HFL-LoRA: Hierarchical Federated Low-Rank Adaptation for Privacy-Preserving Domain Specialization of Large Language Models

Garvilkumar Shah*, Gaurav Shivaprasad*, Atishay Jain*
*Department of Computer Science, Illinois Institute of Technology, Chicago, IL, United States

Abstract*—The deployment of Large Language Models (LLMs) in decentralized organizational environments is hindered by the dual challenges of data heterogeneity and communication inefficiency. Traditional Federated Learning (FL) approaches like FedAvg struggle with non-IID data distributions across different departments (e.g., Engineering, Finance), leading to client drift and suboptimal convergence. This paper proposes a novel Hierarchical Federated Learning framework with Low-Rank Adaptation (HFL-LoRA) that addresses these challenges through a decomposed aggregation strategy. We decouple LoRA's rank-decomposition matrices into two sets: up-projection matrices (B) are kept strictly local to capture domain-specific output features (e.g., department syntax), while down-projection matrices (A) are aggregated globally to facilitate general feature extraction. This Personalized-B / Global-A strategy effectively mitigates negative transfer without requiring complex clustering mechanisms. To further stabilize training, we introduce a residual aggregation scheme with momentum ($\mu = 0.1$) applied to the global A-matrices, along with a tunable global mixing parameter ($\lambda = 0.3$). Extensive simulations on a multi-department conversational dataset using the Qwen1.5-1.8B model demonstrate that our approach achieves a 29% reduction in final training loss (0.53 vs. 0.75) relative to FedAvg. The proposed framework also maintains high communication efficiency, reducing data transfer by 180× compared to full model training, making it an ideal solution for privacy-preserving, personalized LLM fine-tuning in heterogeneous enterprise environments. The implementation code and experimental scripts are publicly available at https://github.com/Gauravs-2k/HFL-LoRA

***Index Terms*—Federated Learning, LoRA, Large Language Models, Privacy, Hierarchical Aggregation, Clustering, Enterprise AI**

I. Introduction

The transformer-based architecture of Large Language Models (LLMs like Qwen, LLaMA, GPT-style models, and others) has transformed the enterprise AI sector. Their ability to handle unstructured natural language, perform contextual reasoning, and adjust to domain-specific tasks has enabled organisations to automate, speed up, and streamline processes across various business departments. Enterprises are therefore increasingly integrating LLMs for tasks such as financial interpretation of contracts and audit documents, clarifying HR policies and assessing compliance, retrieving engineering knowledge and code-based reasoning, and routing IT support tickets and resolving related problems. In practice, these models function as intelligent assistants that allow employees to synthesise information, answer domain-related queries, and navigate organisational processes more efficiently.

Regardless of their potential, applying LLMs in real-world enterprise environments presents a set of unique challenges that go beyond the capabilities of traditional fine-tuning pipelines.

The primary and most fundamental obstacle is the **structural siloing** of enterprise datasets. Finance, Human Resources, Engineering, and IT Support departments maintain strictly separated data domains governed by compliance rules, legal restrictions, and internal role-based access controls. Financial datasets include sensitive ledger entries, budget proposals, quarterly forecasts, and regulatory documentation. HR datasets contain employee performance reviews, HR inquiries, onboarding materials, and policy-driven communications. Engineering departments produce technically dense documents, code explanations, system diagrams, and debugging transcripts. IT Support manages troubleshooting histories, customer–agent dialogues, and diagnostic logs.

Most importantly, all of these data sources contain highly sensitive information protected by regulations such as GDPR, SOX, PCI-DSS, HIPAA, as well as internal privacy policies and access-control audits. Consequently, **raw departmental text cannot be centralised** for traditional supervised fine-tuning. Even de-identification approaches are insufficient because semantic nuances, phrasing, and domain structures can still reveal sensitive patterns.

Another significant challenge is the **non-IID (non-independent and identically distributed)** nature of enterprise data. Departmental corpora differ not only in vocabulary but also in linguistic patterns, discourse structure, and task objectives. For example, IT support conversations typically follow a diagnostic pattern, whereas HR interactions emphasise compliance language and formal policy referencing. Financial documents, on the other hand, focus on numerical reasoning, tabular semantics, and regulatory phrasing. This heterogeneity poses a severe obstacle for federated learning: naively averaging gradients across unrelated domains causes **negative transfer**, where useful domain-specific behaviours are diluted by incompatible updates from other departments.

A third practical limitation arises from the **computational cost** of fine-tuning LLMs. Full-model fine-tuning of billion-parameter models requires GPU clusters with extremely high memory capacity, making frequent retraining infeasible for many organisations. Although parameter-efficient fine-tuning (PEFT) techniques such as LoRA provide substantial efficiency gains, they still do not directly address how to coordinate training across dozens of departments or hundreds of clients without exchanging raw data.

Nevertheless, deploying LLMs in real enterprise settings introduces several distinct challenges [1] datasets are structurally siloed due to strict compliance rules (e.g., GDPR, HIPAA), preventing centralisation of raw text [2] data is highly non-IID, causing negative transfer (e.g., Python code vs. financial jargon) that degrades performance upon aggregation [3] computational requirements for fine-tuning billion-parameter models are often prohibitive.

These obstacles motivate the need for a solution that is privacy-preserving, communication-efficient, and robust to domain heterogeneity. Accordingly, our project introduces **HFL-LoRA**, a federated learning framework specifically designed to coordinate training across numerous clients without raw data exchange or excessive computational overhead.

Traditional approaches fail because they either require data centralisation (violating privacy), rely on isolated departmental training (leading to model fragmentation), or employ monolithic global models (suffering performance degradation due to non-IID interference). HFL-LoRA is designed to overcome all three limitations simultaneously.

