of DLG through a simplified, linear reconstruction pipeline tailored to the architecture under study. Instead of optimizing dummy inputs, our attack directly analyzes how linear components (particularly the first convolutional and fully connected layers) leak sensitive information via gradients. We apply this linear leakage approach across multiple federated clients, storing and inspecting their aggregated gradients after each local training iteration.

To mitigate this attack, we incorporate DP in the form of the DP-SGD algorithm. Specifically, each client applies gradient clipping followed by Gaussian noise addition before sending updates to the server. The two key hyperparameters—clipping threshold and epsilon—are varied systematically to observe their impact on both LPIPS leakage and classification accuracy. For each configuration, we compute the resulting privacy budget (ϵ) using a standard analytical approximation [5].

By applying the attack across a range of DP settings, we identify thresholds where perceptual reconstructions become unrecognizable, albeit often at the cost of model utility. These findings highlight the importance of balancing privacy and performance and show that while DP can substantially degrade reconstructions, meaningful formal privacy guarantees (i.e., low ϵ) are hard to achieve without strong noise and aggressive clipping.

6 Results and Analysis

In this chapter, we present a comprehensive evaluation of our FL framework under different experimental conditions. We begin by assessing standard FL training performance without any privacy mechanisms or attacks, using the full dataset (MNIST) per client and then a reduced dataset of 64 images per client, to observe how client training accuracy evolves under varying data volumes (Figure 6.1). We then analyze how FL test accuracy changes under DP-SGD without attack, applying varying privacy budgets and clipping thresholds for 500 images per client (Figure 6.2) and for a representative case with 64 images per client (Figure 6.3). Next, we shift focus to FL under attack, introducing gradient-based reconstruction attacks and evaluating how training accuracy is impacted, both without DP and with DP as a defense (Figure 6.4). The attack experiments are limited to the 64-image-per-client setting due to the computational complexity of running gradient-based attacks at scale; this small data volume further compounds learning challenges, as both the attack itself and DP noise exacerbate the difficulty of achieving high utility. Building on this, we explore visual leakage through gradient-based reconstructions, comparing ground truth and recovered images with and without DP (Figure 6.5), and systematically evaluate LPIPS-based reconstruction quality and client classification accuracy under varying DP parameters (Figures 6.6–6.7). Finally, we extend the analysis to medical datasets such as PathMNIST, highlighting leakage trends, privacy–utility dynamics, and cross-domain findings (Figures 6.8–6.9). This structure allows us to first characterize FL performance without attack, and then systematically analyze FL under attack and attack mitigation strategies, providing a comprehensive view of learning dynamics in both settings.

We analyze both training and testing accuracy to provide a complete picture of how different FL configurations affect learning. Training accuracy helps us understand how well the model fits client data during local updates and aggregation, and how attacks and defenses influence learning dynamics across rounds. Testing accuracy shows how well the global model generalizes to unseen data across clients. This dual perspective is essential for assessing both convergence and the real-world utility and robustness of FL under privacy and attack conditions.

6.1 Federated Training Performance without Attack

Comparison of FL Training Accuracy(%) With Full and Limited Client Data:

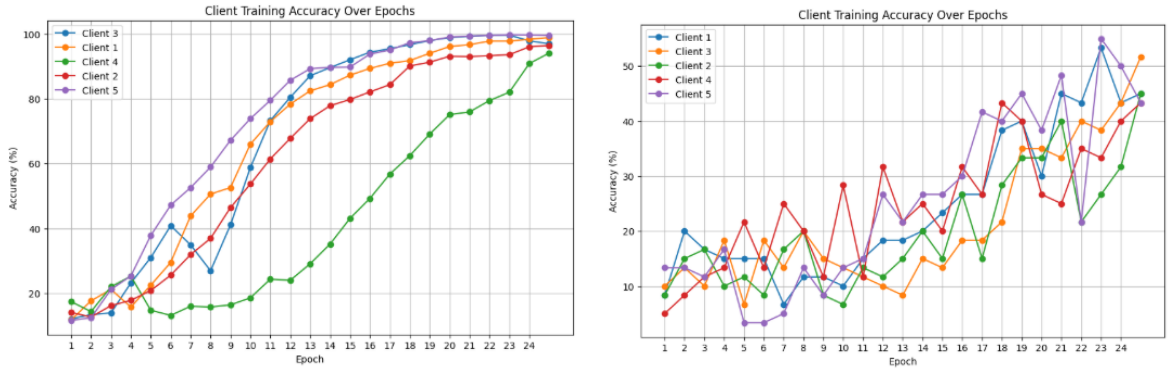


Figure 6.1: MNIST Client training accuracy (Left: FL with 5 clients and full dataset, Right: FL with 5 clients and 320 images total, i.e., 64 per client).

Figure 6.1 shows client-wise training accuracy for 5 clients over 25 communication rounds using the MNIST dataset, comparing two FL setups. Both settings use the same model architecture and training hyperparameters to enable fair comparison across all subsequent experiments. In one case, clients train on their full local datasets(10000 training images per client and 1000 testing images per client), while in the other, each client is limited to only 64 samples to simulate constrained learning. The notable findings from client-wise training accuracy in FL without attack are as follows:

1. Full Dataset with Large Batch Enables Smooth Convergence: In the left panel in Figure 6.1, clients train on their complete local datasets with large batch sizes.

Training is stable and highly effective: all clients quickly reach over 90% accuracy, with minimal variance. This outcome demonstrates that sufficient data volume and well-tuned training configurations allow for fast, smooth convergence in FL.

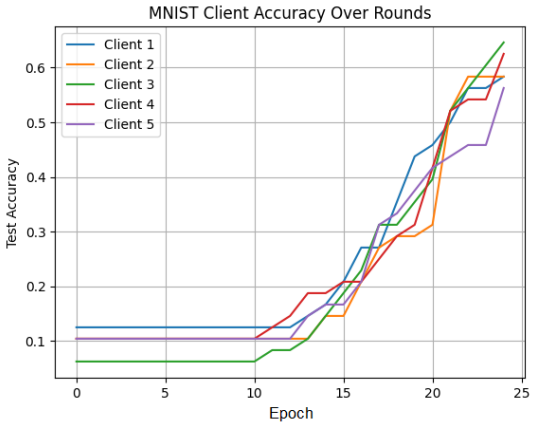
2. Limited Local Data Severely Hinders Learning: In the right panel in Figure 6.1, each client is restricted to just 64 samples and trains using small batch sizes. The resulting learning curves are noisy and exhibit high variance. Most clients remain below 55% accuracy even after 25 rounds. This degradation highlights the sensitivity of FL systems to data scarcity and gradient instability caused by small batch training.

3. Consistent Architecture Highlights Sensitivity to Data Scale: Despite identical model architectures, optimizers, and aggregation strategies across both experiments, performance differs drastically. This underscores the fact that training success in FL is not just a function of the model but heavily depends on data scale and batch size.

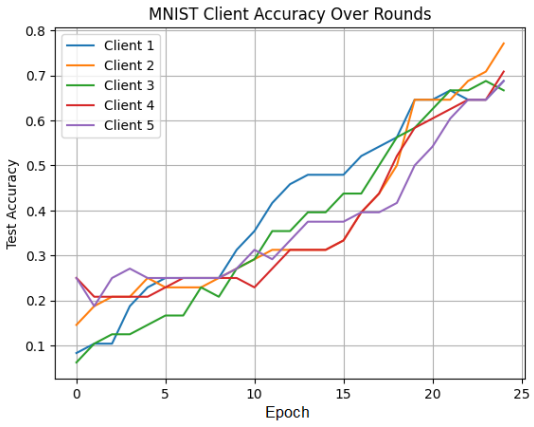
4. Small Data Leads to Unstable Local Updates: With just 64 samples per client, local training suffers from poor generalization and highly variable gradient updates. These inconsistencies propagate through the aggregation process, making it difficult for the global model to converge effectively.

5. Practical FL Requires Careful Data and Batch Size Design: These results emphasize the importance of careful design when deploying FL systems in real-world environments. Ensuring that clients have access to sufficient data, both in volume and diversity is critical for stability, convergence, and overall model performance.

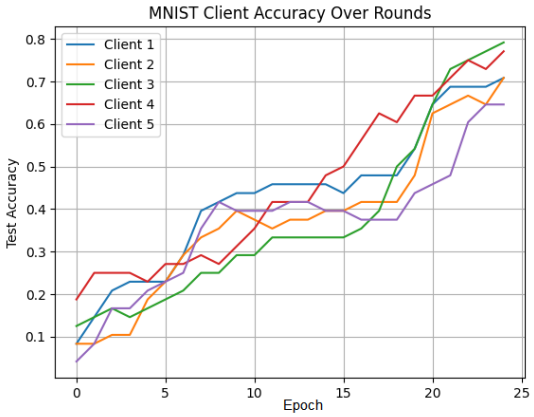
Comparison of FL Test Accuracy (%) with DP-SGD across varying configurations (500 images per client):



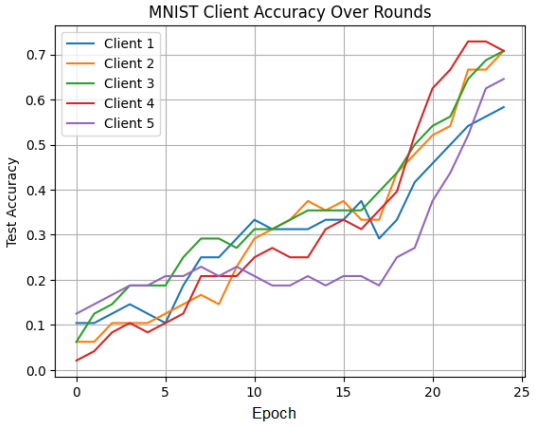
(a) C = 10, $\epsilon = 8.31 \times 10^{15}$



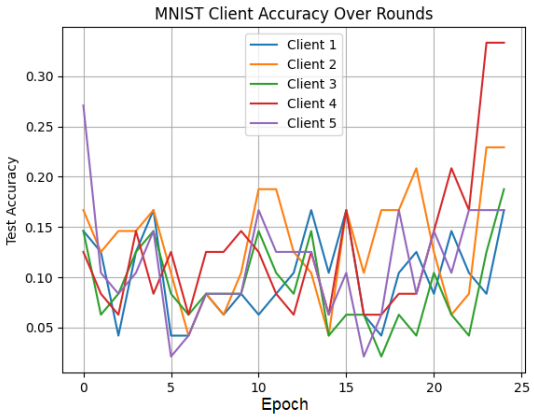
(b) C = 100, $\epsilon = 8.31 \times 10^{15}$



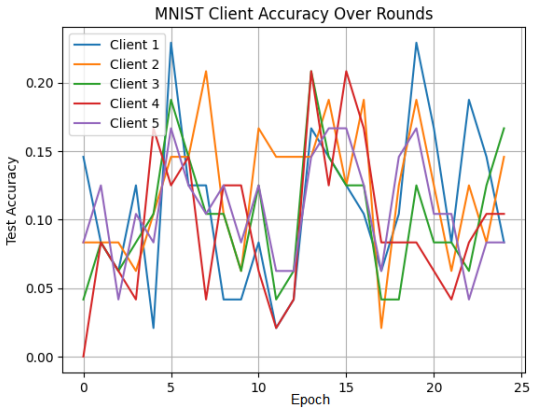
(c) C = 10, $\epsilon = 8.31 \times 10^{12}$



(d) C = 100, $\epsilon = 8.31 \times 10^{12}$



(e) C = 100, $\epsilon = 8.31 \times 10^{6}$



(f) C = 10000, $\epsilon = 8.31 \times 10^{6}$

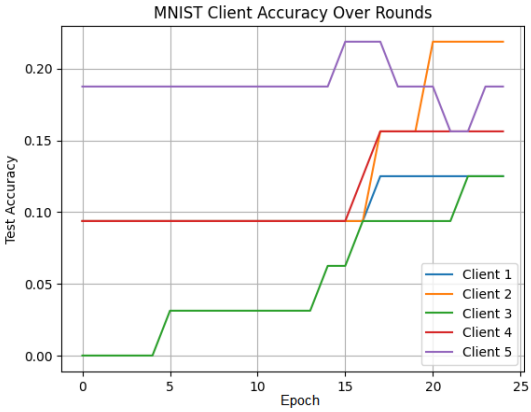
Figure 6.2: MNIST client test accuracy over 25 federated rounds(epochs) with varying DP configurations (500 images per client).

