Scalable and Private Federated Learning Using Distributed Differential Privacy and Secure Aggregation

Wenjing Wei, Farid Nait-Abdesselam and Alla Jammine

Abstract

This article presents DDP-SA, a scalable privacy-preserving federated learning framework that jointly leverages client-side local differential privacy (LDP) and full-threshold additive secret sharing (ASS) for secure aggregation. Unlike existing methods that rely solely on differential privacy or on secure multi-party computation (MPC), DDP-SA integrates both techniques to deliver stronger end-to-end privacy guarantees while remaining computationally practical. The framework introduces a two-stage protection mechanism: clients first perturb their local gradients with calibrated Laplace noise, then decompose the noisy gradients into additive secret shares that are distributed across multiple intermediate servers. This design ensures that (i) no single compromised server or communication channel can reveal any information about individual client updates, and (ii) the parameter server reconstructs only the aggregated noisy gradient, never any client-specific contribution. Extensive experiments show that DDP-SA achieves substantially higher model accuracy than standalone LDP while providing stronger privacy protection than MPC-only approaches. The proposed framework scales linearly with the number of participants and offers a practical, privacy-preserving solution for federated learning applications with controllable computational and communication overhead.

Index Terms

Federated Learning, Differential Privacy, Secure Aggregation, Secret Sharing, Privacy-Preserving Machine Learning

I. Introduction

MACHINE learning (ML) plays a central role in modern society and is widely adopted across numerous industries. It underpins applications in computer vision, speech recognition, natural language processing, and many other domains that significantly benefit users and organizations. Traditionally, ML systems require raw data to be uploaded from users' devices to a central server for model training. However, this centralized paradigm raises substantial privacy concerns, as it exposes sensitive user information to potential leakage [1].

To address these privacy and security challenges, *federated learning* (FL) has emerged as a promising distributed ML framework [2]. FL enables multiple clients (e.g., mobile devices) to collaboratively train a global model by transmitting only locally computed updates, such as gradients or model parameters, while keeping raw data on-device. Although this paradigm provides an initial layer of privacy protection, recent studies have demonstrated that FL remains vulnerable to privacy leakage, particularly through inference attacks that exploit shared updates [3], [4].

These privacy risks predominantly arise from *privacy inference attacks*, in which adversaries analyze shared updates to infer sensitive attributes of users' data [5], [6]. Existing defenses, most notably differential privacy (DP) and secure multi-party computation (MPC), provide partial mitigation but exhibit notable limitations [7]. Differential privacy obscures client updates by adding randomized noise, but stronger privacy requires larger noise magnitudes that significantly degrade model performance. In contrast, MPC-based secure aggregation protocols cryptographically ensure that the server learns only aggregated results [8]–[12], yet they often incur substantial computational and communication overhead.

Motivated by these challenges and the limitations of using DP or MPC alone, we propose a novel privacy-preserving federated learning framework, *Distributed Differential Privacy via Secure Aggregation* (DDP-SA). DDP-SA integrates client-side local differential privacy (LDP) with full-threshold additive secret sharing (ASS), resulting in a principled hybrid mechanism that achieves formal (ϵ, δ) differential privacy at the client level while cryptographically hiding individual updates from both the server and all communication paths. As established in Theorem V-A, the combined mechanism retains its DP guarantee due to post-processing invariance while ensuring that no single client's contribution is ever exposed.

To support scalability, we design a multi-server architecture consisting of n clients and m intermediate servers. This architecture achieves linear communication complexity and generalizes naturally to arbitrary m, extending beyond commonly studied illustrative cases such as $m = 3$. Within this architecture, clients first perturb their gradients with calibrated Laplace noise, then encode the noisy gradients into additive secret shares that are distributed among the intermediate servers. These servers aggregate the received shares and forward only the combined result to the parameter server (PS), which reconstructs the aggregated noisy gradient and updates the global model.

W. Wei, F. Nait-Abdesselam and A. Jammine are with Université Paris Cité, Paris, France (e-mail: wenjing.wei@etu.u-paris.fr, farid.nait-abdesselam@u-paris.fr, and alla.jammine@u-paris.fr).

Another key contribution of our work is a multi-round privacy analysis based on advanced composition. We provide practical guidance for allocating privacy budgets in long-running FL scenarios, which enables system designers to manage privacy loss across many training rounds. Additionally, we conduct a detailed component-wise breakdown of computational and communication costs, separating the overhead introduced by the LDP and MPC components. Our analysis highlights the specific sources of system overhead and demonstrates that secure aggregation remains practical even at a large scale. In the entire FL process, clients never reveal raw data or unprotected gradients, providing resilience against a broad class of privacy inference attacks. Experimental results show that DDP-SA offers stronger privacy guarantees than either LDP or MPC alone, while maintaining acceptable accuracy and efficiency. Moreover, DDP-SA scales effectively to large numbers of clients and servers.

The structure of this paper is organized as follows. Section I introduces the research background, motivation, and main contributions. Section II reviews related work. Section III presents preliminaries. Section IV describes the system overview and details of the DDP-SA framework. Section V provides privacy analysis. Section VI presents experimental results and performance evaluation. Section VII concludes the paper.

II. Related Work

In recent years, FL has emerged as a powerful distributed machine learning paradigm that allows multiple participants to collaboratively train a global model without directly sharing their raw data. While FL offers promising privacy benefits compared to traditional centralized training, it remains vulnerable to a range of privacy inference attacks that can compromise sensitive client information. To address these risks, a growing body of research has focused on integrating advanced privacy-preserving techniques into FL systems, including differential privacy, secure multi-party computation, and homomorphic encryption (HE).

This section provides a comprehensive overview of the current landscape in privacy-preserving federated learning. We begin in Section II-A by categorizing various privacy inference attacks that threaten FL systems and highlighting their mechanisms and impact. We then explore the application of differential privacy in FL and examine its practical implementations and limitations. In addition, we discuss the role of secure computation techniques, particularly MPC and homomorphic encryption, in safeguarding model updates. The section further reviews recent hybrid approaches that combine DP and MPC to balance privacy, efficiency, and model accuracy.

Through this survey of the state of the art, we aim to contextualize the design and motivation behind our proposed privacy-preserving FL framework introduced in the subsequent sections.

A. Privacy Inference Attacks in FL

Federated learning, as a distributed machine learning paradigm, can effectively address the privacy challenges faced by traditional centralized machine learning and has been widely adopted in areas involving users' sensitive data, such as healthcare, finance, and the Internet of Things (IoT). The federated averaging (FedAvg) algorithm is the core algorithm of FL. It includes both the model averaging algorithm and the gradient averaging algorithm [2], [13]. In the model averaging algorithm, users train their local models using stochastic gradient descent (SGD) and send the model parameters to the parameter server (PS) for aggregation. In the gradient averaging algorithm, users upload their gradient parameters to the PS for aggregation. The model averaging algorithm typically requires fewer communication rounds to reach convergence compared to the gradient averaging algorithm.

Even though FL provides some privacy protection compared to traditional machine learning, recent studies [4]–[6], [14]–[19] have shown that attackers can still obtain private information about users by analyzing exchanged model parameters or gradients. Melis et al. [14] revealed an attack strategy that exploits unintended feature leakage from gradients shared during collaborative learning, allowing adversaries to infer sensitive attributes about participants' data without direct access to it. Fredrikson et al. [15] presented model inversion attacks that use confidence information revealed by machine learning models to reconstruct sensitive input data, highlighting the privacy risks associated with exposing high-confidence predictions. In [16], the authors demonstrated an attack in which adversaries can recover original training data from shared model gradients during the training process in deep learning, underscoring the significant privacy risks of gradient sharing in collaborative learning environments. Hitaj et al. [18] showed that adversaries can use generative adversarial networks (GANs) to reconstruct private training data of other participants by exploiting shared model updates in collaborative deep learning settings.

B. Differential Privacy in FL

With increased research interest in differential privacy, many researchers have applied various forms of DP, including central differential privacy, local differential privacy, and distributed differential privacy, to the federated learning process to defend against privacy inference attacks [20]–[32]. Table I summarizes over 70 recent articles on differentially private FL. Hu et al. [20] introduced personalized federated learning with differential privacy, combining personalized model training with DP to improve data privacy and model performance. However, this approach may increase computational complexity and reduce model accuracy due to the noise added for privacy preservation. Geyer et al. [21] proposed a client-level differentially private

federated learning method that integrates DP directly into the FL process, although the added noise can degrade learning performance and reduce accuracy. Wei et al. [22] developed federated learning algorithms incorporating differential privacy to safeguard user data privacy while enabling collaborative model training. A limitation of these algorithms is the inherent privacy-accuracy trade-off, since higher privacy levels typically reduce model accuracy.

Liu et al. [23] introduced FedSel, a method combining federated SGD with local differential privacy and top-k dimension selection, improving data privacy and training efficiency. However, selecting the top-k dimensions may lead to information loss and reduced accuracy. Zhao et al. [24] applied local differential privacy to protect IoT device data in FL while collectively improving model learning, although higher privacy levels can significantly impact learning effectiveness and convergence. Zheng et al. [25] introduced federated f-differential privacy, a flexible DP framework tailored for FL, but its implementation requires careful and sometimes complex privacy parameter selection. Seif et al. [26] proposed wireless federated learning combined with local differential privacy for secure user data protection in distributed training over wireless networks. However, increased noise and unreliable wireless transmission can reduce the accuracy of the federated model. All of the above DP-based schemes share a common limitation, since adding random noise to gradients or parameters inevitably decreases the accuracy of the federated learning model.

C. Secure Multi-party Computation and Homomorphic Encryption in FL

Secure multi-party computation and homomorphic encryption are widely used cryptographic techniques for defending against privacy inference attacks in FL [8]–[12], [99]–[102]. Li et al. [99] proposed a privacy-preserving FL framework employing chained MPC to protect data privacy during collaborative learning among IoT devices. However, chained MPC requires complex cryptographic operations and introduces significant computational and communication overhead, limiting scalability in large IoT networks. Bonawitz et al. [8] introduced a practical secure aggregation protocol for FL, enabling a server to compute the sum of client-updated model parameters without accessing individual contributions. Nevertheless, the protocol requires careful synchronization across clients and is sensitive to user dropout, which can affect reliability and communication efficiency.

Aono et al. [10] proposed privacy-preserving deep learning using additively homomorphic encryption to allow secure computation of neural network functions on encrypted data. Although effective for privacy protection, this approach introduces considerable computational overhead and latency, making it unsuitable for real-time or large-scale applications. Hao et al. [11] developed techniques to enhance federated deep learning efficiency and privacy by using model update sparsification, quantization, and secure aggregation. However, sparsification and quantization introduce additional complexity and may reduce model performance. Overall, cryptography-based approaches tend to incur high communication and computation costs due to the use of encryption or secret sharing.

D. Recent Advances in DP+MPC for FL

Recent research on combining differential privacy with MPC in FL has explored integrated approaches to strengthen end-to-end privacy guarantees [50], [51], [53]–[56], [103]–[105]. Xu et al. [103] presented HybridAlpha, which combines federated learning with differential privacy and MPC to enhance privacy during collaborative model training across different entities. However, the combined use of DP and MPC increases both computation and communication overhead. Keller et al. [104] proposed secure noise sampling within MPC to eliminate the need for clients to trust locally generated randomness, but this improvement comes at the cost of additional interaction steps and MPC computation. Zheng et al. [105] studied optimization techniques for the DP and MPC pipeline to improve the privacy-utility trade-off through coordinated mechanisms, although such coordination increases the complexity of system design and operation. Similarly, Chen et al. [50] characterized the fundamental communication cost of secure aggregation for centrally differentially private federated learning and designed a near-optimal scheme via sparse random projections that matches these bounds. However, achieving such guarantees still incurs substantial per-client communication and additional computational overhead with careful parameter tuning, potentially limiting scalability in large-scale deployments.

To address the limitations described in Sections II-B, II-C, and II-D, we propose a novel privacy-preserving federated learning scheme with distributed differential privacy via secure aggregation, named DDP-SA. This scheme integrates local differential privacy with secure aggregation based on MPC, combining their respective strengths to defend against privacy inference attacks while maintaining acceptable model accuracy and efficiency. After detailed analysis, our DDP-SA framework emphasizes simplicity, scalability, and controllable linear cost through full-threshold additive secret sharing and client-side local DP.

III. Preliminaries

A. FedAvg Algorithm

Federated learning (FL) is a distributed machine learning paradigm that enables multiple clients to collaboratively train a shared global model without centralizing their private datasets. It allows us to formally define the federated averaging (FedAvg) algorithm [2], which serves as the foundation for our DDP-SA framework.

