FedShield-LLM: A Secure and Scalable Federated Fine-Tuned Large Language Model

Md Jueal Mia and M. Hadi Amini, *Senior Member, IEEE*

***Abstract*—Federated Learning (FL) offers a decentralized framework for training and fine-tuning Large Language Models (LLMs) by leveraging computational resources across organizations while keeping sensitive data on local devices. It addresses privacy and security concerns while navigating challenges associated with the substantial computational demands of LLMs, which can be prohibitive for small and medium-sized organizations. FL supports the development of task-specific LLMs for cross-silo applications through fine-tuning but remains vulnerable to inference attacks, such as membership inference and gradient inversion, which threaten data privacy. Prior studies have utilized Differential Privacy (DP) in LLM fine-tuning, which, despite being effective at preserving privacy, can degrade model performance. To overcome these challenges, we propose a novel method, FedShield-LLM, that uses pruning with Fully Homomorphic Encryption (FHE) for Low-Rank Adaptation (LoRA) parameters, enabling secure computations on encrypted model updates while mitigating the attack surface by deactivating less important LoRA parameters. Furthermore, optimized federated algorithms for cross-silo environments enhance scalability and efficiency. Parameter-efficient fine-tuning techniques like LoRA substantially reduce computational and communication overhead, making FL feasible for resource-constrained clients. Experimental results show that the proposed method outperforms existing methods while maintaining robust privacy protection, enabling organizations to collaboratively train secure and efficient LLMs.**

The code and data are available at https://github.com/solidlabnetwork/fedshield-llm.

***Index Terms*—Federated Learning, Large Language Models, Security, Cross-Silo Applications, Fully Homomorphic Encryption, Low-Rank Adaptation.**

I. Introduction

LARGE Language Models (LLMs) such as GPT [8], PaLM [11], and ChatGPT [31] have revolutionized natural language processing by achieving state-of-the-art results across a wide array of language understanding and generation tasks. These foundation models, trained on massive public datasets, exhibit exceptional performance in domains like dialogue generation, summarization, and question answering. However, as LLMs are increasingly fine-tuned for specific domains such as healthcare and finance, they must incorporate sensitive or proprietary data, raising critical privacy and compliance concerns. Regulations like the GDPR and CCPA restrict data sharing and impose legal obligations on organizations handling private information [3, 7, 12]. Moreover, recent studies show that LLMs can memorize and leak training data [9], which presents significant risks when models are trained or fine-tuned on confidential inputs. This makes it imperative to develop solutions that preserve data locality and ensure privacy during domain adaptation.

Training and fine-tuning LLMs also pose substantial computational challenges. Models like GPT-3 require hundreds of GPU-years and millions of dollars in compute resources [22], limiting accessibility to large tech firms. Even with the release of open-source LLMs like LLaMA [37], resource bottlenecks remain, especially for smaller organizations or in distributed collaborative scenarios. Traditional full-model fine-tuning methods involve transmitting large model updates or datasets, which can be impractical in resource-constrained environments typically found in centralized training scenarios. Federated Learning (FL) [28] addresses these challenges by enabling decentralized training, where clients perform local updates on private data and only share model gradients or parameters with a central server. FL supports privacy by design and allows organizations to jointly train models without centralizing data. FL has been explored in resource-constrained and edge scenarios [20, 23], and recent works like OpenFedLLM [43] and FederatedScope-LLM [21] have shown that FL can be adapted for LLM instruction tuning and domain-specific applications.

However, standard FL remains vulnerable to efficiency and security issues. Transmitting full-model updates for large models is bandwidth-intensive and often infeasible. Additionally, model updates can leak private information via inference or gradient inversion attacks (GIA) [16, 51]. To address the communication overhead, researchers have proposed parameter-efficient tuning methods such as Low-Rank Adaptation (LoRA) [19], which inserts

Md Jueal Mia and M. Hadi Amini are with the Knight Foundation School of Computing and Information Sciences, Florida International University, Miami, FL 33199 USA. They are also with Security, Optimization, and Learning for InterDependent networks laboratory (solid lab) (e-mail: mmia001@fiu.edu; moamini@fiu.edu).

low-rank matrices into transformer layers to reduce trainable parameters. LoRA enables clients to only update lightweight adapter layers instead of the full model, drastically reducing computation and communication costs. This makes LoRA highly compatible with federated fine-tuning frameworks [38]. Nonetheless, privacy concerns persist even with reduced update sizes, as model updates can still be exploited by malicious servers or compromised clients.

To ensure end-to-end privacy in federated LLM training, several advanced privacy-preserving techniques have been proposed. Differential Privacy (DP) [13] introduces noise to model updates to provide statistical guarantees but often compromises model accuracy, especially in high-dimensional settings like LLMs [27, 42]. Fully Homomorphic Encryption (FHE) [10] enables computation over encrypted data, offering stronger confidentiality without requiring noise, though at the cost of increased computation. Secure Multi-Party Computation (MPC) [6] enables joint computation without revealing data but also faces scalability limitations. Meanwhile, model pruning [39] can improve efficiency and mitigate attack surfaces by eliminating redundant parameters. Although most prior privacy-preserving LLM frameworks focus on inference-time security in centralized settings, they offer complementary insights relevant to secure model design. For instance, PrivacyAsst [50] leverages HE to safeguard sensitive user inputs during tool-based LLM inference, illustrating how encrypted computation can be integrated into LLM workflows at runtime. Similarly, InferDPT [36] applies local DP to perturb user prompts submitted to black-box LLMs and reconstructs coherent outputs using a local extraction model. Building on these insights, we propose **FedShield-LLM**, a novel federated fine-tuning framework that combines FL, LoRA, FHE, and pruning to ensure scalable, secure, and regulation-compliant training of LLMs across decentralized private datasets. In this work, we detail the design of FedShield-LLM and demonstrate its effectiveness in achieving high performance with strong privacy guarantees. The main contributions of our study are listed below.

- We propose and implement a novel federated fine-tuning mechanism, **FedShield-LLM**, for LLMs, which integrates FHE with model pruning. To the best of our knowledge, this is the first exploration of combining FHE with pruning in federated LLM fine-tuning to mitigate vulnerabilities during both the training and inference phases. Our approach enhances security against inference attacks in adversarial settings including an honest-but-curious server.
- FedShield-LLM minimizes computational and memory overhead by updating only LoRA adapter layers instead of full model parameters. This design makes secure fine-tuning feasible in resource-constrained environments such as edge devices.
- We validate the effectiveness of FedShield-LLM through extensive experiments using the base models *meta-llama/Llama-2-7b-hf* and *meta-llama/Llama-2-13b-hf* across four datasets: *medalpaca/medical_meadow_medical_flashcards*, *vicgalle/alpaca-gpt4*, *TIGER-Lab/MathInstruct*, and *FinGPT/fingpt-sentiment-train*. Results demonstrate that our framework consistently outperforms existing methods in terms of both model performance and security across various domains.

The remainder of this paper is structured as follows: Section II provides a comprehensive review of the related literature. Section III introduces the preliminaries essential for understanding the proposed method. Section IV details the proposed FedShield-LLM methodology, followed by Section V, which presents the experimental setup and results. Section VI offers further discussion, including limitations and implications. Finally, Section VII concludes the paper and outlines directions for future work.

II. Literature Review

Recent advancements in machine learning have sparked significant interest in combining FL with LLMs to enable collaborative model development while preserving data privacy. Researchers have explored various approaches to make this integration practical and secure, including federated and distributed fine-tuning strategies, differential privacy techniques to safeguard individual data contributions, homomorphic encryption to ensure confidentiality during model aggregation, and secure multi-party computation to prevent information leakage during collaborative training. While each of these methods contributes valuable capabilities, they also come with limitations highlighting the need for a comprehensive framework like FedShield-LLM that unifies efficiency, privacy, and robustness.

Recent works have introduced efficient frameworks for distributed and federated LLM adaptation. This includes leveraging decentralized private data across user devices without transferring it to a central server, which enhances privacy and communication efficiency in large-scale LLM adaptation [1]. OpenFedLLM [43] proposes a comprehensive pipeline integrating federated instruction tuning and value alignment for adapting models like LLaMA2-7B across multiple clients without data centralization. It supports various FL strategies, including FedAvg, and demonstrates that federated fine-tuned models

can match the performance of central LLMs like GPT-4 on specialized tasks. FedBiOT [40] addresses client-side limitations using a bi-level optimization scheme in which clients fine-tune compressed LLM emulators (lightweight LoRA modules), while servers maintain alignment with full models. This drastically reduces local resource demands. FederatedScope-LLM [21] is a modular open-source framework that provides unified support for various LLM FL scenarios, including adapter-based tuning and diverse FL algorithms. Separately, FlexLoRA [2] proposes an adaptive LoRA strategy in heterogeneous FL environments: each client trains with a custom LoRA rank, and the server merges these updates using singular value decomposition (SVD) to synthesize a full-rank global model. This avoids bottlenecks from underpowered clients and improves performance across diverse clients and tasks. FedCoLLM [14] further extends this line by co-tuning LLMs and smaller SLMs using LoRA, Secure Aggregation [5], and Knowledge Distillation [18], achieving less than 0.25% of full model communication while preserving performance. Yun et al. [47] introduce a hierarchical clustered sampling method to address non-IID data, combining within-cluster aggregation and multinomial participation to improve fairness and stability. A recent survey [41] categorizes distributed fine-tuning strategies like knowledge distillation and split learning, which offer trade-offs between privacy, performance, and communication. While these frameworks enable distributed LLM adaptation, most assume honest clients and servers, and lack robust protections against inference attacks.

