TEMA

TRUSTED
EXTREMELY PRECISE
MAPPING AND PREDICTION
FOR EMERGENCY
MANAGEMENT

D3.3

Report on Real-Time Federated Analytics

Project Information

Project acronym:	**TEMA**
Project full title:	Trusted Extremely Precise Mapping and Prediction for Emergency Management
Call identifier:	HORIZON-CL4-2022-DATA-01
Type of action:	HORIZON Research and Innovation Actions
Start date:	1 December 2022
End date:	30 November 2026
Grant agreement no:	101093003

D3.3- Report on Real-Time Federated Analytics			
Executive Summary:	**Deliverable D3.3 - Report on real-time federated analytics. It is the third deliverable of Work Package 3 within the TEMA project. The document reports the research results of Task T3.4 over the M13-M36 period of the project.**		
WP:	**3**		
Author(s):	**See table below for a full list of authors**		
Editor:	**Massimo Vilari**		
Leading Partner:	**UNIME**		
Participating Partners:	**ENG**		
Version:	**1.0**	Status:	**Final**
Deliverable Type:	**R Document, report**	Dissemination Level:	**Public**
Official Submission Date:	**30 November 2025**	Actual Submission Date:	**30 November 2025**

Funded by the European Union

This project has received funding from the European Union's HE research and innovation programme under grant agreement No 101093003

Disclaimer

The TEMA consortium consists of the following partners:

No.	Partner Organization Name	Partner Organization Short Name	Country
1	ARISTOTELIO PANEPISTIMIO THESSALONIKIS	AUTH	GR
2	DEUTSCHES ZENTRUM FUR LUFT UND RAUMFAHRT EV	DLR	DE
3	ENGINEERING - INGEGNERIA INFORMATICA SPA	ENG	IT
4	ATOS IT SOLUTIONS AND SERVICES IBERIA SL	ATOS IT	ES
4.1	ATOS SPAIN SA	ATOS SP	ES
5	UNIVERSIDAD DE SEVILLA	USE	ES
6	TECNOSYLVA SL	TSYL	ES
7	NORTHDOCKS GMBH	ND	DE
9	THE LISBON COUNCIL FOR ECONOMIC COMPETITIVENESS ASBL	LC	BE
10	LATITUDO 40 SRL	LAT40	IT
11	NELEN & SCHUURMANS TECHNOLOGY BV	NS	NL
11.1	NELEN & SCHUURMANS CONSULTANCY BV	NS C	NL
12	FRAUNHOFER GESELLSCHAFT ZUR FORDERUNG DER ANGEWANDTEN FORSCHUNG EV	FHHI	DE
13	UNIVERSITA DEGLI STUDI DI MESSINA	UNIME	IT
14	KAJAANIN AMMATTIKORKEAKOULU OY	KAMK	FI

Funded by the European Union

This project has received funding from the European Union's HE research and innovation programme under grant agreement No 101093003

16	KENTRO MELETON ASFALEIAS	KEMEA	GR
17	DIMOS MANTOUDIOU - LIMNIS - AGIAS ANNAS	D.MALIAN	GR
18	REGIONE AUTONOMA DELLA SARDEGNA	RAS	IT
19	BAYERISCHES ROTES KREUZ	BRK	DE
20	KAINUUN HYVINVOINTIALUE	KAHY	FI
21	INTERDISCIPLINARY TRANSFORMATION UNIVERSITY	IT:U	AT

Funded by the European Union
This project has received funding from the European Union's HE research and innovation programme under grant agreement No 101093003

Document Revision History

Version	Description	Contributions
0.1	ToC	Massimo Villari, Lorenzo Carnevale, Antonio Filograna, Francesco Arigliano
0.2	First draft	Massimo Villari, Lorenzo Carnevale, Antonio Filograna, Francesco Arigliano
1.0	Final version ready for submission.	Massimo Villari, Lorenzo Carnevale, Antonio Filograna, Francesco Arigliano

Authors

Name	Partner
Massimo Villari	UNIME
Lorenzo Carnevale	UNIME
Antonio Filograna	ENG
Francesco Arigliano	ENG

Reviewers

Name	Partner
Jose Ramiro Martínez de Dios	USE
Abdalraheem Ijjeh	USE

Funded by the European Union
This project has received funding from the European Union's HE research and innovation programme under grant agreement No 101093003

Table of Contents

Funded by the European Union

This project has received funding from the European Union's HE research and innovation programme under grant agreement No 101093003

Funded by the European Union
This project has received funding from the European Union's HE research and innovation programme under grant agreement No 101093003

Funded by the European Union

This project has received funding from the European Union's HE research and innovation programme under grant agreement No 101093003

List of Figures

Funded by the European Union
This project has received funding from the European Union's HE research and innovation programme under grant agreement No 101093003

Funded by the European Union

This project has received funding from the European Union's HE research and innovation programme under grant agreement No 101093003

List of Tables

Funded by the European Union

This project has received funding from the European Union's HE research and innovation programme under grant agreement No 101093003

List of Terms and Abbreviations

Abbreviation	Meaning
AI	Artificial Intelligence
ARIMA	Autoregressive Integrated Moving Average
BM	Business Mission
CB	Context Broker
CPU	Central Processing Unit
D2.2	Deliverable 2.2
D3.3	Deliverable 3.3
DDPG	Deep Deterministic Policy Gradient
DNN	Deep Neural Network
FL	Federated Learning
GPU	Graphics Processing Unit
I/O	Input/Output
IoT	Internet of Things
JSON-LD	JavaScript Object Notation Linked Data
KPI	Key Performance Indicator
LSTM	Long-Short Term Memory
ML	Machine Learning
NDM	Natural Disaster Management
NLP	Natural Language Processing
NGSI-LD	Next Generation Service Interface with Linked Data
OA4	Objective A4
OC1	Objective C1
OC4	Objective C4
PPO	Proximal Policy Optimization
RBAC	Role-Based Access Control
RNN	Recurrent Neural Network
SAREF	Smart Applications REFerence ontology
SLA	Service Level Agreement
UAV	Unmanned Aerial Vehicle
WP3	Work Package 3
T3.4	Task 3.4
VM	Virtual Machine

Funded by the European Union
This project has received funding from the European Union's HE research and innovation programme under grant agreement No 101093003

Executive Summary

Deliverable D3.3 - "Report on real-time federated analytics" is the third deliverable of the Work Package 3 within the TEMA project. This document encapsulates the research achievements of Task 3.4 (T3.4) - "Real-time federated analytics" over 24 months (M13-M36) of the project. This task focuses on addressing the challenges in developing a edge-cloud continuum computational infrastructure, with a federated distribution of artificial intelligence workloads over a large afflicted area. Towards this end, novel infrastructure and methodologies have been designed and developed to accurately serve as reference architecture on a federated edge-cloud computation for learning models. The focus on natural disaster management is demonstrated through scenarios and use cases (i.e., business mission), where experiments have been carried out. These techniques leverage advanced deployment and orchestration technologies, such as Docker and Kubernetes, useful tools belonging to cloud-based contexts, and advances algorithms based on artificial intelligence methods, such as deep neural network. The main focus is on the capacity to federate heterogeneous nodes distributed over a large geographically area (i.e., Italy, Greece, Spain). The federation enables different organization to take advantage of several computational resources on-demand.

The research under T3.4 has been directed towards the edge-cloud continuum computational infrastructure, with a federated distribution of artificial intelligence workloads. The task addresses the objective OA4, which aims to reduce latency by innovative federated data analytics on a cloud-to-edge continuum. Under the OA4, the T3.4 defines two key performance indicators: i) computational latency through federated edge nodes, and ii) reduce data migration for processing at federated edge nodes. The T3.4 contributed also to address OC1 and OC4, which, respectively, aim to improve natural disaster management systems using new digital technologies and extreme data analytics reducing latency and build a natural disaster management prototype focuses on wildfires, flash floods and regional floods.

Key achievements of this research period include the publication of 10 papers in conference proceedings and the submission of other 2 scientific papers to journals and/or international conferences. Additionally, the development of five TEMA essential components has been instrumental in enhancing the functionalities of the TEMA platform, as outlined in Deliverable D2.2.

This deliverable not only summarizes the main research outputs and technical challenges associated with designing innovative algorithms for extreme data analytics, but can also serve as a crucial reference for big data analytics and natural disaster management researchers. As a public document, it plays a vital role in disseminating TEMA findings to the broader scientific community.

Funded by the European Union

This project has received funding from the European Union's HE research and innovation programme under grant agreement No 101093003

1. Introduction

1.1. Purpose and scope of the document

Deliverable D3.3 - "Report on real-time federated analytics" is the third deliverable of the Work Package 3 (WP3) of the TEMA project. The main purpose of this document is to report the research results of Task 3.4 (T3.4) "Real-time federated analytics" between M13-M36. This deliverable is the first and last for the mentioned task.

The TEMA research efforts in the time period between M13 and M36 were focused on the following areas:

- Defining and building a federated Edge-Cloud continuum infrastructure for Natural Disaster Management.
- Building a Context Information Management using NGSI-LD and the Orion Context Broker component to set data context.
- Defining a offloading framework based on computational resource monitoring, forecasting and anomaly detection along with a scheduler of operational tasks.
- Building a computational resource forecasting model based on CPU utilization forecasting. The model is a Transformer and predicts 5, 10 or 15 minutes in the future.
- Building a computational resource anomaly detection model based on CPU utilization anomaly detection. The model is a Bi-LSTM and recognize anomalies intervals.

1.2. Structure of the document

Note: Each Section of this document contains the research progress made in the TEMA project between M13-M36 for the T3.4. Each Section briefly describes the state of the art and summarizes the research progress in the TEMA project.

The remainder of the chapter is organized as follows.

Chapter 2 gives a short summary on all contents of the Deliverable D3.3.

Chapter 3 provides information on the edge-cloud infrastructure that composes the TEMA platform. Detailed information are reported in the Deliverable D6.2. In addition, the Chapter discusses the methodology and results related with the KPI (Key Performance Indicator) OA4.1 - "Reduce the computational latency through the federated cloud-edge"".

Chapter 4 provides a state-of-the-art overview of Context Information Management systems, focusing on the capacity to use Orion Context Broker as data context system. In addition, the Chapter discusses the methodology and results related with the KPI OA4.2 - "Reduce the data migration for processing through the federated cloud-edge (continuum)".

Funded by the European Union
This project has received funding from the European Union's HE research and innovation programme under grant agreement No 101093003

Chapter 5 provides a state-of-the-art overview of offloading systems, focusing on data and computational resources, as well as the capacity to forecast and detect resource utilization anomalies. Additionally, it proposes an offloading framework based on a computational resource monitor, forecasting, and anomaly detection, as well as a scheduler for migrating operational tasks.

Funded by the European Union
This project has received funding from the European Union's HE research and innovation programme under grant agreement No 101093003

2. Summary of the Work Carried Out

2.1. Objectives

TEMA envisions addressing the challenges in extreme data analytics for NDM (Natural Disaster Management) like regional floods, flash floods, and wildfires by leveraging heterogeneous data sources including edge devices, sensors and IoTs (e.g., drones, wind sensors, stream flow gauges etc.), satellite images, geospatial data, meteorological data, and geosocial media. These data sources are heterogeneous, voluminous, frequently updated, complex, multilingual, dispersed, sparse, and extreme in nature. The main objective of TEMA WP3 "Trustworthy Federated Analytics," is to develop novel methods for accurate and fast/real-time semantic data analytics using inputs from multiple sources and in different modalities. This includes creating algorithms for trustworthy AI (Artificial Intelligence), federated analytics, and DNN (Deep Neural Network) data scarcity mitigation. This work considers distributed semantic analysis across the edge-to-cloud continuum, ncessary to minimize latency, using AI and DNNs for trustworthy and flexible heterogeneous data analytics. Fast computations are crucial for decision-making in emergency situations, supported by the vast cloud computing resources, spanning the entire edge-to-cloud continuum to meet the demands of extreme data analytics.

The specific TEMA objectives linked with T3.4 are derived from existing challenges in the compute continuum, involving cloud and edge nodes in a federated manner. They are presented in this document, along with accompanying Key Performance Indicators (KPIs) and Target Values (TVs) as defined in Section 1.1.1 of Part B of TEMAs Description of the Action (DoA). T3.4 contributes to the OA4, which aims to reduce latency by innovative federated data analytics on a cloud-to-edge continuum. Under the OA4, the T3.4 defines two key performance indicators: i) computational latency through federated edge, and ii) reduce data migration for processing at federated edge. The T3.4 contributed also to address OC1 and OC4, which, respectively, aim to improve natural disaster management systems using new digital technologies and extreme data analytics reducing latency and build a natural disaster management prototype focuses on wildfires, flash floods and regional floods.

Funded by the European Union
This project has received funding from the European Union's HE research and innovation programme under grant agreement No 101093003

2.2. Summary of the Work Carried Out with Respect to the Objectives

2.2.1. Federated Edge-Cloud Intelligence

Federated computation has been explored as a strategic approach to enhance the resilience, scalability, and responsiveness of emergency management systems. This is discussed in Chapter 3. The fire disaster scenario was selected as a representative use case to simulate realistic operational conditions, including distributed data sources, heterogeneous hardware, and intermittent connectivity.

The system architecture integrates different data streams including drone images, environmental sensors, meteorological models, satellite data, and geosocial media within a coherent edge-to-cloud framework. Unlike traditional centralized models, the implemented distributed architecture allows edge nodes to actively participate in both training and inference phases characterizing the AI models. This decentralization of the intelligence allows to reduce reliance on cloud infrastructure, preserving data privacy by keeping raw data locally, and accelerating decision-making processes in time-critical contexts.

The experimental setup involved deploying AI models across multiple edge nodes, each receiving a geographically contextualized subset of the Fire and Smoke Detection Dataset. These nodes performed local training and exchanged model weights through standardized interfaces, allowing for coordinated learning without centralized data aggregation. The system was tested under varying conditions to assess latency, network load, fault tolerance, and computational efficiency.

Although the fire scenario served as the primary testbed, the federated approach is adaptable to other analytical models and datasets relevant to emergency management. The validation confirmed that distributed AI workflows can operate effectively in constrained environments, contributing to the broader goals of the initiative and to the technical assessment of KPI OA4.1, which relates to operational efficiency, responsiveness, and system resilience in disaster response contexts.

2.2.2. Context Information Management

The Chapter 4 refers to the capacity of the TEMA platform to associate context to data, according to the European Telecommunications Standards Institute (ETSI). This is defined a Context Information Management (CIM), which is grounded in design principles that promote architectural flexibility and interoperability, enabling the seamless integration of various systems and domains via a unified API. The standard defines a RESTful NGSI-LD API (Next Generation Service Interface with Linked Data), where ”NGSI-LD” combines NGSI with LD, highlighting its alignment with semantic web standards. This

Funded by the European Union
This project has received funding from the European Union's HE research and innovation programme under grant agreement No 101093003

API allows context providers and consumers to create, update, query, and subscribe to context data in real time. Depending on how the core component, the Context Broker (CB), which implements the standards APIs, is deployed and interconnected, several architectural models are possible: centralized, distributed, or federated.
In this regard, a wildfires business mission (BM) was defined and executed as experiment in the Montiferru use case (real scenario deployed in Sardinia Region, Italy). The validation of the experiment was relevant to achieve the KPI OA4.2, which relates to the reduction of data migration in the NDM context.

2.2.3. Computation Offloading

The work carried out within the objective of offloading, which is further detailed in Chapter 5, focuses on the design, development, and implementation of two core components of the offloading pipeline: a forecasting module and an anomaly detection block.

The forecasting component is accomplished through a Temporal Transformer Model enhanced with a VM/container embedding layer, allowing the system to learn the behavioral characteristics of individual computing entities and to generate highly specific predictions for each of them. This architecture captures both local and long-range temporal dependencies, enabling more accurate modeling of resource utilization patterns in dynamic environments such as cloud and edge infrastructures. Moreover, the model supports multi-step forecasting, allowing it to predict CPU usage for 6, 12, or 24 timesteps into the future, which is particularly valuable for proactive workload management and adaptive offloading strategies. Experimental evaluations demonstrate that the proposed approach significantly outperforms traditional recurrent models, such as LSTM and Autoencoder-based baselines, achieving an R^2 of 0.84 and an MSE of 0.0047, thereby confirming its effectiveness in capturing complex temporal dynamics and enhancing the reliability of the offloading decision process.

Speaking of anomaly detection, our unsupervised solution relies on the implementation of a Bidirectional LSTM with an Autoencoder architecture designed to reconstruct the temporal evolution of CPU utilization curves and identify anomalous patterns. The model operates over variable input windows, supporting sequences of 6, 12, or 24 timesteps, thus allowing the detection of both short-term irregularities and longer-term deviations in resource consumption behavior. By learning to reproduce the normal operational dynamics of each virtual machine or container, the system highlights deviations as potential anomalies when the reconstruction error surpasses an adaptive threshold. Although still in the experimental phase, the results are promising, showing robust reconstruction capabilities with an average mean squared error of approximately 0.0017, indicating high fidelity in modeling the normal CPU consumption trajectories.

Funded by the European Union
This project has received funding from the European Union's HE research and innovation programme under grant agreement No 101093003

3. Federated Edge-Cloud Intelligence

Deploying computationally intensive deep learning models in resource-constrained edge environments is challenging and requires innovative approaches and optimized model architectures [2, 3]. These optimizations are critical for ensuring real-time performance and efficient resource utilization on devices with limited computational power and memory [4]. Additionally, the great amount of data produced by numerous edge devices requires advanced optimization techniques to manage communication overheads and guarantee data privacy and security when ML models are trained [5, 6]. This is particularly relevant for the applications have been studied, where the timely and accurate processing of visual data is fundamental [7].

To address these challenges, the TEMA platform leverages advanced technologies such as Federated Learning to train models while ensuring data privacy across distributed edge devices; the open-source FIWARE framework for efficient context management and data orchestration; and the Edge Intelligence (EI) hierarchy concept, which extends the AI capabilities from cloud-centric models to fully autonomous edge deployments. These approaches enable TEMA to optimize latency, resource utilization, and reliability in mission-critical applications like early fire detection.

Consequently, a significant research focus is optimizing these models for deployment on low-power embedded devices to ensure reliable, real-time detection in harsh environments [8]. To address this issue, researchers have explored various strategies to reduce computational complexity while maintaining acceptable detection accuracy. These strategies include model pruning, quantization, and architectural modifications [9]. This focus fits perfectly with the trend of Edge Intelligence [1], where AI models are tailored for deployment on edge devices by leveraging techniques like AutoML and Neural Architecture Search (NAS) to devise resource-efficient models suited to hardware constraints [3]. Real-world deployments often require adaptive solutions that can dynamically balance the load between edge nodes and cloud servers, particularly in large-scale applications [10]. The growing ubiquity of small-footprint, vision-based sensors calls for deeper integration between model optimization and system-level design. This integration would optimize inference latency, communication costs, and reliability to meet the stringent requirements of early fire detection in distributed environments [4].

The shift towards edge computing is ultimately driven by the exponential growth of data at the periphery and the demand for low-latency responses. This shift necessitates a departure from traditional, cloud-centric processing models, which are often ill-suited for scenarios involving massive data volumes and stringent, real-time requirements [10]. In this emerging paradigm, federated and centralized learning approaches

Funded by the European Union
This project has received funding from the European Union's HE research and innovation programme under grant agreement No 101093003

are not competing strategies, but rather complementary layers of a distributed intelligence continuum. In this continuum, global model convergence and local adaptability coexist to improve the accuracy and responsiveness of mission-critical fire detection applications.

Following from these premises, and building on the comprehensive taxonomy introduced in *"Edge Intelligence: Paving the Last Mile of Artificial Intelligence With Edge Computing"* [1], this Chapter adopts the seven-level hierarchy of Edge Intelligence (EI), which extends from Level 0 - Cloud Intelligence to Level 6 - Full Edge Intelligence. This classification, shown in Figure 1, outlines the progressive migration of AI capabilities from centralized cloud infrastructures to fully distributed edge ecosystems. Specifically, Level 0 (Cloud Intelligence) represents the traditional cloud-centric computation, in which both training and inference occur remotely. Intermediate levels (1-5) progressively decentralize AI tasks by enabling partial inference, collaborative learning, and adaptive model partitioning between the cloud, edge, and device layers. Level 6 (Full Edge Intelligence), on the other hand, denotes complete autonomy, in which both learning and inference are executed locally on edge devices with minimal or no cloud dependency.

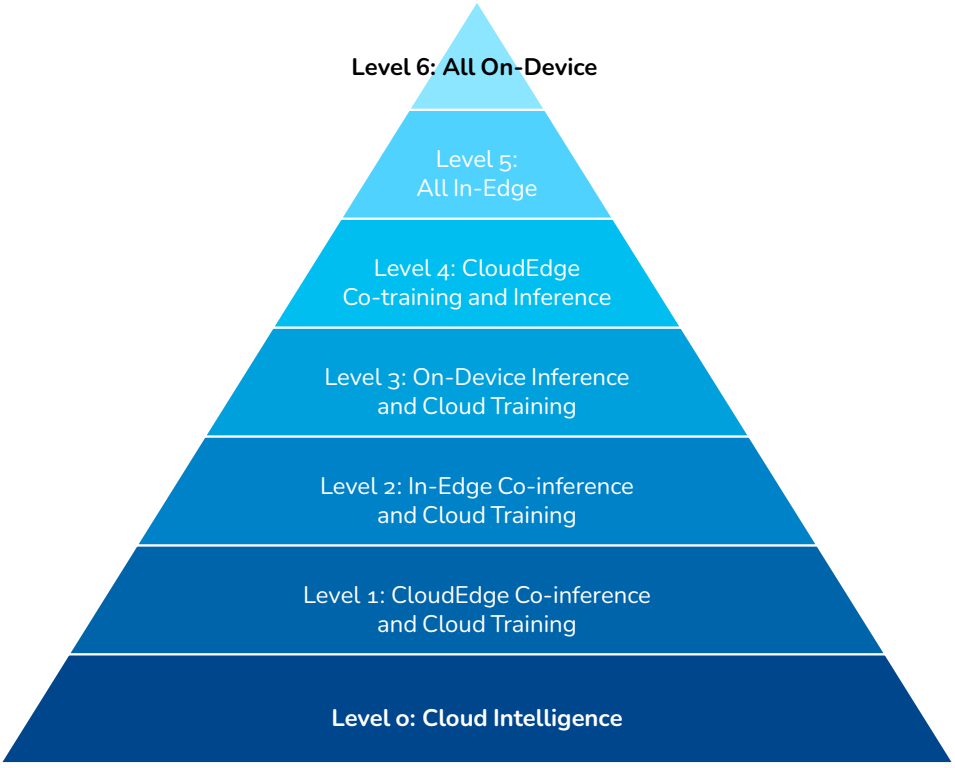


Figure 1. Visual representation of the seven levels of Edge Intelligence [1], ranging from full cloud-based intelligence (Level 0) to fully on-device training and inference (Level 6). As the levels rise, data offloading decreases while autonomy, privacy, and latency efficiency increase.

Funded by the European Union
This project has received funding from the European Union's HE research and innovation programme under grant agreement No 101093003

The Chapter aims to analyze fire and smoke detection under two different deployment and learning paradigms, centralized cloud-based training and fully federated edge implementations, by framing the present study within this hierarchical model. This multi-level approach enables a systematic evaluation of the trade-offs between latency, accuracy, privacy, and computational efficiency. It also positions the proposed research within the broader evolution of edge intelligence architectures for safety-critical, real-time applications.

This research correlates with KPI OA4.1, which is related to the capacity to "reduce the computational latency through the federated cloud-edge", even knows as computing continuum. Specifically, the target value is reducing the computational latency by 10-15% over the state of the art. To achieve this KPI, a disaster scenario was setup and a state of the art fire detection algorithm, based on YOLO, was used as benchmark to compare the traditional and centralized computation with the distributed computation.

The rest of the Chapter discusses the state of the art, the architecture, the methodology and the results of the experiments carried out. The KPI has been achieved and results are shown in the following.

3.1. State of the Art

Recent advances in Edge Intelligence (EI) have profoundly transformed the architectural landscape of AI-based fire and smoke detection systems, moving from cloud-centric processing toward distributed and device-level intelligence. Within the conceptual framework, detailed in Table 1, articulated by Zhou in [3], this evolution can be understood as a progressive migration across the levels of the EI hierarchy, where training and inference are increasingly delegated to edge devices. This paradigm responds to the growing demand for low-latency, privacy-preserving, and context-adaptive computation, which is particularly critical in safety-sensitive scenarios such as wildfire prevention or infrastructure monitoring. The continuous expansion of data generated at the network periphery, coupled with the need for real-time responsiveness, underscores the inadequacy of purely cloud-based paradigms and motivates the integration of distributed learning mechanisms at the edge [10].

Funded by the European Union
This project has received funding from the European Union's HE research and innovation programme under grant agreement No 101093003

Level / Mode	Definition
Level 0: Cloud Intelligence	Training and inferencing the DNN model **fully in the cloud** [1].
Level 1: CloudEdge Co-inference and Cloud Training	Training the DNN model in the cloud, but inferencing the DNN model in an **edge-cloud cooperation manner**. (Here edge-cloud cooperation means that data are partially offloaded to the cloud) [2].
Level 2: In-Edge Co-inference and Cloud Training	Training the DNN model in the cloud, but inferencing the DNN model in an **in-edge manner**. (Here in-edge means that the model inference is carried out within the network edge, realized by fully or partially offloading the data to the edge nodes or nearby devices via D2D communication) [2].
Level 3: On-Device Inference and Cloud Training	Training the DNN model in the cloud, but inferencing the DNN model in a **fully local on-device manner**. (Here on-device means that no data would be offloaded) [2].
Level 4: CloudEdge Cotraining and Inference	Training and inferencing the DNN model **both** in the edge-cloud cooperation manner [2].
Level 5: All In-Edge	Training and inferencing the DNN model **both** in the in-edge manner [2].
Level 6: All On-Device	Training and inferencing the DNN model **both** in the on-device manner [2].

Table 1. Six-level Rating for Edge Intelligence (EI) [1]

Though powerful, centralized ML architectures remain constrained by bandwidth, privacy, and scalability issues, which significantly hinder real-time deployment in dynamic environments [2]. Edge computing provides a compelling alternative, enabling localized processing and inference close to the data source. This minimizes latency and communication overhead while preserving data sovereignty. Avgeris et al.'s SMOKE framework exemplifies this approach by offering a scalable edge infrastructure for early fire detection. This infrastructure leverages distributed camera networks to ensure rapid hazard recognition, even under bandwidth-limited conditions [11]. Similarly, Aramendia et al. demonstrated that neural network models for fire detection can be effectively deployed on embedded edge devices without a significant loss of accuracy. This finding underscores the viability of real-time vision systems with limited resources for critical applications [8].

Federated Learning is a cornerstone of distributed intelligence beyond inference, en-

Funded by the European Union
This project has received funding from the European Union's HE research and innovation programme under grant agreement No 101093003

abling edge nodes to collaboratively update shared models without transferring raw data. Yang et al. introduced the Nesterov Accelerated Gradient (NAG) approach to federated optimization, which achieves faster convergence and reduced communication costs across heterogeneous clients [2]. Canonaco et al. expanded on this idea by proposing an adaptive federated framework that can maintain performance under concept drift. This ensures robustness in dynamic environments, such as fire detection scenarios, where illumination and background conditions change over time [12]. Similarly, the FedVision platform developed by Liu et al. demonstrated the feasibility of large-scale collaborative training for object detection directly at the edge. This approach reduces privacy risks while maintaining high inference accuracy [13].

However, fully realizing Edge Intelligence requires overcoming significant challenges related to device heterogeneity, resource management, and distributed model training [14]. Recent surveys by Meuser et al. and Xu et al. emphasize that next-generation embedded intelligence systems must balance computational efficiency and adaptability. These systems must leverage model partitioning, neural acceleration, and energy-aware scheduling to meet real-time constraints [15, 16]. In this direction, the deployment of Neural Processing Units and on-device AI architectures, as discussed by Zhang and Tao [17] and Wang et al. [18], is redefining the hardware-software co-design of intelligent devices, enabling secure and efficient local inference. Moreover, advances in optimization for non-IID data in federated settings, such as those explored by Efthymiadis et al., further enhance the scalability and robustness of distributed edge learning systems [5].

A work by Peng et al. [9] demonstrated that model compression and architectural optimization techniques (e.g., quantization, Spatial-Piramyd-Pooling-Fast, and GhostNet-based modules) can significantly improve inference speed on low-power devices without compromising detection accuracy an essential step toward Level 6 Edge Intelligence, where both training and inference occur fully at the edge. Collectively, these contributions illustrate the ongoing convergence of federated learning, hardware acceleration, and model optimization as the driving forces behind intelligent, autonomous fire and smoke detection at the network periphery.

Recent works further advance the state of the art by addressing specific trade-offs in model size, detection speed, and explainability. For instance, FCMI-YOLO proposes modules like FasterNext and Mixed Local Channel attention to improve detection of small fire targets, reduce GFLOPs and model parameters compared to YOLOv5s, and demonstrates that with a light design significant improvements can be achieved in practice [19].

The UAV-based study by Vasquez et al. tackles the limited dataset problem and shows that transfer learning can increase generalization while operating under edge constraints of latency and energy [20]. In the same vein, it has shown by Titu et al. that model dis-

Funded by the European Union
This project has received funding from the European Union's HE research and innovation programme under grant agreement No 101093003

tillation (teacher-student) helps balance accuracy and speed (inference on accessible hardware such as Raspberry Pi) for mobile aerial surveillance systems [21].

In the context of semantic segmentation, Lee et al. demonstrates that heavy networks such as DeepLabV3+ can be slimmed without dramatic loss in accuracy, achieving about 10 fps on embedded SoCs like the Qualcomm QCS610; this is useful in UAV scenarios or fixed sensors deployed in remote environments [22].
Furthermore, the system built by Mahmoudi et al. [23] attempts to combine compressed models, Vision Transformers, and explainability mechanisms in order to achieve a good trade-off among performance, interpretability, and hardware requirements at the network edge.

Nevertheless, the full realization of Edge Intelligence involves significant challenges: device heterogeneity, resource management, and the distributed training of complex models remain nontrivial obstacles [14]. Recent investigations emphasize that successful EI systems must integrate strategies such as model partitioning, hardware acceleration, energy-aware scheduling, and robustness under varying environmental conditions [15, 16]. Hardware technologies like NPUs, together with on-device AI architectures, are becoming central for ensuring secure, high-performance local inference [17, 18]. Meanwhile, optimization for non-IID data in federated settings, as explored by Efthymiadis et al., plays a crucial role in the scalability and robustness of distributed systems [5].

Finally, recent work by Yang et al. demonstrates the effectiveness of collaborative architectures combining cloud and edge: the edge model performs inference, while the cloud handles iterative updates based on user feedback to reduce false positives and adapt to new environmental conditions [24]. These examples illustrate how federated learning, hardware accelerators, model compressibility, and collaborative learning emerge as key components for creating robust, autonomous, and context-aware fire/smoke detection systems in the higher levels of the Edge Intelligence pyramid.

Despite the significant advances in Edge Intelligence, several critical gaps remain unaddressed, notably in managing heterogeneity of edge devices, ensuring scalability and robustness of distributed learning under dynamic environmental conditions, optimizing resource allocation efficiently, and guaranteeing privacy and low-latency performance in real-time disaster detection contexts. The TEMA platform explicitly targets these challenges by integrating federated learning frameworks, hardware-accelerated processing, adaptive model optimization techniques, and a scalable cloud-edge continuum architecture to enable robust, autonomous, and privacy-preserving fire and smoke detection systems at scale.

