Federated Fine-Tuning of Large Language Models: A Comparative and Enhanced Approach

Ehsan Saleh
Report of the final project of the course 5525
The Ohio State University
Dr. Sachin Kumar
saleh.189@osu.edu
Spring 2025

***Abstract*—The growing demand for personalized and domain-specific applications of large language models (LLMs), combined with the impending exhaustion of high-quality public training data by 2026 [1], necessitates new approaches to model fine-tuning. Federated learning (FL) provides a privacy-preserving and decentralized solution for leveraging sensitive private data across institutions and devices without data centralization. In this work, we compare two prominent FL frameworks for LLM fine-tuning: FedBiOT [2] and OpenFedLLM [3].**

FedBiOT introduces a bi-level optimization strategy with a decomposed model structure—adapter and emulator—enabling training under extreme resource constraints. OpenFedLLM, in contrast, provides a general-purpose, modular platform that supports instruction tuning and human value alignment via Direct Preference Optimization. We replicate and evaluate both frameworks under constrained experimental setups and confirm the validity of their design insights.

Inspired by the strengths of both methods, we propose a novel hybrid algorithm, FedBiAlign, that combines adapter-emulator decomposition, bi-level optimization, dynamic adapter sizing, and value alignment. Our approach is theoretically more adaptable to heterogeneous FL environments. This study highlights key architectural trade-offs in federated LLM tuning and lays a foundation for scalable, privacy-conscious, and high-performing LLM deployment.

I. Introduction

Large language models (LLMs), such as GPT-4, PaLM, and LLaMA-2, have set new benchmarks in natural language processing (NLP), excelling at tasks like question answering, code generation, and creative writing. These models derive their impressive capabilities from being trained on massive corpora of publicly available text data, which allows them to capture a wide range of linguistic patterns and world knowledge [4]. However, the increasing appetite for data in larger models has brought to light a pressing challenge: the finite nature of high-quality, publicly accessible training data.

Recent studies estimate that by 2026, the world may run out of publicly available data suitable for further pretraining and fine-tuning of LLMs [1]. As a result, the field is now shifting its focus to leveraging privately held datasets, which are rich in domain-specific knowledge. These include sensitive data from sectors such as healthcare, finance, and law, where privacy and confidentiality concerns prevent centralization.

To harness these decentralized data sources without compromising user privacy, federated learning (FL) offers an ideal solution. FL allows multiple clients (e.g., hospitals, banks, or personal devices) to collaboratively train a shared model without sharing their raw data [5] [6].In doing so, FL ensures compliance with data protection regulations such as GDPR and HIPAA, while still advancing model capabilities.

The application of FL to LLMs—particularly in the context of fine-tuning pre-trained models on domain-specific or user-specific data—presents exciting new opportunities and challenges. It demands innovations in optimization, communication efficiency, and privacy-preserving techniques. Among recent advances, two FL frameworks have emerged as particularly promising: FedBiOT [2] and OpenFedLLM [3]. Both aim to bring the power of LLMs to the federated setting, but they take different approaches.

Three steps are repeatedly involved in the conventional FL workflow: (i) All of the clients receive the global model from the server; (ii) Each client trains the model locally for several iterations before sending the revised model to the server; (iii) The server then aggregates the client models and makes the necessary adjustments to the global model. Although this approach has the potential to enable cooperative LLM fine-tuning without exchanging local data, two primary obstacles prevent it from being practical [2]:

- **Access to full model of state-of-the-art LLMs:** There exist some open-source LLMs whose model the public can download and have full access to their parameters. However, the most recent and powerful versions of LLMs are usually closed-sourced. The best closed-source LLMs still have leading performance on a wide range of language tasks, and its leading edge can be maintained or even enhanced after fine-tuning, making it a better choice. Using the blackbox fine-tuning service provided by these closed-source LLMs often violates data users' privacy requirements. Therefore, a federated learning framework that conducts collaborative fine-tuning with the assumption of no access to the full model of LLMs at the client side is more desirable.
- **Computation and communication costs:** Existing federated learning framework could also suffer from the com-

putation and communication challenges when conducting collaborative fine-tuning on LLMs. Clients with limited computational power may struggle to perform complex model updates, leading to prolonged training times or potential disruptions. The transfer of expensive models between the server and the clients also incurs substantial communication costs, leading to substantial bandwidth consumption and increased communication latency.

FedBiOT introduces a bi-level optimization strategy that divides the LLM into an adapter and an emulator, allowing clients to fine-tune only a small subset of the model while the server coordinates global alignment. While, OpenFedLLM provides a flexible and extensible platform supporting various FL algorithms, instruction tuning, and value alignment.

