THEME ARTICLE: CONVERGED COMPUTING: A BEST-OF-BOTH WORLDS OF HPC AND CLOUD

Secure Federated Learning Across Heterogeneous Cloud and High-Performance Computing Resources - A Case Study on Federated Fine-tuning of LLaMA 2

Zilinghan Li, *University of Illinois at Urbana-Champaign, Urbana, IL, 61820, USA*

Shilan He, *University of Illinois at Urbana-Champaign, Urbana, IL, 61820, USA*

Pranshu Chaturvedi, *Argonne National Laboratory, Lemont, IL, 60439, USA*

Volodymyr Kindratenko, *University of Illinois at Urbana-Champaign, Urbana, IL, 61820, USA*

Eliu A Huerta, *Argonne National Laboratory, Lemont, IL, 60439, USA*

Kibaek Kim, *Argonne National Laboratory, Lemont, IL, 60439, USA*

Ravi Madduri, *Argonne National Laboratory, Lemont, IL, 60439, USA*

Abstract—Federated learning enables multiple data owners to collaboratively train robust machine learning models without transferring large or sensitive local datasets by only sharing the parameters of the locally trained models. In this paper, we elaborate on the design of our Advanced Privacy-Preserving Federated Learning (APPFL) framework, which streamlines end-to-end secure and reliable federated learning experiments across cloud computing facilities and high-performance computing resources by leveraging Globus Compute, a distributed function as a service platform, and Amazon Web Services. We further demonstrate the use case of APPFL in fine-tuning a LLaMA 2 7B model using several cloud resources and supercomputers.

Federated learning (FL) is a distributed machine learning paradigm where multiple data owners, referred to as clients, jointly train a machine learning model.[1–2] The process is orchestrated by a central server that only requires the transfer of locally trained model parameters and not the entire datasets. The server aggregates these model parameters and redistributes the updated parameters to the clients for further local training iterations. As FL could leverage diverse training data from multiple clients without explicitly collecting and storing distributed client datasets together as a centralized dataset, it is becoming an increasingly promising approach to train a more robust machine learning model and alleviate the domain shift problem without compromising the privacy of local training datasets.[3] FL is broadly categorized into two types, cross-device FL and cross-silo FL.[2] Cross-device FL involves a large number of unreliable devices, such as IoT or mobile devices, with only a small subset participating in each FL training round. On the other hand, cross-silo FL only has a few reliable clients, typically institutions or organizations equipped with powerful computing resources, including high-performance computing (HPC) systems or cloud computing facilities. This paper specifically focuses on the cross-silo FL settings.

The deployment and launch of cross-silo FL experiments face several key challenges, including the establishment of trust relationships among FL clients, inherent heterogeneity of client computing resources, and tedious coordination of the collaboration efforts. First, trust is paramount in FL to avoid data or model attacks, where a client might maliciously train the model using invalid data or send corrupted model parameters to the server. Second, the computing resources of

XXXX-XXX © 2024 IEEE
Digital Object Identifier 10.1109/XXX.0000.0000000

clients in a federation can vary widely in architecture, operating systems, job scheduling systems, and computing capabilities. Third, as cross-silo FL requires the participation of all clients in each training round, indicating the need for the simultaneous start of client training jobs to avoid resource wastage, it becomes more complex to coordinate the collaboration among multiple clients.

To overcome these challenges, we introduce the Advanced Privacy-Preserving Federated Learning (APPFL) framework which enables easy and streamlined setup of secure end-to-end cross-silo FL experiments. APPFL employs *Globus Compute* as its primary communication backbone for the distributed training process. *Globus Compute* is a distributed function-as-a-service platform that allows for the execution of remote functions on specified computing resources through the configuration of an endpoint.[4] *Globus Compute* endpoints, managed by clients, can be configured on a diverse range of computing systems to support local FL training. Once all client endpoints are configured, the FL server can easily initiate and orchestrate the FL training process by dispatching training tasks as needed. *Globus Compute* is also integrated with the Globus authentication service,[5] linking each FL client with an institutional or organizational Identity and Access Management services for identification. This facilitates building trust relationships among the clients. More details about *Globus Compute* can be found at https://funcx.readthedocs.io/en/latest/index.html. Additionally, the APPFL framework integrates various asynchronous FL algorithms for efficient training and resource utilization, especially beneficial in scenarios with significant disparities in the client computing capabilities. It also features advanced privacy-preserving algorithms to offer an extra layer of protection within the federation and further ensure the privacy of sensitive local data. The APPFL framework is aimed at enabling a wide array of domain experts to easily engage in creating secure federations and running FL experiments for various scientific applications.

