ICASSP 2024 - 2024 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP) | 979-8-3503-4485-1/24/$31.00 ©2024 IEEE | DOI: 10.1109/ICASSP48485.2024.10447454

TOWARDS BUILDING THE FEDERATEDGPT: FEDERATED INSTRUCTION TUNING

Jianyi Zhang[1], Saeed Vahidian[1*], Martin Kuo[1*], Chunyuan Li[2], Ruiyi Zhang[3], Tong Yu[3], Guoyin Wang[4], Yiran Chen[1+]*

[1] Duke University, [2] Microsoft Research, [3] Adobe Research, [4] Amazon

ABSTRACT

While "instruction-tuned" generative large language models (LLMs) have demonstrated an impressive ability to generalize to new tasks, the training phases heavily rely on large amounts of diverse and high-quality instruction data (such as ChatGPT and GPT-4). Unfortunately, acquiring high-quality data, especially when it comes to human-written data, can pose significant challenges both in terms of cost and accessibility. Moreover, concerns related to privacy can further limit access to such data, making the process of obtaining it a complex and nuanced undertaking. To tackle this issue, our study introduces a new approach called **Fed**erated **I**nstruction **T**uning (FedIT), which leverages federated learning (FL) as the learning framework for the instruction tuning of LLMs. This marks the first exploration of FL-based instruction tuning for LLMs. This is especially important since text data is predominantly generated by end users. For example, collecting extensive amounts of everyday user conversations can be a useful approach to improving the generalizability of LLMs, allowing them to generate authentic and natural responses. Therefore, it is imperative to design and adapt FL approaches to effectively leverage these users' diverse instructions stored on local devices while mitigating concerns related to the data sensitivity and the cost of data transmission. In this study, we leverage extensive qualitative analysis, including the prevalent GPT-4 auto-evaluation to illustrate how our FedIT framework enhances the performance of LLMs. Utilizing diverse instruction sets on the client side, FedIT outperforms centralized training with only limited local instructions.

1. INTRODUCTION

Large Language Models (LLMs) have become ubiquitous in natural language processing (NLP) [1, 2, 3], where one single model can perform well on various language tasks, including established tasks such as text generation, machine translation, and question answering, as well as novel application-oriented tasks in human daily life [4, 5]. To align LLM to follow human intents, instruction-tuning has been proposed by fine-tuning LLM on instruction-following data [6, 7, 8]. Though instruction-tuning has demonstrated great effectiveness in improving the zero and few-shot generalization capabilities of LLM, its performance on real-world tasks is contingent on the *quantity, diversity,* and *quality* of the collected instructions [9, 7]. The process of collecting these instructions can be expensive [10, 7]. Besides, the increasing awareness of data sensitivity highlights a significant challenge in acquiring extensive and high-quality instructions [11, 12]. For instance, collecting vast amounts of daily conversations from users is a valuable means of providing guidance for LLMs, enabling them to generate authentic and genuine responses. However, privacy concerns may hinder users from sharing their conversations, resulting in a limited quantity of instructions that are not fully representative of the target population. Likewise, many companies treat their instructions as proprietary assets that are closely guarded. They are reluctant to share their instructions with external parties, as they often contain confidential and proprietary information that is critical to their success and profitability [13].

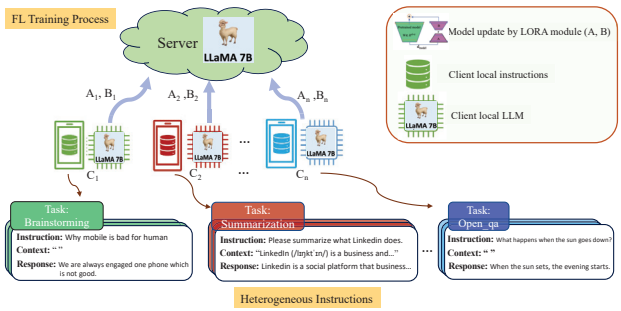

Fig. 1. The framework of Federated Instruction Tuning

We aim to tackle these challenges by exploring the potential of federated learning (FL) as a promising solution [14]. This collaborative learning technique enables many clients to learn a shared model jointly without sharing their sensitive data. In particular, in our proposed federated instruction-tuning, clients initially download a global LLM from a central server and subsequently compute local model updates using their respective local instructions. These local updates are then transmitted back to the server, where they are aggregated and integrated to update the global LLM. Given that clients often have limited computational resources in comparison to traditional centralized training cloud servers, which can uti-

