FlowerTune: A Cross-Domain Benchmark for Federated Fine-Tuning of Large Language Models

Yan Gao[1,2,*] **Massimo Roberto Scamarcia**[3], **Javier Fernandez-Marques**[1,2], **Mohammad Naseri**[1], **Chong Shen Ng**[1], **Dimitris Stripelis**[1], **Zexi Li**[2,4], **Tao Shen**[4], **Jiamu Bai**[5], **Daoyuan Chen**[6], **Zikai Zhang**[7], **Rui Hu**[7], **InSeo Song**[8], **Lee KangYoon**[8], **Hong Jia**[9], **Ting Dang**[10], **Junyan Wang**[11], **Zheyuan Liu**[11], **Daniel Janes Beutel**[1], **Lingjuan Lyu**[12], **Nicholas D. Lane**[1,2]

[1]Flower Labs, [2]University of Cambridge, [3]ethicalabs.ai, [4]Zhejiang University, [5]Penn State University, [6]Alibaba Group, [7]University of Nevada, Reno, [8]Gachon University, [9]The University of Auckland, [10]The University of Melbourne, [11]The University of Adelaide, [12]Sony AI

Abstract

Large Language Models (LLMs) have achieved state-of-the-art results across diverse domains, yet their development remains reliant on vast amounts of publicly available data, raising concerns about data scarcity and the lack of access to domain-specific, sensitive information. Federated Learning (FL) presents a compelling framework to address these challenges by enabling decentralized fine-tuning on pre-trained LLMs without sharing raw data. However, the compatibility and performance of pre-trained LLMs in FL settings remain largely under explored. We introduce the *FlowerTune LLM Leaderboard*, a *first-of-its-kind* benchmarking suite designed to evaluate federated fine-tuning of LLMs across four diverse domains: general NLP, finance, medical, and coding. Each domain includes federated instruction-tuning datasets and domain-specific evaluation metrics. Our results, obtained through a collaborative, open-source and community-driven approach, provide the first comprehensive comparison across 26 pre-trained LLMs with different aggregation and fine-tuning strategies under federated settings, offering actionable insights into model performance, resource constraints, and domain adaptation. This work lays the foundation for developing privacy-preserving, domain-specialized LLMs for real-world applications.

1 Introduction

Large language models (LLMs) [1, 2, 7, 24, 58, 66] have exhibited remarkable performance across a broad spectrum of machine learning tasks and domains, including general natural language processing (NLP), medical question answering [53, 55], financial sentiment analysis [67], and code generation [30, 31]. These capabilities are typically attained through supervised fine-tuning applied to large-scale pre-trained models [21, 26, 30].

Despite their success, the current paradigm of LLM development heavily depends on large volumes of publicly available data [1, 24, 58]. While it is widely acknowledged that increased data volume typically enhances model performance, recent studies have raised concerns that the supply of high-quality public data may be exhausted within a few years [56, 59]. Moreover, there is an increasing demand for specialized LLMs that can integrate domain-specific knowledge not readily accessible in publicly available web-based corpora—particularly in sensitive domains such as healthcare and finance. However, accessing data stored within relevant institutions and organizations raises substan-

[*]Corresponding author: yan@flower.ai

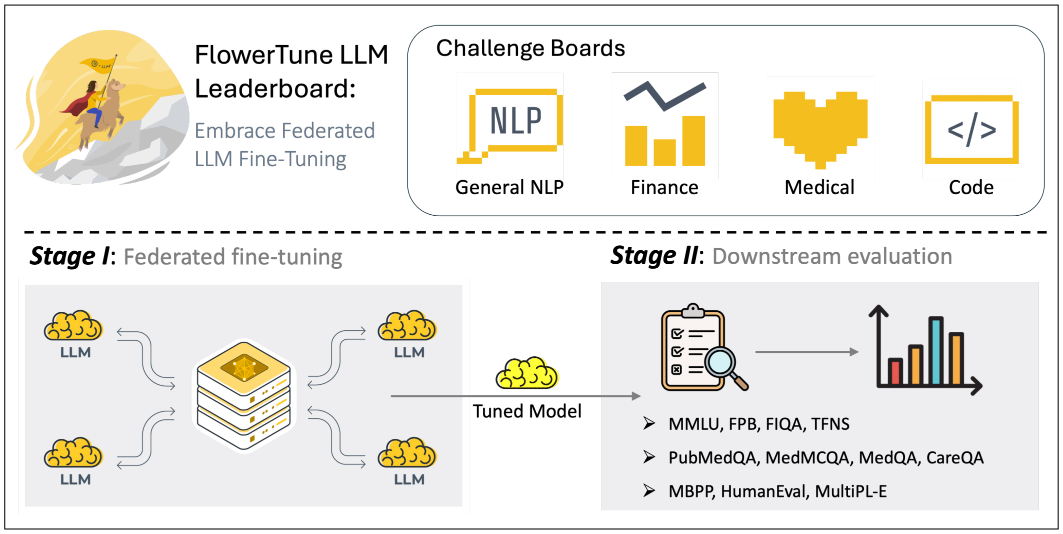
FlowerTune LLM Leaderboard:
Embrace Federated LLM Fine-Tuning
Challenge Boards
NLP
General NLP
Finance
Medical
</>
Code
Stage I: Federated fine-tuning
Stage II: Downstream evaluation
LLM
LLM
LLM
LLM
Tuned Model
MMLU, FPB, FIQA, TFNS
PubMedQA, MedMCQA, MedQA, CareQA
MBPP, HumanEval, MultiPL-E

Figure 1: **Overview of the FlowerTune LLM Leaderboard.** This leaderboard provides four challenges covering: general NLP, finance, medical, and coding. After selecting a challenge, participants can initiate federated fine-tuning using a provided template tailored to the specific scenario. The template is model-agnostic and supports various pre-trained base models, fine-tuning strategies, and aggregation algorithms, enabling flexible adaptation. Upon completion of training, the resulting global LLM is evaluated using domain-specific metrics, with scores reported to reflect the performance and quality of the tuned model.

tial privacy and security concerns, and is further complicated by challenges related to data storage and transfer, and communication infrastructure.

LLM fine-tuning via federated learning (FL) offers a promising solution to address these challenges [36, 69, 70]. FL enables collaborative model optimization by facilitating access to a wider range of sensitive datasets without the need to share raw data. This approach holds significant potential for the development of more fairness-preserving [43, 54] and domain-adapted LLMs. Recent studies [36, 69, 70] have begun to explore FL algorithms and fine-tuning strategies on selected models within this context. With rapid advancements and increasing accessibility of LLM pre-training, a broad array of pre-trained models are now available for downstream fine-tuning. However, their suitability for deployment in FL environments remains largely unexplored. Several critical questions persist: How effectively do these models perform under federated settings using existing aggregation and fine-tuning strategies? Are they compatible with the prevalent resource constraints in FL scenarios, such as communication overhead and memory limitations? And can FL offer a practical pathway for training more domain-specialized LLMs?

To systematically address these open challenges and deepen understanding, we introduce the FlowerTune LLM Leaderboard [2], a benchmarking initiative built on the Flower Platform [13], designed to evaluate the performance of various pre-trained LLMs under various federated fine-tuning scenarios across multiple domains. The leaderboard focuses on four high-impact application areas involving sensitive or private data: general NLP, finance, medical, and coding. Each domain includes a dedicated federated dataset for instruction tuning, along with domain-specific evaluation metrics. By establishing baseline results across a range of models and domains in an open-source, community-driven framework, the FlowerTune LLM Leaderboard has attracted an increasing number of submissions from both academic and industry contributors. Building upon these contributions, we conduct a first-of-its-kind comprehensive set of fine-tuning experiments to benchmark 26 base models under a unified FL setting. This enables valuable insights into the feasibility and effectiveness of deploying federated fine-tuning for LLMs, and would further assist in accelerating the development of more inclusive, privacy-preserving, and domain-specialized language models for real-world applications.

[2]`https://flower.ai/benchmarks/llm-leaderboard`

Table 1: Statistical summary of the datasets and their splits for federated fine-tuning in FlowerTune LLM Leaderboard. The average length of instruction and output is measured in characters.

Challenges	Fine-tuning dataset	Total # samples	# clients	len(instruction)	len(output)
General NLP	alpaca-gpt4 [50]	52 K	20	60	677
Finance	fingpt-sentiment-train [67]	76.8 K	50	112	9
Medical	medical-flashcards [25]	34 K	20	92	349
Code	code-alpaca-20k [16]	20 K	10	74	197

Table 2: Summary of the evaluation datasets used in FlowerTune LLM Leaderboard. MQA represents multiple-choice question answering.

Challenges	Evaluation datasets	Task types	Evaluation metrics
General NLP	MMLU [27] (STEM, Humanities, Social Sciences)	MQA	Accuracy
Finance	FPB [45], FIQA [44], TFNS [73]	Classification	Accuracy
Medical	PubMedQA [33], MedMCQA [48], MedQA [32], CareQA [8]	MQA	Accuracy
Code	MBPP [9], HumanEval [17], MultiPL-E (JS, C++) [15]	Code generation	Pass@1

2 FlowerTune LLM Leaderboard

2.1 Overview

The FlowerTune LLM Leaderboard is a benchmarking initiative that provides tools and standardized baselines for federated fine-tuning and evaluation of LLMs. It aims to facilitate reproducible research and community-driven exploration in this emerging, yet increasingly studied area. For reproducibility, this leaderboard includes two end-to-end pipelines for LLM federated fine-tuning and evaluation, respectively, involving four sensitive data domains: general NLP, finance, medical, and coding. More domains are also planned for future inclusion to further broaden its scope (Figure 1).

2.2 Federated fine-tuning

To emulate data distributions that could be expected across institutions such as medical, financial, and educational organizations, we carefully select four domain-specific public datasets to support the respective challenges. Specifically, we use alpaca-gpt4 [3] [50] for the general NLP domain, fingpt-sentiment-train [4] [67] for finance, medical-flashcards [5] [25] for the medical domain, and code-alpaca-20k [6] [16] for coding. Each dataset includes domain-specific instruction prompts paired with corresponding answers, designed to train an LLM to function as an assistant within its respective domain. We employ Flower Datasets [37] to partition each dataset into multiple shards of approximately equal size, simulating the data distribution across institutions in FL environments. A detailed statistical summary of the datasets and their corresponding splits is presented in Table 1, while representative example samples from each dataset are provided in the Appendix C. Additionally, a quantitative analysis of the client-level data distribution is provided in Appendix A.1. Although the number of samples is relatively balanced across clients within each domain, semantic similarity metrics reveal notable differences among clients, particularly in the general NLP, medical, and code domains.

After selecting a challenge, participants can implement their own federated methods and LLM models using the provided template. This template is model-agnostic and readily adaptable to various fine-tuning techniques and aggregation algorithms. Furthermore, the FlowerTune framework includes built-in tools for measuring key FL system metrics, such as communication overhead and memory usage. A detailed description of the participation process is provided in Appendix D.