APPFL FRAMEWORK

Figure 1 illustrates the process of federated learning using the APPFL framework. In this process, the FL server plays a pivotal role, orchestrating the training by iteratively dispatching training tasks to all FL clients and subsequently gathering results via the *Globus Compute* cloud server. As *Globus Compute* has limitations on the size of the parameters, the large model parameters are reliably exchanged via the

FIGURE 1. Overview of the federated learning process using the APPFL framework.

Amazon Simple Storage Service (S3). The combination of *Globus Compute* and Amazon S3 ensures a secure, robust, and smooth flow of tasks, information, and models between the server and clients. Computing machines, ranging from personal laptops to HPC clusters with varied job schedulers, as well as cloud virtual machines, are all capable of participating as FL clients by configuring a *Globus Compute* endpoint on it. The endpoint runs a daemon process in the background once started and only allocates necessary computing resources when there are pending tasks, especially if the computing resource uses a job scheduler. Consequently, these endpoints allocate resources only after the FL server launches the training by dispatching the initial tasks, thereby reducing resource wastage and minimizing the complexity of coordinating the distributed training. Each client computing machine also installs the APPFL software package, containing auxiliary codes for executing the dispatched local training tasks using the private local datasets. This setup highlights the versatility and adaptability of the framework to various computing environments, making it suitable for a wide range of FL applications.

Launching an FL experiment among various data owners using the APPFL framework involves a structured and secure process. The first step requires one participant to establish a Globus group, inviting other collaborating data owners through their institutional or organizational emails. This step is crucial for ensuring reliable identity and access management between the desired collaborators and FL clients, laying the foundation for an end-to-end trusted relationship. The created Globus group provides a layer of authorization for FL experiments. For the actual conducting of the FL experiment, any collaborator can volunteer to take on the role of the server. This involves gathering essential information from each client, such as the *Globus Compute* endpoint ID and a dataloader file for loading the local dataset. Once these elements are collected, the server utilizes the training script from the APPFL framework software package to initiate the training process by providing the information collected

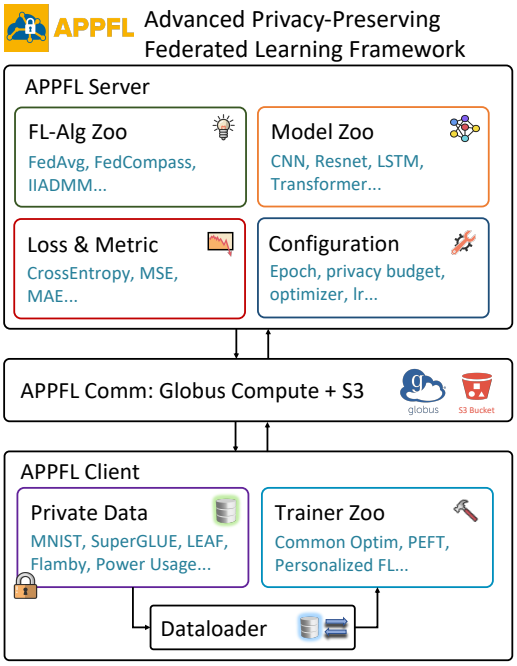

FIGURE 2. Modular design of the APPFL framework.

from clients, alongside the specified model architecture and training hyperparameters. All data owners gain access to the final model parameters at the end of the FL experiment to ensure that every participant benefits from the collaborative effort. Optionally, the experiment can be connected to the resources on Amazon Web Services that can be used to store training logs and results from various experiments along with training visualizations.