Funded by the European Union
This project has received funding from the European Union's HE research and innovation programme under grant agreement No 101093003

3.2. Beyond the State of the Art

An early version of the cloud-edge architecture was already discusses in the Deliverable D3.1. A deep understanding of the advances beyond the state of the art and the progresses are reported in the Deliverable D6.2. In the following, it is reported the main information needed to read the rest of this document and mostly understand the experiments to achieve the KPI OA4.1.

A key feature of this system is its combination of centralized and distributed components, which ensures the efficiency and scalability of disaster monitoring and response. As shown in Figure 2, centralized services handle data aggregation, workflow orchestration, and integration with external systems. Distributed components process information closer to the data sources, reducing latency and enhancing resilience. Following the Compute Continuum paradigm, services are distributed between cloud and edge layers.

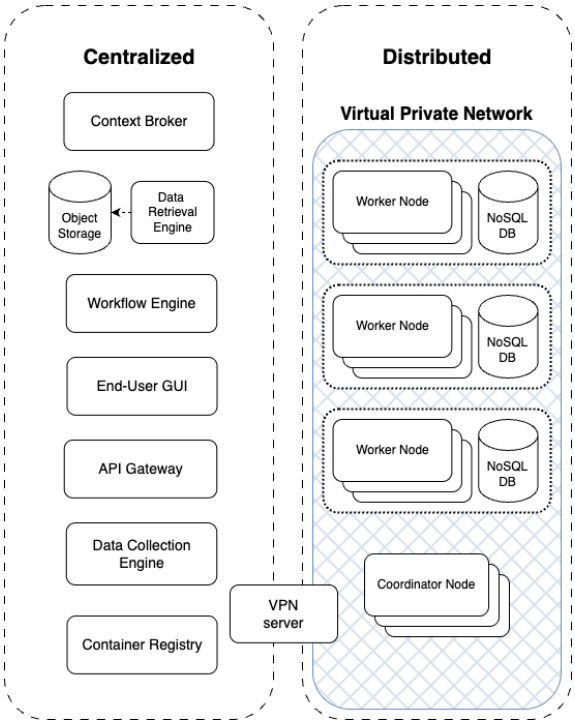


Figure 2. Centralized and distributed components in the cloud-edge continuum

To support this approach, a set of core services ensures scalability, efficiency, security, and adaptability in handling environmental and disaster-related data. These services include:

Context Broker It acts as a central hub to manage and contextualize real-time data streams from multiple sources. It enables seamless integration with external systems for improved decision-making.

Funded by the European Union
This project has received funding from the European Union's HE research and innovation programme under grant agreement No 101093003

Object Storage & Data Retrieval It provides scalable and reliable storage for unstructured data, optimizing accessibility for analytics and external applications.

Workflow Engine It automates and orchestrates data flows within the infrastructure, streamlining processes and optimizing resource utilization.

API Gateway It serves as a centralized entry point for data access, facilitating interoperability with external applications and enabling batch data ingestion.

Data Collection Engine It aggregates and processes real-time and historical datasets, ensuring continuous data availability for analysis.

Container Orchestrator & Registry It manages the automated deployment and scaling of containerized services, ensuring system reliability and adaptability.

The cloud-edge continuum is managed via a private, flat network that relies on Virtual Private Network (VPNs) to securely connect all computing nodes. Distributed components, such as Kubernetes worker nodes and NoSQL databases, collaborate with centralized services to ensure seamless data processing. A public endpoint node is in place to connect these components to external sources, support data ingestion, and facilitate secure external access. This architecture provides a robust framework for managing large-scale environmental and disaster-related data, thereby contributing to the resilience of future smart cities against possible meaningful natural disasters.

The architecture is publicly released as open-source software. Design, implementation and documentation is available in [110].

3.3. Achieving the KPI OA4.1

3.3.1. Business Mission

The use case is defined as a BM, which is a set of activities that technologies and first responders should carry out during a natural disaster, such as a flood or fire. The goal is to assess conditions as the event evolves. It includes complete evaluations through pre-event planning, event management, and subsequent analysis and evaluation. The focus of these missions is to provide cross-functionality among devices, AI tools, and software solutions, as well as to create increasing value over time based on minimum integration criteria. These problems can be solved by establishing minimum integration criteria that maximize interoperability and able to integrate everything into a common cloud-edge data processing environment. This solution guarantees a unique view of a complex workflow deployed across many platforms leveraging different technologies. Before setting up the technical solution conceived as well the definition of its functionalities, has followed the analysis and formalization of user requirements collected in the

Funded by the European Union
This project has received funding from the European Union's HE research and innovation programme under grant agreement No 101093003

pilot area.

The solution has been concretely tested and evaluated in four European Regions. The Bavaria Region of Germany and the Municipality of Mantoudi-Limni-Agia Anna of Greece addressed flooding. The Sardinia Region of Italy and the Municipality of Kajaani in Finland addressed fire management. The solution has been then tested and evaluated also in Sardinia, Italy, useful to address fire management. The impact of extreme events that occurred in the pilot area has been analyzed. During the execution of the pilot, useful information hs been collected and end users needs have been identified, in order to adapt and generalize the functionalities of the whole TEMA solution. In July 2021, the Autonomous Region of Sardinia faced a severe crisis when a widespread 15,000-hectare forest fire caused serious damage in the Montiferru area located in the central part of Sardinia. This scenario has been simulated in real time for TEMA validation. Available data has been considered, specifically about the area's vegetation and geomorphology, details about the damage, safety procedures adopted, the output of meteorological models, fire danger forecasts, satellite images, and videos of the burnt area taken by drones. The TEMA solution has been used to examine post-event territory conditions, particularly with regard to geomorphological risk implications. Using the TEMA solution makes possible to examine and improve disaster management and decision-making procedures.

The final objective is to validate the effectiveness and adaptability of the TEMA solution in realistic disaster settings. In fact, these trials are designed to:

- determine whether the solution successfully addresses existing capability gaps in natural disaster response;
- evaluate the systems performance in managing flash flood and wildfire scenarios using both historical data to test the technologies, training the algorithms and the way they recreated a past scenario and real-time data to evaluate the technologies in a real scenario;
- assess the added value of integrating digital technologies and extreme data analytics into NDM processes.

The evaluation process has been conducted involving experienced end-user evaluators selected from Public Protection and Disaster Relief (PPDR) agencies. These stakeholders tested the operational utility of the TEMA system by simulating disaster response missions in near-real conditions. Their feedback has been essential for measuring the platform's practical impact and refining its capabilities for the next deployment in real-world emergency contexts. In response to specific end-user requirements and leveraging different integrated technologies, a preliminary BM has been developed. These strategic statements define the core purpose and objectives of the TEMA initiative, tailoring them to the specific needs and challenges of end users while ensuring alignment with innovation goals.

Funded by the European Union
This project has received funding from the European Union's HE research and innovation programme under grant agreement No 101093003

The wildfire BM as been defined. However, BMs for other disasters (e.g., floods) may behave in the same way. The BM in Figure 3 starts when an alert is launched. The first action must be taken by a human. Using the TEMA webpage, the operator prepares the necessary information to create a BM. This information is included in the alert and may contain details such as the location of the event (e.g., drawing a polygon on a map) and the type of disaster (e.g., flooding or wildfire). Creating an alert means creating an entity in the Orion-LD Context Broker. As soon as the alert entity is instantiated, all subscribed components will be notified. These components then process the information in the alert entity according to their logic.

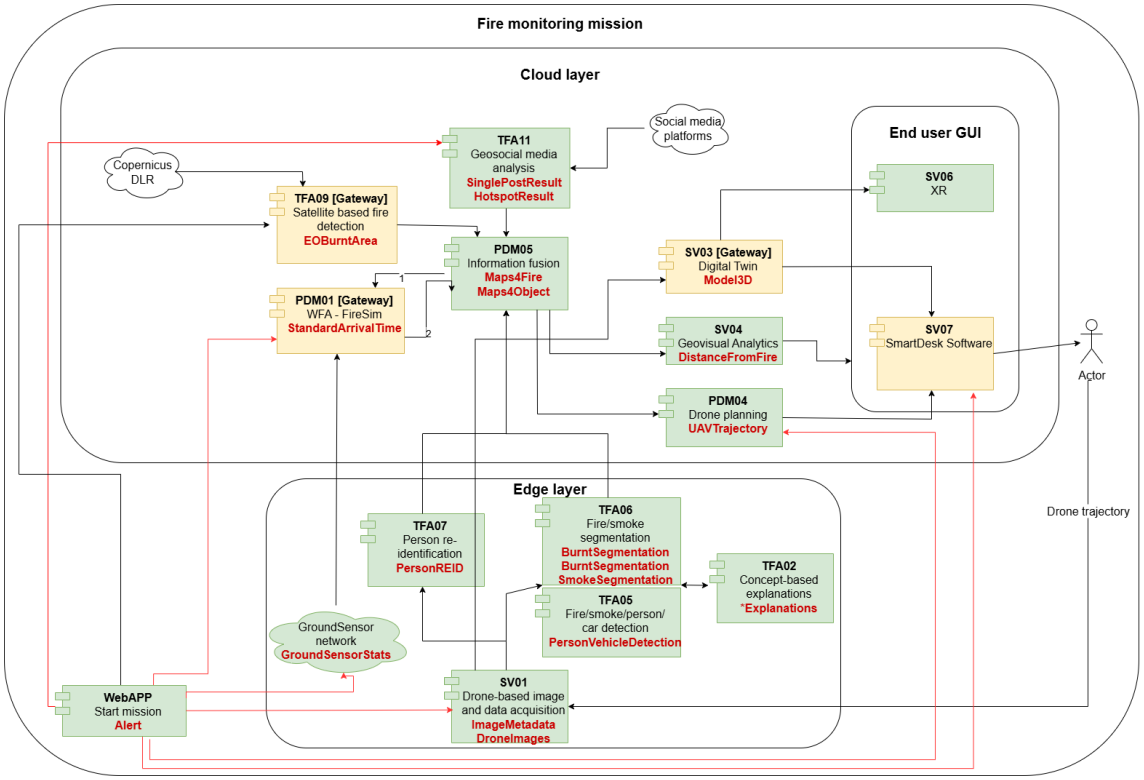


Figure 3. The fire Business Mission

The following is a concise overview of the technical components affected by the alert entity, as well as the data they exchange and manage. The first components notified by the Alert Entity are:

- Drone-based image and data acquisition to take pictures and fly-by footage of the area affected by the fire. The other two components, that are the fire/smoke segmentation and the fire/smoke/person car detection, analyze the images to identify locations of fire, smoke, individuals, and vehicles.
- Ground Sensor Network to acquire weather data from sensors that are either deployed in the field or mounted on drones.

Funded by the European Union
This project has received funding from the European Union's HE research and innovation programme under grant agreement No 101093003

- Fire simulation to create a model of how a fire can evolve, starting with an analysis of the ground and fuel in the area. This model provides information about the fire's path, intensity, flame length, and rate of spread.
- Satellite-based fire detection to provide an analysis of buried areas using satellite data.
- Geo-social media analysis to collect and process posts from social media. It provides the emotions (e.g., fear, sadness, anger, and joy) extracted from each post over time.

The Information Fusion component combines all data to create a probability map showing where the fire is likely to be. The Visual Analytics component uses this probability map as input to provide information on the distance between houses and buildings, as well as to highlight public buildings such as schools, hospitals, and town halls. The system also uses drone images to generate a 3D model of the affected area. This model can be explored through an XR viewer, offering an immersive experience of the burned area, which is useful for those working in the control room. The TEMA solution uses AI to transform raw data into actionable business information. This enables first responders and policymakers to make evidence-based decisions in real time. The TEMA solution can serve as the control room's eyes.

3.3.2. Disaster Scenario

The fire disaster scenario is the operational reference scenario used to validate KPI OA4.1. Focusing on the NDM domain, it addresses the detection and response to forest and urban fires. This scenario was chosen because it provides a realistic, complex, and dynamic environment in which to compare different computational and analytical approaches. The goal is to evaluate the performance of the TEMA framework under distributed processing conditions. The scenario represents the experimental evolution oriented toward the technical and scientific validation of federated computation and federated analytics capabilities. While it does not reproduce a real emergency, it does constitute a controlled experimental environment in which typical operational conditions of a disaster event are simulated using representative datasets and a distributed edge-to-cloud infrastructure.

This scenario is part of the TEMA project's broader framework for monitoring, preventing, and responding to natural disasters. In this context, various actors and technological components collaborate to manage information in a coordinated manner:

- edge nodes and distributed sensors are responsible for data acquisition and preprocessing (e.g., images, meteorological parameters, environmental telemetry);
- central servers and high-performance computing (HPC) infrastructures are responsible for aggregating results and managing global analytical models;
- Public Protection and Disaster Relief (PPDR (Public Protection and Disaster Relief) operators and control centers, who use analytical results to plan and optimize field

Funded by the European Union
This project has received funding from the European Union's HE research and innovation programme under grant agreement No 101093003

operations.

This multi-level interaction reflects the TEMA project's systemic approach, in which the integration of edge, cloud, and human components forms the foundation for more efficient and resilient environmental emergency management.

In an extended fire context, various information sources and sensing systems continuously generate heterogeneous data. These sources include images captured by drones and surveillance cameras, environmental sensor measurements, meteorological data, reports from field operators, and satellite information. This data must be analyzed quickly to provide an up-to-date, coherent view of the fires evolution, estimate flame propagation, and support operational decision-making. Within the TEMA project, a distributed edge-to-cloud architecture manages this data flow. In this architecture, edge nodes actively participate in the analysis and training phases of artificial intelligence models, not just passively acquiring data. This approach reduces dependence on the central cloud, enhances operational resilience, and optimizes processing times by promoting collaborative computation across heterogeneous devices.

To assess the performance of this architecture, two complementary configurations were implemented within the same scenario:

Centralized Configuration The model training and validation are executed on a high performance server using the entire dataset in a monolithic fashion.

Federated Configuration The training is distributed across heterogeneous edge nodes, each equipped with a local subset of the dataset and autonomous computational resources.

The objective is to verify how the two modes behave in terms of efficiency, latency, and model-update capability, simulating a Federated Computation architecture applied to a natural-disaster use case.

The Fire and Smoke Detection Dataset[1] was used for the experiment. The dataset contains annotated images of smoke and flames in various environmental contexts, such as forests, industrial areas, and urban environments. The dataset was organized to realistically distribute the data sources; each experimental node received a local subset of images as if they were captured by cameras and sensors in different geographic areas. This configuration allowed us to evaluate the systems behavior in a non-homogeneous and partially disconnected environment, which is typical of real-world field monitoring operations. Experiments were conducted to ensure the reproducibility and consistency of the training parameters (e.g., number of epochs, batch size, image size, and deterministic seeds), which allowed for an objective and transparent comparison between the two paradigms.

[1] https://www.scidb.cn/en/detail?dataSetId=ce9c940ob44148e1boa749f5c3ebobda

Funded by the European Union
This project has received funding from the European Union's HE research and innovation programme under grant agreement No 101093003

This scenario was also used to test the interoperability and orchestration capabilities of the different TEMA framework modules. While each edge node operated independently, they remained semantically integrated within the system, sharing information, models, and metadata through standard protocols and common interfaces. This enabled the validation of the framework's modularity and scalability, as well as its capacity to integrate diverse information from various sources, including computer vision, environmental sensors, and meteorological and geospatial data. This type of integration is crucial for ensuring the semantic consistency and temporal alignment of data processed at various levels of the edge-to-cloud infrastructure.

The fire disaster scenario has been integrated within the full disaster-management cycle defined by the TEMA project, contributing to the three main operational phases:

- prevention, through the training of AI models capable of early detection of smoke or environmental variations indicative of potential ignition conditions;
- response, through the ability to dynamically update models and provide distributed near-real-time analyses to command centers;
- post-event evaluation, through re-processing of collected data and continuous updating of analytical models to improve future performance.

In this way, the TEMA system not only manages ongoing events, but also enables a continuous learning cycle. In this cycle, analytical results are reintegrated into the training process, which progressively refines the platform's predictive capabilities.

Although it is conducted in a controlled environment, this scenario accurately reproduced the typical challenges of a real operational context, including the geographical distribution of data sources, the heterogeneity of computational resources, intermittent connectivity, and the need for reduced processing times. Parallel execution of centralized and federated experiments allow to analyze the tangible contribution of the distributed approach to improving the TEMA system's resilience, scalability, and analytical responsiveness. Additionally, experimentation on heterogeneous platforms (GPUs, CPUs, and embedded devices) validated the performance of the AI model, as well as the overall efficiency of the architecture.

In conclusion, the fire disaster scenario served as a technical and operational testing ground for the TEMA platform. It demonstrated the frameworks ability to perform distributed training and analytics in a realistic disaster management context.

3.3.3. Experimental Setup

A multi-level experimental setup was designed to systematically validate the KPI OA4.1. This experiment's primary objective is to quantitatively analyze and compare the performance, efficiency, and resilience of diametrically opposed computing paradigms, ranging from purely centralized to fully distributed. Three distinct architectural configurations were implemented, each representing a specific level of this hierarchy, enabling a pro-

Funded by the European Union
This project has received funding from the European Union's HE research and innovation programme under grant agreement No 101093003

gressive comparison of architectural impact:

- **Level 2 (Cloud-Centric Baseline)**: a traditional approach (Level 2: In-Edge Co-inference and Cloud Training), where training is entirely centralized in the cloud. The edge nodes, despite being capable of processing, act only as inference terminals, receiving a model trained elsewhere. This serves as our reference baseline to measure the benefits of the alternatives.
- **Level 4 (Hybrid Federated)**: a hybrid Federated Learning approach (Level 4: Cloud-Edge Co-training and Inference). Here, the training (the computationally intensive task) is moved to the edge, close to the data source, preserving privacy and reducing raw data traffic. However, the coordination and aggregation of models remain centralized, entrusted to a cloud server.
- **Level 5 (Full Edge Federated)**: an "All-In-Edge" Federated Learning approach (Level 5). This is the most advanced configuration, reflecting a true edge-native logic. The entire AI lifecycle, from local training to model aggregation, resides entirely at the network periphery. The cloud is excluded from the training cycle, increasing the system's resilience and autonomy.

This methodical progression from Level 2 to Level 4 makes it possible to isolate and measure the impact of computational decentralization on the key identified parameters of the scenario: model update latency, network traffic volume, resilience to intermittent connections, and resource utilization on heterogeneous devices.

The following table summarizes the three tested configurations. The subsequent subsections detail the specific architecture and data flow of each experiment, as illustrated in their respective diagrams.

Model	Level	Training Type	Cloud Deployment	Edge Deployment
CNN	2	Traditional	Server CPU/GPU (training)	Edge Nodes (inference)
	4	Federated Computation	Server (aggregator)	Edge Nodes CPU/GPU (training/inference)
	5	Federated Computation	None	Edge Node (aggregator) Edge Nodes CPU/GPU (training/inference)

Table 2. Plan of the experiments. Level refers to the six-level for edge intelligence

3.3.3.1. Cloud-Legacy Architecture (Level 2)

The Cloud-Legacy architecture is shown in Figure 4. It represents a modified traditional model, corresponding to Level 2: In-Edge Co-inference and Cloud Training of the EI hierarchy. In this architecture, the entire training process is centralized in the cloud, which manages the data ingestion pipeline and the model training process itself.

Funded by the European Union
This project has received funding from the European Union's HE research and innovation programme under grant agreement No 101093003

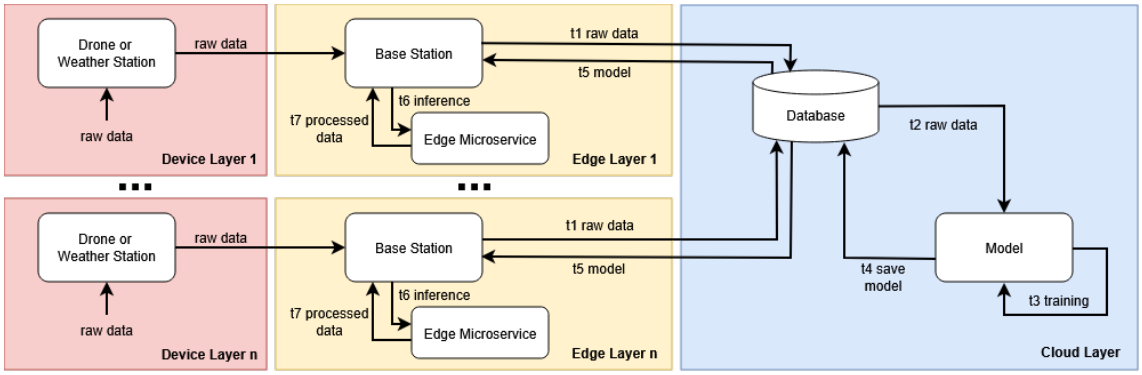


Figure 4. Cloud-Legacy Architecture Level 2

Unlike a purely legacy approach (Level 0), the edge layer is not entirely passive. While it sends raw data to the cloud, it is also the recipient of the trained model and hosts a microservice dedicated to performing local inference. This hybrid approach offloads the inference burden to the periphery, reducing decision-making latency, but it remains completely dependent on the cloud for model training and updates, still requiring the transfer of all raw data to the center.

The architecture is organized into three levels:

- the Device Layer includes the devices in the field that collect raw data;
- the Edge Layer acts as both a bridge for raw data to the cloud and a host for inference microservices;
- the Cloud Layer stores all data, trains the model, and distributes it to the periphery.

The operational flow of this architecture is shown in Table 3.

Time	Description
T0	The raw data is received by the base station
T1	The raw data is sent from the Base Station to the database in the cloud
T2	The raw data is retrieved from the database to feed the training process
T3	The model is trained (or retrained) centrally in the Cloud Layer
T4	The updated model is saved in the central database
T5	The trained model is distributed from the database to the Base Station in the Edge Layer
T6	The model is sent to the local edge microservice to perform inference
T7	The edge microservice returns the processed data (inference results) to the Base Station

Table 3. Operational flow of the Cloud-Legacy Level 2 architecture

Funded by the European Union
This project has received funding from the European Union's HE research and innovation programme under grant agreement No 101093003

3.3.3.2. Cloud-Edge Architecture (Level 4)

The Cloud-Edge architecture is shown in Figure 5. This configuration implements a hybrid federated computation paradigm corresponding to level 4: cloud-edge co-training and inference. The key innovation is moving the training process to the Edge Layer, where raw data is processed and utilized locally.

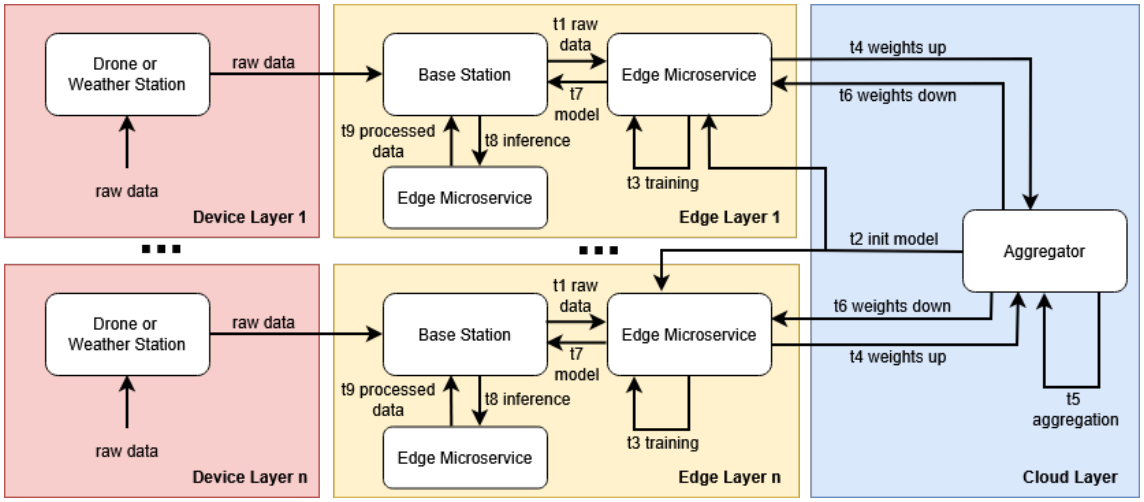


Figure 5. Cloud-Edge Architecture Level 4

In this architecture, the Edge Layer plays an active, computationally intensive role. The Cloud Layer is redefined; it no longer handles raw data, but rather hosts a central "Aggregator". The Aggregator's sole task is to orchestrate the training process. It distributes the global model and receives parameter updates (weights) from the edge nodes. Then, it performs the aggregation.

This approach dramatically improves privacy raw data never leaves the edge and reduces network traffic. However, it still depends on the cloud for coordination. The architecture is organized into three levels:

- the Device Layer collects data;
- the Edge Layer manages raw data, performs local training, and handles inference;
- the Cloud Layer hosts only the Aggregator for federated coordination.

The operational flow is shown in Table 4.

Funded by the European Union
This project has received funding from the European Union's HE research and innovation programme under grant agreement No 101093003

Time	Description
To	The raw data is received by the base station
T1	The raw data is received and maintained locally by the edge microservice
T2	The Aggregator in the cloud distributes the initial global model to the edge nodes
T3	The edge microservice performs local training using its own raw data
T4	The updated model weights (not the data) are sent to the Aggregator in the cloud
T5	The Aggregator performs the aggregation of the weights received from all nodes
T6	The new global aggregated model is sent to the edge microservices
T7	The updated model is sent to the inference microservice
T8	The microservice performs local inference
T9	The microservice performs local inference

Table 4. Operational flow of the Cloud-Edge Level 4 architecture

3.3.3.3. All-In-Edge Architecture (Level 5)

The All-In-Edge architecture is shown in Figure 6. This is the most advanced and decentralized configuration, corresponding to Level 5 of the EI hierarchy, All-In-Edge. The entire artificial intelligence lifecycle, including training and aggregation, takes place entirely within the Edge Layer.

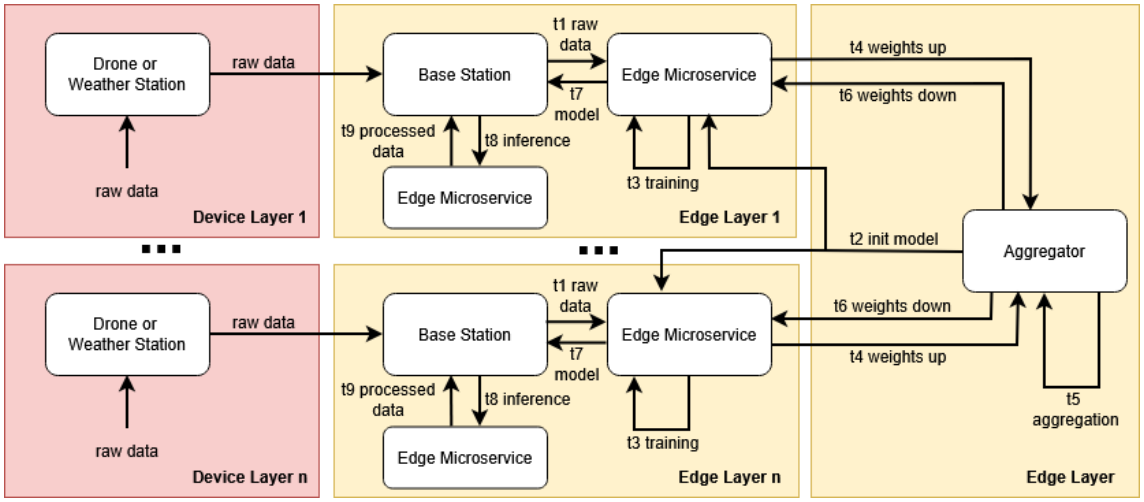


*Figure 6. **All-In-Edge** Architecture Level 5*

In this model, the cloud layer is absent from the training cycle. The Edge Layer is functionally divided. Some nodes act as "clients," performing local training, while another node with adequate resources acts as the "Aggregator." All communications, including model weight uploads and global model downloads, occur entirely among edge nodes

Funded by the European Union
This project has received funding from the European Union's HE research and innovation programme under grant agreement No 101093003

and never they reach the central server.

This approach ensures maximum autonomy and resilience, as well as low latency of the system. The platform can continue to learn and operate even without cloud connectivity. The architecture is organized into just two active levels:

- the Device Layer collects data;
- the Edge Layer, which handles data acquisition, local training, aggregation, and inference.

The operational flow, identical to Level 4 but with different actors, is shown in Table 5.

Time	Description
T0	The raw data is received by the base station
T1	The raw data is maintained locally by the edge microservice (client)
T2	The Aggregator on the edge distributes the initial global model to the other edge nodes
T3	The edge microservice (client) performs local training
T4	The updated model weights are sent to the Aggregator on the edge
T5	The Aggregator on the edge performs the aggregation of weights received from the clients
T6	The new aggregated global model is sent back to the edge microservices (clients)
T7	The updated model is sent to the local inference microservice
T8	The microservice performs local inference
T9	The inference microservice returns the processed data to the Base Station

Table 5. Operational flow of the All-In-Edge Level 5 architecture

3.3.4. Testbed

The empirical validation of the three experimental architectures (Cloud-Legacy Level 2, Cloud-Edge Level 4, and All-In-Edge Level 5) requires a physical and logical infrastructure, or Testbed, to enable their implementation and measurement. This testbed is not just a collection of hardware, but a controlled ecosystem specifically designed to replicate the operational conditions and challenges of the fire disaster scenario described previously.

The objectives of this testbed are multiple, but three key requirements guided its design:

Computational Heterogeneity This is the main challenge. A real emergency scenario does not use a single type of device. The testbed must include heterogeneous hardware ranging from high-performance servers (cloud) to powerful edge-AI devices (like drones) down to very low-cost, low-power sensors (like fixed cameras). The validation of federated computation is based on its ability to orchestrate this diversity.

Scalability Although the testbed uses a limited number of physical nodes, its architecture (based on containers and orchestrators) must allow for simulating and evalu-

Funded by the European Union
This project has received funding from the European Union's HE research and innovation programme under grant agreement No 101093003

ating the system's performance in a large-scale scenario, analyzing how it would behave as the number of federated clients increases.

Reproducibility As this is a technical-scientific deliverable, every result must be verifiable and reproducible. The testbed is therefore rigorously documented, specifying every hardware component and software version used, to ensure that the experiments can be replicated by third parties.

The following sections Hardware Infrastructure and Software Stack, describe in detail the components selected to meet these requirements.

3.3.4.1. Hardware Infrastructure

The testbed's hardware infrastructure is a physical replica of the computational ecosystem described in the fire disaster scenario. It was designed to specifically address the project's central challenge of computational heterogeneity. Validating Federated Computation paradigms (Levels 4 and 5) requires infrastructure capable of orchestrating devices with different capabilities, architectures (e.g., x86 vs. ARM), and resources (e.g., GPU vs. CPU-only). Therefore, the testbed is physically divided into two categories: the central node (Server Cloud) and the peripheral nodes (Edge).
The Server Cloud (High-Performance Node) node represents the "command center" or HPC (High-Performance Computing) infrastructure in a traditional scenario. It serves a dual role:

Centralized Baseline The experiment 1 at the level 2 is where the monolithic training of the entire dataset is performed.

Hybrid Aggregator The experiment 2 at the level 4 is where the Flower aggregator resides, coordinating the edge clients.

