Benchmarking Federated Learning Frameworks' Scalability

Gianluca Mittone[1,*], Samuele Fonio[1], Robert Birke[1] and Marco Aldinucci[1]

[1]Department of Computer Science, University of Turin, Turin, Italy

Abstract

Federated learning (FL) is a distributed machine learning paradigm allowing cooperative model training between multiple parties while maintaining local data privacy. FL can be deployed at various scales, ranging from thousands of low-end devices (e.g., smartphones) to just a few high-performance infrastructures (e.g., HPCs), raising critical concerns about the scalability of state-of-the-art FL frameworks. This preliminary study evaluates the scaling performance of a representative FL framework, i.e., Flower, in high-performance controlled environments, providing insights on such frameworks from a computational performance point of view. Two public Top500 pre-exascale HPC infrastructures are exploited to obtain reliable and comparable results: Leonardo and MareNostrum5. Our findings suggest that the design of current FL frameworks, and especially their communication backends, may overlook computational performance, leading to poor scaling in large-scale scenarios.

Keywords

Artificial Intelligence, Machine Learning, Federated Learning, Scalability, High-Performance Computing

1. Introduction

Federated learning (FL) [1] is a machine learning (ML) paradigm allowing the cooperative training of ML models while enforcing local data privacy. Such property is crucial when working with datasets protected by strict privacy policies, such as medical or financial data, which are firmly ruled by the General Data Protection Regulation (GDPR). Many communities actively investigate FL, from the ML one to the parallel and distributed systems ones [2]. Much research has been done on all its aspects, such as communication efficiency [3], model aggregation [4], and cross-facility deployment [5]. From a computational point of view, training state-of-the-art ML models is becoming increasingly demanding, with large language models (LLMs) being a well-known example [6]. In this regard, FL has been recently regarded as the future way to train LLMs by many works [7], resulting in an efficient strategy to scale computation while addressing privacy concerns. Many FL frameworks (FLFs) are available for deploying FL systems; most are open-source, and each one targets different situations, from local FL simulations to large-scale device ones. Despite many of them adopting very similar design choices, their different implementation can lead to very different computational performance [2]. This work focuses on the single-data-center scalability of a widespread, representative FLF (i.e., Flower) on two High-Performance Computing (HPC) infrastructures (i.e., Leonardo and MareNostrum5), providing valuable insights into the scaling performance of current FLFs design and implementation.

The contribution of this work is threefold and prone to further development. We *i*) present experimental results on the strong and weak scalability of Flower on two pre-exascale HPC infrastructures, *ii* discuss the design and implementation of current FLF, and *iii*) provide usability insights on the Top500 HPC infrastructures in Europe. Future works will delve into larger ML models, stressing the

BigHPC2025: Special Track on Big Data and High-Performance Computing, co-located with the 4th Italian Conference on Big Data and Data Science, ITADATA2025, September 9 – 11, 2025, Torino, Italy.

*Corresponding author.

gianluca.mittone@unito.it (G. Mittone); samuele.fonio@unito.it (S. Fonio); robert.birke@unito.it (R. Birke); marco.aldinucci@unito.it (M. Aldinucci)

https://alpha.di.unito.it/gianluca-mittone/ (G. Mittone); https://alpha.di.unito.it/samuele-fonio/ (S. Fonio); https://alpha.di.unito.it/robert-rene-maria-birke/ (R. Birke); https://alpha.di.unito.it/marco-aldinucci/ (M. Aldinucci)

0000-0002-1887-6911 (G. Mittone); 0009-0003-1870-4233 (S. Fonio); 0000-0003-1144-3707 (R. Birke); 0000-0001-8788-0829 (M. Aldinucci)

communication backend of current FLFs, exploiting other HPCs with different microarchitectures for a broader performance comparison, and testing the scaling of other major FLFs.

2. Methodology

This work provides insights into significant computation and communication bottlenecks of current FLFs. Thanks to abundant, powerful, and reliable hardware, HPC infrastructures allow obtaining sound and reliable performance measurements at different scales. At the same time, insights into the different HPC characteristics are gathered by observing how they behave under the same FL workload.

2.1. Computational Performance Metrics

In HPC, *strong scalability* refers to a system's capability to decrease the runtime of a fixed-size problem as additional computational resources (e.g., processors or nodes) are allocated. Achieving perfect strong scalability is often impeded by inter-process communication overhead, synchronization requirements, and resource contention, which introduce inefficiencies as resources scale. In this context, the strong scalability is evaluated by increasing the number of clients involved in the federation while keeping the global data pool size fixed. This enables the training workload (i.e., the dataset) to be partitioned across multiple processes, reducing training time as each client processes a smaller subset of the data. On the other hand, *weak scalability* measures the system's efficiency as workload and resources grow proportionally. In an ideal scenario, a system with good weak scalability should maintain consistent performance (e.g., constant runtime) as the workload per client remains steady, even as the total problem size and resources grow. In this context, the weak scalability is evaluated by increasing the number of clients involved in the federation while keeping the local data pool size fixed.