* Equal Contribution. Work done at Duke University

+ This work is supported in part by the grants NSF-2112562, NSF-2332744 and ARO W911NF-23-2-0224

lize thousands of GPUs [15] to fully fine-tune all parameters of LLMs, we resort to parameter-efficient tuning techniques. This leads to a significant decrease in computational and communication demands as it reduces the number of trainable parameters on each device. Thus, our proposed framework enables efficient utilization of the computational resources available on local edge devices, which are commonly accessible, as well as their diverse local instructions. Our major contributions are summarized as follows. First, we make the first attempt to leverage FL for instruction tuning (FedIT) of LLMs. We show that we can circumvent the above-mentioned challenges of predominant instruction tuning by exploiting the diverse sets of available instructions from the users in the FL system. Besides, a comprehensive study is conducted on the heterogeneity and diversity within the federated instruction tuning. We employ the GPT-4 auto-evaluation method, which has been widely utilized in related research [16, 17], to demonstrate the effectiveness of our FedIT approach in enhancing response quality by leveraging diverse available instructions. We also have developed and released a GitHub repository called *Shepherd*[1], which has been designed to provide ease of customization and adaptability, thereby offering benefits for future research endeavors in this field.

2. FEDERATED INSTRUCTION TUNING

Drawing on the successful application of FL in various machine learning domains to offer privacy protection, we introduce the FedIT framework. By harnessing the advantages of FL, our framework enables secure and cost-effective LLM instruction tuning. The overall framework, illustrated in Figure 1, involves two primary components: local training operations on the client side and scheduling and aggregation operations on the server side, which work together to ensure efficient training. Our framework assigns an LLM to each client and performs client selection to determine which clients will participate in local instruction tuning. During instruction tuning, clients use their local instruction dataset to update a small, trainable adapter that is added to the pre-trained model weights. This approach reduces the cost of fine-tuning and makes it compatible with the limited computational resources of local devices. Upon completion, clients send the updated adapter back to the server, which aggregates the received adapters' parameters and conducts another round of client selection. This iterative process continues until convergence. Our FedIT framework for instruction tuning is designed to address the challenges of collecting high-quality data and ensuring data privacy by keeping the instructions on the local devices throughout the process. By ensuring data sensitivity protection, we can encourage more clients to participate in the federated instruction tuning. Consequently, the combined instruction dataset from all clients can encompass a broader range of topics, tasks, and valuable information, as clients may come from different areas and possess domain-specific expertise. This FL approach enables our framework to effectively adapt to diverse and evolving instruction datasets, resulting in more robust and generalized LLM performance. Moreover, our FedIT methodology incorporates a parameter-efficient fine-tuning (PEFT) technique, known as LoRA [18], to facilitate local training. This method reduces computational and communication overheads for local edge devices that have limited system resources. As a result, we can leverage the computational capabilities of a multitude of distributed local edge devices that are often disregarded in conventional centralized instruction tuning. This feature enhances the scalability of our FedIT solution, enabling it to address large-scale challenges effectively.

2.1. Heterogeneity of Instructional Data

Beyond the practical benefits of FedIT, our research makes a unique contribution by presenting a scenario for instruction tuning of LLMs where statistical heterogeneity can serve as a positive factor for federated learning. Our work demonstrates that the extensive heterogeneous and diverse set of instructions can, in fact, be a blessing factor for our FedIT approach. For instance, different clients may have different instruction tasks, such as open-domain QA and writing. The content and format of these instructions can be substantially different. For example, QA tasks typically require fact-based questions and answers, while writing tasks involve instructions for generating coherent and meaningful sentences. In