A high-end workstation was used for these tasks, as its specifications are fundamental for establishing a performance benchmark. The NVIDIA GeForce RTX 4090 GPU (24 GB VRAM) establishes the theoretical upper limit for centralized training speed (Level 2), providing the "best-case" time against which to compare the efficiency of the federated approaches. The AMD Ryzen 9 7950X CPU, with its high core count (16c/32t), is crucial for efficiently handling parallel aggregation in the Level 4 scenario, where the server must simultaneously manage connections and incoming models from numerous heterogeneous clients.
The technical specifications of the server cloud are the following:

- **CPU**: AMD Ryzen 9 7950X (16 Cores / 32 Threads, up to 5.7 GHz)
- **GPU**: NVIDIA GeForce RTX 4090 (24 GB GDDR6X VRAM, 16384 CUDA Cores)
- **RAM**: 128 GB DDR5

Funded by the European Union
This project has received funding from the European Union's HE research and innovation programme under grant agreement No 101093003

The peripheral edge nodes are the heart of the experimentation and represents its key variable. To simulate a realistic environment, two archetypes of edge devices were employed, sitting at opposite ends of the computational spectrum. The edge node with High-Performance, called NVIDIA Jetson AGX Orin is the archetype of the "sovereign" Edge-AI node: a peripheral unit with sufficient computational capabilities not only for inference but also for local training (on-device training). Its architecture (ARM) and powerful integrated GPU (Ampere) make it an ideal, high-performance federated client. The 64 GB of unified memory is a key advantage, as it minimizes data transfer latency between the CPU and GPU during training cycles.

In the experiments, this device plays three crucial roles:

1. **Inference Client** (Level 2): executes local inference (on GPU) using the model received from the cloud.
2. **Federated Client** (Level 4 & 5): executes local training (on GPU), generating model updates.
3. **Edge Aggregator** (Level 5): this is the key component that enables the All-In-Edge scenario. Its multi-core CPU is used to run the Flower Aggregator server, demonstrating the feasibility of a complete federated cycle without any cloud dependency.

The technical specifications of the NVIDIA Jetson AGX Orin are the following:

- **CPU**: 12-core Arm Cortex-A78AE v8.2
- **GPU**: NVIDIA Ampere architecture (2048 CUDA Cores, 64 Tensor Core)
- **RAM**: 64 GB LPDDR5 (Unified Memory)

The edge node with low-power, called Raspberry Pi 5 simulates the vast network of low-cost, low-power "smart" sensors (e.g., fixed cameras, environmental sensors) that characterize a large-scale deployment. Its fundamental characteristic is the absence of a dedicated AI accelerator (GPU or NPU).

Its inclusion is methodologically critical:

1. **Tests Pure Heterogeneity**: it forces the federated system to manage clients with orders of magnitude difference in performance (Jetson Orin on GPU vs. Pi 5 on CPU).
2. **Validates "CPU-Only" Support**: it demonstrates that local training is possible, albeit slower, even on minimal hardware.
3. **Tests Aggregator Robustness**: it introduces the problem of "stragglers" (slow nodes). The federated aggregator (in both Level 4 and 5) must be configured to handle (e.g., via timeouts or asynchronous aggregation strategies) clients that take much longer to complete their local training, a typical condition in real-world scenarios.

Funded by the European Union
This project has received funding from the European Union's HE research and innovation programme under grant agreement No 101093003

The technical specifications of the Raspberry Pi 5 are the following:

- **CPU**: Broadcom BCM2712 (Quad-core Arm Cortex-A76, 64-bit)
- **RAM**: 8 GB LPDDR4X

3.3.4.2. Software Stack

A rigorously defined and standardized software stack is the only thing that makes the consistency, portability, and reproducibility of experiments possible across such a heterogeneous hardware infrastructure (x86 vs. ARM, GPU vs. CPU-only). The entire logical architecture is "cloud-native," which means it abstracts the underlying hardware through containerization. This approach guarantees that the same application code runs on every node, from cloud servers to Raspberry Pis, thus eliminating environmental variables.
The following operating systems were used as the foundation for the stack, optimized for their respective hardware to ensure maximum stability and access to necessary drivers:

- **Server Cloud: Ubuntu Server 24.04 LTS** chosen for its robust support for the x86 ecosystem, the latest NVIDIA drivers, and containerization tools.
- **NVIDIA Jetson AGX Orin: NVIDIA JetPack 6** this is a complete software package that includes an operating system based on Ubuntu 22.04 and, fundamentally, the entire stack of CUDA, cuDNN, and TensorRT libraries pre-configured for the ARM architecture.
- **Raspberry Pi 5: Raspberry Pi OS (64-bit)** chosen for its optimized support (based on Debian) for the ARM Cortex-A76 hardware.

The open-source framework Flower [25] was chosen for implementing the federated computation scenarios (Level 4 and Level 5). Flower is a framework that abstracts the complex logistics of networking and communication, relying on two components: a Server (Aggregator) that orchestrates the training rounds and implements the aggregation strategy (e.g., FedAdam), and a lightweight Client that "wraps" the local training logic on each edge node.

The choice of Flower as the federated computation framework was strategic for this experiment, based on three fundamental pillars:

- **Framework Agnosticism**: Flower is agnostic to the machine learning framework. This allowed for native and direct integration with PyTorch and the YOLOv8 architecture, without requiring modifications to the model's code.
- **Lightweight and Heterogeneity Support**: the Flower client is extremely lightweight, a key requirement for deployment on low-resource devices like the Raspberry Pi 5 (CPU-only). Flower natively handles hardware heterogeneity, allowing fast clients (Jetson on GPU) and slow clients (Pi on CPU) to coexist in the same training round, directly addressing the "stragglers" challenge.
- **Architectural Flexibility for Experimentation**: this was the winning feature. Flower allowed us to implement both the Level 4 (Cloud-Edge) and Level 5 (All-In-Edge)

Funded by the European Union
This project has received funding from the European Union's HE research and innovation programme under grant agreement No 101093003

architectures by changing only a single line of configuration: the Server's IP address. For Level 4, the clients pointed to the Server Cloud's IP; for Level 5, they pointed to the Jetson Orin designated as the aggregator. This ensured a fair experimental comparison ("apples-to-apples"), isolating the "location of aggregation" variable.

The core of the experimental setup relies on a specific combination of machine learning frameworks, models, and datasets. These components were selected to directly address the project's key challenges: ensuring real-time performance, enabling deployment on resource-constrained edge devices, and building a robust, reproducible federated computation pipeline.

- **AI Framework (PyTorch)**: the base framework for all model development and training was PyTorch. It was chosen for its flexibility, excellent support for GPU acceleration via CUDA, and its predominance in research, which facilitates the integration of state-of-the-art models. The PyTorch ecosystem is also rapidly expanding into "on-device" computation, making it a forward-looking choice for inference and training on the edge.
- **CNN Model (YOLOv8n)**: the reference model selected for smoke and fire detection is **YOLOv8** [111]. It is a state-of-the-art, single-stage CNN architecture for object detection which, unlike two-stage detectors, treats inference as a single regression problem, examining the image only once. This design is intrinsically faster and more suitable for real-time applications. For these experiments, its smallest and fastest variant, YOLOv8n (nano), was chosen. This decision directly addresses the central challenge identified in the state-of-the-art: the deployment of "computationally intensive deep learning models in resource-constrained edge environments." YOLOv8n is a model optimized for the edge, designed to have a low number of parameters and a low computational cost (GFLOPs). It offers the best trade-off between accuracy (mAP) and inference speed (FPS) on resource-constrained hardware. Its compact architecture also makes it ideal for federated computation, as a smaller model minimizes the communication overhead (weights to be transferred) in each round.
- **Dataset (Fire and Smoke Detection Dataset)**: the public dataset FASDD (Fire and Smoke Detection Dataset) [26] was used for training and validating the models. This is a large-scale dataset (over 100,000 samples) composed of heterogeneous smoke and flame images, which contains bounding box annotations for the "Fire" and "Smoke" classes. The use of this specific public dataset is a methodological pillar for the experiment. First, using a "state-of-the-art" and peer-reviewed dataset ensures the scientific reproducibility of this deliverable. Second, its diversity and richness (urban, industrial, forest, and nighttime scenes) were leveraged to simulate the most critical challenge of federated computation: non-IID (Non-Identical and Independently Distributed) partitioning.

The dataset was divided using a simple random split, resulting in a near IID distribution

Funded by the European Union
This project has received funding from the European Union's HE research and innovation programme under grant agreement No 101093003

of samples across clients. While this simplifies the federated learning setup, it may not fully capture the challenges posed by real-world non-IID data distributions encountered in edge environments.

The experimental setup supporting KPI OA4.1 validation is available as open-source software. The design, implementation, and benchmarking scripts are documented in [112].

3.3.5. Experimental Methodology and Test Scenarios

A precise experimental methodology was defined to validate KPI OA4.1 and rigorously compare the Level 2, 4, and 5 architectures defined in the Experimental Setup. This section describes how the tests were performed and specifies the operating conditions, parameters, and metrics measured to ensure reproducibility and fair comparisons. All experiments started with a common pre-trained base model ($Model_0$). Then, a dedicated experimental dataset was used to train the model further. Specific test scenarios were designed to isolate the impact of key variables on system performance during training on the experimental dataset. The variables included the deployment architecture (centralized versus federated), the type of hardware used (CPU versus GPU), and system scalability (eight versus 16 clients).

As described below, the main metrics collected for each test include the total effective time required to train on the experimental dataset, the accuracy achieved by the final model, and the inference latency measured on a standard edge device. These tests provide the quantitative data necessary for a complete evaluation of the different architectures and for discussing the results, focusing on validating KPI OA4.1.

3.3.5.1. Data Preparation and Base Model Definition

To ensure a fair comparison between the approaches, it was crucial to start from a common point of knowledge. To this end, a base model named $Model_0$ was created first. This model was obtained by pre-training the YOLOv8n architecture on the first subset of the FASDD dataset (Dataset A), which consists of 8,000 training images, 1,000 validation images, and 1,000 test images. Next, a second distinct dataset (Dataset B) was prepared. This dataset was also extracted from FASDD and had the same split: 8,000 images for training, 1,000 for validation, and 1,000 for testing. Dataset B served as the "playing field" for all comparative training tests, whether centralized or federated, and always started from $Model_0$.

3.3.5.2. Experimental Dataset Partitioning Methodology

To directly compare temporal efficiency, the training set from Dataset B (8,000 images) was distributed differently depending on the scenario. In the centralized approach (Level

Funded by the European Union
This project has received funding from the European Union's HE research and innovation programme under grant agreement No 101093003

2), the entire set was treated as a single block, simulating the transfer of the set from eight or 16 clients to a cloud server. In the federated approaches (Levels 4 and 5), however, the images were divided equally among the participating clients (N = 8 or 16). For instance, in the eight-client test, each client received 1,000 unique images from Dataset B, ensuring that each federated client works with the same amount of data and similar statistical distributions. This provides a clear baseline for measuring computational efficiency and the convergence of the federated algorithm as hardware and the number of clients vary.

3.3.5.3. Evaluating Inference Latency on Edge Devices

To evaluate the system's responsiveness in the field, the inference latency metric ($T_{inference}$) was measured directly on edge devices performing the predictions. The methodology was differentiated to compare performance across the different scenarios:

- **Centralized Scenario (Level 2):** in this case, the model trained in the cloud is distributed to the edge nodes for inference. To simulate a low-cost device, the inference latency ($T_{inference}$) was measured on the Raspberry Pi 5
- **Federated Scenarios (Level 4 and 5):** in these scenarios, the edge clients participating in the training are also the devices performing the inference. Therefore, the inference latency ($T_{inference}$) was measured on the clients used in each test:
  - In tests with NVIDIA Jetson AGX Orin clients, inference was performed on the Jetson Orin
  - In tests with Raspberry Pi 5 clients, inference was performed on the Raspberry Pi 5

This approach allows for evaluating the actual inference latency on the specific hardware used in each federated configuration, providing a realistic measure of the system's responsiveness in the field.

3.3.5.4. Definition of Model Accuracy Metrics

To evaluate the quality of the models trained in the different scenarios, standard metrics in the field of object detection were used:

- **Precision:** indicates the percentage of correct detections among all detections made by the model. High precision means the model produces few *false positives* (detecting a fire where there is none). It is calculated as: $\text{Precision} = \frac{TP}{TP+FP}$ (TP = True Positives, FP = False Positives).
- **Recall:** indicates the percentage of actual objects (e.g., fires) that the model was effectively able to detect. High recall means the model produces few *false negatives* (missing an existing fire). It is calculated as: $\text{Recall} = \frac{TP}{TP+FN}$ (FN = False Negatives).
- **mAP50 (mean Average Precision @ IoU=0.50):** this is a composite metric that evaluates the overall accuracy of the model considering both precision and recall.

Funded by the European Union
This project has received funding from the European Union's HE research and innovation programme under grant agreement No 101093003

It is calculated as the area under the Precision-Recall curve, considering a detection correct if the predicted bounding box has an overlap (Intersection over Union - IoU) of at least 50% with the ground truth bounding box. It provides a robust measure of the model's ability to correctly identify objects.

- **mAP50-95:** this is a stricter metric than mAP50. It is calculated by averaging the mAP values obtained at different IoU thresholds, from 0.50 to 0.95 (with increments of 0.05). A high mAP50-95 value indicates that the model not only correctly detects objects but also does so with very precise localization (the predicted bounding boxes are very similar to the real ones).

3.3.5.5. Definition of Key Temporal Performance and Efficiency Metrics

To quantify the temporal performance of the different architectures, the following key metrics were defined:

- $\mathrm{T}_{\text{upload\_data}}$ (Centralized Overhead): represents the total effective time, measured from start to finish, required for all clients (N=8 or N=16) to upload their raw data portion (from 'Dataset B') to the cloud server
- $\mathrm{T}_{\text{training\_centralized}}$**:** indicates the total effective time taken by the cloud server to perform 10 training epochs on the entire 'Dataset B' (8,000 images), starting from the Model_0 model
- $\mathrm{T}_{\text{update\_centralized}}$**:** this is the metric defining the complete lifecycle time for the Level 2 architecture, calculated as: $\mathrm{T}_{\text{update\_centralized}} = \mathrm{T}_{\text{upload\_data}} + \mathrm{T}_{\text{training\_centralized}}$
- $\mathrm{T}_{\text{update\_federated}}$**:** measures the total effective time required to complete the entire federated training process (3 rounds, each with 3 local epochs), always starting from Model_0
- $\mathrm{T}_{\text{inference}}$**:** evaluates the responsiveness in the field, measuring the average time (in milliseconds) needed to process a batch of 5 test images using the final model produced by each experiment, executed on the Raspberry Pi 5 device

3.3.5.6. Executed and Planned Test Scenarios

Based on the defined metrics, the following test scenarios were executed:

- **Scenario 1: Centralized Baseline (Level 2):** Quantifies the performance of the Cloud-Legacy "upload-then-train" approach. To simulate different data origin conditions, the data upload time ($\mathrm{T}_{\text{upload\_data}}$) was measured for both 8 and 16 clients. Subsequently, the training time ($\mathrm{T}_{\text{training\_centralized}}$) was measured on the cloud server for 10 epochs in two distinct hardware configurations: Test 1.1 (GPU) executed on the server's GPU (NVIDIA RTX 4090), and Test 1.2 (CPU) executed on the server's CPU (AMD Ryzen 9). Inference ($\mathrm{T}_{\text{inference}}$) for both resulting models was measured on the Raspberry Pi 5.
- **Scenario 2: Federated Computation (Level 4 - Cloud Aggregator):** Evaluates the hybrid architecture, using the Cloud Server as the aggregator (FedAdam) and the NVIDIA Jetson AGX Orin devices as clients. The time $\mathrm{T}_{\text{update\_federated}}$ was measured

Funded by the European Union
This project has received funding from the European Union's HE research and innovation programme under grant agreement No 101093003

for 3 rounds (3 local epochs/round), starting from Model_0. Tests were performed varying the number of clients and the hardware used for local training on the Jetson devices (GPU - Test 2.1 [8 clients] and 2.2 [16 clients]; CPU - Test 2.3 [8 clients] and 2.4 [16 clients]). For each test, accuracy and $\mathrm{T}_{\mathrm{inference}}$ on the corresponding client device (Jetson) were also recorded.

- **Scenario 3: Federated Computation (Level 5 - Edge Aggregator):** Evaluates the All-In-Edge architecture, using an NVIDIA Jetson AGX Orin as the aggregator and the Raspberry Pi 5 devices as clients (CPU-only). The objective is to measure $\mathrm{T}_{\mathrm{update\_federated}}$ and compare it with Scenario 2, testing configurations with 8 clients (Test 3.1) and 16 clients (Test 3.2). Accuracy and $\mathrm{T}_{\mathrm{inference}}$ on the Raspberry Pi 5 were recorded.

The detailed results of these tests and their comparative analysis regarding KPI OA4.1 are presented in the following section.

3.3.6. Results and Discussion

This Section presents and analyzes the quantitative results obtained from the experimental scenarios defined in the methodology. The primary objective is to provide a rigorous empirical evaluation and a direct comparison between the different computing architectures explored **Cloud-Legacy (Level 2)**, **Cloud-Edge (Level 4)**, and **All-In-Edge (Level 5)** within the specific context of the "Fire Disaster Scenario".

While the experiments were conducted on a stable lab network to ensure controlled conditions, it is important to note that variable network latency and bandwidth also affect the centralized computing model. In centralized architectures, the need to transmit raw data to the cloud can introduce significant delays and bottlenecks, especially in field deployments with limited connectivity. Federated learning architectures alleviate this issue by exchanging model parameters instead of raw data, reducing communication overhead and improving robustness to network variability. However, fluctuating network conditions remain a challenge for both paradigms, emphasizing the need for adaptive strategies to maintain performance in real-world scenarios.

The analysis focuses on the key metrics collected, which were selected to measure the fundamental aspects required by the TEMA project:

1. **Model Accuracy:** Evaluating whether the distributed approaches maintain a performance level (Precision, Recall, mAP) comparable to the centralized baseline.
2. **Temporal Efficiency:** Quantifying the total effective times required to update the model ($\mathrm{T}_{\mathrm{update\_centralized}}$ vs $\mathrm{T}_{\mathrm{update\_federated}}$), which is central to the validation of **KPI OA4.1**.
3. **Inference Latency:** Measuring the system's responsiveness in the field ($\mathrm{T}_{\mathrm{inference}}$) on the specified edge devices.

Funded by the European Union
This project has received funding from the European Union's HE research and innovation programme under grant agreement No 101093003

The trade-offs between the centralized and federated approaches is examined, with particular attention to the critical impact of the available **hardware** (CPU vs. GPU) and **system scalability** (8 vs. 16 clients). The presented data allow for drawing conclusions about the effectiveness and efficiency of federated computation within the TEMA framework, validating its suitability for the emergency management scenario in terms of speed, accuracy, and compliance with operational constraints.

3.3.6.1. Test 1.1 (GPU, Centralized Baseline)

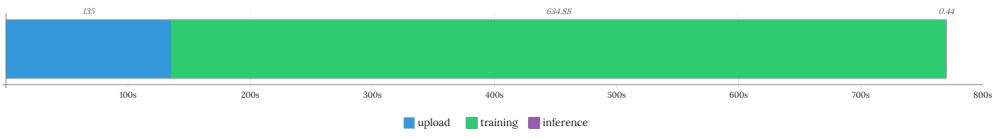
This test established the performance baseline for the Cloud-Legacy (Level 2) approach in a GPU-accelerated environment. Training was executed on the cloud server (NVIDIA RTX 4090) using the entire Dataset B (8,000 images), starting from the $Model_0$ model and training for 10 epochs. To assess the impact of data origin, the data upload time (T_{upload_data}) was measured simulating scenarios with both 8 and 16 edge clients.

The time required for data upload was approximately 135 seconds when originating from 8 clients, and 75 seconds when originating from 16 clients. The effective time taken for the centralized training computation on the GPU ($T_{training\_centralized}$) was constant at 634.88 seconds. Finally, the inference latency ($T_{inference}$) of the resulting model on the target edge device (Raspberry Pi 5), measured on a batch of 5 images, was 440.5 ms (approximately 0.44 seconds).

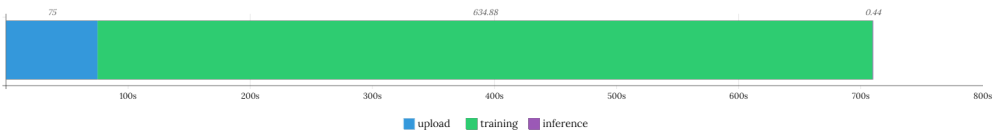
Consequently, the total effective time to complete the entire centralized update cycle ($T_{update\_centralized}$), from data upload to the first inference, was approximately:

- 770.32 seconds (135s + 634.88s + 0.44s) for the 8-client origin scenario.
- 710.32 seconds (75s + 634.88s + 0.44s) for the 16-client origin scenario.

Figure 7. Analysis of Temporal Components - Test 1.1 (GPU)



(a) Originating from 8 clients

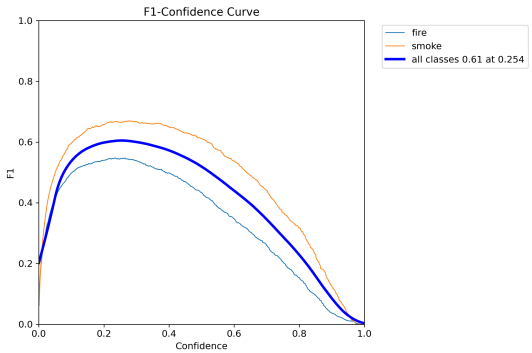


(b) Originating from 16 clients

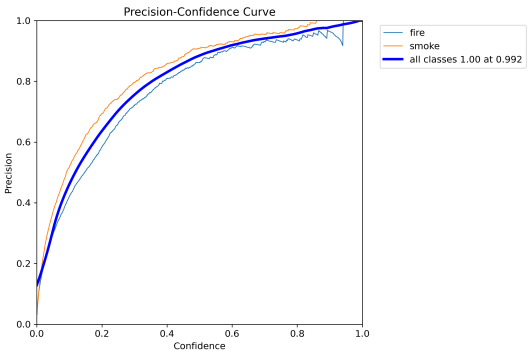
Funded by the European Union
This project has received funding from the European Union's HE research and innovation programme under grant agreement No 101093003

From the perspective of model performance, this configuration achieved the highest accuracy among all tests, with an **mAP50-95 value of 0.3733**. Further analyses of the model's performance, such as the F1, Precision, and Recall confidence curves and the Precision-Recall curve (Figures 8a, 8b, 8c, and 8d), provide deeper insights into the model's robustness.

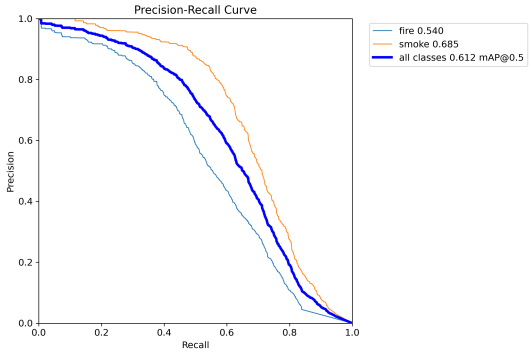
Figure 8. Model Performance Curves - Test 1.1 (Centralized GPU)



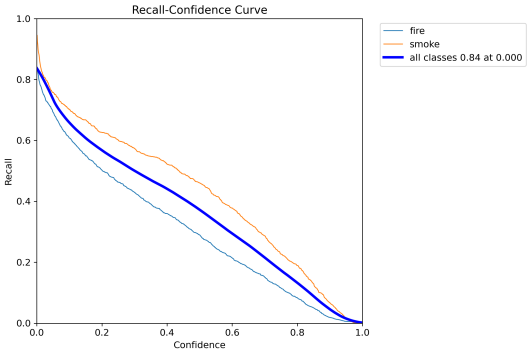
(a) F1-Confidence Curve



(b) Precision-Confidence Curve



(c) Precision-Recall Curve



(d) Recall-Confidence Curve

3.3.6.2. Test 1.2 (CPU, Centralized Baseline)

This test established the performance baseline for the Cloud-Legacy (Level 2) approach using only CPU resources on the cloud server (AMD Ryzen 9). Training was executed using the entire Dataset B (8,000 images), starting from the $Model_0$ model and training for 10 epochs. To assess the impact of data origin, the data upload time (T_{upload_data}) was measured simulating scenarios with both 8 and 16 edge clients.

The time required for data upload was approximately 135 seconds when originating from 8 clients, and 75 seconds when originating from 16 clients. The effective time

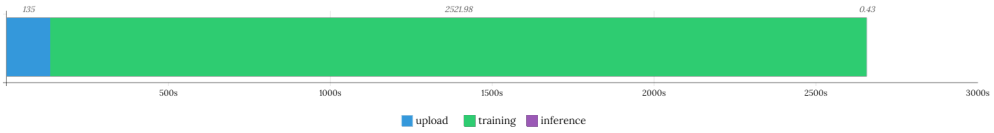
Funded by the European Union
This project has received funding from the European Union's HE research and innovation programme under grant agreement No 101093003

taken for the centralized training computation on the CPU ($T_{training\_centralized}$) was significantly longer, constant at 2521.98 seconds (approximately 42 minutes). Finally, the inference latency ($T_{inference}$) of the resulting model on the target edge device (Raspberry Pi 5), measured on a batch of 5 images, was 426 ms (approximately 0.43 seconds).

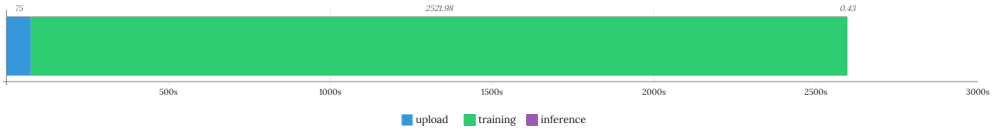
Consequently, the total effective time to complete the entire centralized update cycle ($T_{update\_centralized}$), from data upload to the first inference, was approximately:

- 2657.42 seconds (135s + 2521.98s + 0.44s) for the 8-client origin scenario.
- 2597.41 seconds (75s + 2521.98s + 0.43s) for the 16-client origin scenario.

Figure 9. Analysis of Temporal Components - Test 1.2 (CPU)



(a) Originating from 8 clients

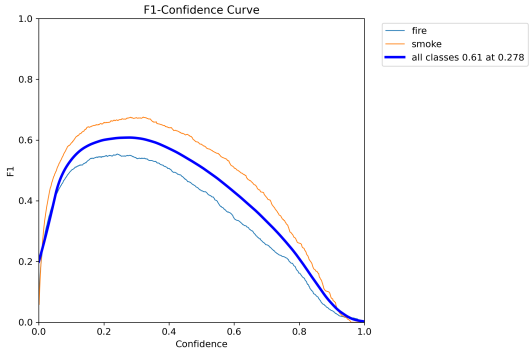


(b) Originating from 16 clients

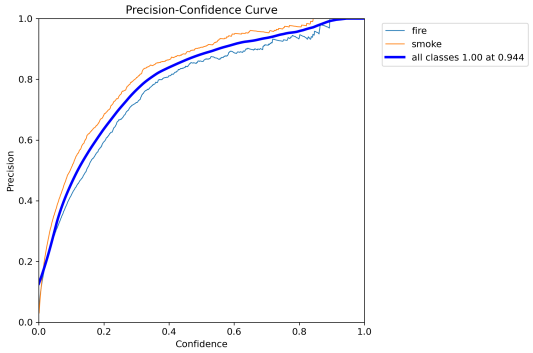
From the perspective of model performance, this CPU-trained configuration achieved high accuracy, very close to the GPU-trained counterpart, with an mAP50-95 value of 0.3699. Further analyses of the model's performance, such as the F1, Precision, and Recall confidence curves and the Precision-Recall curve (Figure 10), provide deeper insights into the model's robustness.

Funded by the European Union
This project has received funding from the European Union's HE research and innovation programme under grant agreement No 101093003

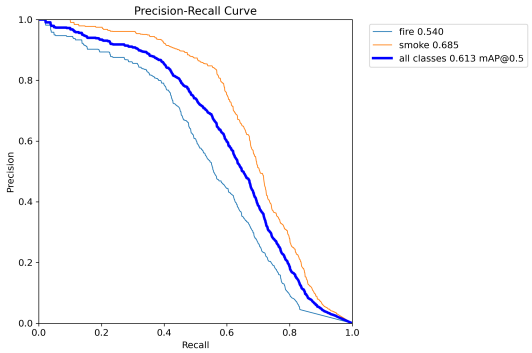
Figure 10. Model Performance Curves - Test 1.2 (Centralized CPU)



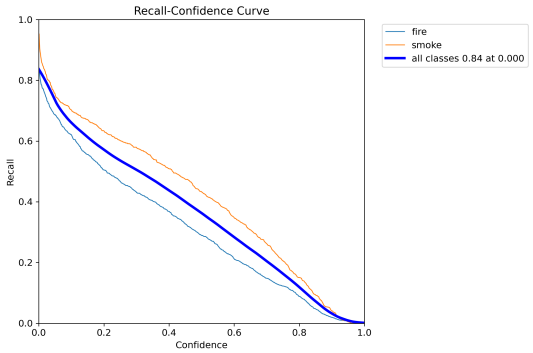
(a) F1-Confidence Curve



(b) Precision-Confidence Curve



(c) Precision-Recall Curve



(d) Recall-Confidence Curve

3.3.6.3. Test 2.1 (GPU, Federated - 8 Clients)

This test evaluated the hybrid **Cloud-Edge (Level 4)** architecture using the Cloud Server as the aggregator (FedAdam) and **8 NVIDIA Jetson AGX Orin** devices as clients. Local training on each client was performed using its integrated **GPU** for 3 local epochs per round, over a total of 3 rounds, starting from the $Model_0$ model.

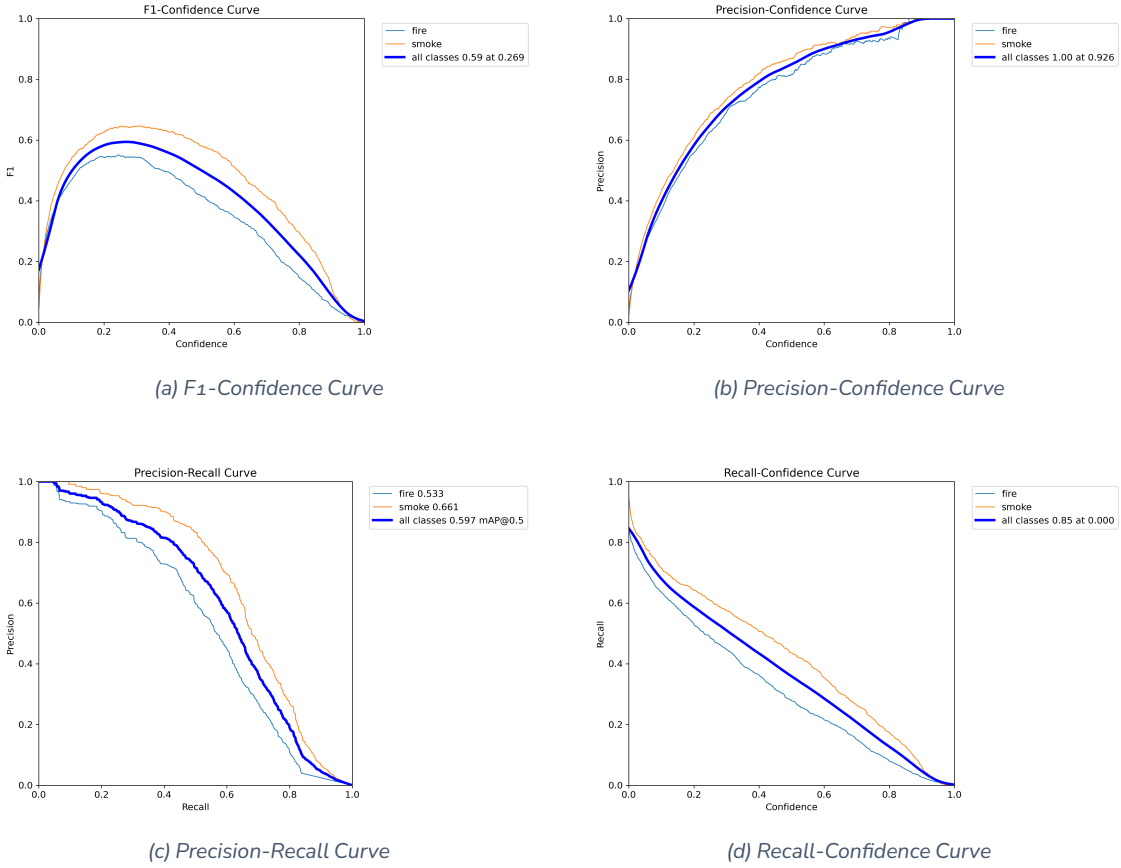
Regarding the temporal performance, the **total effective time to complete the entire federated update process** ($T_{update\_federated}$) encompassing local training, communication, aggregation across 3 rounds, and final evaluation was **641.59 seconds**. Subsequently, the inference latency ($T_{inference}$) of the model produced by this process, measured on the client device (Jetson Orin GPU), averaged approximately **431 ms** (0.43 seconds) for a batch of 5 images.

