Orchestrating Hierarchical Federated Learning Pipelines with the AIoTwin Middleware

Ivan Čilić
University of Zagreb, Faculty of Electrical Engineering and Computing
Zagreb, Croatia
ivan.cilic@fer.unizg.hr

Ana Petra Jukić
University of Zagreb, Faculty of Electrical Engineering and Computing
Zagreb, Croatia
ana-petra.jukic@fer.hr

Katarina Vuknić
University of Zagreb, Faculty of Electrical Engineering and Computing
Zagreb, Croatia
katarina.vuknic@fer.unizg.hr

Ivana Podnar Žarko
University of Zagreb, Faculty of Electrical Engineering and Computing
Zagreb, Croatia
ivana.podnar@fer.unizg.hr

Abstract

This tutorial introduces participants to the practical deployment of Hierarchical Federated Learning (HFL) across the Cloud-Edge-IoT (CEI) continuum. It explains the concept of Federated Learning and its extension to a hierarchical setup, as well as the orchestration aspects of setting up and running HFL pipelines in CEI environments. The tutorial includes a hands-on session which is divided into two parts. In the first part, participants will use the Flower framework to set up and execute standard and hierarchical Federated Learning (FL) tasks, providing hands-on insight into FL without orchestration. The second part focuses on the AIoTwin orchestration middleware, demonstrating how it enables seamless setup and monitoring of HFL pipelines across heterogeneous edge, fog, and cloud environments. The middleware automates the deployment of FL services (clients, aggregators) and handles infrastructure changes affecting the pipeline and global model performance, such as clients joining or leaving a running FL task. Participants are guided through deploying their own custom FL tasks on a real-world distributed cluster, while exploring the middleware's capabilities for orchestration, resource management, and hierarchical coordination. The tutorial is ideal for researchers and practitioners interested in deploying and running edge-to-cloud FL tasks at scale.

The tutorial was presented as part of the 15th International Conference on the Internet of Things (IoT 2025), held on November 18, 2025, in Vienna, Austria. Website: https://iot-conference.org/iot2025/workshops-tutorials/.

Keywords

edge computing, Federated Learning, orchestration, MLOps

1 Introduction and Motivation

IoT systems are inherently distributed: devices are spread across multiple locations and different administrative domains, often using edge computing devices in their vicinity. Therefore, IoT environments are well-suited for deploying Federated Learning (FL) [2] pipelines to create new machine learning (ML) models based on the generated IoT data, while preserving privacy by avoiding the transfer of raw and sensitive data to the cloud. FL pipelines enable model training in local networks in a distributed manner and close to IoT data sources by utilizing nearby edge devices for local training on *client nodes*. However, traditional FL pipelines, which rely on a single central server for aggregation of local models (*cloud aggregator*), face significant limitations when scaled across the CEI continuum: (1) *Communication overhead*: The centralized architecture with a single aggregator incurs high communication costs as each client must send model updates to the cloud aggregator; (2) *Data heterogeneity*: Statistical variation in data across clients leads to poor model convergence; and (3) *Dynamicity of CEI environments*: Traditional FL lacks native mechanisms to handle varying client resources, fluctuating bandwidth, and frequent node failures inherent to edge environments.

To overcome obstacles (1) and (2), **Hierarchical Federated Learning** (HFL) has emerged [1]. HFL introduces an intermediate layer of edge servers to aggregate local client updates before communicating with the global cloud aggregator. This multi-tier architecture can significantly improve the communication efficiency, making HFL pipelines ideal for CEI deployments.

To address obstacle (3) and complex setups of HFL pipelines, we propose the **AIoTwin Framework for Adaptive Orchestration of (H)FL Pipelines**, which supports dynamic adaptation to changes in the CEI continuum. The AIoTwin approach goes beyond static pipeline deployment and requires continuous runtime monitoring of both infrastructure health (e.g., node failures, network conditions) and ML performance of a running HFL pipeline (e.g., model accuracy, convergence rate). By integrating an orchestrator, the system can autonomously make informed decisions and coordinate reconfiguration actions—such as dynamic client-to-aggregator reassignment—to maintain high Quality of Service (QoS) and maximize model performance within defined constraints (e.g., communication cost).

