The Case for Decentralized AI Metadata Tracking and Lineage in Science and Engineering Workflows

Annmary Justine, Aalap Tripathy, Martin Foltin, Suparna Bhattacharya, Sergey Serebryakov, Paolo Faraboschi

Hewlett Packard Labs, Hewlett Packard Enterprise, Spring TX 77389, USA

`{annmary.roy, aalap,tripathy, martin.foltin`
`suparna.bhattacharya, sergey.serebryakov, paolo.faraboschi}@hpe.com`

Abstract

We developed a de-centralized, peer-to-peer AI metadata framework that enables end-to-end metadata & lineage tracking for distributed Machine Learning pipelines spanning edge, High Performance Computing and cloud environments. It enables reproducibility, audit trail, AI model provenance, and incremental model development in environments that may suffer from high latencies, intermittent connectivity & disparate security policies, common in Science use cases. We describe innovations that enable merging of metadata from disconnected sites without a central coordinator and across independently executed steps, and discuss benefits of this new open source framework for example Science use cases.

Introduction

Complex Artificial Intelligence (AI) workflows involve multiple stages with different needs from compute, storage and network and can span multiple sites. In experimental science use cases, it is often desirable to perform part of the analysis at the edge, close to where the data is collected. This allows insights in real time for better control of experiments or quick response to anomalies, to reduce the volume of data transmitted to central servers, and support large interoperable distributed systems. AI inference at the edge can help dynamically steer instruments for more precise or relevant observations in weather monitoring, microscopy, medical imaging, nuclear fusion, etc. [1]. It can help early responders during dynamic extreme weather events such as wildfires and tornados [2]. AI model training is more compute intensive and typically performed in the cloud or at HPC clusters with the possibility of cloud bursting. Consequently, the workflow can span three or four domains: edge, HPC cluster, private cloud (such as, for example, the National Research Platform [2]) and public cloud.

Tracking of AI model and data provenance is essential for AI pipeline reproducibility, trustworthiness, explainability, incremental improvements, reuse and collaborative development. AI metadata tracking solutions available in the industry (MLFlow[3], Weights & Biases[4], AIM[5] etc.) adopt a centralized approach suitable for enterprise models, where data acquisition, model training and model inference are performed in the cloud or at enterprise data centers. This approach is inadequate for heterogeneous workflows representing complex usage patterns that span multiple domains with different compute and data access characteristics as i) they have very limited abstractions to model composite workflows and track their interdependencies, and, ii) do not support the exchange of metadata. As shown in Figure 1, the usage patterns may involve AI inference at the edge for low latency (Type A) or with inference in the datacenter (Type B) both types potentially informing deeper analysis (Type 1), or being used in conjunction with simulations in the loop (Type 2). Recurrent pipelines with model re-training in HPC cluster/cloud (Type 3) optionally complemented by light re-training at the edge (Type 4) also need to be supported.

In this paper, we propose Federated Common Metadata Framework (CMF), a solution that supports all these usage patterns and addresses the weaknesses of centralized approaches by tracking and storing metadata and data locally at the site and distributing required metadata which can then be merged with metadata from other sites to provide lineage and provenance tracking. It decouples the data management from the metadata management, thereby enabling sharing of only the required data and ensuring data privacy.

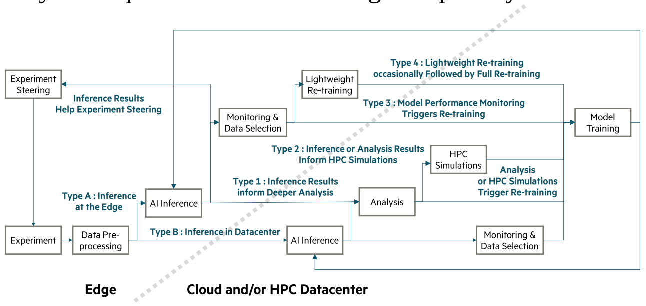
Experiment Steering
Inference Results Help Experiment Steering
Monitoring & Data Selection
Lightweight Re-training
Type 4 : Lightweight Re-training occasionally Followed by Full Re-training
Type 3 : Model Performance Monitoring Triggers Re-training
Model Training
Type 2 : Inference or Analysis Results Inform HPC Simulations
HPC Simulations
Analysis or HPC Simulations Trigger Re-training
Type 1 : Inference Results inform Deeper Analysis
Type A : Inference at the Edge
AI Inference
Analysis
Experiment
Data Pre-processing
Type B : Inference in Datacenter
AI Inference
Monitoring & Data Selection
Edge
Cloud and/or HPC Datacenter