2.3 Downstream evaluation

For each domain, we provide a corresponding evaluation pipeline that includes domain-specific evaluation datasets (Table 2). Specifically, in the general NLP challenge, the MMLU dataset [27]

[3] `https://huggingface.co/datasets/flwrlabs/alpaca-gpt4`
[4] `https://huggingface.co/datasets/flwrlabs/fingpt-sentiment-train`
[5] `https://huggingface.co/datasets/flwrlabs/medical-meadow-medical-flashcards`
[6] `https://huggingface.co/datasets/flwrlabs/code-alpaca-20k`

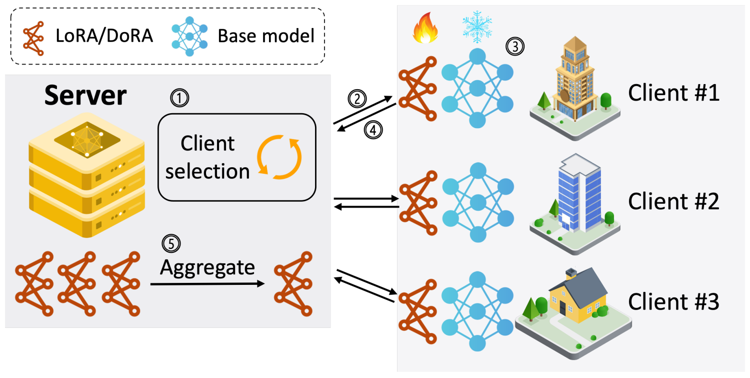

Figure 2: **Illustration of the federated LLM fine-tuning process.** (1) Initialization of LoRA/DoRA adapters and client selection on the server; (2) transmission of adapter parameters to the selected clients; (3) local adapter fine-tuning with the base model frozen; (4) transmission of updated adapter parameters back to the server; (5) aggregation of adapter parameters. This process is repeated in each subsequent FL round.

is used for evaluation. MMLU consists of multiple-choice questions covering a broad range of subjects across STEM, humanities, and social sciences. The evaluation metric is accuracy, reflecting the performance of the tested LLM. For the finance challenge, evaluation is conducted using a sentiment classification pipeline on financial reports and social media posts. Three datasets are utilized: FPB [45], FIQA [44], and TFNS [73]. Model performance is assessed using accuracy as the evaluation metric. In the medical domain, we leverage four datasets—PubMedQA [33], MedMCQA [48], MedQA [32], and CareQA [8]—to evaluate the performance of the tuned LLM as a medical assistant. Each dataset consists of multiple-choice questions covering various medical topics. Evaluation is based on accuracy, reflecting the model's ability to select the correct answers. For the coding challenge, we select MBPP [9], HumanEval [17], and MultiPL-E (JavaScript, C++) [15] as evaluation datasets. These datasets are specifically designed to assess code generation capabilities, and the pass@1 score is used as the evaluation metric, measuring the proportion of correctly generated programs on the first attempt.

The federated fine-tuned LLMs are evaluated in a zero-shot fashion, a challenging yet realistic scenario in which the model does not see any data samples from the evaluation domain during the fine-tuning phase. All models are evaluated under identical configurations to ensure a fair and consistent comparison.

3 Experimental settings

The FlowerTune LLM Leaderboard has attracted significant interest from the community, as evidenced by a growing number of submissions. These submissions, together with the benchmarking code [7] and pipelines provided by FlowerTune, were instructive in allowing us to systematically conduct a *first-of-its-kind* comprehensive set of fine-tuning experiments to benchmark various base models within a unified FL setting. In addition to investigating different base models, we also explore the impact of diverse aggregation algorithms and fine-tuning strategies. Detailed descriptions of the experimental settings are provided in the following sections.

3.1 Base model selection

Unlike traditional data centers, institutions participating in FL environments often lack access to large-scale computational resources, such as high-performance GPUs. As a result, hosting extremely large models is generally impractical. Taking these constraints into account—and aiming to enable effective federated fine-tuning—we focus on LLMs with fewer than 14 billion parameters.

In total, we select 26 base models with parameter sizes ranging from 135 million to 14 billion. These models are categorized into two groups: (1) pre-trained base models without further fine-tuning, and

[7] `https://github.com/yan-gao-GY/flowertune-benchmark`

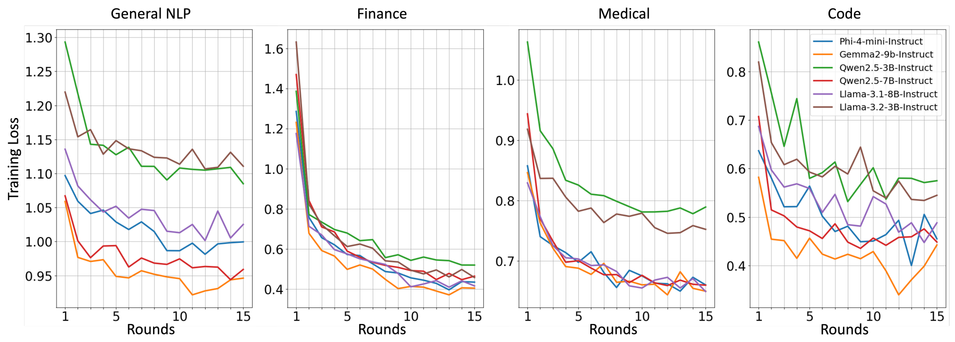

Figure 3: **Average training loss over FL rounds with 6 selected models on four challenges.** The training loss exhibits a consistent downward trend across all tasks, with larger fluctuations observed in the coding challenge.

(2) models that have been fine-tuned after pre-training by their respective providers, denoted with the suffix "-Instruct". Details of the base models are presented in Appendix A.2.

3.2 Federated learning settings

In the benchmarking experiments of different base models, a unified FL configuration is adopted across all challenges. Specifically, 20% of clients are selected per FL round, resulting in 4/20, 10/50, 4/20, and 2/10 clients for general NLP, finance, medical, and coding challenges, respectively (Table 1). Fine-tuning is conducted over 15 rounds for all tasks. During training, two system-level metrics are monitored: communication cost and VRAM usage. Communication cost is measured as the total bi-directional data transmitted between the server and selected clients across all FL rounds, while VRAM usage is recorded per client on a single GPU. The entire federated fine-tuning process is illustrated in Figure 2.

In the investigation of FL strategies, we select four popular algorithms: (1) **FedAvg** [46]: Federated Averaging (FedAvg) aggregates local model updates by computing their weighted average based on client data size. This method serves as a foundational method for many FL algorithms. (2) **FedProx** [40]: Federated Proximal (FedProx) extends FedAvg by introducing a proximal term to the local objective, which helps stabilize training when client data is non-IID by discouraging local models from drifting too far from the global model. (3) **FedAvgM** [28]: FedAvg with Momentum (FedAvgM) enhances FedAvg by incorporating server-side momentum during aggregation, which accelerates convergence and improves performance in heterogeneous environments. (4) **FlexLoRA** [10]: FlexLoRA is a communication-efficient FL method that integrates Low-Rank Adaptation (LoRA) into local training and employs adaptive aggregation to fuse updates, which improves scalability and personalization.

3.3 Local fine-tuning hyper-parameters

During the local fine-tuning phase, we adopt parameter-efficient fine-tuning (PEFT) techniques [21, 26], which substantially reduce computational requirements and accelerate the training process. Specifically, we investigate the following two popular PEFT methods in our federated LLM fine-tuning pipeline: (1) **LoRA** [29] introduces a PEFT method by injecting trainable low-rank matrices into pre-trained model weights, enabling a significant reduction in trainable parameters and computational cost without additional inference latency. (2) **DoRA** [41] builds upon LoRA by decomposing weights into magnitude and direction components, applying LoRA specifically to the directional part to better emulate the learning capacity of full fine-tuning while preserving efficiency. Both methods are applied with quantization, resulting in a QLoRA-style training approach [20].

In the local tuning process, the base model remains frozen, while only the LoRA/DoRA adapters are trained and communicated between clients and the central server. We employ DoRA while benchmarking various base models and additionally compare its performance against LoRA on selected models. Additionally, we integrate FlashAttention-2 [19] to further enhance training speed and computational efficiency. All experiments are conducted on NVIDIA A100 SXM4 (80 GB) GPUs. The complete set of training hyper-parameters is summarized in the Appendix A.3.

Table 3: **Comparison of different base models (Instruct version) federated fine-tuned on the General NLP challenge.** The accuracy values (%) are reported on different downstream tasks. Comm. represents the total communication costs over FL fine-tuning, while Mem. stands for the VRAM costs per client. The highest performance and lowest system overhead are indicated in **bold**, while the second-best results are underlined.

Models	STEM (%)	Social Sciences (%)	Humanities (%)	Average (%)	Comm. (GB)	Mem. (GB)
Mistral-7B-Instruct-v0.3	14.91	32.79	33.75	27.15	12.31	25.28
Gemma2-9B-Instruct	40.34	71.43	47.31	53.03	15.52	59.49
Phi-4-mini-Instruct (3.8B)	**56.90**	75.76	57.79	63.48	21.03	48.09
Llama3.2-1B-Instruct	0.54	1.07	1.49	1.03	3.31	27.53
Llama3.2-3B-Instruct	14.91	32.21	27.65	24.92	7.05	32.68
Llama3.1-8B-Instruct	52.87	69.65	50.01	57.51	8.10	45.85
Qwen2-0.5B-Instruct	20.20	27.46	25.21	24.29	2.61	26.67
Qwen2.5-1.5B-Instruct	41.45	63.67	49.93	51.68	5.48	31.97
Qwen2.5-3B-Instruct	34.22	49.82	34.20	39.41	8.90	34.74
Qwen2.5-7B-Instruct	55.47	**78.23**	**59.81**	**64.50**	11.99	49.31
SmolLM2-135M-Instruct	19.79	20.38	21.98	20.71	**1.42**	**9.06**
SmolLM2-360M-Instruct	14.53	17.48	19.51	17.17	2.52	11.28
SmolLM2-1.7B-Instruct	5.17	9.33	9.73	8.08	4.97	15.17

Table 4: **Comparison of different base models (Instruct version) federated fine-tuned on the Finance challenge.** The accuracy values (%) are reported on different downstream tasks. Comm. represents the total communication costs over FL fine-tuning, while Mem. stands for the VRAM costs per client. The highest performance and lowest system overhead are indicated in **bold**, while the second-best results are underlined.

Models	FPB (%)	FIQA (%)	TFNS (%)	Average (%)	Comm. (GB)	Mem. (GB)
Mistral-7B-Instruct-v0.3	76.98	75.99	55.82	69.60	30.77	17.26
Gemma2-9B-Instruct	**84.65**	**83.22**	**84.76**	**84.21**	38.80	39.39
Phi-4-mini-Instruct (3.8B)	44.80	52.30	27.85	41.65	52.56	23.16
Llama3.2-1B-Instruct	55.94	57.89	51.84	55.23	8.28	11.04
Llama3.2-3B-Instruct	45.46	64.80	38.07	49.44	17.63	16.21
Llama3.1-8B-Instruct	40.18	64.47	32.79	45.81	30.77	27.55
Qwen2-0.5B-Instruct	37.38	47.37	37.81	40.85	6.54	10.78
Qwen2.5-1.5B-Instruct	32.01	59.87	22.07	37.98	13.69	14.58
Qwen2.5-3B-Instruct	29.29	62.50	21.61	37.80	22.26	17.21
Qwen2.5-7B-Instruct	29.70	60.53	21.36	37.20	29.97	28.50
SmolLM2-135M-Instruct	30.53	59.21	24.16	37.97	**3.55**	**4.89**
SmolLM2-360M-Instruct	27.89	58.55	24.41	36.95	6.30	5.55
SmolLM2-1.7B-Instruct	52.97	33.88	59.97	48.94	12.42	9.51

4 Experimental results

This section presents the results of three core experimental investigations: (1) benchmarking the performance of various base models under a unified FL setup, (2) analyzing system performance during federated fine-tuning, and (3) evaluating the impact of different aggregation and fine-tuning techniques. It is important to note that this benchmark aims to provide a standardized testbed of different base models under FL settings, serving as a reference for future studies. All experiments are conducted using a unified configuration without task-specific or model-specific hyper-parameter tuning. More results with particular hyper-parameter tuning are available on the FlowerTune LLM Leaderboard [2].

4.1 Evaluation performance analysis

We examine both instruct and non-instruct base models within federated environments. Results for instruct models are presented in Tables 3–6, while those for non-instruct models are shown in Tables 12–15. Overall, instruct models demonstrate more stable and consistently higher performance across all four domains compared to their non-instruct counterparts. A detailed analysis of the instruct models is provided in the following sections, with the analysis of non-instruct models is included in the Appendix B.1.

Table 3 shows the evaluated performance on the **general NLP** challenge. First, the Qwen2.5-7B-Instruct model achieves the highest average accuracy across the three evaluation datasets, outperforming larger models such as Gemma2-9B-Instruct and LlaMA-3.1-8B-Instruct. Second, Phi-4-Mini-Instruct obtains the second-highest performance (63.48%) despite having a relatively small parameter count of 3.8 billion. Interestingly, Qwen2.5-1.5B-Instruct and SmolLM2-135M-Instruct achieve sur-

Table 5: **Comparison of different base models (Instruct version) federated fine-tuned on the Medical challenge.** The accuracy values (%) are reported on different downstream tasks. Comm. represents the total communication costs over FL fine-tuning, while Mem. stands for the VRAM costs per client. The highest performance and lowest system overhead are indicated in **bold**, while the second-best results are underlined.