DP has been widely explored for protecting data contributions in FL. DP-LoRA [27] applies Gaussian noise to LoRA adapter weights, enabling formal (ϵ, δ) privacy guarantees while maintaining model accuracy. This works well due to LoRA's compressed structure, which reduces the noise scale required. DP-DyLoRA [42] extends this by dynamically adjusting adapter ranks, integrating rank-sensitive noise mechanisms to further optimize the privacy-utility tradeoff. Cross-domain evaluations (e.g., speech, vision, text) show that DP-DyLoRA maintains $< 2\%$ loss in performance even at $\epsilon = 2$ with one million clients. Yu et al. [44] found that parameter-efficient fine-tuning (PEFT) techniques (e.g., adapters) inherently offer stronger DP tradeoffs versus full fine-tuning. Despite these innovations, DP does not prevent poisoning attacks or information leakage through the model's structure and gradients, suggesting a need for complementary cryptographic protections.

HE offers an orthogonal privacy defense by allowing secure aggregation of encrypted model updates. Frameworks such as PrivTuner [24] a centralized fine-tuning framework that integrates FHE with LoRA to enable secure and efficient fine-tuning of AI foundation models. Unlike FL, which relies on decentralized training, PrivTuner ensures data privacy by performing computations on encrypted client data directly on the server, eliminating the need for raw data transmission while maintaining model performance. FHE offers robust security against inference attacks during LLM fine-tuning in an FL environment. However, security challenges can arise in cross-silo FL scenarios, particularly when the server evaluates the performance of the LLM. In such cases, honest-but-curious clients or servers may attempt to infer sensitive information by analyzing the LoRA parameters, which contain knowledge learned from private data.

MPC protocols such as secret sharing and garbled circuits enable joint computation without revealing private data. In FL, systems like S++ [32] and SecureML [30] show that secure aggregation and backpropagation are feasible under encryption. Google's secure aggregation protocol [6] masks individual updates so only the final sum is revealed to the server. These systems strengthen client privacy but suffer from high latency and are hard to scale to models with billions of parameters.

Unlike DP-LoRA, which uses noise to guarantee privacy at some cost to utility, our method avoids accuracy loss by using FHE for exact aggregation. Moreover, whereas PrivTuner applies FHE+LoRA in a single-server setting, FedShield-LLM extends this to a federated multi-client setting and introduces pruning to further reduce risk and overhead. It integrates LoRA-based efficient fine-tuning with FHE and model pruning: LoRA minimizes the size of model updates, making FHE computationally feasible, while pruning compresses the model and reduces the attack surface. Together, these techniques form a cohesive framework for secure, efficient, and robust collaborative fine-tuning of LLMs in adversarial environments.

III. Preliminaries

A. Federated Fine-Tuning of LLMs

FL enables a set of N clients $\{C_1, C_2, \ldots, C_N\}$ to collaboratively fine-tune a shared global LLM with parameters w_t at round t, using decentralized datasets $\{D_1, D_2, \ldots, D_N\}$, without sharing raw data. Each client C_i receives the current global model w_t and performs local training to compute a LoRA-based model update Δw_i, which represents the client-specific parameter change:

$$\Delta w_i := \arg\min_{\Delta w} L_i\big(f_{w_t} + \Delta w, D_i\big), \tag{1}$$

where L_i is the local loss computed over dataset D_i. After local updates, the server aggregates the received client updates using a method such as FedAvg [28].

Assuming equal weight for each client, the global model is updated as:

$$w_{t+1} := w_t + \frac{1}{N}\sum_{i=1}^{N} \Delta w_i, \tag{2}$$

and the updated model w_{t+1} is redistributed to clients for the next communication round.

In practice, transmitting full gradients or model updates is resource-intensive, especially for large LLMs. To mitigate this, recent FL methods adopt PEFT strategies such as LoRA [19], which restrict training to low-dimensional subspaces within transformer layers. In LoRA, the update Δw_i for each weight matrix is represented as a low-rank factorization: $\Delta w_i = A_i \cdot B_i$, where $A_i \in \mathbb{R}^{d\times r}$ and $B_i \in \mathbb{R}^{r\times k}$. Since $r \ll \min(d, k)$, this reduces the number of trainable parameters

B. Threat Model

Even though raw data never leaves local devices in FL, the exchanged model updates Δw_i can leak sensitive information about client C_i's dataset D_i. We consider an inference attack threat model wherein an adversary observes gradients or model updates and attempts to infer private data. The adversary can be an honest-but-curious server (a semi-honest aggregator that faithfully executes FL but analyzes received updates for information) or dishonest clients that deviate from the protocol. For example, a curious central server could try to invert gradients to reconstruct a client's training examples. Attacks such as Deep Leakage from Gradients (DLG) [51] and other GIA [15] have demonstrated that given Δw_i, an attacker can often reconstruct the original inputs used to compute that update. Similarly, an adversary might perform membership inference to determine if a certain sample was in D_i.

We can formalize the privacy breach by the probability $\Pr[\mathcal{A}(\Delta w_i) = x]$ that an adversary $\mathcal{A}$, given model update Δw_i, correctly infers a private data point $x \in D_i$. An effective attack means this probability is much higher than random chance (i.e., $\gg 1/|X|$ for input domain X):

$$\Pr\left[\mathcal{A}(\Delta w_i) = x\right] \gg \frac{1}{|X|}\ , \tag{3}$$

indicating a significant privacy risk.

To mitigate such threats, a strong privacy-preserving approach is the cryptographic encryption of model updates using FHE. FHE schemes—such as CKKS, based on the Ring Learning With Errors (Ring-LWE) problem [10]; enable computations directly over encrypted data. In the context of FL, each client encrypts its model update Δw_i and sends it to the server, which then performs aggregation (e.g., summing or averaging) without ever decrypting the individual updates. The result is a ciphertext of the aggregated model, which can then be decrypted by an authorized party. This process ensures that even an honest-but-curious server or external adversary cannot infer any client's private information from the model updates. Since FHE prevents direct access to individual gradients or parameter updates, it is highly effective at mitigating gradient inversion attacks and membership inference threats, offering strong confidentiality guarantees in adversarial FL settings.

C. Applications

Secure federated fine-tuning of LLMs is highly valuable for real-world scenarios where data is sensitive, proprietary, or subject to privacy regulations. Below we highlight a few domains and use-cases:

1) Healthcare: Hospitals and clinics can collaboratively fine-tune an LLM on electronic health records (EHR) to build medical question-answering systems or decision support tools, while complying with strict privacy laws like HIPAA and GDPR. For example, federated training on patient records can enable a model to learn medical knowledge without any hospital sharing raw data. Prior works have shown the feasibility of FL for clinical natural language tasks, achieving near-centralized performance while preserving privacy [48].

2) Finance: Banks and financial institutions may jointly train LLMs on sensitive data such as transaction logs, fraud detection alerts, or risk assessment reports without exposing their proprietary data to competitors. Federated fine-tuning allows learning from a wider data pool (e.g. several bank's records) to improve models for financial analysis or credit scoring, all while keeping each institution's data siloed. Studies indicate that FL can be applied in the financial domain to enhance models like fraud detectors or credit risk predictors, with appropriate privacy safeguards in place [25].

3) Autonomous Vehicles (AV):: FL enables intelligent vehicles to collaboratively train perception and decision-making models by sharing learned representations from diverse driving contexts such as urban intersections, expressways, and varying weather conditions without transmitting raw sensor data (e.g., LiDAR, radar, camera feeds) to a centralized server. This paradigm preserves user privacy and enhances model robustness across edge devices. Federated approaches have been shown to support critical AV tasks such as lane detection, obstacle avoidance, and traffic sign recognition [49]. The integration of LLMs into AV systems via federated fine-tuning may further enable natural language reasoning and semantic understanding of complex driving commands, improving both safety and human–machine interaction under strict privacy constraints.