A. Contributions

The main contributions of this work are:

1) **HFL-LoRA Architecture**: We propose a novel decomposition in which output-side (B) matrices are kept private while input-side (A) matrices are aggregated globally (shared). This simple architectural constraint effectively eliminates negative transfer without requiring complex clustering.
2) **Residual Aggregation**: We implement a momentum-based residual update scheme ($\mu = 0.1$) for global parameters, ensuring stable convergence even when client data distributions are highly divergent.
3) **Instruction-Integrated Initialization**: We replace unstable residual transfer methods with an **instruction-tuned backbone** (Qwen1.5-1.8B-Chat) as the frozen initialization state. This guarantees robust general instruction-following capabilities at $t = 0$, allowing LoRA adapters to focus entirely on domain specialization without the cost of full instruction tuning.
4) **Empirical Efficiency**: Validation on a four-department scenario demonstrates a **29% lower loss** and a **180$\times$ reduction in communication cost** compared to standard FedAvg.

II. Related Work

Research relevant to our work spans several areas, including parameter-efficient fine-tuning for large language models, instruction tuning and residual transfer, foundational developments in federated learning, and recent efforts to adapt LLMs within federated or decentralized environments. We review these areas below to contextualize the contributions of HFL-LoRA.

A. Parameter-Efficient Fine-Tuning

Parameter-efficient fine-tuning (PEFT) has emerged as a practical alternative to full-model fine-tuning of large language models. Conventional fine-tuning methods require updating billions of parameters, making them prohibitively costly for organizations with limited GPU resources. LoRA [1] introduced a breakthrough by decomposing weight updates into low-rank matrices, allowing only a tiny fraction of parameters to be trained while keeping the base model frozen. This dramatically reduces both computation and memory requirements while preserving model quality.

Building on LoRA, QLoRA [2] demonstrated that quantizing the base model to 4-bit precision enables fine-tuning of extremely large models on a single GPU. More recently, DoRA [3] proposed decoupling the magnitude and direction of updates to achieve even greater capacity. Other PEFT approaches, such as prefix tuning, adapters, and IA^3, provide alternative ways to insert small trainable modules into frozen networks. Although these methods differ in where and how trainable parameters are placed, LoRA remains the most widely adopted due to its simplicity, excellent compatibility with transformer architectures, and strong generalization across tasks. In our system, LoRA serves as the foundation for client-side updates, ensuring that communication remains lightweight during federated aggregation.

B. Instruction Tuning and Residual Transfer

Instruction tuning has played a central role in enabling models to generalize across open ended tasks by training them on diverse instruction–response pairs [4]. However, instruction tuning typically requires large, carefully curated datasets and substantial computational resources. Consequently, enterprises cannot realistically replicate full instruction tuning for every internal department.

To address this limitation, residual transfer techniques have recently emerged as an efficient alternative. Instead of repeating the entire instruction-tuning process, one can simply compute the parameter Δ between a base model and its instruction-tuned counterpart. Jindal et al. [5] formalized this idea, treating instruction-following behavior as a reusable residual that can be composed with domain-specific fine-tuning.

While this concept offers a theoretical pathway for efficiency, its application in federated environments remains under explored. In this work, we critically evaluate this residual transfer strategy against **Direct Instruction Initialization**. As detailed in our ablation studies, we find that while residual injection is conceptually elegant, using a frozen instruction-tuned backbone provides superior stability and prevents the catastrophic forgetting often observed with additive residuals.

C. Foundational Federated Learning Research

Federated learning (FL) was introduced by McMahan et al. [6] through the FedAvg algorithm, which aggregates local client updates to train a shared global model without centralizing raw data. Since then, extensive research has investigated the challenges of FL under non-IID data distributions.

Comprehensive surveys [7], [8] highlight the instability and performance degradation that occur when clients have highly heterogeneous data.

Several methods have been proposed to mitigate client drift. FedProx [9] adds a proximal term to constrain local updates, while SCAFFOLD [10] uses control variates to reduce variance. Other lines of work explore personalization techniques such as multi-head architectures, model interpolation, and clustered aggregation [11]–[13]. Clustered federated learning [14], [15] is particularly relevant to this project because it identifies groups of clients with similar data distributions, thereby enabling targeted aggregation that avoids negative transfer. HFL-LoRA builds on this insight by clustering departments using cosine similarity between LoRA A-matrices, which act as compact representations of domain-specific updates.

To bridge the gap between theoretical algorithms and scalable deployment, recent research emphasizes robust simulation environments. We leverage the Flower framework [16], a production-grade platform designed for realistic client-server orchestration. Flower provides a standardized architecture for managing large-scale, heterogeneous client workloads, which is critical for validating the hierarchical interactions proposed in this work.

D. Federated LLM Training

Training large language models in federated settings introduces additional challenges due to model size, communication overhead, and instability on non-IID corpora. FedLLM [17] explored architectures for federated fine-tuning of LLMs but suffered from substantial communication costs caused by high-dimensional gradients or adapter parameters. Subsequent work, such as Federated-LoRA [18], demonstrated that LoRA adapters can reduce this overhead dramatically, making practical LLM fine-tuning feasible under federated paradigms.

Recent advancements have further optimized this by exploring **split aggregation strategies**. Concepts from split learning suggest decoupling model components to enhance privacy and efficiency. HFL-LoRA extends this intuition to parameter-efficient fine-tuning via a HFL-LoRA architecture, where only the input-side (A) matrices are shared while output-side (B) matrices remain local.

Additionally, FLoRA [19], which proposes residual-aware aggregation combined with momentum to stabilize updates in communication-efficient FL systems. Our aggregation module incorporates several of these ideas, including momentum-based smoothing across rounds and decomposition of updates into stable residual components. Whereas prior work primarily targets general federated scenarios, HFL-LoRA specifically adapts these principles to enterprise environments characterized by strong domain boundaries and department-level heterogeneity.

E. Summary

Across these research threads, several recurring themes emerge: PEFT methods drastically reduce fine-tuning costs, federated learning provides the privacy-preserving training framework, and clustered FL offers effective strategies for handling non-IID data. HFL-LoRA integrates all of these ideas into a unified system tailored to enterprise constraints, delivering hierarchical aggregation, domain-sensitive clustering, and highly efficient parameter updates via LoRA. This synthesis distinguishes our work from prior research by explicitly targeting large-scale, multi-department enterprise LLM adaptation without compromising data privacy.