Models	PubMedQA (%)	MedMCQA (%)	MedQA (%)	CareQA (%)	Average (%)	Comm. (GB)	Mem. (GB)
Mistral-7B-Instruct-v0.3	63.80	18.55	39.83	28.39	37.64	12.31	20.71
Gemma2-9B-Instruct	67.60	**54.46**	**57.34**	**69.58**	**62.25**	15.52	53.40
Phi-4-mini-Instruct (3.8B)	13.60	42.12	47.60	56.08	39.85	21.03	34.38
Llama3.2-1B-Instruct	19.20	7.32	24.12	4.38	13.75	3.31	20.39
Llama3.2-3B-Instruct	30.20	1.65	6.99	4.39	10.81	7.05	23.96
Llama3.1-8B-Instruct	**68.80**	17.83	32.36	30.55	37.39	12.31	36.24
Qwen2-0.5B-Instruct	56.80	8.70	13.75	8.15	21.85	2.61	18.21
Qwen2.5-1.5B-Instruct	48.80	41.33	42.58	50.51	45.80	5.48	22.29
Qwen2.5-3B-Instruct	65.20	39.30	34.25	45.37	46.03	8.90	24.83
Qwen2.5-7B-Instruct	60.80	49.77	56.87	64.06	57.88	11.99	39.05
SmolLM2-135M-Instruct	54.00	20.01	9.35	16.79	25.04	**1.42**	**7.06**
SmolLM2-360M-Instruct	40.20	7.75	3.46	10.96	15.59	2.52	7.79
SmolLM2-1.7B-Instruct	39.20	6.60	11.78	11.44	17.26	4.97	11.82

Table 6: **Comparison of different base models (Instruct version) federated fine-tuned on the Coding challenge.** The pass@1 scores (%) are reported on different downstream tasks. Comm. represents the total communication costs over FL fine-tuning, while Mem. stands for the VRAM costs per client. The highest performance and lowest system overhead are indicated in **bold**, while the second-best results are underlined.

Models	MBPP (%)	HumanEval (%)	MultiPL-E (JS) (%)	MultiPL-E (C++) (%)	Average (%)	Comm. (GB)	Mem. (GB)
Mistral-7B-Instruct-v0.3	40.40	37.80	43.48	33.54	38.81	6.15	22.65
Gemma2-9B-Instruct	**53.40**	58.54	54.04	47.20	**53.29**	7.76	58.05
Phi-4-mini-Instruct (3.8B)	47.80	**60.98**	54.04	38.51	50.33	10.51	38.00
Llama3.2-1B-Instruct	34.80	35.98	22.98	18.63	28.10	1.66	21.29
Llama3.2-3B-Instruct	43.80	54.88	46.58	34.16	44.86	3.53	26.79
Llama3.1-8B-Instruct	50.40	59.76	**58.39**	44.10	53.16	6.15	37.41
Qwen2-0.5B-Instruct	13.20	14.63	13.04	14.91	13.95	1.31	19.74
Qwen2.5-1.5B-Instruct	22.80	6.10	8.07	24.84	15.45	2.74	24.09
Qwen2.5-3B-Instruct	40.20	22.56	6.83	37.89	26.87	4.45	27.41
Qwen2.5-7B-Instruct	48.40	18.90	9.94	**50.31**	31.89	5.99	40.24
SmolLM2-135M-Instruct	8.40	7.93	4.97	4.97	6.57	**0.71**	**8.84**
SmolLM2-360M-Instruct	25.00	18.29	13.04	10.56	16.72	1.26	9.09
SmolLM2-1.7B-Instruct	34.60	28.05	21.12	24.84	27.15	2.48	12.42

prisingly competitive performance, even surpassing some of their larger counterparts. This suggests their potential suitability for more resource-constrained FL scenarios in general NLP fine-tuning.

In the **finance** challenge (Table 4), Gemma2-9B-Instruct achieves the best average performance (84.21%), largely attributable to its greater capacity stemming from the largest number of parameters among the evaluated models. Mid-sized models (e.g., Llama3.2-3B, Qwen2.5-3B) obtain performance comparable to their larger counterparts (e.g., Llama3.1-8B, Qwen2.5-7B), a trend that is also reflected in their training loss trajectories (Figure 3). Smaller models such as LlaMA-3.2-1B-Instruct, Qwen2-0.5B-Instruct, and SmolLM2-135M-Instruct perform better in this challenge. This could be partially due to the relative simplicity of the classification task, which allows even lightweight models to learn effectively.

In the **medical** challenge (Table 5), Gemma2-9B-Instruct again achieves the highest average accuracy of 62.25% across the four evaluation datasets. Note that while most federated fine-tuned models perform well on specific datasets, they lack generalization across all tasks within the medical domain. Overall, the Qwen model family consistently outperforms the LlaMA models of comparable size in this medical QA evaluation, suggesting that LlaMA models may require more careful fine-tuning under FL settings. Notably, SmolLM2-135M-Instruct obtains an acceptable average accuracy of 25.04%, outperforming some larger models and demonstrating its potential in resource-constrained federated environments.

Table 6 presents the evaluation results for the **coding** challenge. Gemma2-9B-Instruct (53.29%) achieved the highest performance followed closely by LlaMA-3.1-8B-Instruct (53.16%), whereas smaller models with fewer than 3 billion parameters generally underperform. This suggests that code generation tasks may inherently require the capacity provided by larger models to achieve strong performance. Phi-4-Mini-Instruct stands out by achieving an average accuracy of 50.33%, demonstrating strong performance relative to its modest size of 3.8 billion parameters.

In summary, larger models generally exhibit superior performance across all challenges. However, the nature of the task—such as question answering, classification, or code generation—should be

Table 7: Comparison of **aggregation methods** using the Qwen2.5-7B base model across four challenges. DoRA is used in the local fine-tuning. The highest performance is indicated in **bold**.

	General NLP				Medical				
Methods	STEM	Social Sciences	Humanities	Avg	PubMedQA	MedMCQA	MedQA	CareQA	Avg
FedAvg	**41.55**	52.62	34.20	42.79	33.40	**17.62**	18.85	**30.21**	**25.02**
FedProx	41.52	**52.78**	34.45	**42.92**	36.20	15.18	15.95	26.31	23.41
FedAvgM	39.80	49.56	**35.30**	41.56	36.80	15.61	14.77	24.98	23.04
FlexLoRA	34.82	46.05	30.31	37.06	**42.40**	13.08	**19.01**	16.42	22.73
	Finance				Code				
Methods	FPB	FIQA	TFNS	Avg	MBPP	HumanEval	MultiPL-E (JS)	MultiPL-E (C++)	Avg
FedAvg	**71.86**	52.30	74.20	66.12	56.60	23.17	52.17	49.07	45.25
FedProx	71.62	54.61	74.08	66.77	55.40	21.95	52.80	48.45	44.65
FedAvgM	73.76	**56.91**	**74.25**	**68.31**	**59.60**	**25.61**	**53.42**	48.45	**46.77**
FlexLoRA	70.96	52.96	72.28	65.40	57.40	15.85	49.69	**50.93**	43.47

Table 8: Comparison of two **fine-tuning techniques** using the Qwen2.5-7B base model across four challenges. FedAvg is used for aggregation. "SS" refers to Social Sciences. For the medical challenge, "PMQA", "MMCQA", "MQA" and "CQA" correspond to PubMedQA, MedMCQA, MedQA, and CareQA, respectively. For the coding challenge, "HE", "M-JS" and "M-C++" represent HumanEval, MultiPL-E (JS), and MultiPL-E (C++). Communication (Comm.) and VRAM (Mem.) costs are reported in gigabytes.

	General NLP						Medical						
Methods	STEM	SS	Humanities	Avg	Comm.	Mem.	PMQA	MMCQA	MQA	CQA	Avg	Comm.	Mem.
LoRA	38.47	48.33	33.75	40.18	11.90	46.07	37.80	16.69	18.93	28.13	25.39	11.90	34.88
DoRA	41.55	52.62	34.20	42.79	11.99	49.23	33.40	17.62	18.85	30.21	25.02	11.99	39.53
	Finance						Code						
Methods	FPB	FIQA	TFNS	Avg	Comm.	Mem.	MBPP	HE	M-JS	M-C++	Avg	Comm.	Mem.
LoRA	73.43	55.92	74.66	68.01	29.74	27.91	56.00	29.27	55.90	50.31	47.87	5.95	38.88
DoRA	71.86	52.30	74.20	66.12	29.97	28.10	56.60	23.17	52.17	49.07	45.25	5.99	39.36

carefully considered when selecting models. For instance, smaller models perform adequately on the finance classification task. While further advancements will benefit from algorithms specifically tailored to federated LLM fine-tuning, this benchmark offers a solid foundation and practical reference for selecting appropriate base models based on task requirements in federated learning settings.

4.2 System performance analysis

Tables 3–6 also report system-level performance metrics for different base models during federated fine-tuning. Overall, both communication costs and VRAM usage remain modest, primarily due to the adoption of the DoRA strategy. Communication overhead is influenced by three main factors: the size of the DoRA/LoRA adapters, the number of clients selected per round, and the total number of federated training rounds. For example, the finance task incurs higher communication costs than other tasks, solely due to the selection of more clients per round under otherwise identical settings. Controlling communication overhead is crucial in FL environments, particularly in scenarios with limited network bandwidth (see Appendix B.2).

VRAM consumption is influenced by several factors, including the architecture and size of the base model, the adapter size, and the input sequence length determined by the training dataset. Memory usage is often regarded as a primary bottleneck in federated on-device training. Experimental results demonstrate that the memory footprint of all evaluated models using FlowerTune tools remains below 80 GB—sufficient to fit within a single modern A100/H100 GPU, or across two mid-tier GPUs. In particular, larger models (e.g., Gemma2-9B-Instruct) demonstrate superior performance but incur higher memory consumption. Conversely, smaller models—such as those in the SmolLM2 family—achieve acceptable performance on specific challenges while requiring approximately 10 GB or less of VRAM. This enables the training of these smaller models on edge devices, such as an NVIDIA Jetson board, MacBook, or Raspberry Pi with 16 GB of RAM, when considering memory consumption alone. However, computational speed represents another critical factor, which can only be accurately assessed on actual edge devices and is generally considerably slower than GPU-based training. These trade-offs between performance and resource efficiency should be carefully considered when deploying models in real-world applications (see Appendix B.2).

4.3 Aggregation and fine-tuning techniques analysis

We evaluate the performance of four FL strategies using the Qwen2.5-7B base model, as presented in Table 7. Overall, the strategies yield comparable results, with only marginal differences observed across the various challenges. Specifically, based on average scores, FedProx achieves the best performance in the general NLP task, FedAvg performs best in the medical task, and FedAvgM outperforms others in both the finance and coding tasks. However, when examining individual evaluation datasets with a specific domain, no single strategy consistently outperforms the others across all datasets. These findings highlight the need for future research to develop more specialized aggregation algorithms tailored to federated LLM fine-tuning.

Table 8 presents a comparison of LoRA and DoRA in federated fine-tuning using the Qwen2.5-7B base model across four challenges. The results indicate that LoRA achieves slightly better average performance than DoRA in the finance and coding tasks, while both methods yield nearly identical accuracy in the medical task. In contrast, DoRA performs marginally better in the general NLP task. In terms of communication and VRAM costs, both LoRA and DoRA demonstrate efficient resource usage within modest limits. Particularly, DoRA incurs slightly higher overhead in both dimensions, mainly due to the additional computation required for decomposing weights into magnitude and direction components.

In summary, off-the-shelf aggregation strategies and fine-tuning techniques yield relatively minor differences in performance. In contrast, the choice of base model has a substantially greater impact, with notable performance variations observed across different domains. These findings highlight the critical importance of base model selection for federated LLM fine-tuning.

5 Related work

5.1 LLM instruction tuning

Instruction tuning is a form of fine-tuning that aligns LLMs with human intent using instruction-response datasets. Unlike task-specific fine-tuning, it promotes generalization across diverse tasks by training models to follow natural language prompts [74]. InstructGPT demonstrated that human-written instructions improve model helpfulness and safety [47], while Self-Instruct reduced data collection costs by using LLMs to generate synthetic instructions [61]. This approach enabled large-scale datasets like Super-NaturalInstructions and inspired methods such as Evo-Instruct and Instruct-SkillMix [34, 60, 72]. Parameter-efficient fine-tuning (PEFT) methods, such as LoRA, address the high cost of fine-tuning by updating only small portions of model parameters [29]. This makes instruction tuning feasible in low-resource environments. Recent studies integrate LoRA into instruction workflows for improved scalability [12, 20], while advanced variants like ALoRA optimize performance by dynamically adjusting parameter allocation [42]. Beyond supervised tuning, reinforcement learning from human feedback (RLHF) aligns model behavior with human preferences, as seen in the training of ChatGPT [11]. While effective, RLHF faces challenges in reward modeling, feedback quality, and scalability [76, 39].