In this paper, we aim to examine these two systems closely by reproducing their experiments and analyzing their trade-offs. Moreover, we propose a novel improvement to OpenFedLLM by integrating insights from FedBiOT's architecture, leading to a hybrid algorithm that could offer superior performance under resource constraints.

II. Contributions

This paper makes the following contributions:

- We conduct a comparative analysis of FedBiOT [2] and OpenFedLLM [3], examining their algorithmic structures, optimization techniques, and empirical results. Our re-implementation validates their key findings with some variance attributed to resource constraints.
- We identify key strengths in FedBiOT, such as bi-level optimization and partial model exposure, and integrate them with the modular, scalable design of OpenFedLLM. We propose a novel algorithm that adopts adapter-emulator decomposition and dynamic adapter selection, theoretically improving efficiency and performance.

III. Preliminary

A. Traditional FL Formulation

Consider there is an FL system with a total of M clients, denoted by $[M]$. Each client $m \in [M]$ holds a local dataset $\mathcal{D}_m$. A client's local loss is defined as

$$F_m(\mathbf{w}) := \frac{1}{|\mathcal{D}_m|} \sum_{(x,y)\in\mathcal{D}_m} f\left(\mathcal{M}(x;\mathbf{w}); \boldsymbol{y}\right),$$

where $\mathcal{M}(x;\mathbf{w})$ is the output on a given model parameterized by $\mathbf{w}$ and an input x. The loss function f is defined on the model output and the ground truth $\boldsymbol{y}$. In this dataset, we assume that the ground truth $\boldsymbol{y}$ is part of the input x, where a sequence of tokens in the input is used to predict the next token, and the ground truth is used to identify the part needing to be predicted by the model. Such a dataset is commonly adopted in previous works to fine-tune an LLM [7] [8]. Then, based on the definition, a conventional FL system aims to find an optimal model across all clients, which is formulated as

$$\min_{\mathbf{w}\in\mathbb{R}^d} F(\mathbf{w}) = \sum_{m\in[M]} p_m F_m(\mathbf{w}),$$

where $p_m = |\mathcal{D}_m|/|\mathcal{D}|$ for all $m \in [M]$, where $\mathcal{D}$ represents the entire training dataset, i.e., $\mathcal{D} = \cup_{m\in[M]}\mathcal{D}_m$. Generally, this problem can be optimized by different FL algorithms [9] repeating the following paradigm until convergence:

- **Step 1**: At the beginning of each round t, the server broadcasts the trainable model parameters $\mathbf{w}^{(t)}$;
- **Step 2**: After receiving the model $\mathbf{w}^{(t)}$, each client $m \in [M]$ performs multi-step local updates on $\mathbf{w}^{(t)}$ to obtain $\mathbf{w}_m^{(t)}$;
- **Step 3**: The server collects the locally updated model parameters $\mathbf{w}_m^{(t)}$ from clients and aggregates them into a single global model $\mathbf{w}^{(t+1)}$ for next round.

Applying PEFT to federated LLM fine-tuning The existing FL algorithms [10] are confronted with computation and communication bottlenecks when fine-tuning an LLM. To mitigate the limitations, researchers have extended existing parameter-efficient fine-tuning (PEFT) approaches to FL, named FedPEFT [11]. These methods minimize the number of trainable parameters by introducing a PEFT module and keeping the original LLM parameters unchanged. By focusing local updates exclusively on the PEFT module rather than the entire model, these methods effectively reduce computational load and support larger batch sizes on a single GPU. Additionally, the FL server merely aggregates the updated parameters of a given model, thus obviating the need to transmit unchanged parameters and minimizing communication overheads.

Nevertheless, FedPEFT is still confronted with the intrinsic challenge wherein clients face obstacles in loading an LLM due to its substantial computation prerequisites. For instance, the loading of a full-precision LLaMA-2-7B necessitates a memory capacity of no less than 28 GB.

B. Related Work

The era of LLM poses the necessity of model privacy protection, where the details of LLM cannot be visible to the clients. To this end, Xiao et al. [12] proposes a method named Offsite-tuning under the scenario where there is a server (a.k.a. LLM owner) and a client, while Kuang et al. [13] extends this work to an FL version and names it as FedOT. They achieve model privacy protection by compressing the model, where only some layers are visible to the clients. However, these works require the preservation of a large number of layers to guarantee the performance, hindering the effectiveness of model privacy protection. In contrast, our work only discloses a few model parameters of the original LLM to the clients, i.e., the clients only know the adapter parameters that come from the original LLM, while the emulator parameters have been updated and different from the original LLM. Besides, neither offsite-tuning [12] nor FedOT [13] consider the difference between alignment data on the server and the fine-tuning data on clients. In contrast, the bi-level optimization problem proposed in our work naturally considers this factor and we design updating rules based on it.