The results shown in Figure 6.2 present MNIST client test accuracy over 25 federated *rounds*, labeled as "epochs" in the figure, under different differential privacy (DP) configurations. Each round corresponds to one global communication cycle, where clients perform local training and send their updated models to the server. The server then performs aggregation (FedAvg) to produce the new global model for the next round. The accuracy is evaluated on each client's private test set after each round to assess generalization performance. The plots capture how varying the clipping threshold C and privacy budget ϵ affects learning dynamics, highlighting the balance between privacy protection and model utility in DP-SGD. The notable findings from Figure 6.2 are as follows:

1. **Impact of Privacy Budget:** Higher ϵ values (weaker privacy, smaller noise) consistently enable the model to learn effectively, with accuracy improving across rounds. As ϵ decreases (stronger privacy, larger noise), model utility declines sharply. In the lowest values of ϵ for our experiments, accuracy remains low and fluctuates significantly, as DP noise overwhelms the gradient signal and hinders meaningful learning.

2. **Effect of Clipping Threshold:** The clipping threshold C strongly influences the interaction between gradients and DP noise. A small threshold (e.g., $C = 10$) helps limit gradient magnitudes, which in turn reduces the impact of added noise and stabilizes training. In contrast, a large threshold (e.g., $C = 10000$) allows large gradients to pass through, causing excessive noise to be added, which severely disrupts learning. The plots show that large clipping thresholds combined with DP noise result in poor convergence and unstable accuracy.

Comparison of FL Test Accuracy(%) With DP-SGD Using Limited Client Data (64 Images per Client):



C = 10, $\epsilon = 8.31 \times 10^5$, 64 images per client

Figure 6.3: MNIST client test accuracy over 25 federated rounds with DP-SGD using 64 training images per client ($\epsilon = 8.31 \times 10^5$, $C = 10$).

Figure 6.3 illustrates the effect of data volume on FL performance under DP-SGD when clients have only 64 training images each. Compared to the 500-image setting, the accuracy is significantly lower and fluctuates throughout training. This demonstrates how limited data availability amplifies the harmful effects of DP noise: with fewer examples, gradient estimates are noisier, and the injected DP noise more easily overwhelms the signal. The results highlight that data scarcity severely limits the ability of DP-SGD to balance privacy and utility, making effective learning under strong privacy constraints much harder in low-data federated settings. Other stronger privacy settings yielded even lower accuracy, so only this representative case is shown.

Summary of Federated Training Performance without Attack:

The results demonstrate how model utility evolves across different FL configurations. Plain FL with 500 training images or more per client achieves the highest accuracy and most stable learning, as expected in the absence of privacy constraints. When the data volume is reduced to just 64 images per client, plain FL performance declines, with slower convergence and lower final accuracy due to the limited training signal. Introducing differential privacy (DP) further impacts utility: FL with DP-SGD and 500 images per client shows a clear privacy–utility trade-off, where increasing privacy (lower ϵ) through

stronger noise or ineffective clipping thresholds leads to reduced accuracy. The impact is even more pronounced when using DP with only 64 training images per client, where the combination of data scarcity and DP noise severely limits learning capacity. These findings highlight the importance of balancing data volume, clipping thresholds, and noise levels to sustain utility in privacy-preserving FL. In the next subsection, we focus on *Federated Training Performance with Attack and Attack Mitigation*, where we analyze how training accuracy and reconstruction vulnerability are affected when gradient-based attacks are introduced and various DP parameters are applied to defend against them.

6.2 Federated Training Performance with Attack and Attack Mitigation

Comparison of FL Training Accuracy(%) Under Attack (Without DP vs. With DP):

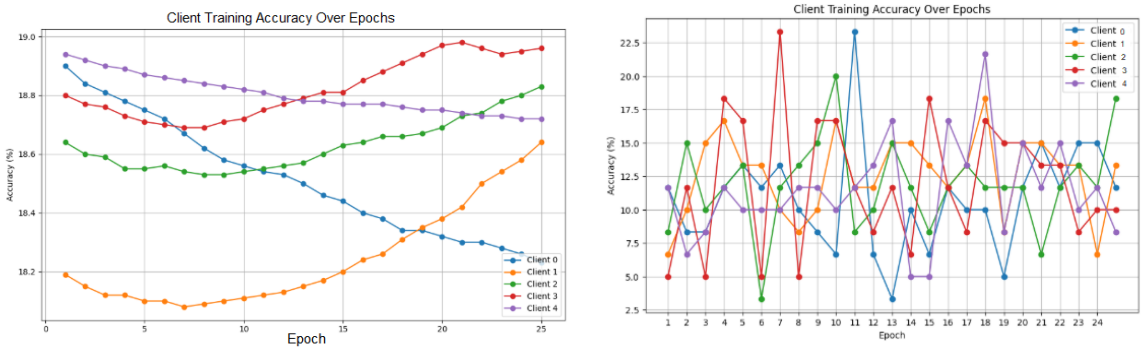


Figure 6.4: MNIST Client training accuracy under attack (Left: without DP, Right: with DP (C = 10000, $\epsilon = 8.31 \times 10^9$)).

Figure 6.4 presents a comparison between two FL scenarios where malicious influence or inference is present. On the left, the setup includes an adversarial or poisoning attack without any privacy-preserving mechanism. Although the accuracies remain somewhat stable, they exhibit slow growth and divergence across clients, indicating the influence of poisoned gradients or manipulative behaviors from certain clients. Some clients improve gradually, while others degrade or remain flat, a sign of compromised update consistency. It is important to note that the overall training accuracies remain low primarily due to

the highly downsampled setting, where each client is trained on only 64 images. This limited data severely restricts generalization, and thus the low accuracy should not be interpreted as a direct effect of the attack or defense. In fact, with full MNIST, even random guessing over 10 classes would yield around 10% accuracy, indicating that our reduced setting operates well below this baseline due to its constrained scale.

The right plot shows a similar setup, but with DP applied with clipping threshold 10000, $\epsilon = 8.31 \times 10^9$ applied during training. Here, the training accuracy is significantly lower and highly unstable across all clients. This degradation is due to the injected DP noise, which disrupts the attacker's influence but also reduces learning signal strength—especially in already noisy or limited-data settings. The random fluctuations and low convergence show the difficulty of balancing privacy, security, and performance in adversarial FL environments.

Comparison of Image Reconstruction in FL on MNIST (Ground Truth vs. Without DP vs. With DP):

Figure 6.5: Image reconstruction in FL on MNIST for a random client(Left: Ground truth, Middle: Without DP, Right: With DP(C = 10000, $\epsilon = 8.31 \times 10^9$)

Figure 6.5 compares reconstructions under standard FL (no DP) and DP-enabled training. The notable findings from Gradient-Based Reconstruction in our FL setting with and without DP are given below:

1. FL Gradients Without DP Reveal Sensitive Input Data: In the middle panel in Figure 6.5, reconstructions from a non-private FL setup show that model gradients can encode sufficient information to recover original input images. The reconstructed MNIST

digits are sharp and easily identifiable, confirming the vulnerability of gradient-sharing protocols in FL without any privacy protection. This leakage is especially pronounced under small batch sizes and over-parameterized models, where precise directional updates amplify inversion success.

2. Gradient Inversion Attacks Pose Real Privacy Risks: Such reconstruction capabilities represent a serious privacy threat, particularly in applications involving sensitive data like handwriting, medical images, or personal texts. The clear structure of the digits in non-private settings indicates that gradients are not only informative but also highly specific to individual training samples—thus violating user privacy despite the absence of raw data sharing.

3. Differential Privacy Significantly Degrades Reconstructions: The rightmost panel in Figure 6.5 shows reconstructions when DP is applied. These images are largely unrecognizable, exhibiting visual noise or broken digit fragments. Here, black squares indicate failed reconstructions where no meaningful image could be recovered. Hazy images represent partial leakage, showing some structure but with significant noise or distortion. This reflects stronger privacy at the black squares and moderate leakage where images appear blurry. This degradation stems from the combined effect of gradient clipping and Gaussian noise injection, which reduce the informativeness of per-client updates. As a result, the attacker's ability to reconstruct user inputs is significantly impaired.

4. DP Strength Determines the Level of Protection: While DP reduces reconstruction success, the effectiveness depends on the strength of the parameters used. In some DP-protected examples, faint digit outlines remain visible, suggesting that noise or clipping were not applied at their most stringent levels. This observation reinforces the importance of tuning DP parameters properly to strike a balance between privacy and model utility.

5. Motivation for a Systematic Exploration of the Privacy–Utility Trade-off: These qualitative results motivate the systematic evaluation in the following sections, where we vary noise multipliers and clipping thresholds to observe their impact on both privacy leakage (via LPIPS and visual reconstructions) and utility (via accuracy). The privacy budget ϵ, derived from these settings, will serve as a reference point for analyzing the effectiveness of DP in protecting user data in FL all throughout our thesis.

6.3 Reconstruction Quality under varying DP Parameters: MNIST & PathMNIST

The level of visual information leakage in FL reconstructions is influenced by both the clipping threshold(C) and the DP budget (ϵ). To systematically interpret the relationship between reconstruction quality and privacy, we categorize the observed LPIPS leakage scores into privacy tiers, as summarized in Table 6.1. In this section, we analyze the reconstruction behavior for the MNIST and PathMNIST datasets under varying DP settings. Supplementary results for additional datasets including RetinaMNIST, CIFAR-10, BreastMNIST, and DermaMNIST are provided in the appendix for further demonstration.

LPIPS quantifies perceptual similarity between original and reconstructed images; lower LPIPS values (e.g., < 0.01) indicate highly accurate reconstructions and thus weaker privacy. Conversely, higher LPIPS values (e.g., > 0.5) suggest strong privacy, where reconstructions are heavily distorted or visually unrecognizable. While ϵ remains an important parameter in controlling DP strength, LPIPS serves as a direct, perceptual measure of privacy leakage, enabling a more practical interpretation of visual risk in FL.

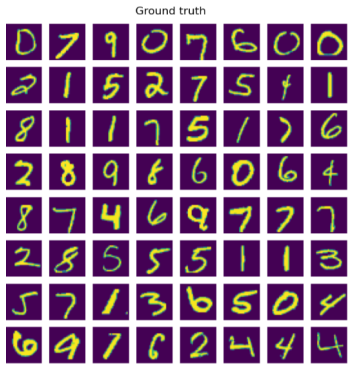
LPIPS Range	**Comment Description**
> 0.50	**High privacy — reconstructions heavily distorted or noisy**
0.35 – 0.50	**Moderate privacy — partial leakage visible**
0.01 – 0.35	**Low privacy — reconstructions largely intact**
< 0.01	**Very low privacy — reconstructions highly accurate**

Table 6.1: Interpretation of comments based on LPIPS leakage ranges

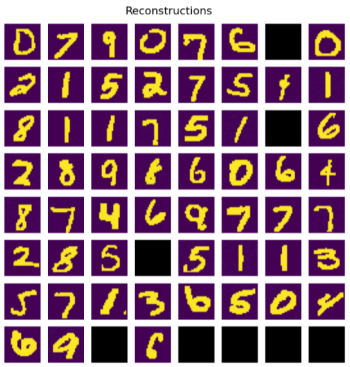
6.3.1 Evaluating Reconstruction and Utility under varying Clipping Thresholds and Privacy Budgets for MNIST

We report privacy leakage (with attack) and model utility (test accuracy without attack) under separate conditions to ensure a fair and meaningful evaluation of the privacy–utility trade-off. Applying a gradient-based reconstruction attack directly alters the learning dynamics by leaking gradient information and introducing linear leakage effects and therefore, measuring utility during an attack would mix up the effects of the attack and the privacy mechanisms. By separating them, we can clearly see how DP protects against leakage and how it affects normal model performance without attackers. As a result, if we measured utility during an active attack, the observed accuracy would reflect both the impact of the attack itself and the privacy mechanisms, making it impossible to isolate the true cost of differential privacy (DP) on model performance.