Figure 2 presents the modular design of the APPFL framework. The APPFL server consists of four parts, federated learning algorithms, machine learning model architectures for training, training loss functions and metrics, and training configurations. The APPFL server supports a range of FL algorithms, including widely-used synchronous algorithms like `FedAvg`,[1] advanced asynchronous algorithms such as `FedCompass`,[6] and privacy-preserving algorithms such as `IIADMM`.[7] For instance, asynchronous algorithms are beneficial for increasing the efficiency of the overall FL training and the utilization of the client computing resources in cases where there are large variations in the client computing capabilities, and privacy-preserving algorithms are helpful to further protect the local data privacy against gradient inversion attacks. The comprehensive support for various FL algorithms allows the APPFL framework to adapt to various FL scenarios and requirements. The framework incorporates several standard machine learning model architectures, including convolutional neural networks (CNN), residual neural networks (ResNet), long short-term memory networks (LSTM), and transformers. It also provides the flexibility for users to utilize custom model architectures for specific tasks. Similarly, for training loss functions and evaluation metrics, the APPFL server also offers both popular default options and the ability to use custom choices to accommodate a wide range of training scenarios. The training configuration component is for setting up necessary hyperparameters for the central aggregation and local training, and users can include their own configuration parameters when devising new aggregation or local training strategies.

On the client side, the APPFL client includes auxiliary trainers that facilitate model training on private local datasets using the provided dataloader. The dataloader, prepared by each client individually, contains the function for loading the local datasets on each client's computing resources and performing necessary pre-processing steps during the remote training executions. Multiple trainers are provided to support a variety of training tasks, from common training procedures to more specialized approaches such as parameter-efficient fine-tuning (PEFT) and personalized federated learning. Users can also customize their own local trainers for their applications. The APPFL communicator, operating on both the FL server and clients, ensures secure and seamless communication among the FL server and FL clients by using *Globus Compute* to dispatch training tasks and collect results, and AWS S3 buckets to transfer global and local model parameters.

The modularity of the framework simplifies customization for users, enabling them to conduct FL experiments on data from various domains, incorporate new machine learning model architectures, devise novel local training algorithms, and develop new FL server aggregation strategies. This flexibility makes the framework versatile and responsive to the evolving needs of users across various applications.

EXPERIMENTS

To demonstrate the effectiveness of the APPFL framework in streamlining FL experiments, we present a case study focusing on the application of APPFL in federated fine-tuning the LLaMA 2 7B,[8] a popular open-source pre-trained large language model (LLM) with approximately seven billion parameters, on the SuperGLUE natural language understanding benchmark.[9] Figure 3 illustrates the overview of the experiment. Each FL client operates on an individual

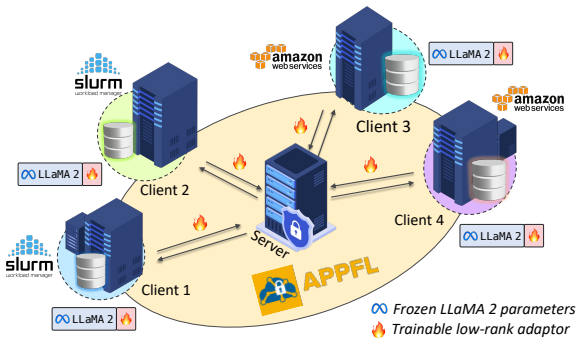

FIGURE 3. Overview of the federated large language model fine-tuning experiments among four heterogeneous clients on HPC nodes and cloud.

computing machine and accesses its local datasets. Each SuperGLUE dataset is partitioned into four client chunks in a non-independent and identically distributed manner following the *dual-Dirichlet partition* strategy introduced in `FedCompass`.[6] The strategy employs two Dirichlet distributions to simulate the distribution of sample classes within one client (with a concentration parameter $\alpha_1 = 2$) and the distribution of sample sizes across clients (with a concentration parameter $\alpha_2 = 8$), respectively. Figure 4 illustrates how local data is distributed among the four clients for various datasets within the SuperGLUE benchmark.

To circumvent transferring the gigabytes of parameters of LLM every training round, a parameter-efficient fine-tuning (PEFT) method, low-rank adaptation (LoRA),[10] is employed. LoRA freezes all parameters of the pre-trained LLM and only trains an additional set of rank decomposition matrices injected into each transformer layer, which substantially reduces the number of trainable parameters. Specifically, for LLaMA 2 7B, with decomposition matrices applied to all query and value matrices, a rank of 8, and a scaling factor of 32, LoRA results in a total of 16.0 MB (megabyte) trainable parameters being exchanged between the FL server and clients.