TABLE I
AN OVERVIEW STUDY OF DIFFERENTIALLY PRIVATE FL [33]

Federated Scenario	Publications	Year	DP Model	Neighborhood Level	Perturbation Mechanism	CM[1]	Downstream Tasks	Model Architecture[2]	Clients Number	ϵ	δ
Horizontal	Chen et al. [27]	2024	DP	SL	Gaussian	tCDP	Classification	LR, Shallow CNN	100	0.3	10^{-2}
	malekmohammadi et al. [28]	2024			Gaussian	AC	Classification	CNN	[20,60]	[0.5,5]	10^{-4}
	Liu et al. [29]	2024			Gaussian	RDP	Classification	CNN	10	[0.1,10]	10^{-3}
	Ling et al. [30]	2024			Gaussian	RDP	Classification	Shallow CNN	10	[1.5,5.5]	10^{-5}
	Xiang et al. [34]	2023			Gaussian	MA	Classification	Shallow CNN,LSTM	[10,20]	[0.12,2]	$[10^{-2}, 10^{-5}]$
	Ruan et al. [31]	2023			Gaussian	RDP	Classification	Shallow CNN, LSTM	[3,10]	[0.25,2]	$[10^{-4}, 10^{-5}]$
	Noble et al. [32]	2022			Gaussian	RDP	Classification	Shallow CNN	10	[3,13]	10^{-6}
	Fu et al. [35]	2022			Gaussian	RDP	Classification	Shallow CNN	10	[2,6]	10^{-5}
	Li et al. [36]	2022			Gaussian	MA	Classification	LR, Shallow CNN	10	[1,16]	10^{-3}
	Ryu et al. [37]	2022			Gaussian	AC	Classification	LR	[10,195]	[0.05,5]	10^{-6}
	Wei et al. [38]	2021			Gaussian	MA	Classification	Shallow CNN	50	[4,20]	10^{-3}
	Liu et al. [39]	2021			Gaussian	GDP	Classification	Shallow CNN	100	[10,100]	10^{-3}
	Zheng et al. [25]	2021			Gaussian	GDP	Classification	Shallow CNN	100	[10,100]	10^{-3}
	Huang et al. [40]	2020			Gaussian, Laplace	AC	Classification	Shallow CNN	10,100,1000	[0.2,8]	$[10^{-2}, 10^{-5}]$
	Wei et al. [22]	2020			Gaussian	MA	Classification	Shallow CNN,LSTM	[10,20]	[0.12,2]	$[10^{-2}, 10^{-5}]$
	Huang et al. [41]	2019			Gaussian	AC	Regression	LR	-	[0.01,0.2]	$[10^{-3}, 10^{-6}]$
	Yang et al. [42]	2023		CL	Gaussian	RDP	Classification	Shallow CNN	50	[2,16]	10^{-3}
	Xu et al. [43]	2023			Gaussian	RDP	Classification	ResNet-50	[1262,9896000]	[10,20]	10^{-7}
	Shi et al. [44]	2023			Gaussian	RDP	Classification	ResNet-18	500	[4,10]	$\frac{1}{500}$
	Zhang et al. [45]	2022			Gaussian	MA	Classification	Shallow CNN, ResNet-18	1920	[1.5,5]	10^{-5}
	Cheng et al. [46]	2022			Gaussian	MA	Classification	Shallow CNN, ResNet-18	3400	[2,8]	$\frac{1}{3400}$
	Bietti et al. [47]	2022			Gaussian	MA	Classification	Shallow CNN	1000	[0.1,1000]	10^{-4}
	Andrew et al. [48]	2021			Gaussian	RDP	Classification	Shallow CNN	[500,342000]	[0.035,5]	$[\frac{1}{500}, \frac{1}{342000}]$
	Mcmahan et al. [49]	2018			Gaussian	MA	Classification	LSTM	[100,763430]	[2.0,4.6]	10^{-9}
	Geyer et al. [21]	2017			Gaussian	MA	Classification	Shallow CNN	100, 1000, 10000	8	$[10^{-3}, 10^{-6}]$
	Chen et al. [50]	2022		CL with SA	Discrete Gaussian	RDP	Classification	Shallow CNN	[100,1000]	[0,10]	10^{-2}
	Chen et al. [51]	2022			Poisson Binomial	RDP	Classification	LR	1000	[0.5,6]	10^{-5}
	Wang et al. [52]	2020			Discrete Gaussian	RDP	Classification	Shallow CNN	100K	[2,4]	10^{-5}
	Stevens et al. [53]	2022			LWE	RDP	Classification	Shallow CNN	[500,1000]	[2,8]	10^{-5}
	Kairouz et al. [54]	2021			Discrete Gaussian	zCDP	Classification	Shallow CNN	3400	[3,10]	$\frac{1}{3400}$
	Agarwal et al. [55]	2021			Skellam	RDP	Classification	Shallow CNN	1000k	[5,20]	10^{-6}
	Kerkouche et al. [56]	2021			Gaussian	MA	Classification	Shallow CNN	[5011,6000]	[0.5,1]	10^{-5}
	Agarwal et al. [57]	2018			Binomial	AC	Classification	LR	25M	[2,4]	10^{-9}
	Naseri et al. [58]	2022		SL, CL	Gaussian	RDP	Classification	Shallow CNN,LSTM	[100,660120]	[1.2,10.7]	10^{-5}
	Yang et al. [59]	2023		SL, CL, CL with SA	Gaussian, Skellam	RDP	Classification	Shallow CNN	[40,500]	[2,8]	10^{-3}
	Triastcyn et al. [60]	2019	Bayesian DP	SL, CL	Gaussian	RDP	Classification	ResNet-50	[100,10000]	[0.2,4]	$[10^{-3}, 10^{-6}]$
	Zhang et al. [61]	2024	LDP	-	Gaussian	zCDP	Classification	LR	20	1	10^{-4}
	Varun et al. [62]	2024			SRR	BC	Classification	Shallow CNN	100	[1,10]	0
	Zhang et al. [63]	2023			Gaussian	AC	Classification	LR, Shallow CNN	100	[3,30]	-
	Wang et al. [64]	2023			EM, DMP-UE	BC	Classification	Shallow CNN	[10,50]	[0.1,1]	0
	Jiang et al. [65]	2023			EM	BC	Classification	Shallow CNN	[100,750]	[0.5,12]	0
	Li et al. [66]	2023			Laplace	BC	Classification	Shallow CNN	100	78.5	0
	Lian et al. [67]	2022			Laplace	BC	Classification	Shallow CNN	5	[3,6]	0
	Mahawaga et al. [68]	2022			RAPPOR	BC	Classification	Shallow CNN	[2,100]	[0.5,10]	0
	Wang et al. [69]	2022			RAPPOR	BC	Classification	LR	[500,1800]	[0.1,10]	0
	Zhao et al. [70]	2022			Adaptive-Harmony	BC	Classification	Shallow CNN	200	[1,10]	0
	Sun et al. [71]	2021			Adaptive-Duchi	BC	Classification	Shallow CNN	[100,500]	[1,5]	0
	Yang et al. [72]	2021			Laplace	BC	Classification	Shallow CNN	[200,1000]	[1,10]	0
	Wang et al. [73]	2020			RRP	AC	Topic Modeling	LDA	150	[5,8]	[0.05,0.5]
	Zhao et al. [24]	2020			Three output, PM-SUB	BC	Classification	LR, SVM	4M	[0.5,4]	0
	Liu et al. [23]	2020			RR, PM	BC	Classification	LR, SVM	4W-10W	[0.5,16]	0
	Wang et al. [74]	2019			PM	BC	Classification	LR, SVM	4M	[0.5,4]	0
	Truex et al. [75]	2020	Condensed LDP	-	EM	BC	Classification	Shallow CNN	50	1	0
	Liu et al. [76]	2023	Shuffle DP	CL	Clipped-Laplace,Shuffle	AC	Classification	LR	10000	25.6	10^{-8}
	Liew et al. [77]	2023			Harmony,Shuffle	RDP	Classification	Shallow CNN	[50000,60000]	[2.8]	-
	Liu et al. [78]	2021			Laplace,Shuffle	BC, AC	Classification	LR	1000	4.696	5×10^{-6}
	Chen et al. [79]	2024		SL	Duchi,Shuffle	GDP	Classification	Shallow CNN	100	[0.5, 100]	10^{-5}
	Girgis et al. [80]	2021			Laplace,Shuffle	AC	Classification	Shallow CNN	60000	[1,10]	10^{-5}
Vertical	Takahashi et al. [81]	2023	Label DP	SL	KRR	BC	Classification	GBDT	3	[0.1,2.0]	-
	Yang et al. [82]	2022			Laplace, KRR	BC	Classification	Shallow CNN	2	1	0
	Oh et al. [83]	2022	DP	CL	Gaussian	RDP	Classification	VGG-16	10	[1,40]	-
	Chen et al. [84]	2020			Gaussian	GDP	Classification	Shallow CNN	[3,8]	-	-
	Wang et al. [85]	2020			Gaussian	AC	Classification	Shallow CNN	2	[0.001, 10]	10^{-2}
	Wu et al. [86]	2020			Laplace	BC	Classification	GBDT	[2,10]	-	-
	Mao et al. [87]	2024	LDP	-	Laplace, RR	BC	Classification	Shallow CNN	5	[0.1,4.0]	0
	Tian et al. [88]	2024			RR	BC	Classification	GBDT	3	4	0
	Li et al. [89]	2022	Condensed LDP	-	Discrete Laplace	BC	Classification	GBDT	2	[0.64,2.56]	0
Transfer	Wan et al. [90]	2023	DP	SL	Gaussian	AC	Recommendection	DeepFM	2	[0.05, 10]	-
	Hoech et al. [91]	2022			Gaussian	AC	Classification	Resnet-18	20	[0.1,0.5]	-
	Tian et al. [92]	2022			Gaussian	GDP	Text Generation	GPT-2	2000	[3,5]	10^{-6}
	Sun et al. [93]	2021			Random Sampling	AC	Classification	Shallow CNN	6	[0.003,0.65]	[0.006,0.65]
	Papernot er al. [94]	2018			Gaussian	RDP	Classification	Resnet-18	2	[0.59,8.03]	10^{-8}
	Papernot er al. [95]	2017			Laplace	MA	Classification	Shallow CNN	2	[2.04,8.19]	$[10^{-5}, 10^{-6}]$
	Dodwadmath et al. [96]	2022		CL	Laplace	MA	Classification	Shallow CNN	10	[11.75,20]	10^{-5}
	Pan et al. [97]	2021			Gaussian	RDP	Classification	Resnet-18	100	[0.95,9.03]	-
	Qi et al. [98]	2023	LDP	-	KRR	BC	Classification	Shallow CNN	[2,5]	[2,7]	0

1. CM=Composition Mechanism, BC=Basic Sequential Composition Theory, AC=Advanced Sequential Composition Theory.
2. LR=Logistic Regression, SVM=Support Vector Machine, GBDT=Gradient Boosting Decision Tree.

Problem Setup. Consider n clients $\{C_1, C_2, \ldots, C_n\}$, where each client C_i holds a private dataset $\mathcal{D}_i$ with $|\mathcal{D}_i| = N_i$ samples. The global objective is to minimize:

$$F(\theta) = \sum_{i=1}^{n} \frac{N_i}{N} F_i(\theta), \quad \text{where } F_i(\theta) = \frac{1}{N_i} \sum_{(x,y)\in\mathcal{D}_i} \ell(\theta; x, y), \tag{1}$$

where $N = \sum_{i=1}^{n} N_i$ is the total number of samples, $\ell(\cdot)$ is the loss function, and θ denotes the model parameters.

FedAvg Algorithm (Model Averaging Variant). At communication round t:

1) **Server broadcast:** The parameter server sends the current global model $\theta^{(t)}$ to all clients.
2) **Local updates:** Each client C_i performs E epochs of local SGD:

$$\theta_i^{(t+1)} = \theta^{(t)} - \eta \sum_{e=1}^{E} \nabla F_i(\theta_i^{(t,e)}), \tag{2}$$

where η is the learning rate and $\theta_i^{(t,e)}$ denotes client i's model after local epoch e.

3) **Server aggregation:** The parameter server computes the weighted average:

$$\theta^{(t+1)} = \sum_{i=1}^{n} \frac{N_i}{N} \theta_i^{(t+1)}. \tag{3}$$

Gradient Averaging Variant. In this work, we focus on the gradient averaging variant in which clients send gradients rather than model parameters. At each round, client C_i computes and sends:

$$g_i^{(t)} = \nabla F_i(\theta^{(t)}) = \frac{1}{N_i} \sum_{(x,y)\in\mathcal{D}_i} \nabla \ell(\theta^{(t)}; x, y). \tag{4}$$

The server updates the global model as:

$$\theta^{(t+1)} = \theta^{(t)} - \eta \sum_{i=1}^{n} \frac{N_i}{N} g_i^{(t)}. \tag{5}$$

This gradient-based formulation is equivalent to the original FedAvg algorithm [2] and serves as the target of our DDP-SA framework, where we apply local differential privacy and secure aggregation to protect $g_i^{(t)}$ during FL.

B. Differential Privacy

Differential privacy introduces randomness into a client's data or model updates before they are transmitted to the server to defend against privacy inference attacks in FL.