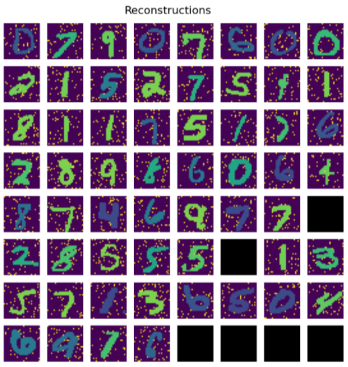
Furthermore, we deliberately use a reduced client dataset (64 images per client) for the reconstruction experiments. This is because gradient inversion attacks — especially when combined with DP mechanisms — are computationally intensive and challenging to run. The 64-image setting strikes a balance between computational feasibility and the ability to demonstrate leakage effectively. At the same time, we evaluate model utility (test accuracy) in a higher-data setting (500 images per client) to reflect more realistic FL deployments and ensure the results represent how DP would affect learning without adversarial interference. This design allows us to separately and clearly study how DP mitigates privacy risks under attack and how it impacts model performance when no attacker is present — which is typically the case in real-world applications requiring privacy guarantees.



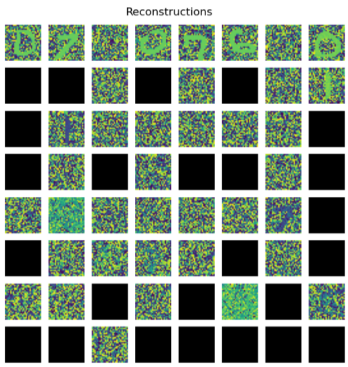
(a) Ground truth (MNIST)



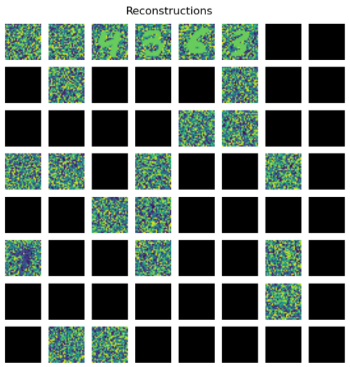
(b) C = 10, $\epsilon = 8.31 \times 10^{15}$, LPIPS = 0.3078



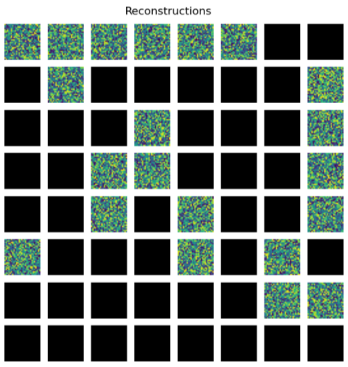
(c) C = 100, $\epsilon = 8.31 \times 10^{15}$, LPIPS = 0.3231



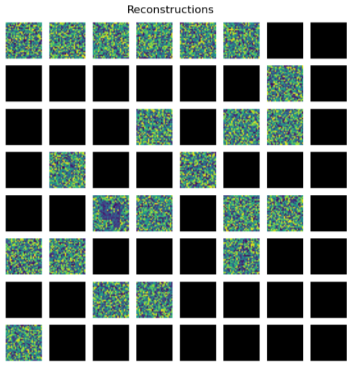
(d) C = 10, $\epsilon = 8.31 \times 10^{12}$, LPIPS = 0.5511



(e) C = 10000, $\epsilon = 8.31 \times 10^{12}$, LPIPS = 0.5940



(f) C = 100, $\epsilon = 8.31 \times 10^{6}$, LPIPS = 0.5932



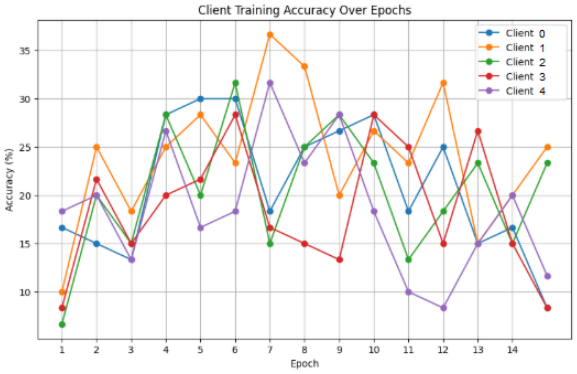
(g) C = 10000, $\epsilon = 8.31 \times 10^{6}$, LPIPS = 0.5961

Figure 6.6: LPIPS reconstructions on **MNIST** for a random client for varying ϵ and varying clipping thresholds arranged by decreasing privacy budget ϵ (i.e., from least private to most private).

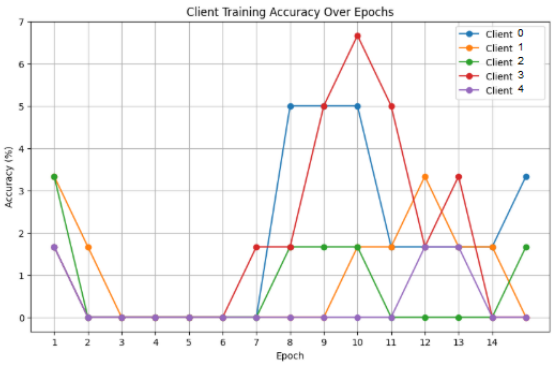
(a) C = 10, $\epsilon = 8.31 \times 10^{15}$

(b) C = 100, $\epsilon = 8.31 \times 10^{15}$

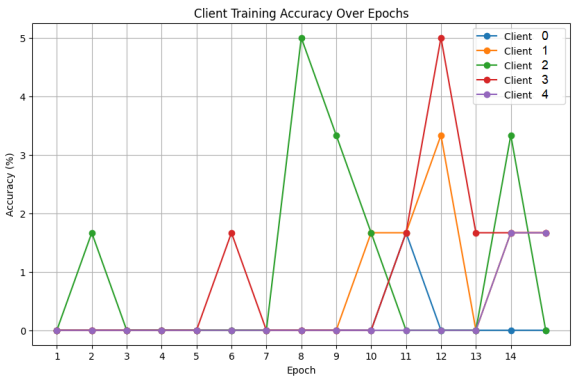
(c) C = 10, $\epsilon = 8.31 \times 10^{12}$



(d) C = 10000, $\epsilon = 8.31 \times 10^{12}$



(e) C = 100, $\epsilon = 8.31 \times 10^{6}$



(f) C = 10000, $\epsilon = 8.31 \times 10^{6}$

Figure 6.7: Client-wise classification accuracy on **MNIST** under varying clipping thresholds and estimated privacy budgets (ϵ) with attack mechanism included.

Epsilon	Clipping Threshold	LPIPS	Test Accuracy	Comments
8.31×10^{15}	10	0.3078	0.75	**Low privacy — reconstructions largely intact; Test accuracy remains high as noise effect is minimal**
8.31×10^{15}	100	0.3231	0.75	**Low privacy — reconstructions largely intact; Test accuracy remains high as noise effect is minimal**
8.31×10^{12}	10	0.5511	0.45	**High privacy — reconstructions heavily distorted or noisy; Test accuracy moderately degraded due to DP noise**
8.31×10^{12}	10^4	0.5940	0.25	**High privacy — reconstructions heavily distorted or noisy; Test accuracy significantly reduced by DP**
8.31×10^{6}	100	0.5932	0.06	**High privacy — reconstructions heavily distorted or noisy; Test accuracy severely degraded under strong DP**
8.31×10^{6}	10^4	0.5961	0.05	**High privacy — reconstructions heavily distorted or noisy; Test accuracy severely degraded under strong DP**

Table 6.2: MNIST privacy–utility analysis: LPIPS leakage and client test accuracy under varying ϵ and clipping thresholds.

The notable findings from our MNIST experiments(as demonstrated in Table 6.2, Figure 6.7 and Figure 6.6) focusing on the effect of varying differential privacy parameters such as clipping threshold and privacy budget (ϵ) are as follows:

1. LPIPS Leakage Reflects the Effectiveness of Differential Privacy: Table 6.6 presents the LPIPS leakage observed on the **MNIST** dataset across various combinations of C and ϵ under the DP-SGD framework. LPIPS measures perceptual similarity between

reconstructed and original images, with higher values indicating higher distortion and stronger privacy.

A clear trend emerges: **strong privacy—reflected in high LPIPS values—occurs only when both the clipping threshold is small and the privacy budget (ϵ) is tight (e.g., $\epsilon = 8.31 \times 10^{12}$ or 8.31×10^{6} with threshold = 10 or 100)**. These configurations suppress sensitive gradient signals effectively, leading to highly distorted reconstructions and degraded model utility. In contrast, **large clipping thresholds (e.g., 10^4)** allow large-magnitude gradients to pass through, resulting in significant visual leakage even under stronger privacy settings. This demonstrates that both ϵ and the clipping threshold must be carefully tuned to balance privacy and utility.

Additionally, moderate clipping with a large privacy budget (e.g., $C = 100$ and $\epsilon = 8.31 \times 10^{15}$) leads to poor privacy, reaffirming that noise alone is not sufficient. These results emphasize that LPIPS-based leakage is governed by the interaction between clipping and the privacy budget, not by either factor alone.

2. Training Accuracy Degrades Under Strong Privacy Budgets: Figure 6.7 shows client-wise classification accuracy across different privacy configurations. We observe that settings with very large privacy budgets (e.g., $\epsilon = 8.31 \times 10^{15}$) allow stable convergence and higher final accuracy across clients, even with relatively tight clipping thresholds like 10 or 100. In contrast, as ϵ is reduced to values such as 8.31×10^{12} or 8.31×10^{6}, model performance degrades significantly—particularly when strong clipping is applied. This indicates that stricter privacy comes at the cost of slower convergence and lower utility, consistent with well-known trade-offs in differential privacy.

3. The Utility–Privacy Trade-off in Federated Learning: The classification results in our experiments, depicted by Table 6.2 clearly illustrate the fundamental tension between privacy and model utility in FL. Configurations with large privacy budgets (e.g., $\epsilon = 8.31 \times 10^{15}$) and loose clipping thresholds (e.g., 100) result in minimal noise injection and weak gradient clipping. This preserves gradient information, allowing the model to converge effectively and achieve high test accuracy (around 0.75), but it leaves the sys-

tem vulnerable to gradient inversion attacks, as shown by low LPIPS leakage scores and highly accurate reconstructions.

Conversely, when stricter privacy configurations are applied—such as reducing ϵ to 8.31×10^{12} or lower, or combining tighter clipping thresholds (e.g., 10) with stronger noise—the test accuracy degrades significantly (falling to 0.45, 0.25, or even 0.06). This is because the stronger privacy constraints introduce substantial perturbation to the gradients and suppress useful learning signals, making optimization harder on the test data.

This behavior reflects the expected privacy–utility trade-off, but also highlights why achieving a perfect balance is challenging: configurations that successfully block visual leakage tend to impair model learning, while those that support strong test performance leave client data vulnerable to reconstruction. Our results emphasize that both the privacy budget and clipping threshold need to be jointly tuned to find an acceptable middle ground for practical deployments.

4. Visual Privacy Risk Beyond Theory: Although ϵ is designed to quantify privacy guarantees theoretically, our findings reveal a disconnect between these values and real-world leakage potential. As shown in Figure 6.6, with extremely large privacy budgets (e.g., $\epsilon = 8.31 \times 10^{15}$), reconstructed images remain visually precise and reveal sensitive content as expected. As ϵ decreases, reconstructions degrade progressively. However, leakage remains substantial until ϵ reaches lower values (e.g., 8.31×10^{6}), when reconstructions collapse and become heavily noisy. In our experiments, we observed that even relatively large privacy budgets (e.g., $\epsilon \sim 10^{6}$) provided strong visual protection, with heavily distorted reconstructions. While theoretical guidelines typically recommend $\epsilon < 10$ for strong privacy, our results suggest that such strict thresholds may be overly conservative in practice and for the implemented experiments, a very weak amount of noise is enough to degrade the reconstructions.