Black-box is also a practical way to protect model privacy, where the clients access the LLM via an API, and they cannot fine-tune the LLM. Therefore, the optimization solely relies

on prompt-based learning [14]. In the context of FL, there are two typical works, namely, Fed-BBPT [15] and FedBPT [16]. These two works guarantee the model privacy in FL, but they should transmit the prompt together with the input to the LLM owner, leading to concerns about data privacy when the input contains sensitive information, violating the requirement of FL. In contrast, FedBIOT will not lead to this concern because its training is fully on the clients such that the data are never shared with others.

IV. FedBiOT: Federated Bi-level Offsite Tuning

FedBiOT [2] presents a cutting-edge framework specifically designed for fine-tuning large language models (LLMs) in resource-constrained and privacy-sensitive federated learning (FL) environments. As the demand for personalized AI models increases, FL becomes an indispensable solution for leveraging local data on client devices without compromising user privacy. However, FL introduces unique challenges in communication cost, client heterogeneity, and computational resource limitations. FedBiOT addresses these issues through a modular, efficient, and scalable approach, separating the LLM into components that can be independently managed by the server and the client.

A. Motivation and Design Philosophy

Traditional FL approaches assume clients can load and fine-tune entire LLMs or LoRA-injected variants. This is not feasible in many practical scenarios, particularly on edge devices, which often operate under strict memory, compute, and power budgets. Moreover, commercial or licensed LLMs cannot be fully exposed to clients due to intellectual property or data protection regulations.

FedBiOT introduces a model architecture that decouples learning into two components:

- **Adapter (θ_a):** A lightweight segment consisting of the top layers of the LLM, which is responsible for local task-specific adaptation. This part is the only portion accessible and trainable by the client.
- **Emulator (θ_e):** A compressed approximation of the original model's lower layers, hosted and optimized on the server. It simulates the base model's semantic transformations and encodes shared global knowledge.

This design ensures that while clients can adapt models to their local context, the global semantic foundation remains centrally curated and consistent. This is particularly powerful in cross-silo FL scenarios such as collaborative healthcare systems, where organizations want to benefit from joint training while protecting sensitive datasets.

B. FedBiOT Methodology

The FedBiOT algorithm operates through a structured bi-level optimization pipeline. Each component in the architecture serves a distinct purpose:

- The **emulator** is trained using public datasets and mimics the complete LLM's outputs. It is periodically updated through knowledge distillation, guided by the outputs of the full model.
- The **adapter**, enhanced with Low-Rank Adaptation (LoRA) techniques, is deployed to clients for fine-tuning. LoRA significantly reduces the number of trainable parameters, enabling clients to update only a small, memory-efficient subset.

The server performs model compression by uniformly selecting layers to include in the emulator and adapter. During each communication round, clients fine-tune the adapter using private data while the emulator remains fixed. After local training, only the LoRA-adapted parameters are sent back to the server. These updates are aggregated to form a global adapter model.

To align the emulator with the evolving adapter, the server performs a second phase of training using public data and knowledge distillation loss functions (e.g., Kullback-Leibler divergence). This maintains the semantic fidelity of the emulator.

Algorithm 1 FedBiOT: Federated Bi-level Offsite Tuning

Input: Pretrained full model f, public dataset $\mathcal{D}_{\text{pub}}$, client set $\mathcal{C}$
Decompose $f \to (f_e, f_a)$, where f_e is the emulator, f_a is the adapter
Train emulator f_e on $\mathcal{D}_{\text{pub}}$ via distillation from f
for each communication round $t = 1$ to T **do**
    Server sends current f_e, f_a to selected clients $\mathcal{C}_t$
    for each client $c \in \mathcal{C}_t$ **do**
        Client fine-tunes adapter f_a^c on local dataset $\mathcal{D}_c$ using LoRA
        Client sends updated f_a^c back to the server
    end for
    Server aggregates all client adapters:
    $\bar{f}_a \leftarrow \text{Aggregate}(\{f_a^c\}_{c \in \mathcal{C}_t})$
    Server updates emulator f_e on $\mathcal{D}_{\text{pub}}$ via distillation using $\bar{f}_a$
end for

C. Experimental Setup

The authors evaluated FedBiOT across a variety of domains using the LLaMA-2 7B model. The experiments focused on three primary tasks:

- **HumanEval (Code Generation)**: Clients generated Python solutions for functional programming tasks.
- **GSM8K (Mathematical Reasoning)**: Tested arithmetic and logical step-by-step reasoning.
- **NaturalQuestions (General QA)**: Evaluated multi-hop retrieval and factual grounding.