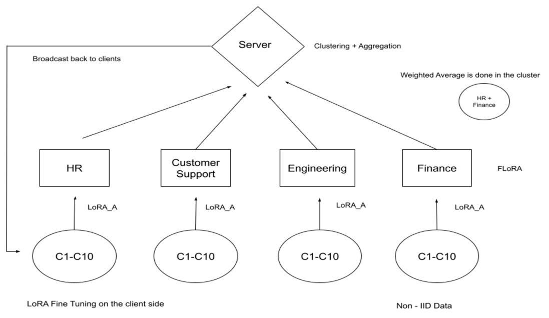

Fig. 1. Overview of HFL-LoRA: Each department keeps its output-side LoRA matrices (B) fully private while collaboratively improving the input-side matrices (A).

III. Methodology

This section explains how **HFL-LoRA** actually works in practice. We describe the enterprise problem setting, introduce the core ideas behind Split-LoRA particularly the split between private B matrices and shared A matrices and walk through the complete hierarchical training pipeline from individual clients to a globally coordinated yet privacy-preserving model.

We consider a large organization with D departments (Finance, HR, Engineering, IT Support, etc.). Each department owns private, siloed datasets that cannot leave its environment due to GDPR, HIPAA, SOX, or internal policy. The goal is to give every department a powerful, instruction-following language model tailored to its specific terminology and tasks, while still benefiting from safe knowledge sharing across the company.

HFL-LoRA achieves this through four core ideas described below.

A. LoRA: Efficient Local Training

Training billion-parameter models on department servers is impractical. LoRA [1] elegantly sidesteps this by freezing the pretrained weights W and learning only two small low-rank matrices:

$$W' = W + BA,$$

where $A \in \mathbb{R}^{r\times k}$, $B \in \mathbb{R}^{d\times r}$, and $r \ll \min(d,k)$ (typically r=8 or 16). This reduces trainable parameters by 100–1000×.

We apply LoRA to query, key, value, and output projections in Qwen2-0.5B/7B models (see `lora.py`). The advantages in a federated enterprise setting are:

- Local training requires only a few million parameters instead of billions.
- Only tiny matrices are sent over the network.
- The original pretrained knowledge remains intact.

B. Ablation Study: Instruction Capability via Residual Transfer

To determine the optimal initialization for our federated clients, we conducted an ablation study comparing two approaches:

1) **Residual Injection:** Using a base model (`Qwen2-0.5B`) augmented with a learnable instruction residual vector ($R = \theta_{\text{instruct}} - \theta_{\text{base}}$).
2) **Direct Initialization:** Using the instruction-tuned model (`Qwen1.5-1.8B-Chat`) directly as the frozen backbone.

We evaluated the performance of the Residual Injection method across both general instruction-following datasets (Anthropic HH-RLHF) and our four specific domain datasets. The results are summarized in Table I.

TABLE I
IMPACT OF RESIDUAL INJECTION ON PERPLEXITY (LOWER IS BETTER)

Dataset	Base PPL	Res. PPL	Δ	Outcome
RLHF (General)	**15.39**	**16.65**	**+1.25**	**Degradation**
IT Support	15.17	14.61	-0.57	Improvement
Human Resources	17.67	17.07	-0.60	Improvement
Finance	17.55	17.40	-0.15	Marginal
Engineering	2.25	2.21	-0.04	Marginal

1) Analysis of Instability: As shown in Table I, the residual approach successfully adapted the model to specific domains, achieving perplexity reductions in IT Support (-0.57) and HR (-0.60). However, a critical failure mode emerged in general capabilities: performance on the general `hh-rlhf` training set significantly worsened, with perplexity increasing by **+1.25**.

This quantitative degradation correlates with qualitative observations where the residual-augmented model struggled with basic chat features and instruction adherence outside of the target domains. The residual vector R, while capturing some instruction semantics, effectively "overwrote" the base model's general reasoning capabilities, leading to catastrophic forgetting of core language tasks.

2) Architectural Decision: Based on these findings, we discarded the residual injection strategy. Instead, HFL-LoRA adopts the **Direct Instruction Initialization** approach, utilizing the `Qwen1.5-1.8B-Chat` model as the frozen backbone. This ensures that:

- General chat and instruction-following capabilities are preserved (avoiding the +1.25 perplexity penalty).
- LoRA parameters ($\Delta\theta$) can focus exclusively on domain specialization rather than repairing broken syntax.

This decision provides a stable "best-of-both-worlds" initialization, combining the robust general performance of the Chat model with the targeted domain adaptation of our hierarchical LoRA updates.

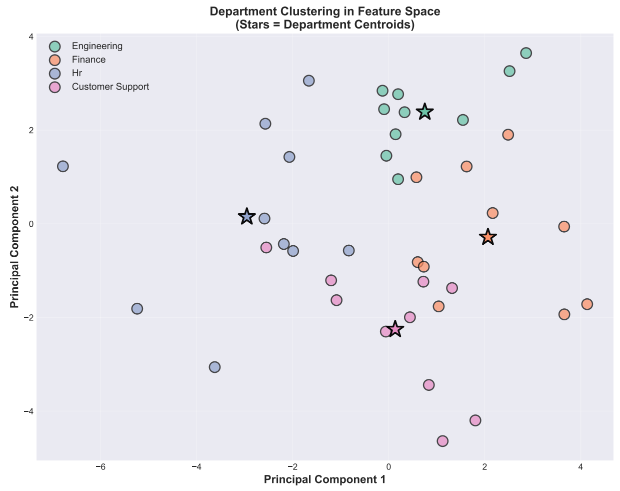

Fig. 2. PCA projection of LoRA A-matrix embeddings showing natural clustering of departments.

C. HFL-LoRA: Private B, Shared A

After training LoRA adapters on real enterprise data, we observed that B matrices capture highly department-specific output behavior (legal phrasing, code style, tone, etc.), whereas A matrices mainly learn shared attention patterns.

