# Personalized Federated Fine-Tuning for LLMs via Data-Driven Heterogeneous Model Architectures

**Anonymous ACL submission**

## Abstract

Large-scale instruction data is essential for aligning pretrained Large Language Models (LLMs) with human instructions, but may contain sensitive information that hinders its public sharing. Federated Learning (FL) enables collaborative fine-tuning of LLMs without data sharing. However, existing approaches to federated LLM fine-tuning usually adopt a uniform model architecture, making it hard to fit the highly heterogeneous data with varying amounts and formats. To address this, we propose FedAMoLE, a lightweight personalized FL framework that enables data-driven heterogeneous model architectures. This framework features an adaptive mixture of LoRA experts (MoLE) module for aggregating heterogeneous models and a reverse selection-based expert assignment strategy that optimizes model architectures based on data distributions. Experiments across five scenarios show that FedAMoLE improves accuracy by an average of 5.14% compared to existing approaches while obtaining good scalability.

## 1 Introduction

The impressive responsiveness of pretrained Large Language Models (LLMs) (Zhao et al., 2023) to human instructions relies on fine-tuning with large amounts of instruction-based text data, as suggested by the scaling law (Kaplan et al., 2020). However, instructional data often contain sensitive information that cannot be directly shared, causing data silos. Federated Learning (FL) (McMahan et al., 2017) provides a compelling solution by enabling multiple organizations to collaboratively train models without sharing raw data, ensuring privacy while leveraging client-side downstream data effectively (Xie et al., 2022; Wang et al., 2024).

Given the large scale of LLMs, recent works have proposed federated fine-tuning approaches that leverage parameter-efficient fine-tuning (PEFT) (Zhang et al., 2023a; Bai et al., 2024; Sun et al., 2024; Wu et al., 2024a; Qin et al., 2024b; Che et al., 2023; Zhang et al., 2023b) and zero-order optimization (Xu et al., 2024; Qin et al., 2024a). However, they only provide one globally consistent model. In cross-silo scenarios, different business organizations often possess data within various domains, causing data distribution heterogeneity (a.k.a. not independent and identically distributed, non-IID). A single model typically struggles to adapt to all domains, resulting in suboptimal accuracy on client-side tasks (Tan et al., 2023).

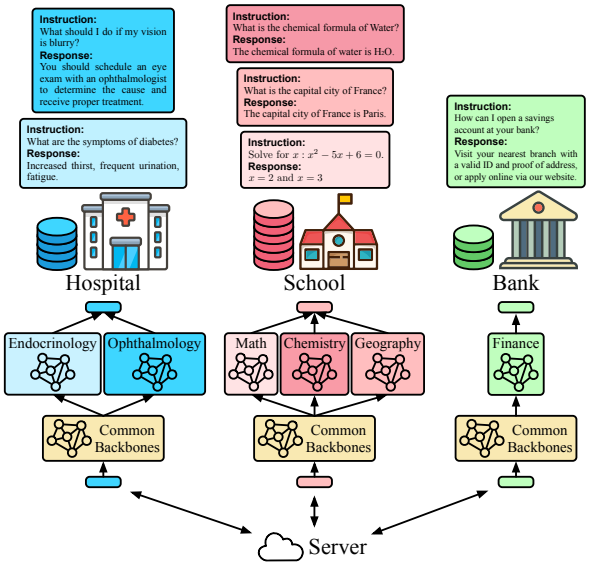
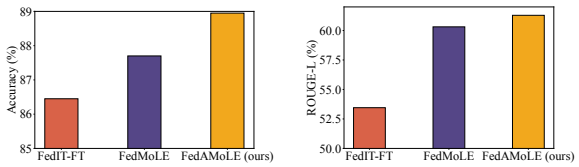



Figure 1: Federated LLM fine-tuning with three organizations, each with data that differ in domain and volume. To better adapt to local data distributions, personalized models with varied architectures are usually preferred.

Prior FL works indicate that enabling multiple personalized models helps to enhance the accuracy on non-IID data (Li et al., 2020b; Qin et al., 2023). However, due to reliance on additional data or high resource costs, existing personalized FL (PFL) methods are hard to be directly applied for LLMs (Appendix A.2). Thus, some studies have explored PFL for LLMs (Jiang et al., 2024; Yang




(a) SNLI (b) Natural Instructions

Figure 2: Effectiveness of heterogeneous and data-driven model architectures. FedIT-FT and FedMoLE denote FL approaches with homogeneous and heterogeneous models, respectively. FedAMole tailors model architectures driven by data distributions based on FedMoLE. Experimental settings aligned with Section 5.5.

et al., 2024; QI et al., 2024). Despite their promising results, they still face two key limitations:

- (L1) **Limited Support for Heterogeneous Models:** A uniform model architecture is insufficient for client data with diverse domains (Tan et al., 2023; Fan et al., 2024a; Ye et al., 2024). As shown in Figure 1, in a modular heterogeneous model, each submodule (expert) may contain knowledge of a specific domain. Considering the different domains contained in client-side data, enabling heterogeneous model architectures contributes to better fitness of personalized models to client-side data. As shown in Figure 2, the approach enabling heterogeneous model architectures, FedMoLE, outperforms the one with homogeneous models, FedIT-FT. Unfortunately, current methods for LLMs either struggle to effectively support heterogeneous model architectures (QI et al., 2024; Luo et al., 2024a; Wu et al., 2024b; Almansoori et al., 2024), or brings significant communication overhead (Mei et al., 2024).

- (L2) **Data-Unaware Model Architectures:** Existing FL methods with heterogeneous LLMs primarily rely on manually predefined model architectures based on resource constraints, lacking joint awareness of local data distribution and cross-client data distribution. As exemplified by FedAMoLE (our approach) in Figure 2, by tailoring personalized model architectures based on data distributions to bridge experts and client-side data domains, the accuracy of LLMs fine-tuned by FL could be further improved.

To address these limitations, this work aims at tailoring personalized model architectures with awareness of data distributions. To enable heterogeneous model architectures (L1), we adopt **m**ixture of **L**oRA **e**xperts (MoLE) architecture as the basis, proposing an **a**daptive **MoLE** (AMoLE) module with a novel shape-invariant router, making FL for LLMs possible to support the aggregation of architecturally heterogeneous models. Further, to optimize model architectures based on client-side data distributions (L2), we introduce a **r**everse **s**election-based **e**xpert **a**ssignment (RSEA) strategy, which enables the candidate experts to select appropriate clients. It optimizes model architectures driven by the data characteristics of all clients in an FL system, alongside FL progress.

This work makes the following contributions:

- We introduce FedAMoLE, a novel PFL framework for LLMs. It enables architecture-level model heterogeneity based on the proposed adaptive MoLE module, without introducing significant communication overhead.

- We propose RSEA, a data-driven model architecture optimization strategy based on a reverse selection pattern of candidate experts, which jointly considers the data characteristics of all clients in the FL system to achieve the optimal design of personalized model architectures.

- We conduct extensive experiments across five non-IID FL scenarios, showing that FedAMoLE improves accuracy by an average of 5.14% over the strongest baseline. Our codes are at `https://anonymous.4open.science/r/FedAMoLE`.

## 2 Related Work

### 2.1 Personalized Federated LLM Fine-Tuning

Current FL methods for LLM fine-tuning usually employ PEFT techniques like LoRA (Zhang et al., 2023a; Bai et al., 2024; Sun et al., 2024; Wu et al., 2024a; Qin et al., 2024b; Babakniya et al., 2023) and prompt tuning (Che et al., 2023) to reduce memory and communication costs stemming from the large scale of LLMs. However, these methods provide only a single global model, which is often difficult to adapt to diverse client-side data.

Recent research explores personalization, including dual adapter integration (Jiang et al., 2024; Yang et al., 2024; QI et al., 2024), LoRA adapter masking (Zhang et al., 2024), and mixtures of feed-forward networks (FFNs) (Mei et al., 2024). Some of them are confined to parameter heterogeneity within the same architecture (QI et al., 2024). However, personalizing only the model parameters without tailoring the model structure may struggle to

adapt to non-IID data distributions, leading to suboptimal performance (Qin et al., 2023). Others rely on manually-defined architectures (Jiang et al., 2024; Yang et al., 2024; Zhang et al., 2024). On one hand, predefined structures may not be optimal and struggle to accurately capture task-specific features; on the other hand, they may fail to precisely match the actual data distribution of each client, thereby limiting personalization capability. Besides, some methods incur substantial communication costs due to the transmission of fine-tuned pre-trained parameters (Mei et al., 2024).

### 2.2 Federated LLM Fine-Tuning with Mixture of Experts

The Mixture of Experts (MoE) architecture, leveraging specialized sub-models (experts) with a gating mechanism (Lepikhin et al., 2021; Fedus et al., 2021), presents a promising avenue for personalized FL (PFL) by enabling model personalization through expert collaboration. Recent studies involving MoE for federated LLM fine-tuning often employ lightweight adapters as experts for resource efficiency (Almansoori et al., 2024). However, these methods primarily address parameter heterogeneity within a unified model architecture. Existing works enabling architecture-level heterogeneity either relies on dense FFN experts Mei et al. (2024), causing high communication costs and incompatibility to non-MoE LLMs like LLaMA (Touvron et al., 2023), or require manually-defined architectures (Fan et al., 2024b), lacking optimization based on data distributions. Different from existing works, our method enables data-driven architecture optimization, which maximizes personalization potential while maintaining low communication cost.

## 3 Problem Formulation

We provide a formal definition of PFL for LLM fine-tuning. In FL, $C$ clients collaboratively solve:

$$\min_{\theta} \mathcal{L} = \frac{1}{C} \sum_{i=1}^{C} \mathbb{E}_{\mathbf{X} \sim \mathcal{D}_i} \mathcal{L}_i(\mathbf{X}|\theta), \qquad (1)$$

aiming at optimizing a global model $\theta$ by minimizing the expected loss $\mathcal{L}_i$ over local data distribution $\mathcal{D}_i$ for each client $i$. To address data heterogeneity across clients, PFL trains a personalized model $\theta_i$ for each client $i$, modifying the objective to:

$$\min_{\theta_1, \theta_2 \ldots, \theta_C} \mathcal{L} = \frac{1}{C} \sum_{i=1}^{C} \mathbb{E}_{\mathbf{X} \sim \mathcal{D}_i} \mathcal{L}_i(\mathbf{X}|\theta_i). \qquad (2)$$

For LLM fine-tuning, $\mathbf{X} = [\mathbf{X}^{\text{instr}}, \mathbf{X}^{\text{resp}}]$ denotes a text sequence, where $\mathbf{X}^{\text{instr}}$ is the instruction and $\mathbf{X}^{\text{resp}}$ is the response. $\mathcal{L}_i$ is the negative log-likelihood (NLL) loss defined as:

$$\mathcal{L}_i(\mathbf{X} \mid \theta_i) = -\frac{1}{T} \sum_{t=1}^{T} \log \left[ p(\mathbf{X}_t^{\text{resp}} \mid \mathbf{X}^{\text{instr}}, \mathbf{X}_{<t}^{\text{resp}}, \theta_i) \right], \quad (3)$$

where $T$ is the response sequence length. In typical PFL, $\theta_i$ uses a uniform structure with varying parameters, while FedAMoLE enables architectural heterogeneity for greater personalization.