From the perspective of model performance, the resulting model achieved an mAP50-95 accuracy of 0.3542. Further analyses of the model's performance, such as the F1, Pre-

Funded by the European Union
This project has received funding from the European Union's HE research and innovation programme under grant agreement No 101093003

cision, and Recall confidence curves and the Precision-Recall curve (Figure 11), provide deeper insights into the model's robustness.

Figure 11. Model Performance Curves - Test 2.1 (Federated GPU, 8 Clients)



(a) F1-Confidence Curve

(b) Precision-Confidence Curve

(c) Precision-Recall Curve

(d) Recall-Confidence Curve

3.3.6.4. Test 2.2 (GPU, Federated - 16 Clients)

This test scaled the previous setup (Test 2.1) to **16 NVIDIA Jetson AGX Orin** clients, maintaining the hybrid **Cloud-Edge (Level 4)** architecture with a Cloud Server aggregator (FedAdam). Local training on each client was again performed using its integrated **GPU** for 3 local epochs per round, over a total of 3 rounds, starting from the $Model_0$ model.

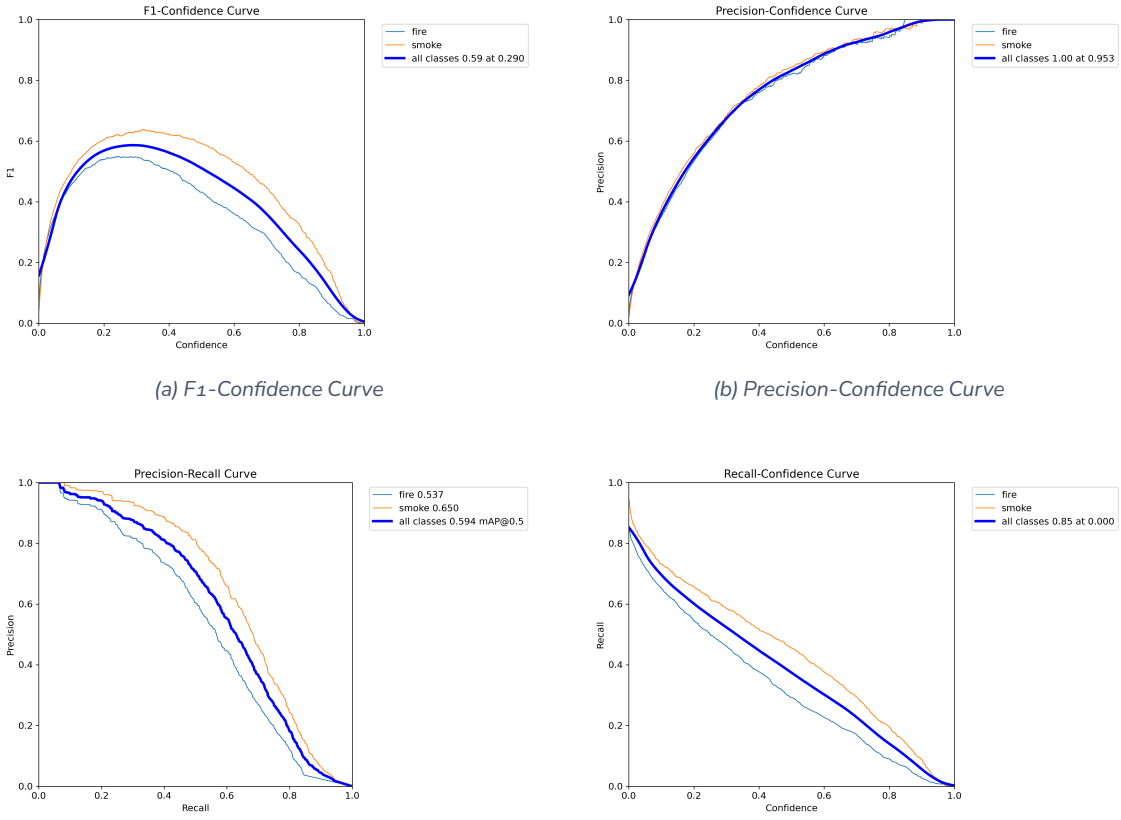
Regarding the temporal performance, increasing the number of parallel clients significantly reduced the **total effective time to complete the entire federated update process** ($T_{update\_federated}$). This time, encompassing local training, communication, aggregation, and final evaluation, was **438.98 seconds**. Subsequently, the inference latency ($T_{inference}$) of the model produced by this process, measured on the client device (Jet-

Funded by the European Union
This project has received funding from the European Union's HE research and innovation programme under grant agreement No 101093003

son Orin GPU), averaged approximately **471 ms** (0.47 seconds) for a batch of 5 images.

From the perspective of model performance, the final model accuracy showed a minimal decrease compared to the 8-client test, achieving an **mAP50-95 accuracy of 0.3483**. Other key metrics included Precision at 0.6778 and Recall at 0.5237. Further performance details and analysis are available in the corresponding performance curves (Figure 12).

Figure 12. Model Performance Curves - Test 2.2 (Federated GPU, 16 Clients)



(a) F1-Confidence Curve

(b) Precision-Confidence Curve

(c) Precision-Recall Curve

(d) Recall-Confidence Curve

3.3.6.5. Test 2.3 (CPU, Federated 8 Clients)

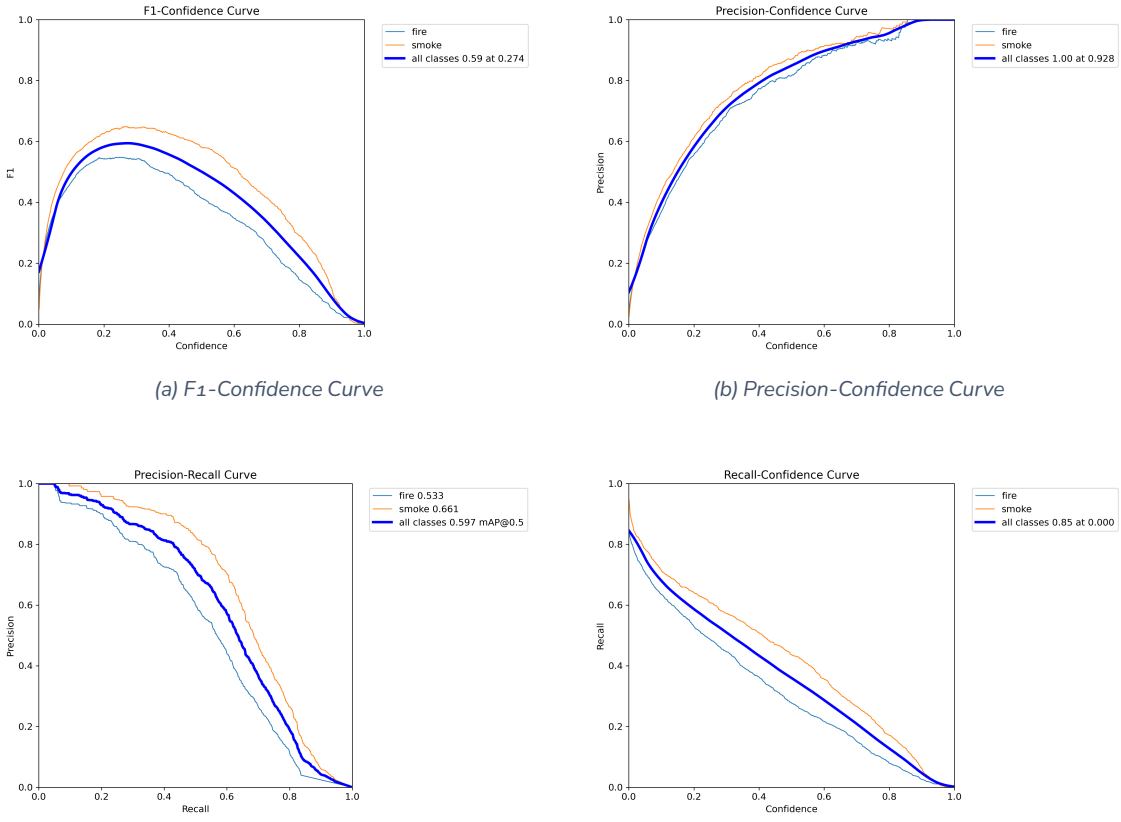
This test evaluated the hybrid **Cloud-Edge (Level 4)** architecture using the Cloud Server as the aggregator (FedAdam) and **8 NVIDIA Jetson AGX Orin** devices as clients. However, in this configuration, local training on each client was performed using only its integrated **CPU** for 3 local epochs per round, over a total of 3 rounds, starting from the $Model_0$ model.

Funded by the European Union
This project has received funding from the European Union's HE research and innovation programme under grant agreement No 101093003

Regarding the temporal performance, executing training on the CPU dramatically increased the **total effective time to complete the entire federated update process** ($T_{update\_federated}$). This time, encompassing local training, communication, aggregation, and final evaluation, was **6347.01 seconds**. Subsequently, the inference latency ($T_{inference}$) of the model produced by this process, measured on the client device (Jetson Orin CPU), averaged approximately **572 ms** (0.57 seconds) for a batch of 5 images.

From the perspective of model performance, the final model accuracy remained comparable to the GPU tests, achieving an **mAP50-95 accuracy of 0.3542**. Other key metrics included Precision at 0.6841 and Recall at 0.5288. Further performance details and analysis are available in the corresponding performance curves (Figure 13).

Figure 13. Model Performance Curves - Test 2.3 (Federated CPU, 8 Clients)



(a) F1-Confidence Curve

(b) Precision-Confidence Curve

(c) Precision-Recall Curve

(d) Recall-Confidence Curve

Funded by the European Union

This project has received funding from the European Union's HE research and innovation programme under grant agreement No 101093003

3.3.6.6. Test 2.4 (CPU, Federated - 16 Clients)

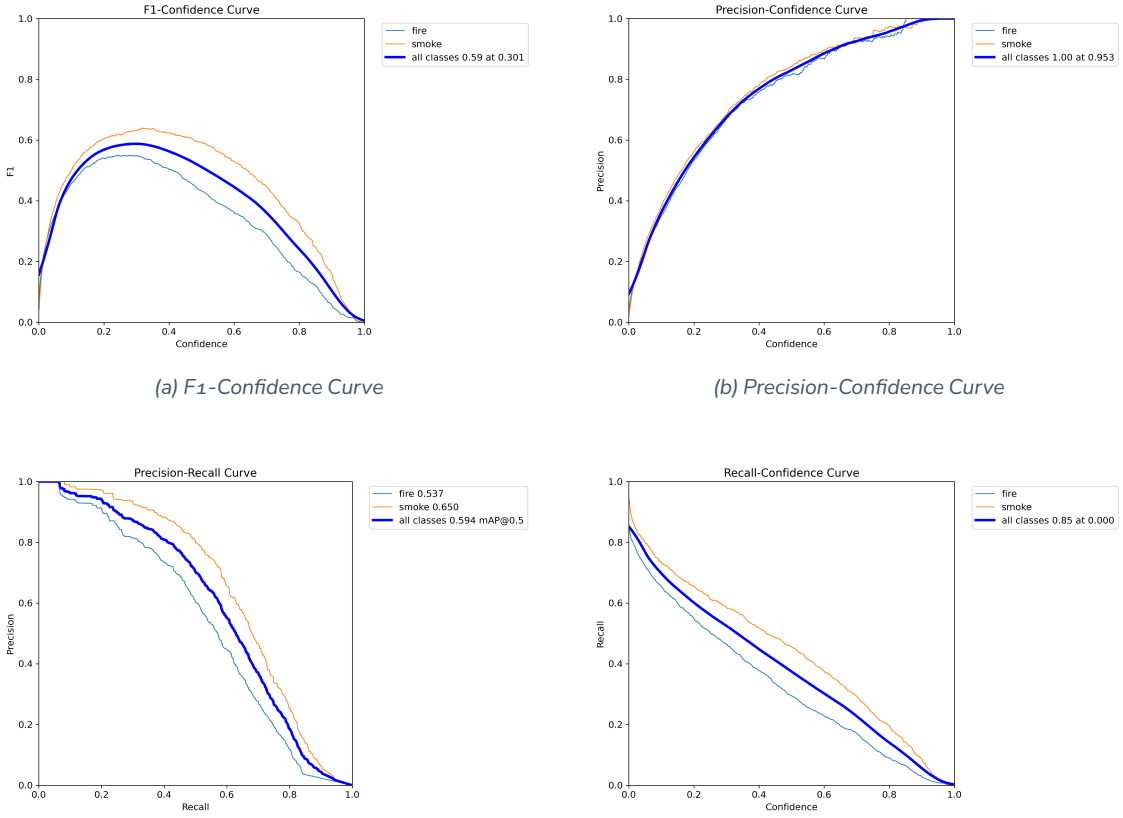
This test scaled the CPU-only federated setup (Test 2.3) to **16 NVIDIA Jetson AGX Orin** clients, maintaining the hybrid **Cloud-Edge (Level 4)** architecture with a Cloud Server aggregator (FedAdam). Local training on each client was performed using only its integrated **CPU** for 3 local epochs per round, over a total of 3 rounds, starting from the $Model_0$ model.

Regarding the temporal performance, increasing the number of parallel clients reduced the **total effective time to complete the entire federated update process** ($T_{update\_federated}$) compared to the 8-client CPU test. This time, encompassing local training, communication, aggregation, and final evaluation, was **3298.93 seconds**. Subsequently, the inference latency ($T_{inference}$) of the model produced by this process, measured on the client device (Jetson Orin CPU), averaged approximately **583 ms** (0.58 seconds) for a batch of 5 images.

From the perspective of model performance, the final model accuracy remained consistent with other federated tests, achieving an **mAP50-95 accuracy of 0.3484**. Other key metrics included Precision at 0.6603 and Recall at 0.5304. Further performance details and analysis are available in the corresponding performance curves (Figure 14).

Funded by the European Union
This project has received funding from the European Union's HE research and innovation programme under grant agreement No 101093003

Figure 14. Model Performance Curves - Test 2.4 (Federated CPU, 16 Clients)



(a) F1-Confidence Curve

(b) Precision-Confidence Curve

(c) Precision-Recall Curve

(d) Recall-Confidence Curve

3.3.6.7. Test 3.1 (CPU, Federated Edge-Only - 8 Clients)

This test allows the evaluation of the **All-In-Edge (Level 5)** architecture, designed for complete autonomy from the cloud. An **NVIDIA Jetson AGX Orin** device acted as the aggregator (using FedAdam), while **8 Raspberry Pi 5** devices served as clients, performing local training using only their **CPUs**. The federated update time ($T_{update\_federated}$) was measured over 3 rounds (3 local epochs each), starting from the $Model_0$ model.

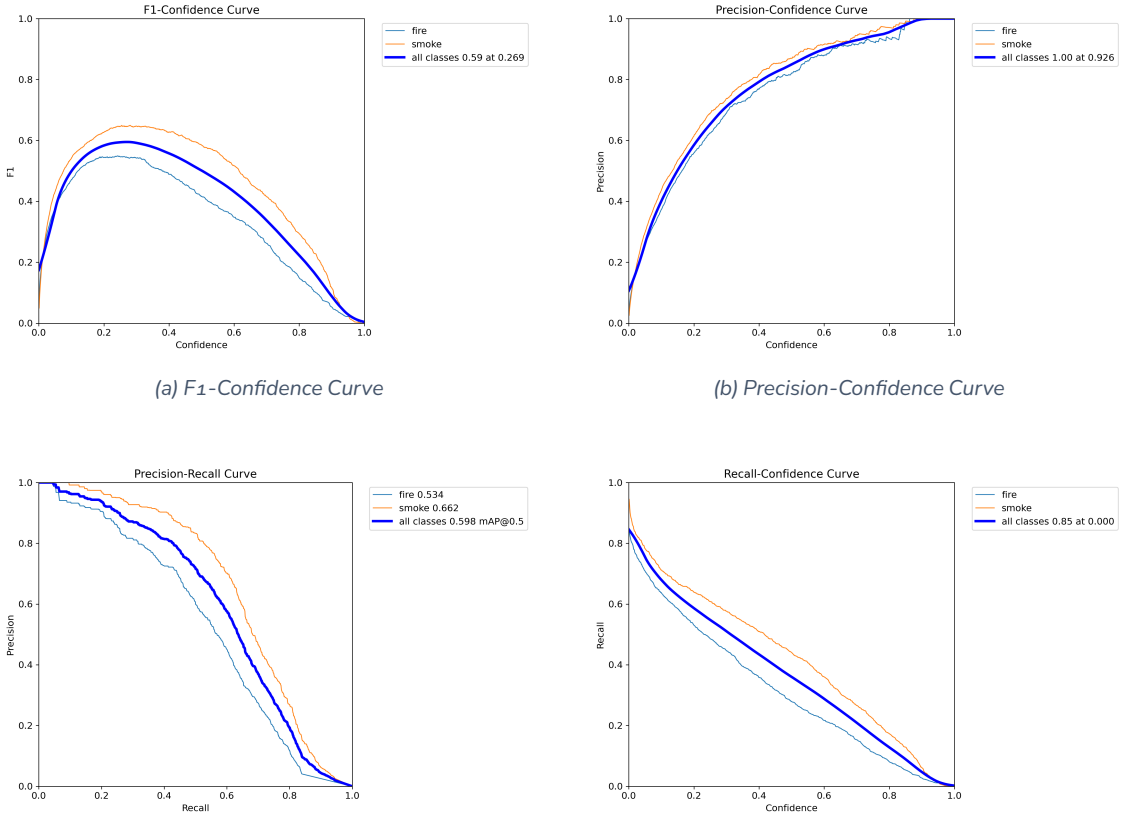
Regarding the temporal performance, the **total effective time to complete the entire federated update process** ($T_{update\_federated}$) in this fully edge-based configuration, including local training, communication handled by the edge aggregator, and final evaluation, was **3587.89 seconds**. Subsequently, the inference latency ($T_{inference}$) of the model produced by this process, measured on the client device (Raspberry Pi 5), averaged approximately **478 ms** (0.48 seconds) for a batch of 5 images.

From the perspective of model performance, the resulting model achieved an **mAP50-95**

Funded by the European Union
This project has received funding from the European Union's HE research and innovation programme under grant agreement No 101093003

accuracy of 0.3541. Other key metrics included Precision at 0.6771 and Recall at 0.5323. This demonstrates that the All-In-Edge configuration can achieve accuracy comparable to other federated setups, even with low-power clients and an edge-based aggregator. Further performance details are available in the corresponding performance curves (Figure 15).

Figure 15. Model Performance Curves - Test 3.1 (Federated CPU, 8 Clients, Level 5)



(a) F1-Confidence Curve

(b) Precision-Confidence Curve

(c) Precision-Recall Curve

(d) Recall-Confidence Curve

3.3.6.8. Test 3.2 (CPU, Federated Edge-Only - 16 Clients)

This test allows to scale the **All-In-Edge (Level 5)** architecture to **16 Raspberry Pi 5** clients, again using an **NVIDIA Jetson AGX Orin** device as the aggregator (FedAdam). Local training on each client was performed using only its **CPU** for 3 local epochs per round, over a total of 3 rounds, starting from the $Model_0$ model.

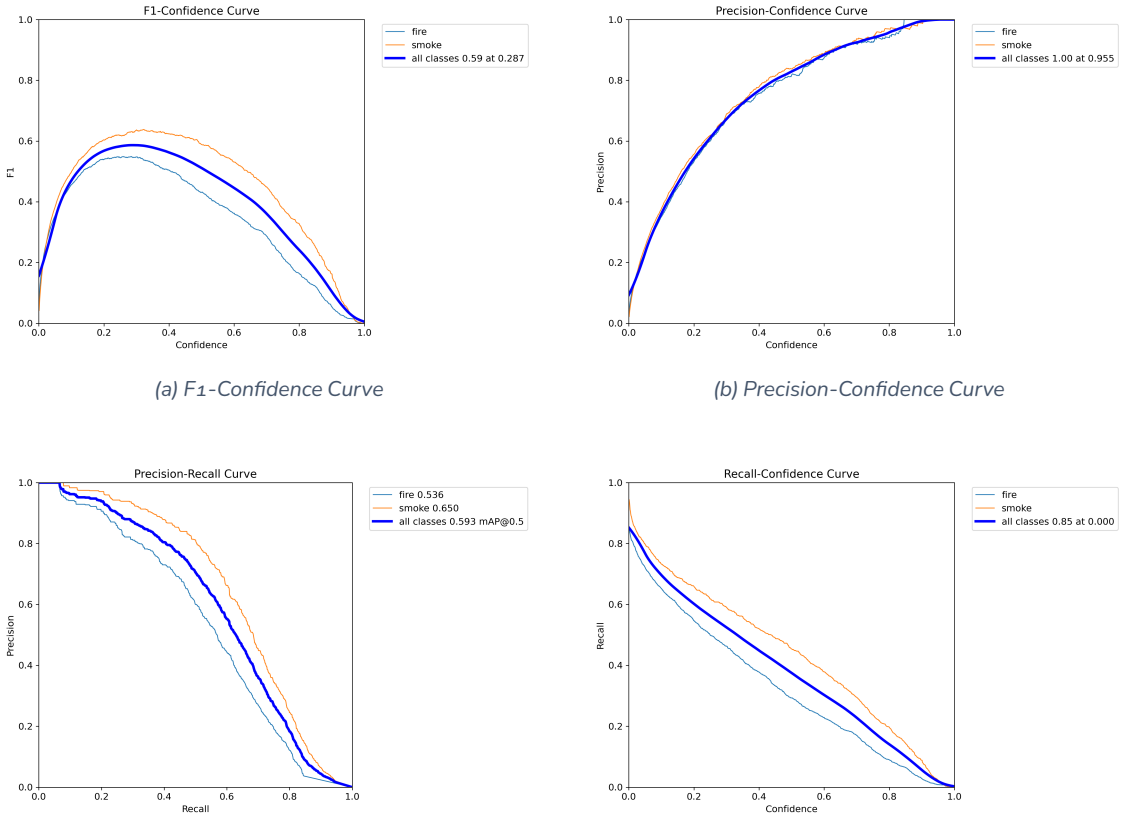
Regarding the temporal performance, increasing the number of parallel clients significantly reduced the **total effective time to complete the entire federated update process** ($T_{update\_federated}$) compared to the 8-client Level 5 test. This time, encompassing

Funded by the European Union
This project has received funding from the European Union's HE research and innovation programme under grant agreement No 101093003

local training, communication handled by the edge aggregator, and final evaluation, was **1998.11 seconds**. Subsequently, the inference latency ($T_{inference}$) of the model produced by this process, measured on the client device (Raspberry Pi 5), averaged approximately **429.5 ms** (0.43 seconds) for a batch of 5 images.

From the perspective of model performance, the final model accuracy remained consistent with other federated tests, achieving an **mAP50-95 accuracy of 0.3486**. Other key metrics included Precision at 0.6565 and Recall at 0.5366. This further confirms that the All-In-Edge configuration can achieve competitive accuracy. Further performance details are available in the corresponding performance curves (Figure 16).

Figure 16. Model Performance Curves - Test 3.2 (Federated CPU, 16 Clients, Level 5)



(a) F1-Confidence Curve

(b) Precision-Confidence Curve

(c) Precision-Recall Curve

(d) Recall-Confidence Curve

3.3.6.9. Model Accuracy: Centralized vs. Federated

A fundamental aspect to evaluate is whether adopting a distributed training paradigm, such as Federated Computation, maintains an accuracy comparable to that of traditional centralized training.

Funded by the European Union
This project has received funding from the European Union's HE research and innovation programme under grant agreement No 101093003

Analyzing the accuracy metrics obtained at the end of each experimental scenario reveals that **centralized** training (Level 2) generally establishes the benchmark for overall performance, typically achieving the highest values for stricter metrics like **mAP50-95**. The models generated through the **federated** approaches (Level 4 and Level 5) demonstrate **highly comparable** performance, achieving similar results and occasionally even slightly surpassing the centralized baseline in specific metrics such as Recall, depending on the configuration.

It is crucial to emphasize that this **small difference** observed in the quantitative metrics **did not translate into a significant functional difference** during qualitative inference tests. Indeed, running inference with the different models (centralized and federated) on the same test images resulted in **practically identical behavior**. As illustrated in Figure 17, which shows the detection results on the image for the **centralized model (Test 1.1)**, the **federated Level 4 model (Test 2.1)**, and the **federated Level 5 model (Test 3.2)**, respectively, both types of models produced the **same detections** (same labels 'fire' and 'smoke' and nearly overlapping bounding boxes) for the objects present. The visual results are, for all practical purposes, indistinguishable.

Figure 17. Visual comparison of inference results on image for models trained centrally (a), federally Level 4 (b), and federally Level 5 (c).

(a) Centralized Level 2 (Test 1.1) *(b) Federated Level 4 (Test 2.1)* *(c) Federated Level 5 (Test 3.2)*

This key result indicates that federated computation is **effective** in producing robust and performative models, with accuracy that is substantially **comparable** to the centralized approach in practical application. The slight metric gap is considered **fully acceptable** in light of the benefits in terms of privacy and temporal efficiency.

3.3.6.10. Temporal Efficiency and KPI Validation

KPI OA4.1 targeted a 10-15% reduction in update time compared to the traditional approach. The total effective update time $T_{update\_centralized}$ (including data upload, training, and first inference) is compared with $T_{update\_federated}$ (including federated rounds and first inference).

Funded by the European Union
This project has received funding from the European Union's HE research and innovation programme under grant agreement No 101093003

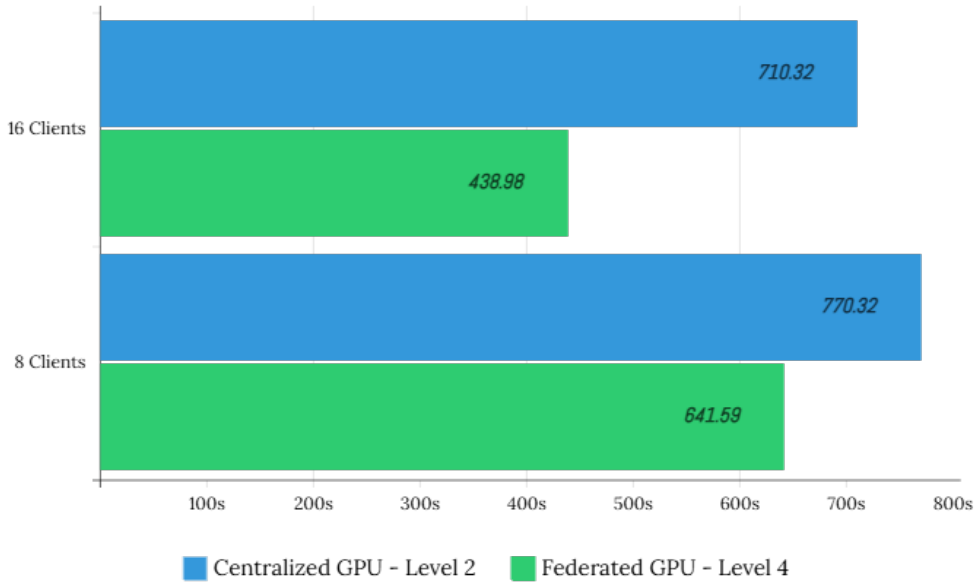


Figure 18. Total Update Time Comparison - GPU Accelerated Environment

Figure 18 illustrates the comparison in the **GPU-Accelerated** environment. It shows the absolute times for the centralized baseline (blue bars, originating from 8 and 16 clients) versus the Level 4 federated approach (green bars, with 8 and 16 clients). The federated approach becomes significantly faster as the number of clients increases.

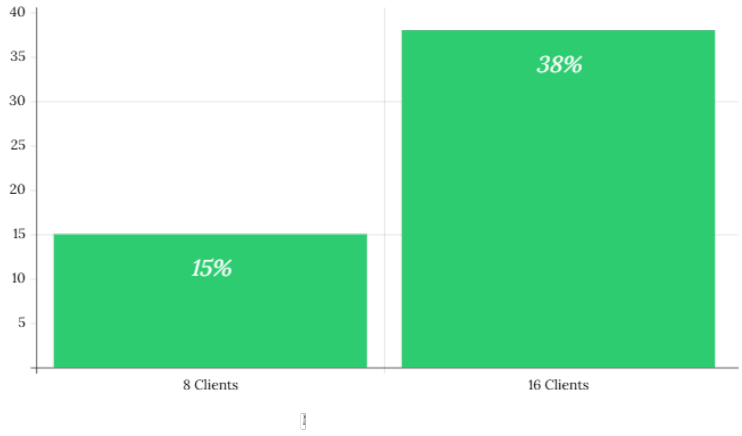


Figure 19. KPI Validation: Percentage Time Reduction (Federated GPU vs Centralized GPU)

Figure 19 directly visualizes the percentage reduction achieved.

- With 8 clients, the federated approach achieved a **17% time reduction**

Funded by the European Union
This project has received funding from the European Union's HE research and innovation programme under grant agreement No 101093003

- With 16 clients, the federated approach achieved a **38% time reduction**

The KPI target of 10-15% is clearly met and exceeded when using GPU-accelerated edge clients.

