Secure and Efficient Federated Learning with Adaptive Differential Privacy and Verifiable Homomorphic Aggregation

Mohaddese Seyedi, Farhad Rahmati, Zahra Seyedi

***Abstract*—Federated Learning (FL) enables collaborative model training without centralizing raw data, but remains vulnerable to gradient inference attacks, malicious aggregation servers, and communication inefficiencies. Existing cryptographic secure aggregation schemes provide confidentiality and verifiability yet lack formal statistical privacy guarantees, while most differential privacy (DP)-based approaches rely on fixed noise injection, resulting in suboptimal privacy–utility tradeoffs.**

This paper proposes HEAD-FL, a secure and efficient federated learning framework that integrates adaptive differential privacy with verifiable homomorphic aggregation. The proposed scheme introduces a round-adaptive Gaussian perturbation mechanism analyzed under the Rényi Differential Privacy (RDP) framework, enabling tight cumulative privacy accounting and explicit conversion to (ε, δ)-DP guarantees. By adopting Federated Averaging (FedAvg) instead of gradient-based aggregation, the framework significantly reduces communication overhead while preserving confidentiality, verifiability, and robustness to client dropouts.

Theoretical analysis and experimental evaluation demonstrate that HEAD-FLachieves improved privacy–utility tradeoffs and enhanced communication efficiency compared with fixed-noise and gradient-based secure aggregation methods, making it suitable for deployment in privacy-sensitive and bandwidth-constrained environments.

***Index Terms*—Privacy Preserving Federated Learning, Adaptive Differential Privacy, Homomorphic Encryption, Verifiable Aggregation.**

I. Introduction

Federated Learning (FL) has emerged as a transformative paradigm for collaborative machine learning, enabling multiple participants to jointly train models without centralizing raw data [1]. By keeping data local and sharing only model updates, FL addresses critical challenges posed by data privacy regulations such as GDPR and HIPAA, while mitigating the limitations of isolated "data islands" in distributed environments. Despite these advantages, FL faces two fundamental challenges that limit its deployment in sensitive domains such as healthcare, finance, and IoT systems: (i) **privacy risks**, where shared model updates may leak sensitive information through attacks such as gradient reconstruction and membership inference [2, 3], and (ii) **security threats**, where malicious participants can compromise the training process through model poisoning, data poisoning, backdoor insertion, or Byzantine attacks [4, 5, 6].

M. Seyedi, F. Rahmati, and Z.Seyedi are with the Department of Mathematics and Computer Science, Amirkabir University of Technology, Tehran, Iran. E-mails: mohaddese.ss@aut.ac.ir, frahmati@aut.ac.ir,zahraseyedi@aut.ac.ir,

Wang et al. [7] proposed FVFL, a flexible and verifiable privacy-preserving FL scheme that combines Paillier homomorphic encryption for confidentiality, Shamir's secret sharing for collusion resistance, and dropout-tolerant verification mechanisms. While FVFL provides strong cryptographic guarantees against honest-but-curious adversaries and malicious servers, it adopts a gradient-based FedSGD aggregation strategy that incurs high communication costs. However, it does not address the risk that participating clients may extract sensitive information from shared updates or model outputs during collaborative training. Moreover, existing DP-based FL schemes typically employ fixed noise levels across all training rounds, leading to suboptimal privacy-utility trade-offs: excessive noise degrades model accuracy, while insufficient noise fails to provide adequate protection when updates are most sensitive.

In this paper, we propose Homomorphic Encryption with Adaptive Differential Privacy for Federated Learning (HEAD-FL), an enhanced federated learning framework that integrates adaptive differential privacy with verifiable secure aggregation to achieve superior privacy-utility-efficiency tradeoffs. Unlike prior approaches, our scheme dynamically adjusts the Gaussian noise magnitude across training rounds according to an adaptive schedule, providing stronger protection when gradients are more informative while minimizing utility loss. By replacing FedSGD with FedAvg-based local training, we reduce communication overhead by approximately 80% without compromising security guarantees. Our adaptive DP mechanism is rigorously analyzed using Rényi Differential Privacy (RDP) composition, yielding tighter privacy bounds under advanced composition compared to fixed-noise schemes. Our main contributions are as follows:

- We design an adaptive differential privacy mechanism that dynamically adjusts noise levels across training rounds based on a principled schedule. At equivalent model utility, HEAD-FL reduces the cumulative privacy budget by approximately 48% compared to fixed-noise schemes, due to the sublinear growth of the privacy cost under the adaptive noise schedule.
- By integrating FedAvg into the FVFL framework in [7], we reduce communication cost by 80% compared to FedSGD-based schemes, making the approach practical for bandwidth-constrained large-scale environments.
- We provide a formal RDP-based privacy analysis that captures the cumulative privacy loss under adaptive noise

allocation, with explicit conversion to (ε, δ)-differential privacy guarantees.
- We provide a **formal security model** that captures confidentiality, differential privacy, and verifiability, and we prove that HEAD-FL meets these definitions under the semi-honest client and malicious server model.
- We conduct experiments on MNIST demonstrating that HEAD-FL achieves equivalent accuracy to fixed-noise baselines while reducing the cumulative privacy expenditure by 48%, with $\varepsilon \approx 15$ compared to $\varepsilon \approx 29$ for fixed DP at $\sigma = 3.0$. Additionally, HEAD-FL reduces communication overhead by $\sim$80% compared to FedSGD-based aggregation.

The remainder of this paper is organized as follows. Section II reviews related work. Section III introduces preliminaries. Section IV describes problem formulation including the system and threat models. Section V presents the proposed HEAD-FL scheme. Section VI defines and analyzes its security properties. Section VII evaluates performance, and Section VIII concludes the paper.

II. Related Work

FL enables collaborative model training while keeping raw data local, but practical FL systems remain vulnerable to privacy leakage from model updates, high communication overhead, and malicious or untrusted aggregation servers.

Secure aggregation protocols ensure that the central server learns only the aggregate of client updates without accessing individual contributions. Bonawitz et al. proposed a practical protocol using pairwise masking with dropout tolerance, achieving $1.73\times$ communication overhead while providing security in honest-but-curious settings [8]. Secret sharing-based approaches using Shamir's scheme provide strong security guarantees and support threshold-based reconstruction [9, 10].

Homomorphic encryption enables computation directly on encrypted data. Paillier encryption provides additive homomorphic properties suitable for aggregating model updates [11], with implementations showing 20-30% computational overhead [12]. CKKS-based schemes enable approximate arithmetic with reduced costs [13]. However, pure homomorphic encryption schemes often lack verifiability mechanisms to detect malicious server behavior.

To address malicious servers, verifiable federated learning schemes enable clients to validate aggregation correctness [14]. Wang et al. proposed FVFL, which integrates Paillier encryption, Shamir secret sharing, and polynomial-based verification to ensure confidentiality, verifiability, and dropout tolerance [7]. While FVFL provides strong cryptographic guarantees, it lacks statistical privacy protections and incurs high communication costs due to FedSGD-based aggregation. Our work extends FVFL by integrating adaptive DP and adopting FedAvg for improved efficiency.

Understanding privacy threats is crucial for motivating defense mechanisms in federated learning. Membership inference attacks (MIAs) determine whether a specific record was used to train a model [3]. Shokri et al. demonstrated that even black-box model access enables effective membership inference [3], while Nasr et al. showed that FL adversaries with white-box access to gradients achieve significantly higher attack success rates [15]. Recent studies exploit temporal patterns in gradient evolution across multiple rounds [16, 17], and source inference attacks identify which client's dataset contains a training record [18].