Definition 1 (Differential Privacy [7]): A randomized algorithm $\mathcal{M}$ with domain $\mathbb{N}^{|\mathcal{X}|}$ is (ϵ, δ)-differentially private if for all $\mathcal{S} \subseteq \text{Range}(\mathcal{M})$ and for all $x, y \in \mathbb{N}^{|\mathcal{X}|}$ such that $||x - y||_1 \leq 1$,

$$\Pr[\mathcal{M}(x) \in \mathcal{S}] \leq e^{\epsilon} \Pr[\mathcal{M}(y) \in \mathcal{S}] + \delta, \tag{6}$$

where ϵ defines the privacy budget and δ is the probability of privacy leakage. When $\delta = 0$, $\mathcal{M}$ is ϵ-differentially private.

Definition 2 (ℓ_1-Sensitivity [7]): The ℓ_1-sensitivity of a function $f : \mathbb{N}^{|\mathcal{X}|} \to \mathbb{R}^k$ is:

$$\Delta f = \max_{\substack{x,y\in\mathbb{N}^{|\mathcal{X}|} \\ ||x-y||_1=1}} ||f(x) - f(y)||_1. \tag{7}$$

Definition 3 (Laplace Distribution [7]): The Laplace distribution with scale b has probability density function:

$$\text{Lap}(x \mid b) = \frac{1}{2b} \exp\left(-\frac{|x|}{b}\right). \tag{8}$$

Its variance is $\sigma^2 = 2b^2$.

Definition 4 (Laplace Mechanism [7]): Given any function $f : \mathbb{N}^{|\mathcal{X}|} \to \mathbb{R}^k$, the Laplace mechanism is:

$$\mathcal{M}_L(x, f(\cdot), \epsilon) = f(x) + (Y_1, \ldots, Y_k), \tag{9}$$

where Y_i are i.i.d. random variables drawn from $\text{Lap}(\Delta f/\epsilon)$.

Proposition 1 (Post-Processing [7]): If $\mathcal{M} : \mathbb{N}^{|\mathcal{X}|} \to R$ is (ϵ, δ)-differentially private and $f : R \to R'$ is any randomized mapping, then $f \circ \mathcal{M} : \mathbb{N}^{|\mathcal{X}|} \to R'$ is also (ϵ, δ)-differentially private.

Differential privacy can be enforced without assuming trust in the central server by applying the mechanism $\mathcal{M}$ locally to each user's data before communication. This model, known as local differential privacy (LDP), is widely used in applications such as telemetry collection by Google, Apple, and Microsoft [106]–[108].

C. Secure Multi-party Computation

Secure multi-party computation (MPC) enables mutually distrusting parties to collaboratively compute a function on their private inputs while ensuring that each party's input remains confidential [109]. In FL, MPC protects the privacy of client data by ensuring that only aggregated information is revealed. MPC can be instantiated through oblivious transfer, secret sharing, and threshold homomorphic encryption. In this paper, we focus on additive secret sharing (ASS), a full-threshold secret sharing scheme.

ASS splits a secret S into n shares $s_1, \ldots, s_n$ in a finite field $\mathbb{Z}_p$ such that:

$$S \equiv \sum_{i=1}^{n} s_i \pmod{p}. \tag{10}$$

Each share is uniformly random and reveals no information about S on its own. The ASS procedure is presented in Algorithm 1.

Algorithm 1 ASS

Input: secret S, number of parties n

Output: shares $s_1, \ldots, s_n$ such that $S \equiv \sum_{i=1}^{n} s_i \pmod p$

Choose a large prime p
for $i = 1$ to $n-1$ **do**
    Sample s_i uniformly at random from $\mathbb{Z}_p$
end for
Compute $s_n \leftarrow S - \left(\sum_{i=1}^{n-1} s_i\right) \bmod p$
for each party $i = 1$ to n **do**
    Send share s_i to party i
end for

Reconstruction: When reconstruction is needed, all parties send their shares $s_1, \ldots, s_n$ to the reconstructor, who computes $S \leftarrow (\sum_{i=1}^{n} s_i) \bmod p$

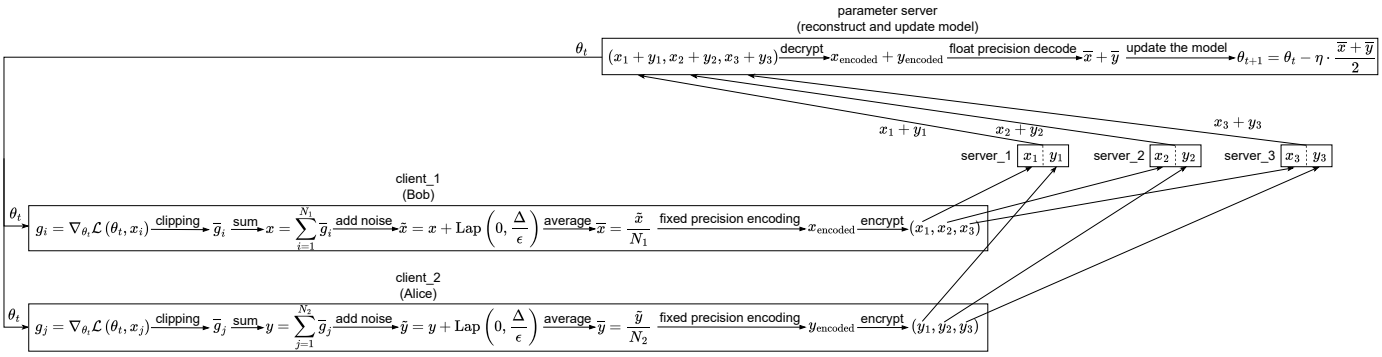

Fig. 1. DDP-SA Framework diagram.

Secure Aggregation in FL. With ASS, each client shares its model updates across multiple intermediate servers. The servers aggregate shares locally and send aggregated shares to the parameter server, which reconstructs the global sum. No individual client update is ever revealed during this process.

Security Interpretation. Under the semi-honest model, any strict subset of additive shares is uniformly random and independent of the secret. Therefore, an adversary controlling fewer than all servers learns nothing about any individual client update, which aligns with classical MPC security definitions [8], [109].

Quantization Error Bound. Let $q(x) = \text{round}(x \cdot \text{SF})/\text{SF}$ be fixed-point encoding with scaling factor $\text{SF} = 10^{d_n}$. Then each coordinate satisfies $|q(x) - x| \leq \frac{1}{2\,\text{SF}}$. If p is chosen larger than the maximum possible aggregated magnitude, wrap-around in $\mathbb{Z}_p$ is avoided and the decoding error remains bounded by $\frac{1}{2\,\text{SF}}$, which is negligible for large SF values.

IV. Methodology

The term "Distributed DP" refers to client-side local perturbation that achieves (ϵ, δ)-DP at the distributed client level, in contrast to central differential privacy. The term "Secure Aggregation" refers to full-threshold additive secret sharing (ASS), which protects noisy updates from being exposed during transmission or to the parameter server. Hence, DDP-SA stands for "Distributed Differential Privacy via Secure Aggregation". Unlike standalone LDP or CDP, and unlike schemes based solely on MPC, DDP-SA jointly provides statistical privacy through DP and cryptographic protection through ASS without degrading the original (ϵ, δ) privacy guarantee. It also prevents the exposure of per-client updates during aggregation.

In this section, we provide a comprehensive overview of the DDP-SA architecture, which includes the system model and the threat model. Fig. 1 illustrates the schematic framework of DDP-SA. For simplicity, the figure considers two clients (Bob and Alice). Local gradients x and y are illustrated as scalars, and the encoded values x_{encoded} and y_{encoded} are divided into m secret shares, to be sent to m intermediate servers (in the illustration, $m = 3$, although the protocol supports arbitrary m).

A. System Model

Our system model consists of three types of entities: clients, intermediate servers, and a parameter server. Compared to the conventional two-layer FL architecture with only clients and a parameter server, our design introduces a layer of intermediate servers that securely aggregate model updates using ASS. This layer ensures that the parameter server reconstructs only the aggregated update and not any single client's contribution. Each client trains a local neural network model through multiple rounds of iterative learning.

Clients: Each client holds a private dataset and has full control over its data. To prevent information leakage, a client computes its local gradient, adds Laplace noise, partitions the noisy gradient into multiple secret shares, and sends one share to each intermediate server. The client also receives global model parameters from the parameter server to compute its local gradient.

Intermediate servers: Each intermediate server possesses modest computational and storage capacity. It receives one secret share per client, performs addition operations on these shares to obtain a partial aggregate, and forwards this result to the parameter server.

Roles of intermediate servers. In addition to share aggregation, intermediate servers provide several system-level benefits:

1) **Share ingress and routing:** Receive per-server shares from all clients and route them reliably.
2) **Batching and compression:** Combine shares in batches to improve network utilization.
3) **Pipelined partial sums:** Forward intermediate sums upstream to reduce end-to-end latency.
4) **Bandwidth offloading:** Replace $O(n \cdot d)$ client-to-server bandwidth with $O(m \cdot d)$ intermediate-to-server bandwidth.
5) **Fault-domain isolation:** Reduce the effects of client churn and stragglers on the parameter server.

Each intermediate server sees only a single additive share per client and performs simple addition in $\mathbb{Z}_p$, without ever decrypting a client's update.

Parameter server: The parameter server receives all aggregated partial sums from the intermediate servers, reconstructs the complete aggregated gradient, computes the average, and updates the global model accordingly. At the beginning of each training round, it broadcasts the updated model parameters to all clients. Because ASS is full-threshold, the parameter server can reconstruct the aggregate only after receiving all m partial sums. Handling dropouts or applying dropout-tolerant aggregation techniques is outside the scope of this work and can be integrated as future improvements.

Motivation for Additive Secret Sharing (ASS): Although the architecture still includes a central parameter server, ASS enhances the end-to-end security of gradient transmission. Specifically, ASS prevents passive adversaries from learning perturbed gradients by observing communication links or compromising intermediate servers. The parameter server receives only aggregated results and, without collusion with all intermediate servers, cannot infer any single client's update. This limited use of MPC focuses solely on secure aggregation, rather than attempting full decentralization.

B. Threat Model

We consider a semi-honest adversary model. All parties follow the protocol but may attempt to infer additional information from the data they observe. Our threat model includes the following assumptions.

Adversary capability bounds:

1) The adversary may corrupt at most f intermediate servers, where $0 \leq f < m$, and may collude with a bounded number of clients (q_c).
2) The adversary may eavesdrop on a subset of communication links but cannot observe all links simultaneously.
3) Communication channels are authenticated to prevent message tampering; confidentiality is provided by ASS rather than transport-layer encryption.

Under these conditions, any strict subset of the m shares is statistically independent of the secret, so adding the intermediate server layer does not increase privacy risk. Confidentiality fails only if an adversary controls all m intermediate servers or if it observes all share-carrying links. This is a fundamental limitation of full-threshold ASS.

Security goal: Under the above assumptions, the confidentiality of each individual client's update is preserved. A strict subset of shares reveals no information about the client's perturbed gradient, and the parameter server learns only the aggregated update.

Failure condition: If an adversary controls all m intermediate servers or simultaneously observes all share-carrying links, it can reconstruct the aggregated secret. This limitation is inherent to full-threshold ASS and consistent with secure aggregation literature [8].

Attacker placements considered:

1) **External eavesdropper:** Can observe only a subset of communication links. Fewer than m captured shares are insufficient to reconstruct any client's update.
2) **Curious parameter server:** Sees only aggregated sums and cannot isolate any client's update unless colluding with all intermediate servers.
3) **Corrupted intermediate servers (up to $f < m$):** Each observes only one share per client. A strict subset of shares leaks no information.
4) **Curious clients:** Observe only the aggregated update, which prevents isolating the contribution of any other client.

Scope: Active adversaries (for example message dropping, replay, or forging) are outside the scope of this work. Such attacks can be mitigated with standard authentication and robustness techniques. Handling large-scale dropouts is also outside our focus and can be incorporated with dropout-tolerant aggregation.

C. DDP-SA

The DDP-SA procedure is presented in Algorithm 2. We consider n clients and m intermediate servers. Each client C_i maintains a private dataset and a local model. Each intermediate server S_j processes secret shares uploaded by clients. The algorithm proceeds as follows:

1) The parameter server broadcasts the initial model parameters to all clients.
2) Each client computes the local gradient, adds Laplace noise, encodes the noisy gradient using fixed precision, generates secret shares, and uploads these shares to the intermediate servers.
3) Each intermediate server aggregates secret shares from all clients and forwards the aggregated share to the parameter server.
4) The parameter server reconstructs the complete aggregated gradient, updates the global model, and broadcasts updated parameters to all clients.

This process repeats until the model converges or the maximum number of training rounds is reached. Fig. 2 shows the workflow of the DDP-SA framework. Each encoded gradient component is divided into m shares, so each intermediate server receives exactly one share per component.

Client-side operations: Each client computes gradients for its samples, clips them using the ℓ_1 norm, sums clipped gradients, adds Laplace noise, averages the noisy gradients, encodes the values using fixed precision, and partitions them into secret shares.