TABLE I: Notation

Notation	Representation
N	Total number of clients in the FL setup
C_i	Client i participating in federated training
D_i	Local dataset of client C_i
D	Set of all local datasets
$x \in D_i$	Private data sample from client C_i
X	Input domain of the training data
p_i	Client aggregation weight, e.g., $p_i = \frac{n_i}{\sum_j n_j}$
w_t	Global model parameters at round t
w_i	Local model initialized with global model w_t on client i
Δw_i	LoRA update from client i
Δw_i^p	Pruned LoRA update of client i
$\bar{w}_t$	Aggregated model update at round t
A_i, B_i	Low-rank matrices where $\Delta w_i = A_i B_i$
d, k, r	Dimensions of $A_i \in \mathbb{R}^{d\times r}$, $B_i \in \mathbb{R}^{r\times k}$
P	Total number of parameters in the LLM
P_{LoRA}	Number of trainable LoRA parameters
m_i	Pruning mask for client i's LoRA parameters
t	Current communication round
R	Total number of communication rounds
L_i	Local loss function of client C_i
e_i^r	Pruning error introduced at round r for client i
$\mathcal{A}$	Adversarial algorithm or attacker
$\Pr[\mathcal{A}(\Delta w_i) = x]$	Probability adversary reconstructs private data x from Δw_i
FHE	Fully Homomorphic Encryption
CKKS	Homomorphic encryption scheme based on Ring-LWE
$\mathcal{C}$	Encryption context used in CKKS
n_c	Number of ciphertexts in CKKS encryption
N_{poly}	Polynomial degree in CKKS ciphertexts
$F(w)$	Global objective function in FL
$F_i(w)$	Local objective function of client i
η	Learning rate
F^*	Optimal (minimum) value of the objective $F(w)$

IV. Methodology

A. Method

Fine-tuning LLMs in federated environments presents challenges related to data privacy, computational efficiency, and scalability. To address these issues, FedShield-LLM integrates FHE for secure computations and unstructured pruning to enhance model security by limiting the attack surface. Additionally, LoRA is employed to reduce the number of trainable parameters during fine-tuning, minimizing computational and memory demands for resource-constrained clients. Based on the FL LLM fine-tuning process from the OpenFedLLM framework [43], our mechanism ensures a secure and efficient fine-tuning process, as illustrated in Figure 1 and detailed in Algorithm 1. All notations are presented in Table I.

As part of the FL task, we integrate FedIT to improve the instruction-following capabilities of LLMs. FedIT enables each client to fine-tune its local model using instruction-response pairs from private datasets, ensuring the model learns to generate responses that align precisely with given instructions. This supervised fine-tuning process enhances the LLM's ability to handle diverse instructions while maintaining strict privacy standards by decentralizing and securing client data [43].

The server begins by distributing the global model w_t to the selected clients. Each client synchronizes the global model with their local model and fine-tunes it on their private dataset D_i using LoRA. LoRA reduces computational and memory demands by updating only the LoRA parameters. The LoRA parameter update is expressed as:

$$\Delta w_i = A_{i,t} \cdot B_{i,t}, \tag{4}$$

where $A_{i,t} \in \mathbb{R}^{d\times r}$ and $B_{i,t} \in \mathbb{R}^{r\times d}$ represent the low-rank matrices for the client i at time t.

After extracting the trainable LoRA parameters, clients apply L1 unstructured pruning to enhance model efficiency and security. This technique identifies and deactivates less significant weights in the LoRA parameters by generating binary masks based on the magnitude of each weight. Binary mask is generated based on the pruning rate as stated in [29]. Specifically, the pruning mask m_i is computed as:

$$m_i \leftarrow \text{mask}(\Delta w_i, p_t, \text{L1 norm}),$$

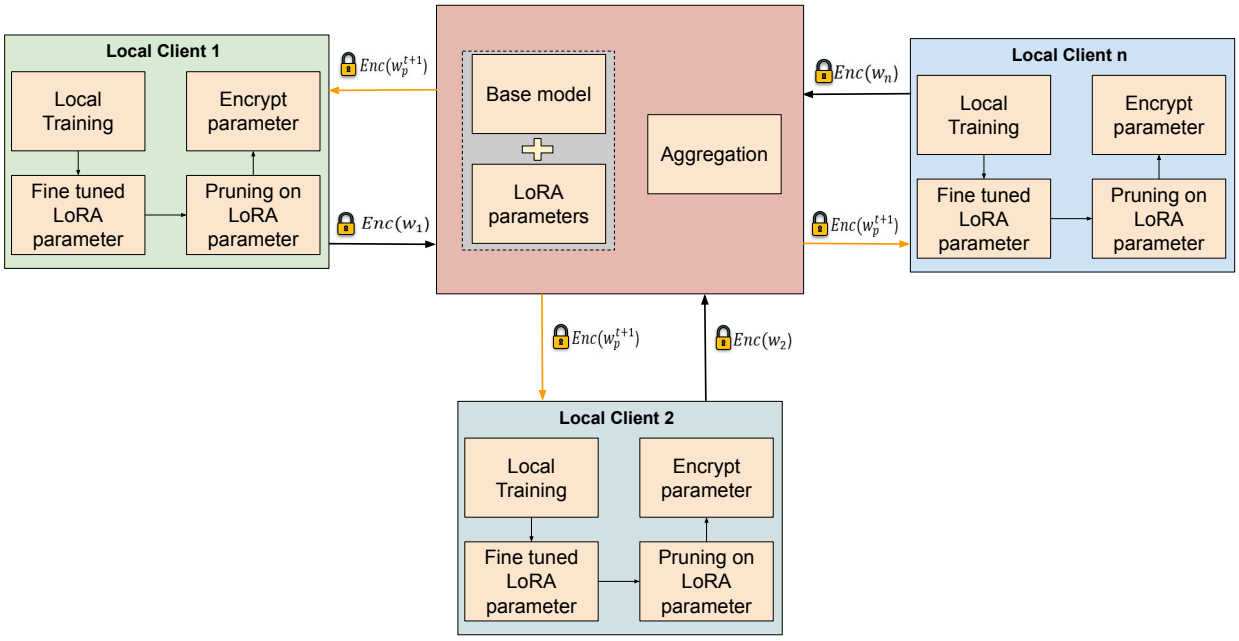

Fig. 1: Overview of our proposed framework.

where Δw_i represents the LoRA parameter updates, p_t is the pruning rate at time t, and the mask function zeroes out weights with the smallest L1 norm, retaining only the most important parameters. The pruned updates Δw_i^p are then computed by applying the mask:

$$\Delta w_i^p \leftarrow \Delta w_i \odot m_i,$$

where $\odot$ denotes element-wise multiplication. This process ensures that smaller, less impactful weights are set to zero, focusing computational resources on the most influential parameters.

To ensure secure communication, clients encrypt their pruned LoRA parameters using the CKKS encryption scheme as:

$$\Delta w_i^p \leftarrow \text{Enc}(\Delta w_i^p, \mathcal{C}),$$

where $\mathcal{C}$ denotes the encryption context. This approach enables computations to be carried out directly on encrypted parameters, safeguarding sensitive information throughout the transmission and aggregation processes. The encrypted updates are then sent to the server, where a secure aggregation method, such as FedAvg, is employed to update the global LoRA parameters without requiring decryption. The LoRA parameters are aggregated as:

$$\bar{w}_t \leftarrow \frac{1}{|n_t|} \sum_{i \in n_t} \Delta w_i^p,$$

where $\bar{w}_t$ represents the aggregated global parameters at time t, n_t is the set of participating clients, and Δw_i^p are the pruned and encrypted updates from client i. After aggregation, the server decrypts the aggregated updates to obtain the global LoRA parameters $\bar{w}_t$, which are then applied to the global model. This updated global model is redistributed to clients for the next round of training.

Algorithm 1 represents the secure distributed LLM fine-tuning process. This process fine-tunes only the parameters based on LoRA. After the final round of communication, the updated LoRA parameters are merged with the base model. This updated base model can then be used as task specific LLM.

The proposed methodology demonstrates how secure, scalable, and efficient FL can be achieved by integrating FHE with pruning and LoRA. Experimental results validate its ability to maintain competitive performance while preserving strict privacy standards, making it suitable for sensitive applications in domains like healthcare, finance, and autonomous systems. The high level overview of our framework is visualized in Figure 2.

B. Convergence Analysis

We provide a convergence guarantee for the FedShield-LLM training algorithm. In essence, FedShield-LLM performs an encrypted and sparsified variant of the standard FedAvg procedure [28]. The added encryption does not alter the numerical update values, and the pruning step can be viewed as a form of gradient sparsification, which has been shown to retain

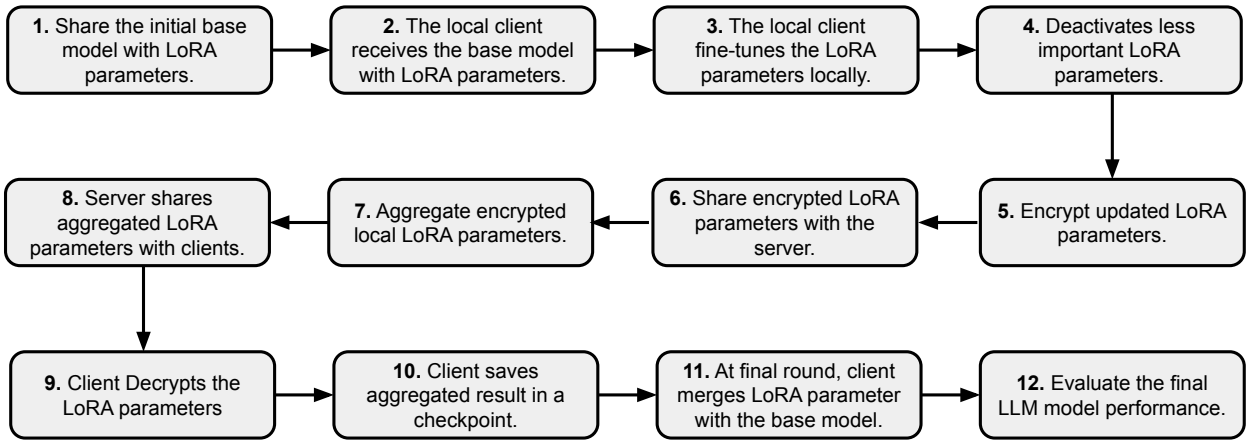

Fig. 2: High-level overview of the proposed mechanism.

convergence properties under certain conditions [23]. Below we formalize the convergence result.