## 4 Approach

### 4.1 Framework Overview

FedAMoLE is designed for personalized federated LLM fine-tuning with the following key objectives: (1) enabling architecture-level model heterogeneity; (2) tailoring personalized model architectures aware of both local and global data distributions; and (3) minimizing additional resource overhead.

Figure 3 overviews the components and processes of FedAMoLE. In it, each client employs a transformer-based local model with $L$ decoder layers. Each layer comprises a self-attention layer (with parameter matrices $\mathbf{Q}$, $\mathbf{K}$, $\mathbf{V}$, and $\mathbf{O}$) and an FFN layer. Personalization is achieved by injecting AMoLE modules into activated linear layers (e.g., the $\mathbf{Q}$ and $\mathbf{V}$ matrices within the self-attention layer, shown in Figure 3). Each AMoLE module contains a shared expert, a domain expert pool, and a token projection, with each client running its own instance. The shared expert and token projection are broadcast to all clients, while domain experts select the client subset with the best data relevance. This data-driven expert assignment approach adaptively combines the shared expert's general knowledge with the domain expert's domain-specific knowledge, enabling each client to find the optimal personalized model architecture tailored to its local data.

In FedAMoLE, each FL round consists of six key steps. First, the server distributes global parameters to all clients, including shared experts, domain experts, and token projections. Next, clients inject these modules into their local LLM backbones to form personalized models (Figure 3, ①). Then, clients freeze their backbones and fine-tune the injected parameters using local training data (Figure 3, ②). Afterward, they compute embeddings for both domain experts and local data using a random subset of their training set, termed as *embedding set* (Figure 3, ③), and return the tuned pa-

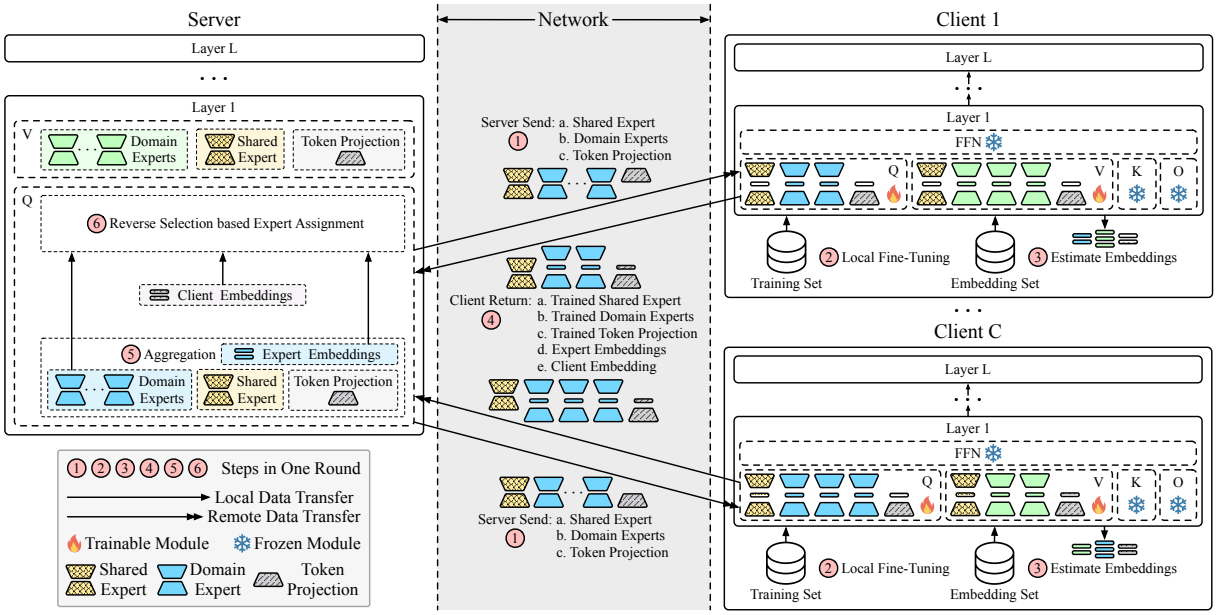



Figure 3: Overview of FedAMoLE. This figure illustrates the workflow of a single round, where Q, K, V, and O are the weight matrices of the self-attention layer. Q and V are each injected with an AMoLE module (detailed in Figure 4) for fine-tuning, while K and O are frozen. Step ⑥ denotes the RSEA strategy, detailed in Figure 5. For clarity, components of the same type (e.g., all shared experts) are shown with an identical color and texture.

rameters along with these embeddings to the server (Figure 3, ④). Finally, the server aggregates the received data (Figure 3, ⑤), calculates relevance scores between client data and expert domains, and enables each domain expert to perform reverse selection by choosing the clients with the highest relevance, thereby determining expert assignments for the next round (Figure 3, ⑥).

### 4.2 Adaptive MoLE Module

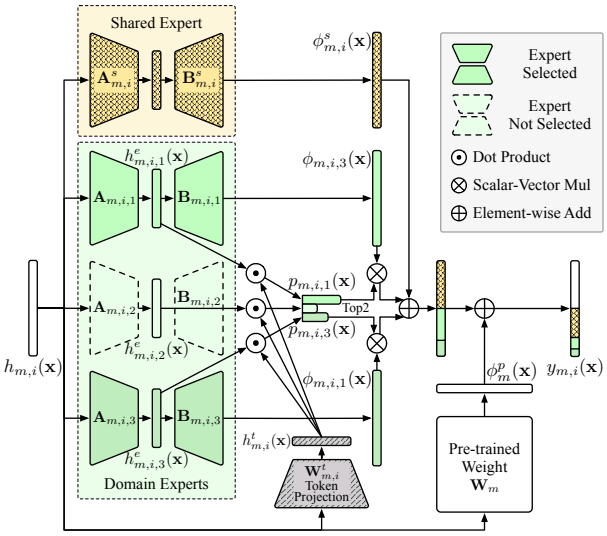



Figure 4: Routing of AMoLE module $m$ at client $i$.

Vanilla MoLE, though efficient, faces federated aggregation challenges due to expert-number-dependent router shape (Appendix B). To address this, we propose the **a**daptive **MoLE** (AMoLE) module, which introduces a novel router with consistent shapes across clients, regardless of local expert configurations, thereby enabling lightweight model heterogeneity with FedAvg compatibility.

The routing process of the AMoLE module is illustrated in Figure 4. In module $m$ of client $i$, the hidden state of the input token $\mathbf{x}$ is denoted as $h_{m,i}(\mathbf{x}) \in \mathbb{R}^d$. This hidden state is projected into a low-dimensional subspace by the $\mathbf{A}$ matrix of each domain expert $j$, generating the corresponding expert embedding $h^e_{m,i,j}(\mathbf{x}) \in \mathbb{R}^r$. Simultaneously, $h_{m,i}(\mathbf{x})$ is also projected by the token projection matrix $\mathbf{W}^t_{m,i} \in \mathbb{R}^{r\times d}$ to obtain the token embedding $h^t_{m,i}(\mathbf{x}) \in \mathbb{R}^r$ in this subspace. Next, the scaled dot product between the token embedding and each expert embedding is computed, followed by softmax normalization to obtain the weighting coefficient $p_{m,i,j}(\mathbf{x})$ for each domain expert $j$. Finally, the top-$k^e$ domain experts with the highest coefficients (where $k^e = 2$ in Figure 4) are selected to extract features $\phi_{m,i,j}(\mathbf{x})$. These features are weighted by the corresponding coefficients, averaged, and combined with the feature from the shared expert $\phi^s_{m,i}(\mathbf{x})$ and the pre-trained feature $\phi^p_m(\mathbf{x})$ to generate the final output $y_{m,i}(\mathbf{x})$.

This process can be formalized as

$$h^e_{m,i,j}(\mathbf{x}) = \mathbf{A}_{m,i,j} h_{m,i}(\mathbf{x}) \tag{4}$$

$$h^t_{m,i}(\mathbf{x}) = \mathbf{W}^t_{m,i} h_{m,i}(\mathbf{x}) \tag{5}$$

$$p_{m,i,j}(\mathbf{x}) = \text{softmax}\left[\frac{h^t_{m,i}(\mathbf{x})^T h^e_{m,i,j}(\mathbf{x})}{\sqrt{d}}\right] \tag{6}$$

$$\mathcal{T}_{m,i}(\mathbf{x}) = \text{top-}k^e(p_{m,i,j}(\mathbf{x})) \tag{7}$$

$$\phi^p_m(\mathbf{x}) = \mathbf{W}_m h_{m,i}(\mathbf{x}) \tag{8}$$

$$\phi^s_{m,i}(\mathbf{x}) = \mathbf{B}^s_{m,i} \mathbf{A}^s_{m,i} h_{m,i}(\mathbf{x}) \tag{9}$$

$$\phi_{m,i,j}(\mathbf{x}) = \mathbf{B}_{m,i,j} \mathbf{A}_{m,i,j} h_{m,i}(\mathbf{x}) \tag{10}$$

$$y_{m,i}(\mathbf{x}) = \phi^p_m(\mathbf{x}) + \phi^s_{m,i}(\mathbf{x}) + \sum_{j \in \mathcal{T}_{m,i}(\mathbf{x})} p_{m,i,j}(\mathbf{x}) \cdot \phi_{m,i,j}(\mathbf{x}), \tag{11}$$

where $\mathbf{W}_m$ is the pre-trained parameter of module $m$, $\mathbf{A}_{m,i,j}$ and $\mathbf{B}_{m,i,j}$ are the parameters of domain expert $j$, $\mathbf{A}^s_{m,i}$ and $\mathbf{B}^s_{m,i}$ are the parameters of the shared expert, and $k^e$ is the number of domain experts each token is routed to.