Server-side operations: Intermediate servers aggregate secret shares from all clients and send the aggregated results to the parameter server. The parameter server reconstructs the aggregated gradient and uses it to update the global model.

Benefits of intermediate servers: Intermediate servers improve scalability by reducing the parameter server's bandwidth load and enabling pipelined aggregation. Since each intermediate server receives only one share per client, no server can infer the client's update in isolation.

V. Theoretical Privacy Analysis

A. Single-Round Privacy Analysis

In this section, we provide formal end-to-end privacy guarantees for the DDP-SA framework and analyze how privacy loss behaves when combining local differential privacy (LDP) with MPC-based secure aggregation.

Theorem 1 (End-to-end Privacy Guarantee): Let $\mathcal{M}_{DDP\text{-}SA}$ denote the DDP-SA mechanism in which each client applies an (ϵ, δ)-LDP mechanism to its local gradient before ASS-based secure aggregation. Then $\mathcal{M}_{DDP\text{-}SA}$ satisfies (ϵ, δ)-differential privacy end-to-end.

Proof sketch: The proof uses two observations. First, each client's local mechanism satisfies (ϵ, δ)-LDP by construction, since it is the Laplace mechanism with an appropriate noise scale. Second, the ASS-based secure aggregation is a deterministic post-processing of the noisy gradients. By the post-processing invariance of differential privacy [7], any deterministic function applied to differentially private outputs preserves the same privacy guarantee. Since the aggregation via ASS is deterministic given the noisy inputs, the end-to-end mechanism inherits the (ϵ, δ)-DP guarantee without degradation. □

Privacy Loss Composition. An important question is whether combining LDP with MPC introduces any additional privacy loss. The following result answers this.

Corollary 1 (No Additional Privacy Loss): The privacy budget of DDP-SA is equal to that of the underlying LDP mechanism. The secure aggregation via ASS introduces zero additional privacy loss.

Proof sketch: This holds because ASS provides information-theoretic security. Any strict subset of secret shares is uniformly random and independent of the underlying secret. Therefore, an adversary that observes only a subset of shares gains no additional information beyond what is already accounted for by the local DP guarantee. □

Advantage over LDP Alone. Although DDP-SA and standalone LDP provide the same formal (ϵ, δ)-DP guarantee, DDP-SA offers stronger protection in realistic adversarial settings:

- **Communication security:** Individual client updates remain cryptographically protected during transmission, whereas LDP alone sends noisy gradients in plaintext.
- **Server-side protection:** The parameter server observes only aggregated updates, not individual client contributions, which provides an extra layer of protection beyond the DP noise.
- **Partial compromise resilience:** If an adversary compromises fewer than all m intermediate servers, it learns nothing about individual client updates because of the information-theoretic security of ASS.

Security Model. The analysis assumes a semi-honest adversary model in which all parties follow the protocol but may attempt to infer private information from their views. Under this model, DDP-SA combines statistical privacy (from DP) with cryptographic privacy (from ASS), providing defense in depth against different attack vectors.

B. Multi-Round Privacy Analysis

For practical FL systems, it is essential to understand how privacy guarantees evolve over multiple training rounds. We now analyze the cumulative privacy loss when the DDP-SA mechanism is executed for T training rounds.

Theorem 2 (Multi-Round Privacy Guarantee): Let $\mathcal{M}_{DDP\text{-}SA}^{(T)}$ denote the DDP-SA mechanism running for T rounds, where each round applies an (ϵ, δ)-LDP mechanism. Then:

1) **Basic composition:** $\mathcal{M}_{DDP\text{-}SA}^{(T)}$ satisfies $(T\epsilon, T\delta)$-differential privacy.
2) **Advanced composition:** For any $\delta' > 0$, $\mathcal{M}_{DDP\text{-}SA}^{(T)}$ satisfies $(\epsilon_{\text{total}}, \delta_{\text{total}})$-differential privacy, where

$$\epsilon_{\text{total}} = \epsilon\sqrt{2T\ln(1/\delta')} + \epsilon T(e^{\epsilon} - 1), \quad \delta_{\text{total}} = T\delta + \delta'. \tag{11}$$

Proof sketch: By Theorem 1, each individual round of DDP-SA satisfies (ϵ, δ)-DP. Applying the standard composition theorems for differential privacy [7] to the sequence of T rounds yields the stated bounds. The basic composition theorem yields $(T\epsilon, T\delta)$-DP. The advanced composition theorem gives a significantly tighter bound for ϵ_{total} when T is large. For example, if $\epsilon = 0.1$, $T = 1000$, and $\delta' = 10^{-4}$, then the advanced composition bound gives $\epsilon_{\text{total}} \approx 24.09$, whereas the basic composition bound gives $\epsilon_{\text{total}} = 100$. The latter corresponds to a much larger privacy loss. Hence, advanced composition is preferable for long-running federated learning systems. □

Privacy Budget Allocation Strategies. To manage cumulative privacy loss over multiple rounds, we consider two allocation strategies for the privacy budget:

1) **Uniform allocation:** Divide a total budget ϵ_{total} equally across T rounds, that is, $\epsilon_{\text{per-round}} = \epsilon_{\text{total}}/T$.
2) **Adaptive allocation:** Allocate more budget to early rounds, when gradients tend to have larger magnitude, using exponential decay,

$$\epsilon_t = \epsilon_{\text{total}} \cdot \frac{\alpha^{t-1}}{\sum_{i=1}^{T} \alpha^{i-1}}, \quad \alpha \in (0, 1). \tag{12}$$

Comparison with Multi-Round LDP. Both DDP-SA and a pure LDP approach experience the same formal composition of DP parameters over multiple rounds, since they apply the same per-round DP mechanism. However, DDP-SA maintains additional protections, such as encrypted communication and server-side protection, in every round. As a result, DDP-SA offers stronger practical protection than LDP alone, even though the formal (ϵ, δ) parameters are identical.

Practical Implications. As shown in Theorem 2, the cumulative privacy loss of DDP-SA can be bounded under both basic and advanced composition. For long-running FL systems, practitioners must carefully choose the total privacy budget and its allocation across rounds. The DDP-SA framework supports such budget management while retaining the cryptographic protections of secure aggregation throughout the entire training process.

VI. Experimental Evaluation

In this section, we present extensive experiments that verify the proposed DDP-SA scheme. The evaluation covers efficiency, accuracy, privacy, and detailed performance analysis.

A. Experimental Setup

Python, PyTorch 1.4.0, and PySyft 0.2.9 were used to implement and evaluate the proposed scheme. All experiments were conducted on GitHub Codespaces equipped with 16 CPU cores, 64 GB RAM, and 128 GB of storage.

A synthetic dataset was created by generating a 10000×2 array of random samples from a uniform distribution. For each row, the two values were summed and the constant 1 was added to obtain the corresponding label. The learning task is therefore a simple linear regression of the form $y = x_1 + x_2 + 1$. The data were split into training, validation, and test sets using a ratio of 60 percent, 20 percent, and 20 percent, respectively, and the training data were distributed evenly among all clients. Because all samples come from the same distribution, only the independent and identically distributed (IID) case is considered.

A two-layer neural network was used for fitting, with two neurons in the input layer and one neuron in the output layer. For the No-Private mechanism (which uses neither MPC nor LDP) and the MPC mechanism, standard SGD with learning rate 0.1 was used. For the LDP and DDP-SA mechanisms, the Adam optimizer with learning rate 0.001 was used. In both LDP and DDP-SA, each client clips per-sample gradients with the same ℓ_1 threshold Δ, sums the clipped gradients, adds IID Laplace noise with scale Δ/ϵ, and averages the noisy gradients locally before transmission and encoding. All optimizers and hyperparameters were identical across LDP and DDP-SA. The privacy budget ϵ was set to 0.1. The sensitivity Δ was chosen as the median of the ℓ_1 norms of the unclipped gradients across training. The number of retained decimal places d_n was set to 10.

Because the differential privacy mechanism is stochastic, each reported result is averaged over multiple runs. In addition, reconstruction at the parameter server requires receipt of all aggregated results from the intermediate servers.

Algorithm 2 DDP-SA

Input: set of clients $C = \{C_1, C_2, \ldots, C_n\}$, number of training rounds T, global model parameters θ, set of intermediate servers $S = \{S_1, S_2, \ldots, S_m\}$, privacy budget ϵ, clipping threshold Δ for the ℓ_1 norm, number of samples N_i for client C_i, learning rate η, total number of samples N across all clients, large prime p, loss function $\mathcal{L}$, gradient $\nabla_{\theta_t}\mathcal{L}(\theta_t, x_j)$, fixed precision scaling factor SF, number of decimal places d_n to preserve

Output: trained global model θ_T

for each round $t = 0, 1, \ldots, T-1$ **do**
  Parameter server broadcasts current model parameters θ_t to all clients
  for each client C_i in parallel **do**
    $\nabla\theta \leftarrow 0$
    for each sample x_j in C_i's local dataset **do**
      $\mathbf{g}_t(x_j) \leftarrow \nabla_{\theta_t}\mathcal{L}(\theta_t, x_j)$
      $\overline{\mathbf{g}}_t(x_j) \leftarrow \mathbf{g}_t(x_j) / \max\left(1, \frac{\|\mathbf{g}_t(x_j)\|_1}{\Delta}\right)$
      $\nabla\theta \leftarrow \nabla\theta + \overline{\mathbf{g}}_t(x_j)$
    end for
    $\tilde{\mathbf{g}}_t \leftarrow \frac{1}{N_i}\left(\nabla\theta + \mathrm{Lap}\left(0, \frac{\Delta}{\epsilon}\right)\right)$
    SF $\leftarrow 10^{d_n}$
    $\tilde{\mathbf{g}}_{t,\text{encoded}} \leftarrow \mathrm{round}(\tilde{\mathbf{g}}_t \times \text{SF})$
    shares $\leftarrow C_i$.secret_share$(\tilde{\mathbf{g}}_{t,\text{encoded}}, S)$
    for each share shares$[j]$ **do**
      send shares$[j]$ to S_j
    end for
  end for
  for each server S_j in parallel **do**
    $\nabla\theta_{\text{agg},j} \leftarrow$ sum of shares from all clients at S_j
    send $\nabla\theta_{\text{agg},j}$ to the parameter server
  end for
  $\nabla\theta_{\text{agg}} \leftarrow \left(\sum_{j=1}^{m} \nabla\theta_{\text{agg},j}\right) \bmod p$
  $\nabla\theta_{\text{agg}} \leftarrow \nabla\theta_{\text{agg}} / \text{SF}$
  $\theta_{t+1} \leftarrow \theta_t - \eta \cdot \frac{N_i}{N} \cdot \nabla\theta_{\text{agg}}$
end for
return θ_T

B. Efficiency Analysis

The efficiency analysis of the DDP-SA scheme focuses on two metrics: communication cost and computational cost. Communication cost is evaluated from the parameter server's perspective and includes communication between the parameter server and clients, as well as between intermediate servers and the parameter server. Communication between clients and intermediate servers is excluded unless stated otherwise.

Fig. 3 reports the total number of communication rounds until convergence under different defensive mechanisms. The No-Private and LDP mechanisms require 2082 and 2444 rounds, respectively. The MPC and DDP-SA mechanisms require 2070 and 2436 rounds, respectively. The results show that MPC behaves similarly to No-Private because neither mechanism introduces local noise and both use SGD with learning rate 0.1. Likewise, DDP-SA behaves similarly to LDP because both use local noise, clipping, and the Adam optimizer with learning rate 0.001. Optimizer choice can also influence round counts.

Fig. 4 shows the number of parameters uploaded per client under each mechanism. Both No-Private and LDP upload 3 parameters (model dimension $d = 3$). MPC and DDP-SA upload $3m$ parameters because each gradient component is split into m secret shares. In our experiments, $m = 3$ was chosen as a practical trade-off between security and cost. The protocol supports arbitrary values of m, communication cost scales linearly with m, and confidentiality holds unless all m paths are compromised.

Fig. 5 shows the total time to convergence for each mechanism. The No-Private and MPC mechanisms take 112 minutes and 138 minutes, respectively. The LDP and DDP-SA mechanisms take 172 minutes and 203 minutes, respectively. Fig. 6 shows the average training time per round. No-Private and MPC require 6.4553 seconds and 8 seconds per round, while LDP and DDP-SA require 8.4452 seconds and 10 seconds per round.

From these results, we conclude that DDP-SA incurs slightly higher communication and computation overhead than LDP or MPC. However, the overhead remains acceptable and controllable for practical settings.