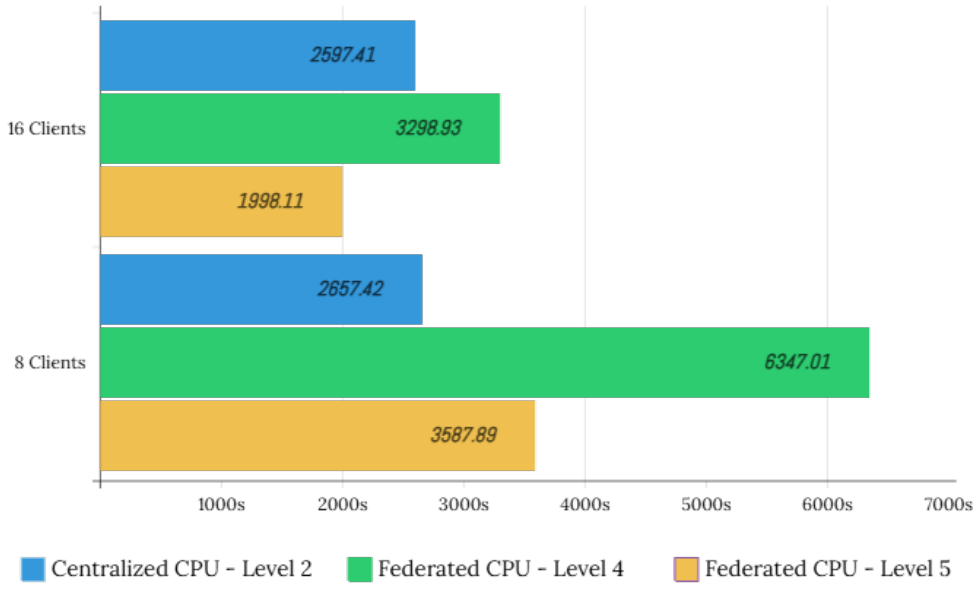


Figure 20. Total Update Time Comparison - CPU Only Environment

Figure 20, on the other hand, shows the comparison in the **CPU-Only** environment. It compares the centralized baseline (blue bars, originating from 8 and 16 clients) against the Level 4 federated approach (green bars, Jetson CPU clients) and the Level 5 federated approach (yellow bars, Raspberry Pi clients). Here, the picture is more complex: the Level 4 federated approach is significantly slower than the centralized baseline, while the Level 5 approach shows competitive times, especially with 16 clients.

Funded by the European Union
This project has received funding from the European Union's HE research and innovation programme under grant agreement No 101093003

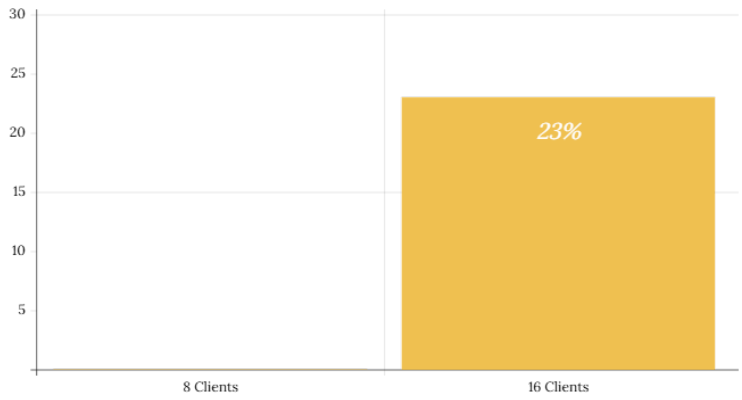


Figure 21. KPI Validation: Percentage Time Reduction/Increase (Federated CPU vs Centralized CPU)

Figure 21 isolates the percentage change for the CPU scenarios.

- The Level 4 federated approach (Jetson CPU clients) was significantly **slower** (negative percentage reduction, not shown explicitly but implied by Figure 20).
- The Level 5 federated approach (Raspberry Pi clients):
  - was **slower** than centralized with 8 clients (negative percentage reduction).
  - was **23% faster** than centralized with 16 clients (calculated from Figure 20

The KPI is met only in the specific configuration of Level 5 with 16 low-power clients. This highlights the strong dependency on hardware and potentially architectural advantages of Level 5.

The experimental results validate the effectiveness of the Federated Computation approach within the TEMA context, demonstrating the achievement of the key objectives set forth. **Temporal efficiency**, measured as the total effective time for model update, showed **significant improvements** compared to the traditional centralized approach in the most performant configurations. The objective of **KPI OA4.1** (10-15% time reduction) was **met and largely exceeded** in the **GPU-accelerated** scenarios using the Cloud-Edge architecture (Level 4), with **time reductions up to 38%** with 16 clients. Furthermore, the KPI was also **met** in the **CPU All-In-Edge (Level 5) configuration with 16 low-power clients** (Raspberry Pi), highlighting the effectiveness of the federated paradigm even under resource constraints at sufficient scale (+23% time reduction).

It is interesting to note the apparent anomaly observed in the CPU-only scenarios: the Level 4 federated system with Jetson Orin clients (using only CPU) performed slower than not only the centralized baseline but also the Level 5 system using less powerful Raspberry Pi clients. This highlights that the Jetson Orin CPU performance, optimized to work in synergy with its GPU, **did not translate into an advantage** for this specific CPU-only federated training task. Factors such as **software optimization for CPU training on**

Funded by the European Union
This project has received funding from the European Union's HE research and innovation programme under grant agreement No 101093003

JetPack 6, **potential internal resource contention within the Jetson SoC**, or a **suboptimal interaction between the Jetson CPU architecture and the specific workload** might have limited its performance in this context, making it less efficient than the Raspberry Pi 5's CPU *for this particular task*.

Overall, these results confirm that, with an adequate hardware and architectural configuration particularly leveraging GPU acceleration on edge clients or optimizing the architecture for fully decentralized scenarios (Level 5) federated computation can significantly accelerate model update cycles.

Funded by the European Union
This project has received funding from the European Union's HE research and innovation programme under grant agreement No 101093003

4. Context Information Management

Software architectures have been already proposed to address the needs identified by climate change, such as floods, wildfires, tornadoes, tsunamis, and earthquakes. Not all of these solutions involve cutting-edge technologies, such as distributed systems, AI, or the most recent computing continuum. The main goal of these architectures is to improve NDM by accelerating extreme data analytics algorithms, emergency phenomenon modeling, evolution predictions, simulations, and interactive visualization. They aim to increase trustworthiness, accuracy, and responsiveness. One weakness of such architectures is the lack of a Context Information Management (CIM) standard. CIM defines methods, structures, and protocols for managing and exchanging contextual information between computer systems to ensure interoperability and consistency between different applications and platforms through a unified API.

In this regard, this chapter discusses the research progress made under T3.4 regarding the NDM architecture. TEMA aims to develop a CIM based on a Next Generation Service Interface with Linked Data (NGSI-LD) using the Context Broker Orion-LD. The proposed framework uses a distributed computing model to process and analyze large amounts of disaster-related data in real time, providing first responders and policymakers with decision support. Unlike conventional centralized disaster management systems, the proposed approach distributes computational workloads across edge and cloud infrastructures, ensuring lower latency and improved system reliability [27]. The proposed solution embraces technological innovation, cross-sector collaboration, economic sustainability, and support for end users.

This research correlates with KPI OA4.2, which is related to the capacity to "reduce data migration for processing at the federated edge". Specifically, the target value is reducing the data migration by 10% over SoA. To achieve this KPI, a business mission (BM) was defined as a disaster use case. Experiments were carried out to demonstrate the quality of data migration in cloud and edge scenarios when the Context Broker acts as the CIM. Various networks (e.g., Ethernet, cellular data, and satellite) were adopted to demonstrate the quality of the architecture in stable and unstable internet scenarios.

The rest of the Chapter discusses the state of the art, the architecture and the role of CIM in disaster scenarios. In addition, the Chapter discusses the methodology to achieve the KPI OA4.2 and reports the results. In conclusion, the KPI is achieved and verified.

Funded by the European Union
This project has received funding from the European Union's HE research and innovation programme under grant agreement No 101093003

4.1. State of the Art

The increasing frequency and severity of natural disasters has accelerated the adoption of data-centric approaches to improve emergency response and disaster management. In this context, cloud services are becoming standard and are often supported by the IoT. However, a lack of solutions is evident at the edge of the network, where initial computation could reduce latency and overcome network issues.

Cloud-based platforms support the processing of disaster-related data, particularly for flood monitoring [28]. However, emergency scenarios require tools that handle the volume, variety, velocity, and other complexities of Big Data [29, 30]. These include tools for misinformation filtering, real-time processing, and data validation [31]. Delays in these areas can severely impact response efforts [28].

Smart urban infrastructures integrated with the IoT present new opportunities for disaster resilience. Research shows that interconnected infrastructures enhance preparedness and recovery capabilities in smart cities [32, 33, 34]. For example, the Disaster Resilient Smart Cities (DRSCs) concept is based on IoT and Big Data Analytics (BDA) to create adaptive environments [35]. In addition, environmental sensors play a vital role by providing real-time data on air quality, soil, and toxic emissions [34, 36]. These sensors improve early warning systems, while social network and crowd-sourced geospatial data offer timely situational information [35, 37]. Natural Language Processing (NLP) and deep learning models such as DistilBERT help classify this unstructured content [37]. Weather systems based on IoT further contribute to the prediction of meteorological risks [36]. Integrating IoT and BDA strengthens decision-making, forecasting, and resource use in urban disaster scenarios [32, 33]. However, challenges such as interoperability, scalability, and the complexity of smart urban systems remain [38, 39].

BDA applies to all disaster phases, from mitigation to recovery [40]. Integrating data from smartphones, infrastructure, and social network into unified analytics platforms enhances crisis response [40, 41]. Addressing data governance, privacy, and processing efficiency is essential for realizing the potential of smart city technologies [35].

In this regard, edge computing and UAVs are promising tools for improving disaster response. UAV-assisted networks that use edge nodes offer better energy use and coverage [41, 42]. DRL algorithms, including Deep Deterministic Policy Gradient (DDPG) and Proximal Policy Optimization (PPO), support efficient resource management in UAV systems [41]. However, real-time processing and maintaining network resilience in extreme conditions remain unresolved issues.

Funded by the European Union
This project has received funding from the European Union's HE research and innovation programme under grant agreement No 101093003

4.2. Beyond the State of the Art

The key component of the architecture described in Section 3 is the message broker. It sorts events and notifications between components, including information about the context. In this regard, the investigation of the ETSI CIM standard [43], which seeks to establish a model and a standardized API for handling context information originating from diverse sources, including IoT devices, legacy systems, and databases. It introduces an information model structured around entities, attributes, and relationships, extending the property graph paradigm to digitally represent real or virtual objects as digital twins.

CIM is grounded in design principles that promote architectural flexibility and interoperability, enabling the seamless integration of various systems and domains via a unified API. The standard defines a RESTful NGSI-LD API, where "NGSI-LD" combines NGSI with LD, highlighting its alignment with semantic web standards. This API allows context providers and consumers to create, update, query, and subscribe to context data in real time. Depending on how the core component, the Context Broker (CB), which implements the standards APIs, is deployed and interconnected, several architectural models are possible: centralized, distributed, or federated.

Funded by the European Union
This project has received funding from the European Union's HE research and innovation programme under grant agreement No 101093003

4.2.1. Information model

The information model is structured according to a three-tier hierarchical framework as reported in Figure 22, designed to ensure both flexibility and consistency.

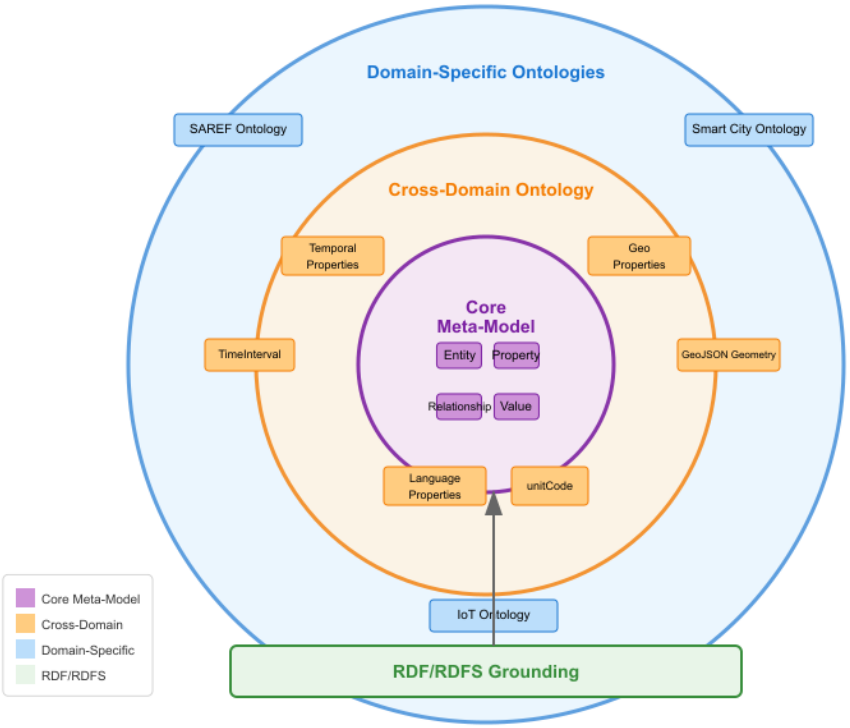


Figure 22. Information Model Structure

At the foundational tier are the Core Meta-Model classes, which align with the formal property graph model. This layer establishes the essential data structures that every implementation is required to support Entity, Property, Relationship, and Value. The second tier comprises the Cross-Domain Ontology, that is a collection of generics, cross-sector classes intended to prevent conflicting or redundant definitions across various domain-specific ontologies. This ontology encompasses universal concepts such as geospatial, temporal, and linguistic attributes, thereby providing a shared foundation for interoperability. The third tier facilitates the development of domain-specific ontologies, such as the SAREF Ontology for smart home applications. These ontologies can be mapped to the NGSI-LD information model to leverage the capabilities offered by the API.

The Core Meta-Model defines four interconnected fundamental concepts:

Entity It represents any object in the real or conceptual world that can be uniquely identified.

Funded by the European Union
This project has received funding from the European Union's HE research and innovation programme under grant agreement No 101093003

Properties They describe the characteristics or attributes of entities.

Relationships They define the connections between different entities.

Values They represent the actual values associated with properties or relationships.

This structure is firmly anchored in Semantic Web standards, particularly RDF and RDFS. Each NGSI-LD entity is a subclass of rdfs:Resource, as are its properties and relationships. Values may be either RDF literals or complex node objects, enabling the representation of sophisticated data structures.

The Cross-Domain Ontology introduces specialised concepts essential for the management of contextual information. For example:

- The Geo Properties manage geospatial information using standards such as GeoJSON, supporting points, lines, and polygons.
- The Temporal Properties capture the temporal evolution of other properties, which is vital for applications requiring time-series analysis.
- The Language Properties facilitate multilingual management of textual content.
- The unitCode property standardizes units of measurement, among others.

Based on this foundation, domain-specific ontologies can be developed to address a wide range of use cases, such as smart cities, agriculture, climate change, industry, and more.

4.2.2. Restfull NGSI-LD APIs

The NGSI-LD API is structured into four principal components, each with distinct responsibilities. The Core API is mandatory for all implementations and manages fundamental operations on entities and attributes. It includes functionality for provisioning (creation, updating, deletion), consumption (querying, retrieval), and subscription to contextual information.

The Temporal API is optional and supports the temporal evolution of entities, enabling operations such as inserting temporal instances, partially updating temporal attributes, and querying temporal changes.

The Registry API provides operations for Context Source Registration and the management of Context Source Registrations (CSR).

The JSON-LD Context API, also optional, offers functionality for storing, managing, and serving JSON-LD contexts, which are crucial for semantic interoperability. Table 6 shows the main core endpoints of the APIs:

Funded by the European Union
This project has received funding from the European Union's HE research and innovation programme under grant agreement No 101093003

Resource	Method	Purpose
`/ngsi-ld/v1/entities`	GET, POST	Search and create entities
`/ngsi-ld/v1/entities/{id}`	GET, PATCH, DELETE	Reading, partial update and deletion
`/ngsi-ld/v1/temporal/entities`	POST	Query of historical context
`/ngsi-ld/v1/subscriptions`	GET, POST	Management of subscriptions to context events
`/ngsi-ld/v1/registration`	POST	Registration of dynamic contexts/context providers
`/ngsi-ld/v1/operations`	POST	RPC or complex batch operations

Table 6. The main core endpoint of the NGSI-LD APIs.

4.2.3. Adopting Context Management in the TEMA Tech Solution

In the TEMA technical solution, Orion-LD as the Context Broker, together with MinIO as Object Storage, were employed to integrate the various components and data sources to be processed.

4.2.3.1. Context Broker

Orion-LD [44] is part of the FIWARE ecosystem. It is a context data management component designed to oversee the full lifecycle of context information. Orion-LD serves as a Context Broker and a CEF building block for context data management, supporting both the NGSI-LD and NGSI-v2 APIs.

It implements the NGSI-LD standard (an extension of JSON-LD for context management systems), maintains compatibility with NGSI-v2, manages context data as linked data using JSON, and supports operations such as updates, queries, registrations, and subscriptions.

Orion-LD adheres to the ETSI specification for the NGSI-LD API, with near-complete compliance with version 1.6.1 of the specification.

4.2.3.2. Object Storage

MinIO [45] is an open-source, distributed object storage server designed to efficiently manage large volumes of unstructured data, including images, videos, documents, logs, and backups. Its cloud-native architecture enables horizontal scalability in a seamless manner, ensuring high availability and high data throughput. With built-in support for erasure coding, MinIO provides hardware fault tolerance and data recovery capabilities.

MinIO allows nodes to be distributed across multiple machines, data centres, or cloud providers, with automatic clustering to balance load and repair corrupted data. It is fully

Funded by the European Union
This project has received funding from the European Union's HE research and innovation programme under grant agreement No 101093003

compatible with the Amazon S3 API, meaning any application developed for S3 can run on MinIO without modification, benefiting from advanced features such as multipart uploads, versioning, and lifecycle policiesmaking migration to and from AWS straightforward.

MinIO integrates naturally into DevOps environments: it offers official Docker images for containerised deployments, a dedicated Kubernetes operator, and a Terraform provider for infrastructure-as-code management. For monitoring, it relies on Prometheus and provides Grafana-ready dashboards, delivering an enterprise-grade solution from the outset. On the security front, MinIO offers multi-layered encryption: data can be encrypted either server-side or client-side, with all communications protected via TLS. Integration with external key management systems ensures compliance with stringent security standards. Access control is based on granular IAM policies, with support for LDAP, Active Directory, OpenID Connect, RBAC, and audit logging to track all operations.

The MinIO Console, an intuitive web interface, enables the management of buckets, policies, users, metrics, and data lifecycle without requiring command-line interactionmaking it accessible to both developers and system administrators.

From an application perspective, MinIO is particularly well-suited for building data lakes for analytics and machine learning projects, thanks to its support for formats such as Apache Parquet and integration with frameworks like Spark, Hadoop, and Kafka.

4.2.3.3. Orion and Minlo as integration middleware

As illustrated in the architecture of the TEMA solution in Figure 2, numerous heterogeneous information sources and components contribute to the processing and chaining of data to generate value-added knowledge. Whenever a source produces data or a component generates output, these can serve as inputs for various processing pipelines leading to visualization.
In this context, the publisher-subscriber mechanism of Orion-LD proved to be highly effective. Each time a component produces output, it uploads the data to MinIO and writes the associated metadata into the relevant entities on Orion-LD. This enables all consumers subscribed to that entity to be notified via push mode, retrieve the data, and commence their own processing.
Figure 23 illustrates the interaction diagram of a producer-consumer pair using this mechanism. Table 7 describes the interactions among the components.

Funded by the European Union
This project has received funding from the European Union's HE research and innovation programme under grant agreement No 101093003

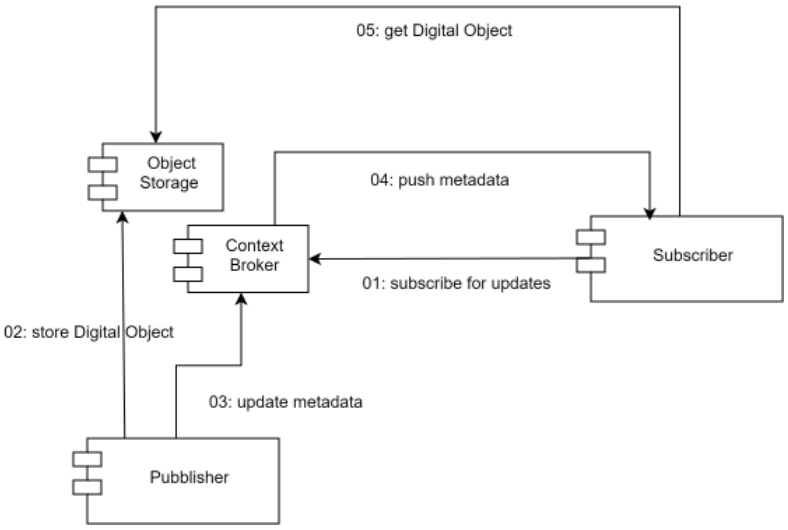


Figure 23. Interaction among components

Interaction	Description
01	The subscriber component subscribes to updates for a specific Context Broker (CB) entity containing the data and metadata of interest.
02	The publisher uploads a Digital Object (e.g., image, video, text, structured data, CSV, etc.) to the Object Storage (MinIO).
03	The publisher updates the metadata of the Digital Object within the Context Broker.
04	The Context Broker notifies the subscriber by sending the updated data to the endpoint specified at the time of subscription.
05	Upon receiving the notification along with the metadata, the subscriber can begin processing after retrieving the data from the Object Storage.

Table 7. Description of the interactions among the components.

4.3. Achieving the KPI OA4.2

T3.4 is related to more KPIs. In this Chapter, the research supports the achievement of KPI OA4.2, which is related to reducing data migration for processing at the federated edge. Specifically, the target value is reducing the data migration by 10% over SoA. This KPI is important because fast data migration is critical. The growing demand for fast, reliable responses has fueled the adoption of cloud-edge distributed architectures, which aim to reduce latency and minimize traffic to centralized data centers. In scenarios where field devices continuously generate large volumes of raw data, minimizing data migration latency and optimizing network usage provides a strategic advantage in terms of both performance and bandwidth efficiency. In this regard, the research conducted under T3.4 is contextualized within disaster scenarios that are later defined. Therefore, experiments were carried out over different network links, such as Ethernet, cellular data, and satellite.

4.3.1. Disaster Scenario

The fire disaster scenario is the reference operational scenario used to validate KPI OA4.2. It focuses on the NDM domain and, in particular, on detecting and responding to forest and urban fires. This scenario was chosen because it provides a realistic, complex, and

Funded by the European Union
This project has received funding from the European Union's HE research and innovation programme under grant agreement No 101093003

dynamic environment in which to compare and contrast different computational and analytical approaches. The goal is to evaluate the performance of the TEMA framework under distributed processing conditions. Consistent with the BM defined in Chapter 3, it represents its experimental evolution, oriented towards the technical and scientific validation of Federated Computation and Federated Analytics capabilities. It does not reproduce a real emergency, but constitutes a controlled experimental environment, in which the typical operational conditions of a disaster event are simulated using representative datasets and a distributed edge-to-cloud infrastructure.

4.3.2. Experimental Setup

A comparative test environment was set up to evaluate the KPI with a benchmark. The latter is defines as a cloud-legacy architecture and the KPI scenario is defined as a cloud-edge architecture. Figure 24 shows the former. This traditional model is based on centralized processing in the cloud. In this architecture, the base station plays a passive role, collecting data from field devices and sending it to the cloud for processing. The cloud handles the entire data management pipeline, from storage to processing. This approach is simple but can lead to bottlenecks when managing large volumes of data or dealing with connectivity issues, which directly impact latency and overall system efficiency. The Cloud-Legacy Architecture is organized into three main levels. The device layer includes the field devices that collect raw data. The edge layer acts as a simple bridge that forwards the data to the cloud without processing it. The cloud layer stores the data and processes the services centrally. The operational flow of the Cloud-Legacy Architecture is shown in Table 8.

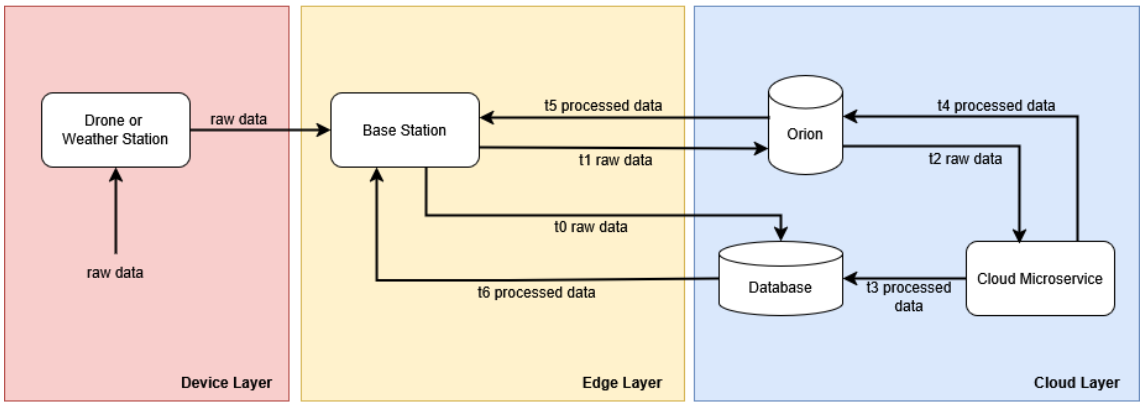


Figure 24. Cloud-Legacy Architecture

Funded by the European Union
This project has received funding from the European Union's HE research and innovation programme under grant agreement No 101093003

Time	Description
T_0	The raw data is received by the base station and stored in the database on the cloud
T_1	The metadata is notified to the context broker
T_2	The microservice in the cloud receives the raw data
T_3	The microservice processes the data
T_4	The processed data is stored in the cloud database
T_5	The processed metadata is notified to the context broker
T_6	The base station receives the processed data

Table 8. Operational flow of the Cloud-Legacy architecture

On the other hand, the cloud-edge architecture is a distributed approach in which the computational load is shared between the edge and the cloud. This reduces the amount of data transmitted upstream. Figure 25 shows the flowchart. This architecture adopts a distributed model in which data processing occurs closer to the source (e.g., at the edge of the network) to reduce latency and improve system efficiency. In this model, the base station plays an active role. After receiving data from field devices, the base station processes the data locally before sending it to the cloud. This approach reduces traffic to the cloud, latency, and optimizes network usage, especially in scenarios with large data volumes or where a quick response is needed. The cloud-edge architecture is organized into three main levels with distributed processing between the edge and the cloud. The device layer includes field-distributed devices that collect raw data, which must be processed to extract information. The edge layer is a local base station that receives and processes data through a microservice. The cloud layer stores data and interacts with services through a context broker, which manages and exposes information. The operational flow of the cloud-edge architecture is shown in Table 9.

Funded by the European Union
This project has received funding from the European Union's HE research and innovation programme under grant agreement No 101093003

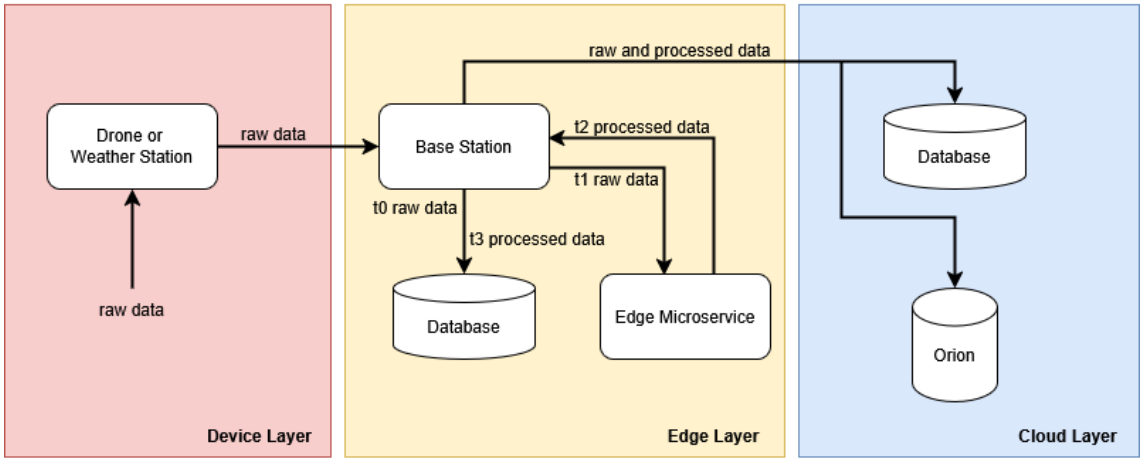


Figure 25. Cloud-Edge Architecture

Time	Description
To	The raw data is received by the base station and stored in the local database
T1	The microservice receives the raw data
T2	The microservice processes the data
T3	The processed data is stored in the local database
T*	Asynchronously, the cloud receives the data, stores it, and notifies the context broker

Table 9. Operational flow of the Cloud-Edge architecture

The experiments were conducted in an area managed by the Civil Protection Department of the Municipality of Montiferru in Sardinia, Italy. The experiments simulated real-world operational conditions typical of environmental emergency scenarios. The primary goal was to measure the impact of the two architectures on end-to-end latency considering all key stages: data transmission, processing, and result delivery. The results of this analysis provide an empirical foundation for selecting the most suitable architecture for data-intensive and mission-critical applications. The study compares the performance of cloud-legacy and cloud-edge architectures in an image processing workflow for wildfire scene segmentation, focusing particularly on end-to-end latency and intermediate process phase duration.

The input consists of images depicting wildfire scenarios of various sizes, ranging from 0.3 MB to 10 MB. Both raw and segmented images were stored in MinIO. The Orion-LD Context Broker managed and distributed image-related metadata between components and microservices. Depending on the tested architecture, the segmentation microservice

Funded by the European Union
This project has received funding from the European Union's HE research and innovation programme under grant agreement No 101093003

was executed either locally or in the cloud. The output was the segmented image, which was available through Orion-LD and MinIO. To simulate a range of network scenarios, each architecture was validated across three connection types:

- Wired fiber-optic connection
- 4G mobile network (Iliad operator)
- Satellite network (Starlink)

This approach enabled a comprehensive evaluation of system performance under real-world connectivity constraints and varying network conditions.

The experiments were repeated ten times for each size and network type. Specifically, tests were conducted under stable and controlled conditions for wired connections. All microservices ran on machines with identical hardware resources. The following metrics were considered:

- Data transmission times
- Processing times
- Total end-to-end latency

Particular attention was given to latency and key intermediate metrics (e.g., upload, transfer, and segmentation). The timing results are detailed as follows:

Upload time The time required to upload the file to Minio.

Send time The time needed to send image metadata via Orion-LD.

Segmentation time The duration for partitioning images through the fire segmentation microservice.

Segmented upload time The time to upload the segmented image to Minio.

Segmented send time The time for transmitting metadata of the segmented image via Orion.

Total time The cumulative time from the initial file upload to the final upload of the segmented image, including all processing and transfer stages.

The experimental setup supporting KPI OA4.2 validation is available as open-source software. The design, implementation, and benchmarking scripts are documented in [113].

Funded by the European Union
This project has received funding from the European Union's HE research and innovation programme under grant agreement No 101093003

4.3.3. Experimental Results

The experimental results highlight the comparative performance of Cloud-Edge and Cloud-Legacy architectures in terms of latency and processing time. Using image files of various sizes (0.3 MB, 1 MB, 5 MB, and 10 MB), the analysis covers the average transmission time, processing duration, and total end-to-end performance of each architecture.

The main differences between the two paradigms lie in resource management and workload distribution. Cloud-Edge integrates edge and cloud resources to optimize latency and data processing near the source. In contrast, Cloud-Legacy relies solely on centralized cloud infrastructure. Cloud-Edge demonstrates superior performance across all workflow phases, reducing latency and accelerating every stage compared to Cloud-Legacy.

The results were compared using visual representations that summarize the statistical distribution of processing times observed in different configurations to assess performance. The aim was to highlight trends in latency and processing time variability as a function of data size and workflow stage. To this end, a chart for every combination of architecture (Cloud-Edge or Cloud-Legacy) and network type (fiber optic, 4G, or satellite) have been generated.
The cloud-legacy benchmark is defined in Figure 26, where results obtained using the wired network are reported. The analysis focuses on processing times measured across different workflow phases and how these times evolve with respect to file size and operational conditions enabled by a dedicated wired connection. Corresponding graphs are provided for each time interval examined, including upload, metadata transmission, segmentation, segmented image upload, result metadata transmission, and end-to-end latency. These visualizations allow us to evaluate the average trends and variability of the measured results. They help highlight the specific performance characteristics and potential limitations of the Cloud-Legacy paradigm in an environment with stable, high-bandwidth connectivity.