Theorem 1 (Convergence of FedShield-LLM)**.** *Let $F(w) = \sum_{i=1}^{N} p_i F_i(w)$ be the global objective, where $F_i(w)$ is the local loss on client i and p_i is a weighting factor (e.g., $p_i = \frac{n_i}{\sum_j n_j}$ for local dataset size n_i). Suppose each F_i is L-smooth and bounded below, and that stochastic gradients have bounded variance. Assume FedShield-LLM uses (1) LoRA-based local updates $\Delta w_i^r = A_i^r B_i^r$, (2) dynamic unstructured pruning with bounded error $\|e_i^r\| \leq \epsilon_r$, and (3) secure aggregation via CKKS encryption. If the learning rate η is sufficiently small, then after R rounds of communication,*

$$\min_{0 \leq r < R} \mathbb{E}\left[\|\nabla F(w^r)\|^2\right] \leq \frac{C}{\sqrt{R}},$$

for some constant $C > 0$. Therefore, FedShield-LLM converges sublinearly to a stationary point of $F(w)$.

Proof Sketch. Our proof builds on the classical convergence analysis of federated SGD [28] and techniques from compressed (sparsified) distributed optimization [23].

Step 1: Baseline dynamics. In the absence of pruning and encryption, FedShield-LLM reduces to FedAvg. In round r, each selected client computes a LoRA update $\Delta w_i^r = A_i^r B_i^r$ and sends it to the server. The server aggregates updates $\bar{\Delta}^r = \sum_i p_i \Delta w_i^r$ and updates the model: $w^{r+1} = w^r + \eta \bar{\Delta}^r$. Under smoothness and bounded variance, this update rule is known to converge at a rate of $O(1/\sqrt{R})$ for non-convex objectives [33, 45].

Step 2: Effect of encryption. CKKS encryption allows the server to compute $\bar{\Delta}^r$ homomorphically, without accessing plaintexts. Since encryption preserves arithmetic operations, the effective update remains unchanged. Thus, secure aggregation does not impact convergence behavior.

Step 3: Impact of pruning. After computing Δw_i^r, each client applies a mask $m_i^r = \text{mask}(\Delta w_i^r, p_t, \text{L1 norm})$ that retains the top-p_t entries (by magnitude), yielding the sparse update $\Delta w_{i,\text{prune}}^r = \Delta w_i^r \circ m_i^r$. Let $e_i^r = \Delta w_i^r - \Delta w_{i,\text{prune}}^r$ denote the pruning error. The server aggregates these pruned updates:

$$\bar{\Delta}^r = \sum_i p_i \Delta w_{i,\text{prune}}^r = \sum_i p_i \Delta w_i^r - \sum_i p_i e_i^r.$$

Hence, the model update becomes: $w^{r+1} = w^r + \eta \bar{\Delta}^r = w^r + \eta \sum_i p_i \Delta w_i^r - \eta \sum_i p_i e_i^r$.

Step 4: Smoothness-based descent. By L-smoothness of F, we have

$$F(w^{r+1}) \leq F(w^r) + \eta \langle \nabla F(w^r), \bar{\Delta}^r \rangle + \frac{L\eta^2}{2} \|\bar{\Delta}^r\|^2.$$

Substituting $\bar{\Delta}^r = \sum_i p_i \Delta w_i^r - \sum_i p_i e_i^r$, we get:
$\mathbb{E}[F(w^{r+1}) - F(w^r)] \leq -\eta \|\nabla F(w^r)\|^2 + \eta \|\nabla F(w^r)\| \cdot \|\bar{e}^r\| + \text{(higher-order terms)}$.
Here, $\bar{e}^r = \sum_i p_i e_i^r$ and $\|\bar{e}^r\| \leq \epsilon_r$ by assumption.

Step 5: Bounded error and telescoping. Since pruning error ϵ_r is small and gradually increases with the pruning schedule

$$p_t = \max\left(0, \frac{t - t_{\text{eff}}}{t_{\text{target}} - t_{\text{eff}}}\right) \cdot (p_{\text{target}} - p_0) + p_0,$$

the additional error terms do not outweigh the descent. Summing over r and using telescoping sums yields:
$\frac{1}{R} \sum_{r=0}^{R-1} \mathbb{E}[\|\nabla F(w^r)\|^2] \leq \frac{2[F(w^0) - F^*]}{\eta R} + \text{(bounded error terms)}$.
This implies $\min_{0 \leq r < R} \mathbb{E}[\|\nabla F(w^r)\|^2] = \mathcal{O}(1/\sqrt{R})$. □

Remark 1. *The convergence guarantee applies in the general non-convex setting, which is appropriate for fine-tuning deep neural networks (including LLMs). If*

Algorithm 1 FedShield-LLM

Input: N: Number of clients, R: Number of rounds, D: Local datasets, w_0: Initial global model, C: CKKS encryption context, p_0: Initial pruning rate, p_{target}: Maximum pruning rate, t_{eff}: Pruning start round, t_{target}: Round for maximum pruning
Output: w_T: Final global model, L: Training loss history
Initialize global model w_0
Set up CKKS encryption context C
for $t = 1$ **to** R **do**
  Select a subset of clients $n_t \subseteq N$
  Compute pruning rate: $p_t \leftarrow \max\left(0, \frac{t - t_{\text{eff}}}{t_{\text{target}} - t_{\text{eff}}}\right) \cdot (p_{\text{target}} - p_0) + p_0$
  for each client $i \in n_t$ **do**
    Synchronize global model w_t to local model w_i
    Load client dataset D_i
    Fine-tune w_i using LoRA for 1 epoch on D_i
    Extract LoRA parameters $\Delta w_i = A_i \cdot B_i$
    Calculate mask: $m_i \leftarrow \text{mask}(\Delta w_i, p_t, \text{L1 norm})$
    Sparsed model updates: $\Delta w_i^p \leftarrow \Delta w_i \odot m_i$
    Encrypt Δw_i^p using CKKS encryption
  end for
  Aggregate encrypted updates: $\bar{w}_t \leftarrow \frac{1}{|n_t|} \sum_{i \in n_t} \Delta w_i^p$
  Decrypt $\bar{w}_t$ to obtain the aggregated update
  Update global model: $w_{t+1} \leftarrow w_t + \bar{w}_t$
  Evaluate global model and compute training loss
  Save model checkpoint periodically
end for
return w_T: Final global model, L: Training loss history

additional structure is assumed—such as μ-strong convexity of $F(w)$—the convergence rate improves to linear. In practice, our experiments show that FedShield-LLM achieves fast empirical convergence, with sharper loss reduction in early rounds compared to Vanilla-FL. The encryption layer ensures secure aggregation without affecting numerical optimization, while the pruning schedule preserves model quality by gradually increasing sparsity.

C. Security and Robustness Analysis of FedShield-LLM

Theorem 2 (Security and Robustness of FedShield--LLM)**.** *Assuming the semantic security of CKKS under the RLWE assumption and given that each client's LoRA update is sparsified via unstructured pruning with rate p_t, the FedShield-LLM protocol protects against inference attacks (e.g., gradient inversion) from both passive adversaries and colluding servers, with negligible advantage for any polynomial-time attacker.*

Proof. **(1) Semantic Security via CKKS:** Each client encrypts its sparsified LoRA update $\Delta w_i^p = \Delta A_i \cdot \Delta B_i$ using the CKKS encryption scheme before transmission. CKKS operates over the polynomial ring $R_q = \mathbb{Z}_q[x]/(x^N + 1)$, with a secret key $s \leftarrow \chi$ and a public key $\mathsf{pk} = (b, a)$ where $b = -a \cdot s + e \pmod q$. The message is first scaled and encoded, and then encrypted as:

$$c_0 = b \cdot u + e_1 + m', \quad c_1 = a \cdot u + e_2$$

The server aggregates encrypted client updates homomorphically and holds only the public key. A trusted party with the secret key performs decryption of the final aggregated ciphertext. Under the RLWE assumption, CKKS ensures IND-CPA security:

$$\text{Enc}(\Delta) \approx_c \text{Enc}(\Delta') \quad \Rightarrow \quad \Pr[\mathcal{A} \text{ distinguishes}] \leq \text{negl}(\lambda)$$

Thus, ciphertexts reveal no meaningful information to adversaries, including honest-but-curious servers.