### 4.3 Local Fine-Tuning and Aggregation

The local fine-tuning on client $i$ optimizes the parameters of the AMoLE module set $\mathcal{M}$:

$$\theta_i = \bigcup_{m \in \mathcal{M}} \theta_{m,i}, \tag{12}$$

while keeping the pre-trained parameters of the local model frozen. Here,

$$\theta_{m,i} = \{\mathbf{A}^s_{m,i}, \mathbf{B}^s_{m,i}, \mathbf{W}^r_{m,i}\} \cup \{\mathbf{A}_{m,i,j}, \mathbf{B}_{m,i,j}\}_{j \in \mathcal{E}_{m,i}} \tag{13}$$

is the learnable parameters of module $m$ for client $i$, where $\mathcal{E}_{m,i}$ is the set of domain experts assigned to module $m$ of client $i$ in the current round.

During each fine-tuning round, client $i$ iteratively updates its parameters $\theta_i$. In each iteration, a mini-batch $\mathcal{B}$ is sampled from the client's training dataset $\mathcal{D}^{train}_i$. Next, the gradient of $\theta_i$ on $\mathcal{B}$ is computed using a loss function $\mathcal{L}_i$ that combines the NLL loss and the load balance loss (Appendix C.1). Finally, with a step size $\eta$, the parameters are updated according to the following rule (simplified):

$$\theta^{t+1}_i = \theta^t_i - \eta \frac{\partial \mathcal{L}_i(\mathcal{B}|\theta^t_i)}{\partial \theta^t_i}. \tag{14}$$

After fine-tuning, each AMoLE module $m$ performs federated aggregation as follows:

$$\mathbf{X}_{m,j} = \frac{1}{|\mathcal{C}_{m,j}|} \sum_{i \in \mathcal{C}_{m,j}} \mathbf{X}_{m,i,j} \tag{15}$$

$$\mathbf{Y}_m = \frac{1}{C} \sum_{i=1}^{C} \mathbf{Y}_{m,i}, \tag{16}$$

where $\mathcal{C}_{m,j}$ denotes the set of clients participating in fine-tuning domain expert $j$ within module $m$ in this round, (15) is applied to both $\mathbf{A}$ and $\mathbf{B}$, (16) is applied to both $\mathbf{A}^s$, $\mathbf{B}^s$ and $\mathbf{W}^t$.

### 4.4 Expert Assignment with Reverse Selection

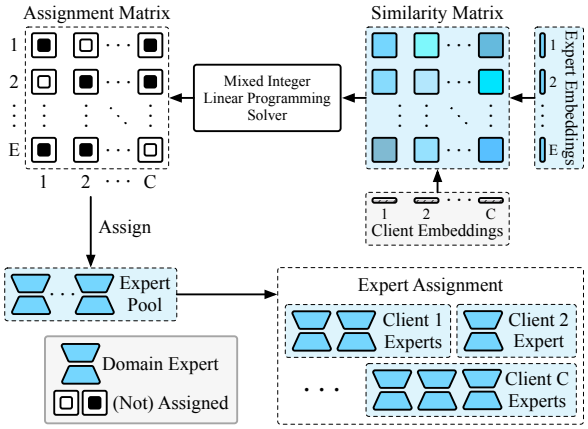



Figure 5: RSEA process. $E$ and $C$ denote the number of domain experts and clients, respectively.

To effectively assign domain experts to clients based on data relevance, we employ a mechanism to quantify the affinity between client data and expert domains using scaled dot product similarity.

Specifically, we define the embedding of client $i$ on module $m$ as the element-wise mean of token embeddings in $m$ for all tokens in the embedded set $\mathcal{D}^{emb}_i$ sampled from client $i$'s training data:

$$\overline{\mathbf{h}}^t_{m,i} = \frac{1}{|\mathcal{D}^{emb}_i|} \sum_{\mathbf{x} \in \mathcal{D}^{emb}_i} h^t_{m,i}(\mathbf{x}). \tag{17}$$

Note that the sampling of $\mathcal{D}^{emb}_i$ is random to ensure its data distribution aligns with $\mathcal{D}^{train}_i$, thereby better reflecting the local data characteristics.

Similarly, we define the embedding of domain expert $j$ in module $m$ as the element-wise mean of $j$'s embeddings across clients $\mathcal{C}_{m,j}$ fine-tuning $j$ in the current round (18), where $\overline{\mathbf{h}}^e_{m,i,j}$ is the embedding of $j$ on $i$, i.e., the element-wise mean of $j$'s embeddings in $m$ for all tokens in $\mathcal{D}^{emb}_i$ (19).

$$\overline{\mathbf{h}}^e_{m,j} = \frac{1}{|\mathcal{C}_{m,j}|} \sum_{i \in \mathcal{C}_{m,j}} \overline{\mathbf{h}}^e_{m,i,j} \tag{18}$$

$$\overline{\mathbf{h}}^e_{m,i,j} = \frac{1}{|\mathcal{D}^{emb}_i|} \sum_{\mathbf{x} \in \mathcal{D}^{emb}_i} h^e_{m,i,j}(\mathbf{x}) \tag{19}$$

The relevance score $s_{m,i,j}$ between client $i$'s data and expert $j$'s domain within module $m$ is then defined as the scaled dot product of their embeddings (20), aligning with the AMoLE training objective (4-6). A higher $s_{m,i,j}$ signifies a stronger affinity, indicating a greater likelihood of expert $j$ being selected by client $i$'s tokens in module $m$.

$$s_{m,i,j} = \frac{\left(\overline{\mathbf{h}}^{t}_{m,i}\right)^{T} \cdot \overline{\mathbf{h}}^{e}_{m,j}}{\sqrt{d}} \tag{20}$$

With the above relevance score, a naive solution would let clients select experts based on domain relevance; however, this approach exposes all expert parameters to clients—undesirable in cross-silo settings due to commercial sensitivities—and lacks a global perspective, risking suboptimal system-wide performance. To this end, we propose a **r**everse **s**election-based **e**xpert **a**ssignment (RSEA) strategy, where experts instead select clients. This ensures that clients remain unaware of irrelevant experts, preserving commercial privacy, while expert assignments are optimized from a global perspective to maximize overall federated performance.

Specifically, each domain expert $j$ in module $m$ selects the top-$k^c$ clients with highest relevance:

$$\mathcal{C}_{m,j} = \text{top-}k^c(\text{softmax}(s_{m,i,j})). \tag{21}$$

To minimize the expert count per module ($k^e$) and constraint expert heterogeneity (up to $b$ experts per module per client), expert assignment within each module is formulated as a mixed integer linear programming (MILP) problem, detailed in Appendix C.3.

# 5 Experiments

This section aims to experimentally demonstrate: (1) FedAMoLE achieves better accuracy on non-IID data (Section 5.2) (2) FedAMoLE has lower resource overhead than MoE-based approaches (Section 5.3) (3) The components proposed in this work contribute positively (Section 5.4-5.5).

## 5.1 Experimental Setup

### 5.1.1 Baselines

To assess FedAMoLE's efficacy, we evaluate it against seven competitive federated LLM fine-tuning methods, encompassing both non-personalized and personalized strategies. Non-personalized baselines include FedIT (Zhang et al., 2023a), FedPrompt, and FedPTuning (Kuang et al., 2023), representing federated averaging (McMahan et al., 2017) with LoRA (Hu et al., 2022), Prompt Tuning (Lester et al., 2021), and P-Tuning (Liu et al., 2022) respectively. For personalized baselines, we consider FedIT-FT, FedPrompt-FT, FedPTuning-FT (each extends its non-personalized counterpart with post-aggregation fine-tuning (Yu et al., 2020)), and FDLoRA (QI et al., 2024) (which employs shared and personalized adapters optimized via few-shot black-box techniques).

### 5.1.2 Datasets & Evaluation Metrics

To comprehensively compare the performance of all methods, we evaluate them on three widely used datasets: SNLI (Bowman et al., 2015), Dolly-15K (Conover et al., 2023), and Natural Instructions (NI) (Wang et al., 2022). Specifically, SNLI (570K sentence pairs, entailment/contradiction/neutral labels) serves as a natural language understanding benchmark, while Dolly-15K (15K question-response pairs, 8 tasks) and NI (v2.8 split, 756 train/119 test tasks) are used for natural language generation evaluation. Following prior federated LLM fine-tuning studies (Wu et al., 2024a; Qin et al., 2024a), we assess individual client model performance using the test accuracy after the last federated round. Aligned with personalized federated learning settings, we report **m**ean **t**est **a**ccuracy after the **l**ast round (MTAL) across clients (Appendix D.1.1). All experiments use LLaMA-3.2-1B as the foundation model.

### 5.1.3 FL Settings

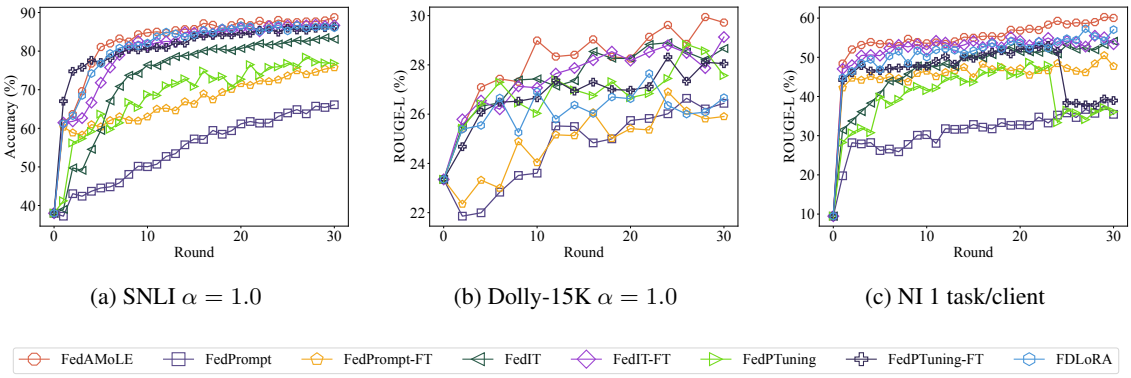
This work targets a cross-silo FL scenario with 10 clients, unless stated otherwise. Federated fine-tuning is conducted for 30 rounds, with each client performing 200 local training steps per round. Datasets are partitioned across clients to simulate non-IID data distributions (Appendix D.1.3).