Figure 1: Example use patterns for AI edge data analysis

Our key contributions in Federated CMF include: 1. Ability to create lineages & associate metadata from independently executed steps (from disassociated sites) without needing the order of steps to be defined before. This is done by identifying each artifact with its content hash and associating it with execution as input or output. Recursive querying of this information enables the creation of lineage chains for artifacts and executions. 2. Ability to merge metadata from disconnected sites without a central coordinator enabled through the hierarchical organization of metadata. 3. Ability to manage metadata and data separately, e.g. by storing metadata in SQLite and data in an Artifact store and having pointers in metadata store to the location of Artifacts.

Federated CMF Design

The key components are explained below (See Fig 2).

Metadata Store: Federated CMF builds upon CMF library [6] which enables tracking of metadata for AI pipeline through implicit & explicit metadata tracking API's. CMF builds upon ML-Metadata [7] library to store and organize metadata.

The metadata store records pipeline metadata through hierarchical abstractions: "Pipeline", "Context" and "Execu-

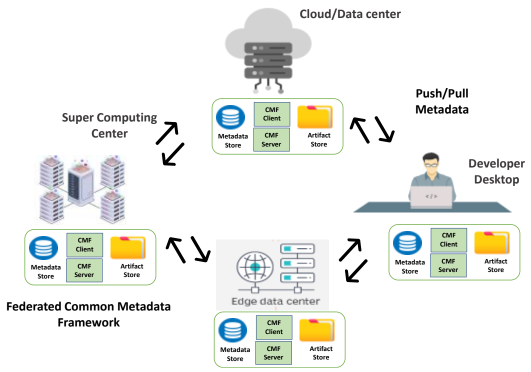

Figure 2: Federated Common Metadata Framework

tion". An Execution is the smallest unit of tracking a "pipeline" (an end-to-end experiment or workflow). A unique execution is created for every distinct experiment. There can be multiple executions for a single stage in a "pipeline" with different hyper parameters, model architectures or artifact versions. Executions of the same type (e.g., training stage) are grouped under a single Context. There could be multiple such Contexts corresponding to preprocessing, training, featurization, stages etc. Multiple contexts are grouped under a Pipeline. Pipeline and Context entities in CMF are singletons. Pipelines and contexts are identified by unique names and are only created once in a metadata store. Attempting to recreate a previously existing Pipeline or Context will incrementally update them. Artifacts used or produced in an Execution are associated with it as input or output properties respectively. This hierarchical tree representation (*Pipeline*→ *Context* → *Execution*) in CMF is the key to tracking pipelines with multiple stages in different sites. By traversing this pipeline tree, Federated CMF framework can build lineages for Artifacts and Executions across disjointed executions.

Artifact Store: Federated CMF needs a mechanism to uniquify artifacts. We build upon data version control e.g. (DVC) [8] which provides this capability for artifact management. The physical files (HDF5, parquet etc.) are stored in an Artifact store. Through integration with DVC [8], we can support different kinds of storage remotes like HTTPS, SSH, Amazon S3, Google Cloud Storage, HDFS, or a local filesystem making it suitable for edge and datacenter use cases. We support both shared and unique Artifact store for users. This is because the metadata store indexes an artifact's unique content-hash and unique URI.