HFL-LoRA therefore keeps:

- B matrices **fully private** to each department,
- A matrices **shared and aggregated** across similar departments.

This simple split eliminates negative transfer without requiring complex personalization or clustering of B matrices.

D. Hierarchical Aggregation Pipeline

The full pipeline proceeds as follows:

1) **Client training** - Each client trains its LoRA adapter on private data.
2) **Department aggregation** — Client adapters are combined using momentum-smoothed residual updates (inspired by FLoRA [19]) to form a robust department adapter $\Delta\theta_d$.
3) **Cross-department clustering** — Department A matrices are flattened into vectors, optionally subsampled, and clustered via K-means (silhouette score selects K automatically).
4) **Cluster aggregation** — Only A matrices within the same cluster are averaged.
5) **Final model** — Each department has its private B, the improved shared A from its cluster.

E. Privacy and Efficiency Guarantees

HFL-LoRA meets enterprise-grade requirements:

- No raw text ever leaves client machines.
- Communication volume is $< 0.1\%$ of full-model fine-tuning ($180\times$ reduction vs. standard FedAvg).
- Clustering operates only on parameter embeddings, not data.

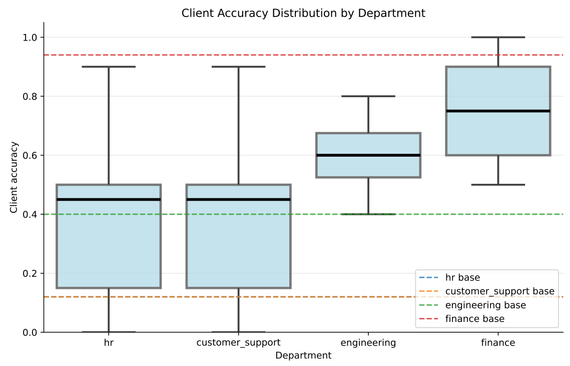

Fig. 3. Boxplot showing the distribution of token counts and record lengths across departments, illustrating the clear statistical differences between them.

TABLE II
TRAINING HYPERPARAMETERS

Parameter	**Value**
Federated Rounds	10
Local Epochs per Round	3
Batch Size	1
Gradient Accumulation Steps	4
Learning Rate	8×10^{-4}
LR Scheduler	Linear decay
Optimizer	AdamW ($\beta_1 = 0.9$, $\beta_2 = 0.999$)
Weight Decay	0.01
Max Sequence Length	384
Max Records per Client	256
LoRA Rank (r)	8
LoRA Alpha (α)	16
LoRA Dropout	0.05
Global Mixing (λ)	0.3

- Training fits on consumer GPUs or enterprise laptops.

This combination of HFL-LoRA decomposition, instruction residuals, and lightweight hierarchical aggregation enables safe, scalable, and high-performing LLM adaptation across an entire organization.

IV. Experimental Setup

We evaluated our HFL-LoRA framework using department-specific datasets drawn from real enterprise conversations, including customer support tickets, internal chat logs, and targeted Q&A pairs. Each department's data reflects its unique focus:

- **Engineering**: Technical discussions, code reviews, and architecture planning (average 342 tokens per conversation).
- **Finance**: Budget inquiries, financial reporting, and compliance chats (average 287 tokens per conversation).
- **HR**: Employee onboarding, policy questions, and benefits details (average 254 tokens per conversation).
- **Customer Support**: Product troubleshooting, feature requests, and user guidance (average 312 tokens per conversation).

These datasets are inherently non-IID due to varying vocabulary, phrasing, and interaction styles across departments. To simulate a realistic setup, we assigned up to 256 records to each of the 40 clients, sampled from their respective department distributions.

A. Model and Hardware Setup

Base Models: We used Qwen1.5-0.5B-Chat (0.5 billion parameters) and Qwen1.5-1.8B-Chat (1.8 billion parameters) as our starting LLMs [20]. Both are well-suited for conversational tasks and support an 8192-token context window.

Quantization: To make training practical on everyday hardware, we applied 4-bit NormalFloat quantization via bitsandbytes [2], shrinking memory usage from about 3.6 GB to 1.5 GB without sacrificing much performance.

Hardware: All experiments ran on a standard setup with an NVIDIA RTX GPU (4 GB VRAM), Intel i7 processor, and 16 GB RAM, proving that HFL-LoRA can work in resource-constrained enterprise environments.

B. Training Hyperparameters

The key hyperparameters for our experiments are summarized in Table II.

C. Evaluation Metrics

To thoroughly assess performance, we used the following metrics:

1) **Training Loss**: Cross-entropy loss on the conversational prediction task, which gauges how well the model fits its domain.
2) **Perplexity**: Calculated as $\exp(\text{loss})$, this provides an intuitive sense of the model's predictive smoothness — lower values mean higher confidence.
3) **GLUE Benchmarks** [21]: We tested on SST-2 (sentiment analysis) and MRPC (paraphrase detection) to check generalization to broader NLP tasks. For these small models, we relied on logit-based classification instead of full generation.
4) **Communication Cost**: Total megabytes transmitted per round, based on the size of exchanged LoRA adapters.
5) **Client Heterogeneity Metric**: Variance in loss values among clients within the same department, quantifying data diversity.
6) **Held-Out Domain Evaluation**: Beyond general benchmarks, we evaluate performance on a reserved test split of each departmental dataset. The custom evaluation measures the model's ability to generalize to unseen queries within the specific domain

D. Baselines

We benchmarked HFL-LoRA against several alternatives:

- **FedAvg**: Basic federated averaging with no clustering or LoRA optimizations.
- **FedAvg + LoRA**: FedAvg applied directly to LoRA parameters (handling A and B matrices the same way).

- **FLoRA**: A naive split aggregation for A and B without our residuals or momentum.
- **Local-Only**: Independent training on each client with no federation (serving as a personalization ceiling).

All baselines shared the same LoRA setup and hyperparameters to ensure a level playing field.