Beyond membership inference, gradient-based attacks can reconstruct training data from shared updates [2]. Gradient inversion attacks exploit information leakage in gradients to recover input features and labels, particularly effective in early training rounds [19, 20]. Gradient clipping combined with DP noise has been shown to effectively reduce attack success rates, though at the cost of model performance [21, 22]. These attacks motivate our design of adaptive noise allocation that provides stronger protection when needed while preserving utility.

Differential privacy provides formal guarantees that machine learning outputs do not reveal sensitive information about individual training records. The foundational work by Abadi et al. established the Gaussian mechanism with gradient clipping for differentially private deep learning, introducing the moments accountant for tight privacy analysis [23]. In federated settings, DP can be applied either centrally at the server or locally at each client before transmission [24, 25]. Local DP provides stronger guarantees against malicious servers but typically requires larger noise magnitudes, leading to degraded model utility [26].

To better analyze cumulative privacy loss under composition, Rényi Differential Privacy (RDP) was introduced as a tighter accounting framework [27]. RDP measures privacy loss using Rényi divergence and enables additive composition.This yields significantly tighter bounds than standard (ε, δ)-DP composition [28]. Several FL systems have adopted RDP-based accounting, with applications demonstrating privacy amplification through subsampling and iteration [29, 30].

Most existing DP-based FL schemes employ fixed noise levels across all training rounds [23, 28, 31]. To address this limitation, recent work has explored adaptive mechanisms where noise allocation varies across iterations [32, 33]. However, these approaches primarily focus on optimization behavior and assume benign servers, without simultaneously addressing cryptographic security or verifiability. Our work addresses these limitations by introducing an adaptive DP mechanism with rigorous RDP composition analysis, integrated with verifiable cryptographic aggregation

Existing work addresses privacy, efficiency, and verifiability largely in isolation. Table I compares our HEAD-FL framework with representative prior work across key dimensions. Our HEAD-FL framework simultaneously addresses these limitations by integrating adaptive differential privacy, verifiable cryptographic aggregation, and communication-efficient FedAvg-based training within a unified design.

III. Preliminaries

This section reviews the core concepts underlying the proposed framework, including federated averaging, differential privacy, homomorphic encryption, and secret sharing. These

TABLE I
FEATURE COMPARISON WITH RELATED WORK

Feature	Abadi [23]	FVFL [7]	Bonawitz [8]	McMahan [34]	SecureML [11]	Adaptive DP [32]	HEAD-FL (Ours)
Statistical Privacy (DP)	✓Fixed	×	×	✓Fixed	×	✓Adaptive	✓**Adaptive**
RDP Composition	✓	×	×	✓	×	×	✓
Cryptographic Security	×	✓	✓	×	✓	×	✓
Verifiable Aggregation	×	✓	×	×	×	×	✓
Communication Efficiency	Moderate	Low	Moderate	High	Moderate	Moderate	**High**
Dropout Tolerance	×	✓	✓	×	×	×	✓
Malicious Server Defense	×	✓	Partial	×	Partial	×	✓

preliminaries establish the notation and security foundations used throughout the paper.

A. Federated Averaging (FedAvg)

We adopt *Federated Averaging (FedAvg)* as the underlying optimization strategy due to its communication efficiency and scalability.

Consider a federated learning system with K clients, where client k holds a private dataset D_k. At communication round $t \in \{1, \ldots, T\}$, the server broadcasts the global model parameters $w^{(t)} \in \mathbb{R}^d$ to the selected clients. Each client initializes its local model with $w^{(t)}$ and performs E epochs of stochastic gradient descent (SGD) on its local data.

Let $\ell(w, x)$ denote the loss on a single sample x. The empirical local objective of client k is

$$L_k(w, D_k) = \frac{1}{|D_k|} \sum_{x \in D_k} \ell(w, x). \tag{1}$$

After local training, client k obtains an updated model $w_k^{(t+1)}$, which is sent to the server. The server aggregates the received models via a weighted average:

$$w^{(t+1)} = \sum_{k=1}^{K} \frac{|D_k|}{\sum_{i=1}^{K} |D_i|} w_k^{(t+1)}. \tag{2}$$

This procedure approximates minimizing the global objective

$$L(w) = \sum_{k=1}^{K} \frac{|D_k|}{\sum_{i=1}^{K} |D_i|} L_k(w, D_k), \tag{3}$$

while significantly reducing communication overhead compared to per-gradient aggregation methods.

B. Differential Privacy

Differential privacy (DP) provides a formal guarantee that the output of a randomized algorithm does not reveal sensitive information about any individual record.

Let $D \simeq D'$ denote two adjacent datasets differing in one record.

Definition 1 (Differential Privacy)**.** *A randomized mechanism $\mathcal{M}$ satisfies (ε, δ)-differential privacy if, for all adjacent datasets D, D' and all measurable sets S,*

$$\Pr[\mathcal{M}(D) \in S] \leq e^{\varepsilon} \Pr[\mathcal{M}(D') \in S] + \delta. \tag{4}$$

1) Gaussian Mechanism: The Gaussian mechanism achieves differential privacy by adding noise proportional to the sensitivity of a function.

a) Clipping.: Given a vector-valued function $F(D)$, its output is first clipped to a fixed ℓ_2-norm C:

$$\bar{F}(D) = F(D) \cdot \min\left(1, \frac{C}{\|F(D)\|_2}\right). \tag{5}$$

b) Noise Addition.: Gaussian noise is added to the clipped output:

$$\tilde{F}(D) = \bar{F}(D) + \mathcal{N}(0, \sigma^2 C^2 I). \tag{6}$$

c) Privacy Guarantee.: With an appropriate choice of σ, the Gaussian mechanism satisfies (ε, δ)-DP. The noise scale σ governs the privacy–utility tradeoff.

C. Paillier Cryptosystem

The Paillier cryptosystem is a probabilistic public-key encryption scheme with additive homomorphic properties, making it suitable for secure aggregation in federated learning.

a) Key Generation.: Two large primes p and q are selected to compute $n = pq$ and the key pair (pk, sk).

b) Encryption.: For a message $m \in \mathbb{Z}_n$, encryption is defined as

$$\text{Enc}_{pk}(m) = g^m r^n \bmod n^2, \tag{7}$$

where $r \in \mathbb{Z}_n^*$ is chosen uniformly at random.

c) Decryption.: The plaintext is recovered using the private key sk.

d) Homomorphic Properties.: Paillier encryption supports:

$$\text{Enc}(m_1) \cdot \text{Enc}(m_2) = \text{Enc}(m_1 + m_2), \tag{8}$$

$$\text{Enc}(m)^k = \text{Enc}(km), \tag{9}$$

enabling secure summation and scaling of encrypted values.

D. Shamir's Secret Sharing

Shamir's secret sharing scheme divides a secret among multiple participants such that only a threshold number can reconstruct it.

To share a secret a_0, a random polynomial of degree $d-1$ is constructed:

$$P(x) = a_0 + a_1 x + \cdots + a_{d-1} x^{d-1}, \tag{10}$$

where the coefficients $a_1, \ldots, a_{d-1}$ are chosen uniformly at random.

Each participant receives a share $(x_i, P(x_i))$. Given any d shares, the secret $a_0 = P(0)$ can be recovered via Lagrange interpolation, while fewer than d shares reveal no information about the secret. This property is leveraged to enable verifiable aggregation and robustness against collusion and client dropouts.

