Federated Learning meets HPC and cloud

Federated Learning meets HPC and cloud

Iacopo Colonnelli, Bruno Casella, Gianluca Mittone, Yasir Arfat, Barbara Cantalupo, Roberto Esposito, Alberto Riccardo Martinelli, Doriana Medić and Marco Aldinucci

Author's copy (preprint) of I. Colonnelli, B. Casella, G. Mittone, Y. Arfat, B. Cantalupo, R. Esposito, A. R. Martinelli, D. Medić, and M. Aldinucci, "Federated Learning meets HPC and cloud," in Astrophysics and Space Science Proceedings, Catania, Italy, 2023, p. 193–199. doi:10.1007/978-3-031-34167-0_39

Abstract HPC and AI are fated to meet for several reasons. This article will discuss some of them and argue why this will happen through the methods and technologies underpinning cloud computing. As a paradigmatic example, we present a new Federated Learning (FL) system that collaboratively trains a deep learning model in different supercomputing centers, where FL is both an AI workload and an enabling technology for the federation of supercomputers. The system is based on the StreamFlow workflow manager designed for hybrid cloud-HPC infrastructures.

1 Introduction

High-Performance Computing (HPC) and Artificial Intelligence (AI) are fated to meet for many reasons. First, HPC infrastructures embrace GPUs for their superior performance-per-watt ratio against general-purpose multicores. Second, the next-generation scientific workflows integrate AI-based steps for their accuracy in approximating and analyzing complex phenomena. Third, AI, specifically Machine Learning (ML), is a perfect workload for GPUs for required performance and development time. Today, we cannot close the circle of seamlessly running AI-enabled scientific workloads into HPC infrastructures, which system software and development tools are not designed for modern workloads, such as ML frameworks, typically designed for the cloud. HPC-cloud convergence is likely to bridge the gap.

Università degli Studi di Torino, Computer Science Dept. e-mail: `name.surname@unito.it`

This work takes Federated Learning (FL) as a paradigmatic example of modern AI applications. FL tackles the problem of collaboratively training an ML model using distributed data silos, where the data within silos cannot be exposed outside the infrastructure (and the owner-tenant) for privacy and secrecy concerns. FL is challenging because it requires federating infrastructures and orchestrating their workload managers (e.g., SLURM, PBS for HPC) cyclically. Indeed, training an FL model requires repeating several (possibly synchronous) rounds while exchanging information among infrastructures at each round. Moreover, the traditional HPC toolkit does not adequately support the design of this kind of application, as the FL workflow needs to express cyclic workflows spawning across different infrastructures (OS/architecture).

The following two sections recap how FL works (Sec. 2) and how hybrid workflows can be used to describe and orchestrate cloud-HPC workloads (Sec. 3). Then, Sec. 4 presents a hybrid workflow describing FL across different HPC clusters with a cloud-based aggregator.

2 Federated Learning

More and more companies are adopting AI models to support their daily decisions, thanks to the growing availability of big data and computing power. Data are often fragmented between different organizations, forming the so-called "data silos". For several reasons, including the increasingly strict privacy regulations (e.g., the European GDPR), merging these data to obtain large datasets is not feasible. *Federated Learning* [12] is a distributed ML technique adopted to address privacy and security concerns.

The first FL algorithm, proposed in 2016 by Google, is `FedAvg` [12], which works with Deep Neural Networks (DNNs). DNNs are still the most used models in FL, even if works training standard machine learning models in federated settings exist in literature [14]. A typical `FedAvg` setting has multiple clients and a central server. Each client takes one step of gradient descent on the current model using its local data, and the server performs a weighted average of the resulting models. It is possible to iterate the local gradient descent steps several times before the averaging step. However, the fact that data in different silos might not be assumed to be independent and identically distributed (IID) poses a key challenge to FL [10]. In the last few years, several FL algorithms (e.g., `FedCurv` [16]) have been proposed to tackle the problem of non-IIDness and a recent work [4] shows that tuning the number of local gradient descent steps might be key to attack this problem, as it empirically shows that often algorithms perform better when this number is increased.