Although this approach does not reflect a real-world FL setting where data is found locally on each client and its size cannot be manipulated, it is still a valid representative and controlled stress test for the FLF's design and implementation, especially for its communication backend. Thus, learning metrics are not considered since these experiments are not representative of any meaningful ML problem. Furthermore, by isolating the communication component under maximum load conditions, insights into the system's handling of high-throughput communication demands can be gathered, which are essential for understanding its scalability limits.

2.2. HPC Infrastructures

In the following, the main information of each HPC used in the reported experiments is provided:

- *Leonardo* is a pre-exascale system hosted by CINECA, Italy. It provides a performance peak of 238.70 PFlop/s (LINPACK Rmax). It consists of 4992 liquid-cooled compute nodes interconnected through an NVIDIA Mellanox network, with Dragon Fly+, with a maximum bandwidth of 200Gbit/s between each pair of nodes. The Booster nodes used in the experiments are equipped with a custom BullSequana X2135 "Da Vinci" blade with an Intel Xeon 8358 Ice Lake CPU (32 cores, 2.6GHz), 512GB DDR4 RAM (8 x 64GB, 3200MHz), 4 NVIDIA custom A100 GPUs (64GB, HBM2e), and 2 NVIDIA InfiniBand HDR 2×100Gb/s network cards.
- *MareNostrum5* (accelerated partition, ACC) is a pre-exascale system hosted by BSC, Spain. It provides a performance peak of 175.30 PFlop/s (LINPACK Rmax). It comprises 1120 compute nodes interconnected through an Infiniband NDR fat-tree network topology. Each compute node is equipped with 2x Intel Sapphire Rapids 8460Y+ at 2.3GHz and 40 cores each (80 cores node), 512 GB of Main memory, using DDR5, 4x Nvidia Hopper GPU with 64 GB HBM2 memory, 480GB on NVMe storage, and 4x NDR200 (BW per node 800Gb/s).

At the time of writing, Leonardo ranks 9th in the Top500 HPC ranking, while MareNostrum5 ACC ranks 11th. Both HPCs are pre-exascale systems adopting the standard Intel+NVIDIA architecture, exposing a similar software stack despite the different hardware characteristics.

Table 1
Characteristics of the ResNet models used in the experiments. *Parameters* denote the number of weights (i.e., floating point numbers) contained by the model, *size* denotes the size in MB of each model when saved on the file system or communicated across the network, and *FLOPs* denotes the amount of GFLOPs required to process a single data batch (32×32 image).

Model	Parameters	Size (MB)	FLOPs (GFLOPs)
ResNet-18	11.7M	45	1.8
ResNet-34	21.8M	83	3.6
ResNet-50	25.6M	97	4.1
ResNet-101	44.5M	170	7.8
ResNet-152	60.2M	230	11.3

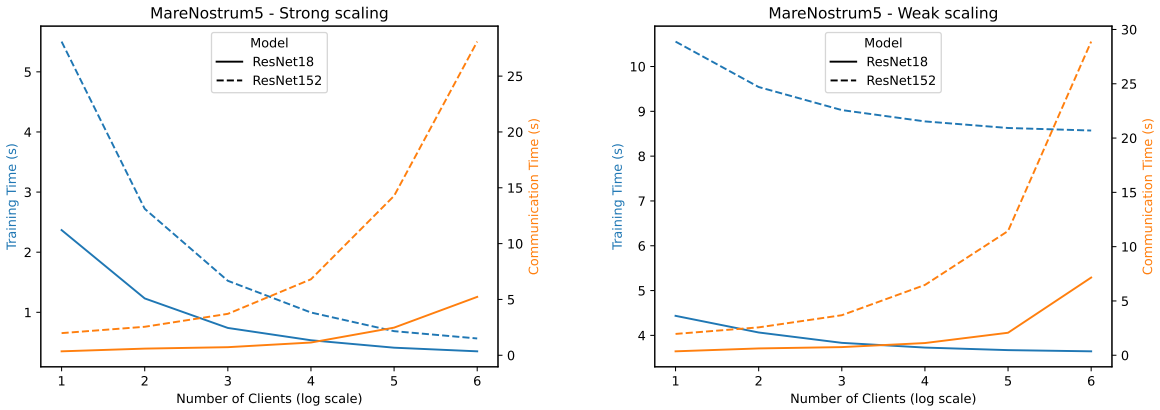
2.3. Flower: A Friendly Federated AI Framework