(2) Robustness from Sparsified Aggregation: Each client applies a binary pruning mask $m_i \in \{0,1\}^d$ to their LoRA update, yielding:

$$\tilde{\Delta w_i} = m_i \odot \Delta w_i, \quad \text{with } \|m_i\|_0 = (1 - p_t)d$$

After homomorphic aggregation and decryption, the server observes only the combined sparse update:

$$\tilde{\Delta w}_{\text{agg}} = \sum_{i \in \mathcal{C}_t} m_i \odot \Delta w_i$$

This results in an underdetermined system, where:

$$\exists \{\Delta w_1', ..., \Delta w_N'\} \neq \{\Delta w_1, ..., \Delta w_N\} : \sum_i m_i \odot \Delta w_i' = \tilde{\Delta w}_{\text{agg}}$$

Due to:

- **Sparsity**, and
- **Large client set size** ($N \gg 1$),

the inversion becomes ill-posed. Even in the worst-case collusion scenario (server and $N - 1$ clients), the remaining client's update is both sparse and low-rank due to LoRA, significantly limiting reconstructability.

Thus, FedShield-LLM integrates homomorphic encryption with secure key separation and sparsified LoRA updates to achieve a two-layered defense against gradient inversion and data inference attacks, ensuring robustness under standard cryptographic assumptions. □

D. Computational Complexity Analysis

We analyze the computational and communication complexity of each component in FedShield-LLM: LoRA fine-tuning, homomorphic encryption with secure aggregation, unstructured pruning, and communication overhead.

1) LoRA Fine-Tuning: Let P be the total model parameters and $P_{\text{LoRA}} \ll P$ the LoRA trainable parameters. Each client trains only $P_{\text{LoRA}} = O(r \cdot d)$ parameters (rank r, hidden size d). The per-iteration time complexity is $O(P + P_{\text{LoRA}})$, dominated by forward/backward over P. Optimizer state and memory usage are only for P_{LoRA}, yielding substantial speed-up and efficiency. Compared to full fine-tuning, LoRA reduces memory and compute significantly, enabling client-side feasibility for cross-silo FL.

2) FHE and Aggregation: Each client encrypts its LoRA update using CKKS. Encryption complexity per client is $O(n_c \cdot N_{\text{poly}} \log N_{\text{poly}})$, where n_c is the number of ciphertexts. With vector packing, updates (e.g., 30M parameters) are reduced to a few thousand ciphertexts. Encryption takes $\sim$15 seconds per client; decryption of aggregated updates is < 1 second. Homomorphic addition on the server is $O(N_{\text{poly}})$ per ciphertext. Compared to MPC-based schemes [6, 34], FHE trades higher compute for simpler one-round communication and supports post-aggregation operations.

3) Unstructured Pruning: Pruning selects a fraction $p_t \in [0.2, 0.5]$ of smallest magnitude parameters. Threshold selection via partial sort is $O(P_{\text{LoRA}})$; zeroing is linear. This introduces negligible runtime and can reduce encryption cost. Since pruning is done *after* gradient computation, it does not affect training dynamics.

4) Communication Overhead: LoRA reduces upload size from P to P_{LoRA} parameters (e.g., from 1200MB to 120MB). Pruning further reduces payload size. Encrypted updates (e.g., 30M values $\rightarrow$ $\sim$180MB) are acceptable in cross-silo FL with high-bandwidth links. While CKKS ciphertexts inflate size, fewer communication rounds and sparse updates reduce overall cost. In contrast, MPC-based secure aggregation incurs lower computational overhead but adds protocol complexity.

Overall, FedShield-LLM achieves computational efficiency via LoRA, minimal pruning cost, and scalable FHE-based aggregation. The end-to-end per-round time is practical for cross-silo deployments (e.g., hospitals or enterprises), and can be further optimized via quantized encryption [29] or hardware acceleration.

V. Experiments and Result Analysis

A. Experimental Setup

Our experiments were conducted on an Ubuntu server equipped with two NVIDIA RTX A6000 GPUs, an Intel Core i9 processor, and 128 GB of RAM. The study explored FL under Independent and Identically Distributed (IID) data scenarios, simulating 3 clients per communication round. Each client performed local training using the Adam optimizer with a learning rate of 5×10^{-5}, a batch size of 16, and gradient accumulation steps set to 1. The sequence length was fixed at 512, and each round consisted of a single local epoch per client. The pruning rate was progressively increased from 20%, achieving the target value of 50% by round 200. Pruning masks were applied to generate a sparsified model by setting less important weights to zero based on the pruning rate. This approach reduced computational overhead while maintaining accuracy. To ensure robust data privacy and security, we implemented HE from TenSEAL [4], a library built on Microsoft SEAL. The CKKS encryption scheme was configured with a polynomial modulus degree of 16384 and coefficient modulus sizes $[60, 40, 40, 40, 60]$. This setup enabled secure aggregation of model updates, sharing only the public key among clients and the server, thereby maintaining the confidentiality of individual client data. Model evaluation was conducted by decrypting the aggregated model weights at the server using the private key when necessary to assess performance. For fine-tuning efficiency, we leveraged PEFT techniques, specifically LoRA, with rank $r = 32$ and alpha $\alpha = 64$. This configuration optimized communication and computation, ensuring scalability in large-scale federated settings.

B. Dataset and Model

Our experiments were conducted using four diverse datasets obtained from Hugging Face: *vicgalle/alpaca-gpt4* (52,002 instruction-response pairs for general-purpose fine-tuning) [35], *FinGPT/fingpt-sentiment-train* (76,772 labeled examples for financial sentiment analysis) [26], *TIGER-Lab/MathInstruct* (262,039 tasks for mathematical reasoning) [46], and *medalpaca/medical_meadow_medical_flashcards* (33,955 medical flashcard entries) [17]. These datasets enabled evaluation across a wide range of domains, including general, financial, instructional, and medical tasks. To ensure fairness, data was distributed across clients using an IID strategy. This involved shuffling each dataset and partitioning it uniformly so that each client received an equal portion of the data, ensuring all clients handled representative shards.

For the model, we used *meta-llama/Llama-2-7b-hf* and *meta-llama/Llama-2-13b-hf*, two transformer-based pre-trained LLMs [37] known for their strong capabilities in natural language understanding and generation tasks. This setup enabled a comprehensive evaluation of our FL framework, demonstrating its effectiveness in securely and efficiently fine-tuning large language models under uniform data distribution across clients.

C. Result Analysis

In this section, we present a comprehensive evaluation of our proposed FedShield-LLM framework. We visualize and analyze training loss curves and text generation

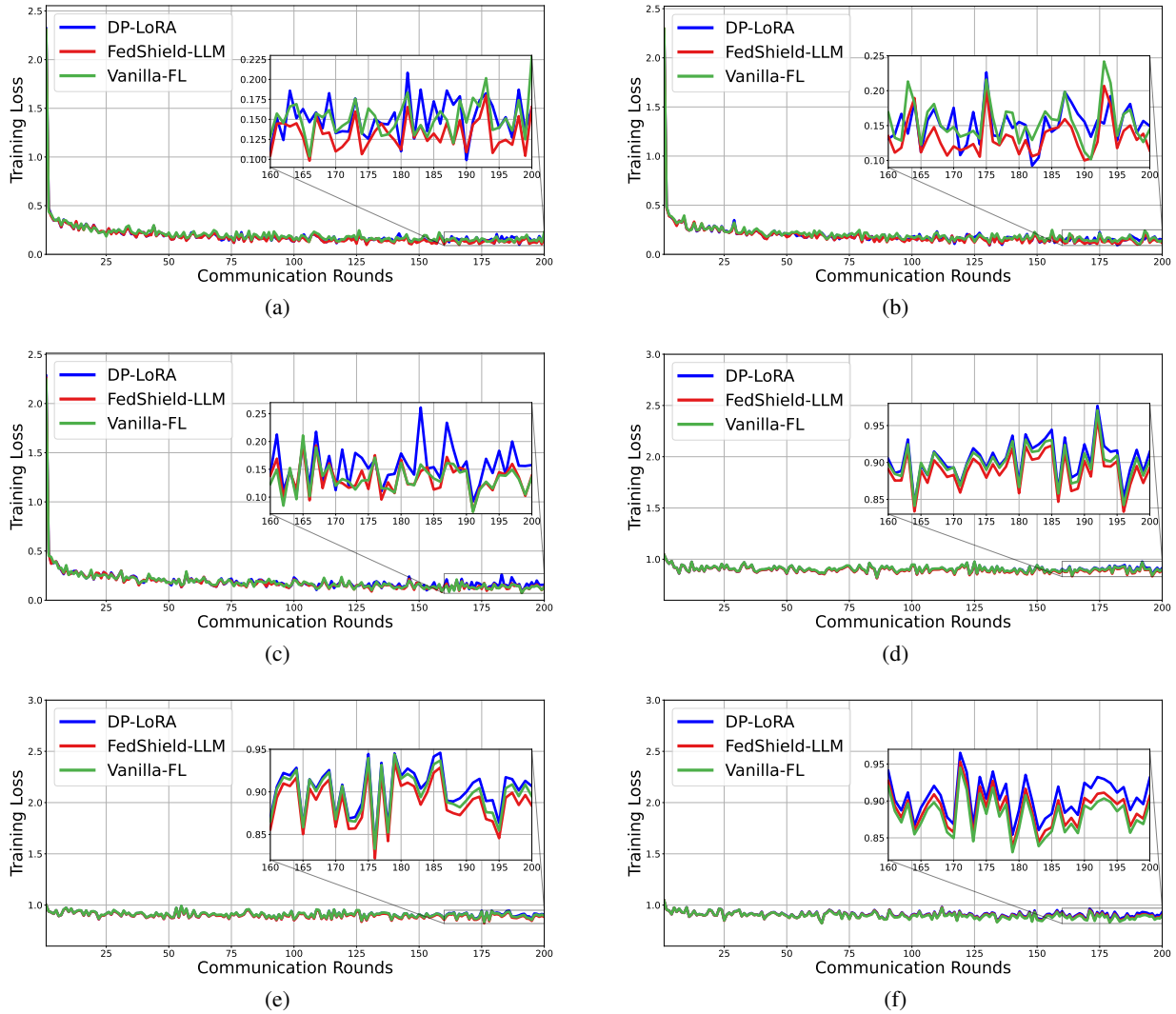

Fig. 3: Comparison of training loss for three clients in LLM fine tuning. Base Model: meta-llama/Llama-2-7b-hf. Datasets: fingpt-sentiment-train and vicgalle/alpaca-gpt4. Each row represents the training loss for a specific dataset, with subfigures (a)–(c) corresponding to fingpt-sentiment-train and (d)–(f) corresponding to vicgalle/alpaca-gpt4.