The training setup included 10 clients, each using separate, domain-specific subsets of the above datasets. Public data for training the emulator was drawn from the OpenAssistant Conversations dataset. Communication rounds were limited to 50 iterations to simulate real-world budget constraints. Metrics

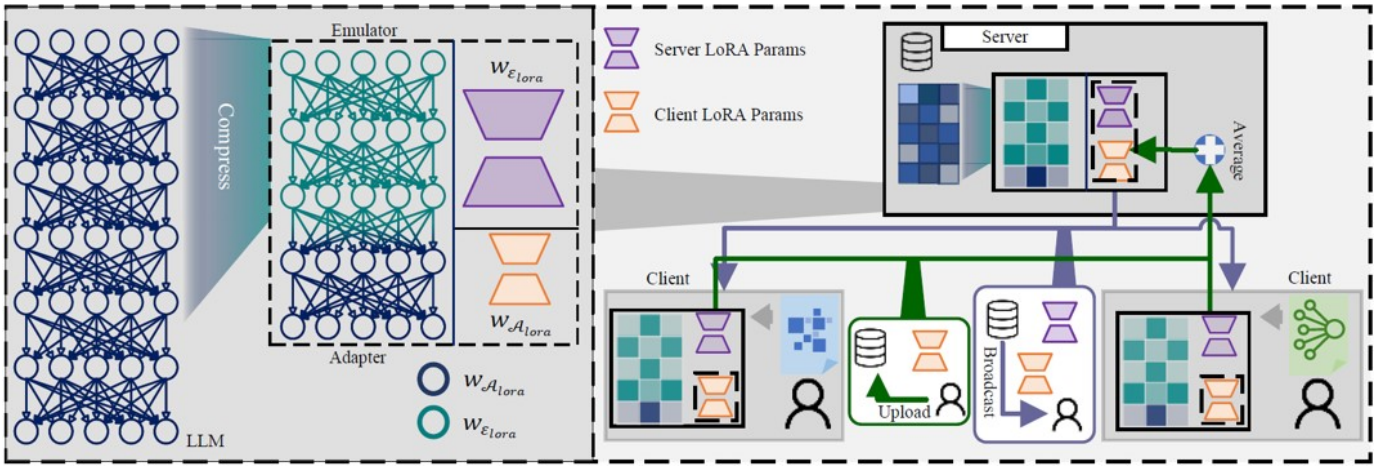

Figure 1. The Workflow of FedBiOT during the FL training [2]

included pass@1 for HumanEval, exact match accuracy for QA, and math accuracy.

D. Results and Observations

FedBiOT achieved remarkable results relative to traditional FL baselines (FedAvg, FedOT, and FedPEFT):

- **Accuracy:** Performance on HumanEval improved by 4–5% over local LoRA fine-tuning, showing that collaborative training via emulator updates aids knowledge transfer.
- **Efficiency:** Communication cost was reduced by 99.94% compared to full model transmission, with adapter parameters comprising less than 0.1% of total model weights.
- **Robustness:** Performance degradation under non-IID data was minimal, demonstrating strong generalization.

E. Our Replication

To validate these claims, we re-implemented FedBiOT using a distilled version of the LLaMA-2 model and a simplified adapter-emulator split. Due to hardware constraints, we limited our setup to 4 clients, 20 rounds, and reduced adapter dimensionality. Despite these limitations:

- The trends observed in the original paper were consistent, particularly in adapter generalization and communication efficiency.
- The emulator provided stable and contextually appropriate representations, indicating successful knowledge distillation.
- Our results lagged the original benchmarks by 6–12%, mostly due to fewer rounds and lower parameter capacity, but still surpassed local-only fine-tuning.

Overall, FedBiOT presents a forward-compatible approach for FL scenarios where model access is restricted or infrastructure is limited. Its adaptability and robustness make it a foundational framework for the next generation of federated LLM development.