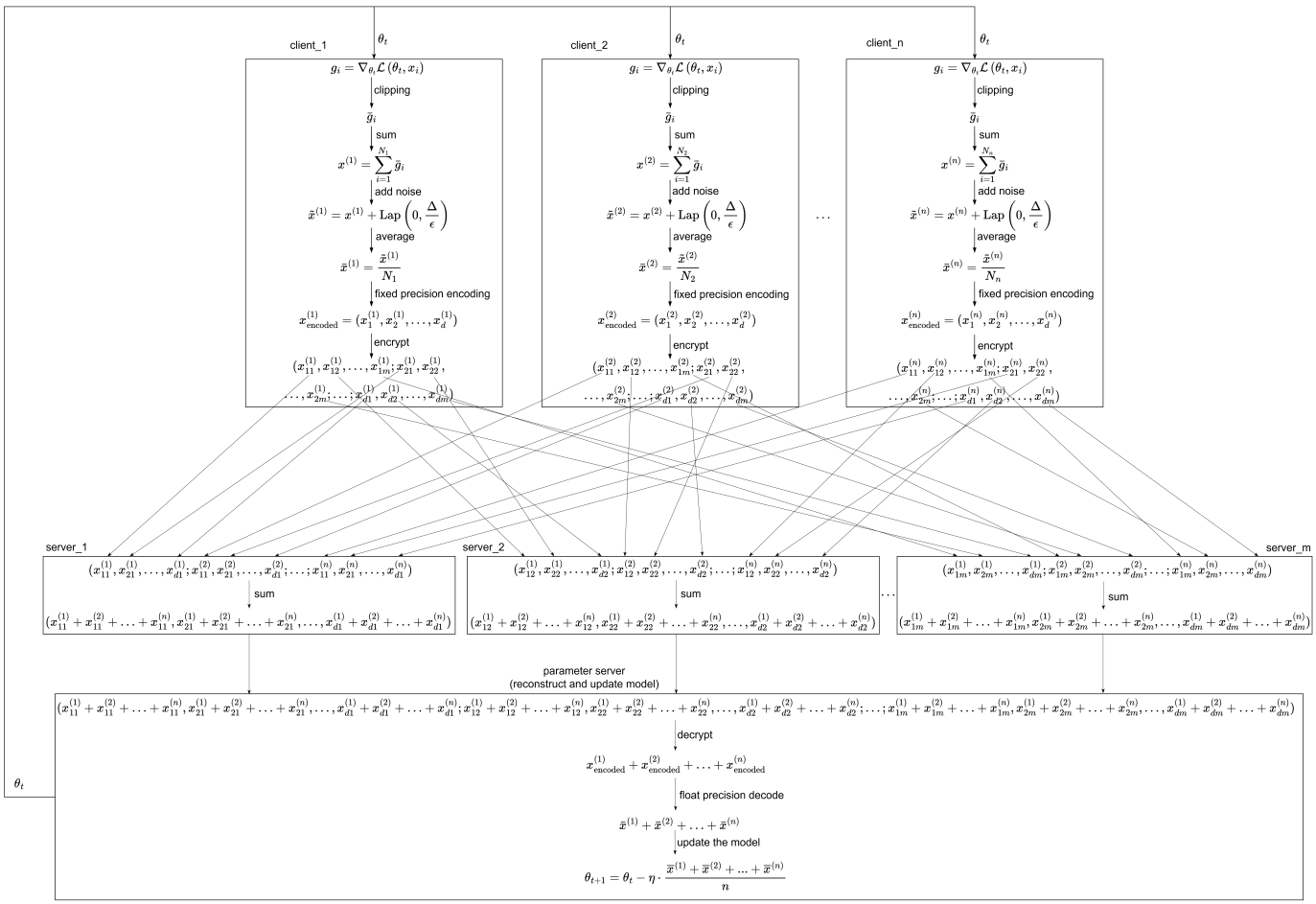

Fig. 2. DDP-SA workflow, general scalable framework with n clients, m intermediate servers, and d-dimensional parameters.

TABLE II
COMPUTATIONAL OVERHEAD BREAKDOWN PER CLIENT PER ROUND

Component	Operation	Time (ms)	Percentage of Total	Scalability
LDP	Gradient Clipping	547.95	92.07%	$O(d)$
	Noise Generation	0.24	0.04%	$O(d)$
	Noise Addition	0.05	0.01%	$O(d)$
MPC	Fixed-Point Encoding	0.22	0.04%	$O(d)$
	Secret Sharing	0.53	0.09%	$O(d \cdot m)$
	Share Transmission	46.13	7.75%	$O(d \cdot m)$
DDP-SA	All Operations	595.12	100.0%	$O(d \cdot m)$

C. Detailed Component-wise Overhead Analysis

We now present a quantitative breakdown of computational and communication overhead to isolate the contributions of LDP and MPC.

Computational Overhead Breakdown. Table II summarizes the per-client per-round computation cost:

- **LDP overhead:** Accounts for 92.12 percent of total computation, dominated by gradient clipping. This cost scales linearly with the parameter dimension d.
- **MPC overhead:** Accounts for 7.88 percent of total computation, dominated by share transmission which scales as $O(d{\cdot}m)$.
- **Combined DDP-SA overhead:** Dominated by gradient clipping with scaling $O(d)$.
- **Primary bottleneck:** Gradient clipping rather than cryptographic operations.
- **Server-side operations excluded:** Aggregation and reconstruction at intermediate servers and the parameter server are not part of the client overhead.

Communication Overhead Breakdown. Table III reports detailed bandwidth usage:

- **LDP:** No additional overhead relative to No-Private.
- **MPC:** Uploads m shares per gradient component, giving a factor of m overhead.

TABLE III
COMMUNICATION OVERHEAD BREAKDOWN PER CLIENT PER ROUND

Component	Direction	Bytes	Percentage of Total	Scalability
LDP	PS to Client	$4d$	50.0%	$O(d)$
	Client to PS	$4d$	50.0%	$O(d)$
MPC	PS to Client	$4d$	25.0%	$O(d)$
	Client to Intermediate Servers (excluded)	$4d \cdot m$	-	$O(d \cdot m)$
	Intermediate Servers to PS	$4d \cdot m$	75.0% (for $m = 3$)	$O(d \cdot m)$
DDP-SA	PS to Client	$4d$	25.0%	$O(d)$
	Client to Intermediate Servers (excluded)	$4d \cdot m$	-	$O(d \cdot m)$
	Intermediate Servers to PS	$4d \cdot m$	75.0% (for $m = 3$)	$O(d \cdot m)$

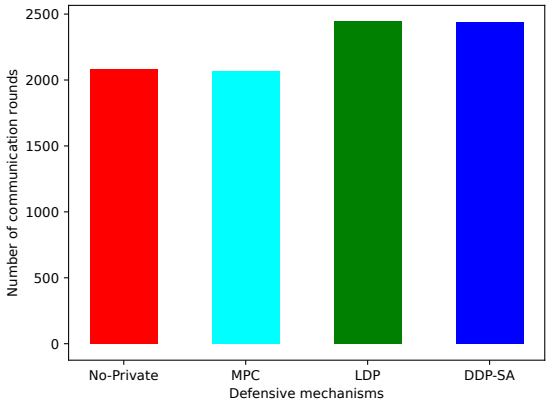

Fig. 3. Number of communication rounds for different defensive mechanisms.

- **DDP-SA:** Identical to MPC for communication overhead.
- **Intermediate server communication:** Adds $4d \cdot m$ bytes to the system but has no effect on clients.

Scalability Analysis. Both tables show how overhead scales with system parameters:

- **Parameter dimension d:** All methods scale linearly with d.
- **Number of intermediate servers m:** LDP unaffected. MPC and DDP-SA scale linearly with m.
- **Number of clients n:** Per-client cost unchanged. Total system overhead grows linearly in n.

From the scalability analysis, we can see that the DDP-SA improves scalability compared to LDP: it converts n client uplinks into m intermediate server uplinks (with $m \ll n$), reduces the parameter server's per-round ingress bandwidth from $4nd$ to $4md$, which enables scalability to many clients and long training horizons.

D. Accuracy Analysis

We use test loss and test R^2 (coefficient of determination) to evaluate the accuracy of the trained global model. Fig. 7(a) shows the test loss under different defensive mechanisms. As shown in Fig. 7(a), the test loss of No-Private and MPC is close to zero (around 10^{-12}), while the test loss of LDP and DDP-SA is 0.0106 and 0.0055, respectively. Thus, the test loss of LDP and DDP-SA is slightly higher than that of No-Private and MPC.

Fig. 7(b) shows the test R^2 for different mechanisms. The test R^2 of both No-Private and MPC is 0.9999, while the test R^2 of LDP and DDP-SA is 0.9357 and 0.9666, respectively. Hence, LDP and DDP-SA exhibit slightly lower test R^2 than No-Private and MPC, while DDP-SA achieves a higher test R^2 than LDP.

From these results, we conclude that DDP-SA incurs some accuracy loss relative to No-Private and MPC, but the loss is acceptable and controllable. Moreover, Fig. 7 shows that MPC and No-Private achieve essentially identical test loss and test R^2, which indicates that the MPC computation and fixed-point encoding are effectively lossless. This confirms that the choice $d_n = 10$ is appropriate and is consistent with the negligible decoding error for large scaling factors SF discussed in Section III-C.

E. Empirical Privacy Evaluation

1) Analysis of Privacy Protection Strength: The privacy budget ϵ quantifies the privacy protection strength. Smaller values of ϵ provide stronger privacy. Fig. 8(b) shows the effect of different values of ϵ on the test R^2. As ϵ increases, the test R^2 of

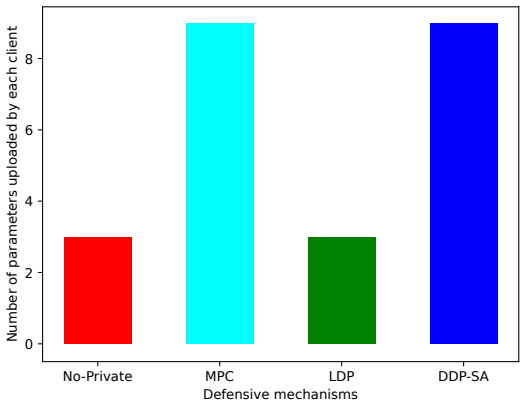

Fig. 4. Number of parameters uploaded per client for different defensive mechanisms. Results shown for $m = 3$.

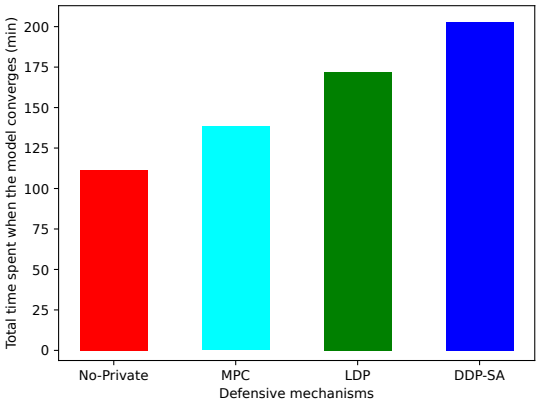

Fig. 5. Total time to convergence for different defensive mechanisms.

both DDP-SA and LDP increases, but DDP-SA consistently achieves higher R^2 than LDP. Hence, for a fixed target accuracy, DDP-SA can operate with a smaller privacy budget than LDP, which means that DDP-SA achieves stronger privacy protection.

MPC can be viewed as a special case of DDP-SA where the privacy budget is effectively infinite (no noise is added to local gradients) and the clipping norm is set to the maximum gradient norm (clipping has no practical effect). In this sense, DDP-SA can also provide stronger privacy protection than pure MPC. The same conclusion can be drawn from Fig. 8(a).

2) Analysis of Privacy Leakage:

Lemma 1 (Strict-subset Indistinguishability): Let S be a client's (noisy) update and let $\{s_1, \ldots, s_m\}$ be its ASS shares over $\mathbb{Z}_p$. For any strict subset $K \subset \{1, \ldots, m\}$,

$$I(S; \{s_k\}_{k \in K}) = 0. \tag{13}$$

Consequently, if each intermediate server (or link) is independently compromised with probability q, then the probability of reconstructing S is q^m, which decreases exponentially in m.

We now analyze privacy leakage for MPC, LDP, and DDP-SA using the DDP-SA workflow.

1) **MPC:** As discussed in Section IV-B, an external adversary can attempt to intercept communication among clients, intermediate servers, and the parameter server. When eavesdropping on client to intermediate server communication, the adversary sees only a single secret share in $\mathbb{Z}_p$, which is uniformly random and independent of the secret, so any strict subset of shares is information-theoretically useless. Interception between intermediate servers and the parameter server reveals only the sum of secret shares, which does not expose any single client update. From the parameter server to the clients, the adversary can observe only global model parameters, which aggregate updates from many clients and do not reveal individual inputs. The parameter server receives sums of secret shares and reconstructs the complete gradient, but secure aggregation prevents it from isolating any individual client's gradient. An intermediate server receives only one share per client and cannot reconstruct the gradient. A local client can access global model parameters. In a two-client scenario, a client could infer the

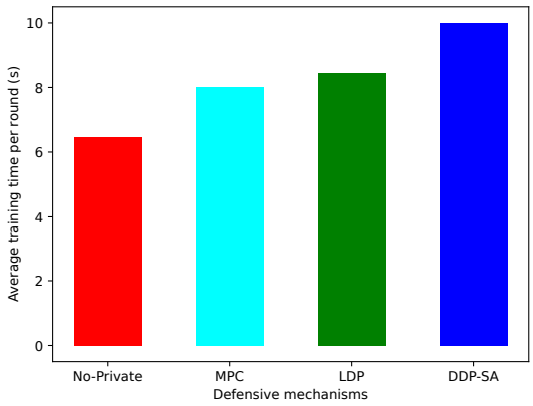

Fig. 6. Average training time per round for different defensive mechanisms.