IV. Problem formulation

A. System Model

We consider a federated learning system composed of three entities:

- **Trusted Authority (TA)**: The TA is responsible for system initialization. It generates the cryptographic parameters, including Paillier encryption keys and Shamir's secret shares, and distributes them securely to the clients. The TA is assumed to be fully trusted and offline after initialization.
- **Clients**: Each client holds a private local dataset that cannot be shared due to privacy concerns or regulatory restrictions. Clients train local models on their datasets and contribute updates to the global model. In our scheme, clients add *adaptive differential privacy noise* to their updates before encryption to protect against gradient inference attacks.
- **Central Server**: The server coordinates the federated learning process. It collects encrypted updates from clients, performs homomorphic aggregation, and returns aggregated results. The server is not trusted: it may attempt to learn client information or forge aggregation results.

We adopt the **FedAvg** aggregation strategy. Each client performs several local training epochs before uploading its encrypted and perturbed update, reducing communication rounds compared to FedSGD. After aggregation and verification, clients update their local models with the noisy global result.

B. Threat Model

We assume the following adversarial behaviors:

- **Honest-but-curious clients**: Clients follow the protocol correctly but may attempt to infer information about other participants from the global model or aggregation results.
- **Malicious server**: The server may attempt to (i) infer client data from updates, (ii) forge aggregation results to mislead participants, or (iii) collude with a subset of clients to compromise others.
- **Collusion attacks**: We assume a subset of clients may collude with the server to compromise other clients' privacy. Our scheme resists such attacks through secret sharing and homomorphic encryption.
- **Dropouts**: In practice, some clients may be unavailable or disconnect during training. The scheme must remain robust and verifiable even when a subset of clients drop out.

C. Design Goals

Our scheme, HEAD-FL, aims to achieve the following goals:

- **Confidentiality**: Local model updates must remain semantically secure against adversaries, relying on Paillier encryption and secret sharing.
- **Adaptive Privacy**: By integrating differential privacy with adaptive noise allocation, the scheme must formally protect against gradient inversion and membership inference, while ensuring tighter composition bounds compared to fixed-noise approaches.
- **Verifiability**: Clients must be able to verify that aggregation results are correct and unmodified, even if the server is malicious or some clients are offline.
- **Efficiency**: The scheme must reduce communication overhead by adopting FedAvg and lightweight verification, enabling deployment in bandwidth-constrained environments such as IoT and mobile systems.

V. Proposed Scheme:HEAD-FL

We now present HEAD-FL, our improved federated learning framework that integrates adaptive differential privacy (DP) into the flexible and verifiable FVFL scheme. HEAD-FL retains the cryptographic guarantees of FVFL while enhancing privacy protection against gradient inference attacks and improving efficiency via FedAvg aggregation.

HEAD-FL consists of three entities: the Trusted Authority (TA), a set of clients, and a central server. The workflow proceeds in four main stages: (i) initialization by the TA, (ii) local training with adaptive DP perturbation by clients, (iii) encrypted aggregation by the server, and (iv) verification and model update by clients. The adaptive DP mechanism dynamically adjusts the Gaussian noise added to client updates based on the training round, achieving a better privacy–utility tradeoff compared to fixed-noise schemes.

A. Phase 1: Initialization

In this phase, the TA sets up the cryptographic environment and distributes parameters to all participants. The TA is assumed fully trusted and goes offline after initialization.

Steps of Initialization:

1) **Paillier Encryption Parameters** Given security parameter K_0, the TA generates Paillier parameters $\{N, g, p, q\}$ with public key $PK = \{N, g\}$, private key $SK = \{\lambda, p, q\}$ where $\lambda = \mathrm{lcm}(p-1, q-1)$, and encryption factor $\gamma = (r^*)^q \bmod N^2$ for random $r^* \in \mathbb{Z}_N^*$.
2) **Encoding Parameters** The TA generates a super-increasing sequence $\vec{a} = (a_1, \ldots, a_{n+1})$ where $a_{n+1} \gg \sum_{i=1}^{n} a_i$, computes $g_i = g^{a_i}$ for $i = 1, \ldots, n+1$, and generates interpolation constants $\vec{k} = (k_1, \ldots, k_{n+1})$ and $\vec{h} = (h_1, \ldots, h_{n+1})$. These parameters $\{\vec{a}, \vec{k}, \vec{h}, g_1, \ldots, g_{n+1}\}$ are distributed to all clients.
3) **Shamir's Secret Sharing** Using private prime p, the TA selects random $r < p$, constructs polynomials $G_1(x)$ and $G_2(x)$, and generates $2m$ subkey pairs from master keys r and $p - r$ at distinct points $\{x_i\}_{i=1}^{m}$:

$$S_i^{(1)} = \{(x_i, G_1(x_i))\}, S_i^{(2)} = \{(x_i, G_2(x_i))\}, i = 1, \ldots, m.$$

The TA computes verification factor $\beta = g^q \bmod N^2$, distributes $\{S_i^{(1)}, S_i^{(2)}\}$ to client i, and shares $\{x_1, \ldots, x_m\}$ and β with all clients.
4) **System Parameters** The TA publishes clipping bound C, initial noise multiplier σ_0, adaptive schedule factor α, and learning rate η.

B. Phase 2: Local Training (FedAvg)

In this phase, each client trains a local model using its private dataset. Unlike FVFL, which relies on FedSGD and requires frequent gradient uploads, our scheme adopts **FedAvg**, where clients perform multiple local epochs before communicating with the server. This reduces communication cost and lowers the risk of adversarial inference from frequent updates.

Steps of Local Training:

1) Each client i maintains its own local model parameters w_i^{t-1} from the previous round.
2) Client i trains its model locally for E epochs on its private dataset D_i using stochastic gradient descent (SGD), resulting in updated parameters $w_i^{t,E}$.
3) The client computes its local model update as

$$\Delta w_i^t = w_i^{t,E} - w_i^{t-1}.$$

The local update Δw_i^t is then clipped and perturbed with adaptive noise in the next phase before secure aggregation.

The local training procedure is summarized in Algorithm 1.

Algorithm 1 Local Client Training (Client i)

Input: Local model w_i^{t-1}, local dataset D_i, epochs E, learning rate η_l
Initialize local model: $w_i^{t,0} \leftarrow w_i^{t-1}$
for $e = 1$ to E **do**
    Sample a minibatch $B \subseteq D_i$
    Compute gradient: $g \leftarrow \nabla \mathcal{L}(w_i^{t,e-1}, B)$
    Update local model: $w_i^{t,e} \leftarrow w_i^{t,e-1} - \eta_l g$
end for
Compute local update: $\Delta w_i^t = w_i^{t,E} - w_i^{t-1}$
Output: Local update Δw_i^t

C. Phase 3: Adaptive DP

Before uploading updates, each client perturbs its local update to ensure privacy against gradient inference and membership attacks. Unlike FVFL, which does not apply statistical privacy protection, HEAD-FL integrates an **adaptive differential privacy (DP)** mechanism. This mechanism dynamically adjusts the Gaussian noise scale across rounds to achieve a better privacy–utility tradeoff.

Steps of Adaptive Perturbation:

1) Each client clips its local update Δw_i^t to bound sensitivity:

$$\overline{\Delta w_i^t} = \frac{\Delta w_i^t}{\max\{1, \frac{\|\Delta w_i^t\|_2}{C}\}}, \tag{11}$$

where C is the clipping threshold.
2) The client computes the adaptive noise scale for round t:

$$\sigma_t = \sigma_0 \cdot \sqrt{1 + \alpha t}, \tag{12}$$

where σ_0 is the base noise multiplier and α controls the growth rate.
3) The client perturbs the clipped update with Gaussian noise:

$$\widetilde{\Delta w_i^t} = \overline{\Delta w_i^t} + \mathcal{N}(0, \sigma_t^2 C^2 I_d). \tag{13}$$

4) The noisy update $\widetilde{\Delta w_i^t}$ is then ready for encryption.