6.3.2 Evaluating Reconstruction and Utility under varying Clipping Thresholds and Privacy Budgets for PathMNIST

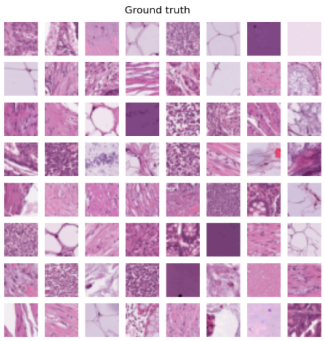
We illustrate how reconstruction quality on PathMNIST varies with ϵ, highlighting the trade-off between reconstruction level and privacy through LPIPS comparisons and corresponding privacy budgets.

The notable findings from our PathMNIST experiments, focusing on the effect of varying DP parameters such as clipping threshold and privacy budget (ϵ) are as follows:

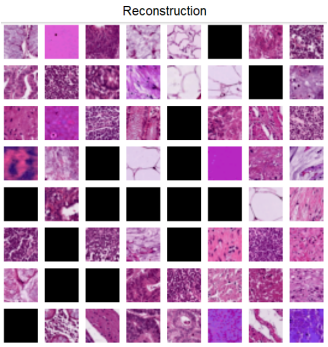
Epsilon	Clipping Threshold	LPIPS	Comments
8.31×10^3	10	0.5437	**High privacy — reconstructions heavily distorted or noisy**
8.31×10^5	10	0.3701	**Moderate privacy — partial leakage visible**
8.31×10^5	100	0.5421	**High privacy — reconstructions heavily distorted or noisy**
8.31×10^5	10000	0.5455	**High privacy — reconstructions heavily distorted or noisy**
8.31×10^6	1	0.0433	**Low privacy — reconstructions largely intact**
8.31×10^8	10	0.0010	**Very low privacy — reconstructions highly accurate**
8.31×10^8	10000	0.3623	**Moderate privacy — partial leakage visible**
8.31×10^{11}	1	0.0013	**Very low privacy — reconstructions highly accurate**
8.31×10^{11}	10	0.0004	**Very low privacy — reconstructions highly accurate**
8.31×10^{11}	100	0.0004	**Very low privacy — reconstructions highly accurate**
8.31×10^{11}	10000	0.0010	**Very low privacy — reconstructions highly accurate**

Table 6.3: LPIPS leakage on **PathMNIST** under varying clipping thresholds, ordered by increasing ϵ.

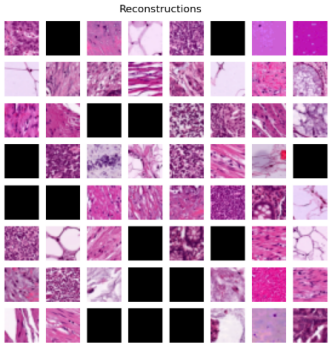
1. Strong Privacy Requires Both Tight Clipping and Small ϵ: Table 6.3 presents LPIPS leakage scores under various combinations of clipping thresholds and privacy budgets on the **PathMNIST** dataset. Strong privacy—reflected in high LPIPS values and distorted reconstructions—is generally achieved only when both the clipping threshold is low and the privacy budget (ϵ) is small. For example, clipping threshold = 10 paired with



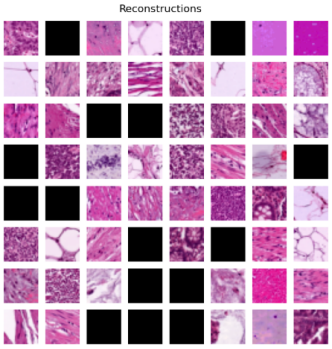
(a) Ground Truth (PathMNIST)



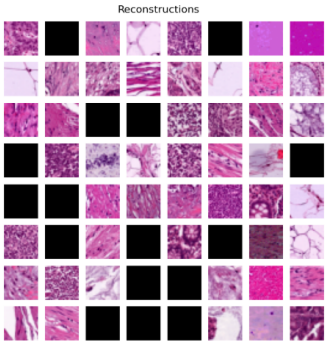
(b) C = 1, $\epsilon = 8.31 \times 10^{11}$, LPIPS = 0.0013



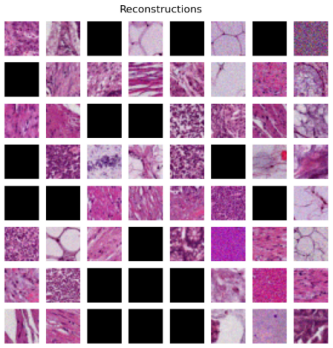
(c) C = 10, $\epsilon = 8.31 \times 10^{11}$, LPIPS = 0.0004



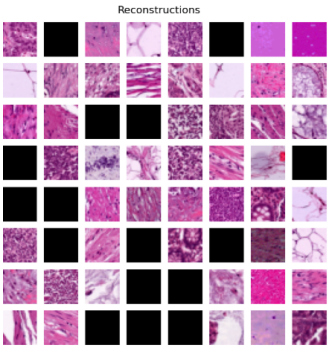
(d) C = 100, $\epsilon = 8.31 \times 10^{11}$, LPIPS = 0.0004



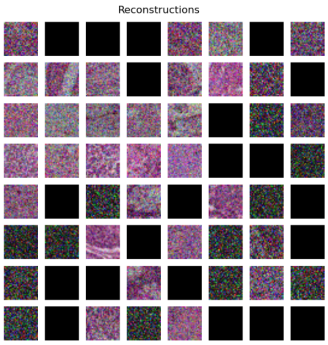
(e) C = 10000, $\epsilon = 8.31 \times 10^{11}$, LPIPS = 0.0010



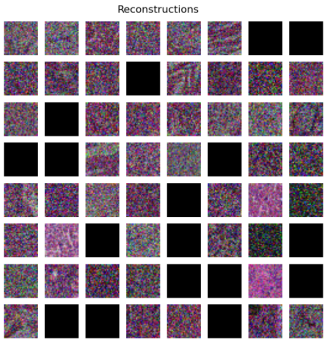
(f) C = 1, $\epsilon = 8.31 \times 10^{7}$, LPIPS = 0.0433



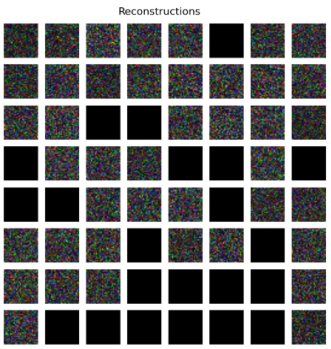
(g) C = 10, $\epsilon = 8.31 \times 10^{8}$, LPIPS = 0.0010



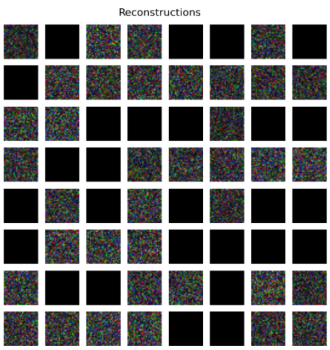
(h) C = 10000, $\epsilon = 8.31 \times 10^{8}$, LPIPS = 0.3623



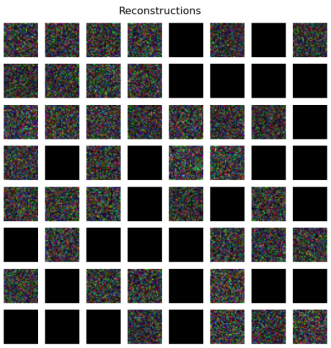
(i) C = 10, $\epsilon = 8.31 \times 10^{5}$, LPIPS = 0.3701



(j) C = 100, $\epsilon = 8.31 \times 10^{5}$, LPIPS = 0.5421



(k) C = 10000, $\epsilon = 8.31 \times 10^{5}$, LPIPS = 0.5455



(l) C = 10, $\epsilon = 8.31 \times 10^{3}$, LPIPS = 0.5437

Figure 6.8: LPIPS reconstructions on **PathMNIST** for a random client for varying ϵ and varying clipping thresholds arranged by decreasing privacy budget ϵ.

$\epsilon = 8.31 \times 10^3$ yields an LPIPS score of 0.54, indicating high privacy, while $\epsilon = 8.31 \times 10^5$ results in 0.37, reflecting moderate privacy.

2. Large ϵ or Loose Clipping Leads to Severe Leakage: In contrast, configurations with large privacy budgets or loose clipping yield highly accurate reconstructions and low LPIPS scores—even at relatively high ϵ values. For instance, clipping threshold = 10 or 100 combined with $\epsilon = 8.31 \times 10^{11}$ produces LPIPS below 0.002, indicating substantial leakage. Surprisingly, even clipping threshold = 1 with $\epsilon = 8.31 \times 10^{11}$ achieves LPIPS = 0.0013, highlighting that visual leakage remains a concern at this level for our experiments.

3. Same ϵ Can Yield Different Privacy Outcomes: Notably, different configurations with the same ϵ value can result in drastically different LPIPS outcomes due to the interaction between clipping and noise. For example, both clipping threshold = 10 and clipping threshold = 10,000 with $\epsilon = 8.31 \times 10^8$ correspond to the same privacy budget, but the former yields LPIPS = 0.0010 (very low privacy) and the latter yields LPIPS = 0.3623 (moderate privacy). This divergence shows that ϵ alone is not sufficient to evaluate real-world privacy risk, especially in vision tasks. The clipping threshold significantly influences the magnitude and structure of the gradients being shared, as larger clipping thresholds allow more signal to pass through, potentially undermining the noise effect and increasing privacy leakage despite identical ϵ values.

4. Implications for Medical Imaging Privacy: These results reinforce that LPIPS is a useful perceptual metric for evaluating reconstruction risk. In datasets like PathMNIST, where medical images may reveal sensitive patterns, overlooking the proper tuning of DP parameters can expose FL systems to high-fidelity reconstruction attacks. Ensuring meaningful privacy requires a careful balance between clipping and noise, beyond what ϵ suggests in theory.

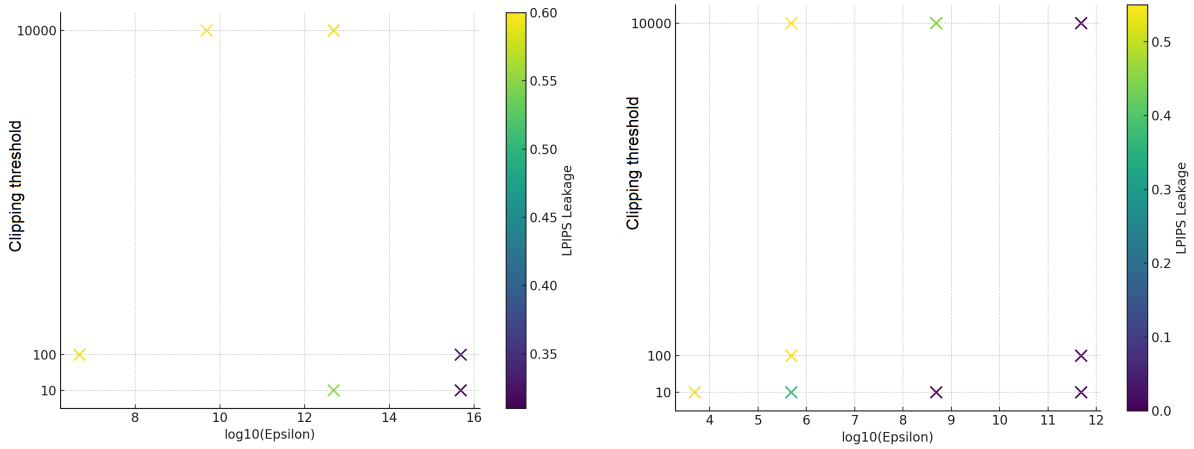
6.3.3 Empirical Analysis of Epsilon vs. LPIPS Leakage

Our experimental objective was to invert the conventional privacy analysis: instead of fixing a target ϵ and adjusting noise accordingly, we held noise and clipping parameters fixed and measured the resulting ϵ based on training settings and reported ϵ instead of noise multipliers. In our **PathMNIST** experiments, by sweeping over ϵ values from approximately 8.31×10^3 up to 8.31×10^{15} and observing corresponding LPIPS scores, we found that meaningful visual degradation (e.g., LPIPS > 0.3) only emerged at lower ϵ — indicating stronger privacy. At higher ϵ values (e.g., 8.31×10^{11} and above), reconstructions were highly accurate (LPIPS < 0.01), reflecting weak privacy guarantees despite high numerical ϵ. In some configurations (e.g., $\epsilon = 8.31 \times 10^6$ with large clipping thresholds), reconstructions were heavily distorted and visually uninformative (suggesting good practical privacy), yet the theoretical ϵ was still large. Our results show that in **PathMNIST** image-based federated learning tasks, high ϵ values frequently correlate with low LPIPS and highly accurate reconstructions — highlighting privacy vulnerabilities even when numerical privacy guarantees are theoretically acceptable.