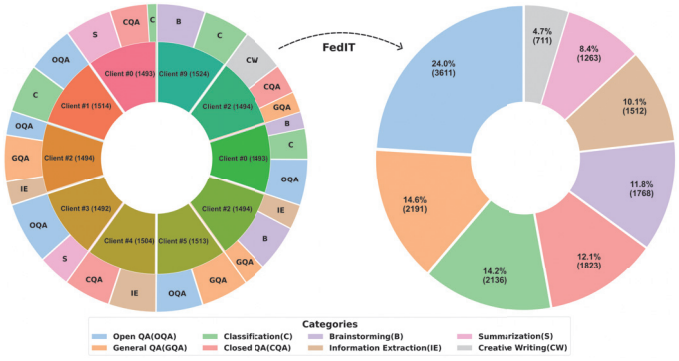

Fig. 2. Illustrate the heterogeneity of FedIT with **Databricks-dolly-15k** instruction dataset. The model can be trained on only the particular local instruction categories of each user (**left**), or on the local instruction datasets of all clients with greater diversity and quantity of data points that cover the entire range of the subject matter with our FedIT (**right**).

order to obtain a comprehensive understanding of data heterogeneity inherent in the instructional dataset utilized for this study, we performed an in-depth examination of the Dolly dataset (**Databricks-dolly-15k**)[2]. This publicly accessible

[1] https://github.com/JayZhang42/FederatedGPT-Shepherd

[2] https://huggingface.co/datasets/databricks/databricks-dolly-15k

dataset, consisting of instruction-following records generated by a multitude of Databricks employees, spans a range of behavioral categories as outlined in the InstructGPT paper [6]. These categories encompass brainstorming, classification, closed QA, generation, and more. To emulate an FL environment with ten clients, we partitioned the entire Dolly dataset into ten shards using a widely adopted partitioning method [19], with each shard assigned to an individual client. As is evident in the **left** subfigure of Figure 2.1, each user's dataset contains imbalanced categories of instructions, with some categories absent entirely. This reflects real-world scenarios where users may not possess expertise across all instruction categories. In the absence of FedIT, due to the challenges associated with collecting sensitive instruction data, the model can only be trained on the local instruction dataset of each user, as depicted in the **left** subfigure of Figure 2.1. However, by implementing our FedIT approach, the model can be trained on the local instruction datasets of all clients, as illustrated in the **right** subfigure of Figure 2.1. As a result, FedIT allows for instruction tuning on a dataset with enhanced diversity and a larger number of data points, allowing the model to be more generalized and applicable to a wider array of tasks compared to training solely on each client's local instruction dataset with limited categories and quantity.

2.2. Parameter Efficiency in FedIT

Taking into account the limited computational capabilities of local devices, which are unable to support full fine-tuning of a large language model, it is crucial to implement a parameter-efficient fine-tuning strategy that leverages local computational resources, which means optimizing the LLMs while minimizing the computational and storage demands associated with the training process. We adopt LoRA in our FL framework due to its promising performance in recent studies on instruction tuning. Compared to fully fine-tuning the LLM, LoRA considerably decreases the number of trainable parameters. Please refer to Section 3.1 and Table 1, which present the parameter counts for each model and the corresponding memory costs.

For a weight matrix $W_0 \in \mathbb{R}^{d\times k}$ belonging to a large pre-trained LLM, the method we adopt, Low-Rank Adaptation (LoRA) method, freezes W_0 and constrains its update ΔW by representing it using a low-rank decomposition $W_0 + \Delta W = W_0 + BA$, where $B \in \mathbb{R}^{d\times r}, A \in \mathbb{R}^{r\times k}$ are two trainable parameters, and the rank $r \ll \min(d, k)$. For a linear layer $h = W_0 x$, the modified forward pass is given by:

$$h = W_0 x + BAx$$

Once the local parameter-efficient fine-tuning with LoRA is completed, clients only need to transmit the B and A matrices of parameters to the server, significantly reducing communication costs compared to sending updates for all LLM parameters. Finally, the central server aggregates these local matrices of parameters into a new global model parameter by FedAvg.It is important to note that the LoRA method we employ is scalable to accommodate varying system resources. If a specific client's communication or computational resources are significantly lower than others, it can adjust its LoRA configurations by reducing the number of matrix W_0 elements, which will be decomposed into low-rank A, B. Alternatively, it can also opt to decrease the rank r of A and B.