This adaptive strategy introduces moderate noise in early rounds to preserve convergence, while gradually increasing noise in later rounds to strengthen privacy when updates are more sensitive. Advanced composition ensures the total privacy budget (ϵ, δ) across all rounds remains bounded. The adaptive perturbation procedure is summarized in Algorithm 2.

Algorithm 2 Adaptive DP Perturbation (Client i)

Input: Local update Δw_i^t, clipping bound C, base noise σ_0, schedule factor α
Clip update: $\overline{\Delta w_i^t} \leftarrow \Delta w_i^t / \max\{1, \|\Delta w_i^t\|_2 / C\}$
Compute noise scale: $\sigma_t \leftarrow \sigma_0 \cdot \sqrt{1 + \alpha t}$
Perturb update: $\widetilde{\Delta w_i^t} \leftarrow \overline{\Delta w_i^t} + \mathcal{N}(0, \sigma_t^2 C^2 I_d)$
Output: Noisy update $\widetilde{\Delta w_i^t}$

D. Phase 4: Encryption and Upload

After perturbation, client i sends a federated learning request with its ID to the cloud. When the cloud waits to receive the request from the client for more than t, it collects the IDs of all online clients and returns to each client the ID set $\{id_1, id_2, \ldots, id_l\}$. Based on this set, client i selects $x_i \in \{x_1, x_2, \ldots, x_m\}$ and computes

$$\beta(x_i) = \prod_{j=1,\, j\neq i}^{l} \frac{x_j}{x_j - x_i} \bmod p. \tag{16}$$

The client then derives the subkey

$$S_i = \beta(x_i) G_1(x_i) + \beta(x_i) G_2(x_i) \bmod p. \tag{17}$$

To ensure data security and reduce communication overhead, each client encrypts its noisy model update using parameters distributed by the TA. Specifically, client i uses the constant sequence $k = \{k_1, k_2, \ldots, k_{n+1}\}$ to interpolate the noisy update $\widetilde{\Delta w_i^t} = \{\widetilde{\Delta w_{i1}^t}, \ldots, \widetilde{\Delta w_{in}^t}\}$ together with the subkey S_i. The client then inputs $h = \{h_1, h_2, \ldots, h_{n+1}\}$ into the function $F_i(x)$ to obtain the values $F_i(h_j)$.

Using the encryption parameters $\{g_1, g_2, \ldots, g_{n+1}, \gamma, N\}$ and the subkey S_i, client i encrypts these values as

$$C_i = g_1^{F_i(h_1)} g_2^{F_i(h_2)} \cdots g_{n+1}^{F_i(h_{n+1})} r_i^N \gamma^{S_i} \pmod{N^2}. \tag{14}$$

Finally, the encrypted value C_i is uploaded to the cloud server, which performs the aggregation step of federated learning.

E. Phase 5: Secure Aggregation

After receiving the ciphertexts C_i from all participating clients, the aggregation server exploits the additive homomorphic property of the encryption scheme to compute the aggregated ciphertext at round t:

$$C^t = \prod_{i=1}^{l} C_i, \tag{15}$$

where C^t denotes the aggregated ciphertext. By the homomorphic property, decrypting C^t yields the sum of the perturbed local updates:

$$\mathrm{Dec}_{sk}(C^t) = \sum_{i=1}^{l} \widetilde{\Delta w_i}^t = \mathrm{Dec}_{sk}(C^t) = \widetilde{\Delta w}^t \tag{16}$$

The server broadcasts C to all clients. Since Paillier encryption is IND-CPA secure, the server cannot infer any information about the individual updates from the ciphertexts. To ensure correctness and defend against malicious aggregation, clients can verify the returned result. The scheme also supports dynamic client participation without affecting security or correctness. The secure aggregation procedure is summarized in Algorithm 3.

Algorithm 3 Secure Aggregation (Server Side)

Input: Encrypted updates $\{c_i^t\}_{i=1}^m$
Aggregate homomorphically: $C^t \leftarrow \prod_{i=1}^m c_i^t$
Broadcast C^t to all clients
Output: Aggregated ciphertext C^t

F. Phase 6:Verification and Model Update

After receiving the aggregation result, each client first decrypts the aggregated ciphertext using the Paillier decryption algorithm to obtain the plaintext aggregation result.

The client then applies the super-increasing sequence $\vec{a}$ to decode the decrypted value and recover the set of function evaluations $\{F(h_1), F(h_2), \ldots, F(h_{n+1})\}$.

Using these points, the client reconstructs the polynomial $F(x)$ via Lagrange interpolation. To verify the correctness of the aggregation, the client evaluates $F(k_{n+1})$ and checks the validity condition

$$\beta^{F(k_{n+1})} \equiv 1 \pmod{N^2},$$

using the secret parameter β.

If the verification succeeds, the aggregation result is accepted.

The client then evaluates $F(x)$ at the constant sequence $\vec{k}$ to compute the aggregated model update

$$\widetilde{\Delta w}^t = \{F(k_1), F(k_2), \ldots, F(k_n)\}.$$

Finally, each client updates its local model parameters as

$$w_i^t \leftarrow w_i^{t-1} + \eta_g \cdot \frac{\widetilde{\Delta w}^t}{|P_t|}, \tag{17}$$

where η_g is the global learning rate, $\widetilde{\Delta}_t$ denotes the aggregated update, and $|P_t|$ is the number of participating clients in round t. The verification and model update procedure is summarized in Algorithm 4.

VI. Effectiveness Analysis

In this section, we analyze the effectiveness of the proposed HEAD-FL scheme. Compared to FVFL, two aspects require redefinition: correctness (to account for the intentional DP perturbation) and data privacy (to the adaptive DP mechanism). Verifiability and flexibility remain unchanged from FVFL and are therefore referenced without repetition.

Algorithm 4 Verification and Model Update (Client i)

Input: Aggregated ciphertext C^t, key share SK_i, parameters $\{\vec{a}, \vec{k}, \vec{h}, \beta\}$, model w_i^{t-1}, $|P_t|$
Decrypt: $M^t \leftarrow \mathrm{Dec}_{SK_i}(C^t)$
Decode: $\{F(h_1), \ldots, F(h_{n+1})\} \leftarrow \mathrm{Decode}(M^t, \vec{a})$
Reconstruct: $F(x) \leftarrow \mathrm{Interpolate}(\{(h_j, F(h_j))\}_{j=1}^{n+1})$
Compute verification value: $v \leftarrow F(k_{n+1})$
if $\beta^v \equiv 1 \pmod{N^2}$ **then**
    Extract update: $\widetilde{\Delta w}^t \leftarrow \{F(k_1), \ldots, F(k_n)\}$
    Update model: $w_i^t \leftarrow w_i^{t-1} + \eta_g \cdot \frac{\widetilde{\Delta w}^t}{|P_t|}$
    Output: Updated model w_i^t
else
    abort \\ Aggregation verification failed
end if

A. Privacy Model: Local Differential Privacy

In this section, we define the privacy notion that our adaptive perturbation mechanism aims to satisfy. Our objective is to ensure that each client's model update remains differentially private before being transmitted to the server. Each client applies a local perturbation process to its model update such that no sensitive information about its training data can be inferred, even if the server or other participants attempt to analyze it.

1) Local Differential Privacy (Privacy Goal): Our privacy requirement is that each client's contribution is protected at the point of release. Accordingly, we adopt a notion of local differential privacy (LDP) applied to the client's model update, rather than to individual data items as in the classical formulation.