The AIoTwin orchestration middleware supports HFL deployments at scale without requiring manual configuration and management of the computing infrastructure, i.e., edge and cloud nodes. The middleware, built on Kubernetes, automates the deployment of HFL services (clients and aggregators) across the available infrastructure and responds to infrastructure changes that affect FL

performance. In other words, the middleware is adaptive and can reconfigure an HFL pipeline at runtime based on the triggers which are either infrastructure-related (e.g., node failure), or related to ML performance (e.g., increase in training loss).

2 AIoTwin HFL Orchestration Middleware: Architecture and Mechanisms

The AIoTwin middleware for adaptive orchestration of HFL pipelines extends the two main components of a general-purpose orchestrator such as Kubernetes: orchestrator and node. The main role of the orchestrator is to manage the FL pipeline, while nodes execute FL-specific services, clients and aggregators. The FL orchestrator extends the general-purpose orchestrator with two new components: *FL Controller*, designed to control the FL pipeline, and *FL Configuration*, responsible for determining the best-fit pipeline configuration for a given environment. Given the benefits and complexity of HFL, the orchestration is primarily designed to support HFL setups, but it also supports flat FL setups.

In a typical (H)FL setup, an ML engineer defines an FL task with the following inputs: (i) an initial ML model, (ii) training parameters, and (iii) the orchestration objective. The initial ML model serves as the starting point from which all clients begin their training during the initialization phase of the pipeline. The training parameters include batch size, learning rate, and similar settings. The orchestration objective can be defined on a case-by-case basis, for example, to either optimize the model's performance within a predefined cost budget or to minimize the cost while achieving a specific performance target, such as a target level of accuracy or loss. Clients need to prepare their training datasets, perform local training, and update their local models based on received aggregated model updates.

In particular, a typical HFL pipeline is executed through the following phases:

(1) *Initialization phase*: Clients prepare their local data for training, and the initial ML model is distributed by the global aggregator (GA) to all clusters and, in turn, to the clients.
(2) *Local training*: Clients train their local models for a predefined number of epochs.
(3) Local aggregation: After local training, clients send their model updates to their dedicated local aggregator (LA) for aggregation. The aggregated model is distributed back to the clients in the cluster.
(4) *Global aggregation*: After a predefined number of local aggregations, usually called local rounds, the edge-aggregated model is sent to the GA for global aggregation. Subsequently, the global model is distributed back to the LAs which forward it to the clients in their cluster. This marks the beginning of a new global round.
(5) *Model convergence and completion*: When the model has converged, or a certain threshold has been reached, the pipeline execution is completed.

3 Tutorial Objectives

This tutorial offers a comprehensive, two-part introduction to building and orchestrating scalable FL pipelines. Participants first learn the fundamentals of standard flat FL using a widely adopted FL framework (Flower[1]) and then use our extensions to setup hierarchical FL deployments exclusively with Flower. In the second part, they apply advanced techniques for adaptive orchestration using the open-source tools from the AIoTwin project, built on top of Kubernetes.

The tutorial is designed for researchers, students, and practitioners interested in distributed ML, edge computing, and service orchestration. Participants should be familiar with Python and have a basic understanding of ML concepts (e.g., neural networks, loss functions). Prior experience with PyTorch is beneficial. All required software (Flower, AIoTwin components) is open-source and provided in pre-configured virtual environments or containers set up by the tutorial organizers.

Learning Outcomes: Upon completion, participants are able to: (1) Set up a standard FL pipeline for model training using the Flower framework. (2) Understand and implement the key architecture of HFL (client, local aggregator, global aggregator). (3) Apply advanced methods for dynamic and adaptive orchestration of HFL pipelines.

4 Tutorial Structure

The tutorial is organized and structured into two parts and includes the following three tasks focusing on practical and hands-on lessons:

(1) **Task 1**: Standard Federated Learning Setup with Flower
(2) **Task 2**: Hierarchical Federated Learning with AIoTwin Flower Wrapper
(3) **Task 3**: Orchestrating HFL pipelines with AIoTwin Middleware

The first part relates to the usage of the Flower framework with flat and hierarchical pipelines[2]. The goal of **Task 1** is to establish a working baseline for distributed model training and understand the limitations of a flat FL architecture. Participants use the Python-based Flower framework (version 1.7.0) and the widely-used CIFAR-10 dataset. Participants execute the aggregator and multiple clients, observe the aggregation process (e.g., FedAvg), and track the global model's convergence rate and communication requirements.