Funded by the European Union
This project has received funding from the European Union's HE research and innovation programme under grant agreement No 101093003

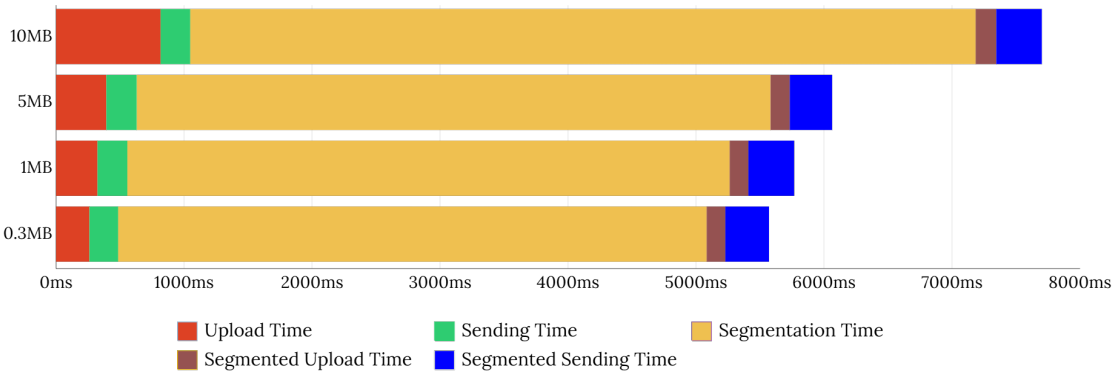


Figure 26. Overall time comparison of data migration in the Cloud-Legacy architecture with wired network

The results for the Cloud-Legacy architecture operating over the Starlink satellite are presented in Figure 27. The analysis focuses on processing and transmission times at different stages of the workflow. Specifically, it examines how performance evolves with increasing file sizes and the unique challenges posed by satellite connections. Please note that not all connections in the system rely on Starlink. Specifically, data transmission from the base station to the central database and the Context Broker occurs via the Starlink satellite network. Meanwhile, communications between cloud services are handled via high-performance wired connections. Dedicated graphs offer detailed views of each time interval, including upload and metadata transmission; segmentation; segmented image upload; result metadata transmission; and end-to-end latency. These visualizations facilitate assessing average trends and variability in the measured results. They illustrate how the system behaves under high-latency and variable bandwidth conditions, which are common in non-terrestrial networks.

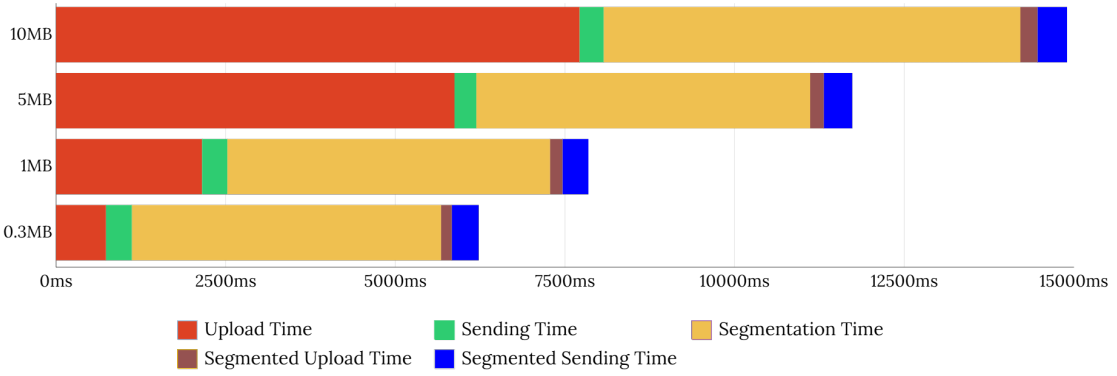


Figure 27. Overall time comparison of data migration in the Cloud-Legacy architecture with starlink network

The results concerning the Cloud-Legacy architecture using a 4G mobile network (Iliad

Funded by the European Union
This project has received funding from the European Union's HE research and innovation programme under grant agreement No 101093003

operator) are shown in Figure 28. The analysis examines processing and transmission times at different stages of the workflow. It explores how performance changes with varying file sizes, considering the characteristics and constraints of a mobile broadband connection. It is important to note that not all of the system's connections rely on the 4G network. Specifically, data transmission from the base station to the central database and Context Broker occurs over a mobile network. Meanwhile, communications between internal cloud services are handled through high-performance wired connections. Dedicated graphs are presented for each analyzed time interval: upload; metadata transmission; segmentation; segmented image upload; result metadata transmission; and end-to-end latency. These visualizations illustrate the central tendencies and variability of the results, enabling identification of system behaviors specific to mobile network environments.

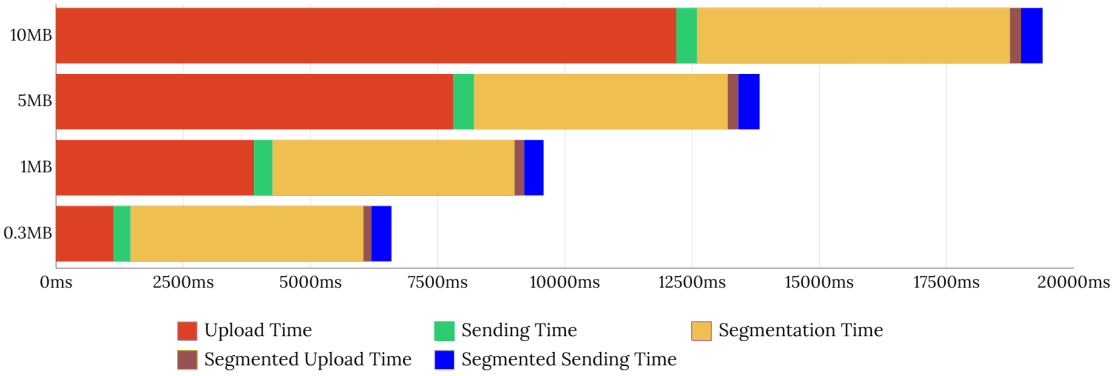


Figure 28. Overall time comparison of data migration in the Cloud-Legacy architecture with a 4G network

After the results of the cloud-legacy architecture are described, the results obtained for the cloud-edge architecture are reported. Particular attention is paid to the times measured during the different workflow phases and how these times evolve with respect to file size and network type. For each analyzed time interval (e.g., upload, metadata transmission, segmentation, uploading the segmented image, transmitting the result metadata, and end-to-end latency), the corresponding graph is provided in Figure 29. These graphs allow us to evaluate the average trend and variability of the results obtained under different operating conditions.

Funded by the European Union
This project has received funding from the European Union's HE research and innovation programme under grant agreement No 101093003

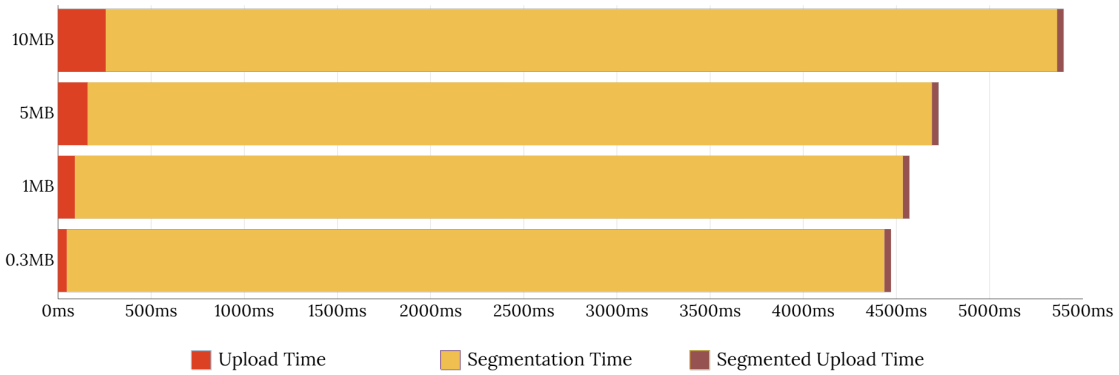


Figure 29. Overall time comparison of data migration in the Cloud-Edge architecture

Additionally, the effectiveness of the cloud-edge architecture compared to the cloud-legacy architecture have been evaluated by calculating the percentage reduction in end-to-end latency for each file size tested over various network scenarios. This analysis provides a clear picture of how distributed processing offers substantial advantages in high-performance and limited connectivity environments. The percentage of latency reduction is computed using the following equation:

$$\text{Latency Reduction (\%)} = \frac{\text{Latency}_{\text{Cloud-Legacy}} - \text{Latency}_{\text{Cloud-Edge}}}{\text{Latency}_{\text{Cloud-Legacy}}} \times 100 \quad (1)$$

The comparative results for each network environment are shown across the full range of file sizes. This illustrates the conditions under which the edge paradigm provides the greatest performance advantage. Figure 30 illustrates the reduction in end-to-end latency achieved by shifting from a cloud-legacy wired network to a cloud-edge network with Starlink and 4G capabilities across all file sizes. The analysis shows that distributed processing is beneficial, even in a low-latency, high-bandwidth network environment. In conclusion, the target value of the KPI OA4.2 to reduce the data migration by 10% over SoA is achieved.

Funded by the European Union
This project has received funding from the European Union's HE research and innovation programme under grant agreement No 101093003

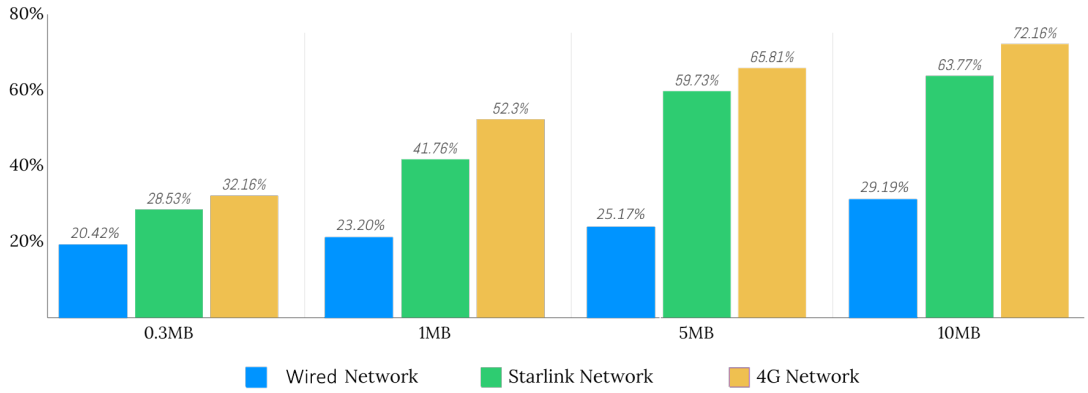


Figure 30. Percentage reduction in end-to-end latency achieved by the Cloud-Edge architecture compared to Cloud-Legacy across different network scenarios and file sizes

Funded by the European Union
This project has received funding from the European Union's HE research and innovation programme under grant agreement No 101093003

5. Computation Offloading

Edge-cloud infrastructures are dynamic, heterogeneous systems. Within an organization, several nodes with different computational capacities are typically deployed and used for tasks that may become computationally intensive over time. At the edge of the network, hardware requirements may be limited to low-energy, low-cost, and low-computational-capacity devices. While this is beneficial for distributing computing across low-power, low-cost devices in line with the United Nations' 2030 Agenda for Sustainable Development, sometimes the computing resources are insufficient for executing tasks under the constraints imposed by the Service Level Agreement (SLA). For example, processing AI models may be impractical for edge nodes without GPUs, or for executing computational tasks in autonomous unmanned vehicles. The disparity in computing resources can be even greater if the presence of multiple organizations organized is considered into a federation. In all these cases, computational offloading may be necessary.

Computation offloading is defined as the process of transferring computationally intensive tasks from resource-constrained devices to those with greater computational capacity. The goal is to enhance battery life and computing efficiency while minimizing execution time and costs. Therefore, tasks are migrated to other nodes. These nodes are typically remote cloud nodes with powerful computational capacity. However, the disaster scenarios in which the TEMA partners conducted their research were very limited. Cloud nodes may not always be available due to network disconnection caused by a disaster. Therefore, although computational tasks always start executing from edge nodes, they may migrate to other edge or cloud nodes within or outside the same organization.

In this regard, this chapter discusses the research progress made under T3.4 regarding computational offloading. TEMA aims to develop a fully autonomous system that monitors and analyzes computational resources to make optimal offloading decisions. The utilization of these resources (i.e., CPU, memory, disk I/O, and network) is forecast in one or more steps in the future and then studied for anomalies. This may generate an alarm event indicating that the target task on the target machine needs to be offloaded. A scheduler component may elect the target machine to which the task will be migrated. The offloading process is then complete.

The rest of the chapter discusses the state of the art in computational offloading, computational resource forecasting, anomaly detection, and scheduling. It also proposes a reference architecture for fully autonomous computational offloading based on AI algorithms and describes how T3.4 surpasses the current state of the art in these areas.

Funded by the European Union
This project has received funding from the European Union's HE research and innovation programme under grant agreement No 101093003

5.1. State of the Art

Computational offloading is a well-known problem [46]. It has been studied in computer science since 2000. When correlated with edge-cloud computing or the computing continuum [47], the trend has steadily grown since 2018 with a peak on 2024, as shown in Figure 31. The need to make research on computational offloading is even more relevant if the countries where research is conducted most is taken into account. According to Figure 32, Europe plays a minor role, with only Greece and Finland ranking among the top 15 countries in the world.

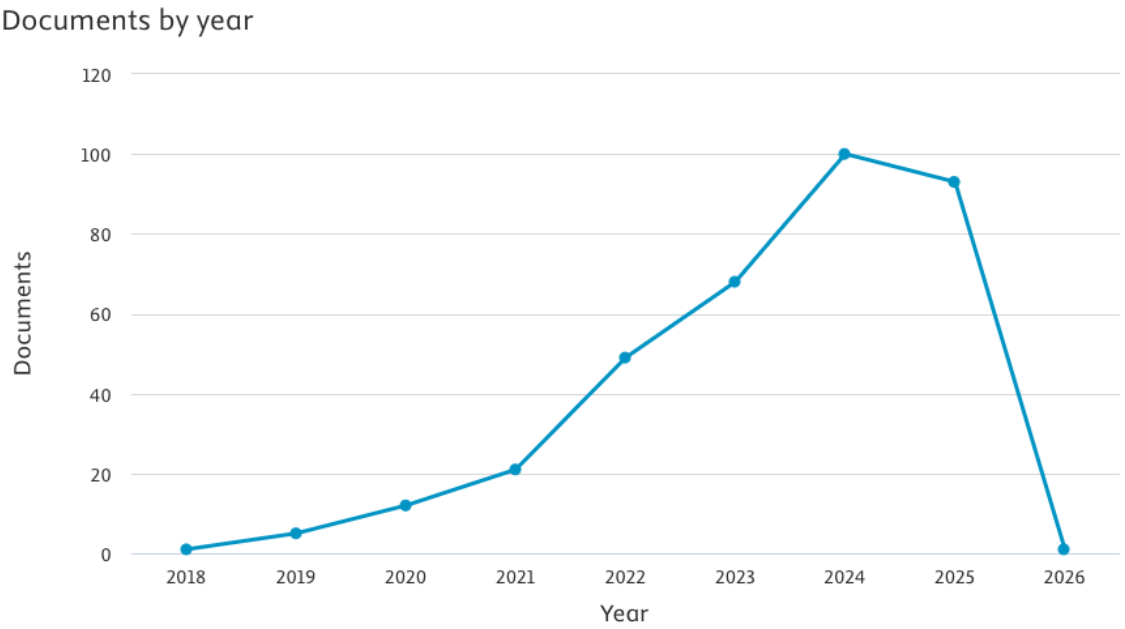


Figure 31. Documents by year according to Scopus search within article title, abstract and keywords with search documents: ("computational offloading" OR "task offloading") AND ("compute continuum" OR "cloud-edge").

Funded by the European Union
This project has received funding from the European Union's HE research and innovation programme under grant agreement No 101093003

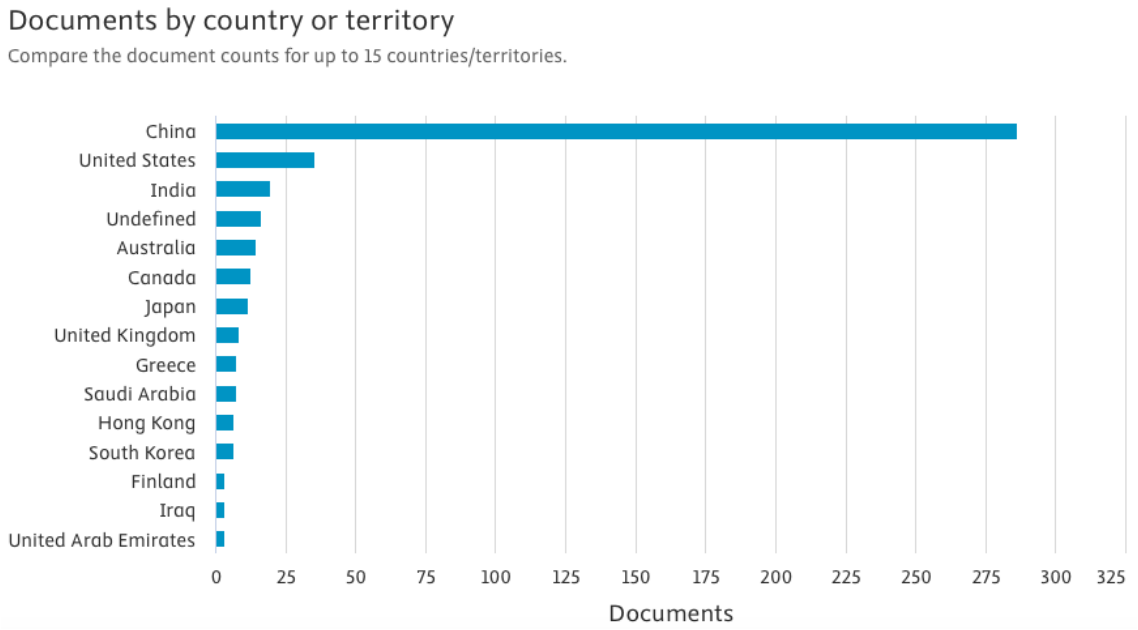


Figure 32. Documents by country or territory according to Scopus search within article title, abstract and keywords with search documents: ("computational offloading" OR "task offloading") AND ("compute continuum" OR "cloud-edge").

Computation offloading is often studied in terms of energy consumption, with the goal of maximizing battery life or reducing energy usage. Xu et al. discussed in [48] how offloading mobile applications poses a significant challenge to optimizing execution time and energy consumption for mobile devices. The authors proposed Computation Offloading Method (COM), a method for IoT-enabled cloud-edge computing. They employed the Non-Dominated Sorting Genetic Algorithm III to address the multi-objective optimization problem of task offloading in cloud-edge computing. Anajemba et al. discussed in [49] proposed a cooperative offloading technique based on the Lagrangian Suboptimal Convergent Computation Offloading Algorithm (LSCCOA) for multi-access edge computing in a distributed IoT network. The results minimized the weighted sum of transmit power consumption.

Another important aspect of computation offloading is communication capability. Kai et al. in [50] investigated a collaborative computation offloading and communication and computation resource allocation scheme to develop a collaborative computing framework. They formulated a sum latency minimization problem for all mobile devices, considering the offloading strategy, computation resources, delivery rates, and power allocation. This is a non-convex problem that is difficult to solve. To solve this optimization problem, the authors used the classic successive convex approximation (SCA) approach to transform the non-convex optimization problem into a convex one.

Computation offloading has been also studied in terms of security [51]. Blockchain and AI

Funded by the European Union
This project has received funding from the European Union's HE research and innovation programme under grant agreement No 101093003

have been proposed to secure the computation offloading in a cloud-edge environment in [52]. Zhang et al. proposed a blockchain-empowered federated deep actor-critic-based task offloading algorithm to address the secure and low-latency computation offloading problem. The coupling between the long-term security constraint and short-term queuing delay optimization was decoupled by using Lyapunov optimization.

To our knowledge, no scientific studies address computational offloading as a fully autonomous framework that predicts computing resources, searches for anomalies in those resources, and schedules tasks accordingly.

5.1.1. Computational Resource Forecasting

Resource management plays a fundamental role in optimizing performance and energy efficiency in both large-scale and micro data centers [53, 54]. Within this context, CPU utilization prediction has emerged as a key enabler for proactive workload management and dynamic resource provisioning. A survey of the recent literature reveals that most approaches rely on traditional statistical or recurrent models, particularly Autoregressive Integrated Moving Average (ARIMA) and Long-Short Term Memory (LSTM) [55, 56], while Transformer-based architectures remain comparatively underexplored.

Early works such as Duggan et al. used in [57] a Recurrent Neural Network (RNN) trained via Backpropagation-Through-Time on PlanetLab data, demonstrating effective one- and multi-step CPU forecasting. Janardhanan et al. compared in [58] ARIMA and LSTM models using Google Cluster traces, showing that LSTM significantly outperforms ARIMA for long-term predictions despite limited reproducibility details. Mason et al. introduced in [59] a bio-inspired optimization strategy using Particle Swarm Optimization, Covariance Matrix Adaptation Evolution Strategy, and Differential Evolution to fine-tune RNNs on PlanetLab traces, improving prediction accuracy for VM workloads.

More recent studies have explored hybrid and continuum-oriented approaches. Bauer et al. proposed in [60] the *UtilML*, a LSTM-based model enhanced with LeakyReLU and batch normalization, validated on the Alibaba Cloud GPU trace using RMSE, MAPE, and sMAPE. Daraghmeh et al. proposed in [61] a multilevel learning pipeline combining Isolation Forest anomaly detection, k-means clustering, and ensemble regression, achieving high R^2, MAE, and MSE performance via Gradient Boosting. Wang et al. proposed in [62] an AIOps-inspired framework integrating statistical, machine learning, and deep learning methods (e.g., SARIMAX, XGBoost, LSTM) for CPU forecasting after noise removal with a Butterworth filter, showing XGBoost as the top performer. Roque et al. applied in [63] Support Vector Regression optimized via Particle Swarm Optimization for one-step-ahead CPU predictions in NFV contexts. Wang et al. introduced in [64] ExtremoNet, which integrates Isolation Forest with multivariate regression for predicting extreme CPU loads on Alibaba Cloud container traces. Carnevale et al. proposed in [65] a federated learning framework using bidirectional LSTM mod-

Funded by the European Union
This project has received funding from the European Union's HE research and innovation programme under grant agreement No 101093003

els distributed across edge nodes and aggregated via Federated Averaging, preserving privacy while maintaining high predictive accuracy across MAE, RMSE, and R^2 metrics.

Overall, the literature demonstrates that LSTM-based models remain the dominant baseline for CPU utilization forecasting, typically with small input windows (e.g., two past samples predicting one step ahead) and short-term prediction horizons (5 to 30 minutes). Among the reviewed studies, only UtilML [60] explicitly addresses the computing continuum, though its GPU-based dataset diverges from typical edge architectures. To fill this gap, our work proposes a continuum-oriented approach: a model trained in two phases, first on the Microsoft Azure Trace dataset (VM-based) and then fine-tuned on the Alibaba Trace dataset (container-based), to bridge virtualization layers and enable generalizable, cross-platform CPU workload forecasting.

5.1.2. Anomaly Detection

Anomaly detection evolved through several methodological eras. It began with statistical roots, where Pearsons work on Gaussian standardization and principal components provided the first quantitative tools for identifying outliers [66, 67]. With the advent of computation, distance-based and clustering methods such as k-Nearest-Neighbours and k-means operationalized anomaly detection by isolating points distant from dense clusters [68, 69].

Probabilistic approaches followed. Hidden Markov Models modeled regime shifts in sequential data [70], while Bayesian networks enabled reasoning under uncertainty with incomplete labels [71]. Next, tree-based models, such as Isolation Forest, improved scalability, offering near-linear anomaly scoring for large telemetry datasets [72].
The deep learning revolution re-framed the task. Recurrent networks forecast temporal trends and detect deviations via residuals [73]. Autoencoders identify abnormal samples through high reconstruction error [74], later enhanced by MemAE [75], DAGMM [76], Donut [77], USAD [78], and Bi-LSTM hybrids for real-time monitoring [79].

Adversarial models (e.g., GANomaly, TadGAN) further expanded expressiveness through encoderdecoder games that amplify the distinction between normal and abnormal patterns [80, 81]. Representation learning methods such as Deep SVDD, OmniAnomaly, Graph Convolutional Networks, and contrastive frameworks like TS2Vec generalised anomaly detection to complex, multivariate, and relational data [82, 83, 84].

Finally, attention-based architectures (e.g.,Informer, Anomaly Transformer) unified forecasting and reconstruction using efficient sparse attention and association-discrepancy regularisation [85, 86].

Emerging research explores yet richer paradigms. Diffusion-based generators denoise

Funded by the European Union
This project has received funding from the European Union's HE research and innovation programme under grant agreement No 101093003

time-series to provide calibrated likelihoods [87], reinforcement-learning agents adaptively maximize long-term detection reward [88], and meta-learning frameworks automate algorithm selection and hyperparameter tuning across diverse datasets [89].

5.1.3. Scheduling

With the advent and widespread adoption of cloud technology, scheduling VMs and containers has become fundamental to deploying a cloud ecosystem and guaranteeing acceptable service quality. In their work [90], Beloglazov and Buyya aimed to optimize energy consumption in cloud data centers through the dynamic consolidation of VMs. Their approach addresses the issue of inefficient resource utilization, wherein physical hosts often operate at low utilization levels while consuming a significant amount of power.

The increased usage of cloud services has led to the adoption of large-scale cluster management, which optimizes the potential of horizontal scalability one of the fundamental properties of cloud architecture. Initially, Google's management was fragmented, which caused issues with long-running, latency-sensitive production services that were often deployed separately from non-production services. This resulted in resource inefficiencies. The Borg system, defined in [91], was created to solve this problem. Borg provided a unified cluster management system that could manage and colocate both types of workloads on the same shared physical infrastructure. However, the Borg design had a significant bottleneck: its scheduler was monolithic, meaning a single process was responsible for managing the entire infrastructure, resulting in poor scalability and low flexibility. Omega [92] introduced an important architectural change, introducing a decoupled architecture based on shared state and optimistic concurrency control (OCC). Separating state management from scheduling logic allowed Omega to achieve significantly higher scheduling throughput and greater architectural flexibility.

A recent study by El Horry and Zbakh [93] provides a comprehensive overview of the microservice scheduling landscape. This topic has become a critical area of research due to the widespread adoption of container-based architectures. The study offers a clear taxonomy, categorizing existing microservice scheduling techniques into four main families: mathematical modeling, heuristic techniques, meta-heuristic techniques, and machine learning techniques. Furthermore, the survey summarizes the key performance metrics used to evaluate modern schedulers, including latency, computation and communication costs, and throughput. The survey also contextualizes the scheduling process within the typical cloud hierarchy, in which containers are deployed onto VMs or directly into physical machines.

Funded by the European Union
This project has received funding from the European Union's HE research and innovation programme under grant agreement No 101093003

5.2. Beyond the State of the Art

The rapid proliferation of cloud and edge infrastructures has brought the computing continuum paradigm to the forefront of distributed systems research. The continuum enables latency-sensitive applications, real-time analytics, and pervasive AI by seamlessly integrating heterogeneous and dynamically shared resources, as discussed in [94]. However, the diversity of computational resources and the uncertainty of workloads across this distributed landscape pose substantial challenges for efficient resource management [95, 96].

In this regard, the research conducted under T3.4 focuses on workload prediction and anomaly detection for computation offloading. Although ML has greatly advanced workload prediction for virtualized cloud infrastructures, the rise of containerization introduces new complexities. Containers are more elastic and transient than traditional VMs, so they have highly volatile workloads that require more precise predictive mechanisms. Therefore, it is a critical step toward unified resource optimization to bridge forecasting models trained on VM workloads with models suitable for containerized environments. Traditional statistical and recurrent approaches often fail to generalize across heterogeneous environments because they struggle to capture short-term fluctuations and long-range temporal dependencies. Accurate CPU utilization forecasting is essential for anticipating demand, optimizing scheduling, and sustaining performance and energy efficiency at scale [62].

Moreover, the growing complexity and scale of modern cyber-physical systems and cloud-native infrastructures have made real-time monitoring essential. Data centers, edge computing environments, and IoT networks continuously emit high-frequency, high-dimensional telemetry streams encompassing CPU load, memory usage, disk I/O, and network throughput. Rapidly identifying anomalous behavior within these streams is vital to averting service degradation, preserving operational resilience, and sustaining user confidence. Failures in such interconnected environments can swiftly cascade, making proactive anomaly detection an operational necessity rather than a luxury.

5.2.1. Framework

The framework shown in Figure 33 is defined under the T3.4. Although its structure is simple, its components are particularly advanced.

Funded by the European Union
This project has received funding from the European Union's HE research and innovation programme under grant agreement No 101093003

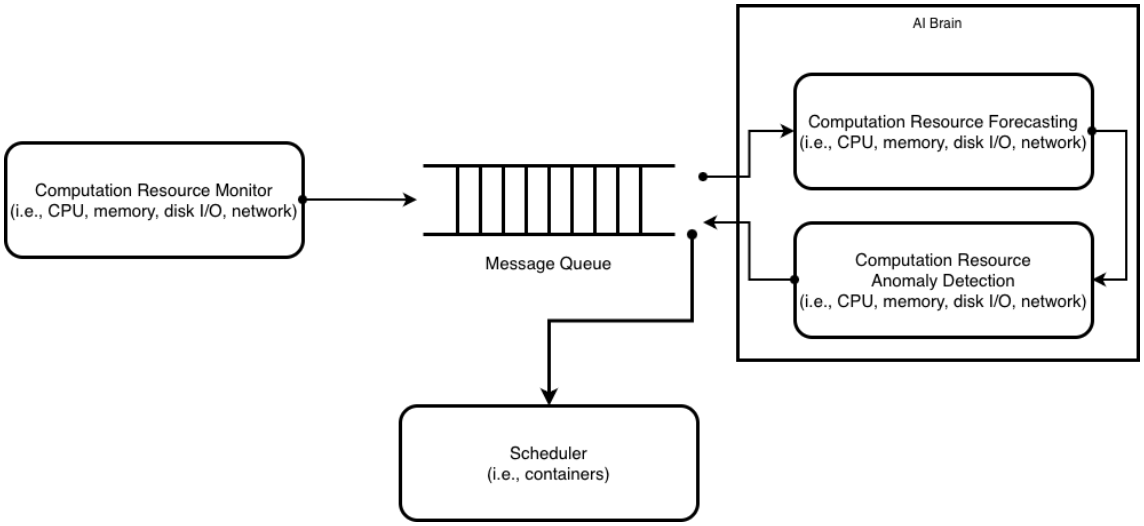


Figure 33. The offloading framework includes four main components: a computation resource monitor, a computation resource forecasting model, a computation resource anomaly detection model and a scheduler. The framework operates on operational data, such as CPU, memory, disk I/O and network, to migrate containers.