Similarly, in **MNIST**, we observed that lower ϵ values (e.g., 8.31×10^{12} or below) led to meaningful visual distortion (LPIPS > 0.5), indicating high privacy, whereas higher ϵ values (e.g., 8.31×10^{15}) resulted in low privacy, with largely intact reconstructions (LPIPS ≈ 0.3).

6.3.4 Notable Findings from LPIPS vs. Privacy Budget Analysis

1. LPIPS Increases with ϵ Across Datasets: Figure 6.9 presents two scatterplots showing the relationship between the DP privacy budget ϵ, LPIPS leakage, and clipping thresholds for the **PathMNIST** and **MNIST** datasets. A consistent trend is observed in both datasets: as ϵ increases—indicating weaker privacy guarantees—LPIPS also decreases. This confirms that lower noise allows more informative gradients to pass through, leading to reconstructions that are perceptually closer to the original data. Conversely, settings with very low ϵ (due to strong noise) exhibit high LPIPS values, reinforcing the protective role of noise in mitigating gradient-based privacy attacks.



(a) **MNIST:** LPIPS vs Epsilon colored by clipping threshold

(b) **PathMNIST:** LPIPS vs Epsilon colored by clipping threshold

Figure 6.9: Comparison of LPIPS leakage vs. privacy budget (ϵ) under various clipping thresholds for PathMNIST and MNIST datasets

2. C Significantly Affects Leakage: In both scatterplots, the color scale encodes the clipping threshold magnitude. Settings with larger clipping thresholds (yellow hues) tend to show higher LPIPS values, even when ϵ remains constant. This highlights that weak clipping alone can undermine privacy, enabling visual leakage despite theoretically strong privacy guarantees based on ϵ. On the other hand, tighter clipping (purple hues) restricts gradient magnitude more effectively and contributes to better privacy protection—even at relatively large ϵ values. These findings underscore that both clipping and noise must be jointly tuned, as relying on ϵ alone may not ensure practical privacy.

3. The Threshold Effect and Limitations of ϵ alone: The visual trendlines suggest a clipping threshold effect: once noise levels pass a certain point, LPIPS drops sharply—yet the formal ϵ value may still remain relatively high (e.g., $> 10^6$). This observation raises concerns about relying solely on (ϵ, δ) values for assessing privacy risk in deep learning, particularly for vision tasks.

Although we observe increasing reconstruction distortion as ϵ decreases, it is important to note that the privacy budgets evaluated in this work are still considered extremely large by differential privacy standards. Theoretical foundations of DP, as originally proposed by Dwork et al. [3], suggest that acceptable privacy levels typically correspond to $\epsilon \leq 1$,

while many applied deep learning studies adopt values below 10. In practice, even large-scale deployments tend to restrict ϵ to below 50. By contrast, the smallest values tested in our experiments remain orders of magnitude higher, and therefore do not meet theoretical thresholds for strong privacy guarantees.

What is notable, however, is that despite these high ϵ values—formally considered weak privacy—we still observe significant degradation in reconstruction quality and, in some cases, complete protection against visual leakage. This unexpected result highlights a gap between theoretical guarantees and empirical robustness, suggesting that differential privacy may provide practical utility even in regimes beyond its conventional bounds.

4. Toward a More Practical Interpretation of DP: This analysis is not aimed at certifying strict compliance with DP deployment standards, but rather at empirically exploring the relationship between theoretical privacy guarantees and observable visual leakage. The results highlight the limitations of interpreting ϵ in isolation and advocate for combining perceptual metrics like LPIPS with (ϵ, δ) to assess practical privacy risk. In vision tasks—where even small visual features can carry sensitive information—such dual evaluation is critical for designing effective privacy-preserving learning systems.

6.3.5 Comparison with existing works

DLG introduced by [75], exposes vulnerabilities in FL by reconstructing client data from shared gradients. Follow-up studies like iDLG improved reconstruction accuracy, revealing that gradient information often retains sensitive data. These findings underscore the importance of protecting gradients using privacy-preserving mechanisms like DP or encryption. However, the effectiveness of these defenses against gradient-based attacks is still being actively studied, especially in multi-client FL scenarios.

Scale-MIA [76] proposes a latent space inversion attack designed to reconstruct training data from aggregated model updates, even under secure aggregation. Although the method claims to be privacy-preserving, the released code relies on full access to raw training data to simulate the attack, which contradicts the fundamental assumptions of

FL. Additionally, although the paper asserts that only partial data is leaked, the quality of reconstructions degrades sharply when less data is available, indicating that its effectiveness is tightly coupled to data volume and accessibility. In contrast, the original DLG attack works by minimizing the distance between real and dummy gradients, enabling reconstruction in centralized settings [75]. However, when applied to FL, DLG fails because gradients are aggregated across clients, breaking the direct gradient-to-input mapping and making optimization ineffective. The EPAFL [77] framework reinforces this by showing that traditional DLG-based attacks are computationally inefficient and ineffective under realistic settings—particularly when client-level gradients are not directly accessible. EPAFL does not consider fully decentralized gradient aggregation, limiting its applicability to practical federated environments where such reconstruction is significantly more constrained. LOKI [67] demonstrates large-scale data reconstruction in FL by bypassing aggregation via customized parameters, leaking up to 86% of client data in a single round. Since our work adopts a closed-form linear reconstruction approach inspired by LOKI, our results demonstrate that applying DP can effectively mitigate the attack. Specifically, even under relatively high privacy budgets (e.g., $\epsilon > 5$ for our specific experimental settings), the injected noise significantly distorts the gradients, rendering LOKI-style reconstruction ineffective.

Our method adopts a closed-form reconstruction approach similar to LOKI [67], but crucially demonstrates that can effectively break the attack. Unlike DLG [75] and iDLG, which fail under gradient aggregation, and Scale-MIA [76], which assumes unrealistic access to raw data, our method operates under practical federated settings without such assumptions. Even with relatively high privacy budgets, the injected DP noise distorts gradients enough to prevent meaningful reconstruction, addressing key vulnerabilities highlighted by previous attacks while validating DP as a viable defense.

7 Discussion

7.1 Impact of Reconstruction Attacks on FL Performance

The primary objective of this work was to explore how client model learning is affected by reconstruction attacks in an FL setting, both with and without DP. Our experiments demonstrate that the application of an imprint layer significantly degrades model performance. Similarly, the incorporation of DP, while essential for privacy preservation, also introduces noticeable utility loss.

7.2 Effectiveness of DP as a Defense Mechanism

To understand how DP influences the effectiveness of reconstruction attacks, we conducted experiments using varying values of key DP parameters—specifically the clipping threshold and the privacy budget (ϵ). These variations allowed us to observe the trade-offs between privacy and utility in both attack and non-attack scenarios, and to quantify how stricter privacy settings (i.e., smaller ϵ and tighter clipping thresholds) make it increasingly difficult for attackers to reconstruct input data from shared gradients. Our findings confirm that while stronger DP configurations offer better resistance against reconstruction, they also result in higher degradation in model performance.

Our results show that even with very large ϵ values (e.g., $\sim 10^{11}$), utility can degrade significantly due to the high noise multipliers used to implement differential privacy. While this might suggest that ϵ alone does not fully reflect the practical cost of DP, it is

also important to note that utility dropped even when the attack mechanism was incorporated, regardless of ϵ. This indicates that the interaction between DP, FL dynamics, and the attack itself may contribute to the observed degradation, not just the magnitude of ϵ. These findings reinforce the need for empirical evaluation, as the theoretical privacy guarantee does not always predict real-world performance.

7.3 Training Scenarios and Their Impact

7.3.1 Effect of the Malicious Imprint Layer

The injection of the imprint layer introduces architectural changes and additional computations, solely aimed at capturing activations for input reconstruction. This added overhead distorts the natural gradient flow, leading to suboptimal weight updates and reduced model accuracy. The imprint mechanism interferes with the task-oriented optimization process, causing slower convergence and degraded performance even in the absence of any privacy defense.

7.3.2 Combined Effect with DP

When DP is introduced alongside the imprint layer, the distortions are compounded. DP mechanisms—particularly gradient clipping and Gaussian noise addition—further obscure the learning signal. As both the imprint layer and DP modify the gradient landscape, the training becomes increasingly noisy and less effective, reducing the model's ability to generalize. This setup imposes a dual burden on the model: malicious architecture interference and randomization from DP, both negatively affecting convergence.

7.4 Accuracy vs. Privacy Trade-offs in DP-SGD

7.4.1 Gradient Clipping and Noise Injection

In DP-SGD, each gradient is first clipped to a fixed threshold bound C, and Gaussian noise is added with standard deviation proportional to $\sigma \times C$. The choice of C primarily controls how much each gradient is scaled before noise is added, but it does not directly determine the privacy budget ϵ. Instead, for a given noise multiplier σ, the privacy loss depends on the ratio of noise to sensitivity (where sensitivity is set by C). A smaller C clips gradients more aggressively, which can slow learning by limiting the useful signal. A larger C clips fewer gradients, preserving more signal, but may require proportionally more noise to maintain the same privacy guarantee — otherwise, the noise's protective effect is reduced. Importantly, if C is chosen excessively large relative to the natural gradient thresholds, gradients may not be clipped at all, and unnecessary noise is added without meaningful privacy benefit. Therefore, tuning C is about balancing gradient preservation and efficient use of the privacy budget, rather than directly controlling ϵ.

7.4.2 Interplay of ϵ and clipping threshold

Our experiments demonstrate that higher privacy budgets (ϵ), which correspond to less injected noise, lead to improved model accuracy but greater vulnerability to reconstruction attacks. Conversely, lower ϵ values (stronger noise) enhance privacy protection but significantly hinder optimization and model performance. The clipping threshold serves as a sensitivity control: larger clipping bounds allow more gradient signal to pass through, but they require proportionally more noise to maintain the same level of privacy — otherwise, the protective effect of the noise is diminished. This interplay between the clipping threshold and noise is critical; improper tuning of these parameters can result in either poor model utility or insufficient privacy protection.

7.5 Performance of the DLG Attack with Imprint Layers

The DLG attack, when paired with the imprint layer, proved to be highly effective in data recovery, especially when no privacy mechanism was in place. However, as DP was applied with progressively stricter configurations, the quality of the reconstructed images deteriorated significantly—highlighting DP's role as a practical defense against such attacks.

8 Conclusion

In this work, we investigated the vulnerability of FL systems to gradient-based reconstruction attacks, with a particular focus on the role of imprint layers and the effectiveness of DP in mitigating such attacks. Our findings reveal that while the imprint layer amplifies the success of reconstruction attacks, the application of DP—especially with carefully selected clipping thresholds and privacy budgets—significantly reduces the quality of recovered data, offering a viable defense mechanism.