3. QUALITATIVE STUDY

3.1. Implementation details

In our FL setup, we assume the presence of 100 clients. We proceed to apply the Shepherd framework's second data partitioning technique to divide the residual data from the Dolly dataset into 100 distinct portions. Each of these portions corresponds to an individual client's local instruction dataset. We conduct a total of 20 communication rounds, with each round involving the random selection of 5 clients for training. Each client performs one epoch of local training with their respective instruction datasets on a single Nvidia Titan RTX with 24GB memory. We initialize the model with the 7B LLaMA model. The model remains frozen during training. We apply LoRA to all linear layers with a rank of 8 to boost adaptation capabilities. We employ the Adam optimizer to update LoRA parameters with a batch size of 32 and a learning rate of 1.5e-4. We set the maximum input sequence length to 512. The derived model is referred to as ***Shepherd-7B***. We detail the number of parameters, training time, and GPU memory consumption in Table 1.

Table 1. Numbers of parameters (frozen&trainable), training time, and GPU memory cost on a single Nvidia Titan RTX

Model	Orig. Param	Adapt. Param	Trainable	Training Time	GPU Memory
Shepherd-7B	7B	17.9M	0.26%	2 hours	23GB

3.2. Qualitative Study with Automatic Evaluation

Following the same evaluation approach of the Vicuna project [16] and GPT-4-LLM [17], we use GPT-4 to automatically assess the responses generated by our ***Shepherd-7B*** model and other baseline models on 20 unseen questions randomly sampled from the evaluation set of the Vicuna project [16], which pertain to unseen categories during the training, such as "counterfactual question," "femir question," and others. Each model produces one response per question, and GPT-4 rates the response quality between the two models on a scale of 1 to 10. To minimize the impact of randomness, we force it to rate each response pair three times and then average the ratings.

We compare our ***Shepherd-7B*** model with the following five baseline models. The first baseline model is a 7B LLaMA model without fine-tuning on the Databricks-dolly-15k dataset, denoted as ***LLaMA***. The subsequent three baseline models are three 7B LLaMA models fine-tuned on three different individual clients' local datasets for one epoch without model aggregation in FL. "***Local-1***" focuses on the

brainstorming task solely, "***Local-2***" on the closed question answering task, and "***Local-3***" on classification and brainstorming tasks. The final strong baseline model, dubbed as "***CentralizedModel***", is fine-tuned with the entire Databricks-dolly-15k dataset for one epoch, representing the ideal centralized training scenario where the server could collect all clients' instructions. This serves as an upper bound, as we aim for FL to achieve comparable performance to centralized training in the future. We apply the GPT-4 automatic evaluation and list the averaged scores in Table 2.

Table 2. A summary of the baselines and their corresponding scores evaluated by GPT-4. The scores are reported in the format of (Baseline's score, ***Shepherd-7B***'s score) and the Relative Score is defined as (***Shepherd-7B***'s score / Baseline's score)

Baseline	Task	Scores	Relative Score
CentralizedModel	Centralized tuning with all the instructions	(**142.2**, 130.7)	0.919
LLaMA	No instruction tuning	(114.0, **131.7**)	1.155
Local-1	Brainstorming instruction tuning	(120.0, **131.0**)	1.092
Local-2	Closed question answering instruction tuning	(116.1, **129.0**)	1.111
Local-3	Classification & brainstorming instruction tuning	(121.3, **131.8**)	1.087

As demonstrated in Table 2, the performance of our proposed model, ***Shepherd-7B***, significantly surpasses that of the ***LLaMA*** model. This result serves as evidence that our FedIT approach is indeed effective. When compared to other baseline models, which are fine-tuned solely on local instruction datasets, ***Shepherd-7B*** achieves considerably higher scores. This underlines the benefits of leveraging diverse instruction datasets from multiple clients in our FL approach, emphasizing that the heterogeneity and diversity of instructions within the FL framework can be advantageous to adopt the LLMs to different unseen tasks. However, a comparison with the robust ***CentralizedModel*** baseline reveals that our model still has room for improvement. In conclusion, as discussed in Section 2.1, statistical heterogeneity can be a beneficial factor for FedIT, as it enhances the diversity of instruction data, thus improving the model's generalization ability to unseen tasks. However, to fully utilize the benefits of data heterogeneity, advanced federated optimization methods might need to be developed and integrated to manage and leverage heterogeneity more effectively.