In **Task 2**, we transition the flat FL setup into a multi-tier HFL architecture using the AIoTwin Extension of the Flower Framework (*fl-service*). The implementation of the FL service is available in the following repository: https://github.com/AIoTwin/fl-service. In this setup, participants configure three distinct roles: global aggregator, local aggregator and client. Participants execute a tiered deployment, observing how the local aggregation step reduces the communication load on the cloud server and also track the global model's convergence rate.

In the second part of the tutorial, we present the framework for adaptive orchestration of HFL pipelines and explain the need for pipeline reconfiguration during runtime. We also introduce our original reconfiguration validation algorithm (RVA) [3] which evaluates pipeline reconfiguration after a specific number of global rounds following a reconfiguration. The target is to maximize the ML model accuracy within a specified communication cost budget.

[1] Flower Labs GmbH, "Flower A Friendly Federated Learning Framework," https://flower.ai/
[2] Instructions for Task 1 and 2 are specified here: https://github.com/AIoTwin/fl-service/tree/tutorial.

https://doi.org/10.34749/3061-1008.2025.10

Building on the HFL architecture deployed in Task 2, **Task 3** provides a hands-on exercise for adaptive HFL pipeline management using the AIoTwin Framework for Adaptive Orchestration of FL Pipelines (*fl-orchestrator*), published as open-source software in the following repository: https://github.com/AIoTwin/fl-framework. Participants explore how configuration files (e.g., YAML) define the list of nodes and their roles, and start an HFL pipeline within a predefined cluster. Participants are divided into teams, and each team uses eight pre-configured virtual machines for one HFL task/pipeline. They can also set up an experiment with dynamic changes of the infrastructure: a new client joining a running pipeline[3].

Furthermore, we demonstrate how orchestration logic can be implemented to optimize an objective (e.g., maximize model accuracy) while satisfying a constraint (e.g., communication cost budget), mirroring real-world operational requirements of CEI deployments. More details on how an orchestration objective can be tailored to maximize model performance within a predefined communication budget can be found in [3].

5 Conclusion

This tutorial provides a unique and highly relevant opportunity to gain practical experience with HFL and its critical orchestration challenges in the context of real-world deployments in the CEI continuum. Through hands-on experimentation with the established Flower framework and the novel, open-source AIoTwin orchestration middleware, participants can move beyond theoretical understanding to actively implement adaptive, resilient, and communication-efficient distributed ML solutions. The skills gained from configuring, deploying, and managing HFL pipelines are directly applicable to developing next-generation AIoT systems that operate reliably in heterogeneous and dynamic real-world environments while shaping the future of decentralized intelligence.

6 Acknowledgments

This work has been supported by the Horizon Europe WIDERA program under the grant agreement No. 101079214 (AIoTwin). Ana Petra Jukić was supported by the Croatian Science Foundation under project no. DOK-NPOO-2023-10-1182.

References

[1] Lumin Liu, Jun Zhang, S.H. Song, and Khaled B. Letaief. 2020. Client-Edge-Cloud Hierarchical Federated Learning. In *ICC 2020 - 2020 IEEE International Conference on Communications (ICC)*. 1–6. doi:10.1109/ICC40277.2020.9148862

[2] Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Aguera y Arcas. 2017. Communication-Efficient Learning of Deep Networks from Decentralized Data. In *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics (Proceedings of Machine Learning Research, Vol. 54)*, Aarti Singh and Jerry Zhu (Eds.). PMLR, 1273–1282. https://proceedings.mlr.press/v54/mcmahan17a.html

[3] Ivan Čilić, Anna Lackinger, Pantelis A. Frangoudis, Ivana Podnar Žarko, Alireza Furutanpey, Ilir Murturi, and Schahram Dustdar. 2025. Reactive Orchestration for Hierarchical Federated Learning Under a Communication Cost Budget. In *2025 IEEE International Conference on Machine Learning for Communication and Networking (ICMLCN)*. 1–7. doi:10.1109/ICMLCN64995.2025.11140562

[3] Instructions for Task 3 are provided here: https://github.com/AIoTwin/fl-orchestrator/blob/iot-conf-tutorial/tutorial/README.md.

https://doi.org/10.34749/3061-1008.2025.10