However, this comes at the cost of reduced model performance, highlighting the inherent trade-off between privacy and utility. By varying DP parameters such as the clipping threshold and ϵ, we were able to demonstrate this balance and quantify the protective impact of DP against data leakage.

Overall, our study underscores the need for robust privacy-preserving techniques in FL, particularly in scenarios involving sensitive data. Future work should aim to enhance the scalability and generalizability of such methods while minimizing their impact on model accuracy. Exploring more adaptive, personalized, or hybrid privacy strategies may offer promising directions toward practical and secure FL deployments.

9 Challenges and Limitations

- **Computational Overhead:** Running gradient-based reconstruction attacks like DLG introduced substantial computational overhead, particularly when combined with differential privacy mechanisms and client aggregation across multiple federated rounds. This overhead made it impractical to conduct attack experiments at large scales, such as with full datasets per client. As a result, we limited attack experiments to the 64-image-per-client setting, which allowed us to keep computations tractable while still demonstrating the privacy risks and defense effectiveness. We used 64 images per client (vs. 500) in privacy–utility and LPIPS trade-off analyses because the high computational overhead of gradient-based attacks made large-scale experiments infeasible. This limitation also created an opportunity to study how data scarcity, DP noise, and attack interference collectively impact model utility, highlighting the compounded challenges of achieving strong privacy without severely degrading learning performance.

- **Interpreting Practical Privacy at High ϵ Values:** A key challenge in our experiments is interpreting the practical impact of high privacy budgets (e.g., $\epsilon = 100$ or above). While such values are typically considered weak from a theoretical differential privacy perspective, we observe that reconstruction quality already begins to degrade around this range. As ϵ decreases further, reconstructions become increasingly distorted or fail entirely. This suggests that meaningful visual leakage only becomes apparent at extremely large ϵ values, highlighting a disconnect between formal privacy guarantees and actual perceptual data leakage.

- **Imprint Layer Limitations:** Although useful for analyzing the impact of attacks,

the imprint layer may not generalize well across different network architectures or tasks, potentially limiting its broader applicability.

- **Static Privacy Settings:** The use of fixed DP parameters across all clients and training rounds does not reflect the flexibility of adaptive or personalized privacy mechanisms, which could better balance privacy and performance in practical deployments.

- **Scope and Generalizability:** The study was conducted under controlled conditions with a modest number of clients and simplified configurations. While sufficient for demonstrating key findings, future work should explore more realistic settings to validate scalability and robustness.

10 Future Direction

Building upon the findings of this study, several promising directions emerge for future research:

10.1 Scalability and Experimental Extensions

- **Client Scaling and Heterogeneity:** Increasing the number of clients and varying their data distributions can help assess the robustness of attacks and models under more realistic, heterogeneous FL settings.
- **Larger Models and Datasets:** Evaluating deeper architectures on more complex datasets can reveal whether increased model capacity and input diversity naturally reduce leakage effectiveness.
- **Extended Training with More Resources:** Running longer training on larger datasets using high-performance hardware would clarify how attack feasibility evolves as clients contribute more data and gradients over time.

10.2 Improving Model Utility under Privacy Constraints

- **Tighter Privacy Accounting:** Using advanced frameworks like Rényi DP allows for stronger utility by enabling tighter control over cumulative privacy loss across rounds.
- **Adaptive Privacy Mechanisms:** Dynamically adjusting noise levels or gradient clipping based on training sensitivity can improve the trade-off between privacy and

accuracy.

- **Efficient Communication and Sharing:** Techniques such as knowledge distillation and selective update sharing can reduce leakage risk while maintaining performance.
- **Personalization and Local Adaptation:** Personalized FL approaches, such as partial model sharing or meta-learning, help sustain client-level accuracy even under strict privacy budgets.

10.3 Developing More Realistic and Robust Attack Methods

- **Enhanced Gradient Inversion:** Strengthening existing attacks (e.g., DLG) through multi-step optimization, better priors, or label recovery can expose deeper vulnerabilities.
- **Generative and Black-box Attacks:** Leveraging GANs or diffusion models, and designing attacks that require limited access to model internals, will better reflect realistic threat settings.
- **Cross-client and Temporal Inference:** Exploiting aggregated gradients across clients or training rounds can reveal long-term privacy risks often overlooked by static analyses.

10.4 Designing Stronger Defense Mechanisms

- **Gradient Obfuscation and Aggregation:** Techniques like gradient masking, selective sharing, and secure aggregation (e.g., MPC, homomorphic encryption) reduce exposure to reconstruction.

- **Noise-Injection Strategies:** Using structured or data-aware noise (e.g., dropout-based or adaptive noise) can preserve learning while hindering attacks more effectively than naïve Gaussian noise.
- **Adversarial and Information-Theoretic Defenses:** Incorporating adversarial training or limiting mutual information between gradients and inputs can offer stronger, model-level protection.
- **Formal Privacy Guarantees:** Pursuing certified or verified privacy bounds ensures robustness against adaptive attacks beyond empirical defenses.

References

[1] C. Zhang, Y. Xie, H. Bai, B. Yu, W. Li, and Y. Gao, "A survey on federated learning", *Knowledge-Based Systems*, vol. 216, p. 106 775, 2021.

[2] B. Zhao, K. R. Mopuri, and H. Bilen, "Idlg: Improved deep leakage from gradients", *arXiv preprint arXiv:2004.10374*, 2020.

[3] C. Dwork, A. Roth, *et al.*, "The algorithmic foundations of differential privacy", *Foundations and Trends® in Theoretical Computer Science*, vol. 9, no. 3–4, pp. 211–407, 2014.

[4] C. Dupuy, R. Arava, R. Gupta, and A. Rumshisky, "An efficient dp-sgd mechanism for large scale nlu models", in *ICASSP 2022-2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, IEEE, 2022, pp. 4118–4122.

[5] M. Abadi, A. Chu, I. Goodfellow, *et al.*, "Deep learning with differential privacy", in *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, ACM, 2016, pp. 308–318. DOI: 10.1145/2976749.2978318.

[6] R. Zhang, P. Isola, A. A. Efros, E. Shechtman, and O. Wang, "The unreasonable effectiveness of deep features as a perceptual metric", in *CVPR*, 2018.

[7] L. Zhu, Z. Liu, and S. Han, "Deep leakage from gradients", in *Advances in Neural Information Processing Systems (NeurIPS)*, vol. 32, 2019, pp. 14 747–14 756.

[8] J. Geiping, H. Bauermeister, H. Drozdovski, and M. Moeller, "Inverting gradients—how easy is it to break privacy in federated learning?", *Advances in Neural Information Processing Systems (NeurIPS)*, vol. 33, pp. 16 937–16 947, 2020.

[9] N. Rieke, J. Hancox, W. Li, *et al.*, "The future of digital health with federated learning", *npj Digital Medicine*, vol. 3, no. 1, pp. 1–7, 2020.

[10] X. Hao, L. Huang, L. Wang, Y. Xu, and S. Xu, "Gradient inversion with generative image prior", in *International Conference on Learning Representations (ICLR)*, 2023.

[11] E. Bagdasaryan, A. Veit, Y. Hua, D. Estrin, and V. Shmatikov, "How to backdoor federated learning", in *Proceedings of the 23rd International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2020, pp. 2938–2948. arXiv: 1807.00459. [Online]. Available: https://arxiv.org/abs/1807.00459.

[12] H. B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas, "Communication-efficient learning of deep networks from decentralized data", *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics (AISTATS)*, pp. 1273–1282, 2017. arXiv: 1602.05629. [Online]. Available: https://arxiv.org/abs/1602.05629.

[13] R. Shokri, M. Stronati, C. Song, and V. Shmatikov, "Membership inference attacks against machine learning models", in *2017 IEEE Symposium on Security and Privacy (SP)*, IEEE, 2017, pp. 3–18.

[14] I. Mironov, "Rényi differential privacy", in *2017 IEEE 30th Computer Security Foundations Symposium (CSF)*, IEEE, Aug. 2017, pp. 263–275. DOI: 10.1109/csf.2017.11. [Online]. Available: http://dx.doi.org/10.1109/CSF.2017.11.

[15] T. Li, A. K. Sahu, A. Talwalkar, and V. Smith, "Federated learning: Challenges, methods, and future directions", *IEEE Signal Processing Magazine*, vol. 37, no. 3, pp. 50–60, 2020. DOI: 10.1109/MSP.2020.2975749.

[16] H. B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. Y. Arcas, "Communication-efficient learning of deep networks from decentralized data", *arXiv preprint arXiv:1602.05629*, 2016. [Online]. Available: http://arxiv.org/abs/1602.05629.

[17] J. Konecný, H. B. McMahan, F. X. Yu, P. Richtárik, A. T. Suresh, and D. Bacon, "Federated learning: Strategies for improving communication efficiency", *arXiv preprint arXiv:1610.05492*, 2018. [Online]. Available: http://arxiv.org/abs/1610.05492.

[18] L. He, A. Bian, and M. Jaggi, "Cola: Decentralized linear learning", *arXiv preprint arXiv:1808.04883*, 2018. [Online]. Available: http://arxiv.org/abs/1808.04883.

[19] K. Bonawitz, V. Ivanov, B. Kreuter, *et al.*, "Practical secure aggregation for privacy-preserving machine learning", *ACM SIGSAC Conference on Computer and Communications Security*, pp. 1175–1191, 2017. DOI: 10.1145/3133956.3133982.

[20] R. C. Geyer, T. Klein, and M. Nabi, "Differentially private federated learning: A client level perspective", *arXiv preprint arXiv:1712.07557*, 2017. [Online]. Available: https://arxiv.org/abs/1712.07557.

[21] K. B. et al., "Towards federated learning at scale: System design", *arXiv preprint arXiv:1902.01046*, 2019. [Online]. Available: http://arxiv.org/abs/1902.01046.

[22] A. L. et al., "Decentralized bayesian learning over graphs", *arXiv preprint arXiv:1905.10466*, 2019. [Online]. Available: http://arxiv.org/abs/1905.10466.

[23] Q. Y. et al., "Federated machine learning: Concept and applications", *ACM Transactions on Intelligent Systems and Technology (TIST)*, vol. 10, no. 2, pp. 1–19, 2019.

[24] Y. L. et al., "Federated transfer learning for cross-domain image classification", *arXiv preprint arXiv:2003.13945*, 2020. [Online]. Available: http://arxiv.org/abs/2003.13945.

[25] S. K. et al., "Scaffold: Stochastic controlled averaging for federated learning", *arXiv preprint arXiv:1910.06378*, 2020. [Online]. Available: http://arxiv.org/abs/1910.06378.

[26] S. R. et al., "Adaptive federated optimization", *arXiv preprint arXiv:2003.00295*, 2020. [Online]. Available: http://arxiv.org/abs/2003.00295.

[27] H. X. et al., "Verifynet: Secure and private federated learning", *arXiv preprint arXiv:1912.04878*, 2019. [Online]. Available: http://arxiv.org/abs/1912.04878.

[28] T. S. B. et al., "Federated learning for predicting clinical outcomes in patients with covid-19", *Scientific Reports*, vol. 8, no. 1, pp. 1–13, 2018.

[29] R. H. et al., "Federated learning for gboard: Collaborative on-device machine learning", *arXiv preprint arXiv:1811.03604*, 2018. [Online]. Available: http://arxiv.org/abs/1811.03604.

[30] M. A.-U. et al., "Federated learning for personalization: Applications and challenges", *arXiv preprint arXiv:1910.14481*, 2019. [Online]. Available: http://arxiv.org/abs/1910.14481.

[31] S. S. et al., "Federated learning for autonomous vehicles: Privacy and communication trade-offs", *IEEE Internet of Things Journal*, 2020.

[32] L. Zhu, Z. Liu, and S. Han, "Deep leakage from gradients", in *Advances in Neural Information Processing Systems (NeurIPS)*, 2020, pp. 14 747–14 756. arXiv: 1906.08935. [Online]. Available: https://arxiv.org/abs/1906.08935.