Recent work has explored efficient on-device fine-tuning of LLMs to address privacy and resource constraints. GSQ-Tuning [75] enables fully integer-based on-device LLM fine-tuning using group-shared exponents, while QEFT [38] improves efficiency by updating only sensitive weight columns in mixed precision. These methods extend PEFT methods like LoRA [29] and QLoRA [20] for resource-constrained settings, and are applicable to federated cross-device fine-tuning. While our study focuses on cross-silo fine-tuning, we acknowledge that recent advances in on-device fine-tuning offer promising directions that could further enhance privacy and efficiency in federated settings, and we leave their exploration to future work.

5.2 Federated LLM fine-tuning

Federated LLM fine-tuning facilitates privacy-preserving training across decentralized data sources. Several frameworks and benchmarks have been developed to support this paradigm. OpenFedLLM [70] introduces a general framework for instruction tuning and alignment of LLMs without sharing raw data. FederatedScope-LLM [36] provides a modular package supporting PEFT and automated benchmarking. FATE-LLM [22] focuses on industrial deployment, supporting both horizontal and vertical federated settings. FedLLM-Bench [69] addresses the benchmarking gap by offering realistic datasets, fine-tuning methods, and evaluation pipelines to support empirical studies. To enrich training data, FedIT-U2S [68] transforms unstructured client data into instruction-response pairs, while FedDCA [62] augments domain coverage to improve model generalization.

Despite these advancements, federated LLM fine-tuning faces challenges such as resource constraints, system heterogeneity, personalization, and communication efficiency. Parameter-efficient fine-tuning methods, particularly LoRA-based techniques, have become central to addressing these issues. FedBiOT [64] addresses limited resources by enabling clients to tune lightweight adapters instead of full models. FedALT [14] improves LoRA performance in federated contexts by stabilizing client updates via adaptive training. Further improvements in LoRA are seen in FFA-LoRA [57], which freezes select LoRA matrices to reduce communication, and in pFedLoRA [71], which supports model heterogeneity using shared small adapters. FLoRA [63] introduces aggregation strategies for heterogeneous LoRA modules, while FDLoRA [52] deploys dual adapters for balancing global and personalized knowledge. FedRand [49] enhances privacy via randomized LoRA parameter updates, and LoRA-A2 [35] adapts LoRA structure under extreme heterogeneity.

Beyond adapter tuning, several approaches focus on client-specific optimization. FedMKT [23] enhances generalization by transferring knowledge between small and large models. FedBis [65] integrates user preference alignment through binary selectors. These methods reflect growing attention to personalization and robustness in federated LLM fine-tuning. However, existing work has not systematically examined the impact of diverse base models on federated fine-tuning, particularly in terms of downstream performance and system-level behavior—an essential but underexplored aspect of this emerging field.

6 Conclusion

As the development of LLMs continues to advance, challenges surrounding data availability, privacy, domain specificity, and the practicalities of data storage and communication infrastructure are becoming increasingly pronounced. Federated LLM fine-tuning provides a potential solution through collaborative model optimization on sensitive and institutionally siloed data without compromising privacy. To address the lack of standardized evaluation in this emerging area, we introduced the *FlowerTune LLM Leaderboard*, an open-source and community-driven benchmarking platform designed to assess the performance of pre-trained LLMs under federated fine-tuning across four critical application domains: general NLP, finance, medical, and code. By providing domain-specific datasets, evaluation protocols, and baseline results, our initiative establishes a foundation for systematic, reproducible research on federated LLM fine-tuning.

Limitations and future work. This study focuses primarily on cross-silo federated settings; future work may explore a broader range of scenarios, including more resource-constrained environments and more heterogeneous data distributions. Additionally, the experiments were conducted using a unified configuration following a systematic protocol but without extensive hyper-parameter tuning for each individual model-dataset pair. Performance could likely be improved through more careful hyper-parameter optimization. Furthermore, the experimental results indicate that several classical federated aggregation algorithms (e.g., FedAvg, FedProx) exhibit only marginal performance differences in the context of LLM adapter tuning. This finding highlights the need for the community to develop aggregation algorithms specifically tailored to this parameter space (i.e., low-rank adapters) to more effectively integrate knowledge from diverse clients, representing a promising direction for future research. We envision this work as a catalyst for developing more inclusive, secure, and domain-adapted language models, and invite the broader research community to contribute to and extend this benchmarking effort.

Acknowledgments and Disclosure of Funding

We sincerely thank all the participants in the FlowerTune LLM Leaderboard from the community for their valuable contributions and feedback. We are especially grateful to the following teams and individuals: Abdelkareem Elkhateb, AI4EOSC Team, Alessandro Pinto, CAR@AIML, Daniel Hinjos García, Dongqi Cai, FL-finetune-JB-DC, Gachon Cognitive Computing Lab, Massimo R. Scamarcia, mHealth Lab, Nut Chukamphaeng, T-IoI@UNR, and ZJUDAI.

References

[1] M. Abdin, J. Aneja, H. Behl, S. Bubeck, R. Eldan, S. Gunasekar, M. Harrison, R. J. Hewett, M. Javaheripi, P. Kauffmann, et al. Phi-4 technical report. *arXiv preprint arXiv:2412.08905*, 2024.

[2] A. Abouelenin, A. Ashfaq, A. Atkinson, H. Awadalla, N. Bach, J. Bao, A. Benhaim, M. Cai, V. Chaudhary, C. Chen, et al. Phi-4-mini technical report: Compact yet powerful multimodal language models via mixture-of-LoRAs. *arXiv preprint arXiv:2503.01743*, 2025.

[3] M. AI. Mistral-Small-24B-Base-2501. `https://huggingface.co/mistralai/Mistral-Small-24B-Base-2501`, 2024.

[4] M. AI. Mistral-Small-24B-Instruct-2501. `https://huggingface.co/mistralai/Mistral-Small-24B-Instruct-2501`, 2024.

[5] M. AI. Mistral-7B-v0.3. `https://huggingface.co/mistralai/Mistral-7B-v0.3`, 2024.

[6] M. AI. Mistral-7B-Instruct-v0.3. `https://huggingface.co/mistralai/Mistral-7B-Instruct-v0.3`, 2024.

[7] L. B. Allal, A. Lozhkov, E. Bakouch, G. M. Blázquez, G. Penedo, L. Tunstall, A. Marafioti, H. Kydlíček, A. P. Lajarín, V. Srivastav, et al. SmolLM2: When Smol Goes Big–Data-Centric Training of a Small Language Model. *arXiv preprint arXiv:2502.02737*, 2025.

[8] A. Arias-Duart, P. A. Martin-Torres, D. Hinjos, P. Bernabeu-Perez, L. U. Ganzabal, M. G. Mallo, A. K. Gururajan, E. Lopez-Cuena, S. Alvarez-Napagao, and D. Garcia-Gasulla. Automatic evaluation of healthcare LLMs beyond question-answering. In L. Chiruzzo, A. Ritter, and L. Wang, editors, *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 2: Short Papers)*, pages 108–130, Albuquerque, New Mexico, Apr. 2025. Association for Computational Linguistics.

[9] J. Austin, A. Odena, M. Nye, M. Bosma, H. Michalewski, D. Dohan, E. Jiang, C. Cai, M. Terry, Q. Le, et al. Program synthesis with large language models. *arXiv preprint arXiv:2108.07732*, 2021.

[10] J. Bai, D. Chen, B. Qian, L. Yao, and Y. Li. Federated fine-tuning of large language models under heterogeneous language tasks and client resources. *arXiv e-prints*, pages arXiv–2402, 2024.

[11] Y. Bai, A. Jones, K. Ndousse, A. Askell, A. Chen, N. DasSarma, D. Drain, S. Fort, D. Ganguli, T. Henighan, et al. Training a helpful and harmless assistant with reinforcement learning from human feedback. *arXiv preprint arXiv:2204.05862*, 2022.

[12] C. C. S. Balne, S. Bhaduri, T. Roy, V. Jain, and A. Chadha. Parameter efficient fine tuning: A comprehensive analysis across applications. *arXiv preprint arXiv:2404.13506*, 2024.

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

[14] J. Bian, L. Wang, L. Zhang, and J. Xu. FedALT: Federated Fine-Tuning through Adaptive Local Training with Rest-of-the-World LoRA. *arXiv preprint arXiv:2503.11880*, 2025.

[15] F. Cassano, J. Gouwar, D. Nguyen, S. Nguyen, L. Phipps-Costin, D. Pinckney, M.-H. Yee, Y. Zi, C. J. Anderson, M. Q. Feldman, et al. MultiPL-E: a scalable and polyglot approach to benchmarking neural code generation. *IEEE Transactions on Software Engineering*, 49(7): 3675–3691, 2023.

[16] S. Chaudhary. Code Alpaca: An Instruction-following LLaMA model for code generation. `https://github.com/sahil280114/codealpaca`, 2023.

[17] M. Chen, J. Tworek, H. Jun, Q. Yuan, H. P. de Oliveira Pinto, J. Kaplan, H. Edwards, Y. Burda, N. Joseph, G. Brockman, A. Ray, R. Puri, G. Krueger, M. Petrov, H. Khlaaf, G. Sastry, P. Mishkin, B. Chan, S. Gray, N. Ryder, M. Pavlov, A. Power, L. Kaiser, M. Bavarian, C. Winter, P. Tillet, F. P. Such, D. Cummings, M. Plappert, F. Chantzis, E. Barnes, A. Herbert-Voss, W. H. Guss, A. Nichol, A. Paino, N. Tezak, J. Tang, I. Babuschkin, S. Balaji, S. Jain, W. Saunders, C. Hesse, A. N. Carr, J. Leike, J. Achiam, V. Misra, E. Morikawa, A. Radford, M. Knight, M. Brundage, M. Murati, K. Mayer, P. Welinder, B. McGrew, D. Amodei, S. McCandlish, I. Sutskever, and W. Zaremba. Evaluating large language models trained on code. *arXiv preprint arXiv:2107.03374*, 2021.

[18] Y. Chu, J. Xu, Q. Yang, H. Wei, X. Wei, Z. Guo, Y. Leng, Y. Lv, J. He, J. Lin, et al. Qwen2-audio technical report. *arXiv preprint arXiv:2407.10759*, 2024.

[19] T. Dao. Flashattention-2: Faster attention with better parallelism and work partitioning. *arXiv preprint arXiv:2307.08691*, 2023.

[20] T. Dettmers, A. Pagnoni, A. Holtzman, and L. Zettlemoyer. QLoRA: Efficient finetuning of quantized LLMs. *Advances in Neural Information Processing Systems*, 36:10088–10115, 2023.

[21] N. Ding, Y. Qin, G. Yang, F. Wei, Z. Yang, Y. Su, S. Hu, Y. Chen, C.-M. Chan, W. Chen, et al. Parameter-efficient fine-tuning of large-scale pre-trained language models. *Nature Machine Intelligence*, 5(3):220–235, 2023.

[22] T. Fan, Y. Kang, G. Ma, W. Chen, W. Wei, L. Fan, and Q. Yang. FATE-LLM: A industrial grade federated learning framework for large language models. *arXiv preprint arXiv:2310.10049*, 2023.

[23] T. Fan, G. Ma, Y. Kang, H. Gu, Y. Song, L. Fan, K. Chen, and Q. Yang. FedMKT: Federated Mutual Knowledge Transfer for Large and Small Language Models. In *Proceedings of the 31st International Conference on Computational Linguistics*, pages 243–255, 2025.

[24] A. Grattafiori, A. Dubey, A. Jauhri, A. Pandey, A. Kadian, A. Al-Dahle, A. Letman, A. Mathur, A. Schelten, A. Vaughan, et al. The Llama 3 Herd of Models. *arXiv preprint arXiv:2407.21783*, 2024.

[25] T. Han, L. C. Adams, J.-M. Papaioannou, P. Grundmann, T. Oberhauser, A. Löser, D. Truhn, and K. K. Bressem. MedAlpaca–An Open-Source Collection of Medical Conversational AI Models and Training Data. *arXiv preprint arXiv:2304.08247*, 2023.