For all datasets in the SuperGLUE benchmark, each sample is transformed into the Stanford Alpaca prompt format.[11] Table 1 shows the detailed prompt instructions and inputs for each SuperGLUE dataset. The APPFL PEFT local trainer minimizes the cross-entropy loss for the labeled prompt outputs using the AdamW optimizer with a learning rate of 10^{-4} and a decay factor of 0.85.[12] `FedAvg` is used as the FL algorithm. The number of global communication rounds is set to 5 and the training batch size is set to 4 for all datasets. Given the varying characteristics of the datasets in the SuperGLUE benchmark, we have tailored the number of training batches of each training round and the maximum token length for each dataset in the training configurations, as detailed in Table 2. Notably, the term "All" for the batch number indicates that each client utilizes the entirety of available local training samples in every local training round.

To reflect real-world variability in computing resources, the four clients are operating on four heterogeneous computing machines. Specifically, two of these clients are deployed on HPC setups within the Delta supercomputer at the National Center for Supercomputing Applications (NCSA), using the Slurm job scheduler. These two differ in their GPU capabilities: one uses an NVIDIA A40 GPU, while the other employs an NVIDIA A100 Tensor Core GPU. The remaining two clients leverage Amazon Web Services (AWS) Elastic Compute Cloud (EC2) virtual machines with different specifications: one runs on a `g4ad.xlarge` instance and the other on a `g4ad.4xlarge` instance. This diverse computational setup provides a realistic testbed for the APPFL framework, demonstrating its applicability and adaptability in heterogeneous computing environments.

Table 3 presents a comparative analysis of the performance achieved by the LLaMA 2 7B model when fine-tuned under different settings: federated learning (`FL`), global training using centralized data (`Global`), and local training with client local corpus (`Local`). Since the labels for the SuperGLUE test datasets are not publicly available, the evaluation is based on the validation datasets from this benchmark. The results highlighted in the table reveal a notable pattern: models fine-tuned through FL outperform those fine-tuned locally on individual client data. This finding underscores the effectiveness of FL in enhancing model robustness. By leveraging the diverse local training corpora of various clients, FL manages to train more comprehensive models without the need for explicit data sharing. However, there remains a slight performance discrepancy when compared to models trained with centralized data, which likely arises from the inherent data heterogeneity across different clients. Additionally, the experiment showcases the adaptability of the APPFL framework in leveraging FL for a broad spectrum of applications, as well as the capability to facilitate FL experiments across a wide range of computing environments, from HPC nodes to cloud computing facilities, making it a valuable tool for conducting FL experiments in real-world settings.

This article has been accepted for publication in Computing in Science & Engineering. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/MCSE.2024.3382583

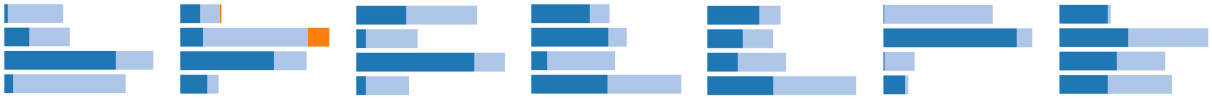

FIGURE 4. Local data distributions among four clients for the SuperGLUE datasets, where different colors indicate samples with different labels.

TABLE 1. Detailed Stanford Alpaca prompt instructions and inputs for the SuperGLUE datasets.

Dataset	Instruction	Input
BoolQ	The following reading comprehension question requires you to understand the following passage and answer a question related to the passage. Please answer with only "True" or "False" to the question: {sample['question']}?	sample['passage']
CB	Please determine whether the hypothesis "{sample['hypothesis']}" entails, contradicts, or is unrelated to the following premise: "{sample['premise']}". Please respond with either "Entailment", "Contradiction", or "Neutral".	N/A
COPA	Given the following premise, please determine whether Choice One, {sample['choice1']}, or Choice Two, {sample['choice2']}, is the {sample['question']} of the premise. Please respond with either "One" or "Two".	sample['premise']
MultiRC	Given the following paragraph, please determine whether "{sample['answer']}" is a correct answer to the question "{sample['question']}". Please respond with either "Yes" or "No".	sample['paragraph']
RTE	Please determine whether the sentence "{sample['premise']}" entails the hypothesis "{sample['hypothesis']}" or not. Please respond with either "Yes" or "No".	N/A
WiC	Please determine whether the word "{sample['word']}" is used in the same way in the following two sentences: "{sample['sentence1']}" and "{sample['sentence2']}" Please respond with either "Yes" or "No".	N/A
WSC	Please carefully read the following passages. For each passage, you must identify whether the pronoun marked in *bold* refers to the "quoted" noun.	sample['text']. \n Question: In the passage above, does the pronoun sample['span2_text'] refer to sample['span1_text']

TABLE 2. Number of training batches per local round and maximum token lengths for different datasets in the SuperGLUE benchmark.