[33] M. Nasr, R. Shokri, and A. Houmansadr, "Comprehensive privacy analysis of deep learning: Passive and active white-box inference attacks against centralized and federated learning", in *2019 IEEE Symposium on Security and Privacy (SP)*, IEEE, 2019, pp. 739–753.

[34] A. N. Bhagoji, S. Chakraborty, P. Mittal, and S. Calo, "Analyzing federated learning through an adversarial lens", in *International Conference on Machine Learning (ICML)*, PMLR, 2019, pp. 634–643.

[35] C. Fung, C. J. Yoon, and I. Beschastnikh, "Limitations of model poisoning defenses in federated learning", in *Proceedings of the 23rd International Symposium on Research in Attacks, Intrusions and Defenses (RAID)*, 2020, pp. 1–16.

[36] P. Blanchard, R. Guerraoui, and J. Stainer, "Machine learning with adversaries: Byzantine tolerant gradient descent", in *Advances in Neural Information Processing Systems (NeurIPS)*, 2017, pp. 119–129. arXiv: 1703.02757. [Online]. Available: https://arxiv.org/abs/1703.02757.

[37] E. M. El Mhamdi, R. Guerraoui, and S. Rouault, "Hidden: Hiding malicious updates in federated learning", in *Advances in Neural Information Processing Systems (NeurIPS)*, vol. 33, 2020, pp. 12 667–12 679.

[38] C. Xie, O. Koyejo, and I. Gupta, "Fall of empires: Breaking federated learning through malicious clients", in *29th USENIX Security Symposium (USENIX Security)*, 2020, pp. 485–500.

[39] T. Orekondy, B. Schiele, and M. Fritz, "Knockoff nets: Stealing functionality of black-box models", in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019, pp. 4954–4963.

[40] M. Jagielski, N. Carlini, D. Berthelot, A. Kurakin, and N. Papernot, "High accuracy and high fidelity extraction of neural networks", in *29th USENIX Security Symposium (USENIX Security)*, 2020, pp. 1345–1362.

[41] I. J. Goodfellow, J. Shlens, and C. Szegedy, "Explaining and harnessing adversarial examples", in *International Conference on Learning Representations (ICLR)*, 2015.

[42] A. Kurakin, I. Goodfellow, and S. Bengio, "Adversarial examples in the physical world", *arXiv preprint arXiv:1607.02533*, 2016.

[43] N. Papernot, P. McDaniel, and I. Goodfellow, "Practical black-box attacks against machine learning", in *Proceedings of the 2017 ACM on Asia Conference on Computer and Communications Security (AsiaCCS)*, 2017, pp. 506–519.

[44] S. Truex, L. Liu, K.-H. Chow, M. E. Gursoy, and W. Wei, "Ldp-fed: Federated learning with local differential privacy", in *Proceedings of the Third ACM International Workshop on Edge Systems, Analytics and Networking*, New York, NY, USA: ACM, 2020, pp. 61–66. DOI: 10.1145/3378679.3394533. [Online]. Available: https://doi.org/10.1145/3378679.3394533.

[45] A. Triastcyn and B. Faltings, "Federated learning with bayesian differential privacy", in *2019 IEEE International Conference on Big Data (Big Data)*, IEEE, 2019, pp. 2587–2596. DOI: 10.1109/BigData47090.2019.9005465.

[46] O. Choudhury, A. Gkoulalas-Divanis, T. Salonidis, *et al.*, *Differential privacy-enabled federated learning for sensitive health data*, 2020. arXiv: 1910.02578 [cs.LG]. [Online]. Available: https://arxiv.org/abs/1910.02578.

[47] M. Naseri, J. Hayes, and E. De Cristofaro, *Local and central differential privacy for robustness and privacy in federated learning*, 2022. arXiv: 2009.03561 [cs.CR]. [Online]. Available: https://arxiv.org/abs/2009.03561.

[48] L. Sun, J. Qian, and X. Chen, *Ldp-fl: Practical private aggregation in federated learning with local differential privacy*, 2021. arXiv: 2007.15789 [cs.CR]. [Online]. Available: https://arxiv.org/abs/2007.15789.

[49] R. Hu, Y. Guo, H. Li, Q. Pei, and Y. Gong, "Personalized federated learning with differential privacy", *IEEE Internet of Things Journal*, vol. 7, no. 10, pp. 9530–9539, 2020. DOI: 10.1109/JIOT.2020.2991416.

[50] R. C. Geyer, T. Klein, and M. Nabi, *Differentially private federated learning: A client level perspective*, 2018. arXiv: 1712.07557 [cs.CR]. [Online]. Available: https://arxiv.org/abs/1712.07557.

[51] I. Mironov, "Rényi differential privacy", in *2017 IEEE 30th Computer Security Foundations Symposium (CSF)*, IEEE, 2017, pp. 263–275. DOI: 10.1109/CSF.2017.11.

[52] Y. Wang and X. Wu, "Differentially private data clustering with adaptive distance protection", *IEEE Transactions on Knowledge and Data Engineering*, 2019.

[53] J. Lee and C. Clifton, "Towards noise-tolerant frequent itemset mining under differential privacy", in *Proceedings of the 17th ACM SIGKDD*, 2011.

[54] Y. e. a. Tang, "Dp-lv: Practical and privacy-preserving latent tree model learning via differential privacy", *IEEE Transactions on Knowledge and Data Engineering*, 2020.

[55] R. e. a. Chen, "Differentially private sequential data publishing via variable-length n-grams", *Proceedings of the VLDB Endowment*, 2012.

[56] D. e. a. Cyffers, "Decentralized and differentially private learning", *arXiv preprint arXiv:2011.06860*, 2020.

[57] R. e. a. Zhao, "Local differential privacy for federated learning: Algorithms and performance analysis", in *Proceedings of the 2020 IEEE International Conference on Communications (ICC)*, 2020.

[58] Y. e. a. Wu, "Fedml: A research library and benchmark for federated machine learning", *arXiv preprint arXiv:2007.13518*, 2021.

[59] M. e. a. Gong, "Preserving privacy in federated learning with differential privacy and homomorphic encryption", in *Proceedings of the AAAI Conference on Artificial Intelligence*, 2020.

[60] N. e. a. Tran, "Secure decentralized training framework with differential privacy", *IEEE Transactions on Dependable and Secure Computing*, 2021.

[61] X. e. a. Li, "Chain-ppfl: A privacy-preserving federated learning scheme for iot devices", *IEEE Internet of Things Journal*, 2020.

[62] J. e. a. Pan, "Differentially private regression for discrete data with application to the analysis of covid-19 data", in *Proceedings of the 26th International Conference on Information and Knowledge Management (CIKM)*, 2017.

[63] J. e. a. Zhang, "Functional mechanism: Regression analysis under differential privacy", *Proceedings of the VLDB Endowment*, 2012.

[64] J. e. a. Xu, "Differentially private model publishing for deep learning", in *Proceedings of the 2017 IEEE Symposium on Security and Privacy*, 2017.

[65] M. e. a. Fang, "Privacy protection for deep learning with gradient perturbation in federated learning", in *Proceedings of the 2020 ACM Workshop on Privacy in the Electronic Society (WPES)*, 2020.

[66] B. Jayaraman and D. Evans, "Evaluating differential privacy for deep learning models", in *Proceedings of the 31st USENIX Security Symposium*, 2019.

[67] J. C. Zhao, A. Sharma, A. R. Elkordy, Y. H. Ezzeldin, S. Avestimehr, and S. Bagchi, *Loki: Large-scale data reconstruction attack against federated learning through model manipulation*, 2023. arXiv: 2303.12233 [cs.LG]. [Online]. Available: https://arxiv.org/abs/2303.12233.

[68] H. Qi, M. Brown, and D. G. Lowe, *Low-shot learning with imprinted weights*, 2018. arXiv: 1712.07136 [cs.CV]. [Online]. Available: https://arxiv.org/abs/1712.07136.

[69] H. Zhang, J. Hong, Y. Deng, M. Mahdavi, and J. Zhou, "Understanding deep gradient leakage via inversion influence functions", *arXiv preprint arXiv:2309.13016*, 2023.

[70] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition", *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[71] J. Yang, Y. Shi, H. Sun, *et al.*, "Medmnist v2: A large-scale lightweight benchmark for 2d and 3d biomedical image classification", *Scientific Data*, vol. 8, no. 1, p. 111, 2021.

[72] A. Krizhevsky and G. Hinton, "Learning multiple layers of features from tiny images", *Technical Report, University of Toronto*, 2009.

[73] A. Paszke, S. Gross, F. Massa, *et al.*, "Pytorch: An imperative style, high-performance deep learning library", in *Advances in Neural Information Processing Systems (NeurIPS)*, 2019, pp. 8024–8035.

[74] R. Zhang, P. Isola, A. A. Efros, E. Shechtman, and O. Wang, "The unreasonable effectiveness of deep features as a perceptual metric", in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018, pp. 586–595.

[75] L. Zhu, Z. Liu, and S. Han, "Deep leakage from gradients", in *Advances in Neural Information Processing Systems (NeurIPS)*, 2019.

[76] S. Shi, N. Wang, Y. Xiao, *et al.*, "Scale-mia: A scalable model inversion attack against secure federated learning via latent space reconstruction", in *Network and Distributed System Security (NDSS) Symposium*, 2025. [Online]. Available: `https://github.com/unknown123489/Scale-MIA`.

[77] N. Tabassum, K.-H. Chow, X. Wang, W. Zhang, and Y. Wu, "On the efficiency of privacy attacks in federated learning", in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, 2024.

11 Supplementary materials

11.1 Effect of varying Parameters: Clipping Threshold and Privacy Budget on Reconstruction Quality over additional datasets BreastMNIST, DermaMNIST, RetinaMNIST, CIFAR10

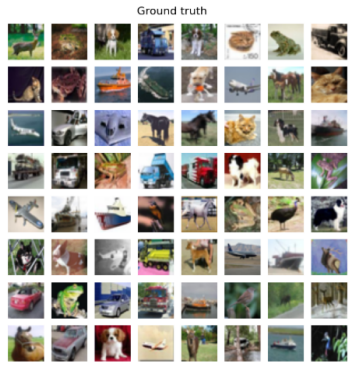
Dataset	Epsilon	Clipping Threshold	LPIPS	Comments
BreastMNIST	8.31×10^9	10,000	0.4096	**Moderate privacy — partial leakage visible**
	8.31×10^{12}	10,000	5.70e-05	**Very low privacy — reconstructions highly accurate**
DermaMNIST	8.31×10^9	10,000	0.5007	**High privacy — reconstructions heavily distorted**
	8.31×10^{12}	10,000	0.0010	**Very low privacy — reconstructions highly accurate**
RetinaMNIST	8.31×10^9	10,000	0.5484	**High privacy — reconstructions heavily distorted**
	8.31×10^{12}	10,000	0.0001	**Very low privacy — reconstructions highly accurate**

Table 11.1: LPIPS leakage across various datasets under varying privacy budgets and fixed clipping threshold, ordered by increasing ε.

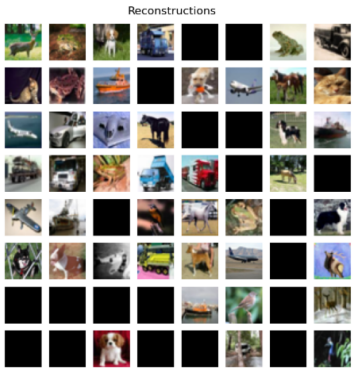
Epsilon	**Clipping Threshold**	**LPIPS**	**Comments**
8.31×10^9	10^4	1.26×10^{-5}	**Very low privacy — reconstructions highly accurate**
8.31×10^6	10^4	5.97×10^{-1}	**Low privacy — reconstructions largely intact**
8.31×10^6	10	5.44×10^{-5}	**Low privacy — reconstructions largely intact**
8.31×10^3	10^4	6.65×10^{-1}	**High privacy — reconstructions heavily distorted**
8.31×10^3	10	6.42×10^{-1}	**High privacy — reconstructions heavily distorted**

Table 11.2: LPIPS leakage on **CIFAR-10** under varying clipping thresholds, ordered by increasing ε.