[26] Z. Han, C. Gao, J. Liu, J. Zhang, and S. Q. Zhang. Parameter-efficient fine-tuning for large models: A comprehensive survey. *arXiv preprint arXiv:2403.14608*, 2024.

[27] D. Hendrycks, C. Burns, S. Basart, A. Zou, M. Mazeika, D. Song, and J. Steinhardt. Measuring massive multitask language understanding. *Proceedings of the International Conference on Learning Representations (ICLR)*, 2021.

[28] T.-M. H. Hsu, H. Qi, and M. Brown. Measuring the effects of non-identical data distribution for federated visual classification. *arXiv preprint arXiv:1909.06335*, 2019.

[29] E. J. Hu, Y. Shen, P. Wallis, Z. Allen-Zhu, Y. Li, S. Wang, L. Wang, W. Chen, et al. LoRA: Low-rank adaptation of large language models. *ICLR*, 1(2):3, 2022.

[30] D. Huang, J. M. Zhang, M. Luck, Q. Bu, Y. Qing, and H. Cui. Agentcoder: Multi-agent-based code generation with iterative testing and optimisation. *arXiv preprint arXiv:2312.13010*, 2023.

[31] M. A. Islam, M. E. Ali, and M. R. Parvez. Mapcoder: Multi-agent code generation for competitive problem solving. *arXiv preprint arXiv:2405.11403*, 2024.

[32] D. Jin, E. Pan, N. Oufattole, W.-H. Weng, H. Fang, and P. Szolovits. What disease does this patient have? A large-scale open domain question answering dataset from medical exams. *Applied Sciences*, 11(14):6421, 2021.

[33] Q. Jin, B. Dhingra, Z. Liu, W. Cohen, and X. Lu. PubMedQA: A dataset for biomedical research question answering. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2567–2577, 2019.

[34] S. Kaur, S. Park, A. Goyal, and S. Arora. Instruct-SkillMix: A Powerful Pipeline for LLM Instruction Tuning. In *The Thirteenth International Conference on Learning Representations*.

[35] J. Koo, M. Jang, and J. Ok. Towards robust and efficient federated low-rank adaptation with heterogeneous clients. *arXiv preprint arXiv:2410.22815*, 2024.

[36] W. Kuang, B. Qian, Z. Li, D. Chen, D. Gao, X. Pan, Y. Xie, Y. Li, B. Ding, and J. Zhou. FederatedScope-LLM: A comprehensive package for fine-tuning large language models in federated learning. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 5260–5271, 2024.

[37] F. Labs. Flower Datasets. `https://flower.ai/docs/datasets`, 2025.

[38] C. Lee, J.-g. Jin, Y. Cho, and E. Park. QEFT: Quantization for efficient fine-tuning of LLMs. *arXiv preprint arXiv:2410.08661*, 2024.

[39] H. Lee, S. Phatale, H. Mansoor, K. R. Lu, T. Mesnard, J. Ferret, C. Bishop, E. Hall, V. Carbune, and A. Rastogi. RLAIF: Scaling reinforcement learning from human feedback with AI feedback.

[40] T. Li, A. K. Sahu, M. Zaheer, M. Sanjabi, A. Talwalkar, and V. Smith. Federated optimization in heterogeneous networks. *Proceedings of Machine learning and systems*, 2:429–450, 2020.

[41] S.-Y. Liu, C.-Y. Wang, H. Yin, P. Molchanov, Y.-C. F. Wang, K.-T. Cheng, and M.-H. Chen. DoRA: Weight-decomposed low-rank adaptation. In *Forty-first International Conference on Machine Learning*, 2024.

[42] Z. Liu, J. Lyn, W. Zhu, and X. Tian. ALoRA: Allocating Low-Rank Adaptation for Fine-tuning Large Language Models. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 622–641, 2024.

[43] L. Lyu, X. Xu, Q. Wang, and H. Yu. Collaborative fairness in federated learning. *Federated Learning: Privacy and Incentive*, pages 189–204, 2020.

[44] M. Maia, S. Handschuh, A. Freitas, B. Davis, R. McDermott, M. Zarrouk, and A. Balahur. WWW'18 Open Challenge: Financial Opinion Mining and Question Answering. In *Companion proceedings of the the web conference 2018*, pages 1941–1942, 2018.

[45] P. Malo, A. Sinha, P. Korhonen, J. Wallenius, and P. Takala. Good debt or bad debt: Detecting semantic orientations in economic texts. *Journal of the Association for Information Science and Technology*, 65, 2014.

[46] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas. Communication-efficient learning of deep networks from decentralized data. In *Artificial intelligence and statistics*, pages 1273–1282. PMLR, 2017.

[47] L. Ouyang, J. Wu, X. Jiang, D. Almeida, C. Wainwright, P. Mishkin, C. Zhang, S. Agarwal, K. Slama, A. Ray, et al. Training language models to follow instructions with human feedback. *Advances in neural information processing systems*, 35:27730–27744, 2022.

[48] A. Pal, L. K. Umapathi, and M. Sankarasubbu. MedMCQA: A large-scale multi-subject multi-choice dataset for medical domain question answering. In G. Flores, G. H. Chen, T. Pollard, J. C. Ho, and T. Naumann, editors, *Proceedings of the Conference on Health, Inference, and Learning*, volume 174 of *Proceedings of Machine Learning Research*, pages 248–260. PMLR, 07–08 Apr 2022.

[49] S. Park, S. Lee, B. Kim, and S. J. Hwang. FedRand: Enhancing Privacy in Federated Learning with Randomized LoRA Subparameter Updates. *arXiv preprint arXiv:2503.07216*, 2025.

[50] B. Peng, C. Li, P. He, M. Galley, and J. Gao. Instruction Tuning with GPT-4. *arXiv preprint arXiv:2304.03277*, 2023.

[51] PT GoTo Gojek Tokopedia Tbk and AI Singapore. Gemma2 9B CPT Sahabat-AI v1 Instruct. `https://huggingface.co/GoToCompany/gemma2-9b-cpt-sahabatai-v1-instruct`, 2024.

[52] J. Qi, Z. Luan, S. Huang, C. Fung, H. Yang, and D. Qian. FDLoRA: Personalized Federated Learning of Large Language Model via Dual LoRA Tuning. *arXiv preprint arXiv:2406.07925*, 2024.

[53] K. Saab, T. Tu, W.-H. Weng, R. Tanno, D. Stutz, E. Wulczyn, F. Zhang, T. Strother, C. Park, E. Vedadi, et al. Capabilities of Gemini models in medicine. *arXiv preprint arXiv:2404.18416*, 2024.

[54] Y. Shi, H. Yu, and C. Leung. Towards fairness-aware federated learning. *IEEE Transactions on Neural Networks and Learning Systems*, 2023.

[55] K. Singhal, T. Tu, J. Gottweis, R. Sayres, E. Wulczyn, M. Amin, L. Hou, K. Clark, S. R. Pfohl, H. Cole-Lewis, et al. Toward expert-level medical question answering with large language models. *Nature Medicine*, pages 1–8, 2025.

[56] Spiceworks. Is LLM Training Data Running Out?, 2024. URL `https://www.spiceworks.com/tech/artificial-intelligence/articles/is-llm-training-data-running-out/`.

[57] Y. Sun, Z. Li, Y. Li, and B. Ding. Improving LoRA in privacy-preserving federated learning. In *The Twelfth International Conference on Learning Representations*.

[58] G. Team. Gemma. *Unpublished*, 2024. doi: 10.34740/KAGGLE/M/3301. URL `https://www.kaggle.com/m/3301`.

[59] P. Villalobos, A. Ho, J. Sevilla, T. Besiroglu, L. Heim, and M. Hobbhahn. Will we run out of data? Limits of LLM scaling based on human-generated data. *arXiv preprint arXiv:2211.04325*, 2022.

[60] Y. Wang, S. Mishra, P. Alipoormolabashi, Y. Kordi, A. Mirzaei, A. Naik, A. Ashok, A. S. Dhanasekaran, A. Arunkumar, D. Stap, et al. 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, 2022.

[61] Y. Wang, Y. Kordi, S. Mishra, A. Liu, N. A. Smith, D. Khashabi, and H. Hajishirzi. Self-Instruct: Aligning Language Models with Self-Generated Instructions. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 13484–13508, 2023.

[62] Z. Wang, Y. Du, X. Ma, Y. Jiang, Z. Qian, and S. Chen. Federated Instruction Tuning of LLMs with Domain Coverage Augmentation. *arXiv preprint arXiv:2409.20135*, 2024.

[63] Z. Wang, Z. Shen, Y. He, G. Sun, H. Wang, L. Lyu, and A. Li. FLoRA: Federated Fine-Tuning Large Language Models with Heterogeneous Low-Rank Adaptations. *arXiv preprint arXiv:2409.05976*, 2024.

[64] F. Wu, Z. Li, Y. Li, B. Ding, and J. Gao. FedBiOT: LLM local fine-tuning in federated learning without full model. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 3345–3355, 2024.

[65] F. Wu, X. Liu, H. Wang, X. Wang, and J. Gao. On the client preference of LLM fine-tuning in federated learning. *arXiv preprint arXiv:2407.03038*, 2024.

[66] A. Yang, B. Yang, B. Zhang, B. Hui, B. Zheng, B. Yu, C. Li, D. Liu, F. Huang, H. Wei, et al. Qwen2. 5 technical report. *arXiv preprint arXiv:2412.15115*, 2024.

[67] H. Yang, X.-Y. Liu, and C. D. Wang. FinGPT: Open-Source Financial Large Language Models. *FinLLM Symposium at IJCAI 2023*, 2023.

[68] R. Ye, R. Ge, F. Yuchi, J. Chai, Y. Wang, and S. Chen. Leveraging unstructured text data for federated instruction tuning of large language models. In *International Workshop on Trustworthy Federated Learning*, pages 119–131. Springer, 2024.

[69] R. Ye, R. Ge, X. Zhu, J. Chai, D. Yaxin, Y. Liu, Y. Wang, and S. Chen. FedLLM-Bench: Realistic benchmarks for federated learning of large language models. *Advances in Neural Information Processing Systems*, 37:111106–111130, 2024.

[70] R. Ye, W. Wang, J. Chai, D. Li, Z. Li, Y. Xu, Y. Du, Y. Wang, and S. Chen. OpenFedLLM: Training large language models on decentralized private data via federated learning. In *Proceedings of the 30th ACM SIGKDD conference on knowledge discovery and data mining*, pages 6137–6147, 2024.

[71] L. Yi, H. Yu, G. Wang, X. Liu, and X. Li. pFedLoRA: Model-Heterogeneous Personalized Federated Learning with LoRA Tuning. *arXiv preprint arXiv:2310.13283*, 2023.

[72] W. Zeng, C. Xu, Y. Zhao, J.-G. Lou, and W. Chen. Automatic instruction evolving for large language models. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 6998–7018, 2024.

[73] ZeroShot. Twitter financial news sentiment dataset. `https://huggingface.co/datasets/zeroshot/twitter-financial-news-sentiment`, 2022.

[74] S. Zhang, L. Dong, X. Li, S. Zhang, X. Sun, S. Wang, J. Li, R. Hu, T. Zhang, F. Wu, et al. Instruction tuning for large language models: A survey. *arXiv preprint arXiv:2308.10792*, 2023.

[75] S. Zhou, S. Wang, Z. Yuan, M. Shi, Y. Shang, and D. Yang. GSQ-Tuning: Group-Shared Exponents Integer in Fully Quantized Training for LLMs On-Device Fine-tuning. *arXiv preprint arXiv:2502.12913*, 2025.

[76] D. M. Ziegler, N. Stiennon, J. Wu, T. B. Brown, A. Radford, D. Amodei, P. Christiano, and G. Irving. Fine-tuning language models from human preferences. *arXiv preprint arXiv:1909.08593*, 2019.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: We have accurately stated the main contributions of this paper in the abstract and introduction.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: We include the discussion of the limitations in the conclusion section.

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [NA]

Justification: The paper does not include theoretical results.

Guidelines:

- The answer NA means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: We provide code of all conducted experiments for reproducibility.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [Yes]

Justification: The experiments in this paper were conducted using public datasets with designed data splitting for FL. The links for code and datasets are provided in the paper. All hyper-parameters used in the experiments are shown in the Appendix.