Dataset	Batch number	Max token length
BoolQ	200	350
CB	All	350
COPA	All	300
MultiRC	200	600
RTE	200	200
WiC	200	200
WSC	All	220

CONCLUSION

In this paper, we introduce the design of the APPFL framework, a sophisticated software package to streamline the initiation and execution of secure and reliable end-to-end federated learning experiments across a diverse range of applications. This framework is adept at handling heterogeneous computing environments, from HPC systems to cloud-based resources. We showcase the framework's capabilities through a comprehensive case study, illustrating how APPFL can be seamlessly applied to the federated fine-tuning of

TABLE 3. Performance of the fine-tuned LLaMA 2 7B using federated learning (FL), global training, and local training.

Dataset	# Val. Samples	FL (%)	Global (%)	Local Avg (%)	Local (%)
BoolQ	3270	80.34	81.01	72.73	[69.72, 69.45, 77.80, 73.94]
CB	56	78.57	82.14	62.95	[46.43, 76.79, 67.86, 60.71]
BOPA	100	83.00	89.00	74.00	[76.00, 68.00, 70.00, 82.00]
MultiRC	4850	68.38	71.62	65.22	[70.54, 63.72, 61.80, 64.81]
RTE	227	87.36	85.28	84.66	[86.28, 85.20, 84.48, 82.67]
WiC	638	64.11	66.14	53.76	[59.87, 50.00, 55.64, 49.53]
WSC	104	72.12	75.96	64.36	[64.42, 68.27, 57.69, 63.46]

large language models using parameter-efficient fine-tuning methods. Looking to the future, there is potential for further enhancement via improving the quality and accessibility of FL-as-a-Service provided by APPFL. Our ultimate goal is to empower a broader range of domain experts from large institutions, universities, and national laboratories, to effortlessly conduct FL experiments in various AI applications, thus expanding the horizons of collaborative, privacy-preserving AI research and development.

ACKNOWLEDGMENTS

This material is based upon work supported by the U.S. Department of Energy, Office of Science, under contract number DE-AC02-06CH11357. This research is also part of the Delta research computing project, which is supported by the National Science Foundation (award OCI 2005572), and the State of Illinois. Delta is a joint effort of the University of Illinois at Urbana-Champaign and the National Center for Supercomputing Applications.

REFERENCES

1. B. McMahan, E. Moore, D. Ramage, S. Hampson, B. A. y Arcas, "Communication-efficient learning of deep networks from decentralized data," in *Artificial Intelligence and Statistics*, PMLR, 2017, pp. 1273–1282.
2. P. Kairouz, *et al.*, "Advances and open problems in federated learning," *Foundations and Trends in Machine Learning*, vol. 14, no. 1-2, pp. 1–210, 2021, doi:10.1561/2200000083
3. T.-H. Hoang, *et al.*, "Enabling end-to-end secure federated learning in biomedical research on heterogeneous computing environments with APPFLx," *arXiv preprint*, 2023, doi:10.48550/arXiv.2312.08701.
4. R. Chard, *et al.*, "FuncX: A federated function serving fabric for science," in *Proceedings of the 29th International Symposium on High-Performance Parallel and Distributed Computing*, 2020, pp. 65–76, doi:10.1145/3369583.3392683.
5. S. Tuecke, *et al.*, "Globus auth: A research identity and access management platform," in *2016 IEEE 12th International Conference on e-Science (e-Science)*, IEEE, 2016, pp. 203–212, doi:10.1109/eScience.2016.7870901.
6. Z. Li, *et al.*, "FedCompass: efficient cross-silo federated learning on heterogeneous client devices using a computing power aware scheduler," *arXiv preprint*, 2023, doi:10.48550/arXiv.2309.14675.
7. M. Ryu, Y. Kim, K. Kim, and R. K. Madduri, "APPFL: Open-source software framework for privacy-preserving federated learning," in *2022 IEEE International Parallel and Distributed Processing Symposium Workshops (IPDPSW)*, IEEE, 2022, pp. 1074–1083, doi:10.1109/IPDPSW55747.2022.00175.
8. H. Touvron, *et al.*, "Llama 2: Open foundation and fine-tuned chat models," *arXiv preprint*, 2023, doi:10.48550/arXiv.2307.09288.
9. A. Wang, *et al.*, "Superglue: A stickier benchmark for general-purpose language understanding systems," in *Advances in Neural Information Processing Systems*, vol. 32, 2019.
10. E. J. Hu, *et al.*, "LoRA: Low-rank adaptation of large language models," in *International Conference on Learning Representations*, 2022.
11. R. Taori, *et al.*, "Stanford Alpaca: An Instruction-following LLaMA model," 2023. [Online]. Available: https://github.com/tatsu-lab/stanford_alpaca
12. I. Loshchilov and F. Hutter, "Decoupled Weight Decay Regularization," in *International Conference on Learning Representations*, 2019.