Flower [8] is a widespread FLF used for simulation and real-world deployment, widely accepted by the research community and industry. It is open-source, fully implemented in Python, exploits gRPC as a communication backend, and is based on a client-server architecture. Flower focuses on offering a smooth and straightforward user experience, which is in line with its design: the full Python implementation allows ML practitioners a fast and seamless interaction with the library, and the client-server architecture provides a well-known distributed framework with which the developer can easily interact. gRPC as a communication backend is sufficiently flexible to accommodate the dynamicity in client participation that is typical of FL. Furthermore, Flower exhibits better computational performance when compared with its competitors [9]. Flower's broad adoption, top-tier performance, and technical characteristics matching most state-of-the-art FLFs (e.g., Intel OpenFL, NVIDIA Flare, FedML) make it ideal as a representative FLF for this study.

3. Experiments and Results

Experiments focus on training the whole range of ResNets [10] available in PyTorch on the CIFAR-10 [11] dataset. ResNets range from the smallest, ResNet-18, to the largest, ResNet-152, providing increasingly heavier requirements from both the computational (more FLOPs required for the training) and communication (more bits to be transmitted) perspectives. It is worth mentioning that the kernel size for the skip connections was 1×1. Table 1 reports the different characteristics of the various ResNets used in the reported experiments. CIFAR-10 provides 50,000 32×32 pixel images for training and 10000 for testing, organized in 10 classes. The batch size used in the experiments is 256, and the results presented are the average over 5 rounds of communication for soundness, which constitute the entire training, with one epoch per round. The number of clients used increases by powers of 2, e.g., $2, 4, 8, 16, 32, 64$ clients. Federated averaging is used as an aggregation strategy, without using any outer optimizer; SGD is used as a local optimizer, and no learning rate schedule is set. Each client is provided with a full, exclusive computational node (as a consequence, with 4 GPUs). Such 1-to-1 mapping between clients and nodes is a crucial aspect of this study: by providing each client with more than enough resources to complete its ML task, we cannot encounter computing bottlenecks on the client side. At the same time, for simplicity, no intra-node distributed training approaches are adopted. In this way, each client exploits just one GPU, leaving the intra-node communication times and the different intra-node DNN training distribution techniques out of the equation, making the comparison between different infrastructures as fair as possible.

Figure 1a provides MareNostrum5 strong scalability results; only the smallest and biggest model results are plotted for readability. It is noticeable that the computing workload diminishes as the number of clients grows, as expected; however, it is interesting to see that communication soon becomes the bottleneck of the entire computation. It is worth noting that with 16 clients, the communication time equals the training time for ResNet18 and overcomes it for ResNet152. On the other hand, Figure 1b



(a) Strong scaling

(b) Weak scaling scenario

Figure 1: Computing and communication time per round for ResNet18 and ResNet152 by client count (power of 2) on MareNostrum5 ACC. Data points are obtained as the average of 5 runs.

Table 2
Compute (training) and communication times on MareNostrum5 ACC and Leonardo for the different ResNet models on 16 clients in a weak scalability scenario.

Model	MareNostrum5 ACC		Leonardo	
	Compute	Communication	Compute	Communication
ResNet-18	3.7278	1.1231	6.5661	7.5534
ResNet-34	4.0955	2.2041	7.1080	14.7686
ResNet-50	4.4388	2.5427	7.6506	16.4344
ResNet-101	6.4293	4.6070	9.4005	29.8678
ResNet-152	8.7750	6.4730	12.5656	41.9910

provides Marenostrum5 weak scalability results. While the training time varies slightly with a growing number of clients, the communication cost exhibits the same growing behaviour as the strong scaling scenario. This fact highlights a critical performance issue in communication management that is unrelated to the computational aspect and is directly linked to the number of clients participating in the training. Such issues may not be just found in Flower but also in similar FLFs.

The computing capabilities these experiments require are not particularly demanding for the underlying architectures, while the communication infrastructure is more stressed. Table 2 allows comparing the two HPC infrastructures from both points of view for the proposed use case. Results show that Marenostrum5's training time is generally lower, as each node has more powerful CPUs and GPUs. The communication time on Marenostrum5 also consistently outperforms Leonardo's, indicating better interconnection performances, which is expected given the different network topologies (fat-tree vs. DragonFly+) and interconnection bandwidth.