The typical runtime architecture of FL frameworks (e.g., Intel OpenFL [15] and Flower [2]) is a master/worker. Each worker is deployed onto a different location owning a private portion of the federated dataset, where it trains a private copy of a DNN model. At the end of each training round, each worker sends the serialized state of its model to the master, which computes an aggregated model using a con-

figurable aggregation strategy and broadcasts it back to workers for the next round. Some recent FL frameworks drop the constraint of a single centralized aggregator, either relying on a tree-based infrastructure or implementing a fully decentralized peer-to-peer aggregation protocol. However, since the runtime infrastructure and the communication topology are always intrinsic characteristics of the framework implementation, users cannot seamlessly customize them.

3 Hybrid workflows

A *hybrid workflow* is a workflow whose steps can span multiple, heterogeneous, and independent computing infrastructures. Support for *multiple* infrastructures implies that each step can target a different execution *location*, which must have access to all the input dependencies. Locations can be *heterogeneous*, exposing different hardware resources and protocols for authentication, communication, resource allocation, and job execution. Plus, they can be *independent* of each other, meaning that direct communications among them may not be allowed.

Therefore, a hybrid workflow model should be *composite*, enclosing a *workflow graph* that describes a complex application in terms of steps and their dependencies, a *location topology* that represents a heterogeneous execution environment in terms of HW/SW resources and communication channels, and a *mapping relation* of each workflow step onto one or more execution locations. Each location can have two distinct roles. A *control* location belongs to the Workflow Management System control plane and can execute a command locally, offload it to other locations, or initiate a data transfer. Conversely, a *processing* location can only execute or delegate commands sent by other locations.

Hybrid workflows unify a description of software components' interactions (steps and dependencies) with a description of the execution environment (locations and channels) in a unique model while keeping the two planes well separated. The global knowledge of the execution environment allows a WMS to act as an orchestrator, managing locations' lifecycles and data transfers automatically without requiring modifications at the infrastructure level. Therefore, they represent the first step toward an *Internet of Workflows* [9], i.e., the ability to deploy workflows across multiple organizational and geographical boundaries.

As a practical example, they offer a way to federate HPC infrastructures at the application level, managing them as accelerators [6], i.e., offloading specific portions of complex workflows running elsewhere, from the user's workstation to general-purpose Cloud VMs. In addition, the explicit mapping of steps onto location allows for avoiding unnecessary data transfers, making hybrid workflows suitable for describing distributed applications where data movements are not allowed, like FL workloads.

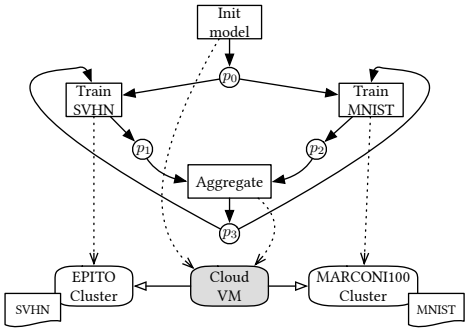

Fig. 1 Hybrid workflow model of an FL pipeline mapped onto a cloud+HPC topology. Steps are represented as squares, ports as circles, and dependencies as arrows with black-filled heads. Locations are represented as squashed rectangles and communication channels as arrows with white-filled heads. Control locations are colored in grey, while processing locations are left blank. Finally, mapping relations are expressed as dotted arrows.

4 Hybrid Federated Learning workflows

In research scenarios, cross-silo FL pipelines are prone to run on heterogeneous and independent execution environments. Indeed, data providers are usually independent entities with different data treatment protocols, storage infrastructures, and access policies. Unfortunately, this independence also leads to inconsistent data formats, which need ad-hoc preprocessing pipelines before joining an FL experiment. Moreover, data protection measures are likely to disallow direct inbound and outbound connections involving the data storage nodes.

Conversely, existing FL frameworks are designed to work with homogeneous infrastructures by deploying an agent on each worker node or relying on Function-as-a-Service (FaaS) frameworks to submit training commands. Most also require direct bidirectional interconnections between the aggregation control plane and the training nodes to exchange models' states directly. Representing an FL pipeline as a hybrid workflow can solve these problems.