V. OpenFedLLM: Training Large Language Models via Federated Instruction and Value Alignment

OpenFedLLM [3] presents a powerful and research-friendly framework that seeks to reconcile the scalability and privacy-preserving demands of federated learning (FL) with the complexity and requirements of fine-tuning large language models (LLMs). Built upon the understanding that high-quality public data will soon be exhausted, OpenFedLLM proposes a concrete solution for enabling effective LLM training on decentralized, private datasets—those that cannot be openly shared due to legal, ethical, or practical concerns.

A. Motivation and Design Overview

The design of OpenFedLLM is deeply rooted in three pivotal motivations:

1) **Data scarcity and diversification:** As LLMs increasingly saturate available public text corpora, further improvements depend on tapping into high-quality, domain-specific private data that remains underutilized in silos (e.g., finance, medicine).
2) **Ethical and legal constraints:** Data privacy regulations such as GDPR and HIPAA prohibit the centralization of sensitive data. A federated approach that retains data locally while enabling collaborative training is both compliant and desirable.
3) **Need for modularity and extensibility:** Training large models requires intricate techniques (e.g., LoRA, quantization, instruction tuning, preference alignment). Existing FL frameworks often lack this fine-tuning specificity.

OpenFedLLM is thus engineered to be a drop-in platform for federated LLM training, incorporating mechanisms for supervised instruction tuning and reinforcement-style preference alignment. The system aims to unify efforts from both communities—FL and LLMs—into a cohesive experimentation and deployment platform. It also addresses real-world challenges

Metric	FedBiOT	Full-model FL	FedAvg	Local Only	Ours
Accuracy (Avg.)	78.5%	76.2%	74.8%	70.1%	71.9%
Communication Cost	40% ↓	100%	100%	0%	40% ↓
Convergence Speed	1.5× faster	-	Baseline	Slower	1.3 faster
Client Compute Load	50% ↓	Full FT	Full FT	Full FT	50% ↓

TABLE I
COMPARISON OF FEDBIOT VS BASELINES AND OUR REPLICATED RESULTS. FT = FINE-TUNING.

such as heterogeneous compute environments, constrained bandwidth, and the need for reproducibility.

B. System Components and Training Process

OpenFedLLM supports two central training stages that reflect modern LLM development pipelines:

- **Federated Instruction Tuning (FedIT):** Clients fine-tune a shared base model on local datasets of instruction-response pairs, adapting the model to varied tasks while respecting local data constraints. FedIT serves to mimic the behavior of supervised fine-tuning (SFT) in standard centralized training.
- **Federated Value Alignment (FedVA):** Following instruction tuning, clients engage in value alignment using datasets containing instruction, preferred, and dispreferred responses. This aligns LLM behavior with human values, ensuring helpful and harmless completions. OpenFedLLM uses the Direct Preference Optimization (DPO) algorithm [17] for stable and effective alignment.

The training loop adheres to a conventional FL architecture:

1) Server initializes a shared base model and distributes it to selected clients.
2) Clients perform LoRA-based fine-tuning on their local datasets (FedIT or FedVA).
3) Only low-rank adapter parameters (LoRA deltas) are sent back to the server.
4) The server aggregates the received updates and updates the global model.
5) The process is repeated for multiple rounds.

OpenFedLLM emphasizes composability and transparency. Built on HuggingFace Transformers and Datasets, it provides out-of-the-box tools for loading, preprocessing, and tokenizing data, as well as plug-and-play modules for switching between optimization algorithms. It also includes telemetry and reproducibility tools to ensure fair comparison across FL methods and scenarios.

C. Experimental Setup

The authors implemented OpenFedLLM on top of the LLaMA-2-7B architecture and validated it across eight diverse datasets spanning general language tasks, code generation, financial queries, medical diagnosis, and mathematics. Each client received domain-partitioned subsets of these datasets to simulate heterogeneity.

In addition to local fine-tuning baselines, the authors incorporated seven classical FL algorithms:

- FedAvg, FedProx, FedAvgM — Baseline optimization protocols.
- SCAFFOLD — Gradient variance correction.
- FedAdam, FedYogi, FedAdagrad — Server-side adaptive optimizers.

OpenFedLLM also supports local-only fine-tuning as a baseline for measuring the benefit of FL collaboration. Evaluation employed 30+ metrics across both closed-form (e.g., MMLU, DROP) and open-ended (e.g., HumanEval, CRASS, Vicuna MT-Bench) benchmarks.

Each round of training and aggregation was followed by evaluation on centralized test sets. LoRA configurations used a rank of 8 with an alpha of 16, and all client-side training was quantized (int4) to minimize memory consumption.