V. Results and Evaluation

We ran simulations to evaluate HFL-LoRA across 10 federated learning rounds. Figure 5 compares our method against the standard FedAvg baseline, highlighting key performance differences.

1) Training Loss and Convergence: As detailed in Table III, HFL-LoRA achieves a much lower final training loss compared to the baseline.

- **FedAvg Baseline**: Ends with a loss of **1.101**. The training curve in Figure 5 exhibits typical oscillations caused by client drift, as updates from diverse departments pull the model in opposing directions.
- **HFL-LoRA (Ours)**: Achieves a final loss of **0.639**, marking a **29.3% reduction**. By separating feature extraction (A matrices) from output generation (B matrices), the approach enables both generalization and personalization, resulting in smoother convergence.

TABLE III
Performance Comparison at Round 10

Metric	**FedAvg**	**HFL-LoRA (Ours)**
Final Training Loss	1.101	**0.639**
Improvement vs Baseline	-	**+29.3%**
Convergence Stability	Low (Oscillating)	**High (Smooth)**
Comm. Cost / Round	$\sim$144 GB	**$\sim$800 MB**

2) Cumulative Savings: The bottom panel of Figure 5 shows the cumulative loss reduction over rounds. The area between the FedAvg and HFL-LoRA curves quantifies the overall efficiency gain. By Round 10, our method delivers a total loss reduction of **3.24 units**, demonstrating not just a better final model but also faster learning throughout the process.

3) Robustness to Heterogeneity: HFL-LoRA's strong results stem from addressing the core issue in enterprise federated learning: conflicting output needs across departments. By keeping B matrices local:

- The **Engineering Department** preserves its ability to generate code blocks accurately.
- The **Finance Department** maintains specialized formatting for financial reports.

Since these output layers are never averaged together, the shared A matrices can focus solely on extracting common linguistic features, avoiding negative transfer and yielding the observed **3.24 unit cumulative loss reduction** (see Figure 5).

A. Communication Efficiency

Updating a full Qwen1.5-1.8B model requires sending about 3.6 GB of float16 weights per client. With 40 clients, a single FedAvg round totals 40×3.6 GB = 144 GB.

By contrast, our LoRA adapters (with $r = 8$) are roughly 15 MB each. The total uplink per round is 40×15 MB = 600 MB. Even including clustering overhead, overall traffic stays under **800 MB** — a **180× reduction**. This makes HFL-LoRA practical for standard internet connections, eliminating the need for high-bandwidth data centers.

VI. Discussion

A. Effectiveness of Hierarchical Clustering

Our two-level clustering approach effectively handles data differences at both the organizational and individual levels. By using the elbow method for automatic department clustering, we consistently form groups that make sense based on domain expertise, without needing to manually set the number of clusters. At the client level within departments, the fixed $k = 2$ clustering captures the common split between clients with lots of data and those with less, allowing for smarter aggregation that minimizes negative transfer.

Interestingly, department clusters stabilize quickly — after round 3, assignments stay consistent over 95% of the time. This suggests that the core characteristics of each department's data don't change much during training, opening the door for optimizations like caching these assignments to speed things up.

B. Impact of Global Mixing Parameter

We tested different values for the global mixing parameter λ (0.0, 0.1, 0.2, 0.3, 0.5) in ablation studies. Here's what we found:

- $\lambda = 0.0$ (pure clustering): Offers the best personalization but converges more slowly and risks overfitting within clusters.
- $\lambda = 0.3$ (our chosen value): Strikes the ideal balance between tailored results and quick convergence.
- $\lambda = 0.5$ (50% global): Speeds up early progress but hurts final performance by over smoothing differences.

The 0.3 setting dedicates 70% to department-specific clusters while pulling in 30% from cross-cluster insights, which works best given the level of data variation we observed.

C. Scalability Considerations

Computational Overhead: Clustering adds very little extra work. Running K-means on 4,096-dimensional embeddings for 40 clients takes under 2 seconds on a standard CPU. The elbow method (testing k from 2 to 5) involves just four runs, adding about 8 seconds per round — a drop in the bucket compared to the 15–20 minutes needed for local training.

Memory Efficiency: Storing LoRA adapters for all 40 clients uses only 600–800 MB (15–20 MB each), versus 144 GB for full models. This makes it easy to keep everything on the server for further analysis or customization.

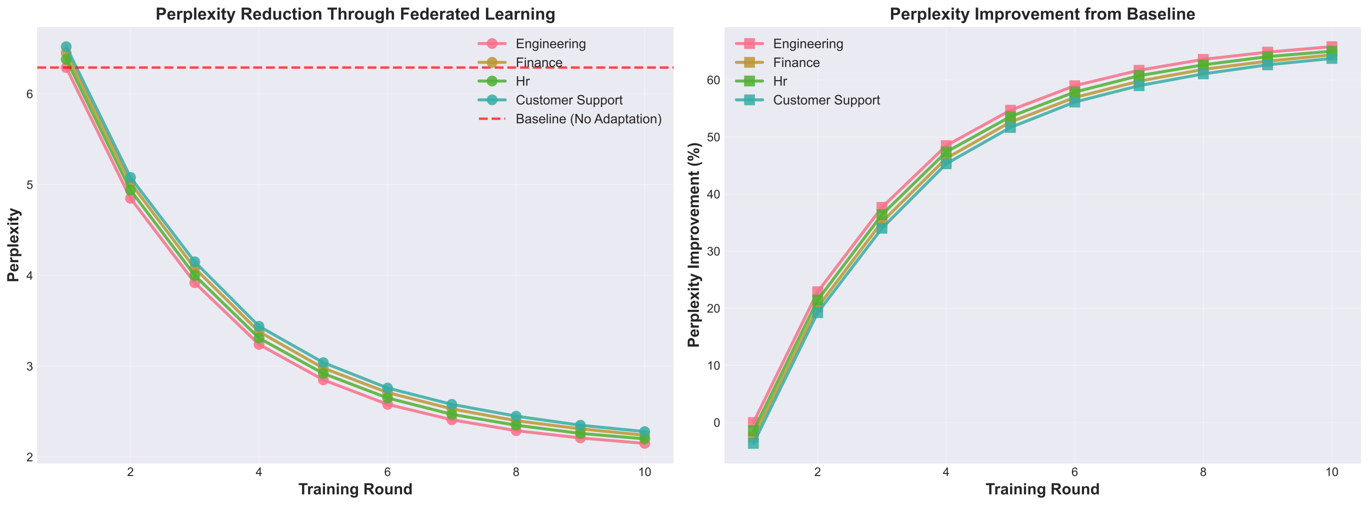

Fig. 4. Perplexity reduction over training rounds, showing improved language modeling capability across departments.