The framework operates using operational data such as CPU and memory utilization, disk I/O operations, and network bandwidth. These computational resources are collected by monitoring VMs and containers from any organization on the TEMA platform. The data is stored centrally in a node of the TEMA federation and accessed proactively. The Monitor service may be a Prometheus installation, which is a system and service monitoring system. Prometheus collects metrics from configured targets at given intervals, evaluates rule expressions, displays the results, and triggers alerts when specified conditions are met. For a complete understanding of the TEMA platform, please refer to Chapter 3.

The message queue receives data from each component and sorts it to the correct destination. It typically adopts a publish-subscribe paradigm based on topics. Some components subscribe to one or more topics to listen for data, while others publish data over one or more topics. In the TEMA platform, the message queue is replaced by the Orion Context Broker, which sorts data for the TEMA components. Therefore, data is proactively sent to the message queue and monitored by a primary component using AI models. The AI brain includes a small workflow of two microservices. First, data of computational resources are received by computational resource forecasting, and then they are analyzed by computational resource anomaly detection. The former is an AI model based on time-series analysis. A preliminary study was conducted in [65] under T3.4. The methodology was based on federated learning, and the results were promising when training with the Microsoft Azure Trace dataset based on VMs. Unfortunately, training an accurate model required significant time, even with the help of a powerful GPU. Therefore, a different approach was studied, which is discussed in the next sec-

Funded by the European Union
This project has received funding from the European Union's HE research and innovation programme under grant agreement No 101093003

tions. The result of computational resource forecasting is predicting the computational resources one or more steps ahead. This is then the input for computational resource anomaly detection, which is an AI model based on time-series analysis. It uses the same methodology as the computational resource forecasting component, but with a different model class. An interval of predicted computational resources may be classified as anomalous, resulting in an alarm being generated and sent back to the message queue.

Finally, the alarm event is sent to the scheduler. Considering the target container, the scheduler decides which target nodes to migrate it to. The scheduler is a component that was first investigated under T3.4 but has not yet been designed or implemented.

5.2.2. Computational Resource Forecasting

Recent research [97] has explored Transformer-based architectures, originally developed for natural language processing, as powerful tools for time-series forecasting. Their self-attention mechanism enables modeling of complex temporal dependencies without the sequential bottlenecks of recurrent networks [98]. Despite their success in other domains, applying temporal Transformers to workload forecasting across virtualization layers remains understudied. In the following, Transformers has been used to resolve the problem of CPU utilization forecasting. The problem definition setup a window of samples in the past, called lookback, to predict one, two or three steps in the future. Each sample has an interval of 5 minutes. Therefore, predicting one step in the future means looking 5 minutes ahead, predicting two steps in the future means looking 10 minutes ahead and predicting three steps in the future means looking 15 minutes ahead. The choice of predicting a few step in the future is two-fold. On the one hand, this is explained in [64]. Indeed, this avoids accumulation of errors in the long-term forecast if it is present. Our choice minimizes errors and guarantees the reliability of the forecasted results. On the other hand, considering a $t = 300s$, it is enough time to be informed about the future behavior of the infrastructure resource, plan and act the migration from the source to a target destination [99]. The choice is therefore confirmed by the literature.

To the best of our knowledge, the proposed framework is the first to leverage transfer learning across virtualization layers, from VMs to containers. It adopts large-scale datasets such as Microsoft Azure[2] [114] and Alibaba Cloud[3] [115]. While existing models [60, 61] are trained on isolated or homogeneous datasets, the proposed two-phase training pipeline enables cross-domain adaptation, improving generalization across heterogeneous infrastructures within the computing continuum. This constitutes a significant step toward unifying predictive resource management between cloud and edge

[2] https://github.com/Azure/AzurePublicDataset
[3] https://github.com/alibaba/clusterdata

Funded by the European Union
This project has received funding from the European Union's HE research and innovation programme under grant agreement No 101093003

environments. The pipeline is shown in Figure 34. This constitutes a significant step toward unifying predictive resource management between cloud and edge environments.

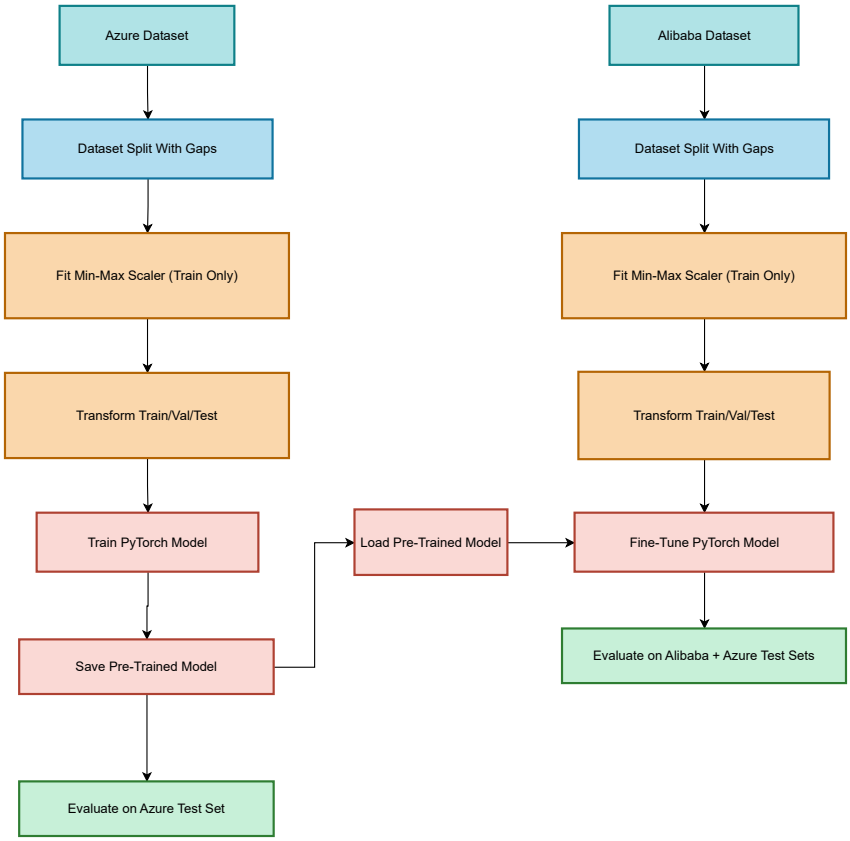


Figure 34. Pipeline for transfer learning between datasets. The model is first pretrained on the Microsoft Azure dataset (left branch), including splitting, scaling, and training. The resulting pretrained model is then fine-tuned on the Alibaba dataset (right branch) using its own splits and scaling, followed by evaluation on both Alibaba and Microsoft Azure testsets.

The experiments were executed on a dedicated VM provisioned with 8 GB of RAM and powered by an AMD Ryzen 9 7950X processor with 16 cores, a high-performance CPU designed for parallel workloads. For accelerated training and inference, the system was equipped with an NVIDIA RTX 4090 Super GPU, featuring 24 GB of dedicated VRAM, which provided sufficient memory to accommodate large Transformer models and multi-step forecasting experiments without requiring aggressive batch size reductions or gradient checkpointing trade-offs. The software stack was centered on Python 3 as the main programming language, with the PyTorch deep learning framework serving as the backbone for model implementation. PyTorch v2.6.0 was chosen for its efficient memory management, flexibility for custom architecture design, and strong integration with

Funded by the European Union
This project has received funding from the European Union's HE research and innovation programme under grant agreement No 101093003

the CUDA framework, which enabled direct exploitation of the GPUs parallel processing capabilities. Other modules used are Numpy v2.3.1 for the array management, Pandas v2.3.2 for the `.csv` files processing, Matplotlib v3.10.0 for data visualization and Scikit-learn v1.7.2 and Scipy v1.16.1 for metrics calculations, such as R^2, MSE, MAE, RMSE and confidence intervals in inference time calculations.

The quality of the methodology is expressed in terms of:

- $\boldsymbol{R^2}$: it measures the proportion of the variance in the dependent variable that is predictable from the independent variables, providing an indication of the model's accuracy and is defined as:

$$\boldsymbol{R^2} = 1 - \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{n}(y_i - \bar{y})^2} \tag{2}$$

  where y_i represents the real value, $\bar{y}$ represents the average of the real values, and $\hat{y}$ the predicted value. The calculated value $\boldsymbol{R^2}$ must lie between 0 and 1: a value close to 1 indicates a higher predictive capacity. A negative value $\boldsymbol{R^2}$ implies that the model is not usable.
- **Mean Squared Error** (MSE): measures the average of squared errors between predicted values and actual values. It emphasizes larger errors more than MAE and is commonly used for regression tasks:

$$\text{MSE} = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2 \tag{3}$$

- **Root Mean Squared Error** (RMSE): measure the standard deviation between predicted values and actual values. It is useful for understanding the absolute error when the errors are squared to prevent positive and negative values from canceling each other out.

$$\text{RMSE} = \sqrt{\text{MSE}} \tag{4}$$

- **Mean Absolute Error** (MAE): measures the average of the absolute errors between predicted values and actual values. It is useful for understanding the accuracy of a model's predictions:

$$\text{MAE} = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i| \tag{5}$$

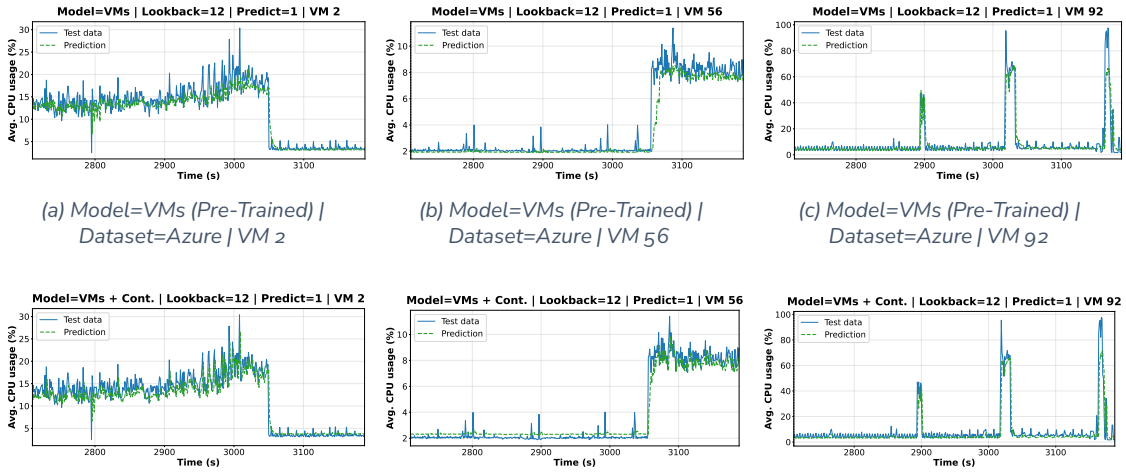
- **Average inference time**: measures the average time in seconds for making an inference.
- **Model size**: measures the size of the model in MB.

Unlike previous architectures, which are either large-scale and data center-oriented [65] or rely on complex ensemble strategies [61], the proposed Temporal Transformer achieves

Funded by the European Union
This project has received funding from the European Union's HE research and innovation programme under grant agreement No 101093003

a compact model size of only 3.2 MB. It maintains real-time inference with average latencies below 0.04 s on container traces. This lightweight design allows for deployment on resource-constrained devices while retaining accuracy comparable to or higher than that of deeper recurrent networks.

Experiments were setup with 6, 12 and 24 steps as lookback window size. The obtained results underline specific common trends across the different experimental scenarios as follows: i) Azure, ii) Alibaba and iii) Azure after Transfer Learning. Experimental evaluation shows that the proposed approach achieves higher predictive accuracy ($R^2 > 0.85$, MAE ≈ 0.036) compared to traditional recurrent and ensemble-based baselines [57, 59, 62]. When it comes to qualitative analysis, referring to the Azure dataset, Figure 35 clearly shows how the predicted curve closely follows the testing values, often matching the shape of the actual timeseries. This qualitative evidence is consistent with the quantitative metrics previously discussed, which confirm that the transfer learning process improved the ability of the model to capture relevant patterns.



(a) Model=VMs (Pre-Trained) | Dataset=Azure | VM 2

(b) Model=VMs (Pre-Trained) | Dataset=Azure | VM 56

(c) Model=VMs (Pre-Trained) | Dataset=Azure | VM 92

(d) Model=VMs+Cont. (Fine-Tuned) | Dataset=Azure | VM 2

(e) Model=VMs+Cont. (Fine-Tuned) | Dataset=Azure | VM 56

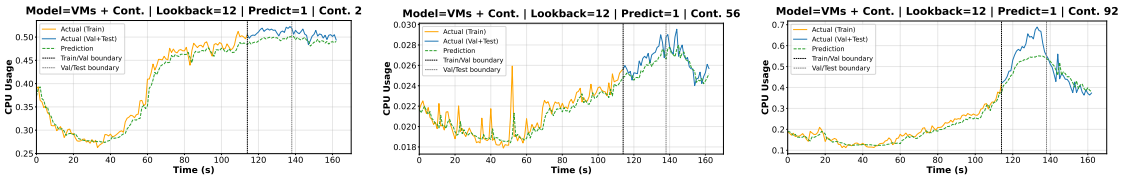
(f) Model=VMs+Cont. (Fine-Tuned) | Dataset=Azure | VM 92

Figure 35. Comparison of prediction results on the Microsoft Azure Trace dataset. Subfigures 35a-35b-35c show model trained only on the Azure dataset, while Subfigures 35d-35e-35f show the corresponding model fine-tuned with transfer learning.

Regarding the predictions on the Alibaba dataset as shown in Figure 36, the test curves exhibit a good alignment with the actual values, showing only minimal deviations. This result highlights the ability of the Temporal Transformer, once fine-tuned through transfer learning, to generalize effectively across different environments. The predicted series not only follows the overall trend of the ground truth, but also captures local fluctuations and short-term variations, which are typically harder to reproduce. This behavior further

Funded by the European Union
This project has received funding from the European Union's HE research and innovation programme under grant agreement No 101093003

confirms the robustness of the approach and supports the conclusions drawn from the quantitative metrics.



(a) Model=VMs+Cont. (Fine-Tuned) | Dataset=Alibaba | Cont. 2

(b) Model=VMs+Cont. (Fine-Tuned) | Dataset=Alibaba | Cont. 56

(c) Model=VMs+Cont. (Fine-Tuned) | Dataset=Alibaba | Cont. 92

Figure 36. Comparison of prediction results on the Alibaba dataset on Training-Validation-Test data. Subfigures 36a-36b-36c show the corresponding model trained with transfer learning.

Beyond quantitative improvements, qualitative analyses confirm that the model captures both steady-state and burst-phase CPU behaviors with high fidelity, demonstrating its capability to replicate complex utilization patterns that are typical of real-world cloud workloads. Using multi-head attention enables the model to learn correlations across temporal steps and multiple VMs or containers simultaneously. This allows for fine-grained awareness of inter-instance dependencies. This contrasts with classical single-stream models, which treat each instance independently. Multi-head attention opens new perspectives for multi-tenant resource forecasting in federated or decentralized environments.

The repository containing the implementation is available on GitHub [116], while the adaptation to a Function-as-a-Service (FaaS) environment is provided in a separate repository [117].

5.2.3. Anomaly Detection

Although anomaly detection has been extensively studied, many existing methods perform poorly in large-scale production deployments. Classical statistical approaches, such as control charts and threshold-based techniques, are easy to implement but depend on rigid assumptions about data distribution and stationarity. This leads to fragility under dynamic workloads. Modern ML models, such as autoregressive methods and RNNs, improve upon these approaches by learning temporal dependencies. However, they are limited to short-term dynamics due to their constrained memory, reducing their ability to model long-range or multimodal temporal correlations. The proposed solution uses a bidirectional LSTM (Bi-LSTM) with autoencoder as the basis for the anomaly detection framework. This architecture is excellent at modeling temporal dependencies in telemetry signals because it processes sequences in both directions, forward and backward. This enables the network to capture long-range correlations and bidirectional contextual cues that are often missed by standard LSTMs or Gated Recurrent Units (GRUs). The autoencoder component is introduced to learn a compact latent representation of normal system behavior by minimizing the reconstruction error between

Funded by the European Union
This project has received funding from the European Union's HE research and innovation programme under grant agreement No 101093003

the input and its reconstruction. This mechanism allows the model to detect subtle deviations in CPU utilization patterns without requiring explicit anomaly labels, making it particularly suitable for unsupervised anomaly detection in large-scale cloud environments.

In the following, Bi-LSTM Autoencoder has been used to resolve the problem of CPU utilization anomaly detection. The problem definition setup a window of samples in the past, called lookback, to predict one, two or three steps in the future. Each sample has an interval of 5 minutes. Therefore, predicting one step in the future means looking 5 minutes ahead, predicting two steps in the future means looking 10 minutes ahead and predicting three steps in the future means looking 15 minutes ahead. The choice of predicting a few step in the future was already discussed in the previous Section and it is confirmed by the literature.

First, the Bi-LSTMn Autoencoder is trained on multiple VMs instances from the Microsoft Azure Trace[4] dataset. During this phase, the network learns a general representation of normal system dynamics by minimizing the reconstruction error between predicted and observed CPU utilization windows. Next, the model is fine-tuned using the Alibaba Cloud Trace[5] dataset to improve its adaptability and robustness across different cloud environments. Cross-domain fine-tuning enables the model to generalize across infrastructures with different orchestration policies and temporal load distributions, enhancing its ability to distinguish genuine anomalies from workload-induced variability. The training pipeline is shown in Figure 37.

[4] https://github.com/Azure/AzurePublicDataset
[5] https://github.com/alibaba/clusterdata

Funded by the European Union
This project has received funding from the European Union's HE research and innovation programme under grant agreement No 101093003

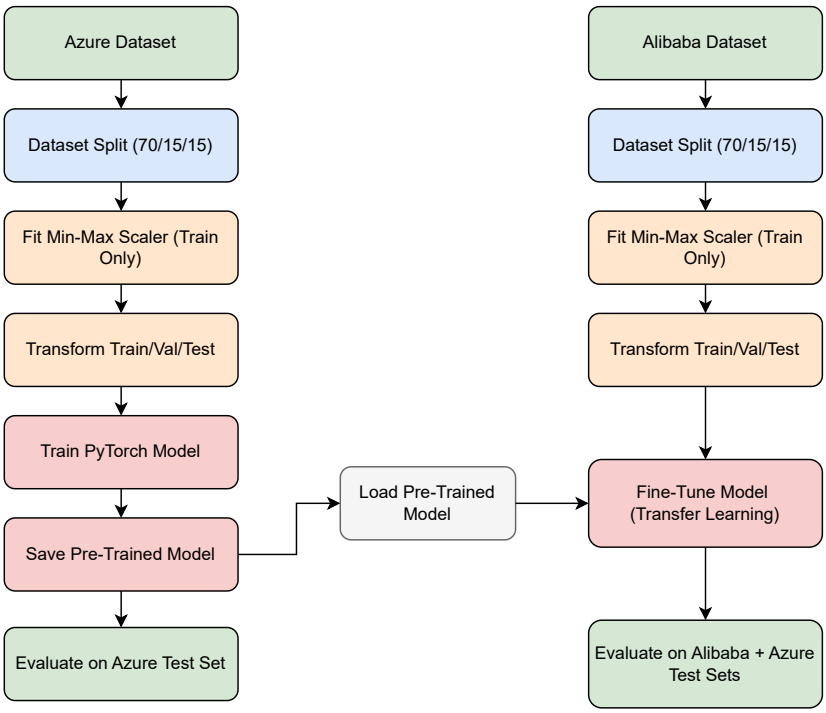


Figure 37. Pipeline for transfer learning between datasets. The model is first pretrained on the Microsoft Azure dataset (left branch), including data splitting, scaling, and training. The resulting pretrained model is then fine-tuned on the Alibaba dataset (right branch) using its own splits and scaling, followed by evaluation on both Alibaba and Microsoft Azure test sets.

Once trained, the Bi-LSTM Autoencoder can detect anomalies in CPU utilization patterns across multiple VMs or containers in parallel. The detection process uses reconstruction-based thresholds; windows with a high mean absolute error (MAE) or root mean square error (RMSE) relative to the models baseline prediction are marked as anomalous. Empirical results demonstrate that the proposed approach achieves high reconstruction quality, with MAE values below 0.04 and RMSE values around 0.06. These results confirm the models ability to accurately represent normal operational behavior.

The experimental evaluation on a dedicated workstation with an Intel Core i7-14700K processor has been carried out in which the 64 GB of DDR5 system memory and an NVIDIA RTX 5070 GPU have been setup. This configuration provided ample computational capacity for model training and inference tasks. The software stack is built on Python 3 as the primary programming language, and the PyTorch deep learning framework serves as the backbone for model implementation. PyTorch v2.6.0 was chosen for its efficient memory management, flexibility in designing custom architectures, and strong integration with the NVIDIA CUDA framework. This enabled a fully compatible workflow with the GPU's parallel processing capabilities. The other modules used

Funded by the European Union
This project has received funding from the European Union's HE research and innovation programme under grant agreement No 101093003

are NumPy v2.3.1 for array management, Pandas v2.3.2 for processing .csv files, Matplotlib v3.10.0 for data visualization, and Scikit-learn v1.7.2 and SciPy v1.16.1 for calculating metrics such MSE, MAE, RMSE, and confidence intervals in inference time calculations.

To quantitatively assess the reconstruction quality and anomaly detection performance of the autoencoder-based models, several metrics were employed.

- **Mean Squared Error** (MSE): measures the average of squared errors between predicted values and actual values. It emphasizes larger errors more than MAE and is commonly used for regression tasks:

$$\text{MSE} = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2 \tag{6}$$

- **Root Mean Squared Error** (RMSE): measure the standard deviation between predicted values and actual values. It is useful for understanding the absolute error when the errors are squared to prevent positive and negative values from canceling each other out.

$$\text{RMSE} = \sqrt{\text{MSE}} \tag{7}$$

- **Mean Reconstruction Error** (MRE): represents the average reconstruction error across all time windows, used as a baseline for anomaly thresholding

$$\text{MRE} = \frac{1}{N}\sum_{t=1}^{N} e_t, \quad e_t = |x_t - \hat{x}_t| \tag{8}$$

- **Standard Deviation of Reconstruction Error** (Std Error): quantifies the variability of reconstruction errors. A high standard deviation indicates unstable reconstruction behavior, often due to anomalies.

$$\sigma_e = \sqrt{\frac{1}{N}\sum_{t=1}^{N}(e_t - \text{MRE})^2} \tag{9}$$

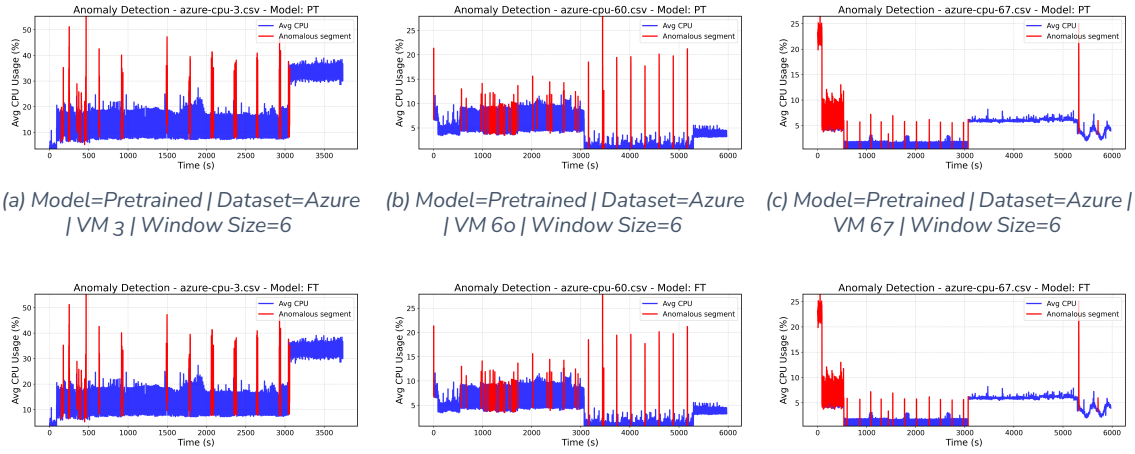
- **Reconstruction Error Ratio** (RER): estimates the ratio between the maximum and mean reconstruction errors.

$$\text{RER} = \frac{\max(e_t)}{\text{MRE}} \tag{10}$$

Experiments were set up with lookback window sizes of 6, 12, and 24 steps. Since each sample has an interval of five minutes, these lookback windows correspond to historical horizons of 30, 60, and 120 minutes, respectively. Each configuration was treated as an

Funded by the European Union
This project has received funding from the European Union's HE research and innovation programme under grant agreement No 101093003

independent experiment to assess the influence of different temporal spans on reconstruction accuracy and anomaly detection sensitivity. The reconstruction results confirm the stability and robustness of the proposed Bi-LSTM autoencoder across different temporal configurations and transfer settings. For each tested window size, the pretrained model on the Azure dataset (PT-Azure Test) achieved consistently low reconstruction errors, with an average MAE of approximately 0.013 and RMSE close to 0.038. These results indicate the network's high capability to reproduce normal temporal patterns in CPU utilization, even when the input window length varies by a factor of four. Similar results across windows suggest that the latent representation captures scale-invariant temporal dependencies and that longer lookback horizons do not significantly degrade reconstruction quality. In addition, the model is highly efficient from a deployment standpoint. With a footprint of only 566 KB, it may run on constrained edge devices, such as embedded gateways or industrial controllers. Results of the reconstruction result are shown in Figure 38.



(a) Model=Pretrained | Dataset=Azure | VM 3 | Window Size=6

(b) Model=Pretrained | Dataset=Azure | VM 60 | Window Size=6

(c) Model=Pretrained | Dataset=Azure | VM 67 | Window Size=6

(d) Model=Fine-Tuned | Dataset=Azure | VM 3 | Window Size=6

(e) Model=Fine-Tuned | Dataset=Azure | VM 60 | Window Size=6

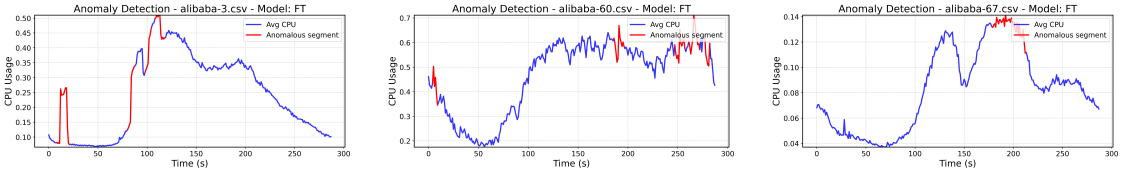
(f) Model=Fine-Tuned | Dataset=Azure | VM 67 | Window Size=6

Figure 38. Comparison of reconstruction results on the Microsoft Azure Trace dataset. Subfigures 38a38c correspond to the model trained only on the Azure dataset (Pretrained), while Subfigures 38d38f show the same VMs after Fine-Tuning with transfer learning from the Alibaba dataset.

After fine-tuning with transfer learning on the Alibaba Cloud Traces dataset (also referred to as the PT-Alibaba test), a moderate increase in reconstruction error was observed. The MAE became approximately [0.019 - 0.020] and the RMSE became approximately 0.042. This increase in error is expected due to the domain shift between the two datasets. Despite differences in workload characteristics, the model exhibits stable reconstruction behavior, with a standard deviation of reconstruction error around [0.017 - 0.018], which remains close to that of the Azure-only configuration. The increase in RER from approximately 2.6 to 3.2 reflects the higher dynamic variability of the Alibaba traces but remains within acceptable bounds for reliable anomaly scoring. Results are

Funded by the European Union
This project has received funding from the European Union's HE research and innovation programme under grant agreement No 101093003

shown in 39.



(a) Model=Fine-Tuned | Dataset=Alibaba | Cont. 3 | Window Size=6

(b) Model=Fine-Tuned | Dataset=Alibaba | Cont. 60 | Window Size=6

(c) Model=Fine-Tuned | Dataset=Alibaba | Cont. 67 | Window Size=6

Figure 39. Comparison of prediction results on the Alibaba dataset. Subfigures 39a-39b-39c show the corresponding model trained with transfer learning.

The resulting framework is ideal for Computing Continuum applications, in which consistent anomaly detection is required across cloud and edge layers. Its architecture enables the simultaneous monitoring of multiple instances, supporting distributed analytics and localized decision-making without the need for centralized supervision. The combination of bidirectional temporal modeling, cross-domain fine-tuning, robust evaluation metrics, and a lightweight design makes the proposed Bi-LSTM autoencoder a reliable, scalable, and resource-efficient solution for real-time anomaly detection in distributed computing infrastructures.

Funded by the European Union
This project has received funding from the European Union's HE research and innovation programme under grant agreement No 101093003

6. Conclusion

Deliverable D3.3 - "Report on real-time federated analytics" is the third deliverable of the WP3 of the TEMA project. This document reports the initial research results of T3.4 "Real-time federated analytics" over 24 months (M13-M36) of the project.

The main outputs of the research carried out were 10 publications in conference proceedings and the submission of other 2 scientific papers to journals and/or international conference. Additionally, 5 TEMA essential components were developed, supporting the TEMA platform functionalities described in Deliverable D2.2. These TEMA technologies will be interlinked and orchestrated through the developed TEMA Core. This document serves as a summary of the main research outputs and serves as a reference point for researchers summarizing the technical challenges of designing novel federated cloud-edge architecture. Finally, as a public deliverable, it assists in the dissemination of the projects results to the scientific community.

The techniques developed in T3.4 are used in all other tasks because they represent an essential part of the TEMA platform. Overall, TEMA technology integration will be performed in Task T6.2.

Funded by the European Union
This project has received funding from the European Union's HE research and innovation programme under grant agreement No 101093003

References

[1] Z. Zhou, X. Chen, E. Li, L. Zeng, K. Luo, and J. Zhang, "Edge intelligence: Paving the last mile of artificial intelligence with edge computing," *Proceedings of the IEEE*, vol. 107, p. 17381762, Aug. 2019.

[2] Z. Yang, W. Bao, D. Yuan, N. H. Tran, and A. Y. Zomaya, "Federated learning with nesterov accelerated gradient," *IEEE Transactions on Parallel and Distributed Systems*, vol. 33, pp. 4863–4873, 09 2022.

[3] Z. Zhou, X. Chen, E. Li, L. Zeng, K. Luo, and J. Zhang, "Edge intelligence: Paving the last mile of artificial intelligence with edge computing," *Proceedings of the IEEE*, vol. 107, pp. 1738–1762, 06 2019.

[4] S. Muksimova, S. Umirzakova, J. Baltayev, and Y. I. Cho, "Lightweight deep learning model for fire classification in tunnels," *Fire*, vol. 8, pp. 85–85, 02 2025.

[5] F. Efthymiadis, A. Karras, C. Karras, and S. Sioutas, "Advanced optimization techniques for federated learning on non-iid data," *Future Internet*, vol. 16, pp. 370–370, 10 2024.

[6] Y. Zhao, M. Li, L. Lai, N. Suda, D. Civin, and V. Chandra, "Federated learning with non-iid data," *arXiv (Cornell University)*, 01 2018.

[7] S. Chitram, S. Kumar, and S. Thenmalar, "Enhancing fire and smoke detection using deep learning techniques," pp. 7–7, 03 2024.

[8] J. Aramendia, A. Cabrera, J. Martín, J. Ángel Gumiel, and K. Basterretxea, "Neural network implementation for fire detection in critical infrastructures: A comparative analysis on embedded edge devices," *Electronics*, vol. 14, pp. 1809–1809, 04 2025.