Guidelines:

- The answer NA means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyper-parameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

   Answer: [Yes]

   Justification: The details for all training and test data are provided in the paper. All hyper-parameters used in the experiments are shown in the Appendix.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: The error bars are reported in the Appendix figures. However, we did not provide error bars in the tables in the main text because it would be too computationally expensive.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).

- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: We specify which types of GPUs were used for all experiments.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: The research conducted in the paper conforms with the NeurIPS Code of Ethics.

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: We discussed both potential positive societal impacts and negative societal impacts in the paper.

    Guidelines:

    - The answer NA means that there is no societal impact of the work performed.
    - If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.

- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

    Answer: [Yes]

    Justification: We provide clear instructions in the provided code repository on how to use the released model checkpoints.

    Guidelines:

    - The answer NA means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: All assets used in this paper are explicitly mentioned and properly respected.

    Guidelines:

    - The answer NA means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.

- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [Yes]

    Justification: The code, model, and datasets introduced in the paper are well documented.

    Guidelines:

    - The answer NA means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [NA]

    Justification: This paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [NA]

    Justification: This paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.

- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does not impact the core methodology, scientific rigorousness, or originality of the research, declaration is not required.

    Answer: [NA]

    Justification: We do not use LLMs for any core method development in this research in the paper.

    Guidelines:

    - The answer NA means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy (`https://neurips.cc/Conferences/2025/LLM`) for what should or should not be described.

A Detailed experimental settings

This section provides details on client-level data distribution (Section A.1), the base models (Section A.2), and hyper-parameter configurations (Section A.3) used in the experiments.

A.1 Client-level data distribution

From Table 9, we observe that the number of samples is relatively balanced across clients within each domain – a realistic assumption for cross-institutional FL settings. Variation in instruction and response lengths arises primarily from the inherent properties of the source datasets. We did not partition client data based on class distributions, as the instruction-tuning datasets used in this study are unlabeled. However, the semantic similarity metrics reveal meaningful differences across clients in each domain, particularly in the general NLP, medical, and code domains, where similarity scores approach 0, indicating a notable degree of heterogeneity. The higher similarity values observed in the finance domain are likely inherent to the original dataset prior to splitting.

Table 9: Statistical summary of client-level data distribution for four challenges. "# of Samples" indicates the number of data points per client. "Instruct Avg Length" and "Response Avg Length" represent the average sequence lengths of instructions and responses, respectively, for each client. For "Semantic Similarity," we compute embeddings for each instruction–response pair using the pre-trained all-MiniLM-L6-v2 model and then average these embeddings for each client. After applying dimensionality reduction, we calculate the cosine similarity between each client and all other clients, reporting the average similarity value per client to illustrate semantic variation (with values closer to 1 indicating greater similarity).

	General NLP
# of Samples	[2601, 2601, 2600, 2600, 2600, 2600, 2600, 2600, 2600, 2600, 2600, 2600, 2600, 2600, 2600, 2600, 2600, 2600, 2600, 2600]
Instruct Avg Length	[59, 59, 59, 60, 59, 59, 59, 59, 60, 59, 59, 60, 59, 59, 60, 58, 59, 60, 59, 59]
Response Avg Length	[668, 668, 663, 671, 680, 673, 676, 664, 686, 672, 688, 708, 646, 692, 688, 680, 689, 693, 671, 660]
Semantic Similarity	[-0.02, 0.04, 0.03, -0.13, -0.07, 0.03, 0.04, -0.14, -0.0, -0.0, -0.1, -0.14, 0.04, -0.05, -0.04, -0.14, -0.08, -0.14, -0.05, 0.04]
	Finance
# of Samples	[1536, 1536, 1536, 1536, 1536, 1536, 1536, 1536, 1536, 1536, 1536, 1536, 1536, 1536, 1536, 1536, 1536, 1536, 1536, 1536, 1536, 1536, 1535, 1535, 1535, 1535, 1535, 1535, 1535, 1535, 1535, 1535, 1535, 1535, 1535, 1535, 1535, 1535, 1535, 1535, 1535, 1535, 1535, 1535, 1535, 1535, 1535, 1535, 1535, 1535]
Instruct Avg Length	[224, 225, 223, 223, 224, 223, 225, 223, 220, 226, 226, 224, 223, 226, 228, 225, 223, 226, 223, 227, 225, 225, 228, 225, 221, 225, 223, 225, 223, 221, 224, 224, 225, 226, 223, 226, 224, 226, 221, 226, 227, 223, 223, 220, 228, 229, 224, 227, 226, 224]
Response Avg Length	[9, 9, 9, 9, 9, 9, 9, 9, 9, 9, 9, 9, 9, 9, 9, 9, 9, 9, 9, 9, 9, 9, 9, 9, 9, 9, 9, 9, 9, 9, 9, 9, 9, 9, 9, 9, 9, 9, 9, 9, 9, 9, 9, 9, 9, 9, 9, 9, 9, 9]
Semantic Similarity	[0.72, 0.47, 0.29, 0.65, 0.51, 0.79, 0.79, 0.35, 0.33, 0.67, 0.78, 0.14, 0.79, 0.78, 0.76, 0.7, 0.7, 0.77, 0.71, 0.13, 0.78, 0.58, 0.72, 0.79, 0.64, 0.79, 0.65, 0.63, 0.46, 0.31, 0.79, 0.61, 0.69, 0.79, 0.79, 0.79, 0.77, 0.79, 0.24, 0.41, 0.77, 0.34, 0.77, 0.26, 0.74, 0.76, 0.79, 0.58, 0.77, 0.77]
	Medical
# of Samples	[1698, 1698, 1698, 1698, 1698, 1698, 1698, 1698, 1698, 1698, 1698, 1698, 1698, 1698, 1698, 1697, 1697, 1697, 1697, 1697]
Instruct Avg Length	[92, 92, 92, 90, 92, 93, 94, 91, 92, 91, 92, 91, 91, 92, 92, 92, 91, 93, 92, 92]
Response Avg Length	[346, 361, 346, 331, 340, 357, 362, 343, 343, 343, 353, 354, 355, 345, 351, 358, 341, 346, 344, 352]
Semantic Similarity	[-0.08, -0.04, -0.02, -0.08, -0.05, -0.03, -0.03, -0.03, -0.05, -0.07, -0.04, -0.07, -0.08, -0.03, -0.02, -0.06, -0.06, -0.09, -0.02, -0.08]
	Code
# of Samples	[2003, 2003, 2002, 2002, 2002, 2002, 2002, 2002, 2002, 2002]
Instruct Avg Length	[96, 96, 101, 97, 97, 99, 97, 98, 98, 98]
Response Avg Length	[199, 191, 193, 202, 193, 201, 200, 190, 196, 200]
Semantic Similarity	[0.03, -0.1, -0.24, -0.27, -0.05, -0.15, 0.05, 0.03, 0.04, -0.14]

A.2 Base model configurations

Table 10 presents detailed specifications of the base models used in the experiments, including the number of parameters and vocabulary size for each model.

A.3 Hyper-parameter configurations

To ensure a fair comparison across different pre-trained base models, consistent hyper-parameters are applied to all tested LLMs. Table 11 outlines the local training hyper-parameter configurations used for base model benchmarking, as discussed in Section 4. For the Mistral 24B models, the DoRA rank and alpha values are adjusted to 16 and 32, respectively, to accelerate training and enable execution on a single GPU.

Table 10: Detailed information of the base models used in the experiments.

Groups	Model name	# parameters	Vocabulary size (thousand)
Non-Instruct	Mistral-7B-v0.3 [5]	7 B	33
	Mistral-Small-24B-Base-2501 [3]	24 B	131
	Gemma-2-9B [58]	9 B	256
	Phi-4 [1]	14 B	100
	Llama-3.2-1B [24]	1 B	128
	Llama-3.2-3B [24]	3 B	128
	Llama-3.1-8B [24]	8 B	128
	Qwen2-0.5B [18]	0.5 B	152
	Qwen2.5-1.5B [66]	1.5 B	152
	Qwen2.5-3B [66]	3 B	152
	Qwen2.5-7B [66]	7 B	152
	SmolLM2-135M [7]	135 M	49
	SmolLM2-360M [7]	360 M	49
	SmolLM2-1.7B [7]	1.7 B	49
Instruct	Mistral-7B-Instruct-v0.3 [6]	7 B	33
	Mistral-Small-24B-Instruct-2501 [4]	24 B	131
	Gemma-2-9B-Instruct [51]	9 B	256
	Phi-4-Mini-Instruct [2]	3.8 B	200
	Llama-3.2-1B-Instruct [24]	1 B	128
	Llama-3.2-3B-Instruct [24]	3 B	128
	Llama-3.1-8B-Instruct [24]	8 B	128
	Qwen2-0.5B-Instruct [18]	0.5 B	152
	Qwen2.5-1.5B-Instruct [66]	1.5 B	152
	Qwen2.5-3B-Instruct [66]	3 B	152
	Qwen2.5-7B-Instruct [66]	7 B	152
	SmolLM2-135M-Instruct [7]	135 M	49
	SmolLM2-360M-Instruct [7]	360 M	49
	SmolLM2-1.7B-Instruct [7]	1.7 B	49

B Additional experimental results

B.1 Federated fine-tuning on non-instruct pre-trained base models

Tables 12–15 present the performance and system metrics of non-instruct pre-trained base models after federated fine-tuning across four challenges. Overall, these non-instruct base models underperform compared to their instruct-tuned counterparts, particularly in the case of smaller models, which may require more extensive fine-tuning to achieve competitive results. Gemma2-9B and Phi-4 demonstrate consistently strong performance across all domains, albeit with the highest communication and memory overhead. Notably, smaller models such as Qwen2.5-7B and Qwen2.5-3B achieve competitive results on the finance task, outperforming larger models in the 9B and 14B parameter range. In contrast, the Llama model family exhibits suboptimal performance on the medical challenge, suggesting a need for more tailored fine-tuning strategies in this domain.

B.2 Pareto analysis of evaluation performance

To analyze the results from tables 12 – 15 from a computational trade-off perspective, we present the Pareto plots in figures 4 – 7 for the different base models (non-instruct version) when federated fine-tuned for the different tasks. When evaluating communication costs, the Qwen family of models stands out largely as the model family with the optimal trade-off. The Phi-4 model has a strong model performance (with the exception of the Finance challenge), but this is outweighed by the communication costs, which can vary up to a factor of 5. For performance versus VRAM evaluation, the results are mixed: there is strictly no optimal model in the General NLP challenge, whereas for

Table 11: Local training hyper-parameter configurations used for base model benchmarking. The values of DoRA rank and Alpha are adjusted to 16 and 32 when training Mistral 24B models.

Hyper-parameters	Values
DoRA/LoRA rank (r)	32
DoRA/LoRA Alpha	64
Batch size	16
Optimizer	AdamW
Sequence length	512
Maximum number of steps	10
Accumulation steps	1
Maximum gradient norm	1.0
LR scheduler over rounds	Cosine annealing
LR scheduler over steps	Constant
Maximum LR	5e-5
Minimum LR	1e-6
Quantization	4-bit
Precision	bf16

Table 12: **Comparison of different non-instruct base models federated fine-tuned on the General NLP challenge.** The accuracy values (%) are reported on different downstream tasks. Comm. represents the total communication costs over FL fine-tuning, while Mem. stands for the VRAM costs per client. The highest average performance is indicated in **bold**, while the second-highest is underlined.

Models	STEM (%)	Social Sciences (%)	Humanities (%)	Average (%)	Comm. (GB)	Mem. (GB)
Mistral-7B-v0.3	3.39	12.97	25.80	14.05	12.31	25.08
Gemma2-9B	28.83	54.99	41.59	41.80	15.52	60.48
Phi-4 (14B)	37.87	72.67	46.99	**52.51**	50.77	57.70
Llama3.2-1B	3.36	4.58	2.23	3.39	3.31	29.08
Llama3.2-3B	1.21	1.43	0.79	1.14	7.05	32.97
Llama3.1-8B	0.13	0.62	0.23	0.33	8.10	46.58
Qwen2-0.5B	2.73	5.36	3.63	3.91	2.61	28.40
Qwen2.5-1.5B	21.12	21.68	24.19	22.33	5.48	32.13
Qwen2.5-3B	4.09	6.82	2.72	4.55	8.90	33.86
Qwen2.5-7B	41.55	52.62	34.20	42.79	11.99	49.23
SmolLM2-135M	3.68	1.30	3.17	2.72	1.42	9.37
SmolLM2-360M	0.03	0.03	0.06	0.04	2.52	10.91
SmolLM2-1.7B	0.13	0.19	0.32	0.21	4.97	15.25

the Finance, Medical, and Coding challenges, the Qwen and Mistral models appear to be an optimal choice. Notably, the SmolLM2 model family stands out showing strong model performance relative to their VRAM usage.