## 5.2 Comparison of Accuracy

From Table 1, FedAMoLE achieves the best results across all five scenarios. Notably, on highly heterogeneous NI (1 task/client), it improves Rouge-L by 5.25% over the strongest baseline, FDLoRA; on SNLI with typical heterogeneity ($\alpha = 1.0$), FedAMoLE outperforms others, improving classification accuracy by 2.43% over the best baseline, FedIT-FT. This significant gain is due to FedAMoLE's data-aware, module-level adaptive expert assignment strategy, which builds personalized models tailored to each client's local data, en-

| | Approaches | Dolly-15K | | | NI | SNLI | Average |
|---|---|---|---|---|---|---|---|
| | | $\alpha = 0.1$ | $\alpha = 1.0$ | $\alpha = 100.0$ | 1 task/client | $\alpha = 1.0$ | |
| Vanilla | FedIT | 28.69±1.73 | 28.67±1.28 | 27.31±0.55 | 54.09±0.23 | 83.06±0.63 | 44.36 |
| | FedPrompt | 26.01±2.16 | 26.44±1.22 | 24.57±0.78 | 35.45±1.08 | 66.10±2.78 | 35.71 |
| | FedPTuning | 24.11±3.14 | 27.57±0.79 | 26.16±1.12 | 36.10±18.36 | 76.83±6.14 | 38.15 |
| Personalized | FedIT-FT | 28.80±0.16 | 29.13±1.18 | 27.17±0.40 | 53.34±0.26 | 86.67±0.58 | 45.02 |
| | FedPrompt-FT | 26.76±0.57 | 25.91±0.98 | 24.63±1.17 | 47.70±0.67 | 75.75±1.46 | 40.15 |
| | FedPTuning-FT | 27.28±1.28 | 28.05±1.42 | 25.89±0.43 | 38.97±20.72 | 86.34±3.20 | 41.30 |
| | FDLoRA | 27.75±0.17 | 26.67±0.66 | 25.79±1.99 | 57.04±1.90 | 86.06±1.32 | 44.66 |
| | FedAMoLE | **29.87±1.40** | **29.72±1.43** | **28.28±0.84** | **60.04±1.09** | **88.78±0.38** | **47.34** |
| | Gains | 3.71 | 2.02 | 3.55 | 5.25 | 2.43 | 5.14 |

Table 1: Comparison of MTAL on three tasks across different data heterogeneity scenarios. Each cell reports the mean and standard deviation of MTAL calculated over three random seeds. Each column represents the results under a specific data heterogeneity scenario, where the best-performing approach is highlighted in **bold**, the second-best method is underlined, and "gains" denote the relative MTAL improvement of FedAMoLE over the best baseline.

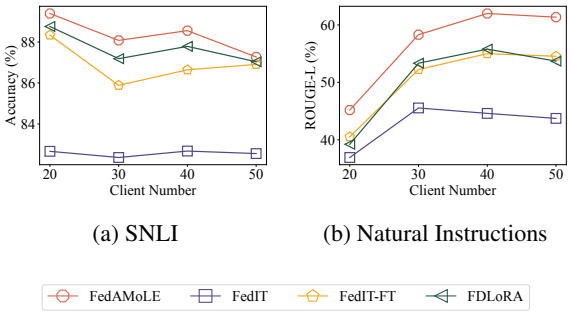



(a) SNLI $\alpha = 1.0$ (b) Dolly-15K $\alpha = 1.0$ (c) NI 1 task/client

Figure 6: MTA trends during training. Each point represents the MTA averaged over three random seeds at a round.

| Approaches | Mem (GB) | Up (MB) | Down (MB) |
|---|---|---|---|
| FedMoE | 13.89 | 1850 | 1850 |
| FedAMoLE | 7.99 | 25.66 | 25.63 |

Table 2: Comparison of memory, up- and down-link volumes. FedMoE data from (Mei et al., 2024); FedAMoLE data from one FL round on AG News using a switch transformer under FedMoE's setup.




(a) SNLI (b) Natural Instructions

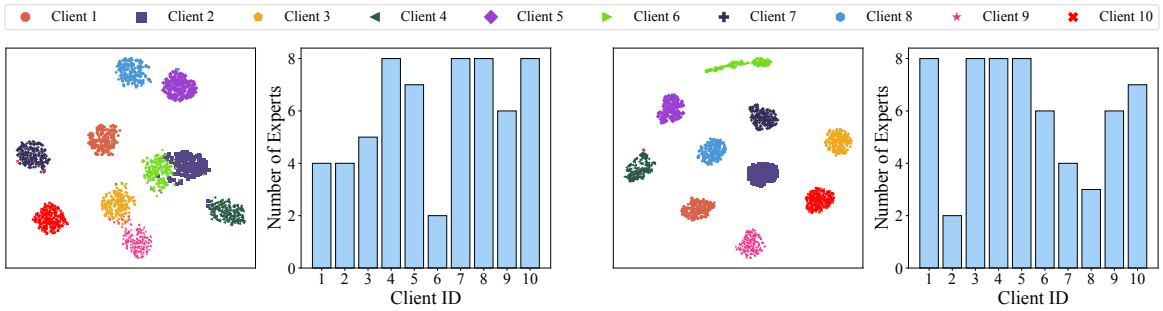
Figure 7: MTAL with different numbers of clients

hancing performance on heterogeneous data (5.4). Additionally, as shown in Figure 6, FedAMoLE converges quickly across three typical heterogeneous data scenarios and consistently outperforms other approaches during training. This indicates that even if training is stopped early for efficiency, FedAMoLE still achieves better results, demonstrating its practicality. This advantage comes from FedAMoLE's ability to dynamically adjust expert assignments per client in each round, allowing it to respond promptly to changes in relevance between experts and client data as training progresses.

Furthermore, Figure 7 illustrates the impact of increasing the number of clients. Notably, FedAMoLE consistently outperforms the strongest baselines on both natural language generation and understanding tasks under typical data heterogeneity settings, demonstrating its strong scalability.

### 5.3 Comparison of Overhead

FedAMoLE significantly reduces the memory and communication overhead compared to MoE-based methods. As shown in Table 2, FedMoE using a traditional MoE architecture incurs GB-level per-round communication cost, while FedAMoLE




(a) SNLI ($\alpha = 1.0$)

(b) Natural Instructions with one task per client

Figure 8: Visualization of the relevance between client data and domain experts in an AMoLE module after one round of fine-tuning. Each point in the scatter plot represents the t-SNE reduced relevance vector between a text sequence and all domain experts (computed as in (17-20), but in (17), $\mathcal{D}_i^{emb}$ contains only one example), with 200 points per client. The bar chart illustrates the number of domain experts assigned to each client in the next round.

| FedIT | FedMoLE | FedAMoLE | | | |
|---|---|---|---|---|---|
| +FT | | -A | -R | -S | |
| 86.45 | 87.70 | 86.90 | 88.05 | 87.50 | **88.95** |

Table 3: Ablation study. Each cell represents the MTAL obtained on SNLI with a single random seed.

leverages LoRA's efficiency to compress this communication overhead to 25.66 MB (approximately 1% of the original volume) and reduces memory usage to half that of FedMoE, which makes federated LLM fine-tuning practical in real-world scenarios.

### 5.4 Effectiveness of Personalization

Figure 8 shows the effectiveness of FedAMoLE's personalization. In the scatter plot, each point represents a text sequence's relevance to all domain experts. Points near the origin indicate uniform relevance—aligned with the global distribution and requiring fewer domain experts—while distant points show higher relevance to specific experts, reflecting divergence from global distribution and necessitating more experts. FedAMoLE's expert assignments follow this pattern: in Figure 8b, clients 2, 7, and 8 have points near the origin, indicating similar local and global data distributions; the bar chart confirms fewer experts assigned to these clients.

### 5.5 Ablation Study

To evaluate the contributions of the proposed AMoLE module, RSEA strategy, and shared expert, we design four ablation variants: (1) FedMoLE, where the RSEA strategy is removed and experts are randomly assigned in the first round, remaining fixed thereafter; (2) FedAMoLE-R, where the RSEA strategy is removed and experts are randomly assigned in each round; (3) FedAMoLE-A, which replaces the AMoLE module with a vanilla MoLE—here, each router has a uniform output dimension (equal to the total number of assignable experts) to ensure aggregation compatibility, and expert routing follows (24) but restricts top-$k^e$ selection to experts assigned in the current round; (4) FedAMoLE-S, where the shared expert is ablated, and only domain expert collaboration is used.

Table 3 presents the ablation study results. FedMoLE significantly outperforms FedIT-FT in accuracy, demonstrating the benefits of heterogeneous models. Full FedAMoLE further improves accuracy compared to FedMoLE, highlighting the advantages of data-driven model architectures. The performance degradation of both FedAMoLE-A and FedAMoLE-R, relative to full FedAMoLE, underscores the importance of the AMoLE module and the RSEA strategy. Moreover, the inferior performance of FedAMoLE-S compared to full FedAMoLE emphasizes the necessity of shared experts for general knowledge sharing, rather than relying solely on domain experts.

## 6 Conclusion

This work introduces FedAMoLE, a novel PFL framework for LLM fine-tuning, characterized by an adaptive MoLE module to enable architectural model heterogeneity, and a data-driven expert assignment strategy for architecture optimization. It overcomes the uniform and data-unaware model architectures of existing methods. Extensive experiments show that FedAMoLE achieves superior accuracy over existing methods in non-IID settings, highlight the potential of data-driven heterogeneous models for LLM fine-tuning with FL.

## Limitations

Despite its contributions, this work has some limitations. Firstly, while MoLE exhibits enhanced capability in handling data heterogeneity in FL, it introduces slightly increased memory and communication overhead compared to single LoRA approaches. Secondly, to ensure distributional alignment, the embedding set is randomly sampled from the training set. However, more refined sampling strategies may exist to further improve the accuracy of domain relevance evaluation. Additionally, to ensure data privacy and communication efficiency, domain relevance calculation utilizes dot product of embedding means, instead of the mean of embedding dot products, which may introduce potential inaccuracies. Furthermore, while parallel computation of domain experts is theoretically possible, serial implementation for simplicity led to suboptimal latency. Future work will address these limitations by investigating optimized embedding set sampling techniques, domain relevance evaluation algorithms and implementations.

## References


Abdulla Jasem Almansoori, Samuel Horváth, and Martin Takáč. 2024. Collaborative and efficient personalization with mixtures of adaptors.

Sara Babakniya, Ahmed Roushdy Elkordy, Yahya H. Ezzeldin, Qingfeng Liu, Kee-Bong Song, Mostafa El-Khamy, and Salman Avestimehr. 2023. SLoRA: Federated Parameter Efficient Fine-Tuning of Language Models.