This work evaluates hybrid workflows to execute a cross-cluster FL pipeline, where two independent HPC clusters train two different models on a private dataset and a Cloud VM acts as a centralized aggregator (see Fig. 1). The pipeline trains a VGG16 [17] with Group Normalization [18] over two datasets: a standard MNIST [11], residing on the CINECA MARCONI100 HPC facility located in Bologna (node: 2x16-core IBM POWER9 AC922, 256GB RAM and 4 NVIDIA V100 GPUs), and a grayscaled version of SVHN [13], residing within HPC4AI [1] facility located in Torino (node: 80-core Arm Neoverse-N1, 512GB RAM, 2 NVidia A100 GPU). The same DNN and datasets have been tested as a baseline in an article [3] showing that the aggregation of parameters at test time is a novel form of Transfer Learning.

The FL pipeline has been described using an extension of the Common Workflow Language (CWL) open standard [8], a vendor-agnostic language based on declarative

Table 1 Accuracy and time to solution for all the evaluated FL pipelines. OpenFL has not been evaluated on the Hybrid Cloud+HPC environment because it does not support this configuration, as it requires direct bidirectional communications between the aggregator and each worker node.

		StreamFlow			OpenFL		
		MNIST acc.	SVHN acc.	Time	MNIST acc.	SVHN acc.	Time
Cloud	100 rounds, 1 epoch/round	99.36%	92.74%	2h40m	97.91%	93.15%	3h06m
	50 rounds, 2 epochs/round	99.37%	92.74%	2h20m	98.88%	94.21%	2h09m
Hybrid	100 rounds, 1 epoch/round	99.29%	93.06%	2h57m	–	–	–
	50 rounds, 2 epochs/round	99.34%	92.85%	1h45m	–	–	–

YAML files. The official CWL standard does not include iterative patterns, which are fundamental to expressing the FL training process. However, a `Loop` construct has recently been proposed as an extension[1]. As far as the authors know, the present work describes the first real-case iterative CWL workflow. At runtime, the StreamFlow framework [7] has been used to orchestrate the workflow over the hybrid cloud+HPC infrastructure described above.

Two FL configurations have been tested: 100 rounds of 1 epoch each and 50 rounds of 2 epochs each. Both used the `FedAvg` aggregation strategy. In addition, to compare performances with a baseline, the same configurations have been tested on a pure cloud version of the pipeline, in which two worker VMs has executed the training steps (8 cores, 32GB RAM, 1 NVIDIA T4 GPU each) using both the Intel OpenFL framework [15] and the StreamFlow pipeline (see Table 1). The two frameworks are comparable in terms of accuracy and time-to-solution when running on cloud VMs, meaning that general-purpose hybrid workflows are not "too general" to provide adequate performance. The overhead introduced by the remote cross-HPC executions (mainly due to queue times and transfers over low-performing networks) is largely compensated by the computing power and high-end interconnections provided by HPC facilities, and the benefits increase with heavier jobs (i.e., with more epochs per round, but also with larger datasets and models).

5 Conclusion and future work

This work investigates the potential of hybrid workflow models to realize Federated Learning (FL) systems across heterogeneous infrastructures with limited connectivity between worker nodes (e.g., supercomputing modules). To the authors' knowledge, the reported experiment is the first usage of FL across supercomputers. Also, it suggests an effective way to federate supercomputing centers at the application level. The overhead introduced by using a general-purpose workflow system does not significantly hinder execution time. A drawback is the need to execute each round in each supercomputer in a single bulk. Some finer-grained alternatives will

[1] https://github.com/common-workflow-language/common-workflow-language/issues/495

be explored in the future, like Jupyter-based literate hybrid workflows [5]. The code used for experimental evaluation is publicly available[2].

Acknowledgements This article describes work undertaken in the context of the ACROSS project, "HPC Big Data Artificial Intelligence Cross Stack Platform Towards Exascale" (EuroHPC-01-2019, G.A. n. 955648), and The European PILOT project, "Pilot using Independent Local & Open Technologies" (EuroHPC-JU funding under grant no. 101034126, with support from the Horizon2020 programme). This work has also been supported by CINECA Italian national supercomputing center (ISCRA grant cod. HP10CHRZE7).