Similarly, figures 8 – 11 show the Pareto plots for the different base models (Instruct version) when federated fine-tuned for the different tasks, which are computed from tables 3 – 6. When evaluating performance versus communication costs, the Qwen family of models again stands out for the General NLP and Medical challenges, whereas the Llama model families dominate the frontier with a strong Pass@1 scores relative to the communication costs. Depending on the downstream tasks, the SmolLM model family show a strong performance for the Medical challenge with a fraction of the communication cost. Interestingly, when evaluating performance versus VRAM costs, we observe a similar behaviour where the Qwen model family stands out for the General NLP and Medical challenges. The Mistral-7B-Instruct-v0.3 model is a strong contender, dominating the Finance challenge and lying on the frontier for the remaining challenges. Similar to the non-instruct results, the SmolLM2 model families tend to perform reasonably well for the VRAM requirements.

B.3 Analysis on 24B base models

Additionally, we extend our investigation to the upper bounds of federated fine-tuning by experimenting with state-of-the-art 24B-parameter models—Mistral-Small-24B-Base-2501 [3] and Mistral-Small-24B-Instruct-2501 [4]—which have been reported to achieve performance comparable to Llama-3.3-70B [4].

Table 13: **Comparison of different non-instruct base models federated fine-tuned on the Finance challenge.** The accuracy values (%) are reported on different downstream tasks. Comm. represents the total communication costs over FL fine-tuning, while Mem. stands for the VRAM costs per client. The highest average performance is indicated in **bold**, while the second-highest is underlined.

Models	FPB (%)	FIQA (%)	TFNS (%)	Average (%)	Comm. (GB)	Mem. (GB)
Mistral-7B-v0.3	73.60	68.09	39.82	60.50	30.77	17.24
Gemma2-9B	27.89	59.54	20.64	36.02	38.80	43.09
Phi-4 (14B)	29.13	61.51	26.51	39.05	126.93	40.81
Llama3.2-1B	46.12	42.11	40.58	42.94	8.28	12.64
Llama3.2-3B	59.74	22.04	60.80	47.53	17.63	16.40
Llama3.1-8B	60.31	39.14	61.35	53.60	30.77	27.30
Qwen2-0.5B	51.65	53.95	37.77	47.79	6.54	13.94
Qwen2.5-1.5B	65.18	35.53	52.05	50.92	13.69	17.54
Qwen2.5-3B	74.75	51.32	75.08	**67.05**	22.26	20.27
Qwen2.5-7B	71.86	52.30	74.20	66.12	29.97	28.10
SmolLM2-135M	29.54	57.89	22.24	36.56	3.55	5.28
SmolLM2-360M	50.74	24.34	51.63	42.24	6.30	6.36
SmolLM2-1.7B	46.95	45.07	48.87	46.96	12.42	10.41

Table 14: **Comparison of different non-instruct base models federated fine-tuned on the Medical challenge.** The accuracy values (%) are reported on different downstream tasks. Comm. represents the total communication costs over FL fine-tuning, while Mem. stands for the VRAM costs per client. The highest average performance is indicated in **bold**, while the second-highest is underlined.

Models	PubMedQA (%)	MedMCQA (%)	MedQA (%)	CareQA (%)	Average (%)	Comm. (GB)	Mem. (GB)
Mistral-7B-v0.3	64.80	0.05	1.26	0.02	16.53	12.31	20.56
Gemma2-9B	61.00	26.89	27.18	27.22	**35.57**	15.52	50.57
Phi-4 (14B)	62.60	11.40	8.88	32.24	28.78	50.77	46.24
Llama3.2-1B	24.20	2.06	0.16	1.07	6.87	3.31	20.07
Llama3.2-3B	0.20	0.02	0.08	0.00	0.08	7.05	24.23
Llama3.1-8B	24.20	2.06	0.16	1.07	6.87	12.31	36.31
Qwen2-0.5B	51.80	10.11	13.90	9.50	21.33	2.61	18.53
Qwen2.5-1.5B	0.00	8.99	2.04	6.85	4.47	5.48	21.36
Qwen2.5-3B	27.80	6.26	2.28	2.54	9.72	8.90	24.79
Qwen2.5-7B	33.40	17.62	18.85	30.21	25.02	11.99	39.53
SmolLM2-135M	2.20	18.81	8.80	17.75	11.89	1.42	6.88
SmolLM2-360M	7.20	0.05	0.08	0.02	1.84	2.52	7.65
SmolLM2-1.7B	0.80	3.87	18.38	1.23	6.07	4.97	12.04

Table 16 presents the evaluation results of both models across the four challenges. As expected, the instruct-tuned model outperforms the non-instruct base model across all tasks. In particular, Mistral-24B-Instruct achieves state-of-the-art performance in the general NLP (65.28%) and coding (63.62%) challenges, while also delivering competitive results in the finance challenge (83.06%). However, its performance on the medical QA task remains relatively modest.

From a system perspective, despite the model's large size, the use of a reduced DoRA configuration—compared to the settings in Table 11—results in manageable communication and memory overhead. This enables efficient training on a single NVIDIA H100 NVL GPU (94 GB).

B.4 Analysis on domain-specific pre-trained base models

Table 17 presents the evaluation results of several domain-specific pre-trained base models after federated fine-tuning on medical and coding tasks. These models were initially fine-tuned on domain-relevant data (medical or coding) by their respective providers prior to federated training. As shown, they outperform general-purpose base models reported in Tables 3–6, which is expected given their prior exposure to domain-specific knowledge before participating in federated fine-tuning.

Table 15: **Comparison of different non-instruct base models federated fine-tuned on the Coding challenge.** The pass@1 scores (%) are reported on different downstream tasks. Comm. represents the total communication costs over FL fine-tuning, while Mem. stands for the VRAM costs per client. The highest average performance is indicated in **bold**, while the second-highest is underlined.

Models	MBPP (%)	HumanEval (%)	MultiPL-E (JS) (%)	MultiPL-E (C++) (%)	Average (%)	Comm. (GB)	Mem. (GB)
Mistral-7B-v0.3	40.20	30.49	36.02	28.57	33.82	6.15	22.27
Gemma2-9B	50.40	49.39	44.72	39.75	46.07	7.76	56.58
Phi-4 (14B)	62.40	34.76	35.40	52.80	**46.34**	25.39	47.04
Llama3.2-1B	20.40	17.68	14.29	16.15	17.13	1.66	20.89
Llama3.2-3B	33.40	27.44	24.84	19.25	26.23	3.53	24.99
Llama3.1-8B	45.60	35.37	45.34	37.89	41.05	6.15	36.96
Qwen2-0.5B	19.00	10.37	16.77	16.15	15.57	1.31	18.37
Qwen2.5-1.5B	42.40	9.15	39.75	26.71	29.50	2.74	22.48
Qwen2.5-3B	40.00	15.85	42.24	36.65	33.68	4.45	27.03
Qwen2.5-7B	56.60	23.17	52.17	49.07	45.25	5.99	39.36
SmolLM2-135M	3.40	4.27	4.97	3.11	3.94	0.71	7.01
SmolLM2-360M	21.00	14.02	12.42	9.94	14.35	1.26	8.24
SmolLM2-1.7B	31.80	23.17	24.22	23.60	25.70	2.48	11.87

Table 16: **Evaluation performance using Mistral-24B models on four challenges.** "SS" refers to Social Sciences. For the medical challenge, "PMQA", "MMCQA", "MQA" and "CQA" correspond to PubMedQA, MedMCQA, MedQA, and CareQA, respectively. For the coding challenge, "HE", "M-JS" and "M-C++" represent HumanEval, MultiPL-E (JS), and MultiPL-E (C++).

	General NLP						Medical						
Models	STEM	SS	Humanities	Avg	Comm.	Mem.	PMQA	MMCQA	MQA	CQA	Avg	Comm.	Mem.
Mistral-24B-Base-2501	52.81	77.90	45.04	58.58	13.66	64.85	71.80	1.15	0.86	1.73	18.88	13.66	54.24
Mistral-24B-Instruct-2501	54.14	79.43	62.27	65.28	13.66	65.22	72.20	26.20	30.71	44.24	43.34	13.66	54.50
	Finance						Code						
Models	FPB	FIQA	TFNS	Avg	Comm.	Mem.	MBPP	HE	M-JS	M-C++	Avg	Comm.	Mem.
Mistral-24B-Base-2501	80.94	84.21	78.39	81.18	34.16	47.87	56.40	42.68	55.28	45.96	50.08	6.83	58.99
Mistral-24B-Instruct-2501	85.97	80.59	82.62	83.06	34.16	48.14	62.00	69.51	62.73	60.25	63.62	6.83	56.85

Table 17: **Performance evaluation using domain-specific pre-trained base models on medical and coding challenges.** Results are based on submissions to the FlowerTune LLM Leaderboard [2].

Models	MBPP (%)	HumanEval (%)	MultiPL-E (JS) (%)	MultiPL-E (C++) (%)	Average (%)	Comm. (GB)
Deepseek-Coder-7B-Instruct-v1.5	56.80	64.63	55.90	57.76	58.77	2.9
Qwen2.5-Coder-7B-Instruct	64.00	27.43	72.67	60.24	56.08	1.5
	PubMedQA (%)	MedMCQA (%)	MedQA (%)	CareQA (%)	Average (%)	Comm. (GB)
Bio-Medical-Llama-3-8B	70.40	60.93	65.82	55.36	63.12	1.6

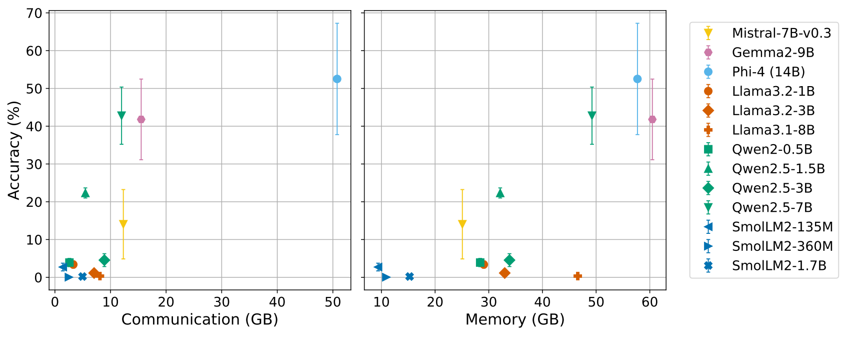

Figure 4: **Accuracy (%) versus system performance for different non-instruct base models federated fine-tuned on the General NLP challenge, presented in table 12.** The errorbar indicates ± 1 Std. Dev. on the different downstream tasks.

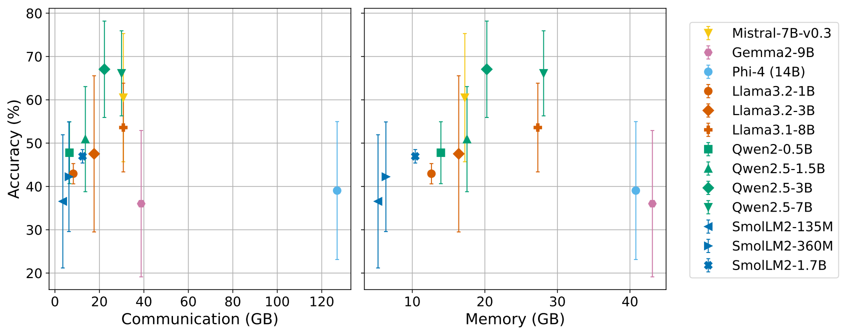

Figure 5: **Accuracy (%) versus system performance for different non-instruct base models federated fine-tuned on the Finance challenge, presented in table 13.** The errorbar indicates ±1 Std. Dev. on the different downstream tasks.