Jiamu Bai, Daoyuan Chen, Bingchen Qian, Liuyi Yao, and Yaliang Li. 2024. Federated Fine-tuning of Large Language Models under Heterogeneous Tasks and Client Resources. *Preprint*, arXiv:2402.11505.

Suresh Bolusani, Mathieu Besançon, Ksenia Bestuzheva, Antonia Chmiela, João Dionísio, Tim Donkiewicz, Jasper van Doornmalen, Leon Eifler, Mohammed Ghannam, Ambros Gleixner, Christoph Graczyk, Katrin Halbig, Ivo Hedtke, Alexander Hoen, Christopher Hojny, Rolf van der Hulst, Dominik Kamp, Thorsten Koch, Kevin Kofler, Jurgen Lentz, Julian Manns, Gioni Mexi, Erik Mühmer, Marc E. Pfetsch, Franziska Schlösser, Felipe Serrano, Yuji Shinano, Mark Turner, Stefan Vigerske, Dieter Weninger, and Lixing Xu. 2024. The SCIP optimization suite 9.0. Technical Report, Optimization Online.

Samuel R. Bowman, Gabor Angeli, Christopher Potts, and Christopher D. Manning. 2015. A large annotated corpus for learning natural language inference. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 632–642, Lisbon, Portugal. Association for Computational Linguistics.

Tianshi Che, Ji Liu, Yang Zhou, Jiaxiang Ren, Jiwen Zhou, Victor Sheng, Huaiyu Dai, and Dejing Dou. 2023. Federated learning of large language models with parameter-efficient prompt tuning and adaptive optimization. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 7871–7888, Singapore. Association for Computational Linguistics.

Chaochao Chen, Xiaohua Feng, Jun Zhou, Jianwei Yin, and Xiaolin Zheng. 2023. Federated Large Language Model: A Position Paper.

Liam Collins, Hamed Hassani, Aryan Mokhtari, and Sanjay Shakkottai. 2021. Exploiting shared representations for personalized federated learning. In *Proceedings of the 38th International Conference on Machine Learning, ICML 2021, 18-24 July 2021, Virtual Event*, volume 139 of *Proceedings of Machine Learning Research*, pages 2089–2099. PMLR.

Mike Conover, Matt Hayes, Ankit Mathur, Jianwei Xie, Jun Wan, Sam Shah, Ali Ghodsi, Patrick Wendell, Matei Zaharia, and Reynold Xin. 2023. Free dolly: Introducing the world's first truly open instruction-tuned LLM. https://www.databricks.com/blog/2023/04/12/dolly-first-open-commercially-viable-instruction-tuned-llm.

Yuyang Deng, Mohammad Mahdi Kamani, and Mehrdad Mahdavi. 2020. Adaptive Personalized Federated Learning.

Nan Du, Yanping Huang, Andrew M. Dai, Simon Tong, Dmitry Lepikhin, Yuanzhong Xu, Maxim Krikun, Yanqi Zhou, Adams Wei Yu, Orhan Firat, Barret Zoph, Liam Fedus, Maarten P. Bosma, Zongwei Zhou, Tao Wang, Yu Emma Wang, Kellie Webster, Marie Pellat, Kevin Robinson, Kathleen S. Meier-Hellstern, Toju Duke, Lucas Dixon, Kun Zhang, Quoc V. Le, Yonghui Wu, Zhifeng Chen, and Claire Cui. 2022. Glam: Efficient scaling of language models with mixture-of-experts. In *International Conference on Machine Learning, ICML 2022, 17-23 July 2022, Baltimore, Maryland, USA*, volume 162 of *Proceedings of Machine Learning Research*, pages 5547–5569. PMLR.

Chen Dun, Mirian Hipolito Garcia, Guoqing Zheng, Ahmed Hassan Awadallah, Robert Sim, Anastasios Kyrillidis, and Dimitrios Dimitriadis. 2023. FedJETs: Efficient just-In-time personalization with federated mixture of experts.

Alireza Fallah, Aryan Mokhtari, and Asuman E. Ozdaglar. 2020. Personalized federated learning with theoretical guarantees: A model-agnostic meta-learning approach. In *Advances in Neural Information Processing Systems 33: Annual Conference*

*on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*.

Boyu Fan, Siyang Jiang, Xiang Su, Sasu Tarkoma, and Pan Hui. 2024a. A survey on model-heterogeneous federated learning: Problems, methods, and prospects. In *2024 IEEE International Conference on Big Data (BigData)*, pages 7725–7734, Washington, DC, USA. IEEE.

Dongyang Fan, Bettina Messmer, and Martin Jaggi. 2024b. On-Device Collaborative Language Modeling via a Mixture of Generalists and Specialists.

William Fedus, Barret Zoph, and Noam Shazeer. 2021. Switch transformers: Scaling to trillion parameter models with simple and efficient sparsity. *Journal of Machine Learning Research*, 23:120:1–120:39.

Chongyang Gao, Kezhen Chen, Jinmeng Rao, Baochen Sun, Ruibo Liu, Daiyi Peng, Yawen Zhang, Xiaoyuan Guo, Jie Yang, and V. S. Subrahmanian. 2024. Higher layers need more LoRA experts. *arXiv.org*, abs/2402.8562.

Avishek Ghosh, Jichan Chung, Dong Yin, and Kannan Ramchandran. 2020. An efficient framework for clustered federated learning. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*.

Binbin Guo, Yuan Mei, Danyang Xiao, Weigang Wu, Ye Yin, and Hongli Chang. 2020. PFL-MoE: Personalized Federated Learning Based on Mixture of Experts.

Filip Hanzely and Peter Richtárik. 2020. Federated Learning of a Mixture of Global and Local Models.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2022. Lora: Low-rank adaptation of large language models. In *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022*. OpenReview.net.

Yutao Huang, Lingyang Chu, Zirui Zhou, Lanjun Wang, Jiangchuan Liu, Jian Pei, and Yong Zhang. 2021. Personalized cross-silo federated learning on non-iid data. In *Thirty-Fifth AAAI Conference on Artificial Intelligence, AAAI 2021, Thirty-Third Conference on Innovative Applications of Artificial Intelligence, IAAI 2021, The Eleventh Symposium on Educational Advances in Artificial Intelligence, EAAI 2021, Virtual Event, February 2-9, 2021*, pages 7865–7873. AAAI Press.

Martin Isaksson, Edvin Listo Zec, Rickard Cöster, Daniel Gillblad, and Šarūnas Girdzijauskas. 2022. Adaptive expert models for personalization in federated learning.

Robert A. Jacobs, Michael I. Jordan, Steven J. Nowlan, and Geoffrey E. Hinton. 1991. Adaptive mixtures of local experts. *Neural Computation*, 3(1):79–87.

Feibo Jiang, Li Dong, Siwei Tu, Yubo Peng, Kezhi Wang, Kun Yang, Cunhua Pan, and Dusit Niyato. 2024. Personalized wireless federated learning for large language models.

Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. 2020. Scaling laws for neural language models. *ArXiv preprint*, abs/2001.08361.

Sai Praneeth Karimireddy, Satyen Kale, Mehryar Mohri, Sashank J. Reddi, Sebastian U. Stich, and Ananda Theertha Suresh. 2020. SCAFFOLD: stochastic controlled averaging for federated learning. In *Proceedings of the 37th International Conference on Machine Learning, ICML 2020, 13-18 July 2020, Virtual Event*, volume 119 of *Proceedings of Machine Learning Research*, pages 5132–5143. PMLR.

Weirui Kuang, Bingchen Qian, Zitao Li, Daoyuan Chen, Dawei Gao, Xuchen Pan, Yuexiang Xie, Yaliang Li, Bolin Ding, and Jingren Zhou. 2023. FederatedScope-LLM: A Comprehensive Package for Fine-tuning Large Language Models in Federated Learning.

Dmitry Lepikhin, HyoukJoong Lee, Yuanzhong Xu, Dehao Chen, Orhan Firat, Yanping Huang, Maxim Krikun, Noam Shazeer, and Zhifeng Chen. 2021. Gshard: Scaling giant models with conditional computation and automatic sharding. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net.

Brian Lester, Rami Al-Rfou, and Noah Constant. 2021. The power of scale for parameter-efficient prompt tuning. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 3045–3059, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Quentin Lhoest, Albert Villanova del Moral, Yacine Jernite, Abhishek Thakur, Patrick von Platen, Suraj Patil, Julien Chaumond, Mariama Drame, Julien Plu, Lewis Tunstall, Joe Davison, Mario Šaško, Gunjan Chhablani, Bhavitvya Malik, Simon Brandeis, Teven Le Scao, Victor Sanh, Canwen Xu, Nicolas Patry, Angelina McMillan-Major, Philipp Schmid, Sylvain Gugger, Clément Delangue, Théo Matussière, Lysandre Debut, Stas Bekman, Pierric Cistac, Thibault Goehringer, Victor Mustar, François Lagunas, Alexander Rush, and Thomas Wolf. 2021. Datasets: A community library for natural language processing. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 175–184, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Dengchun Li, Yingzi Ma, Naizheng Wang, Zhengmao Ye, Zhiyuan Cheng, Yinghao Tang, Yan Zhang, Lei Duan, Jie Zuo, Cal Yang, and Mingjie Tang.

2024. MixLoRA: Enhancing large language models fine-tuning with LoRA based mixture of experts. *arXiv.org*, abs/2404.15159.

Tian Li, Anit Kumar Sahu, Manzil Zaheer, Maziar Sanjabi, Ameet Talwalkar, and Virginia Smith. 2020a. Federated optimization in heterogeneous networks. In *Proceedings of Machine Learning and Systems 2020, MLSys 2020, Austin, TX, USA, March 2-4, 2020*. mlsys.org.

Xiang Li, Kaixuan Huang, Wenhao Yang, Shusen Wang, and Zhihua Zhang. 2020b. On the convergence of fedavg on non-iid data. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

Chin-Yew Lin. 2004. ROUGE: A package for automatic evaluation of summaries. In *Text Summarization Branches Out*, pages 74–81, Barcelona, Spain. Association for Computational Linguistics.

Tao Lin, Lingjing Kong, Sebastian U. Stich, and Martin Jaggi. 2020. Ensemble distillation for robust model fusion in federated learning. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*.