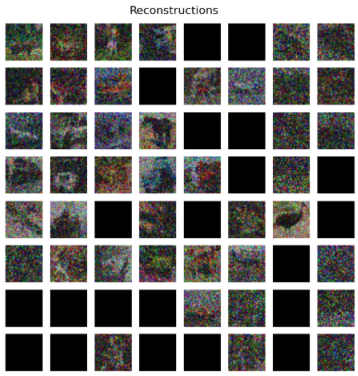
From Table 11.2 and Table 11.1, we observe a consistent trend across BreastMNIST, DermaMNIST, RetinaMNIST, and CIFAR-10: higher privacy budgets (larger ϵ) lead to significantly improved reconstructions, as reflected by lower LPIPS scores. At low ϵ (e.g., 10^3–10^6), reconstructions are heavily distorted, while at high ϵ (e.g., 10^9–10^{12}), reconstructed images closely resemble the originals. Full reconstruction samples and detailed analysis for each dataset are provided in the main experimental section.



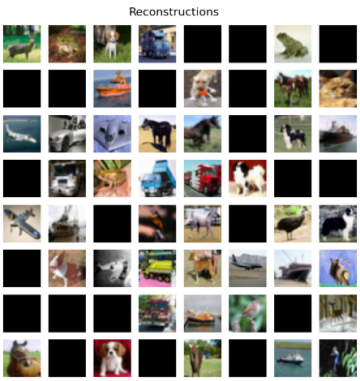
(a) Ground Truth (CIFAR-10)



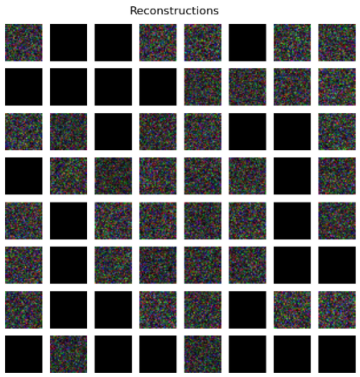
(b) C = 10^4, $\epsilon = 8.31 \times 10^9$



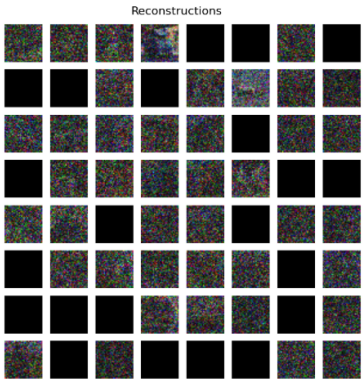
(c) C = 10^4, $\epsilon = 8.31 \times 10^6$



(d) C = 10, $\epsilon = 8.31 \times 10^6$

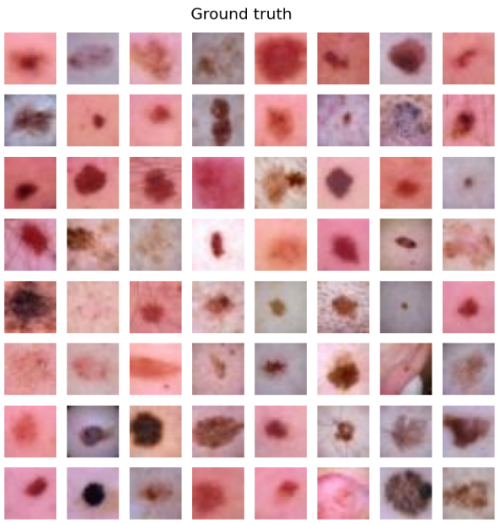


(e) C = 10^4, $\epsilon = 8.31 \times 10^3$

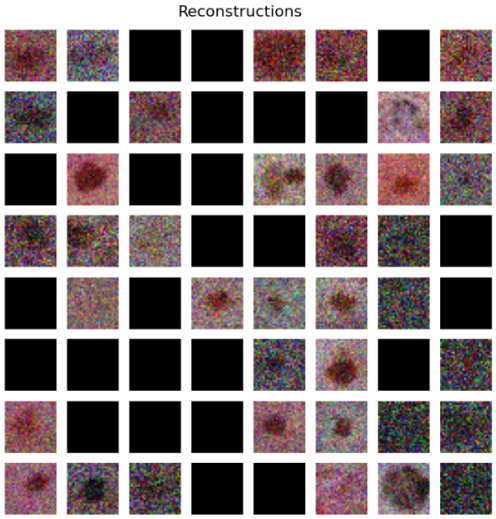


(f) C = 10, $\epsilon = 8.31 \times 10^3$

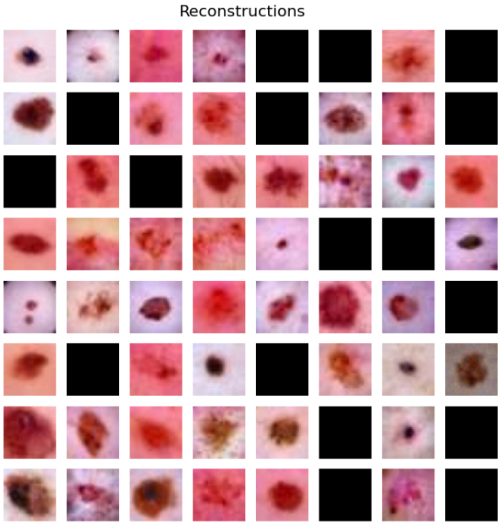
Figure 11.1: LPIPS reconstructions on **CIFAR-10** for a random client for varying ϵ and varying clipping thresholds arranged by decreasing privacy budget ϵ.



(a) Ground truth (DermaMNIST)

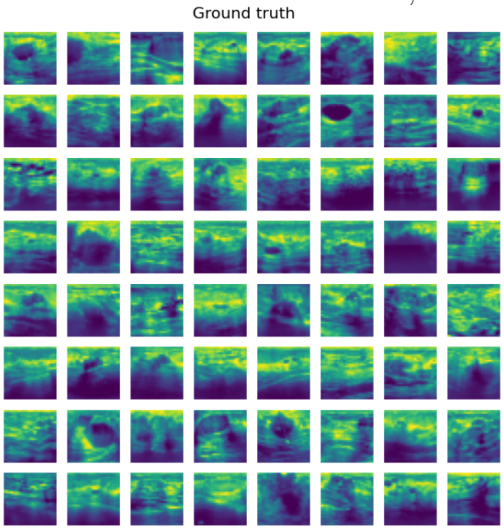


(b) C = 10000, $\epsilon = 8.31 \times 10^6$

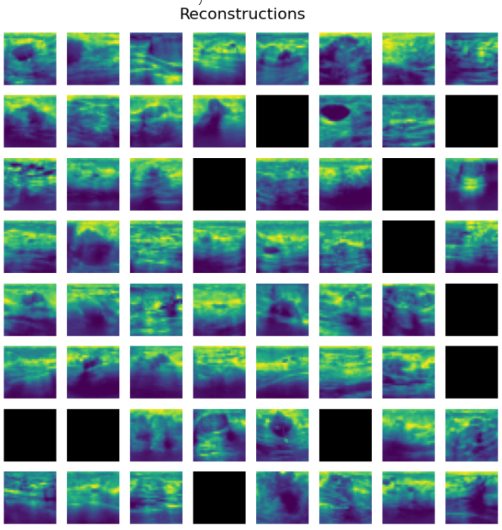


(c) C = 10000, $\epsilon = 8.31 \times 10^9$

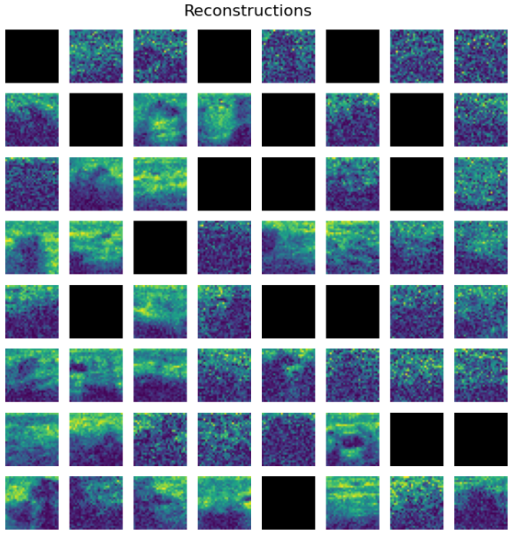
Figure 11.2: LPIPS reconstructions on **DermaMNIST** for a random client for varying ϵ arranged by decreasing privacy budget ϵ.



(a) Ground truth (BreastMNIST)

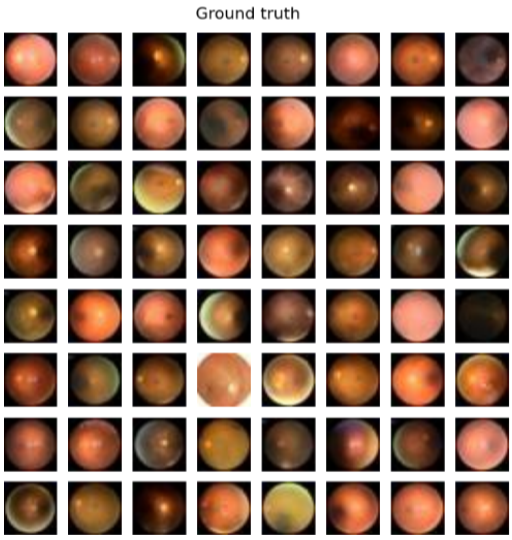


(b) C = 10000, $\epsilon = 8.31 \times 10^{11}$

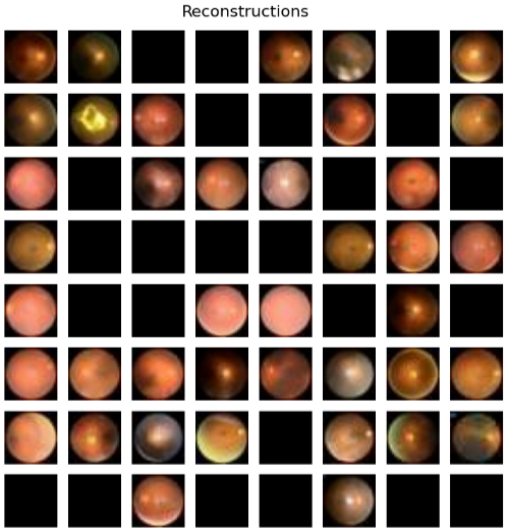


(c) C = 10000, $\epsilon = 8.31 \times 10^{8}$

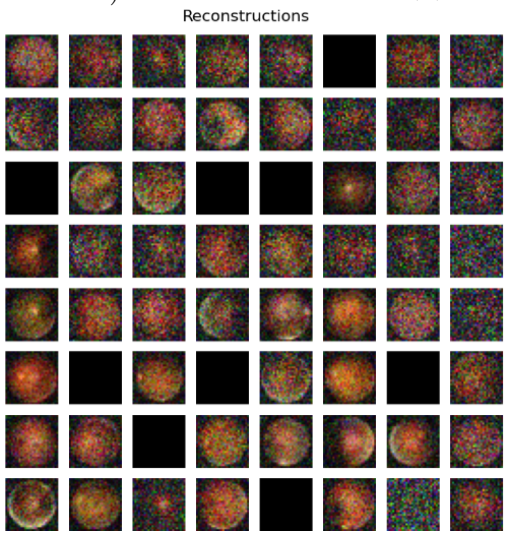
Figure 11.3: LPIPS reconstructions on **BreastMNIST** for a random client for varying ϵ arranged by decreasing privacy budget ϵ.



(a) Ground truth (RetinaMNIST)



(b) C = 10000, $\epsilon = 8.31 \times 10^{11}$



(c) C = 10000, $\epsilon = 8.31 \times 10^{8}$

Figure 11.4: LPIPS reconstructions on **RetinaMNIST** for a random client for varying ϵ arranged by decreasing privacy budget ϵ.