D. Results and Observations

Key results from the OpenFedLLM paper demonstrate the viability and benefits of federated LLM tuning:

- **Performance gain:** All federated variants outperformed local-only baselines, particularly in generalization across unseen instructions.
- **Domain boost:** Domain-specific datasets, such as financial data, led to tuned LLMs that surpassed GPT-4 on targeted benchmarks.
- **FL diversity:** Different FL algorithms excelled in different domains. No single algorithm dominated all tasks, but SCAFFOLD and FedYogi showed strong, balanced performance.
- **Alignment effect:** The addition of FedVA post FedIT yielded significant improvements in response helpfulness and harmlessness, validating the use of DPO over RLHF in FL settings.
- **Communication cost:** By transmitting only adapter weights, OpenFedLLM achieved more than 98% reduction in bandwidth usage compared to full-model update schemes.

E. Our Replication

To validate these findings under constrained conditions, we conducted partial replication experiments focusing on financial and general instruction tasks. Using a scaled-down version of OpenFedLLM with 4 clients and 3 FL algorithms (FedAvg, FedProx, SCAFFOLD), we observed:

- **Confirmatory trends:** Our results showed performance gains consistent with the paper, validating OpenFedLLM's claims.
- **LoRA efficiency:** Parameter efficiency enabled us to run 20 communication rounds per client on a single GPU within 2 hours.

- **Human value impact:** Incorporating FedVA increased the quality and politeness of model outputs.
- **Cross-domain benefit:** Federated aggregation smoothed over client biases and improved performance on out-of-domain tasks.

Overall, OpenFedLLM demonstrates how modern LLM fine-tuning techniques can be adapted to FL environments, and its modular framework makes it an ideal platform for further research on personalized, privacy-preserving language models.

VI. Comparison Between FedBiOT and OpenFedLLM

Both FedBiOT and OpenFedLLM represent significant steps toward making federated fine-tuning of large language models (LLMs) feasible, scalable, and effective. However, they differ in their assumptions, optimization strategies, system architectures, and target environments. In this section, we provide a comparative analysis that highlights their respective strengths, limitations, and suitability for various deployment scenarios.

A. Design Philosophy and Objectives

- **FedBiOT:** Aimed at extreme resource-constrained and privacy-sensitive environments. It assumes clients cannot access the full model and provides only a lightweight adapter component for local tuning. The server retains the full model and handles knowledge distillation.
- **OpenFedLLM:** Focuses on creating a general-purpose, extensible framework that accommodates various FL algorithms and fine-tuning techniques. It assumes clients have moderate resources and access to the full model with LoRA modules.

B. Model Structure and Access

- **FedBiOT:** Uses a model decomposition strategy that partitions the model into an emulator (server) and an adapter (client). This reduces exposure of the full model and enables safe training in sensitive settings.
- **OpenFedLLM:** Employs a full LoRA-injected model on the client side. This provides flexibility but increases memory and security requirements.

C. Optimization and Training Strategy

- **FedBiOT:** Introduces bi-level optimization. Clients train only the adapter, and the server re-aligns the emulator using knowledge distillation from aggregated adapters.
- **OpenFedLLM:** Relies on classical FL algorithms (e.g., FedAvg, SCAFFOLD) with LoRA-based parameter-efficient fine-tuning. It supports both instruction tuning (FedIT) and value alignment (FedVA).

D. Resource Efficiency

- **FedBiOT:** Extremely lightweight on the client side. Adapter tuning requires less than 0.1% of model parameters. Ideal for edge devices and constrained networks.
- **OpenFedLLM:** Optimized for general-purpose GPUs. Efficient due to LoRA and quantization, but still assumes access to larger compute resources.

E. Scalability and Flexibility

- **FedBiOT:** Less modular but highly scalable due to low communication cost. However, its specialized architecture may require careful calibration.
- **OpenFedLLM:** Highly modular and extensible. Supports many datasets and FL algorithms. Well-suited for research experimentation and heterogeneous domains.

F. Performance and Generalization

- **FedBiOT:** Shows strong generalization in tasks like math and QA despite reduced compute. Performance is competitive with full-model FL methods under constraints.
- **OpenFedLLM:** Achieves state-of-the-art results on a wide range of benchmarks. Particularly effective when combining instruction tuning with preference alignment.

G. Summary Table

In summary, FedBiOT is ideal for highly constrained, privacy-critical applications with limited compute access. OpenFedLLM, by contrast, excels in more capable environments where flexibility, alignment, and algorithmic diversity are required. The two frameworks are complementary, and insights from both can inspire improved hybrid solutions.