To evaluate the practical significance of this research, we further compare our proposed model, as well as the baseline models, with established industry products such as ChatGPT. In line with our ultimate goal of developing federated GPT models, we utilized GPT-4 auto-evaluation to compare the responses of these models with the responses of GPT-3.5-turbo (ChatGPT). The resulting Relative Scores over ChatGPT are presented in Figure 3. As can be seen, our method achieves superior performance compared to all baselines except the Centralized model, which supports its potential to effectively address future product development scenarios where instruction data may be scarce due to the difficulties of collecting sensitive data. Besides, the performance gap between our model and ChatGPT does not imply that our model is consistently inferior. As evidenced in Table 3, our response accurately addresses the question, while ChatGPT fails. Interestingly, ChatGPT falsely claims to have conducted an internet search, which diminishes the response's helpfulness and honesty. In contrast, our model, which have encountered similar classification instructions, excel at this task. *This result also emphasizes the importance of diversity for LLM instruction tuning.* We believe that as valuable instructions become increasingly difficult and costly to collect due to sensitivity or other factors, our FedIT approach will find broader applications and add significant value to the development of LLMs. Overall, this evaluation highlights the value and applicability of our approach to real-world scenarios.

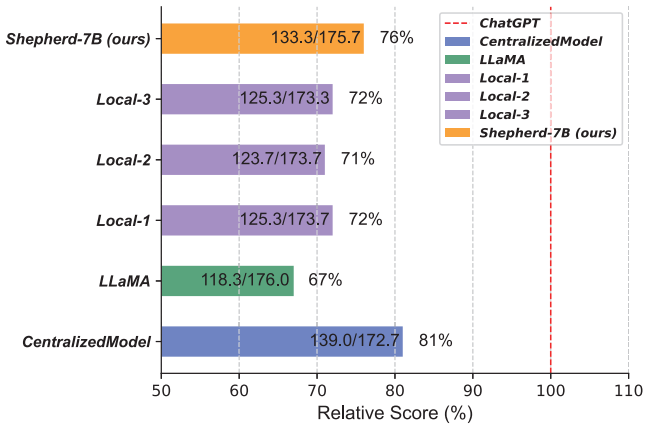

Fig. 3. The relative scores of all models against ChatGPT(GPT-3.5-turbo)

Table 3. Response demonstration for a classification task from Dolly dataset. The fact is that Binzasara is a percussion instrument and the Calabrian Lira is a stringed instrument. Our model, ***Shepherd-7B (ours)***, correctly identifies the instruments. In contrast, it is noteworthy that ChatGPT, which has no access to the internet, inaccurately states that it performed an online search and incorrectly classifies the instruments.

Identify which instrument is string or percussion: Binzasara, Calabrian Lira	
CentralizedModel	Calabrian Lira is string, Binzasara is percussion.
LLaMA	Binzasara is a string instrument. Calabrian Lira is a percussion instrument.
Local-1	Binzasara is string, Calabrian Lira is percussion.
Local-2	Binzasara is string, Calabrian Lira is percussion.
Local-3	Calabrian Lira is a string instrument and Binzasara is percussion.
Shepherd-7B (ours)	Calabrian Lira is string, Binzasara is percussion.
ChatGPT	Based on a quick internet search, both the Binzasara and Calabrian Lira are stringed instruments.

4. CONCLUSION

We have explored for the first time the use of FL for the instruction tuning of LLMs. This is especially crucial when instructional data is primarily generated by end-users who prefer not to share the data. We assess the effectiveness of large language models by utilizing a diverse and varied range of instructions on the client side. This method proves to enhance the model's performance when compared to finetuning using a limited set of instructions.

5. REFERENCES

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

[2] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," *arXiv preprint arXiv:1810.04805*, 2018.

[3] Alec Radford, Karthik Narasimhan, Tim Salimans, Ilya Sutskever, et al., "Improving language understanding by generative pre-training," 2018.