CMF supports external artifacts and their metadata via the mechanism of uniform resource identifiers (URIs). External artifacts are artifacts whose life cycle is not managed by CMF. When such an external artifact needs to become part of a CMF lineage graph, CMF creates a derived artifact with its URI encoding the location of the source artifact. CMF uses the URI's scheme field to identify the artifact owner, and the remaining fields (authority, path, query and fragment) are used to uniquely locate the artifact within the owner's name space. For instance, for MLFlow-managed ML model artifacts, the following URI template is used: `mlflow:///runs/RUN_ID`, where `RUN_ID` is a unique MLFlow run identifier. For AIM-managed artifacts, the URI template is `aim:///runs/RUN_HASH`. Similarly, CMF encodes other artifact types, such as datasets and results of hyperparameter search experiments.

CMF Server: Any site within the Federated CMF can function as the server by running the command "*cmf server up*". This creates a rest endpoint that can be accessed over HTTPS and used by other clients to push or pull metadata. On the startup of the server, a token is generated. Connecting clients should pass the same token to be able to push and pull. Our intent here is simplicity: this model does not preclude more rigorous authentication/authorization systems to be employed over this scheme. When the server receives metadata from authorized clients, it verifies the payload & merges the metadata. By design in a metadata store pipeline names are unique and under a pipeline, context names are unique. The merge step identifies the branch in the pipeline tree under which an incoming execution fits or creates a new context branch or creates a new pipeline tree. Artifact metadata from different sites can be merged using its content hash as the joining key.

CMF Client: It provides commands to push or pull metadata to/from the server. The pulled metadata is stored in the local store. It also enables the push and pull of subsets of metadata. Out of the multiple executions (experiments) to find the optimum model the researcher may be interested only in the execution which produced the best result. Further, compression techniques are employed to make this subset exchange between client-server bandwidth optimized. CMF client also enables the export of metadata from the cmf metadata store in JSON format to a flat file. The JSON format follows the OpenLineage [9] specifications for compatibility and can be imported into Federated CMF server or in GUI tools like Marquez [10] which support open lineage format to visualize the extracted metadata and lineage. "*cmf artifact pull*" enables pulling artifact from the remote store to the local store selectively. The design enables sharing of data when needed, but each site can use its local artifact store enabling data locality.

In summary Federated CMF enables: 1. Different sites to work independently and merge their metadata after completion and iterate on workflows. 2. It enables bandwidth optimized transfer of only relevant metadata. 3. It eliminates the need for a central coordinator (or quorum). 4. Enables data privacy by managing and sharing data and metadata separately. Many distributed scientists working at each domain can get near real-time view of metadata from each other employing CMF Clients/Servers described earlier.

Modern multi-site AI for Science Workflows

We outline four science workflows, which motivates the need for a federated approach to metadata management with reference to Fig. 1 (workflow type).

AI integrated real-time fire management (Type A1&2): Realtime fire management involves multiple intertwined workflows distributed across different sites and in heterogeneous compute resources [2]. Detection of fire at the edge [12], triggers a fire modeling workflow at the supercomputing facility to predict the growth and spread of fire. Fire growth model is fed with real-time measurements from the field on the perimeter of fire, the weather conditions etc., to dynamically adjust parameters of the model (post-fire retraining). This heterogenous workflow is fed with data from diverse sources like sensors, weather forecast centers, camera images etc. with different formats, different storage backends time series database or filesystem. With Federated CMF, the data can reside at the best suited artifact repository while independently tracking references to the data.
Magnetic Confinement Fusion (Type A4&B3): Attaining self-heating plasma in Tokamak rings is a key step towards harnessing clean fusion energy. Understanding & optimizing plasma characteristics is the primary goal of experiments at DIII-D National Fusion Facility [12]. Each plasma discharge experiment produces tens of GB of data that are fed to multilayer analysis (increase to TBs on future ITER Tokamak). Results from rapid analysis inform configuration for the next discharge. For example, Magnetic Equilibrium Reconstruction accelerated by AI (EFIT-AI) – is required within 5-7 minutes window between experiments. New approaches to improve resolution & quantify uncertainties will require significantly higher compute resources not available at fusion facilities. Federated CMF is architected to enable development of reproducible & extensible pipelines that include i) training of an ensemble of AI models at HPC facilities, ii) concurrent deployment of simplified models at fusion facility (the edge) for real time analysis & complex models at HPC for deep insight, iii) monitoring of model performance and lightweight retraining at the edge after new data & iv) occasional full-scale model retraining at HPC center.