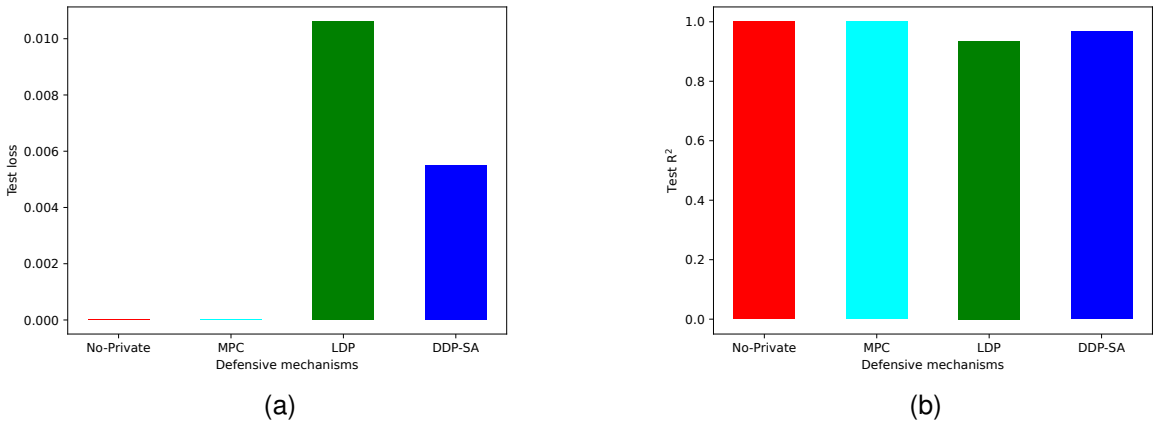

Fig. 7. Accuracy for different defensive mechanisms. (a) Test loss. (b) Test R^2.

other client's gradient from the difference between the global and its own gradient, which can reveal private information. However, with more than two clients, only aggregated gradients are available, which obscure individual contributions.

2) **LDP:** If an adversary eavesdrops on communication between a client and the parameter server, it observes only locally perturbed gradients. The adversary cannot recover the exact original data due to the noise, although, depending on the noise level, some limited inference may be possible. The parameter server receives only perturbed gradients and aggregate statistics based on them. It cannot deduce precise information about any individual update. Because LDP is applied locally before any sharing, no client or server can reverse the perturbation and recover the original data. Any further computation or model training on these noisy gradients preserves the DP guarantees by the post-processing property.
3) **DDP-SA:** For DDP-SA, if an adversary eavesdrops on client to intermediate server communication, it observes only a single secret share per client, which is uniformly random and independent of the underlying noisy update. Thus, no information can be inferred from any strict subset of shares. If the adversary intercepts traffic between intermediate servers and the parameter server, it observes only partial sums of shares which do not reveal individual contributions. Observing communication from the parameter server to the clients allows access only to the global model parameters, which are functions of the locally perturbed gradients. Due to LDP and the post-processing invariance of DP, these global parameters do not leak additional information beyond what is already permitted by the DP guarantee.

   The parameter server can reconstruct the aggregated noisy gradient but cannot deduce any individual client's gradient because of secure aggregation. Intermediate servers receive only one share per client and cannot learn the underlying update. Local clients see only the global model parameters and, under LDP, cannot reconstruct other clients' data.

Based on this analysis, we conclude that DDP-SA protects client data throughout the entire federated learning process and provides end-to-end privacy protection. By combining local perturbation with secure aggregation, DDP-SA reduces the risk of privacy leakage more effectively than either LDP or MPC alone, while maintaining controllable accuracy loss.

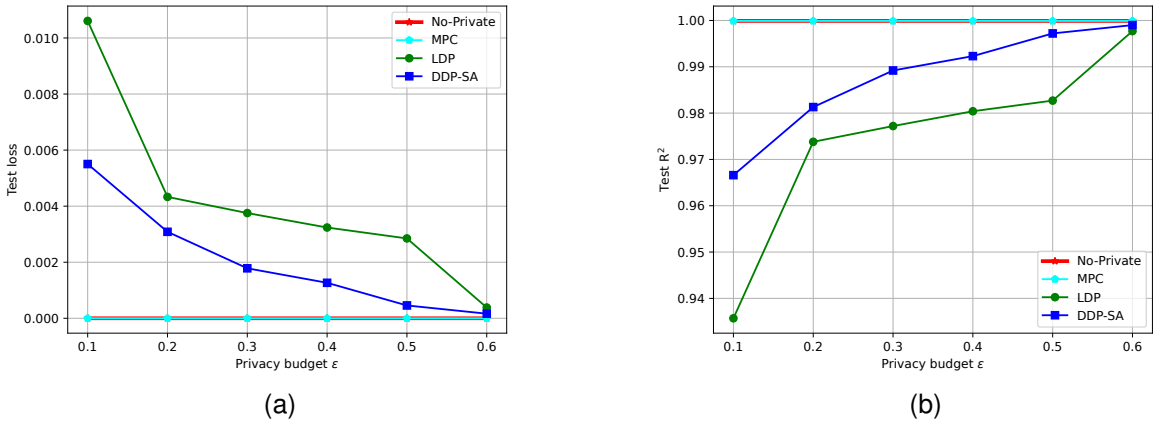

Fig. 8. Accuracy as a function of privacy budget ϵ. (a) Test loss vs. ϵ. (b) Test R^2 vs. ϵ.

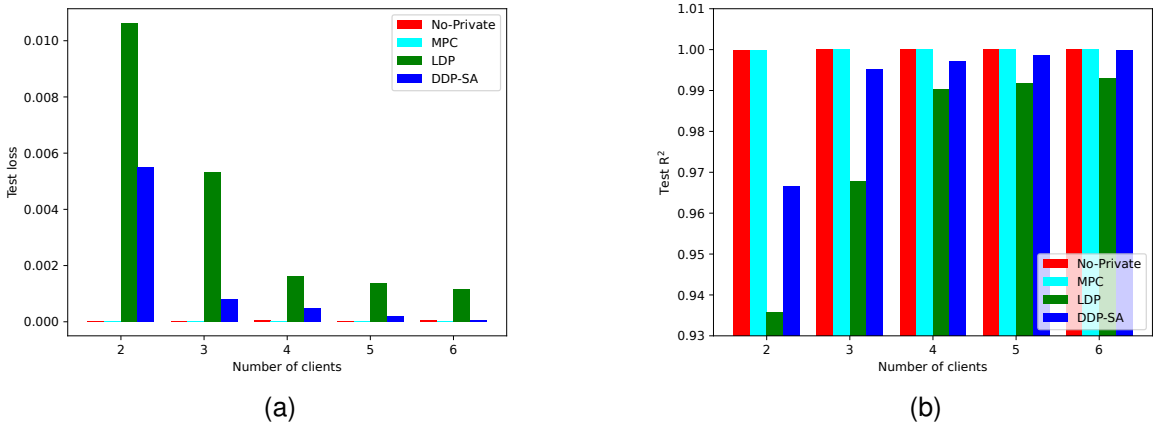

Fig. 9. Accuracy as a function of the number of clients n. (a) Test loss vs. n. (b) Test R^2 vs. n.

3) Analysis of Privacy Inference Attacks: We now discuss several common types of privacy inference attacks in the context of MPC, LDP, and DDP-SA.

1) **Membership inference attacks:** By adding noise to client updates under local differential privacy, DDP-SA and LDP prevent adversaries from reliably determining whether a specific sample was used in training. The noise masks the contribution of individual records, which mitigates membership inference attacks.
2) **Property inference attacks:** DDP-SA and LDP perturb gradients before aggregation, which hides fine-grained patterns that might reveal sensitive properties of the training data that are not explicitly modeled. This significantly reduces the effectiveness of property inference attacks.
3) **Training data or label inference attacks:** Secure aggregation in DDP-SA and MPC ensures that the model updates visible to the parameter server are aggregated and not attributable to any single client. This makes it difficult to reconstruct training inputs or labels from observed updates.
4) **Class representative attacks:** By obfuscating individual gradients through LDP and only revealing aggregates through secure aggregation, DDP-SA and LDP prevent adversaries from reconstructing representative samples for a particular class from the observed gradients.

In summary, DDP-SA is designed to mitigate a wide range of privacy inference attacks, including membership inference, property inference, training data or label inference, and class representative attacks. By combining local differential privacy with secure aggregation, it offers stronger protection than LDP or MPC used in isolation.

F. Performance Analysis

To evaluate the performance of the proposed DDP-SA scheme under varying conditions, we consider three key factors that influence the accuracy of the global model: the privacy budget ϵ, the number of clients n, and the number of communication

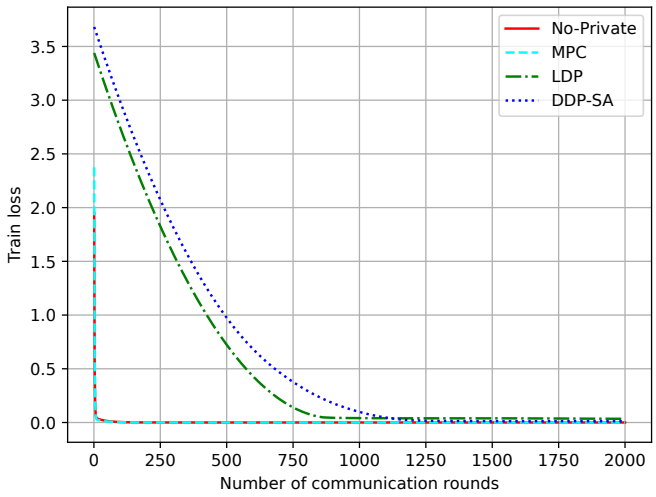

Fig. 10. Training loss as a function of the number of communication rounds T.

rounds T.

1) Evaluation with respect to ϵ: Fig. 8 shows how different values of ϵ affect model accuracy. In this experiment, ϵ is varied from 0.1 to 0.6, while all other settings remain fixed. From Fig. 8(a), the test loss of both No-Private and MPC remains close to zero (around 10^{-12}) for all values of ϵ. The test loss of LDP and DDP-SA decreases as ϵ increases, and the loss for DDP-SA is consistently lower than that for LDP. When ϵ reaches 0.6, the test loss of both LDP and DDP-SA is close to 10^{-4}.

From Fig. 8(b), the test R^2 of No-Private and MPC remains at 0.9999 for all values of ϵ. The test R^2 of LDP and DDP-SA increases with ϵ, and the value for DDP-SA is always higher than that for LDP. When ϵ reaches 0.6, the test R^2 of both LDP and DDP-SA is close to 0.9999.

These observations reflect the fundamental trade-off in differential privacy. Larger ϵ implies weaker privacy but higher accuracy, whereas smaller ϵ implies stronger privacy but lower accuracy. Overall, Fig. 8 shows that DDP-SA achieves better accuracy than LDP for all tested values of ϵ.

2) Evaluation with respect to n: The number of participating clients can also affect model accuracy. In this experiment, the number of clients n is increased from 2 to 6 while keeping all other settings fixed. Fig. 9 shows the resulting accuracy.

From Fig. 9(a), the test loss of No-Private and MPC remains close to zero (about 10^{-12}) for all values of n. The test loss of LDP and DDP-SA decreases as n increases, and the loss for DDP-SA is always lower than that for LDP. When $n = 6$, the test loss of DDP-SA is close to 10^{-6}.

Fig. 9(b) shows that the test R^2 of No-Private and MPC remains close to 1 for all values of n. The test R^2 of LDP and DDP-SA increases with n, and DDP-SA consistently achieves higher R^2 than LDP. When $n = 6$, the test R^2 of DDP-SA is close to 1.

This behavior can be explained by the averaging effect of noise. As the number of clients increases, the average of the added noise tends to zero, and the average noisy gradient approaches the true average gradient. Consequently, the resulting model parameters become closer to the true parameters, which reduces test loss and increases test R^2. Overall, Fig. 9 shows that DDP-SA achieves better accuracy than LDP as the number of clients increases.

3) Evaluation with respect to T: Fig. 10 shows the effect of the number of communication rounds T on model accuracy. The training loss decreases rapidly as T increases. The number of communication rounds required for convergence is 1041 and 1035 for No-Private and MPC, and 1222 and 1218 for LDP and DDP-SA, respectively. Thus, LDP and DDP-SA require more rounds to reach convergence. Furthermore, the final training loss of DDP-SA is lower than that of LDP.

The increase in required rounds for LDP and DDP-SA is due to the noise added to local gradients, which introduces randomness into the optimization trajectory. This requires more iterations to reach a stable solution. Nevertheless, once converged, DDP-SA achieves better accuracy than LDP, as shown by the lower training loss.