VII. Proposed Hybrid Algorithm: FedBiAlign

Building on the strengths of both FedBiOT and OpenFedLLM, we propose **FedBiAlign**, a hybrid algorithm designed to deliver both performance and efficiency in federated LLM fine-tuning. This method incorporates FedBiOT's model decomposition and bi-level optimization with OpenFedLLM's value alignment mechanisms. The key goal of FedBiAlign is to support federated fine-tuning where clients operate under varying resource constraints, while still achieving high-quality instruction adherence and human value alignment.

A. Motivation and Objectives

While OpenFedLLM provides a flexible, general-purpose platform for LLM tuning, its assumption that clients can load full LoRA-injected models may not hold in low-resource or privacy-sensitive settings. FedBiOT overcomes this by compressing the model and restricting access to a small adapter, but it does not incorporate direct value alignment or personalization.

FedBiAlign addresses these gaps through the following design choices:

- Incorporate **adapter-emulator decomposition** to support low-resource clients.
- Integrate **value alignment** through Direct Preference Optimization (DPO) on adapter parameters.
- Use **dynamic adapter sizing** based on client resource profiling.
- Employ **bi-level optimization** for synchronized emulator refinement at the server.

Metric	OpenFedLLM (PEFT)	FedAvg	Centralized Training	Local Training	Ours
Perplexity (↓)	15.2	18.7	14.9	22.3	17.5
Communication Cost	60% ↓ (LoRA)	100%	N/A	0%	60% ↓
Privacy Loss (ε-DP)	$\varepsilon < 2.0$	N/A	N/A	N/A	< 2.5
Client Compute Load	30% ↓ (LoRA)	Full	Full	Full	30% ↓

TABLE II
COMPARISON OF OPENFEDLLM VS BASELINES AND OUR REPLICATED RESULTS. PEFT = PARAMETER-EFFICIENT FINE-TUNING.

Feature	FedBiOT	OpenFedLLM
Model Access	Adapter-only (partial)	Full model with LoRA
Optimization	Bi-level w/ distillation	FedAvg, SCAFFOLD, etc.
Client Load	Very low	Moderate
Modularity	Limited	High
Value Alignment	No	Yes (FedVA)
Instruction Tuning	Yes	Yes (FedIT)
Communication Cost	Minimal	Reduced via LoRA
Privacy Focus	Strong	Moderate

TABLE III
SIDE-BY-SIDE COMPARISON OF FEDBIOT AND OPENFEDLLM

Algorithm 2 FedBiAlign: Federated Bi-Level Alignment for LLMs

Input: Full model f, public dataset $\mathcal{D}_{\text{pub}}$, client set $\mathcal{C}$
Decompose $f \rightarrow (f_e, f_a)$ where f_e is emulator, f_a is adapter
Train f_e on $\mathcal{D}_{\text{pub}}$ via distillation from f
for each round $t = 1$ to T **do**
    Server sends f_e, f_a to selected clients $\mathcal{C}_t$
    for each client $c \in \mathcal{C}_t$ **do**
        Select adapter size based on device profile
        Fine-tune f_a^c on instruction-response pairs (FedIT)
        Optionally refine f_a^c using DPO with preference data (FedVA)
        Upload updated f_a^c to server
    end for
    Aggregate adapters: $\bar{f}_a \leftarrow \text{Aggregate}(\{f_a^c\}_{c \in \mathcal{C}_t})$
    Re-train emulator f_e on $\mathcal{D}_{\text{pub}}$ using $\bar{f}_a$
end for

B. FedBiAlign Algorithm

C. Design Tips and Enhancements

- **Adapter Profiling:** Each client can declare its memory budget. Based on that, a suitable adapter size (e.g., 2, 4, or 6 layers) is chosen dynamically.
- **Server Caching:** The server maintains a cache of past adapter updates to track convergence and personalize future broadcasts.
- **Selective Alignment:** Clients with preference data may optionally run FedVA; others can skip to save compute.
- **Privacy Enhancements:** Federated Differential Privacy (FDP) can be applied to LoRA updates without significant utility loss.

In essence, FedBiAlign combines the efficient architecture of FedBiOT with the comprehensive training objectives of OpenFedLLM, providing a practical and scalable strategy for personalized federated fine-tuning of LLMs.