outputs produced by our fine-tuned models in comparison with GPT-4o, Vanilla-FL, and DP-LoRA. Furthermore, we report BERT scores to quantitatively assess the quality of generated text. Our main paper focuses on results obtained using two base models—*Llama-2-7b-hf* and *Llama-2-13b-hf* across two datasets: *fingpt-sentiment-train* and *alpaca-gpt4*. In addition, we provide a comparative analysis of model performance against DP-LoRA and Vanilla-FL based on loss metrics.

Figures 3(a–c) illustrate the training loss trends on the fingpt-sentiment-train dataset for three clients. In the federated fine-tuning of LLMs, the proposed FedShield-LLM consistently outperforms both Vanilla FL and DP-LoRA in terms of training loss reduction and convergence speed. Across Clients 1, 2, and 3, the proposed method maintains lower and more stable training loss throughout the training process. Notably, it achieves faster convergence, especially during the initial communication rounds. In contrast, DP-LoRA exhibits significantly higher loss across all rounds, indicating reduced performance compared to both FedShield-LLM and Vanilla FL. These results collectively demonstrate the superior performance, robustness, and efficiency of the proposed approach for federated fine-tuning of LLMs, highlighting its adaptability and effectiveness across diverse client datasets.

Figures 3(d–f) illustrate the results of fine-tuning LLMs using the Alpaca-GPT4 dataset, demonstrating consistent performance across all clients with minimal variability in training loss trends. The proposed

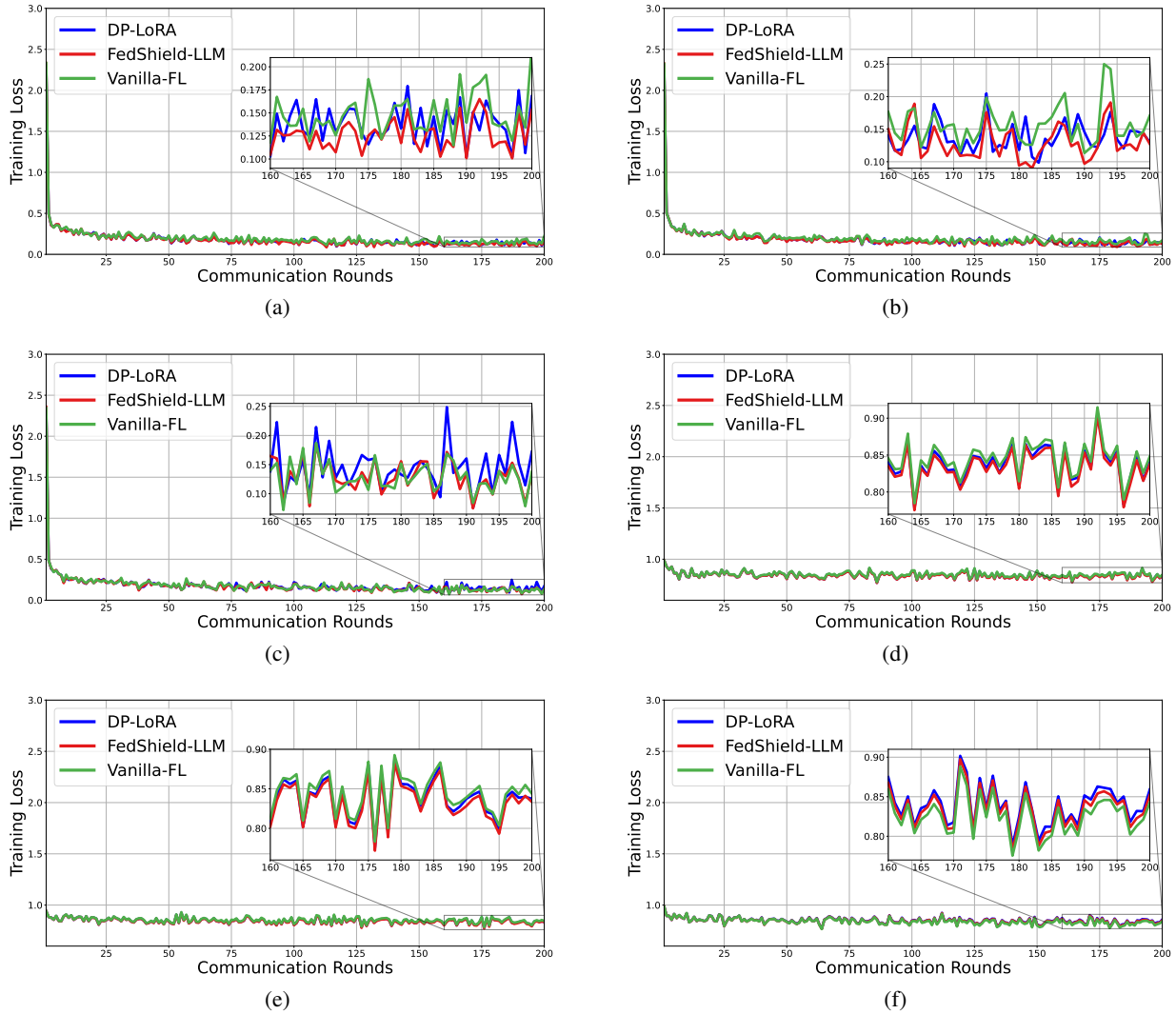

Fig. 4: Comparison of training loss for three clients in LLM fine tuning. Base Model: meta-llama/Llama-2-13b-hf. Datasets: fingpt-sentiment-train and vicgalle/alpaca-gpt4. Each row represents the training loss for a specific dataset, with subfigures (a)–(c) corresponding to fingpt-sentiment-train and (d)–(f) corresponding to vicgalle/alpaca-gpt4.

FedShield-LLM approach consistently achieved lower training loss compared to both Vanilla FL and DP-LoRA throughout all training rounds for Clients 1, 2, and 3. Similar to the previous dataset, DP-LoRA exhibited higher training loss across all rounds, indicating less effective optimization. These results highlight the superior optimization efficiency and training stability of the proposed method. Furthermore, the uniform trends observed across all clients underscore the robustness and scalability of FedShield-LLM, reinforcing its effectiveness in fine-tuning LLMs within a federated setting using the Alpaca-GPT4 dataset.

Figure 4 presents a comparative analysis of training loss across three clients during the federated fine-tuning of the LLaMA-2-13B model on two benchmark datasets: fingpt-sentiment-train (subfigures a–c) and vicgalle/alpaca-gpt4 (subfigures d–f). The proposed FedShield-LLM consistently achieves lower training loss compared to Vanilla FL and DP-LoRA across all communication rounds and clients. For both datasets, FedShield-LLM demonstrates improved convergence behavior and reduced variance, indicating enhanced training stability and robustness under heterogeneous client data distributions. Notably, DP-LoRA exhibits higher and more fluctuating loss values, reflecting its vulnerability to optimization inefficiencies in federated setups. The consistent performance of FedShield-LLM across diverse datasets and clients highlights its effectiveness in mitigating privacy-preserving fine-tuning challenges and its scalability to large model architectures in FL

environments.

The results of the reasoning task are summarized below, comparing the performance of our proposed model with Vanilla FL, DP-LoRA, and GPT-4o on generated text based on sample questions. For this evaluation, we utilized a fine-tuned model trained on the Alpaca-GPT4 dataset, built upon the *meta-llama/Llama-2-13b-hf* architecture. The findings indicate that our proposed model achieves performance nearly comparable to GPT-4o's reference outputs on the evaluated questions, while outperforming both Vanilla FL and DP-LoRA. Notably, despite incorporating FHE and pruning for enhanced security and communication efficiency, the proposed method maintains high effectiveness. These results highlight the robustness, privacy-preserving capability, and practical utility of our approach in secure federated fine-tuning of LLMs.

Category: Reasoning

Question: One morning after sunrise, a person was standing facing a pole. The shadow of the pole fell exactly to that person's right. Can you tell me the direction towards which the shadow was pointing — east, south, west, or north? Explain your reasoning steps. To which direction was that person facing? How do you solve this?