In summary, the performance analysis shows that DDP-SA achieves better accuracy than LDP as the privacy budget ϵ, the number of clients n, and the number of communication rounds T increase, while still providing stronger privacy guarantees.

VII. Conclusion

In this paper, we proposed DDP-SA, a novel privacy-preserving federated learning framework designed to address privacy leakage in the federated learning process. The framework integrates local differential privacy and secure multi-party computation to protect clients' gradients during training, thereby offering stronger defense against privacy inference attacks. Extensive experimental results demonstrate that DDP-SA provides enhanced privacy guarantees compared to using LDP or MPC alone,

while maintaining acceptable efficiency and accuracy. In addition, DDP-SA safeguards clients' private data throughout the entire federated learning workflow and effectively mitigates various types of privacy inference attacks.

We also analyzed the performance of DDP-SA under different conditions and showed that it offers superior utility compared to LDP-based approaches. Future work includes exploring optimization strategies to further improve model accuracy and training efficiency, as well as extending the framework to non-IID data distributions and a wider range of model architectures.

REFERENCES

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

[2] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas, "Communication-efficient learning of deep networks from decentralized data," in *Artificial intelligence and statistics*. PMLR, 2017, pp. 1273–1282.

[3] L. T. Phong, Y. Aono, T. Hayashi, L. Wang, and S. Moriai, "Privacy-preserving deep learning: Revisited and enhanced," in *Applications and Techniques in Information Security: 8th International Conference, ATIS 2017, Auckland, New Zealand, July 6–7, 2017, Proceedings*. Springer, 2017, pp. 100–110.

[4] 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.

[5] P. Kairouz, H. B. McMahan, B. Avent, A. Bellet, M. Bennis, A. N. Bhagoji, K. Bonawitz, Z. Charles, G. Cormode, R. Cummings *et al.*, "Advances and open problems in federated learning," *Foundations and Trends® in Machine Learning*, vol. 14, no. 1–2, pp. 1–210, 2021.

[6] H. Lee, J. Kim, R. Hussain, S. Cho, and J. Son, "On defensive neural networks against inference attack in federated learning," in *ICC 2021-IEEE International Conference on Communications*. IEEE, 2021, pp. 1–6.

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

[8] K. Bonawitz, V. Ivanov, B. Kreuter, A. Marcedone, H. B. McMahan, S. Patel, D. Ramage, A. Segal, and K. Seth, "Practical secure aggregation for privacy-preserving machine learning," in *proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, 2017, pp. 1175–1191.

[9] G. Xu, H. Li, S. Liu, K. Yang, and X. Lin, "Verifynet: Secure and verifiable federated learning," *IEEE Transactions on Information Forensics and Security*, vol. 15, pp. 911–926, 2020.

[10] Y. Aono, T. Hayashi, L. Wang, S. Moriai *et al.*, "Privacy-preserving deep learning via additively homomorphic encryption," *IEEE transactions on information forensics and security*, vol. 13, no. 5, pp. 1333–1345, 2017.

[11] M. Hao, H. Li, G. Xu, S. Liu, and H. Yang, "Towards efficient and privacy-preserving federated deep learning," in *ICC 2019-2019 IEEE international conference on communications (ICC)*. IEEE, 2019, pp. 1–6.

[12] M. Hao, H. Li, X. Luo, G. Xu, H. Yang, and S. Liu, "Efficient and privacy-enhanced federated learning for industrial artificial intelligence," *IEEE Transactions on Industrial Informatics*, vol. 16, no. 10, pp. 6532–6542, 2020.

[13] H. B. McMahan, E. Moore, D. Ramage, and B. A. y Arcas, "Federated learning of deep networks using model averaging," *ArXiv*, vol. abs/1602.05629, 2016. [Online]. Available: https://api.semanticscholar.org/CorpusID:16861557

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

[15] M. Fredrikson, S. Jha, and T. Ristenpart, "Model inversion attacks that exploit confidence information and basic countermeasures," in *Proceedings of the 22nd ACM SIGSAC conference on computer and communications security*, 2015, pp. 1322–1333.

[16] L. Zhu, Z. Liu, and S. Han, "Deep leakage from gradients," in *Proceedings of the 33rd International Conference on Neural Information Processing Systems*, 2019, pp. 14 774–14 784.

[17] B. Zhao, K. R. Mopuri, and H. Bilen, "idlg: Improved deep leakage from gradients," *arXiv preprint arXiv:2001.02610*, 2020.

[18] B. Hitaj, G. Ateniese, and F. Perez-Cruz, "Deep models under the gan: information leakage from collaborative deep learning," in *Proceedings of the 2017 ACM SIGSAC conference on computer and communications security*, 2017, pp. 603–618.

[19] J. Geiping, H. Bauermeister, H. Dröge, and M. Moeller, "Inverting gradients-how easy is it to break privacy in federated learning?" *Advances in neural information processing systems*, vol. 33, pp. 16 937–16 947, 2020.

[20] 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.

[21] R. C. Geyer, T. Klein, and M. Nabi, "Differentially private federated learning: A client level perspective," *arXiv preprint arXiv:1712.07557*, 2017.

[22] K. Wei, J. Li, M. Ding, C. Ma, H. H. Yang, F. Farokhi, S. Jin, T. Q. Quek, and H. V. Poor, "Federated learning with differential privacy: Algorithms and performance analysis," *IEEE Transactions on Information Forensics and Security*, vol. 15, pp. 3454–3469, 2020.

[23] R. Liu, Y. Cao, M. Yoshikawa, and H. Chen, "Fedsel: Federated sgd under local differential privacy with top-k dimension selection," in *Database Systems for Advanced Applications: 25th International Conference, DASFAA 2020, Jeju, South Korea, September 24–27, 2020, Proceedings, Part I 25*. Springer, 2020, pp. 485–501.

[24] Y. Zhao, J. Zhao, M. Yang, T. Wang, N. Wang, L. Lyu, D. Niyato, and K.-Y. Lam, "Local differential privacy-based federated learning for internet of things," *IEEE Internet of Things Journal*, vol. 8, no. 11, pp. 8836–8853, 2021.

[25] Q. Zheng, S. Chen, Q. Long, and W. Su, "Federated f-differential privacy," in *International Conference on Artificial Intelligence and Statistics*. PMLR, 2021, pp. 2251–2259.

[26] M. Seif, R. Tandon, and M. Li, "Wireless federated learning with local differential privacy," in *2020 IEEE International Symposium on Information Theory (ISIT)*. IEEE, 2020, pp. 2604–2609.

[27] L. Chen, X. Ding, Z. Bao, P. Zhou, and H. Jin, "Differentially private federated learning on non-iid data: Convergence analysis and adaptive optimization," *IEEE Transactions on Knowledge and Data Engineering*, vol. 36, no. 9, pp. 4567–4581, 2024.

[28] S. Malekmohammadi, Y. Yu, and Y. Cao, "Noise-aware algorithm for heterogeneous differentially private federated learning," in *Proceedings of the 41st International Conference on Machine Learning*, 2024, pp. 34 461–34 498.

[29] J. Liu, J. Lou, L. Xiong, J. Liu, and X. Meng, "Cross-silo federated learning with record-level personalized differential privacy," in *Proceedings of the 2024 ACM SIGSAC Conference on Computer and Communications Security*, 2024, pp. 303–317.

[30] X. Ling, J. Fu, K. Wang, H. Liu, and Z. Chen, "Ali-dpfl: Differentially private federated learning with adaptive local iterations," in *2024 IEEE 25th International Symposium on a World of Wireless, Mobile and Multimedia Networks (WoWMoM)*. IEEE, 2024, pp. 349–358.

[31] W. Ruan, M. Xu, W. Fang, L. Wang, L. Wang, and W. Han, "Private, efficient, and accurate: Protecting models trained by multi-party learning with differential privacy," in *2023 IEEE Symposium on Security and Privacy (SP)*. IEEE, 2023, pp. 1926–1943.

[32] M. Noble, A. Bellet, and A. Dieuleveut, "Differentially private federated learning on heterogeneous data," in *International Conference on Artificial Intelligence and Statistics*. PMLR, 2022, pp. 10 110–10 145.

[33] J. Fu, Y. Hong, X. Ling, L. Wang, X. Ran, Z. Sun, W. H. Wang, Z. Chen, and Y. Cao, "Differentially private federated learning: A systematic review," *arXiv preprint arXiv:2405.08299*, 2024.

[34] Z. Xiang, T. Wang, W. Lin, and D. Wang, "Practical differentially private and byzantine-resilient federated learning," *Proceedings of the ACM on Management of Data*, vol. 1, no. 2, pp. 1–26, 2023.

[35] J. Fu, Z. Chen, and X. Han, "Adap dp-fl: Differentially private federated learning with adaptive noise," in *2022 IEEE International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom)*. IEEE, 2022, pp. 656–663.

[36] Z. Li, H. Zhao, B. Li, and Y. Chi, "Soteriafl: A unified framework for private federated learning with communication compression," *Advances in Neural Information Processing Systems*, vol. 35, pp. 4285–4300, 2022.

[37] M. Ryu and K. Kim, "Differentially private federated learning via inexact admm with multiple local updates," *arXiv preprint arXiv:2202.09409*, 2022.

[38] K. Wei, J. Li, M. Ding, C. Ma, H. Su, B. Zhang, and H. V. Poor, "User-level privacy-preserving federated learning: Analysis and performance optimization," *IEEE Transactions on Mobile Computing*, vol. 21, no. 9, pp. 3388–3401, 2021.

[39] J. Liu, J. Lou, L. Xiong, J. Liu, and X. Meng, "Projected federated averaging with heterogeneous differential privacy," *Proceedings of the VLDB Endowment*, vol. 15, no. 4, pp. 828–840, 2021.

[40] X. Huang, Y. Ding, Z. L. Jiang, S. Qi, X. Wang, and Q. Liao, "Dp-fl: a novel differentially private federated learning framework for the unbalanced data," *World Wide Web*, vol. 23, pp. 2529–2545, 2020.

[41] Z. Huang, R. Hu, Y. Guo, E. Chan-Tin, and Y. Gong, "Dp-admm: Admm-based distributed learning with differential privacy," *IEEE Transactions on Information Forensics and Security*, vol. 15, pp. 1002–1012, 2019.

[42] X. Yang, W. Huang, and M. Ye, "Dynamic personalized federated learning with adaptive differential privacy," *Advances in Neural Information Processing Systems*, vol. 36, pp. 72 181–72 192, 2023.

[43] Z. Xu, M. Collins, Y. Wang, L. Panait, S. Oh, S. Augenstein, T. Liu, F. Schroff, and H. B. McMahan, "Learning to generate image embeddings with user-level differential privacy," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 7969–7980.

[44] Y. Shi, Y. Liu, K. Wei, L. Shen, X. Wang, and D. Tao, "Make landscape flatter in differentially private federated learning," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 24 552–24 562.

[45] X. Zhang, X. Chen, M. Hong, Z. S. Wu, and J. Yi, "Understanding clipping for federated learning: Convergence and client-level differential privacy," in *International Conference on Machine Learning, ICML 2022*. PMLR, 2022, pp. 26 048–26 067.

[46] A. Cheng, P. Wang, X. S. Zhang, and J. Cheng, "Differentially private federated learning with local regularization and sparsification," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 10 122–10 131.

[47] A. Bietti, C.-Y. Wei, M. Dudik, J. Langford, and S. Wu, "Personalization improves privacy-accuracy tradeoffs in federated learning," in *International Conference on Machine Learning*. PMLR, 2022, pp. 1945–1962.

[48] G. Andrew, O. Thakkar, B. McMahan, and S. Ramaswamy, "Differentially private learning with adaptive clipping," *Advances in Neural Information Processing Systems*, vol. 34, pp. 17 455–17 466, 2021.

[49] H. B. McMahan, D. Ramage, K. Talwar, and L. Zhang, "Learning differentially private recurrent language models," in *International Conference on Learning Representations*, 2018, pp. 1–14.

[50] W.-N. Chen, C. A. C. Choo, P. Kairouz, and A. T. Suresh, "The fundamental price of secure aggregation in differentially private federated learning," in *International Conference on Machine Learning*. PMLR, 2022, pp. 3056–3089.

[51] W.-N. Chen, A. Ozgur, and P. Kairouz, "The poisson binomial mechanism for unbiased federated learning with secure aggregation," in *International Conference on Machine Learning*. PMLR, 2022, pp. 3490–3506.

[52] L. Wang, R. Jia, and D. Song, "D2p-fed: Differentially private federated learning with efficient communication," *arXiv preprint arXiv:2006.13039*, 2020.

[53] T. Stevens, C. Skalka, C. Vincent, J. Ring, S. Clark, and J. Near, "Efficient differentially private secure aggregation for federated learning via hardness of learning with errors," in *31st USENIX Security Symposium (USENIX Security 22)*, 2022, pp. 1379–1395.

[54] P. Kairouz, Z. Liu, and T. Steinke, "The distributed discrete gaussian mechanism for federated learning with secure aggregation," in *International Conference on Machine Learning*. PMLR, 2021, pp. 5201–5212.