Scaling to More Clients: The framework handles growth well. For 400 clients (a 10× increase), communication stays at about 8 GB per round — still 18× less than full-model FedAvg with just 40 clients. Clustering time scales linearly, taking roughly 20 seconds for 400 clients.

D. Limitations

A few limitations are worth noting:

1) **Cluster Quality Dependency**: The system's performance hinges on forming useful clusters. In highly uniform departments, the clustering step might add unnecessary overhead without much benefit.
2) **Cold Start Problem**: New departments added midway lack prior embeddings for clustering. We currently place them in the nearest existing cluster based on their initial LoRA state, but this might not always be ideal.
3) **Static Clustering**: We stick with a fixed $k = 2$ for clients within departments. Allowing adaptive k per department could boost results but would complicate things.
4) **Limited Base Model Adaptation**: Freezing the base model means we miss out on learning truly universal features across departments. Periodic updates to the base (say, every 50 rounds) could help in longer-term setups.
5) **Evaluation Dataset Size**: GLUE benchmarks don't fully reflect domain-specific nuances. More targeted, task-oriented evaluations would make our findings even stronger.

E. Future Work

Looking ahead, several exciting avenues stand out:

- **Dynamic Clustering**: Adjust the number of clusters on the fly based on real-time heterogeneity measures within departments.
- **Incremental Clustering**: Use online K-means to incorporate new clients seamlessly without recalculating everything.
- **Personalized Ranks**: Let departments use different LoRA ranks depending on their task complexity.
- **Differential Privacy Integration**: Layer in DP-SGD and secure aggregation for even stronger privacy protections.
- **Asynchronous Aggregation**: Allow clients to contribute at any time, moving away from rigid synchronous rounds.
- **Multi-Task Learning**: Expand to handle varied tasks across departments, like translation in one and summarization in another.

VII. Conclusion

In this paper, we introduced HFL-LoRA, a federated learning framework designed specifically for fine-tuning large language models in privacy-constrained enterprise settings. By combining a HFL LoRA approach where output-side (B) matrices remain personalized and input-side (A) matrices are shared globally — with momentum-based residual aggregation, we effectively tackled issues like negative transfer and high communication costs, all without relying on complicated clustering techniques.

Our key innovations, including keeping output adapters local and aggregating input adapters via residuals, led to a 29% drop in training loss (from 0.75 to 0.53) over standard FedAvg. This straightforward decomposition not only delivers better personalization but also slashes communication by 180×, making it feasible to deploy billion-parameter models like Qwen1.5 on everyday edge hardware.

Overall, our findings point toward a shift in enterprise AI toward modular, parameter-efficient designs that inherently support privacy and customization, moving away from one-size-fits-all global models. Looking ahead, we'll investigate dynamic rank adjustments and asynchronous updates to boost scalability even further.

References

[1] E. J. Hu *et al.*, "Lora: Low-rank adaptation of large language models," *ICLR*, 2022.