References

1. Aldinucci, M., Rabellino, S., Pironti, M., Spiga, F., Viviani, P., et al.: HPC4AI, an AI-on-demand federated platform endeavour. In: ACM Computing Frontiers (2018)
2. Beutel, D.J., Topal, T., Mathur, A., Qiu, X., Parcollet, T., Lane, N.D.: Flower: A friendly federated learning research framework. CoRR **abs/2007.14390** (2020)
3. Casella, B., Chisari, A.B., Battiato, S., Giuffrida, M.V.: Transfer learning via test-time neural networks aggregation. In: Proceedings of the 17th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications, VISIGRAPP 2022, Volume 5: VISAPP, 2022. SCITEPRESS, Setúbal (2022)
4. Casella, B., Esposito, R., Cavazzoni, C., Aldinucci, M.: Benchmarking fedavg and fedcurv for image classification tasks. In: Proceedings of the 1st Italian Conference on Big Data and Data Science, ITADATA 2022, 2022, CEUR Workshop Proceedings. Aachen (2022)
5. Colonnelli, I., Aldinucci, M., Cantalupo, B., Padovani, L., Rabellino, S., et al.: Distributed workflows with Jupyter. Future Generation Computer Systems **128**, 282–298 (2022)
6. Colonnelli, I., Cantalupo, B., Esposito, R., Pennisi, M., Spampinato, C., Aldinucci, M.: HPC Application Cloudification: The StreamFlow Toolkit. In: PARMA-DITAM workshop 2021, *Open Access Series in Informatics (OASIcs)*, vol. 88, pp. 5:1–5:13. Dagstuhl, Germany (2021)
7. Colonnelli, I., Cantalupo, B., Merelli, I., Aldinucci, M.: StreamFlow: cross-breeding cloud with HPC. IEEE Transactions on Emerging Topics in Computing **9**(4), 1723–1737 (2021)
8. Crusoe, M.R., Abeln, S., Iosup, A., Amstutz, P., Chilton, J., et al.: Methods included: Standardizing computational reuse and portability with the common workflow language. Commun. ACM **65**(6), 54–63 (2022)
9. Dubé, N., Roweth, D., Faraboschi, P., Milojicic, D.S.: Future of HPC: the internet of workflows. IEEE Internet Comput. **25**(5), 26–34 (2021)
10. Kairouz, P., McMahan, H.B., Avent, B., Bellet, A., Bennis, M., et al.: Advances and open problems in federated learning. Found. Trends Mach. Learn. (2021)
11. LeCun, Y., Bottou, L., Bengio, Y., Haffner, P.: Gradient-based learning applied to document recognition. Proc. IEEE **86**(11), 2278–2324 (1998)
12. McMahan, B., Moore, E., Ramage, D., Hampson, S., y Arcas, B.A.: Communication-efficient learning of deep networks from decentralized data. In: Proc. of the 20th Intl. Conference on Artificial Intelligence and Statistics, AISTATS 2017, *Proc. of Machine Learning Research*, vol. 54, pp. 1273–1282. PMLR, Fort Lauderdale, FL, USA (2017)
13. Netzer, Y., Wang, T., Coates, A., Bissacco, A., Wu, B., Ng, A.Y.: Reading digits in natural images with unsupervised feature learning. In: NIPS Workshop on Deep Learning and Unsupervised Feature Learning 2011 (2011)
14. Polato, M., Esposito, R., Aldinucci, M.: Boosting the federation: Cross-silo federated learning without gradient descent. In: Intl. Joint Conference on Neural Networks, IJCNN 2022, Padua, Italy, 2022. IEEE, New York (2022)

[2] `https://github.com/alpha-unito/streamflow-fl`

15. Reina, G.A., Gruzdev, A., Foley, P., Perepelkina, O., Sharma, M., Davidyuk, I., et al.: Openfl: An open-source framework for federated learning. CoRR **abs/2105.06413** (2021)
16. Shoham, N., Avidor, T., Keren, A., Israel, N., Benditkis, D., et al.: Overcoming forgetting in federated learning on non-iid data. CoRR **abs/1910.07796** (2019)
17. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. In: 3rd Intl. Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, 2015, Conference Track Proceedings (2015)
18. Wu, Y., He, K.: Group normalization. In: Computer Vision - ECCV 2018 - 15th European Conference, Munich, Germany, Proceedings, Part XIII, *Lecture Notes in Computer Science*, vol. 11217, pp. 3–19. Springer, Heidelberg (2018)