A client-side randomization mechanism $\mathcal{M}$ satisfies $(\varepsilon_t, \delta_t)$-DP in round t if, for any two datasets D and D' differing in one record and any measurable set S,

$$\Pr[\mathcal{M}(D) \in S] \leq e^{\varepsilon_t} \Pr[\mathcal{M}(D') \in S] + \delta_t.$$

This guarantee ensures that the output of the mechanism remains statistically indistinguishable under the modification of any single record. Consequently, even an adversary—whether the server or another participant—cannot reliably detect the presence or influence of a specific example. Any change in the update attributable to adding or removing one record is masked by the local randomization, preventing inference attacks based on gradient or parameter deviations.

2) Privacy Goal of the Adaptive Mechanism: The goal of our adaptive scheme is to design a local perturbation mechanism that satisfies:

a) Each client achieves per-round $(\varepsilon_t, \delta_t)$-DP locally; and
b) The cumulative privacy loss over all training rounds remains bounded under advanced composition.

Formally, if each round t satisfies $(\varepsilon_t, \delta_t)$-DP under mechanism $\mathcal{M}_i^t$, then after T rounds the total privacy budget should remain within a fixed bound $(\varepsilon_{\text{total}}, \delta_{\text{total}})$, determined using composition rules.

In the next subsection, we introduce **Rényi Differential Privacy (RDP)** as a mathematical framework for quantifying

this cumulative privacy loss and for proving that our adaptive perturbation mechanism satisfies the defined local DP model.

B. Rényi Differential Privacy Framework

To analyze the cumulative privacy loss of our adaptive local perturbation mechanism, we employ the framework of Rényi Differential Privacy (RDP). RDP provides tighter composition guarantees than traditional (ε, δ)-DP and enables convenient conversion between privacy accounting frameworks.

Definition 2 (Rényi divergence)**.** *For two probability distributions P and Q defined on the same measurable space and for any order $\alpha > 1$, the Rényi divergence of order α is*

$$D_\alpha(P\|Q) = \frac{1}{\alpha - 1} \log \left(\mathbb{E}_{x\sim Q} \left[\left(\frac{P(x)}{Q(x)} \right)^\alpha \right] \right). \tag{18}$$

Definition 3 (Rényi Differential Privacy)**.** *A randomized mechanism $\mathcal{M}$ satisfies (α, ε)-RDP if, for all adjacent datasets D and D',*

$$D_\alpha(\mathcal{M}(D)\|\mathcal{M}(D')) \le \varepsilon. \tag{19}$$

RDP enjoys three key properties that make it particularly suitable for multi-round privacy analysis:

Property 1 (Composition). If mechanisms $\mathcal{M}_1, \ldots, \mathcal{M}_T$ satisfy (α, ε_1)-RDP, ..., (α, ε_T)-RDP respectively, then their adaptive composition satisfies $(\alpha, \sum_{i=1}^T \varepsilon_i)$-RDP. Thus, RDP parameters compose by simple addition.

Property 2 (Post-processing). If $\mathcal{M}$ satisfies (α, ε)-RDP and f is any (possibly randomized) function, then $f \circ \mathcal{M}$ also satisfies (α, ε)-RDP. This follows from the data-processing inequality for Rényi divergence.

Property 3 (Conversion to (ε, δ)-DP). If $\mathcal{M}$ satisfies $(\alpha, \varepsilon_{\text{RDP}})$-RDP, then for any $0 < \delta < 1$ it also satisfies (ε', δ)-DP where

$$\varepsilon' = \varepsilon_{\text{RDP}} + \frac{\log(1/\delta)}{\alpha - 1}. \tag{20}$$

Optimizing over $\alpha > 1$ yields the tightest (ε, δ)-DP bound for a chosen δ.

Intuitively, RDP measures the distinguishability of outputs when $\mathcal{M}$ is applied to two neighboring datasets. Rather than bounding the probability ratio directly as in (ε, δ)-DP, RDP bounds the expected α-moment of that ratio, providing a smooth parameterization that is particularly effective for analyzing Gaussian mechanisms.

C. Per-Round Privacy Analysis

We now establish the RDP guarantee for a single round of our adaptive perturbation mechanism. The analysis proceeds in three steps: (1) reviewing the general Gaussian mechanism RDP bound, (2) instantiating it with our clipping and noise parameters, and (3) incorporating the adaptive noise schedule.

1) RDP Analysis of the Gaussian Mechanism: Consider the Gaussian mechanism that releases

$$\mathcal{M}(D) = f(D) + \mathcal{N}(0, \sigma^2 I), \tag{21}$$

where $f(D)$ has ℓ_2-sensitivity Δ, meaning that for any two adjacent datasets $D \sim D'$,

$$\|f(D) - f(D')\|_2 \le \Delta. \tag{22}$$

Given the dataset D, the output distribution is $\mathcal{M}(D) \sim \mathcal{N}(f(D), \sigma^2 I)$. For two adjacent datasets D and D', the mechanism outputs two Gaussian distributions differing only in their means:

$$\mathcal{M}(D) \sim \mathcal{N}(\mu_0, \sigma^2 I), \quad \mathcal{M}(D') \sim \mathcal{N}(\mu_1, \sigma^2 I), \tag{23}$$

where $\mu_0 = f(D)$ and $\mu_1 = f(D')$.

Lemma 1 (Mironov 2017)**.** *[27] The Rényi divergence of order $\alpha > 1$ between two Gaussian distributions with identical covariance $\sigma^2 I$ and means μ_0, μ_1 satisfying $\|\mu_0 - \mu_1\|_2 \le \Delta$ is*

$$D_\alpha(\mathcal{N}(\mu_0, \sigma^2 I)\|\mathcal{N}(\mu_1, \sigma^2 I)) = \frac{\alpha}{2\sigma^2}\|\mu_0 - \mu_1\|_2^2. \tag{24}$$

Applying Lemma 1 to the Gaussian mechanism, we obtain

$$\varepsilon_{\text{RDP}}(\alpha) = \max_{D\sim D'} D_\alpha(\mathcal{M}(D)\|\mathcal{M}(D')) = \frac{\alpha\Delta^2}{2\sigma^2}. \tag{25}$$

Therefore, the Gaussian mechanism with sensitivity Δ and noise standard deviation σ satisfies $(\alpha, \alpha\Delta^2/(2\sigma^2))$-RDP.

2) Application to Our Adaptive Mechanism: We now instantiate this general result for our local perturbation mechanism in round t. Recall from Algorithm 3 that each client clips its local update as defined in (11) and adds Gaussian noise with adaptive variance according to Eq. (27).

The clipping step ensures that the ℓ_2-sensitivity of the local update is bounded by C.

1) **Clips** its local update to sensitivity C:

$$\bar{\Delta w}_i^t = \Delta w_i^t / \max\left\{1, \|\Delta w_i^t\|_2 / C\right\}. \tag{26}$$

2) **Adds Gaussian noise** with adaptive variance:

$$\tilde{\Delta w}_i^t = \bar{\Delta w}_i^t + \mathcal{N}(0, \sigma_t^2 C^2 I_d). \tag{27}$$

The clipping operation ensures that the ℓ_2-sensitivity of the clipped update is bounded:

$$\Delta = C. \tag{28}$$

The noise covariance is $\sigma_t^2 C^2 I$, which means the noise standard deviation per coordinate is $\sigma_t C$. Substituting $\Delta = C$ and $\sigma = \sigma_t C$ into the general Gaussian RDP bound yields

$$\varepsilon_{\text{RDP}}^{(t)}(\alpha) = \frac{\alpha C^2}{2(\sigma_t C)^2} = \frac{\alpha}{2\sigma_t^2}. \tag{29}$$

3) Incorporating the Adaptive Schedule: Our adaptive noise schedule from Algorithm 3 sets

$$\sigma_t = \sigma_0 \cdot \sqrt{1 + \alpha t}, \tag{30}$$

where σ_0 is the base noise multiplier and α controls the growth rate. To avoid notational confusion with the RDP order parameter, we denote the schedule growth rate as κ, so

$$\sigma_t^2 = \sigma_0^2(1 + \kappa t). \tag{31}$$

Substituting equation (31) into the per-round RDP bound, we obtain the privacy guarantee for round t:

$$\varepsilon_{\text{RDP}}^{(t)}(\alpha) = \frac{\alpha}{2\sigma_0^2(1 + \kappa t)}. \tag{32}$$

This completes the per-round privacy analysis of the adaptive perturbation mechanism.