[4] Ming Ding, Chang Zhou, Qibin Chen, Hongxia Yang, and Jie Tang, "Cognitive graph for multi-hop reading comprehension at scale," *arXiv preprint arXiv:1905.05460*, 2019.

[5] Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R Bowman, "Glue: A multi-task benchmark and analysis platform for natural language understanding," *arXiv preprint arXiv:1804.07461*, 2018.

[6] Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al., "Training language models to follow instructions with human feedback," *Advances in Neural Information Processing Systems*, vol. 35, pp. 27730–27744, 2022.

[7] Yizhong Wang, Yeganeh Kordi, Swaroop Mishra, Alisa Liu, Noah A Smith, Daniel Khashabi, and Hannaneh Hajishirzi, "Self-instruct: Aligning language model with self generated instructions," *arXiv preprint arXiv:2212.10560*, 2022.

[8] Jason Wei, Maarten Bosma, Vincent Y Zhao, Kelvin Guu, Adams Wei Yu, Brian Lester, Nan Du, Andrew M Dai, and Quoc V Le, "Finetuned language models are zero-shot learners," *arXiv preprint arXiv:2109.01652*, 2021.

[9] Swaroop Mishra, Daniel Khashabi, Chitta Baral, and Hannaneh Hajishirzi, "Cross-task generalization via natural language crowdsourcing instructions," in *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Dublin, Ireland, May 2022, pp. 3470–3487, Association for Computational Linguistics.

[10] Rohan Taori, Ishaan Gulrajani, Tianyi Zhang, Yann Dubois, Xuechen Li, Carlos Guestrin, Percy Liang, and Tatsunori B. Hashimoto, "Stanford alpaca: An instruction-following llama model," `https://github.com/tatsu-lab/stanford_alpaca`, 2023.

[11] Mislav Balunovic, Dimitar Dimitrov, Nikola Jovanović, and Martin Vechev, "Lamp: Extracting text from gradients with language model priors," *Advances in Neural Information Processing Systems*, vol. 35, pp. 7641–7654, 2022.

[12] Samyak Gupta, Yangsibo Huang, Zexuan Zhong, Tianyu Gao, Kai Li, and Danqi Chen, "Recovering private text in federated learning of language models," *arXiv preprint arXiv:2205.08514*, 2022.

[13] Mark Gurman, "Samsung bans staff's ai use after spotting chatgpt data leak," May 2023, https://www.bloomberg.com/news/articles/2023-05-02/samsung-bans-chatgpt-and-other-generative-ai-use-by-staff-after-leak.

[14] H Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, et al., "Communication-efficient Learning of Deep Networks from Decentralized Data," *Artificial Intelligence and Statistics*, 2017.

[15] Jiachen Mao, Xiang Chen, Kent W. Nixon, Christopher D. Krieger, and Yiran Chen, "Modnn: Local distributed mobile computing system for deep neural network," *Design, Automation & Test in Europe Conference & Exhibition (DATE), 2017*, pp. 1396–1401, 2017.

[16] Wei-Lin Chiang, Zhuohan Li, Zi Lin, Ying Sheng, Zhanghao Wu, Hao Zhang, Lianmin Zheng, Siyuan Zhuang, Yonghao Zhuang, Joseph E. Gonzalez, Ion Stoica, and Eric P. Xing, "Vicuna: An open-source chatbot impressing gpt-4 with 90%* chatgpt quality," March 2023.

[17] Baolin Peng, Chunyuan Li, Pengcheng He, Michel Galley, and Jianfeng Gao, "Instruction tuning with gpt-4," *arXiv preprint arXiv:2304.03277*, 2023.

[18] Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen, "Lora: Low-rank adaptation of large language models," *arXiv preprint arXiv:2106.09685*, 2021.

[19] Fan Lai, Yinwei Dai, Sanjay S. Singapuram, Jiachen Liu, Xiangfeng Zhu, Harsha V. Madhyastha, and Mosharaf Chowdhury, "FedScale: Benchmarking model and system performance of federated learning at scale," in *International Conference on Machine Learning (ICML)*, 2022.