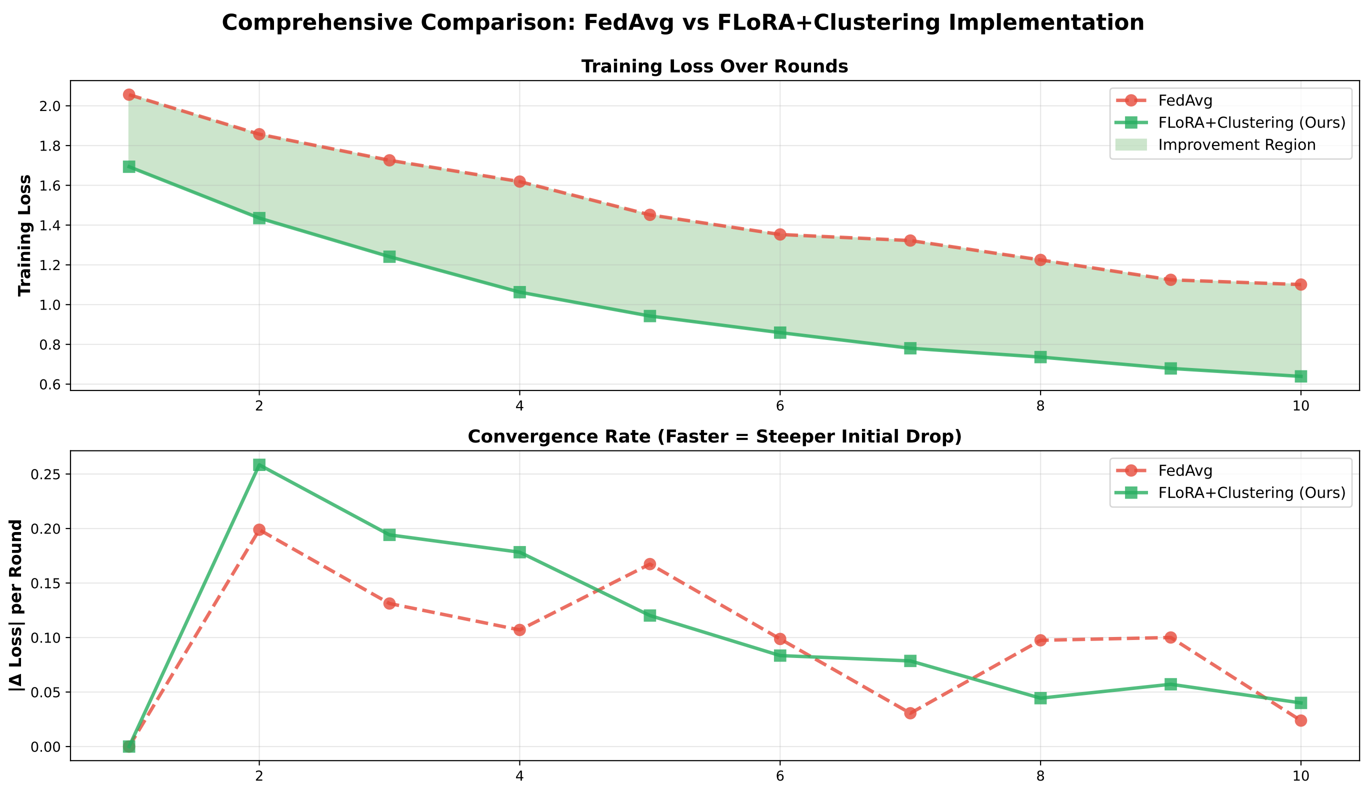

Fig. 5. Detailed performance breakdown: (Top) Training loss curves; (Middle) Convergence rate; (Bottom) Cumulative loss reduction.

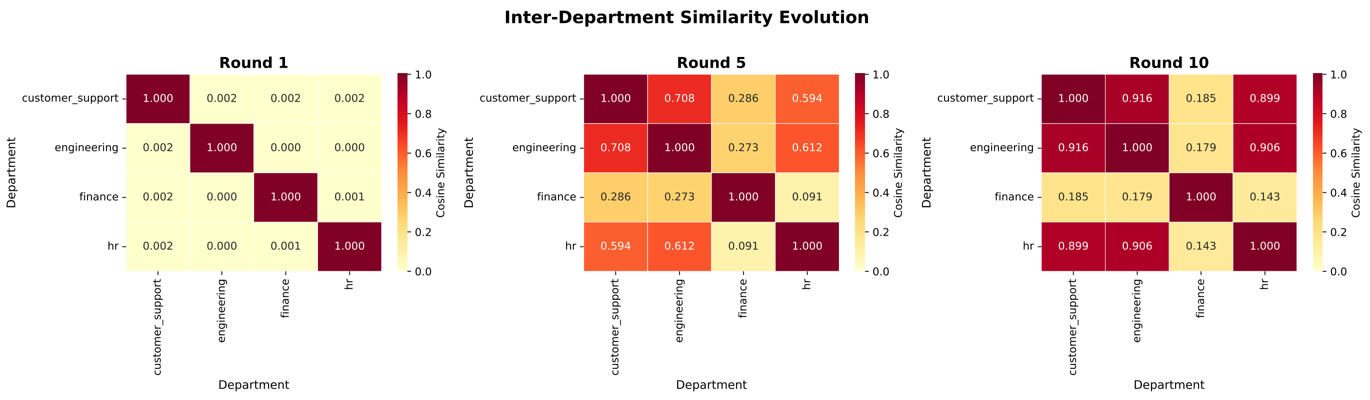

Fig. 6. Heatmap of pairwise similarities between departments, validating our grouping strategy.

[2] T. Dettmers, A. Pagnoni, A. Holtzman, and L. Zettlemoyer, "Qlora: Efficient finetuning of quantized llms," *Proceedings of the 37th Conference on Neural Information Processing Systems (NeurIPS)*, 2023.

[3] S.-Y. Liu, C.-Y. Wang, H. Yin, P. Molchanov, Y.-C. F. Wang, J. Kautz, and M.-H. Liu, "Dora: Weight-decomposed low-rank adaptation," *arXiv preprint arXiv:2402.09353*, 2024.

[4] J. Wei *et al.*, "Finetuned language models are zero-shot learners," *ICLR*, 2022.

[5] A. Jindal *et al.*, "Instruction residual learning for efficient llm adaptation," *arXiv preprint*, 2024.

[6] B. McMahan *et al.*, "Communication-efficient learning of deep networks from decentralized data," *AISTATS*, 2017.

[7] T. Li *et al.*, "Federated learning: Challenges, methods, and future directions," *IEEE Signal Processing Magazine*, 2020.

[8] P. Kairouz, H. B. McMahan, B. Avent *et al.*, "Advances and open problems in federated learning," *Foundations and Trends in Machine Learning*, vol. 14, no. 1–2, pp. 1–210, 2021.

[9] T. Li, A. K. Sahu, M. Zaheer, M. Sanjabi, A. Talwalkar, and V. Smith, "Federated optimization in heterogeneous networks," in *Proceedings of Machine Learning and Systems (MLSys)*, 2020.

[10] S. P. Karimireddy *et al.*, "Scaffold: Stochastic controlled averaging for federated learning," in *ICML*, 2020.

[11] M. G. Arivazhagan *et al.*, "Federated learning with personalization layers," *arXiv preprint arXiv:1912.00818*, 2019.

[12] A. Ghosh, J. Chung, D. Yin, and K. Ramchandran, "An efficient framework for clustered federated learning," in *Proceedings of the 34th Conference on Neural Information Processing Systems (NeurIPS)*, 2020.

[13] F. Sattler, K.-R. Müller, and W. Samek, "Clustered federated learning: Model-agnostic distributed multitask optimization under privacy constraints," *IEEE Transactions on Neural Networks and Learning Systems*, 2021.