[9] R. Peng, C. Cui, and W. Yun, "Real-time fire detection algorithm on low-power endpoint device," *Research Square (Research Square)*, 11 2024.

[10] Y. Li, "Federated learning for time series forecasting using hybrid model," 01 2019.

[11] M. Avgeris, D. Spatharakis, D. Dechouniotis, N. Kalatzis, I. Roussaki, and S. Papavassiliou, "Where there is fire there is smoke: A scalable edge computing framework for early fire detection," *Sensors*, vol. 19, pp. 639–639, 02 2019.

[12] G. Canonaco, A. Bergamasco, A. Mongelluzzo, and M. Roveri, "Adaptive federated learning in presence of concept drift," *2022 International Joint Conference on Neural Networks (IJCNN)*, pp. 1–7, 07 2021.

[13] Y. Liu, A. Huang, Y. Luo, H. Huang, Y. Liu, Y. Chen, L. Feng, T. Chen, H. Yu, and Q. Yang, "Fedvision: An online visual object detection platform powered by federated learning," vol. 34, pp. 13172–13179, Association for the Advancement of Artificial Intelligence, 04 2020.

Funded by the European Union

This project has received funding from the European Union's HE research and innovation programme under grant agreement No 101093003

[14] A. R. Khouas, M. R. Bouadjenek, H. Hacid, and S. Aryal, "Training machine learning models at the edge: A survey," *arXiv (Cornell University)*, 03 2024.

[15] T. Meuser, L. Lovén, M. Bhuyan, S. G. Patil, S. Dustdar, A. Aral, S. Bayhan, C. Becker, E. de Lara, A. Y. Ding, J. Edinger, J. Gross, N. Mohan, A. D. Pimentel, Étienne Rivière, H. Schulzrinne, P. Simoens, G. Solmaz, and M. Welzl, "Revisiting edge ai: Opportunities and challenges," *IEEE Internet Computing*, vol. 28, pp. 49–59, 07 2024.

[16] Y. Xu, D. Feng, M. Zhao, Y. Sun, and X. Xia, "Edge intelligence empowered metaverse: Architecture, technologies, and open issues," *IEEE Network*, vol. 37, pp. 92–100, 10 2023.

[17] J. Zhang and D. Tao, "Empowering things with intelligence: A survey of the progress, challenges, and opportunities in artificial intelligence of things," *IEEE Internet of Things Journal*, vol. 8, pp. 7789–7817, 11 2020.

[18] X. Wang, Z. Tang, J. Guo, T. Meng, C. Wang, T. Wang, and W. Jia, "Empowering edge intelligence: A comprehensive survey on on-device ai models," 03 2025.

[19] J. Lu, Y. Zheng, L. Guan, B. Lin, W. Shi, J. Zhang, and Y. Wu, "Fcmi-yolo: An efficient deep learning-based algorithm for real-time fire detection on edge devices," *PLOS One*, vol. 20, p. e0329555, Aug. 2025.

[20] G. Vazquez, S. Zhai, and M. Yang, "Detecting wildfire flame and smoke through edge computing using transfer learning enhanced deep learning models," 2025.

[21] M. F. S. Titu, M. A. Pavel, G. K. O. Michael, H. Babar, U. Aman, and R. Khan, "Real-time fire detection: Integrating lightweight deep learning models on drones with edge computing," *Drones*, vol. 8, p. 483, Sept. 2024.

[22] Y. J. Lee, H. G. Jung, and J. K. Suhr, "Semantic segmentation network slimming and edge deployment for real-time forest fire or flood monitoring systems using unmanned aerial vehicles," *Electronics*, vol. 12, p. 4795, Nov. 2023.

[23] S. Mahmoudi, M. Gloesener, M. Benkedadra, and J.-S. Lerat, "Edge ai system for real-time and explainable forest fire detection using compressed deep learning models," in *Proceedings of the 20th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications*, p. 847854, SCITEPRESS - Science and Technology Publications, 2025.

[24] M. Yang, S. Qian, and X. Wu, "Realtime fire and smoke detection with transfer learning based on cloudedge collaborative architecture," *IET Image Processing*, vol. 18, p. 37163728, July 2024.

[25] D. J. Beutel, T. Topal, A. Mathur, X. Qiu, T. Parcollet, Y. Gao, and A. Sani, "Flower: A friendly federated learning research framework," in *Workshop on Federated Learning for User Privacy and Data Confidentiality at ICML 2020*, 2020.

[26] M. Wang, P. Yue, L. Jiang, D. Yu, T. Tuo, and J. Li, "Fasdd: An open-access 100,000-level flame and smoke detection dataset for deep learning in fire detection," *Scientific Data*, vol. 11, no. 1, p. 178, 2024.

[27] M. Sarkar and P. K. Sahoo, "Leveraging edge computing for video data streaming in uav-based emergency response systems," *Sensors*, 2024.

[28] A. J. Rosenblum, C. M. Wend, Z. Akhtar, L. Rosman, J. D. Freeman, and D. J. Barnett, "Use of big data in disaster recovery: An integrative literature review," *Disaster Medicine and Public Health Preparedness*, vol. 17, no. 2, 2023.

[29] X. Song, H. Zhang, R. Akerkar, H. Huang, S. Guo, L. Zhong, Y. Ji, A. L. Opdahl, H. Purohit, A. Skupin, A. Pottathil, and A. Culotta, "Big data and emergency management: Concepts, methodologies, and applications," *IEEE Transactions on Big Data*, vol. 8, no. 2, p. 397 419, 2022.

[30] M. Yu, C. Yang, and Y. Li, "Big data in natural disaster management: A review," *Geosciences (Switzerland)*, vol. 8, no. 5, 2018.

[31] S. Akter and S. F. Wamba, "Big data and disaster management: a systematic review and agenda for future research," *Annals of Operations Research*, vol. 283, no. 1-2, p. 939 959, 2019.

[32] L. B. Elvas, B. M. Mataloto, A. L. Martins, and J. C. Ferreira, "Disaster management in smart cities," *Smart Cities*, 2021.

[33] S. A. Shah, D. Z. Seker, S. Hameed, and D. Draheim, "The rising role of big data analytics and iot in disaster management: Recent advances, taxonomy and prospects," *IEEE Access*, vol. 7, p. 54595 54614, 2019.

[34] F. Zeng, C. Pang, and H. Tang, "Sensors on internet of things systems for the sustainable development of smart cities: A systematic literature review," *Sensors*, vol. 24, no. 7, 2024.

[35] S. A. Shah, D. Z. Seker, M. M. Rathore, S. Hameed, S. Ben Yahia, and D. Draheim, "Towards disaster resilient smart cities: Can internet of things and big data analytics be the game changers?," *IEEE Access*, vol. 7, p. 91885 91903, 2019.

[36] R. P. Das, D. Muduli, and A. K. Luhach, "An investigation into the practical use of internet of things technology in smart city applications," in *Proceedings of the 1st International Conference on Cognitive, Green and Ubiquitous Computing (IC-CGU 2024)*, 2024.

[37] S. Kamoji, N. D. Jayaprakash, C. B. Biju, and G. A. Benoy, "Data analytics for disaster management response using sentiments," in *Proceedings of the International Conference on Sustainable Computing and Smart Systems (ICSCSS 2023)*, p. 1690 1695, 2023.

Funded by the European Union

This project has received funding from the European Union's HE research and innovation programme under grant agreement No 101093003

[38] F. Zeng, C. Pang, and H. Tang, "Sensors on the internet of things systems for urban disaster management: A systematic literature review," *Sensors*, vol. 23, no. 17, 2023.

[39] A. Samarakkody, D. Amaratunga, and R. Haigh, "Characterising smartness to make smart cities resilient," *Sustainability (Switzerland)*, vol. 14, no. 19, 2022.

[40] C. Yang, G. Su, and J. Chen, "Using big data to enhance crisis response and disaster resilience for a smart city," in *Proceedings of the 2017 IEEE 2nd International Conference on Big Data Analysis (ICBDA 2017)*, p. 504 507, 2017.

[41] X. Tang, F. Chen, F. Wang, and Z. Jia, "Disaster-resilient emergency communication with intelligent air-ground cooperation," *IEEE Internet of Things Journal*, vol. 11, no. 3, p. 5331 5346, 2024.

[42] Z. Qadir, F. Ullah, H. S. Munawar, and F. Al-Turjman, "Addressing disasters in smart cities through uavs path planning and 5g communications: A systematic review," *Computer Communications*, vol. 168, p. 114 135, 2021.

[43] "Context information management (cim) ngsi-ld api," Tech. Rep. GS CIM 009 V1.8.1, ETSI, Mar. 2024.

[44] "Orion-ld github repository." https://github.com/FIWARE/context.Orion-LD.

[45] "Minio website." https://min.io/. Accessed: 2025-07-15.

[46] S. Zhang, N. Yi, and Y. Ma, "A survey of computation offloading with task types," 2024.

[47] F. Saeik, M. Avgeris, D. Spatharakis, N. Santi, D. Dechouniotis, J. Violos, A. Leivadeas, N. Athanasopoulos, N. Mitton, and S. Papavassiliou, "Task offloading in edge and cloud computing: A survey on mathematical, artificial intelligence and control theory solutions," *Computer Networks*, vol. 195, p. 108177, 2021.

[48] X. Xu, Q. Liu, Y. Luo, K. Peng, X. Zhang, S. Meng, and L. Qi, "A computation offloading method over big data for iot-enabled cloud-edge computing," *Future Generation Computer Systems*, vol. 95, pp. 522–533, 2019.

[49] J. H. Anajemba, T. Yue, C. Iwendi, M. Alenezi, and M. Mittal, "Optimal cooperative offloading scheme for energy efficient multi-access edge computation," *IEEE Access*, vol. 8, pp. 53931–53941, 2020.

[50] C. Kai, H. Zhou, Y. Yi, and W. Huang, "Collaborative cloud-edge-end task offloading in mobile-edge computing networks with limited communication capability," *IEEE Transactions on Cognitive Communications and Networking*, vol. 7, no. 2, pp. 624–634, 2021.

[51] A. Ometov, O. L. Molua, M. Komarov, and J. Nurmi, "A survey of security in cloud, edge, and fog computing," *Sensors*, vol. 22, no. 3, 2022.

[52] S. Zhang, Z. Wang, Z. Zhou, Y. Wang, H. Zhang, G. Zhang, H. Ding, S. Mumtaz, and M. Guizani, "Blockchain and federated deep reinforcement learning based secure cloud-edge-end collaboration in power iot," *IEEE Wireless Communications*, vol. 29, no. 2, pp. 84–91, 2022.

[53] R. Estrada, I. Valeriano, and X. Aizaga, "CPU Usage Prediction Model: A Simplified VM Clustering Approach," in *Complex, Intelligent and Software Intensive Systems* (L. Barolli, ed.), vol. 176, pp. 210–221, Cham: Springer Nature Switzerland, 2023. Series Title: Lecture Notes on Data Engineering and Communications Technologies.

[54] M. Fahimullah, R. Gupta, S. Ahvar, and M. Trocan, "Explaining Predictive Scheduling in Cloud," in *Recent Challenges in Intelligent Information and Database Systems* (E. Szczerbicki, K. Wojtkiewicz, S. V. Nguyen, M. Pietranik, and M. Krótkiewicz, eds.), vol. 1716, pp. 81–91, Singapore: Springer Nature Singapore, 2022. Series Title: Communications in Computer and Information Science.

[55] K. N. Quoc, V. Tong, C. Dao, T. N. Le, and D. Tran, "Boosted regression for predicting CPU utilization in the cloud with periodicity," *The Journal of Supercomputing*, Aug. 2024.

[56] S.-c. Park and Y.-h. Kim, "An Improvement of Multi-Cluster Stability of Private Cloud Systems through LSTM-Based CPU Usage Prediction," *The Journal of Korean Institute of Communications and Information Sciences*, vol. 47, pp. 1081–1095, Aug. 2022.

[57] M. Duggan, K. Mason, J. Duggan, E. Howley, and E. Barrett, "Predicting host CPU utilization in cloud computing using recurrent neural networks," in *2017 12th International Conference for Internet Technology and Secured Transactions (ICITST)*, (Cambridge), pp. 67–72, IEEE, Dec. 2017.

[58] D. Janardhanan and E. Barrett, "CPU workload forecasting of machines in data centers using LSTM recurrent neural networks and ARIMA models," in *2017 12th International Conference for Internet Technology and Secured Transactions (ICITST)*, (Cambridge), pp. 55–60, IEEE, Dec. 2017.

[59] K. Mason, M. Duggan, E. Barrett, J. Duggan, and E. Howley, "Predicting host CPU utilization in the cloud using evolutionary neural networks," *Future Generation Computer Systems*, vol. 86, pp. 162–173, Sept. 2018.

[60] C. Bauer, N. Mehran, R. Prodan, and D. Kimovski, "Machine Learning Based Resource Utilization Prediction in the Computing Continuum," in *2023 IEEE 28th International Workshop on Computer Aided Modeling and Design of Communication Links and Networks (CAMAD)*, (Edinburgh, United Kingdom), pp. 219–224, IEEE, Nov. 2023.

Funded by the European Union

This project has received funding from the European Union's HE research and innovation programme under grant agreement No 101093003

[61] M. Daraghmeh, A. Agarwal, and Y. Jararweh, "A Multilevel Learning Model for Predicting CPU Utilization in Cloud Data Centers," in *2023 IEEE Intl Conf on Dependable, Autonomic and Secure Computing, Intl Conf on Pervasive Intelligence and Computing, Intl Conf on Cloud and Big Data Computing, Intl Conf on Cyber Science and Technology Congress (DASC/PiCom/CBDCom/CyberSciTech)*, (Abu Dhabi, United Arab Emirates), pp. 1016–1023, IEEE, Nov. 2023.

[62] Z. Wang, H. Zhao, and R. Dai, "Research on Stacked Model for Cluster CPU Utilization Prediction Based on Butterworth Filter," in *2023 4th International Conference on Big Data, Artificial Intelligence and Internet of Things Engineering (ICBAIE)*, (Hangzhou, China), pp. 191–197, IEEE, Aug. 2023.

[63] R. Roque, R. Paula, D. Bezerra, I. Rodrigues, R. Lins, M. Tenca, and G. Almeida, "A PSO-SVR Based Resource Allocation Approach for Network Function Virtualization," in *2023 IEEE Latin American Conference on Computational Intelligence (LA-CCI)*, (Recife-Pe, Brazil), pp. 1–6, IEEE, Oct. 2023.

[64] C. Wang and Z. Wang, "Isolated Forest-Based Prediction of Container Resource Load Extremes," *Applied Sciences*, vol. 14, p. 2911, Mar. 2024.

[65] L. Carnevale, D. Balouek, S. Sebbio, M. Parashar, and M. Villari, "Private distributed resource management data: Predicting cpu utilization with bi-lstm and federated learning," in *2025 IEEE 25th International Symposium on Cluster, Cloud and Internet Computing (CCGrid)*, pp. 266–275, 2025.

[66] K. Pearson, "Contributions to the mathematical theory of evolution. ii. skew variation in homogeneous material," *Philosophical Transactions of the Royal Society of London A*, vol. 186, pp. 343–414, 1895.

[67] K. Pearson, "On lines and planes of closest fit to systems of points in space," *The London, Edinburgh, and Dublin Philosophical Magazine*, vol. 2, no. 11, pp. 559–572, 1901.

[68] T. M. Cover and P. E. Hart, "Nearest neighbor pattern classification," *IEEE Transactions on Information Theory*, vol. 13, no. 1, pp. 21–27, 1967.

[69] J. B. MacQueen, "Some methods for classification and analysis of multivariate observations," in *Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability*, vol. 1, pp. 281–297, University of California Press, 1967.

[70] L. Rabiner, "A tutorial on hidden markov models and selected applications in speech recognition," *Proceedings of the IEEE*, vol. 77, no. 2, pp. 257–286, 1989.

[71] J. Pearl, *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 1988.

Funded by the European Union

This project has received funding from the European Union's HE research and innovation programme under grant agreement No 101093003

[72] F. T. Liu, K. M. Ting, and Z.-H. Zhou, "Isolation forest," in *Proceedings of the 2008 IEEE International Conference on Data Mining (ICDM)*, pp. 413–422, 2008.

[73] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Comput.*, vol. 9, p. 17351780, Nov. 1997.

[74] G. E. Hinton and R. R. Salakhutdinov, "Reducing the dimensionality of data with neural networks," *Science*, vol. 313, no. 5786, pp. 504–507, 2006.

[75] D. Gong, L. Liu, V. Le, B. Saha, M. R. Mansour, S. Venkatesh, and A. van den Hengel, "Memorizing normality to detect anomaly: Memory-augmented deep autoencoder for unsupervised anomaly detection," in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 1705–1714, October 2019.

[76] B. Zong, Q. Song, M. R. Min, W. Cheng, C. Lumezanu, D. Cho, and H. Chen, "Deep autoencoding gaussian mixture model for unsupervised anomaly detection," in *International Conference on Learning Representations*, 2018.

[77] H. Xu, Y. Chen, and et al., "Unsupervised anomaly detection via variational auto-encoder for seasonal kpis," in *SIGKDD*, 2018.

[78] J. Audibert and et al., "Usad: Unsupervised anomaly detection on multivariate time series," in *ICDM*, 2021.

[79] M. N. Amin, L. Hubner, F. S. Bayram, and A. Jesser, "Real-time anomaly detection with lstm-autoencoder network on microcontrollers for industrial applications," in *Proceedings of the 2024 8th International Conference on Graphics and Signal Processing*, ICGSP '24, (New York, NY, USA), p. 4246, Association for Computing Machinery, 2024.

[80] S. Akçay, A. Atapour-Abarghouei, and T. Breckon, "Ganomaly: Semi-supervised anomaly detection via adversarial training," *Asian Conference on Computer Vision*, 2018.

[81] A. Geiger, D. Liu, S. Alnegheimish, A. Cuesta-Infante, and K. Veeramachaneni, " TadGAN: Time Series Anomaly Detection Using Generative Adversarial Networks ," in *2020 IEEE International Conference on Big Data (Big Data)*, (Los Alamitos, CA, USA), pp. 33–43, IEEE Computer Society, Dec. 2020.

[82] J. Cao and et al., "Omnianomaly: A hybrid model," in *KDD*, 2019.

[83] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," in *International Conference on Learning Representations (ICLR)*, 2017. arXiv:1609.02907.

[84] Z. Yue, Y. Wang, J. Duan, T. Yang, C. Huang, and B. Xu, "Learning timestamp-level representations for time series with hierarchical contrastive loss," *CoRR*, vol. abs/2106.10466, 2021.

Funded by the European Union

This project has received funding from the European Union's HE research and innovation programme under grant agreement No 101093003

[85] H. Zhou, S. Zhang, J. Peng, S. Zhang, J. Li, H. Xiong, and W. Zhang, "Informer: Beyond efficient transformer for long sequence time-series forecasting," *CoRR*, vol. abs/2012.07436, 2020.

[86] J. Xu, H. Wu, J. Wang, and M. Long, "Anomaly transformer: Time series anomaly detection with association discrepancy," in *International Conference on Learning Representations (ICLR)*, 2022. arXiv:2110.02642.

[87] I. Pintilie, A. Manolache, and F. Brad, "Time series anomaly detection using diffusion-based models," 2023.

[88] T. Wu and J. Ortiz, "Rlad: Time series anomaly detection through reinforcement learning and active learning," in *Proceedings of the 7th SIGKDD Workshop on Mining and Learning from Time Series (MiLeTS 21)*, pp. 1–6, ACM, 2021.

[89] J. M. Navarro, A. Huet, and D. Rossi, "Meta-learning for fast model recommendation in unsupervised multivariate time series anomaly detection," in *Proceedings of the Second International Conference on Automated Machine Learning* (A. Faust, R. Garnett, C. White, F. Hutter, and J. R. Gardner, eds.), vol. 224 of *Proceedings of Machine Learning Research*, pp. 24/1–19, PMLR, 12–15 Nov 2023.

[90] A. Beloglazov and R. Buyya, "Optimal online deterministic algorithms and adaptive heuristics for energy and performance efficient dynamic consolidation of virtual machines in cloud data centers," *Concurrency and Computation: Practice and Experience*, vol. 24, no. 13, pp. 1397–1420, 2012.

[91] A. Verma, L. Pedrosa, M. Korupolu, D. Oppenheimer, E. Tune, and J. Wilkes, "Large-scale cluster management at google with borg," in *Proceedings of the Tenth European Conference on Computer Systems*, EuroSys '15, (New York, NY, USA), Association for Computing Machinery, 2015.

[92] M. Schwarzkopf, A. Konwinski, M. Abd-El-Malek, and J. Wilkes, "Omega: flexible, scalable schedulers for large compute clusters," in *Proceedings of the 8th ACM European Conference on Computer Systems*, EuroSys '13, (New York, NY, USA), p. 351364, Association for Computing Machinery, 2013.

[93] "Microservices scheduling algorithms: A survey study," 2023.

[94] S. Dustdar, V. C. Pujol, and P. K. Donta, "On distributed computing continuum systems," *IEEE Transactions on Knowledge and Data Engineering*, vol. 35, no. 4, pp. 4092–4105, 2022.

[95] M. Masdari and A. Khoshnevis, "A survey and classification of the workload forecasting methods in cloud computing," *Cluster Computing*, vol. 23, no. 4, pp. 2399–2424, 2020.

Funded by the European Union

This project has received funding from the European Union's HE research and innovation programme under grant agreement No 101093003

[96] J. Gao, H. Wang, and H. Shen, "Machine learning based workload prediction in cloud computing," in *2020 29th international conference on computer communications and networks (ICCCN)*, pp. 1–9, IEEE, 2020.

[97] Q. Wen, T. Zhou, C. Zhang, W. Chen, Z. Ma, J. Yan, and L. Sun, "Transformers in time series: A survey," 2023.

[98] A. Hameed, J. Violos, A. Leivadeas, N. Santi, R. Grünblatt, and N. Mitton, "Toward qos prediction based on temporal transformers for iot applications," *IEEE Transactions on Network and Service Management*, vol. 19, no. 4, pp. 4010–4027, 2022.

[99] R. Hat, "How to create and scale 6,000 virtual machines in 7 hours with red hat openshift virtualization." https://cloud.redhat.com/learning/learn:how-create-and-scale-6000-virtual-machines-7-hours-red-hat-openshift-virtualization/resource/resources:virtual-machine-migration-workflows, 2024.

Funded by the European Union

This project has received funding from the European Union's HE research and innovation programme under grant agreement No 101093003

List of Publications as Outputs of D3.3

[100] G. Di Modica, A. Galletta, L. Carnevale, A. Alkhansa, A. Costantini, D. Cesini, P. Bellavista, and M. Villari, "Orchestration of containerized applications in the cloud continuum," in *2023 IEEE International Conference on Pervasive Computing and Communications Workshops and other Affiliated Events (PerCom Workshops)*, pp. 44–49, 2023.

[101] C. Sicari, A. Catalfamo, L. Carnevale, A. Galletta, D. Balouek-Thomert, M. Parashar, and M. Villari, "Tema: Event driven serverless workflows platform for natural disaster management," in *2023 IEEE Symposium on Computers and Communications (ISCC)*, pp. 1–6, 2023.

[102] R. Marino, L. Carnevale, and M. Villari, "When robotics meets distributed learning: the federated learning robotic network framework," in *2023 IEEE Symposium on Computers and Communications (ISCC)*, pp. 1–6, 2023.

[103] L. Carnevale, A. Filograna, F. Arigliano, R. Marino, A. Ruggeri, and M. Fazio, "Supporting the natural disaster management distributing federated intelligence over the cloud-edge continuum: the tema architecture," in *Proceedings of the IEEE/ACM 10th International Conference on Big Data Computing, Applications and Technologies*, BDCAT '23, (New York, NY, USA), Association for Computing Machinery, 2024.

[104] S. Sebbio, G. Morabito, A. Catalfamo, L. Carnevale, and M. Fazio, "Federated learning on raspberry pi 4: A comprehensive power consumption analysis," in *Proceedings of the IEEE/ACM 16th International Conference on Utility and Cloud Computing*, UCC '23, (New York, NY, USA), Association for Computing Machinery, 2024.

[105] L. Carnevale, D. Balouek, S. Sebbio, M. Parashar, and M. Villari, "Private distributed resource management data: Predicting cpu utilization with bi-lstm and federated learning," in *2025 IEEE 25th International Symposium on Cluster, Cloud and Internet Computing (CCGrid)*, pp. 266–275, 2025.

[106] S. Sebbio, L. Carnevale, D. Balouek, M. Parashar, and M. Villari, "Data-driven operational artificial intelligence for computing continuum: A natural disaster management use case," in *2025 IEEE 25th International Symposium on Cluster, Cloud and Internet Computing Workshops (CCGridW)*, pp. 92–99, 2025.

[107] M. A. Gambito, L. Carnevale, M. R. Jabbarpour, B. Javadi, and M. Villari, "Hierarchical federated learning for natural disaster management," in *2024 IEEE/ACM 17th International Conference on Utility and Cloud Computing (UCC)*, pp. 282–289, 2024.

Funded by the European Union

This project has received funding from the European Union's HE research and innovation programme under grant agreement No 101093003

[108] P. DellAcqua, M. Fazio, L. Carnevale, and M. Villari, "Knowledge distillation and federated learning for data-driven monitoring of electrical vehicle li-battery," in *2024 IEEE Symposium on Computers and Communications (ISCC)*, pp. 1–5, 2024.

[109] R. Marino, L. Carnevale, M. Fazio, and M. Villari, "Make federated learning a standard in robotics by using ros2," in *Proceedings of the IEEE/ACM 10th International Conference on Big Data Computing, Applications and Technologies*, BDCAT '23, (New York, NY, USA), Association for Computing Machinery, 2024.

Funded by the European Union

This project has received funding from the European Union's HE research and innovation programme under grant agreement No 101093003

List of D3.3 Public Repositories

[110] FCRLab-Unime, "Continuum architecture." https://github.com/fcrlab-unime/continuum-architecture, 2025. Accessed: 2025-10-22.

[111] G. Jocher, A. Chaurasia, and J. Qiu, "Yolo by ultralytics." https://github.com/ultralytics/ultralytics, 2023.

[112] FCRLab-Unime, "Centralized and federated experiments." https://github.com/fcrlab-unime/CAFE, 2025. Accessed: 2025-10-22.

[113] FCRLab-Unime, "Cloud architecture gap evaluation." https://github.com/fcrlab-unime/CAGE, 2025. Accessed: 2025-10-22.

[114] Microsoft Azure, "Microsoft azure public dataset repository." https://github.com/Azure/AzurePublicDataset, 2019. Accessed: 2025-10-19.

[115] Alibaba Group, "Alibaba cluster trace program - cluster data repository." https://github.com/alibaba/clusterdata, 2018. Accessed: 2025-10-19.

[116] FCRLab-Unime, "Cpu forecasting transformer." https://github.com/fcrlab-unime/Temporal-Transformer-for-CPU-Forecasting, 2025. Accessed: 2025-10-19.

[117] FCRLab-Unime, "Cpu forecasting transformer (faas)." https://github.com/fcrlab-unime/Temporal-Transformer-for-CPU-Forecasting-FaaS, 2025. Accessed: 2025-10-19.

[118] D. Franklin, "jetson-containers." https://github.com/dusty-nv/jetson-containers, 2020–Present. Accessed: 2025-10-22.

Funded by the European Union

This project has received funding from the European Union's HE research and innovation programme under grant agreement No 101093003

Annex A: Jetson Environment Configuration

This section provides a detailed description of the software environment configured on NVIDIA Jetson AGX Orin devices, used as edge computing nodes to execute deep learning and federated learning workloads.
The setup aimed to create a reproducible, container-based infrastructure optimized for GPU acceleration and consistent across all nodes in the deployment.

Installation of NVIDIA JetPack 6

The configuration process started with the installation of **NVIDIA JetPack 6**, which provides an Ubuntu 22.04 LTS distribution optimized for the Jetson architecture. JetPack includes all the key components required for AI workloads on embedded hardware:

- NVIDIA GPU drivers optimized for inference and training workloads on Jetson.
- CUDA Toolkit for GPU-based parallel computation.
- cuDNN for optimized deep neural network operations.
- TensorRT for high-performance model optimization and inference.
- Supporting libraries such as OpenCV, GStreamer, and Vulkan for vision and multimedia applications.

The installation was performed using the **NVIDIA SDK Manager** on a Linux host, which flashes the operating system and related components onto the Jetson AGX Orin through a USB connection. In some cases, pre-configured JetPack images can be written directly to NVMe or SD storage to simplify deployment and ensure consistent system configurations across devices.

Containerized Environment for Deep Learning

To ensure software reproducibility, ease of maintenance, and isolation of dependencies, the experimental environment was implemented using **Docker containers**. This approach enables precise control of software versions, ensures repeatable results, and simplifies the management of Python and CUDA dependencies.
Although NVIDIA provides several official JetPack-compatible containers through the **NVIDIA NGC** catalog, these do not always cover all framework combinations or library versions required in complex experimental setups. Therefore, the open-source framework **`jetson-containers`** [118] was used to generate customized container images optimized for the specific JetPack 6 (L4T r36.x) installation.

Funded by the European Union
This project has received funding from the European Union's HE research and innovation programme under grant agreement No 101093003

Building Containers with `jetson-containers`

The `jetson-containers` project, maintained by NVIDIA engineer Dustin Franklin, provides a modular and automated framework for building and running AI containers on Jetson devices. It ensures full compatibility with the installed L4T release and integrates seamlessly with the NVIDIA Container Runtime, granting direct access to GPU resources, CUDA kernels, and TensorRT acceleration from within the containers.
Unlike prebuilt NGC images, `jetson-containers` builds each image locally using a flexible layered Docker architecture. This process allows fine-grained control over:

- The versions of PyTorch, TensorFlow, and other AI frameworks.
- The inclusion of auxiliary libraries such as `torchvision`, `torchaudio`, and OpenCV.
- System-level packages and dependencies (e.g., `ffmpeg`, `numpy`, `pandas`).
- Optimization flags and build arguments specific to the Orin platform (ARM64 + Ampere GPU architecture).

Each container is built using pre-defined recipes (e.g., `pytorch`, `tensorrt`, `ros`, `deepstream`), or custom definitions when specific software stacks are required. The resulting images are fully compatible with the NVIDIA runtime and support GPU-accelerated workloads directly on-device, providing near-native performance.

Deployment and Package Configuration

After building the container image, it was deployed on each Jetson AGX Orin device. The environment included all dependencies required for the deep learning and federated learning workloads, specifically:

- **Ultralytics YOLOv8** for real-time object detection and computer vision tasks.
- **Flower (FLWR)** for distributed and federated learning experiments.
- Additional scientific libraries such as `NumPy`, `Pandas`, and `OpenCV`.

Each container was executed using the NVIDIA runtime (`-runtime nvidia`) to enable direct GPU access, and with host networking mode to simplify communication between local and remote nodes in federated learning setups. This ensured uniform environments and eliminated discrepancies due to local configuration differences.

Reproducibility and Scalability

The combination of NVIDIA JetPack 6 and custom containers built with `jetson-containers` provided a robust, scalable, and reproducible software foundation for edge AI workloads. By isolating all dependencies and automating the build process, the system guarantees consistent performance and simplifies maintenance. New Jetson nodes can be integrated into the infrastructure simply by deploying the same container image, without reconfiguration or manual library setup.
This architecture enables rapid scaling of edge computing experiments, while preserving deterministic behavior and ensuring full compatibility with the Jetson Orin hardware and the underlying CUDA software stack.

Funded by the European Union
This project has received funding from the European Union's HE research and innovation programme under grant agreement No 101093003