GPT-4o

In the morning, the sun is in the east, so shadows fall toward the west. Since the shadow of the pole is falling to the person's right, and the shadow is pointing west, the person must be facing south. Thus, the shadow is pointing west, and the person is facing south.

Vanilla-FL

The direction towards which the shadow was pointing is south. The reason for this is that the sun rises in the east and sets in the west, so if the shadow is falling to the right of the person, it means that the sun is in the east, and the person is facing south.
To determine the direction towards which the person was facing, we can use the following steps:
1. Determine the direction of the shadow: The shadow is falling to the right of the person, so the person is facing south.
2. Determine the direction of the sun: The sun is in the east, so the person is facing south.
3. Confirm the direction: The person is facing south, which is the opposite direction of the sun. This confirms that the person is facing south.

DP-LoRA

The direction towards which the shadow was pointing is south.
To solve this, we need to understand the direction of the sun's movement. The sun rises in the east and sets in the west. During the day, the sun moves from east to west, and its shadow moves in the opposite direction.
Since the shadow of the pole fell to the person's right, it means that the sun was in the east, and the person was facing east.
The person was facing east.

FedShield-LLM

The direction towards which the shadow was pointing is south. The reason for this is that the sun rises in the east and sets in the west, so if the shadow is falling to the right of the person, it means that the sun is in the east, and the person is facing south.
To determine the direction towards which the person was facing, we can use the following steps:
1. Identify the direction of the shadow: The shadow is falling to the right of the person, so the person is facing south.
2. Identify the direction of the sun: The sun is in the east, so the person is facing south.
3. Confirm the direction: The person is facing south, as the sun is in the east and the shadow is falling to the right.

Detailed results for other question categories, including reasoning, generic, knowledge, common-sense, Fermi, and coding tasks are available upon request.

These comparisons showcased the versatility and consistency of our proposed model across a wide range of domains and question types, demonstrating its effective performance in generating accurate and coherent responses.

TABLE II: Average BERTScore comparison across different methods while GPT-4o is baseline.

Model	Precision	Recall	F1 Score
Vanilla-FL	0.5683	0.5867	0.5756
DP-LoRA	0.6287	0.6031	0.6130
FedShield-LLM	**0.6738**	**0.7012**	**0.6865**

For this evaluation, we used seven questions in various categories. The response quality of Vanilla-FL, DP-LoRA, and FedShield-LLM was assessed using BERTScore, which computes semantic similarity between generated responses (candidates) and GPT-4o outputs (references). Specifically, we employed the pre-trained `bert-base-uncased` model for English text to obtain precision, recall, and F1 scores for each response. As shown in Table II, FedShield-LLM achieved the highest average F1 score of 0.6865, outperforming both DP-LoRA (0.6130) and Vanilla-FL (0.5756). In terms of precision and recall, FedShield-LLM also led with scores of 0.6738 and 0.7012, respectively, compared to DP-LoRA's 0.6287 precision and 0.6031 recall, and Vanilla-FL's 0.5683 precision and 0.5867 recall. While GPT-4o a multimodal and significantly larger model achieves a perfect BERTScore self-F1 of 1.0 on its own outputs, these results indicate that FedShield-LLM, a lightweight and privacy-preserving 7B and 13B parameter model, can produce responses that are nearly comparable to GPT-4o's within the evaluated scope. This underscores the effectiveness of our approach in semantically aligning with high-quality responses on specialized tasks, despite substantial differences in model scale and architecture.

Our proposed model defends against inference attacks as the server can only access encrypted LoRA parameters. In this case, the server has no knowledge of the actual model parameters. However, even if we allow the server to decrypt the model, an honest-but-curious server with access to the model parameters will still be unable to infer sensitive information through gradient inversion or reverse engineering attacks. This is because the attacker will only have access to the sparsified model. Therefore, our proposed model provides robust security against adversaries.

VI. Discussion

The results of this study underscore the effectiveness of FedShield-LLM in enhancing the performance and security of LLMs in FL. With the same hyperparameters and dataset, FedShield-LLM consistently outperformed Vanilla federated LLM and DP-LoRA, achieving lower training loss and generating text of higher quality, nearly comparable to GPT-4o. Our method demonstrated superior text generation across diverse question types and proved to be particularly suitable for cross-silo environments with resource constraints. By leveraging parameter-efficient fine-tuning through LoRA, FedShield-LLM significantly reduces computational and memory requirements, making it practical for environments with limited resources. Additionally, the integration of FHE with unstructured pruning optimized model parameters while ensuring robust data privacy, addressing critical challenges in secure distributed LLM training. To our knowledge, this is the first implementation of such an approach, positioning FedShield-LLM as a robust and efficient framework for sensitive and resource-constrained FL. applications.

VII. Conclusion

In this study, we proposed a secure and efficient mechanism, FedShield-LLM, for fine-tuning LLMs in FL by integrating FHE with unstructured pruning. As part of FHE, the CKKS encryption scheme ensures that model parameters remain encrypted throughout training and aggregation, protecting client data privacy. All model parameters are encrypted layer-wise and shared with the server to keep them secure during the communication and aggregation process. Unstructured pruning enhances security by deactivating less significant weights in LoRA parameters, reducing the attack surface and mitigating risks from inference attacks. Experimental results demonstrate that the proposed framework outperforms existing methods in text generation performance, while maintaining robust privacy guarantees and computational efficiency. This makes the approach suitable for real-world applications in sensitive domains. Future work will focus on addressing other categories of adversarial attacks during the training phase to further enhance the robustness of the framework in distributed environments.

Acknowledgement

This work is based upon the work supported by the National Center for Transportation Cybersecurity and Resiliency (TraCR) (a U.S. Department of Transportation National University Transportation Center) headquartered at Clemson University, Clemson, South Carolina, USA. Any opinions, findings, conclusions, and recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of TraCR, and the U.S. Government assumes no liability for the contents or use thereof.

REFERENCES

[1] Hadi Amini, Md Jueal Mia, Yasaman Saadati, Ahmed Imteaj, Seyedsina Nabavirazavi, Urmish Thakker, Md Zarif Hossain, Awal Ahmed Fime, and SS Iyengar. Distributed llms and multimodal large language models: A survey on advances, challenges, and future directions. *arXiv preprint arXiv:2503.16585*, 2025.

[2] Jiamu Bai, Daoyuan Chen, Bingchen Qian, Liuyi Yao, and Yaliang Li. Federated fine-tuning of large language models under heterogeneous tasks and client resources. *arXiv preprint arXiv:2402.11505*, 2024.

[3] Jeeyun Sophia Baik. Data privacy against innovation or against discrimination?: The case of the california consumer privacy act (ccpa). *Telematics and Informatics*, 52, 2020.

[4] Ayoub Benaissa, Bilal Retiat, Bogdan Cebere, and Alaa Eddine Belfedhal. Tenseal: A library for encrypted tensor operations using homomorphic encryption. *arXiv preprint arXiv:2104.03152*, 2021.

[5] Keith Bonawitz, Vladimir Ivanov, Ben Kreuter, Antonio Marcedone, H Brendan McMahan, Sarvar Patel, Daniel Ramage, Aaron Segal, and Karn Seth. Practical secure aggregation for federated learning on user-held data. *arXiv preprint arXiv:1611.04482*, 2016.

[6] Keith Bonawitz, Vladimir Ivanov, Ben Kreuter, Antonio Marcedone, H Brendan McMahan, Sarvar Patel, Daniel Ramage, Aaron Segal, and Karn Seth. Practical secure aggregation for privacy-preserving machine learning. In *proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, pages 1175–1191, 2017.

[7] Shawn Marie Boyne. Data protection in the united states. *The American Journal of Comparative Law*, 66(suppl_1):299–343, 2018.

[8] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.

[9] Nicholas Carlini, Florian Tramer, Eric Wallace, Matthew Jagielski, Ariel Herbert-Voss, Katherine Lee, Adam Roberts, Tom Brown, Dawn Song, Ulfar Erlingsson, et al. Extracting training data from large language models. In *30th USENIX security symposium (USENIX Security 21)*, pages 2633–2650, 2021.

[10] Jung Hee Cheon, Andrey Kim, Miran Kim, and Yongsoo Song. Homomorphic encryption for arithmetic of approximate numbers. In *Advances in Cryptology–ASIACRYPT 2017: 23rd International Conference on the Theory and Applications of Cryptology and Information Security, Hong Kong, China, December 3-7, 2017, Proceedings, Part I 23*, pages 409–437. Springer, 2017.

[11] Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, et al. Palm: Scaling language modeling with pathways. *Journal of Machine Learning Research*, 24(240):1–113, 2023.

[12] Badhan Chandra Das, M Hadi Amini, and Yanzhao Wu. Security and privacy challenges of large language models: A survey. *ACM Computing Surveys*, 57(6):1–39, 2025.

[13] Cynthia Dwork, Aaron Roth, et al. The algorithmic foundations of differential privacy. *Foundations and Trends® in Theoretical Computer Science*, 9(3–4):211–407, 2014.