Autonomous Electron Microscopy & Computational Steering (Type A2&3): Advances in microscopy now make it possible to record convergent beam electron diffraction (CBED) pattern at every scan position of a scanning transmission electron microscope creating GB/s of 4D-STEM data. AI/ML methods are increasingly used to process this 4D-STEM data to quantify the 3D atomic structure of materials [13]. New CBED images are converted to atomic structure descriptors using pretrained neural networks (*i*). These networks are typically trained (*ii*) at a HPC facility using simulated CBED pattern data. However, since the initial scan is sparse a Bayesian optimization estimate (*iii*) is generated through a material energetics simulation of the imaged region to reduce variability in the area imaged by the microscope. Therefore, the next set of scan positions is chosen to steer the next CBED imaging run (*iv*). Federated CMF is built to support such decentralized, tightly coupled computational processes (*i-iv*) in different infrastructure domains (edge and HPC facility or cloud) in tight unison [14].
High Energy Physics Particle Tracking Pipeline (Type A4): Deep learning-based pattern recognition methods are now regularly used for High energy Physics (HEP) particle tracking [15]. The leading techniques involve successive filtering of detector impact cloud points from sensors in collider experiments to isolate possible trajectories of exotic particles. This involves metric learning using fully connected multi-layer perceptron's, binary classifier for edge filtering and a graph neural network (GNN) to improve purity of selected points. This work faces numerous challenges: large datasets, large parameter space with cascaded inputs/outputs making optimization difficult, distributed algorithm development, multiple compute environments (HPC, on-prem & cloud). Federated CMF has been applied in such a scenario to capture metadata for the entire experiment population by capturing data lineage, metrics, network architecture, hyperparameters.

Early Results

We applied Federated CMF to high energy physics particle trajectory reconstruction pipeline (ExatrkX [15]) that has six stages including data pre-processing and two neural network models, where outputs from one model become inputs of another model. Figure 3 shows the parallel coordinate plot describing dependence of output metrics (Efficiency, Fake Rate, Duplication Rate) on selected hyper-parameters (Train-*) from different model training experiments. The figure shows improvement of metrics after the base model (Orange) trained in HPC facility has been further tuned with additional data at an edge facility (Purple). Note that only one hyper-parameter (*TrainGNN*) affecting the final pipeline stage has been tuned at the edge. Tracking of lineage from different pipeline stages enable CMF to associate an

execution output as the input for another execution and build end to end execution lineages from disjointed executions distributed between different sites.
In this case, it associates output from the model 1 trained in HPC facility with input of the model 2 tuned at the edge.
We have also assessed the performance of metadata merges, for high energy physics particle tracking pipeline as shown in Figure 4. The results show the time taken to merge 170

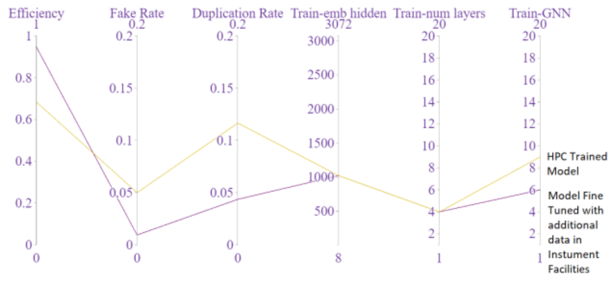

Figure 3: Parallel coordinate plot: Exa.trkX model training

and 90 executions are in the order of few seconds whereas their execution times are 2550 mins and 1350 mins respectively. The exact time depends on the existing metadata in