Zilinghan Li is a Master of Science student in Computer Science at the University of Illinois at Urbana-Champaign and a research assistant at Argonne National Laboratory. His research interests include federated learning, distributed computing, and natural language processing. He received dual Bachelor's degrees from the University of Illinois at Urbana-Champaign and Zhejiang University. Contact him at zl52@illinois.edu.

Shilan He is a Master of Science student in Electrical and Computer Engineering at the University of

Illinois at Urbana-Champaign. Her research interests include reinforcement learning, federated learning, and communication networks. She received dual Bachelor's degrees from the University of Illinois at Urbana-Champaign and Zhejiang University. Contact her at shilanh2@illinois.edu.

Pranshu Chaturvedi is a research engineer in the Data Science and Learning Division at Argonne National Laboratory. He is interested in trustworthy AI, federated learning, and privacy preserving machine learning. He obtained his Bachelor's degree in Computer Science and Statistics from the University of Illinois at Urbana-Champaign. Contact him at pranshu.01.c@gmail.com.

Volodymyr Kindratenko is an Assistant Director at the National Center for Supercomputing Applications (NCSA) at the University of Illinois at Urbana-Champaign where he serves as the Director for the Center for Artificial Intelligence Innovation (CAII). He holds an Adjunct Associate Professor appointment in the Departments of Electrical and Computer Engineering (ECE) and a Research Associate Professor appointment in the Department of Computer Science (CS). Dr. Kindratenko received a D.Sc. degree from the University of Antwerp, Belgium. His research interests include high-performance computing, special-purpose computing architectures, and machine learning systems and applications. He is an IEEE senior member. Contact him at kindrtnk@illinois.edu.

Eliu A Huerta is the Lead for Translational AI and Computational Scientist in the Data Science and Learning Division at Argonne National Laboratory, and CASE Senior Scientist at the Department of Computer Science at the University of Chicago. He received a Master of Advanced Study in Applied Mathematics and Theoretical Physics and a Ph.D. in Theoretical Astrophysics from the University of Cambridge, United Kingdom. His research interests are at the interface of artificial intelligence, theoretical astrophysics, extreme scale computing, scientific visualization and mathematics. Contact him at elihu@anl.gov.

Kibaek Kim is a Computational Mathematician in the Mathematics and Computer Science Division at Argonne National Laboratory, and a Senior Scientist at-Large at the University of Chicago Consortium for Advanced Science and Engineering. His research focuses primarily on computational optimization, including stochastic programming and integer programming, with applications to complex systems. Dr. Kim obtained a Ph.D. degree in Industrial Engineering and Management Sciences from Northwestern University. He is an IEEE senior member. Contact him at kimk@anl.gov.

Ravi Madduri is a Computer Scientist in the Data Science and Learning Division at Argonne National Laboratory, and a Senior Scientist at-Large at the University of Chicago Consortium for Advanced Science and Engineering. His research interests are in AI for science, large-scale computation, and biomedical informatics. He received a Master's degree in Computer Science from Illinois Institute of Technology. Contact him at madduri@anl.gov.