[14] Tao Fan, Yan Kang, Guoqiang Ma, Lixin Fan, Kai Chen, and Qiang Yang. Fedcollm: A parameter-efficient federated co-tuning framework for large and small language models. *arXiv preprint arXiv:2411.11707*, 2024.

[15] Jonas Geiping, Hartmut Bauermeister, Hannah Dröge, and Michael Moeller. Inverting gradients-how easy is it to break privacy in federated learning? *Advances in neural information processing systems*, 33:16937–16947, 2020.

[16] Yuhao Gu, Yuebin Bai, and Shubin Xu. Cs-mia: Membership inference attack based on prediction confidence series in federated learning. *Journal of Information Security and Applications*, 67:103201, 2022.

[17] Tianyu Han, Lisa C Adams, Jens-Michalis Papaioannou, Paul Grundmann, Tom Oberhauser, Alexander Löser, Daniel Truhn, and Keno K Bressem. Medalpaca–an open-source collection of medical conversational ai models and training data. *arXiv preprint arXiv:2304.08247*, 2023.

[18] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015.

[19] Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, Weizhu Chen, et al. Lora: Low-rank adaptation of large language models. *ICLR*, 1(2):3, 2022.

[20] Ahmed Imteaj, Urmish Thakker, Shiqiang Wang, Jian Li, and M Hadi Amini. A survey on federated learning for resource-constrained iot devices. *IEEE Internet of Things Journal*, 9(1):1–24, 2021.

[21] Weirui Kuang, Bingchen Qian, Zitao Li, Daoyuan Chen, Dawei Gao, Xuchen Pan, Yuexiang Xie, Yaliang Li, Bolin Ding, and Jingren Zhou.

Federatedscope-llm: A comprehensive package for fine-tuning large language models in federated learning. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 5260–5271, 2024.

[22] Lambda Labs. Openai's gpt-3: Language model: A technical overview. Blog post: https://lambda.ai/blog, 2020.

[23] Tian Li, Anit Kumar Sahu, Manzil Zaheer, Maziar Sanjabi, Ameet Talwalkar, and Virginia Smith. Federated optimization in heterogeneous networks. *Proceedings of Machine learning and systems*, 2:429–450, 2020.

[24] Yang Li, Wenhan Yu, and Jun Zhao. Privtuner with homomorphic encryption and lora: A p3eft scheme for privacy-preserving parameter-efficient fine-tuning of ai foundation models. *arXiv preprint arXiv:2410.00433*, 2024.

[25] Tao Liu, Zhi Wang, Hui He, Wei Shi, Liangliang Lin, Ran An, and Chenhao Li. Efficient and secure federated learning for financial applications. *Applied Sciences*, 13(10):5877, 2023.

[26] Xiao-Yang Liu, Guoxuan Wang, Hongyang Yang, and Daochen Zha. Fingpt: Democratizing internet-scale data for financial large language models. *arXiv preprint arXiv:2307.10485*, 2023.

[27] Xiao-Yang Liu, Rongyi Zhu, Daochen Zha, Jiechao Gao, Shan Zhong, Matt White, and Meikang Qiu. Differentially private low-rank adaptation of large language model using federated learning. *ACM Transactions on Management Information Systems*, 2023.

[28] Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Aguera y Arcas. Communication-efficient learning of deep networks from decentralized data. In *Artificial intelligence and statistics*, pages 1273–1282. PMLR, 2017.

[29] Md Jueal Mia and M Hadi Amini. Quancrypt-fl: Quantized homomorphic encryption with pruning for secure federated learning. *arXiv preprint arXiv:2411.05260*, 2024.

[30] Payman Mohassel and Yupeng Zhang. Secureml: A system for scalable privacy-preserving machine learning. In *2017 IEEE symposium on security and privacy (SP)*, pages 19–38. IEEE, 2017.

[31] OpenAI. Gpt-4 technical report. https://openai.com/research/gpt-4, 2023.

[32] Prashanthi Ramachandran, Shivam Agarwal, Arup Mondal, Aastha Shah, and Debayan Gupta. S++: A fast and deployable secure-computation framework for privacy-preserving neural network training. *arXiv preprint arXiv:2101.12078*, 2021.

[33] Sashank Reddi, Zachary Charles, Manzil Zaheer, Zachary Garrett, Keith Rush, Jakub Konečnỳ, Sanjiv Kumar, and H Brendan McMahan. Adaptive federated optimization. *arXiv preprint arXiv:2003.00295*, 2020.

[34] Pushpa Singh, Murari Kumar Singh, Rajnesh Singh, and Narendra Singh. Federated learning: Challenges, methods, and future directions. In *Federated Learning for IoT Applications*, pages 199–214. Springer, 2022.

[35] Rohan Taori, Ishaan Gulrajani, Tianyi Zhang, Yann Dubois, Xuechen Li, Carlos Guestrin, Percy Liang, and Tatsunori B Hashimoto. Stanford alpaca: An instruction-following llama model, 2023.

[36] Meng Tong, Kejiang Chen, Jie Zhang, Yuang Qi, Weiming Zhang, Nenghai Yu, Tianwei Zhang, and Zhikun Zhang. Inferdpt: Privacy-preserving inference for black-box large language models. *IEEE Transactions on Dependable and Secure Computing*, 2025.

[37] Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*, 2023.

[38] Ziyao Wang, Zheyu Shen, Yexiao He, Guoheng Sun, Hongyi Wang, Lingjuan Lyu, and Ang Li. Flora: Federated fine-tuning large language models with heterogeneous low-rank adaptations. *arXiv preprint arXiv:2409.05976*, 2024.

[39] Dongxian Wu and Yisen Wang. Adversarial neuron pruning purifies backdoored deep models. *Advances in Neural Information Processing Systems*, 34:16913–16925, 2021.

[40] Feijie Wu, Zitao Li, Yaliang Li, Bolin Ding, and Jing Gao. Fedbiot: Llm local fine-tuning in federated learning without full model. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 3345–3355, 2024.

[41] Yebo Wu, Chunlin Tian, Jingguang Li, He Sun, Kahou Tam, Li Li, and Chengzhong Xu. A survey on federated fine-tuning of large language models. *arXiv preprint arXiv:2503.12016*, 2025.

[42] Jie Xu, Karthikeyan Saravanan, Rogier van Dalen, Haaris Mehmood, David Tuckey, and Mete Ozay. Dp-dylora: Fine-tuning transformer-based models on-device under differentially private federated learning using dynamic low-rank adaptation. *arXiv preprint arXiv:2405.06368*, 2024.

[43] Rui Ye, Wenhao Wang, Jingyi Chai, Dihan Li, Zexi Li, Yinda Xu, Yaxin Du, Yanfeng Wang, and Siheng Chen. Openfedllm: Training large language models on decentralized private data via federated learning. In *Proceedings of the 30th ACM SIGKDD*

Conference on Knowledge Discovery and Data Mining, pages 6137–6147, 2024.

[44] Da Yu, Saurabh Naik, Arturs Backurs, Sivakanth Gopi, Huseyin A Inan, Gautam Kamath, Janardhan Kulkarni, Yin Tat Lee, Andre Manoel, Lukas Wutschitz, et al. Differentially private fine-tuning of language models. *arXiv preprint arXiv:2110.06500*, 2021.

[45] Hao Yu, Sen Yang, and Shenghuo Zhu. Parallel restarted sgd with faster convergence and less communication: Demystifying why model averaging works for deep learning. In *Proceedings of the AAAI conference on artificial intelligence*, volume 33, pages 5693–5700, 2019.

[46] Xiang Yue, Xingwei Qu, Ge Zhang, Yao Fu, Wenhao Huang, Huan Sun, Yu Su, and Wenhu Chen. Mammoth: Building math generalist models through hybrid instruction tuning. *arXiv preprint arXiv:2309.05653*, 2023.

[47] Sheng Yun, Zakirul Alam Bhuiyan, Md Taufiq Al Hasib Sadi, and Shen Su. Privacy-preserving federated learning through clustered sampling on fine-tuning distributed non-iid large language models. In *2023 IEEE Intl Conf on Parallel & Distributed Processing with Applications, Big Data & Cloud Computing, Sustainable Computing & Communications, Social Computing & Networking (ISPA/BDCloud/SocialCom/SustainCom)*, pages 531–538. IEEE, 2023.

[48] Lihong Zhang and Yue Li. Federated learning with layer skipping: Efficient training of large language models for healthcare nlp. *arXiv preprint arXiv:2504.10536*, 2025.

[49] Weishan Zhang, Baoyu Zhang, Xiaofeng Jia, Hongwei Qi, Rui Qin, Juanjuan Li, Yonglin Tian, Xiaolong Liang, and Fei-Yue Wang. Federated intelligence for intelligent vehicles. *IEEE Transactions on Intelligent Vehicles*, 2024.

[50] Xinyu Zhang, Huiyu Xu, Zhongjie Ba, Zhibo Wang, Yuan Hong, Jian Liu, Zhan Qin, and Kui Ren. Privacyasst: Safeguarding user privacy in tool-using large language model agents. *IEEE Transactions on Dependable and Secure Computing*, 2024.

[51] Ligeng Zhu, Zhijian Liu, and Song Han. Deep leakage from gradients. *Advances in neural information processing systems*, 32, 2019.