D. Cumulative Privacy Analysis

Having established the per-round RDP guarantee, we now derive the total privacy loss accumulated over all T communication rounds. The additive composition property of RDP (Property 1) states that if each round t satisfies (α, ε_t)-RDP, then the joint mechanism over all rounds satisfies $(\alpha, \sum_t \varepsilon_t)$-RDP.

Applying this property to our adaptive mechanism, the total RDP parameter after T rounds is

$$\varepsilon_{\text{RDP}}^{\text{total}}(\alpha) = \sum_{t=0}^{T-1} \varepsilon_{\text{RDP}}^{(t)}(\alpha) = \sum_{t=0}^{T-1} \frac{\alpha}{2\sigma_0^2(1+\kappa t)} = \frac{\alpha}{2\sigma_0^2} \sum_{t=0}^{T-1} \frac{1}{1+\kappa t}. \tag{33}$$

For notational convenience, we define the cumulative schedule factor:

$$S(T, \kappa) = \sum_{t=0}^{T-1} \frac{1}{1+\kappa t}, \tag{34}$$

which captures the aggregate effect of the time-varying noise schedule. Using this notation, the total RDP guarantee becomes

$$\varepsilon_{\text{RDP}}^{\text{total}}(\alpha) = \frac{\alpha}{2\sigma_0^2} S(T, \kappa). \tag{35}$$

Remark 1. *The cumulative factor $S(T, \kappa)$ can be approximated for large T. When $\kappa > 0$, the sum behaves asymptotically as*

$$S(T, \kappa) \approx \frac{1}{\kappa} \log(1 + \kappa T) + O(1), \tag{36}$$

which grows logarithmically rather than linearly in T. This demonstrates that the adaptive schedule—by increasing noise over time—provides better privacy-utility tradeoffs than fixed-noise schemes, where $S(T, 0) = T$ would grow linearly.

E. Conversion to (ε, δ)-Differential Privacy

To express our privacy guarantee in the standard (ε, δ)-DP framework, we apply Property 3 from Section VI.B. A mechanism satisfying $(\alpha, \varepsilon_{\text{RDP}})$-RDP also satisfies (ε', δ)-DP for any $0 < \delta < 1$, where

$$\varepsilon'(\alpha, \delta) = \varepsilon_{\text{RDP}} + \frac{\log(1/\delta)}{\alpha - 1}. \tag{37}$$

Substituting equation (35) into this conversion formula, the overall (ε, δ)-DP guarantee of our T-round adaptive perturbation mechanism is

$$\varepsilon(\alpha, \delta) = \frac{\alpha}{2\sigma_0^2} S(T, \kappa) + \frac{\log(1/\delta)}{\alpha - 1}. \tag{38}$$

Optimization. For a target failure probability δ, we can optimize over $\alpha > 1$ to find the tightest ε bound:

$$\varepsilon^*(\delta) = \min_{\alpha > 1} \left\{ \frac{\alpha}{2\sigma_0^2} S(T, \kappa) + \frac{\log(1/\delta)}{\alpha - 1} \right\}. \tag{39}$$

This optimization is typically performed numerically. In practice, we search over a grid of α values (e.g., $\alpha \in [1.01, 100]$) to find the minimum.

Interpretation. Equations (35)–(39) together characterize the total privacy loss under our locally applied Gaussian perturbations with adaptive variance. The key insight is that $S(T, \kappa)$ grows sublinearly in T when $\kappa > 0$, yielding tighter privacy bounds compared to fixed-noise schemes where the cumulative privacy cost would be proportional to T. This advantage becomes more pronounced in long training runs, making our adaptive approach particularly suitable for scenarios requiring many communication rounds while maintaining strong privacy guarantees.

F. Confidentiality

The confidentiality of client updates in HEAD-FL is ensured through Paillier homomorphic encryption and Shamir's secret sharing, inherited from FVFL [7]. The Paillier cryptosystem provides semantic security (IND-CPA) under the Decisional Composite Residuosity assumption, ensuring that the server cannot distinguish between encryptions of different updates. The secret key is distributed using (d, n)-threshold Shamir's secret sharing, providing collusion resistance: any coalition of fewer than d clients (potentially including the server) cannot decrypt ciphertexts or learn information about honest clients' updates. We refer readers to [7] for detailed security proofs.

Our adaptive DP mechanism provides an additional layer of protection: while Paillier encryption ensures confidentiality of ciphertexts, the DP noise protects against inference attacks on the decrypted aggregate, ensuring (ε, δ)-DP guarantees as established in Sections VI.B–VI.E.

G. Verifiability

The verifiability guarantee remains identical to FVFL: clients can verify the correctness of aggregation using Shamir's secret sharing and Lagrange interpolation. As the DP noise is added before encryption, it does not interfere with the verification mechanism. For details, we refer to [7].

H. Flexibility

The flexibility of FVFL—supporting client dropouts without affecting aggregation or verification—is preserved in HEAD-FL. The adaptive DP mechanism and FedAvg training do not alter this property. For details, we refer to [7].

VII. Performance Analysis

In this section, we evaluate the proposed HEAD-FL framework across five dimensions: convergence behavior, privacy–utility tradeoff, cumulative privacy budget, communication efficiency, and scalability. All experiments are conducted on the MNIST dataset [**?**], which comprises 60,000 training and 10,000 test samples. The federated system consists of $K = 10$ clients with IID data partitioning. The global model is a convolutional neural network trained over $T = 50$ communication rounds with $E = 5$ local epochs per round, learning rate $\eta = 0.01$, and mini-batch size 64. For the differential privacy mechanism, we set the clipping bound $C = 1.0$, base noise multiplier $\sigma_0 = 3.0$, adaptive schedule growth rate $\kappa = 0.05$, failure probability $\delta = 10^{-5}$, and Rényi order $\alpha = 10$. We compare HEAD-FL against two baselines: Fixed DP-FVFL, which applies a constant noise level $\sigma = 3.0$ across all rounds, and FVFL [7], the original FedSGD-based scheme without differential privacy.

A. Convergence Analysis

Figure 1 presents the test accuracy over 50 communication rounds for HEAD-FL and Fixed DP-FVFL across three noise configurations $\sigma \in \{2.0, 3.0\}$. In all three settings, both methods converge to approximately the same final accuracy, confirming that the adaptive noise schedule does not degrade model utility relative to fixed-noise schemes. The close alignment of both curves across all privacy levels provides empirical support for the theoretical claim that adaptive and fixed DP mechanisms with the same σ_0 yield equivalent utility, while differing in their cumulative privacy expenditure. As expected, higher σ values result in marginally lower final accuracy due to increased noise injection, but the gap between HEAD-FL and Fixed DP-FVFL remains negligible in all cases.