3.1. Conclusions

A performance measurement of a representative FLF (i.e., Flower), on two European pre-exascale Top500 HPC systems (Leonardo and MareNostrum5) is conducted. The same codebase is tested with growing model sizes and growing client populations. Experimental results highlight that current FLFs may overlook computational performance in their design, especially from the communication point of view, which can be particularly detrimental in large-scale deployments. However, to confirm this, a more comprehensive benchmark examining more FLFs is required, and an in-depth study of their communication backend is necessary to establish whether such issues are derived from software design or from the backend itself.

Acknowledgments

This work is partially supported by the EuroHPC-JU funding under grant No. 101093441, with support from the Horizon-EuroHPC-JU-2021-COE-01 (SPACE CoE), and by the Spoke "FutureHPC & BigData" of the ICSC - Centro Nazionale di Ricerca in "High-Performance Computing, Big Data and Quantum Computing", funded by the European Union - NextGenerationEU.

Declaration on Generative AI

The author(s) have not employed any Generative AI tools.

References

[1] B. McMahan, E. Moore, D. Ramage, S. Hampson, B. A. y Arcas, Communication-efficient learning of deep networks from decentralized data, in: Proceedings of the 20th International Conference on Artificial Intelligence and Statistics, AISTATS, volume 54 of *Proceedings of Machine Learning Research*, PMLR, 2017, pp. 1273–1282. URL: http://proceedings.mlr.press/v54/mcmahan17a.html.

[2] G. Mittone, N. Tonci, R. Birke, I. Colonnelli, D. Medic, A. Bartolini, R. Esposito, E. Parisi, F. Beneventi, M. Polato, M. Torquati, L. Benini, M. Aldinucci, Experimenting with emerging RISC-V systems for decentralised machine learning, in: Proceedings of the 20th ACM International Conference on Computing Frontiers, CF, ACM, 2023, pp. 73–83. doi:10.1145/3587135.3592211.

[3] C. Wu, F. Wu, R. Liu, L. Lyu, Y. Huang, X. Xie, Fedkd: Communication efficient federated learning via knowledge distillation, CoRR abs/2108.13323 (2021). URL: https://arxiv.org/abs/2108.13323. arXiv:2108.13323.

[4] S. P. Karimireddy, S. Kale, M. Mohri, S. J. Reddi, S. U. Stich, A. T. Suresh, SCAFFOLD: stochastic controlled averaging for federated learning, in: Proceedings of the 37th International Conference on Machine Learning, ICML, volume 119 of *Proceedings of Machine Learning Research*, PMLR, 2020, pp. 5132–5143. URL: http://proceedings.mlr.press/v119/karimireddy20a.html.

[5] I. Colonnelli, R. Birke, G. Malenza, G. Mittone, A. Mulone, J. Galjaard, L. Y. Chen, S. Bassini, G. Scipione, J. Martinovič, V. Vondrák, M. Aldinucci, Cross-facility federated learning, Procedia Computer Science 240 (2024) 3–12. doi:https://doi.org/10.1016/j.procs.2024.07.003, proceedings of the First EuroHPC user day.

[6] J. Sevilla, L. Heim, A. Ho, T. Besiroglu, M. Hobbhahn, P. Villalobos, Compute trends across three eras of machine learning, in: International Joint Conference on Neural Networks, IJCNN, IEEE, 2022, pp. 1–8. doi:10.1109/IJCNN55064.2022.9891914.

[7] L. Sani, A. Iacob, Z. Cao, B. Marino, Y. Gao, T. Paulik, W. Zhao, W. F. Shen, P. Aleksandrov, X. Qiu, N. D. Lane, The future of large language model pre-training is federated, CoRR abs/2405.10853 (2024). doi:10.48550/ARXIV.2405.10853. arXiv:2405.10853.

[8] D. J. Beutel, T. Topal, A. Mathur, X. Qiu, T. Parcollet, N. D. Lane, Flower: A friendly federated learning research framework, CoRR abs/2007.14390 (2020). URL: https://arxiv.org/abs/2007.14390. arXiv:2007.14390.

[9] S. Fonio, Benchmarking federated learning frameworks for medical imaging tasks, in: Image Analysis and Processing - ICIAP Proceedings, Part II, volume 14366 of *Lecture Notes in Computer Science*, Springer, 2023, pp. 223–232. doi:10.1007/978-3-031-51026-7_20.

[10] K. He, X. Zhang, S. Ren, J. Sun, Deep residual learning for image recognition, in: 2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR, IEEE Computer Society, 2016, pp. 770–778. doi:10.1109/CVPR.2016.90.

[11] A. Krizhevsky, Learning multiple layers of features from tiny images, 2009. URL: https://www.cs.toronto.edu/~kriz/learning-features-2009-TR.pdf.