[55] N. Agarwal, P. Kairouz, and Z. Liu, "The skellam mechanism for differentially private federated learning," *Advances in Neural Information Processing Systems*, vol. 34, pp. 5052–5064, 2021.

[56] R. Kerkouche, G. Ács, C. Castelluccia, and P. Genevès, "Compression boosts differentially private federated learning," in *2021 IEEE European Symposium on Security and Privacy (EuroS&P)*. IEEE, 2021, pp. 304–318.

[57] N. Agarwal, A. T. Suresh, F. X. X. Yu, S. Kumar, and B. McMahan, "cpsgd: Communication-efficient and differentially-private distributed sgd," *Advances in Neural Information Processing Systems*, vol. 31, 2018.

[58] M. Naseri, J. Hayes, and E. De Cristofaro, "Local and central differential privacy for robustness and privacy in federated learning," in *Proceedings of the 29th Network and Distributed System Security Symposium (NDSS)*, 2022.

[59] Y. Yang, B. Hui, H. Yuan, N. Gong, and Y. Cao, "{PrivateFL}: Accurate, differentially private federated learning via personalized data transformation," in *32nd USENIX Security Symposium (USENIX Security 23)*, 2023, pp. 1595–1612.

[60] 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.

[61] J. Zhang, D. Fay, and M. Johansson, "Dynamic privacy allocation for locally differentially private federated learning with composite objectives," in *ICASSP 2024-2024 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2024, pp. 9461–9465.

[62] M. Varun, S. Feng, H. Wang, S. Sural, and Y. Hong, "Towards accurate and stronger local differential privacy for federated learning with staircase randomized response," in *14th ACM Conference on Data and Application Security and Privacy*. ACM, 2024.

[63] S. Zhang, J. Zhang, G. Zhu, S. Long, and L. Zhetao, "Personalized federated learning method based on bregman divergence and differential privacy (in chinese)," *Journal of Software*, vol. 35, no. 11, pp. 5249–5262, 2023.

[64] B. Wang, Y. Chen, H. Jiang, and Z. Zhao, "Ppefl: Privacy-preserving edge federated learning with local differential privacy," *IEEE Internet of Things Journal*, vol. 10, no. 17, pp. 15 488–15 500, 2023.

[65] X. Jiang, X. Zhou, and J. Grossklags, "Signds-fl: Local differentially private federated learning with sign-based dimension selection," *ACM Transactions on Intelligent Systems and Technology (TIST)*, vol. 13, no. 5, pp. 1–22, 2022.

[66] Y. Li, G. Wang, T. Peng, and G. Feng, "Fedta: Locally-differential federated learning with top-k mechanism and adam optimization," in *Ubiquitous Security*, G. Wang, K.-K. R. Choo, J. Wu, and E. Damiani, Eds. Singapore: Springer Nature Singapore, 2023, pp. 380–391.

[67] Z. Lian, Q. Yang, Q. Zeng, and C. Su, "Webfed: Cross-platform federated learning framework based on web browser with local differential privacy," in *ICC 2022-IEEE International Conference on Communications*. IEEE, 2022, pp. 2071–2076.

[68] P. C. Mahawaga Arachchige, D. Liu, S. Camtepe, S. Nepal, M. Grobler, P. Bertok, and I. Khalil, "Local differential privacy for federated learning," in *European Symposium on Research in Computer Security*. Springer, 2022, pp. 195–216.

[69] C. Wang, X. Wu, G. Liu, T. Deng, K. Peng, and S. Wan, "Safeguarding cross-silo federated learning with local differential privacy," *Digital Communications and Networks*, vol. 8, no. 4, pp. 446–454, 2022.

[70] J. Zhao, M. Yang, R. Zhang, W. Song, J. Zheng, J. Feng, and S. Matwin, "Privacy-enhanced federated learning: A restrictively self-sampled and data-perturbed local differential privacy method," *Electronics*, vol. 11, no. 23, p. 4007, 2022.

[71] L. Sun, J. Qian, and X. Chen, "Ldp-fl: Practical private aggregation in federated learning with local differential privacy," in *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence*. International Joint Conferences on Artificial Intelligence Organization, 2021.

[72] G. Yang, S. Wang, and H. Wang, "Federated learning with personalized local differential privacy," in *2021 IEEE 6th International Conference on Computer and Communication Systems (ICCCS)*. IEEE, 2021, pp. 484–489.

[73] Y. Wang, Y. Tong, and D. Shi, "Federated latent dirichlet allocation: A local differential privacy based framework," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 34, no. 04, 2020, pp. 6283–6290.

[74] N. Wang, X. Xiao, Y. Yang, J. Zhao, S. C. Hui, H. Shin, J. Shin, and G. Yu, "Collecting and analyzing multidimensional data with local differential privacy," in *2019 IEEE 35th International Conference on Data Engineering (ICDE)*. IEEE, 2019, pp. 638–649.

[75] 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*, 2020, pp. 61–66.

[76] Y. Liu, S. Zhao, L. Xiong, Y. Liu, and H. Chen, "Echo of neighbors: Privacy amplification for personalized private federated learning with shuffle model," in *Proceedings of the AAAI Conference on Artificial Intelligence*, 2023.

[77] S. P. Liew, S. Hasegawa, and T. Takahashi, "Shuffled check-in: Privacy amplification towards practical distributed learning," in *Computer Security Symposium 2023 (CSS 2023)*. Information Processing Society of Japan, 2023.

[78] R. Liu, Y. Cao, H. Chen, R. Guo, and M. Yoshikawa, "Flame: Differentially private federated learning in the shuffle model," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 35, no. 10, 2021, pp. 8688–8696.

[79] E. Chen, Y. Cao, and Y. Ge, "A generalized shuffle framework for privacy amplification: Strengthening privacy guarantees and enhancing utility," vol. 38, no. 10, pp. 11 267–11 275, 2024.

[80] A. Girgis, D. Data, S. Diggavi, P. Kairouz, and A. T. Suresh, "Shuffled model of differential privacy in federated learning," in *International Conference on Artificial Intelligence and Statistics*. PMLR, 2021, pp. 2521–2529.

[81] H. Takahashi, J. Liu, and Y. Liu, "Eliminating label leakage in tree-based vertical federated learning," *arXiv preprint arXiv:2307.10318*, 2023.

[82] X. Yang, J. Sun, Y. Yao, J. Xie, and C. Wang, "Differentially private label protection in split learning," *arXiv preprint arXiv:2203.02073*, 2022.

[83] S. Oh, J. Park, S. Baek, H. Nam, P. Vepakomma, R. Raskar, M. Bennis, and S.-L. Kim, "Differentially private cutmix for split learning with vision transformer," *arXiv preprint arXiv:2210.15986*, 2022.

[84] T. Chen, X. Jin, Y. Sun, and W. Yin, "Vafl: a method of vertical asynchronous federated learning," *arXiv preprint arXiv:2007.06081*, 2020.

[85] C. Wang, J. Liang, M. Huang, B. Bai, K. Bai, and H. Li, "Hybrid differentially private federated learning on vertically partitioned data," *arXiv preprint arXiv:2009.02763*, 2020.

[86] Y. Wu, S. Cai, X. Xiao, G. Chen, and B. C. Ooi, "Privacy preserving vertical federated learning for tree-based models," *arXiv preprint arXiv:2008.06170*, 2020.

[87] Y. Mao, Z. Xin, Z. Li, J. Hong, Q. Yang, and S. Zhong, "Secure split learning against property inference, data reconstruction, and feature space hijacking attacks," in *Computer Security – ESORICS 2023*, ser. Lecture Notes in Computer Science, G. Tsudik, M. Conti, K. Liang, and G. Smaragdakis, Eds., vol. 14347. Springer, 2024, pp. 23–43.

[88] Z. Tian, R. Zhang, X. Hou, L. Lyu, T. Zhang, J. Liu, and K. Ren, "Federboost: Private federated learning for gbdt," *IEEE Transactions on Dependable and Secure Computing*, vol. 21, no. 3, pp. 1274–1285, 2024.

[89] X. Li, Y. Hu, W. Liu, H. Feng, L. Peng, Y. Hong, K. Ren, and Z. Qin, "Opboost: a vertical federated tree boosting framework based on order-preserving desensitization," *arXiv preprint arXiv:2210.01318*, 2022.

[90] S. Wan, D. Gao, H. Gu, and D. Hu, "Fedpdd: A privacy-preserving double distillation framework for cross-silo federated recommendation," *arXiv preprint arXiv:2305.06272*, 2023.

[91] H. Hoech, R. Rischke, K. Müller, and W. Samek, "Fedauxfdp: Differentially private one-shot federated distillation," in *Trustworthy Federated Learning*, ser. Lecture Notes in Computer Science, R. Goebel, H. Yu, B. Faltings, L. Fan, and Z. Xiong, Eds., vol. 13448. Cham: Springer, 2023, pp. 100–114.

[92] Z. Tian, Y. Zhao, Z. Huang, Y.-X. Wang, N. L. Zhang, and H. He, "Seqpate: Differentially private text generation via knowledge distillation," *Advances in Neural Information Processing Systems*, vol. 35, pp. 11 117–11 130, 2022.

[93] L. Sun and L. Lyu, "Federated model distillation with noise-free differential privacy," in *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence (IJCAI-21)*, 2021, pp. 1563–1570.

[94] N. Papernot, S. Song, I. Mironov, A. Raghunathan, K. Talwar, and U. Erlingsson, "Scalable private learning with pate," in *International Conference on Learning Representations*, 2018.

[95] N. Papernot, M. Abadi, U. Erlingsson, I. Goodfellow, and K. Talwar, "Semi-supervised knowledge transfer for deep learning from private training data," in *International Conference on Learning Representations*, 2017.

[96] A. Dodwadmath and S. U. Stich, "Preserving privacy with pate for heterogeneous data," in *NeurIPS 2022 Workshop on Distribution Shifts: Connecting Methods and Applications*, 2022.

[97] Y. Pan, J. Ni, and Z. Su, "Fl-pate: Differentially private federated learning with knowledge transfer," in *2021 IEEE Global Communications Conference (GLOBECOM)*. IEEE, 2021, pp. 1–6.

[98] T. Qi, F. Wu, C. Wu, L. He, Y. Huang, and X. Xie, "Differentially private knowledge transfer for federated learning," *Nature Communications*, vol. 14, no. 1, p. 3785, 2023.

[99] Y. Li, Y. Zhou, A. Jolfaei, D. Yu, G. Xu, and X. Zheng, "Privacy-preserving federated learning framework based on chained secure multiparty computing," *IEEE Internet of Things Journal*, vol. 8, no. 8, pp. 6178–6186, 2021.

[100] S. Hardy, W. Henecka, H. Ivey-Law, R. Nock, G. Patrini, G. Smith, and B. Thorne, "Private federated learning on vertically partitioned data via entity resolution and additively homomorphic encryption," *arXiv preprint arXiv:1711.10677*, 2017.

[101] D. Chai, L. Wang, K. Chen, and Q. Yang, "Secure federated matrix factorization," *IEEE Intelligent Systems*, vol. 36, no. 5, pp. 11–20, 2020.

[102] Y. Liu, Y. Kang, C. Xing, T. Chen, and Q. Yang, "A secure federated transfer learning framework," *IEEE Intelligent Systems*, vol. 35, no. 4, pp. 70–82, 2020.

[103] R. Xu, N. Baracaldo, Y. Zhou, A. Anwar, and H. Ludwig, "Hybridalpha: An efficient approach for privacy-preserving federated learning," in *Proceedings of the 12th ACM workshop on artificial intelligence and security*, 2019, pp. 13–23.

[104] H. Keller, H. Möllering, T. Schneider, O. Tkachenko, and L. Zhao, "Secure noise sampling for dp in mpc with finite precision," in *Proceedings of the 19th International Conference on Availability, Reliability and Security*, 2024, pp. 1–12.

[105] C. Zheng, L. Wang, Z. Xu, and H. Li, "Optimizing privacy in federated learning with mpc and differential privacy," in *Proceedings of the 2024 3rd Asia Conference on Algorithms, Computing and Machine Learning*, 2024, pp. 165–169.

[106] Ú. Erlingsson, V. Pihur, and A. Korolova, "Rappor: Randomized aggregatable privacy-preserving ordinal response," in *Proceedings of the 2014 ACM SIGSAC conference on computer and communications security*, 2014, pp. 1054–1067.

[107] D. P. Team, "Learning with privacy at scale," 2017. [Online]. Available: https://api.semanticscholar.org/CorpusID:43986173

[108] B. Ding, J. Kulkarni, and S. Yekhanin, "Collecting telemetry data privately," *Advances in Neural Information Processing Systems*, vol. 30, 2017.

[109] R. Canetti, U. Feige, O. Goldreich, and M. Naor, "Adaptively secure multi-party computation," in *Proceedings of the twenty-eighth annual ACM symposium on Theory of computing*, 1996, pp. 639–648.