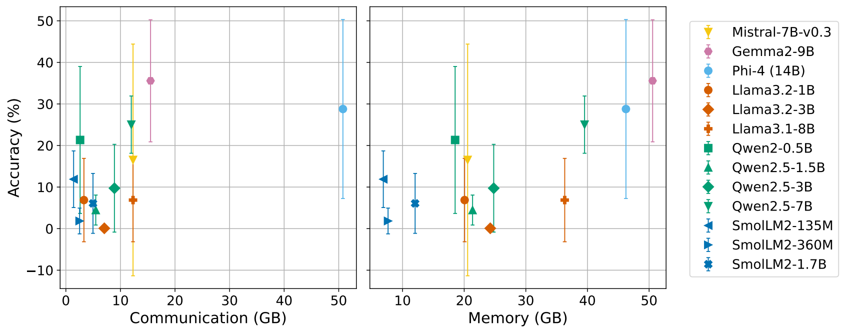

Figure 6: **Accuracy (%) versus system performance for different non-instruct base models federated fine-tuned on the Medical challenge, presented in table 14.** The errorbar indicates ±1 Std. Dev. on the different downstream tasks.

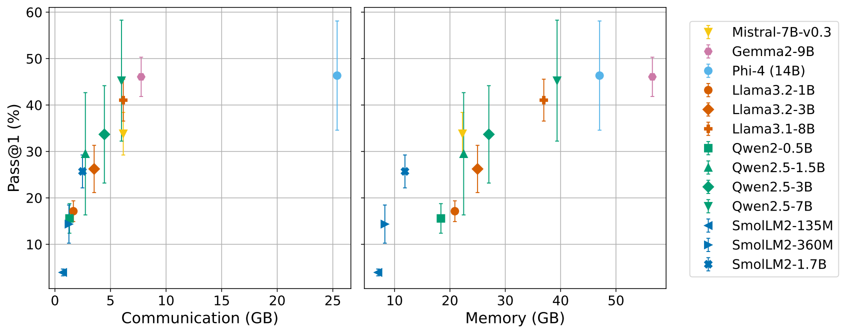

Figure 7: **Pass@1 scores (%) versus system performance for different non-instruct base models federated fine-tuned on the Coding challenge, presented in table 15.** The errorbar indicates ±1 Std. Dev. on the different downstream tasks.

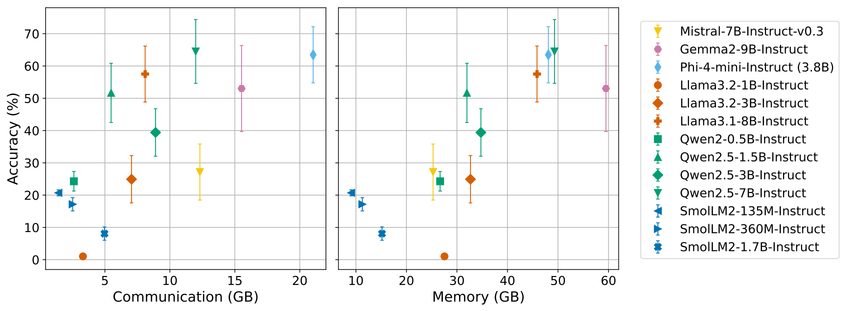

Figure 8: **Accuracy (%) versus system performance for different base models (Instruct version) federated fine-tuned on the General NLP challenge, presented in table 3.** The errorbar indicates ±1 Std. Dev. on the different downstream tasks.

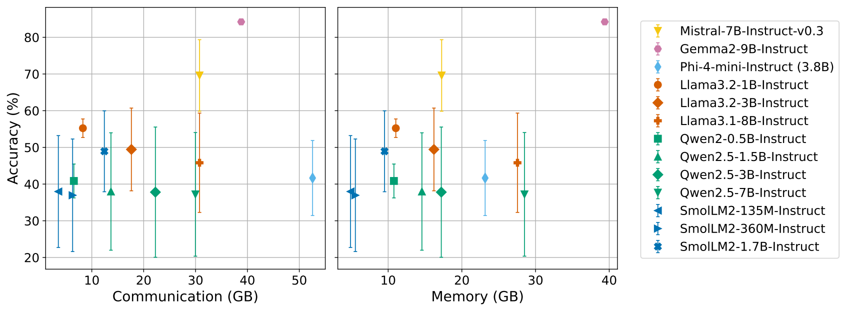

Figure 9: **Accuracy (%) versus system performance for different base models (Instruct version) federated fine-tuned on the Finance challenge, presented in table 4.** The errorbar indicates ±1 Std. Dev. on the different downstream tasks.

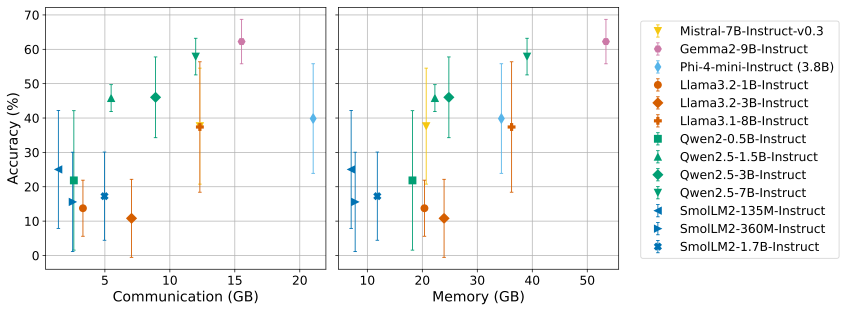

Figure 10: **Accuracy (%) versus system performance for different base models (Instruct version) federated fine-tuned on the Medical challenge, presented in table 5.** The errorbar indicates ±1 Std. Dev. on the different downstream tasks.

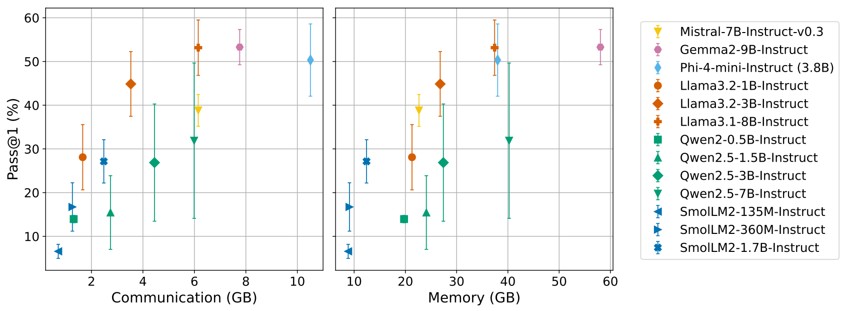

Figure 11: **Pass@1 scores (%) versus system performance for different base models (Instruct version) federated fine-tuned on the Coding challenge, presented in table 6.** The errorbar indicates ±1 Std. Dev. on the different downstream tasks.

C Example samples from training datasets

Tables 18–21 show the example samples from the training datasets used in four challenges.

Table 18: Example sample from the training dataset used in **GeneralNLP** challenge.

Instruction What are the three primary colors? # *Response* The three primary colors are red, blue, and yellow. These colors are called primary because they cannot be created by mixing other colors and all other colors can be made by combining them in various proportions. In the additive color system, used for light, the primary colors are red, green, and blue (RGB).

Table 19: Example sample from the training dataset used in **Finance** challenge.

Instruction Google parent Alphabet Inc. reported revenue and earnings that fell short of analysts' expectations, showing the company's search advertising juggernaut was not immune to a slowdown in the digital ad market. The shares fell more than 6%. # *Response* Moderately negative

Table 20: Example sample from the training dataset used in **Medical** challenge.

Instruction What is the relationship between very low Mg2+ levels, PTH levels, and Ca2+ levels? # *Response* Very low Mg2+ levels correspond to low PTH levels which in turn results in low Ca2+ levels.

Table 21: Example sample from the training dataset used in **Coding** challenge.

# *Instruction* 	
Write a replace method for a string class which replaces the given string with a given set of characters. string = "Hello World!" replace_with = "Greetings!"
 	
# *Response* 	
def replace(self, replace_with):
new_string = ""
for char in self:
if char == " ":
new_string += replace_with
else:
new_string += char
return new_string

D How to participate in the FlowerTune LLM Leaderboard?

The FlowerTune LLM Leaderboard is open for ongoing submissions. Please refer to Figure 12 for a guide on how to participate. Detailed instructions and submission guidelines are available on the official leaderboard website [2]. In addition to contributions to the existing challenges, we welcome proposals for new settings and tasks. We look forward to your participation and collaboration in advancing the frontiers of federated LLM fine-tuning.

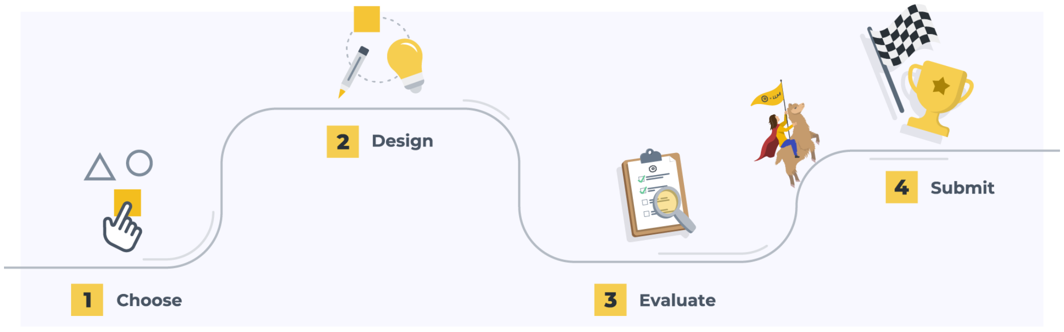

Figure 12: Workflow for participating in the FlowerTune LLM Leaderboard. (1) Explore the available challenge boards and select a challenge; (2) propose and implement a federated fine-tuning approach using the provided template code; (3) evaluate the performance of your fine-tuned LLM; (4) submit your results to claim a position on the leaderboard.

D.1 Overview of GPUs used by participants

Submissions to the FlowerTune LLM Leaderboard have come from a number of institutions and individual contributors, and Figure 13 shows which GPUs were used more frequently. The NVIDIA A100 (40GB) and A40 (which has 48GB of VRAM) were the most popular cards. This is unsurprising as that amount of VRAM is needed to fine-tune the largest LLMs considered in these leaderboards. These cards are also widely available in all major cloud providers.

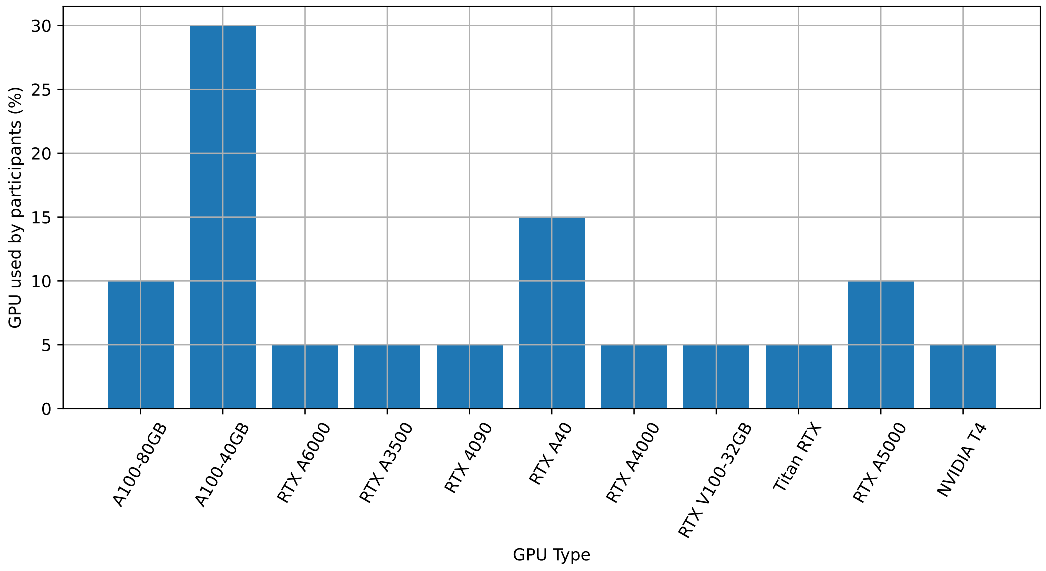

Figure 13: GPU types used across submissions from institutional and individual community contributors.