Xiao Liu, Kaixuan Ji, Yicheng Fu, Weng Tam, Zhengxiao Du, Zhilin Yang, and Jie Tang. 2022. P-tuning: Prompt tuning can be comparable to fine-tuning across scales and tasks. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 61–68, Dublin, Ireland. Association for Computational Linguistics.

Jun Luo, Chen Chen, and Shandong Wu. 2024a. Mixture of experts made personalized: Federated prompt learning for vision-language models.

Jun Luo and Shandong Wu. 2022. Adapt to adaptation: Learning personalization for cross-silo federated learning. In *Proceedings of the Thirty-First International Joint Conference on Artificial Intelligence, IJCAI 2022, Vienna, Austria, 23-29 July 2022*, pages 2166–2173. ijcai.org.

Tongxu Luo, Jiahe Lei, Fangyu Lei, Weihao Liu, Shizhu He, Jun Zhao, and Kang Liu. 2024b. MoELoRA: Contrastive learning guided mixture of experts on parameter-efficient fine-tuning for large language models. *arXiv.org*, abs/2402.12851.

Sourab Mangrulkar, Sylvain Gugger, Lysandre Debut, Younes Belkada, Sayak Paul, and Benjamin Bossan. 2022. PEFT: State-of-the-art parameter-efficient fine-tuning methods.

Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Agüera y Arcas. 2017. Communication-efficient learning of deep networks from decentralized data. In *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics, AISTATS 2017, 20-22 April 2017, Fort Lauderdale, FL, USA*, volume 54 of *Proceedings of Machine Learning Research*, pages 1273–1282. PMLR.

Hanzi Mei, Dongqi Cai, Ao Zhou, Shangguang Wang, and Mengwei Xu. 2024. FedMoE: Personalized Federated Learning via Heterogeneous Mixture of Experts.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Köpf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. 2019. Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 8024–8035.

Daniel Peterson, Pallika Kanani, and Virendra J. Marathe. 2019. Private Federated Learning with Domain Adaptation.

Jiaxing QI, Zhongzhi Luan, Shaohan Huang, Carol Fung, Hailong Yang, and Depei Qian. 2024. FDLoRA: Personalized Federated Learning of Large Language Model via Dual LoRA Tuning.

Zhen Qin, Daoyuan Chen, Bingchen Qian, Bolin Ding, Yaliang Li, and Shuiguang Deng. 2024a. Federated full-parameter tuning of billion-sized language models with communication cost under 18 kilobytes. In *Forty-first International Conference on Machine Learning, ICML 2024, Vienna, Austria, July 21-27, 2024*. OpenReview.net.

Zhen Qin, Shuiguang Deng, Mingyu Zhao, and Xueqiang Yan. 2023. Fedapen: Personalized cross-silo federated learning with adaptability to statistical heterogeneity. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, KDD 2023, Long Beach, CA, USA, August 6-10, 2023*, pages 1954–1964. ACM.

Zhen Qin, Zhaomin Wu, Bingsheng He, and Shuiguang Deng. 2024b. Federated data-efficient instruction tuning for large language models. *ArXiv preprint*, abs/2410.10926.

J. Rapin and O. Teytaud. 2018. Nevergrad - a gradient-free optimization platform.

Matthias Reisser, Christos Louizos, Efstratios Gavves, and Max Welling. 2021. Federated Mixture of Experts.

Noam Shazeer, Azalia Mirhoseini, Krzysztof Maziarz, Andy Davis, Quoc V. Le, Geoffrey E. Hinton, and Jeff Dean. 2017. Outrageously large neural networks: The sparsely-gated mixture-of-experts layer. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net.

Tao Shen, Jie Zhang, Xinkang Jia, Fengda Zhang, Gang Huang, Pan Zhou, Kun Kuang, Fei Wu, and Chao Wu. 2020. Federated Mutual Learning.

Youbang Sun, Zitao Li, Yaliang Li, and Bolin Ding. 2024. Improving LoRA in privacy-preserving federated learning. *ArXiv preprint*, abs/2403.12313.

Alysa Ziying Tan, Han Yu, Lizhen Cui, and Qiang Yang. 2023. Towards Personalized Federated Learning. *IEEE Transactions on Neural Networks and Learning Systems*, 34(12):9587–9603.

Yue Tan, Guodong Long, Lu Liu, Tianyi Zhou, Qinghua Lu, Jing Jiang, and Chengqi Zhang. 2022. Fedproto: Federated prototype learning across heterogeneous clients. In *Thirty-Sixth AAAI Conference on Artificial Intelligence, AAAI 2022, Thirty-Fourth Conference on Innovative Applications of Artificial Intelligence, IAAI 2022, The Twelveth Symposium on Educational Advances in Artificial Intelligence, EAAI 2022 Virtual Event, February 22 - March 1, 2022*, pages 8432–8440. AAAI Press.

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

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, Aurelien Rodriguez, Armand Joulin, Edouard Grave, and Guillaume Lample. 2023. LLaMA: Open and Efficient Foundation Language Models. *Preprint*, arXiv:2302.13971.

Haodi Wang, Tangyu Jiang, Yu Guo, Fangda Guo, Rongfang Bie, and Xiaohua Jia. 2024. Label noise correction for federated learning: A secure, efficient and reliable realization. In *2024 IEEE 40th International Conference on Data Engineering (ICDE)*, pages 3600–3612. IEEE.

Yizhong Wang, Swaroop Mishra, Pegah Alipoormolabashi, Yeganeh Kordi, Amirreza Mirzaei, Atharva Naik, Arjun Ashok, Arut Selvan Dhanasekaran, Anjana Arunkumar, David Stap, Eshaan Pathak, Giannis Karamanolakis, Haizhi Lai, Ishan Purohit, Ishani Mondal, Jacob Anderson, Kirby Kuznia, Krima Doshi, Kuntal Kumar Pal, Maitreya Patel, Mehrad Moradshahi, Mihir Parmar, Mirali Purohit, Neeraj Varshney, Phani Rohitha Kaza, Pulkit Verma, Ravsehaj Singh Puri, Rushang Karia, Savan Doshi, Shailaja Keyur Sampat, Siddhartha Mishra, Sujan Reddy A, Sumanta Patro, Tanay Dixit, and Xudong Shen. 2022. Super-NaturalInstructions: Generalization via declarative instructions on 1600+ NLP tasks. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 5085–5109, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Kang Wei, Jun Li, Ming Ding, Chuan Ma, Howard H Yang, Farhad Farokhi, Shi Jin, Tony QS Quek, and H Vincent Poor. 2020. Federated learning with differential privacy: Algorithms and performance analysis. *IEEE transactions on information forensics and security*, 15:3454–3469.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. 2019. HuggingFace's Transformers: State-of-the-art Natural Language Processing.

Feijie Wu, Zitao Li, Yaliang Li, Bolin Ding, and Jing Gao. 2024a. FedBiOT: LLM local fine-tuning in federated learning without full model. volume abs/2406.17706.

Panlong Wu, Kangshuo Li, Ting Wang, Yanjie Dong, Victor C. M. Leung, and Fangxin Wang. 2024b. FedFMSL: Federated learning of foundation models with sparsely activated LoRA. *IEEE Transactions on Mobile Computing*, 23(12):15167–15181.

Yuexiang Xie, Zhen Wang, Dawei Gao, Daoyuan Chen, Liuyi Yao, Weirui Kuang, Yaliang Li, Bolin Ding, and Jingren Zhou. 2022. Federatedscope: A flexible federated learning platform for heterogeneity. *ArXiv preprint*, abs/2204.05011.

Mengwei Xu, Dongqi Cai, Yaozong Wu, Xiang Li, and Shangguang Wang. 2024. {FwdLLM}: Efficient federated finetuning of large language models with perturbed inferences. In *2024 USENIX Annual Technical Conference (USENIX ATC 24)*, pages 579–596.

Yiyuan Yang, Guodong Long, Tao Shen, Jing Jiang, and Michael Blumenstein. 2024. Dual-personalizing adapter for federated foundation models.

Mang Ye, Xiuwen Fang, Bo Du, Pong C. Yuen, and Dacheng Tao. 2024. Heterogeneous federated learning: State-of-the-art and research challenges. *ACM Computing Surveys*, 56(3):1–44.

Liping Yi, Han Yu, Chao Ren, Heng Zhang, Gang Wang, Xiaoguang Liu, and Xiaoxiao Li. 2024. pFedMoE: Data-Level Personalization with Mixture of Experts for Model-Heterogeneous Personalized Federated Learning.

Tao Yu, Eugene Bagdasaryan, and Vitaly Shmatikov. 2020. Salvaging Federated Learning by Local Adaptation.

Edvin Listo Zec, Olof Mogren, John Martinsson, Leon René Sütfeld, and Daniel Gillblad. 2020. Specialized federated learning using a mixture of experts.

Ziwei Zhan, Wenkuan Zhao, Yuanqing Li, Weijie Liu, Xiaoxi Zhang, Chee Wei Tan, Chuan Wu, Deke Guo, and Xu Chen. 2024. FedMoE-DA: Federated Mixture of Experts via Domain Aware Fine-grained Aggregation.

Jianyi Zhang, Saeed Vahidian, Martin Kuo, Chunyuan Li, Ruiyi Zhang, Guoyin Wang, and Yiran Chen. 2023a. Towards building the federatedgpt: Federated instruction tuning. In *IEEE International Conference on Acoustics, Speech, and Signal Processing*, pages 6915–6919. arXiv.

Pengyu Zhang, Yingbo Zhou, Ming Hu, Junxian Feng, Jiawen Weng, and Mingsong Chen. 2024. Personalized Federated Instruction Tuning via Neural Architecture Search.

Zhuo Zhang, Yuanhang Yang, Yong Dai, Qifan Wang, Yue Yu, Lizhen Qu, and Zenglin Xu. 2023b. FedPETuning: When federated learning meets the parameter-efficient tuning methods of pre-trained language models. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 9963–9977, Toronto, Canada. Association for Computational Linguistics.

Wayne Xin Zhao, Kun Zhou, Junyi Li, Tianyi Tang, Xiaolei Wang, Yupeng Hou, Yingqian Min, Beichen Zhang, Junjie Zhang, Zican Dong, et al. 2023. A survey of large language models. *ArXiv preprint*, abs/2303.18223.

Zhuangdi Zhu, Junyuan Hong, and Jiayu Zhou. 2021. Data-free knowledge distillation for heterogeneous federated learning. In *Proceedings of the 38th International Conference on Machine Learning, ICML 2021, 18-24 July 2021, Virtual Event*, volume 139 of *Proceedings of Machine Learning Research*, pages 12878–12889. PMLR.