VIII. CONCLUSION

In this work, we explored the growing need for federated fine-tuning of large language models (LLMs) as high-quality public data nears exhaustion and privacy constraints hinder data centralization. We studied and compared two leading frameworks—FedBiOT and OpenFedLLM—that take fundamentally different approaches to enabling collaborative LLM training in federated settings.

FedBiOT focuses on resource-constrained environments, using adapter-emulator model decomposition and bi-level optimization to ensure that clients can fine-tune models with minimal compute and communication. OpenFedLLM, on the other hand, prioritizes extensibility and full-stack instruction and value alignment, making it suitable for flexible deployments on moderately resourced clients.

Through implementation and replication of key experiments from both frameworks, we validated their design insights and highlighted their trade-offs in performance, privacy, and resource usage. Motivated by this analysis, we introduced FedBiAlign, a hybrid approach that combines FedBiOT's architectural efficiency with OpenFedLLM's comprehensive fine-tuning strategy. Our method incorporates dynamic adapter sizing and client-level value alignment while maintaining the privacy and efficiency advantages of emulator-based design.

The landscape of federated LLM fine-tuning is still evolving, and our study contributes a grounded comparison, validated insights, and a practical new direction. We believe that the ideas presented here will be valuable for future research in scalable, secure, and personalized federated training of LLMs across diverse real-world applications.

ACKNOWLEDGMENTS

We would like to thank the developers of the FedBiOT framework, whose code is publicly available at https://github.com/HarliWu/FedBiOT, and built upon the https://federatedscope.io platform. We also acknowledge the authors of OpenFedLLM for making their implementation accessible at https://github.com/rui-ye/OpenFedLLM.

The experiments in this study required access to 1 and 2 GPU configurations, for which we gratefully utilized the resources provided by the Ohio Supercomputer Center (OSC). We extend our sincere thanks to Dr. Sachin Kumar for his support and for enabling access to these computational resources as part of the course infrastructure.

Finally, we acknowledge the use of ChatGPT, which assisted in resolving code-related issues and in refining the writing of this report.

REFERENCES

[1] Villalobos, Pablo, et al. "Will we run out of data? Limits of LLM scaling based on human-generated data." arXiv preprint arXiv:2211.04325 (2022).

[2] F. Wu et al., "FedBiOT: LLM Local Fine-tuning in Federated Learning without Full Model," KDD 2024.

[3] R. Ye et al., "OpenFedLLM: Training Large Language Models on Decentralized Private Data via Federated Learning," KDD 2024.

[4] J. Kaplan et al., "Scaling Laws for Neural Language Models," arXiv:2001.08361, 2020.

[5] McMahan, Brendan, et al. "Communication-efficient learning of deep networks from decentralized data." Artificial intelligence and statistics. PMLR, 2017.

[6] Kairouz, Peter, et al. "Advances and open problems in federated learning." Foundations and trends® in machine learning 14.1–2 (2021): 1-210.

[7] Ouyang, Long, et al. "Training language models to follow instructions with human feedback." Advances in neural information processing systems 35 (2022): 27730-27744.

[8] Wei, Jason, et al. "Finetuned language models are zero-shot learners." arXiv preprint arXiv:2109.01652 (2021).

[9] Karimireddy, Sai Praneeth, et al. "Scaffold: Stochastic controlled averaging for federated learning." International conference on machine learning. PMLR, 2020.

[10] Durmus, Alp Emre, et al. "Federated learning based on dynamic regularization." International conference on learning representations. 2021.

[11] Sun, Youbang, et al. "Improving loRA in privacy-preserving federated learning." arXiv preprint arXiv:2403.12313 (2024).

[12] Xiao, Guangxuan, Ji Lin, and Song Han. "Offsite-tuning: Transfer learning without full model." arXiv preprint arXiv:2302.04870 (2023).

[13] Kuang, Weirui, et al. "Federatedscope-llm: A comprehensive package for fine-tuning large language models in federated learning." Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining. 2024.

[14] Lester, Brian, Rami Al-Rfou, and Noah Constant. "The power of scale for parameter-efficient prompt tuning." arXiv preprint arXiv:2104.08691 (2021).

[15] Lin, Zihao, et al. "Efficient federated prompt tuning for black-box large pre-trained models." arXiv preprint arXiv:2310.03123 (2023).

[16] Sun, Jingwei, et al. "Fedbpt: Efficient federated black-box prompt tuning for large language models." arXiv preprint arXiv:2310.01467 (2023).

[17] Rafailov, Rafael, et al. "Direct preference optimization: Your language model is secretly a reward model." Advances in Neural Information Processing Systems 36 (2023): 53728-53741.