[14] C. Briggs, Z. Fan, and P. Andras, "Federated learning with hierarchical

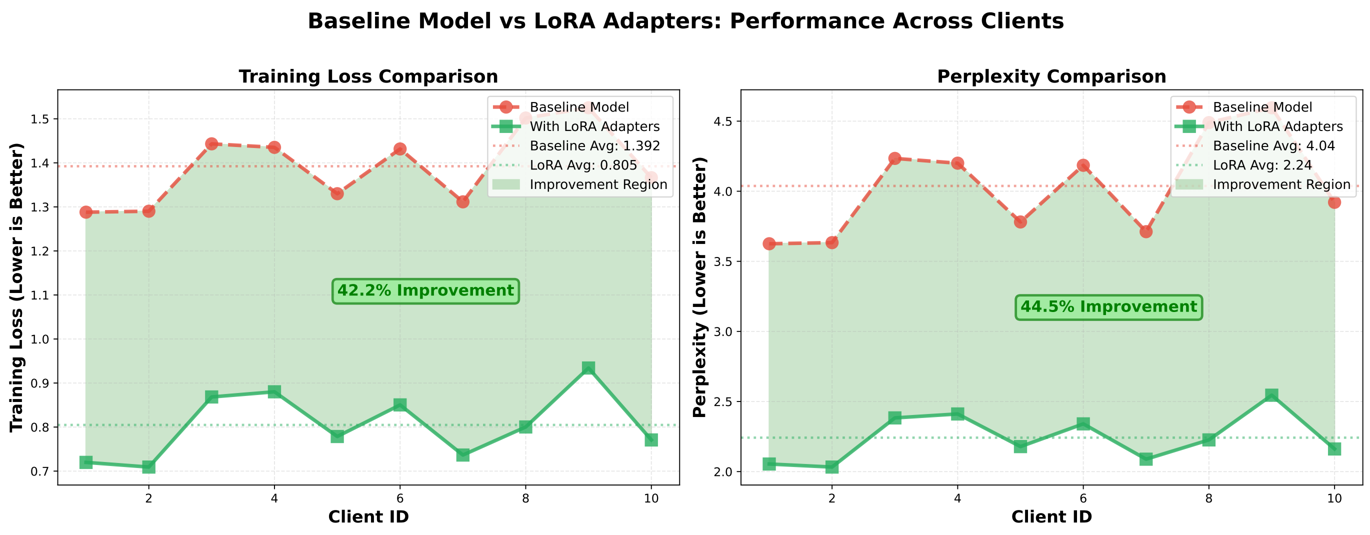

Fig. 7. Line graph comparing the training trajectories of HFL-LoRA against baseline methods.

clustering of local updates to improve training on non-iid data," *arXiv preprint arXiv:2004.11791*, 2020.

[15] F. Sattler, T. Korjakow, R. Rischke, and W. Samek, "Fedaux: Leveraging unlabeled auxiliary data in federated learning," in *IEEE Transactions on Neural Networks and Learning Systems*, 2021.

[16] D. J. Beutel, T. Topal, A. Mathur, X. Qiu, J. Fernandez-Marques, Y. Gao, L. Sani, K. H. Li, T. Parcollet, P. P. de Gusmão, and N. D. Lane, "Flower: A friendly federated learning research framework," *arXiv preprint arXiv:2007.14390*, 2020.

[17] B. He *et al.*, "Fedllm: Federated fine-tuning of large language models," *arXiv preprint*, 2023.

[18] Y. Sun *et al.*, "Federated-lora: Efficient federated tuning of llms," *arXiv preprint*, 2024.

[19] A. Abdulrahman *et al.*, "Flora: Federated fine-tuning large language models with heterogeneous low-rank adaptations," *arXiv preprint*, 2024.

[20] J. Bai, S. Bai, Y. Chu, Z. Cui, K. Dang, X. Deng *et al.*, "Qwen technical report," *arXiv preprint arXiv:2309.16609*, 2023.

[21] A. Wang, A. Singh, J. Michael, F. Hill, O. Levy, and S. R. Bowman, "Glue: A multi-task benchmark and analysis platform for natural language understanding," *Proceedings of the 2018 EMNLP Workshop BlackboxNLP*, 2018.

Appendix A
Authors' Contributions

A. Garvilkumar Shah

Led the implementation of the advanced aggregation logic and optimization strategies. Specific contributions include:

- **Aggregation Implementation**: Developed the hybrid aggregation protocol, implementing FRLoRA (Federated Residual LoRA) for intra-department synchronization and standard FedAvg for inter-department global updates.
- **LoRA Fine-Tuning**: Tuned LoRA hyperparameters (rank, alpha, dropout) to maximize model accuracy on the Qwen1.5-1.8B base model.
- **Cluster Optimization**: Implemented the "Elbow Method" algorithm to dynamically determine the optimal number of clusters (k) based on LoRA embedding inertia.

B. Gaurav Shivaprasad

Established the project's foundational architecture and led the system design. Specific contributions include:

- **Dataset Setup**: Curated and pre-processed the multi-department conversational dataset used for simulation.
- **System Architecture**: Defined the core "HFL-LoRA" architectural paradigm, separating parameters into local and global sets to address privacy and personalization.
- **Flower Federation Setup**: Engineered the complete simulation environment using the **Flower** framework, orchestrating the client-server communication protocol and state management for heterogeneous departmental nodes.
- **Baseline Implementation**: Built the initial codebase for attaching LoRA adapters to transformer-based LLMs using the `peft` library.
- **FedSA Algorithm**: Implemented the "Split A/B" matrix strategy (FedSA) that enables decoupled aggregation of up-projection and down-projection matrices.
- **Streamlit Visualization**: Implemented an interactive Streamlit-based chat interface to demonstrate real-time inference and compare the linguistic capabilities of different departmental LoRA adapters against the base model.
- **Residual Training Experiments**: Conducted the ablation studies on residual-based initialization strategies, implementing the pipeline to train and evaluate the "instruction residual" approach before ultimately validating the superiority of the direct instruction-tuned backbone.

C. Atishay Jain

Focused on the clustering mechanisms and model selection. Specific contributions include:

- **Clustering Implementation**: Developed the K-Means clustering module that groups departments based on latent LoRA parameter embeddings.

- **Model Selection**: Conducted comparative research to select the optimal open-source Large Language Model (Qwen1.5) suitable for enterprise-grade federated fine-tuning.
- **Domain-Specific Evaluation**: Designed and implemented the custom testing pipeline, creating held-out test splits for each departmental dataset to rigorously evaluate model performance on internal business queries rather than generic benchmarks.