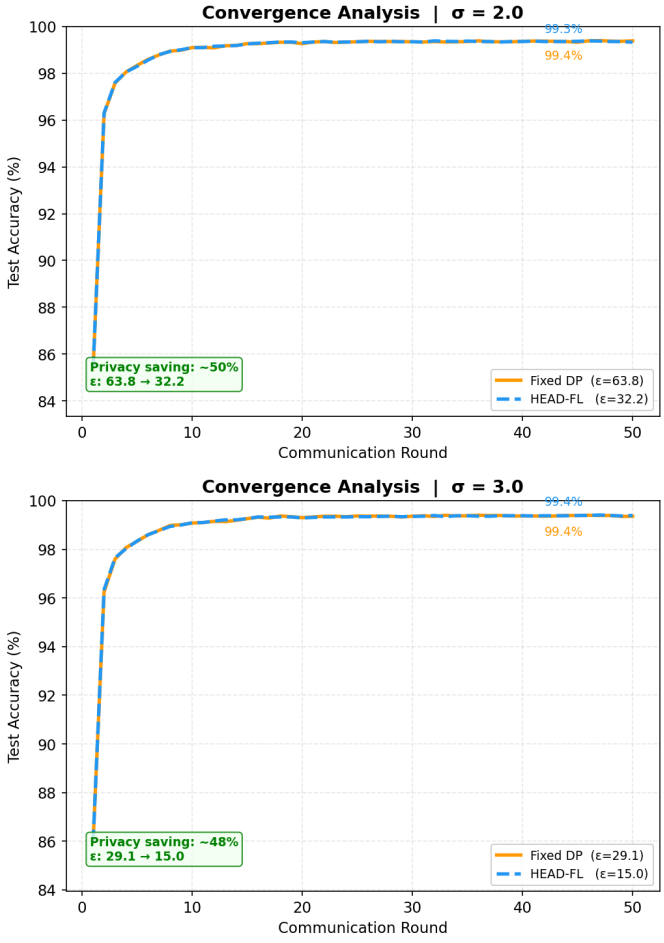

Fig. 1. Convergence analysis of HEAD-FL and Fixed DP-FVFL at $\sigma \in \{2.0, 3.0\}$. Both schemes reach equivalent accuracy across all noise levels, while HEAD-FL achieves approximately 48–51% lower cumulative privacy budget ε in each case.

B. Privacy–Utility Tradeoff

We evaluate the efficiency of HEAD-FL by analyzing the relationship between cumulative privacy cost (ϵ) and model accuracy. As shown in Fig. 2, the fixed-noise baseline accumulates privacy cost linearly, reaching $\epsilon \approx 29$ after 50 rounds. In contrast, HEAD-FL's adaptive noise schedule—where the per-round Renyi Differential Privacy (RDP) contribution decreases as σ_t grows—results in a sublinear accumulation curve that reaches only $\epsilon \approx 15$, a saving of approximately 48%.

Fig. 3 evaluates the efficiency of this expenditure by plotting test accuracy against the cumulative privacy budget (ϵ). While both schemes follow nearly identical accuracy trajectories, indicating that adaptive noise does not degrade learning efficiency, HEAD-FL reaches saturation at a significantly lower cost. The horizontal gap at the 97% accuracy level confirms that HEAD-FL achieves the same peak utility as the baseline while requiring ∼48% less privacy budget, effectively doubling the privacy protection for the same level of utility.

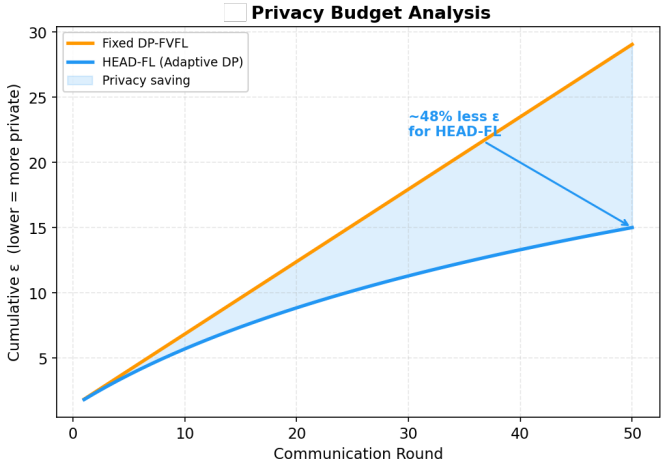

Fig. 2. Cumulative privacy budget ε over communication rounds at $\sigma_0 = 3.0$, $\kappa = 0.05$.

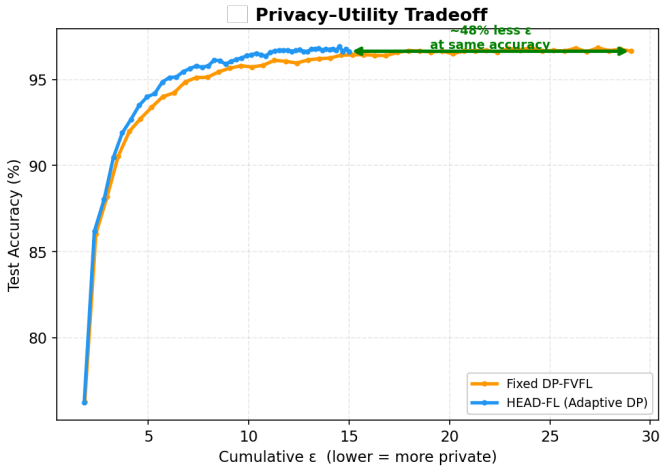

Fig. 3. Privacy–utility tradeoff curve, plotting test accuracy against cumulative ϵ as training progresses round by round.

C. Communication Efficiency

Figure 4 compares the total communication cost of HEAD-FL and FVFL [7] over 50 rounds. FVFL adopts a FedSGD-based aggregation strategy, requiring each client to upload a full gradient vector at every local step, resulting in a total communication cost of approximately 4,350 MB across all clients after 50 rounds. HEAD-FL replaces FedSGD with

FedAvg, where each client uploads the model update once per communication round after completing $E = 5$ local epochs. This reduces the total upload to approximately 870 MB, achieving an $\sim$80% reduction in communication overhead. Importantly, this reduction does not affect the cryptographic security guarantees, as the homomorphic aggregation and verification mechanisms operate on the uploaded model updates regardless of whether FedSGD or FedAvg is used.

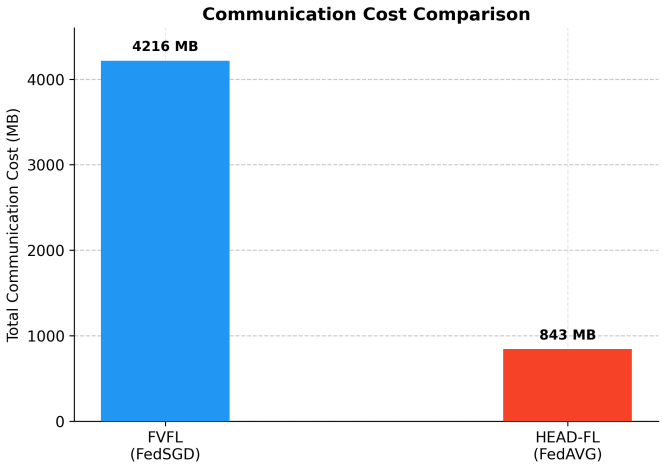

Fig. 4. Total communication cost after 50 rounds. HEAD-FL reduces overhead by $\sim$80% compared to FVFL (FedSGD).

D. Scalability

Figure 5 evaluates the scalability of HEAD-FL and Fixed DP-FVFL as the number of participating clients increases from 10 to 200. Both methods maintain stable accuracy across all client counts, with HEAD-FL consistently matching Fixed DP-FVFL throughout the entire range. The stability of both schemes under increasing client populations confirms that the adaptive DP mechanism and the FedAvg aggregation strategy scale effectively to large federated systems without accuracy degradation. This property is particularly relevant for deployment in bandwidth-constrained environments such as IoT networks and mobile systems, where the number of participating devices may be large and variable.