# A More Related Work

## A.1 Federated Fine-Tuning for LLMs

Current federated fine-tuning approaches for LLMs often adopt PEFT techniques to reduce memory usage during local fine-tuning and communication cost for the transmission of trainable parameters. Some studies focus on establishing benchmarks (Zhang et al., 2023b) and infrastructures (Kuang et al., 2023) for federated LLM fine-tuning. Other works like (Sun et al., 2024) address inconsistencies between local and global objectives in LoRA fine-tuning to improve performance. Additional efforts focus on reducing resource overhead, including leveraging quantization techniques to reduce memory footprint (Xu et al., 2024), compressing model updates into fixed random seeds and loss values through zero-order optimization to minimize communication costs (Qin et al., 2024a), fine-tuning compressed LLMs to reduce computational burden (Wu et al., 2024a), and optimizing resource utilization across heterogeneous clients with SVD-based LoRA pruning (Bai et al., 2024). While the above studies have significantly improved model accuracy, advanced the ecosystem, and minimized computational, memory, and communication overheads in federated LLM fine-tuning, they overlook the challenge of data heterogeneity, which causes performance degradation in practical FL applications.

## A.2 Personalized Federated Learning

To address the challenges posed by data heterogeneity, personalized federated learning (PFL) has been widely studied. Many approaches focus on parameter-level personalization by adapting model parameters to client-specific distributions through techniques such as fine-tuning (Yu et al., 2020), regularization (Li et al., 2020a; Karimireddy et al., 2020), multi-task learning (Huang et al., 2021), meta learning (Fallah et al., 2020), model interpolation (Hanzely and Richtárik, 2020; Deng et al., 2020; Luo and Wu, 2022), clustering (Ghosh et al., 2020), and parameter decoupling (Collins et al., 2021). Although these methods help mitigate performance degradation, their homogeneous model architectures limit personalization capability. To further tailor models to individual clients, some works explore architecture-level personalization using knowledge distillation (Lin et al., 2020; Zhu et al., 2021), mutual learning (Shen et al., 2020), prototype learning (Tan et al., 2022), and ensemble learning (Qin et al., 2023). However, knowledge distillation often requires additional data (e.g., public datasets), which contradicts the federated learning goal of addressing data scarcity. Moreover, prototype learning is challenging to apply to LLM fine-tuning because its reliance on fixed prototypes may not capture the dynamic variability required for text generation. Meanwhile, mutual learning and ensemble learning necessitate training multiple models locally—an impractical solution given the high resource cost of training a single LLM.

## A.3 Personalized Federated Learning with Mixture of Experts

Mixture-of-Experts (MoE) architectures have attracted attention in PFL due to their ability to achieve model heterogeneity while maintaining constant computational overhead. Early MoE-based PFL research focused on small-scale models like CNNs and RNNs, often treating the entire model as a single expert. Some works (Peterson et al., 2019; Guo et al., 2020; Zec et al., 2020; Yi et al., 2024) combined a shared expert with a personalized expert using a gating network. Others supported mixing multiple experts through

personalized weighting coefficients (Reisser et al., 2021), clustering (Isaksson et al., 2022), client selection (Dun et al., 2023), or similarity-based expert aggregation (Zhan et al., 2024). However, these works are primarily designed for small-scale neural networks and incur significant memory and communication overhead, making them unsuitable for LLMs.

# B Preliminaries

## B.1 Mixture of Experts and A Lightweight Improvement

Mixture of experts (MoE) (Jacobs et al., 1991; Shazeer et al., 2017; Lepikhin et al., 2021; Du et al., 2022; Fedus et al., 2021) is a technology that enables the scaling of LLMs with constant computational overhead. A typical MoE layer consists of a linear router and multiple sub-modules termed "experts". Given a specific input, the router sparsely selects a fixed number of experts based on its features. This process can be formalized as

$$\begin{aligned} p_i &= \text{softmax}(\mathbf{W}_r\mathbf{x})_i \\ \mathcal{T} &= \text{top-}k(p_i) \\ \mathbf{y} &= \sum_{i\in\mathcal{T}} p_i E_i(\mathbf{x}), \end{aligned} \tag{22}$$

where $\mathbf{x} \in \mathbb{R}^d$ represents the input features, $\mathbf{W}_r \in \mathbb{R}^{N\times d}$ denotes the parameter weights of the router ($N$ is the total number of experts), $p_i$ represents the weighting coefficient of expert $i$ for the input $\mathbf{x}$, $\mathcal{T}$ is the set of indices of the $k$ selected experts, $E_i$ is the $i$-th expert sub-module (such as a two-layer feed-forward neural network), and $\mathbf{y}$ is the final output.

The adaptive collaboration of domain experts in MoE enables effective adaptation to diverse data distributions, making it a promising solution for personalizing client models in FL. However, applying MoE to federated LLM fine-tuning incurs significant communication costs due to the transfer of dense expert sub-modules between the server and clients. A natural solution to this limitation is to make MoE experts more lightweight.

Mixture-of-LoRA-Experts (MoLE) architectures (Li et al., 2024; Luo et al., 2024b; Gao et al., 2024) address this by using LoRA adapters as experts within the MoE framework.

Low-rank adaptation (LoRA) (Hu et al., 2022) is a widely adopted Parameter-Efficient Fine-Tuning (PEFT) technique for LLMs. LoRA assumes that the fine-tuning updates $\Delta\mathbf{W}$ to the pre-trained parameters $\mathbf{W} \in \mathbb{R}^{h\times d}$ are highly sparse and can be approximated by the product of two low-rank matrices $\mathbf{A} \in \mathbb{R}^{r\times d}$ and $\mathbf{B} \in \mathbb{R}^{h\times r}$, known as LoRA adapters, with $r \ll \min(d, h)$ to minimize parameter size. The fine-tuned parameters $\mathbf{W}$ during forward computation can then be expressed as:

$$\mathbf{y} = \mathbf{W}\mathbf{x} + \Delta\mathbf{W}\mathbf{x} = \mathbf{W}\mathbf{x} + \mathbf{B}\mathbf{A}\mathbf{x}. \tag{23}$$

By leveraging the lightweight nature of LoRA adapters, MoLE substantially reduces the parameter size of MoE experts, enabling it to achieve the same level of local model personalization as MoE while significantly lowering communication overhead. The forward computation of MoLE is given by:

$$\begin{aligned} p_i &= \text{softmax}(\mathbf{W}_r\mathbf{x})_i \\ \mathcal{T} &= \text{top-}k(p_i) \\ \mathbf{y} &= \mathbf{W}\mathbf{x} + \sum_{i\in\mathcal{T}} p_i \mathbf{B}_i\mathbf{A}_i\mathbf{x}. \end{aligned} \tag{24}$$

## B.2 Limitations of the Vanilla MoLE Module

Assigning varying numbers of domain experts to instances of a traditional MoLE module across clients challenges federated aggregation. Consider the following scenario: if two client instances of a MoLE module are assigned to three and five domain experts, respectively, the corresponding router dimensions would be $3 \times d$ and $5 \times d$, as defined in (24). This dimensional inconsistency prevents direct aggregation of routers via FedAvg (McMahan et al., 2017). The core issue originates that the parameter dimension of a traditional MoLE router is tied to the number of domain experts, resulting in shape mismatches across module instances with varying domain expert counts.

# C Approach Details

## C.1 Loss Function

The loss function used during local fine-tuning is defined as:

$$\mathcal{L}_i(\mathcal{B} \mid \theta_i) = \frac{1}{|\mathcal{B}|}\sum_{\mathbf{x}\in\mathcal{B}}\left[\mathcal{L}_i(\mathbf{x} \mid \theta_i) + \beta \cdot \mathcal{L}_i^{\text{LB}}(\mathcal{B} \mid \theta_i)\right], \tag{25}$$

where $\mathcal{B}$ is a mini-batch of text sequences containing $T$ tokens sampled from client $i$'s training dataset $\mathcal{D}_i^{train}$, $\mathcal{L}_i$ represents the NLL loss (refer to (3)), and

$$\mathcal{L}_i^{\text{LB}}(\mathcal{B}|\theta_i) = \sum_{m\in\mathcal{M}}\left(|\mathcal{E}_{m,i}|\sum_{j\in\mathcal{E}_{m,i}} \overline{f}_{m,i,j}\cdot\overline{p}_{m,i,j}\right) \tag{26}$$

represents the load balance loss (Fedus et al., 2021) weighted by $\beta$, which aims to encourage each domain expert to process roughly the same number of tokens, ensuring sufficient training. $\overline{f}_{m,i,j}$ and $\overline{p}_{m,i,j}$ are defined as follows:

$$\begin{aligned}\overline{f}_{m,i,j} &= \frac{1}{T}\sum_{\mathbf{X}\in\mathcal{B}}\sum_{\mathbf{x}\in\mathbf{X}}\mathbb{1}\{\arg\max_{j'} p_{m,i,j'}(\mathbf{x}) = j\} \\ \overline{p}_{m,i,j} &= \frac{1}{T}\sum_{\mathbf{X}\in\mathcal{B}}\sum_{\mathbf{x}\in\mathbf{X}} p_{m,i,j}(\mathbf{x}).\end{aligned} \tag{27}$$

### C.2 Privacy Analysis

The computation of expert $j$'s embedding $\overline{\mathbf{h}}^e_{m,j}$ requires each client $i$ to upload their local $\overline{\mathbf{h}}^e_{m,i,j}$. Since $\overline{\mathbf{h}}^e_{m,i,j}$ represents the average of client $i$'s local token features. Privacy concerns on this can be solved by mature methods such as differential privacy (Wei et al., 2020), so this work does not further discuss on these approaches.

### C.3 Domain Expert Assignment Problem

We model the domain expert assignment in module $m$ as the following optimization problem:

**Definition C.1.** *(domain expert assignment problem)*

$$\begin{aligned}&\max_{\mathbf{D}_m} \langle \mathbf{P}_m, \mathbf{D}_m \rangle \\ \textit{s.t.}\quad &\forall i : k^e \le \sum_j d_{m,i,j} \le b \\ &\forall j : \sum_i d_{m,i,j} = k^c \\ &\forall i,j : d_{m,i,j} \in \{0,1\},\end{aligned}$$

where $\mathbf{P}_m = \{p_{m,i,j}\}$ is the selection probability matrix, with

$$p_{m,i,j} = \frac{\mathrm{e}^{s_{m,i,j}}}{\sum_i \mathrm{e}^{s_{m,i,j}}} \tag{28}$$

representing the probability that domain expert $j$ in module $m$ selects client $i$. $\mathbf{D}_m = \{d_{m,i,j}\}$ is the assignment matrix, where $d_{m,i,j}$ is a binary variable indicating whether domain expert $j$ in module $m$ is assigned to client $i$ (1 for assigned, 0 for not assigned). The optimization objective is to maximize the inner product of $\mathbf{P}_m$ and $\mathbf{D}_m$, aiming to make the final domain expert assignment scheme align as closely as possible with the domain experts' selection preferences. The first constraint ensures that the number of domain experts assigned to each client is between $k^e$ and $b$; the second constraint ensures that each domain expert selects $k^c$ clients.