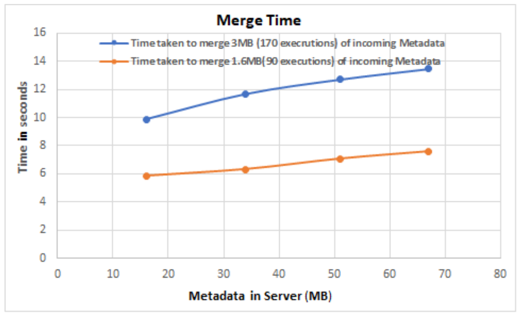

Figure 4: Metadata merge time in CMF Server

server. The results indicate that the time taken for merges (for a payload size) is comparable as we increase the size of existing metadata in the server. This makes it scalable to merge data to servers having sizeable metadata accumulated from incremental merges from across different sites or workflows. Code for Federated CMF is available at [16].

Related Work

MLFlow [3], Weights and Biases [4], and DataFed [17] enable synchronization of metadata from experimental & computational facilities to a central server and their presentation via a linked data portal. In contrast, Federated CMF enables peer to peer exchanges, bandwidth optimal transfers and merging of metadata independently at any site. Metadata can be exported/imported between servers making it ideal for shared compute scenarios. Although Workflow systems like Kepler scientific workflow [18] provides metadata tracking, the tracked metadata is trapped inside the system, making it difficult to correlate with metadata outside of its ecosystem. In contrast Federated CMF can track metadata & artifacts as derived metadata enabling FAIR lineage across heterogeneous workflow systems.

Conclusion & Future Work

We present the approach and early results to manage metadata of heterogeneous AI workflows. The results are promising, and we intend to deploy this in heterogeneous science workflows to further optimize them.

References

[1] Stevens, Rick, et. al. AI for Science: Report on the DOE Town Halls on Artificial Intelligence (AI) for Science. United States: N. p., 2020. Web. doi:10.2172/1604756.

[2] Ilkay Altintas, Building cyberinfrastructure for translational impact: The WIFIRE example, Journal of Computational Science,Volume 52,2021,101210

[3] MLFlow Tracking, https://www.mlflow.org/docs/latest/tracking.html.

[4] Weights & Biases, https://wandb.ai/site/experiment-tracking, Accessed 10/12/2022

[5] AIM, https://aimstack.io/#features,

[6] A. Justine et.al., Self-learning Data Foundation for Scientific AI, SMC-2022 proceedings, Springer CCIS series, in print

[7] ML Metadata https://www.tensorflow.org/tfx/guide/mlmd, Accessed 11/7/2022

[8] Data Version Control https://dvc.org/doc

[9] OpenLineage JsonSchema, https://github.com/OpenLineage/OpenLineage/blob/main/spec/OpenLineage.md

[10] Marquez, https://marquezproject.github.io/marquez/

[11] Dewangan, Anshuman, et al. “FIgLib & SmokeyNet: Dataset and Deep Learning Model for Real-Time Wildland Fire Smoke Detection”. Remote Sens. 2022, 14(4), 1007.

[12] Humphreys, David, Kupresanin, A., et. al Advancing Fusion with Machine Learning Research Needs Workshop Report. US: N. p., 2020. Web. doi:10.1007/s10894-020-00258-1

[13] Mukherjee, Debangshu, et al. "A roadmap for edge computing enabled automated multidimensional transmission electron microscopy." arXiv preprint arXiv:2210.02538 (2022).

[14] Al-Najjar, Anees, et al. "Enabling Autonomous Electron Microscopy for Networked Computation and Steering." arXiv preprint arXiv:2210.09791 (2022).

[15] Ju, Xiangyang, Murnane, Daniel, Calafiura, Paolo, et. al. 2021. "Performance of a geometric deep learning pipeline for HL-LHC particle tracking". United States. https://doi.org/10.1140/epjc/s10052-021-09675-8.

[16] Federated CMF, https://github.com/HewlettPackard/cmf/tree/federated_cmf

[17] DataFed System Overview, https://ornl.github.io/DataFed/system/overview.html , Accessed 10/14/2022

[18] Kepler, https://kepler-project.org/