VIII. Conclusion

In this paper, we proposed HEAD-FL, an enhanced federated learning framework that addresses the key limitation of FVFL [7] — the absence of statistical privacy protection against gradient inversion and membership inference attacks. By integrating an adaptive differential privacy mechanism into the FVFL framework, HEAD-FL provides formal (ε, δ)-DP guarantees that protect client data even when shared gradients are exposed to adversarial analysis. The adaptive noise schedule reduces cumulative privacy expenditure by approximately 48% compared to fixed-noise schemes at equivalent utility, as established through RDP composition analysis and confirmed experimentally. By replacing FedSGD with FedAvg, the framework reduces communication overhead by $\sim$80% while preserving the confidentiality, verifiability, collusion

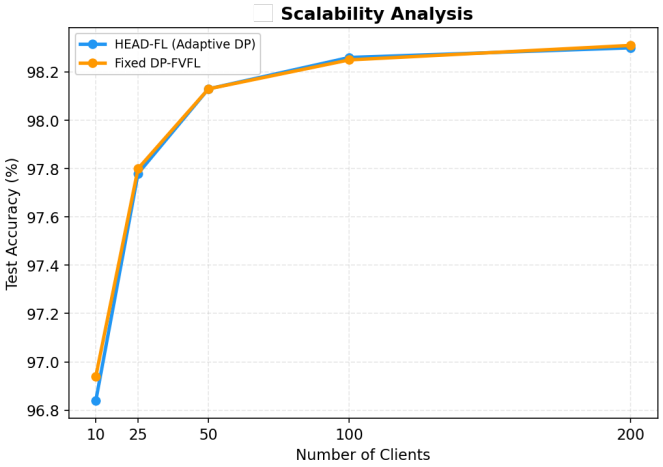

Fig. 5. Test accuracy vs number of clients (10–200) for HEAD-FL and Fixed DP-FVFL.

resistance, and dropout tolerance properties of the original FVFL scheme. HEAD-FL thus provides a unified solution that simultaneously addresses gradient-level privacy, cryptographic security, and communication efficiency for federated learning in privacy-sensitive environments.

References

[1] H. B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas, "Communication-efficient learning of deep networks from decentralized data," in *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2017, pp. 1273–1282.

[2] L. Zhu, Z. Liu, and S. Han, "Deep leakage from gradients," *Advances in neural information processing systems*, vol. 32, 2019.

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

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

[5] E. Bagdasaryan, A. Veit, Y. Hua, D. Estrin, and V. Shmatikov, "How to backdoor federated learning," in *International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2020.

[6] P. Blanchard, R. Guerraoui, and J. Stainer, "Machine learning with adversaries: Byzantine tolerant gradient descent," in *Advances in Neural Information Processing Systems (NeurIPS)*, 2017.

[7] G. Wang, L. Zhou, Q. Li, X. Yan, X. Liu, and Y. Wu, "Fvfl: A flexible and verifiable privacy-preserving federated learning scheme," *IEEE Internet of Things Journal*, vol. 11, no. 13, pp. 23 268–23 281, 2024.

[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] T. Ryffel, A. Trask, M. Dahl, B. Wagner, J. Mancuso, D. Rueckert, and J. Passerat-Palmbach, "A generic framework for privacy preserving deep learning," *arXiv preprint arXiv:1811.04017*, 2018.

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

[11] P. Mohassel and Y. Zhang, "Secureml: A system for scalable privacy-preserving machine learning," in *2017 IEEE symposium on security and privacy (SP)*. IEEE, 2017, pp. 19–38.

[12] J. H. Cheon, A. Kim, M. Kim, and Y. Song, "Homomorphic encryption for arithmetic of approximate numbers," in *International conference on the theory and application of cryptology and information security*. Springer, 2017, pp. 409–437.

[13] H. Kim, S.-H. Kim, J. Y. Hwang, and C. Seo, "Efficient privacy-preserving machine learning for blockchain network," *Ieee Access*, vol. 7, pp. 136 481–136 495, 2019.

[14] X. Ma, J. Ma, H. Li, Q. Jiang, and S. Gao, "Armor: A trust-based privacy-preserving framework for decentralized friend recommendation in online social networks," *Future Generation Computer Systems*, vol. 79, pp. 82–94, 2018.

[15] 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.

[16] N. Carlini, S. Chien, M. Nasr, S. Song, A. Terzis, and F. Tramer, "Membership inference attacks from first principles," in *2022 IEEE symposium on security and privacy (SP)*. IEEE, 2022, pp. 1897–1914.

[17] J. Ye, A. Maddi, S. K. Murakonda, V. Bindschaedler, and R. Shokri, "Enhanced membership inference attacks against machine learning models," in *Proceedings of the 2022 ACM SIGSAC conference on computer and communications security*, 2022, pp. 3093–3106.

[18] H. Hu, Z. Salcic, L. Sun, G. Dobbie, and X. Zhang, "Source inference attacks in federated learning," in *2021 IEEE International Conference on Data Mining (ICDM)*. IEEE, 2021, pp. 1102–1107.

[19] 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.

[20] 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.

[21] M. A. Rahman, T. Rahman, R. Laganière, N. Mohammed, and Y. Wang, "Membership inference attack against differentially private deep learning model." *Trans. Data Priv.*, vol. 11, no. 1, pp. 61–79, 2018.

[22] B. Jayaraman and D. Evans, "Evaluating differentially private machine learning in practice," in *28th USENIX security symposium (USENIX security 19)*, 2019, pp. 1895–1912.

[23] M. Abadi, A. Chu, I. Goodfellow, H. B. McMahan, I. Mironov, K. Talwar, and L. Zhang, "Deep learning with differential privacy," in *Proceedings of the 2016 ACM SIGSAC conference on computer and communications security*, 2016, pp. 308–318.

[24] R. Shokri and V. Shmatikov, "Privacy-preserving deep learning," in *Proceedings of the 22nd ACM SIGSAC conference on computer and communications security*, 2015, pp. 1310–1321.

[25] 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.

[26] M. Naseri, J. Hayes, and E. De Cristofaro, "Toward robustness and privacy in federated learning: Experimenting with local and central differential privacy," *arXiv preprint arXiv:2009.03561*, 2020.

[27] I. Mironov, "Rényi differential privacy," in *2017 IEEE 30th computer security foundations symposium (CSF)*. IEEE, 2017, pp. 263–275.

[28] Y.-X. Wang, B. Balle, and S. P. Kasiviswanathan, "Subsampled rényi differential privacy and analytical moments accountant," in *The 22nd international conference on artificial intelligence and statistics*. PMLR, 2019, pp. 1226–1235.

[29] V. Feldman, I. Mironov, K. Talwar, and A. Thakurta, "Privacy amplification by iteration," in *2018 IEEE 59th Annual Symposium on Foundations of Computer Science (FOCS)*. IEEE, 2018, pp. 521–532.

[30] B. Balle, G. Barthe, and M. Gaboardi, "Privacy amplification by subsampling: Tight analyses via couplings and divergences," *Advances in neural information processing systems*, vol. 31, 2018.

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

[32] R. Bassily, O. Thakkar, and A. Guha Thakurta, "Model-agnostic private learning," *Advances in neural information processing systems*, vol. 31, 2018.

[33] 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.

[34] H. B. McMahan, D. Ramage, K. Talwar, and L. Zhang, "Learning differentially private recurrent language models," *arXiv preprint arXiv:1710.06963*, 2017.