Problem C.1 can be reduced to a mixed integer linear programming (MILP) problem, which is NP-hard. However, in the domain expert assignment scenario, the problem size is typically small (a few hundred decision variables) and can be quickly solved using solvers such as Gurobi or SCIP.

## D More Experiments

### D.1 Experimental Setup Details

#### D.1.1 Definition of MTAL

The **m**ean **t**est **a**ccuracy achieved by each personalized LLM after the **l**ast round (MTAL) is defined as:

$$\text{MTAL} = \text{MTA}[-1]. \tag{29}$$

Here, MTA is the mean test accuracy:

$$\text{MTA}[t] = \frac{1}{C}\sum_{i=1}^{C} acc_i[t], \tag{30}$$

where $acc_i[t]$ denotes the performance metric of client $i$'s local model at round $t$. For SNLI, $acc$ represents classification accuracy. For Dolly-15K and NI, we use Rouge-L (Lin, 2004) as the evaluation metric for the local model, as adopted in a series of federated LLM tuning approaches (Qin et al., 2024a; Bai et al., 2024).

#### D.1.2 Implementation

In the experiments, all approaches are implemented using `PyTorch 2.4.0` (Paszke et al., 2019) along with the libraries of `Transformers` (Wolf et al., 2019), `PEFT` (Mangrulkar et al., 2022), and `Datasets` (Lhoest et al., 2021). Experiments are conducted on a server with 8 NVIDIA RTX 4090 GPUs with models loaded in `BF16` format, unless stated otherwise.

Unless otherwise specified, all approaches use the Adam optimizer with a batch size of 1, a learning rate $\eta = 5e^{-5}$, and a learning rate decay of 0.99 per round. For LoRA-based approaches, we set $r = 8$, $\alpha = 16$, and a dropout rate of 5%, applying LoRA to the $\mathbf{Q}$ and $\mathbf{V}$ matrices in the self-attention layers. For Prompt Tuning, the virtual prompt is initialized using the soft prompt, whose token count is used as the number of virtual tokens. For P-Tuning, we set the virtual token count to 20 and use an MLP as the prompt encoder. In FDLoRA, the InnerOpt phase synchronizes the shared and personalized adapters every 5 rounds; the OuterOpt phase uses a

| Approaches | Mem (MB) | Up (MB) | Down (MB) | Train (s) | Aggregation (s) | Test (s) |
|---|---|---|---|---|---|---|
| FedIT-FT | 2782.11 | 1.64 | 1.64 | 4.85 | 0.02 | 2.37 |
| FedPrompt-FT | 2827.18 | 0.29 | 0.29 | 4.47 | 0.00 | 3.20 |
| FedPTuning-FT | 2841.88 | 0.16 | 0.16 | 4.19 | 0.00 | 2.99 |
| FDLoRA | 2775.86 | 1.64 | 1.64 | 4.14 | 0.03 | 2.60 |
| FedAMoLE (15 experts) | 2788.95 | 7.59 | 7.59 | 13.25 | 4.97 | 4.92 |
| FedAMoLE (20 experts) | 2792.07 | 9.24 | 9.23 | 14.44 | 5.20 | 5.20 |
| FedAMoLE (25 experts) | 2812.88 | 10.88 | 10.87 | 15.57 | 5.49 | 5.63 |
| FedAMoLE (30 experts) | 2797.96 | 12.52 | 12.51 | 16.49 | 5.85 | 5.72 |

Table 4: Comparison of system overhead. All the statistics were obtained by performing one round of federated fine-tuning using a server with one NVIDIA RTX 3090 GPU. These columns present the maximum memory usage, upload data, download data, training latency, aggregation latency, and testing (inference) latency per client in one round. For FDLoRA, the black-box optimization is included in the testing latency. For FedAMoLE, the RSEA process is included in the aggregation latency, and system performance is evaluated under different total numbers of experts per module.

learning rate of 1.0 and a momentum of 0.5; the FusionOpt phase uses `nevergrad` (Rapin and Teytaud, 2018) as the black-box optimizer with an L1 regularization term weighted by 0.05. For FedAMoLE, we use SCIP (Bolusani et al., 2024) to solve the optimization problem, setting the total assignable experts per module to 30, with $k^e = 2$, $k^c = 2$, $b = 8$, and $\beta = 1e^{-3}$.

For Dolly-15K and NI, we use the Alpaca (Taori et al., 2023) prompt template with the following soft prompt:

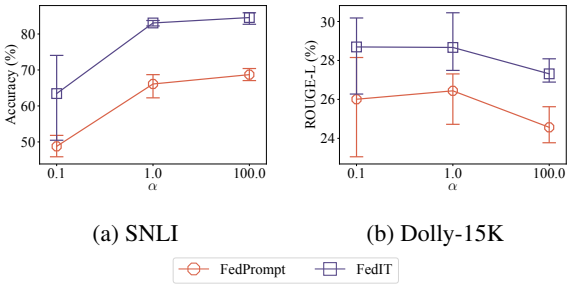
> **Soft Prompt**: Generate a clear and direct response that accurately fulfills the task based on the provided instruction and any given context.

For experiments on SNLI, the adopted prompt template and soft prompt are as follows:

> **Prompt Template**: Suppose {premise} Can we infer that "{hypothesis}"? Yes, No, or Maybe?\n
> **Soft Prompt**: Based on the premise, determine if the hypothesis logically follows. Answer with 'Yes' if it does, 'No' if it doesn't, or 'Maybe' if uncertain.

#### D.1.3 Non-IID Simulation

For SNLI and Dolly-15K, we partition the dataset among 10 clients with Dirichlet distributions with $\alpha$ values of 0.1, 1.0, and 100.0 to build label-skewed (Chen et al., 2023) non-IID scenarios. For NI, we select 10 tasks with the largest sample sizes, assigning each client a unique task to create a feature-skewed (Tan et al., 2023) non-IID scenario. Each client randomly samples 80% of its local data as the training set, 10% as the validation set, and 10% as the test set. Due to computational constraints, validation and test sets for SNLI are capped at 200 samples each, while for Dolly-15K and NI, the validation set is capped at 200 samples and the test set at 50 samples.

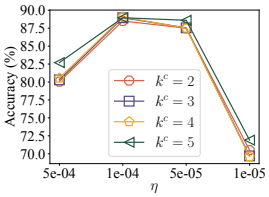



(a) SNLI (b) Dolly-15K

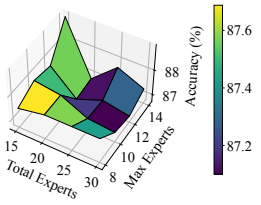
Figure 9: MTAL of vanilla FL approaches across different data heterogeneity scenarios. Each data point represents the average result over three random seeds, with error bars showing the range between the best and worst values across the three seeds.

### D.2 Necessity of Personalization in Non-IID Settings

Figure 9 illustrates the performance of two vanilla FL approaches across varying data heterogeneity. On SNLI, performance declines significantly as heterogeneity increases, aligning with findings from existing FL works for relatively small models (Tan et al., 2023). Our results confirm that LLM federated fine-tuning is similarly affected, underscoring the need for personalization mechanisms. In contrast, on Dolly-15K, accuracy remains stable despite data heterogeneity, likely because the dataset has fewer task types (8) with smaller inter-task differences, limiting the impact of heterogeneous partitioning.




(a) The impact on accuracy by the learning rate $\eta$ and the number of clients reversely selected by each expert.




(b) The impact by the number of total experts and maximum experts per module on the accuracy of fine-tuned LLMs.

Figure 10: Hyperparameter sensitivity of FedAMoLE. All the statistics are obtained on the SNLI dataset.

Figure 6 demonstrates the effectiveness of personalization in mitigating performance degradation caused by data heterogeneity. On SNLI and NI (Figs.6a and 6c), post-aggregation fine-tuning noticeably improves the performance of vanilla FL approaches, while FDLoRA further enhances this with its dynamic fusion of shared and personalized adapters. FedAMoLE, incorporating the AMoLE module and RSEA strategy, achieves even greater personalization and consistently outperforms all baselines. On Dolly-15K, where data heterogeneity is limited, the advantages of personalization are less pronounced, yet FedAMoLE maintains its leading performance, showcasing its versatility.

### D.3 Hyper-parameter Sensitivity

As shown in Figure 10a, the performance of FedAMoLE is minimally affected by the number of clients each expert selects. However, over-large or over-small the learning rate declines the performance of FedAMoLE, indicating the need for an appropriate learning rate to ensure optimal performance.

Figure 10b illustrates the impact of the total number of experts per module and the maximum number of experts per client on FedAMoLE's performance. It shows that a smaller number of experts per module requires more heterogeneous assignments (larger max experts per client), while a larger number of experts per module benefits from more balanced assignments (smaller max experts per client) to achieve optimal performance. Overall, these two parameters have limited impact, with model accuracy fluctuating within 1.5%, demonstrating FedAMoLE's robustness to hyperparameter settings.

### D.4 Detailed Comparison of Overhead

Table 4 presents the system efficiency of various PFL baselines and FedAMoLE. We observe that despite employing multiple experts per module, FedAMoLE's memory usage remains comparable to the baselines and is nearly unaffected by the total number of experts per module. This is because each expert is a lightweight LoRA adapter with negligible parameters compared to the pre-trained model. Even when the total number of experts per module is set to 30—a relatively large number—the per-client upload and download data is only 12.52 MB, which is higher than the baselines but acceptable with modern high-speed internet.

FedAMoLE also maintains acceptable per-round training and inference latencies of 16.49 s and 5.72 s, respectively, even when fine-tuning a frozen 1B-parameter LLM with up to 30 experts. The increased latency is primarily due to two factors: the RSEA strategy—which requires clients to perform model inference on an embedding dataset—and the current implementation of the AMoLE module, which executes domain experts sequentially for simplicity. Despite these factors, the latency remains manageable.