Computer Communications 210 (2023) 356–375

Contents lists available at ScienceDirect

Computer Communications

journal homepage: www.elsevier.com/locate/comcom

Enabling federated learning of explainable AI models within beyond-5G/6G networks

José Luis Corcuera Bárcena [a], Pietro Ducange [a], Francesco Marcelloni [a], Giovanni Nardini [a,b], Alessandro Noferi [a], Alessandro Renda [a,*], Fabrizio Ruffini [a], Alessio Schiavo [a], Giovanni Stea [a], Antonio Virdis [a]

[a] Department of Information Engineering, University of Pisa, Largo Lucio Lazzarino 1, Pisa, 56122, Italy
[b] Center for Logistic Systems, University of Pisa, Via dei Pensieri 60, Livorno, 57124, Italy

ARTICLE INFO

Keywords:
Federated learning
Trustworthy AI
Explainable AI
Machine learning
B5G/6G networks
QoE forecasting

ABSTRACT

The quest for trustworthiness in Artificial Intelligence (AI) is increasingly urgent, especially in the field of next-generation wireless networks. Future Beyond 5G (B5G)/6G networks will connect a huge amount of devices and will offer innovative services empowered with AI and Machine Learning tools. Nevertheless, private user data, which are essential for training such services, are not an asset that can be unrestrictedly shared over the network, mainly because of privacy concerns. To overcome this issue, Federated Learning (FL) has recently been proposed as a paradigm to enable collaborative model training among multiple parties, without any disclosure of private raw data. However, the initiative to natively integrate FL services into mobile networks is still far from being accomplished. In this paper we propose a novel FL-as-a-Service framework that provides the B5G/6G network with flexible mechanisms to allow end users to exploit FL services, and we describe its applicability to a Quality of Experience (QoE) forecasting service based on a vehicular networking use case. Specifically, we show how FL of eXplainable AI (XAI) models can be leveraged for the QoE forecasting task, and induces a benefit in terms of both accuracy, compared to local learning, and trustworthiness, thanks to the adoption of inherently interpretable models. Such considerations are supported by an extensive experimental analysis on a publicly available simulated dataset. Finally, we assessed how the learning process is affected by the system deployment and the performance of the underlying communication and computation infrastructure, through system-level simulations, which show the benefits of deploying the proposed framework in edge-based environments.

1. Introduction

The next generation of mobile networks is envisioned to leverage the most advanced methodologies from the field of Artificial Intelligence (AI), in order to support users running innovative applications and services, as well as to realize powerful and reliable in-network functionalities. Indeed, the design of the upcoming 6G network architecture is tightly intertwined with AI, as demonstrated by the activities carried out by major joint research efforts around the world, including European Union projects such as *DEDICAT 6G*[1], *6G Brains*[2] and *Hexa-X*[3] [1].

In fact, AI features are expected to be integrated *within* the mobile network, in order to improve a wide range of management tasks, such as orchestration of Virtual Network Functions, as well as user-plane operations, such as augmenting the capabilities of applications run by the users of the mobile network. One of the vertical sectors that are likely to be most impacted by these developments is the automotive field: for example, vehicular applications relying on video streaming transmitted over the network (e.g., see-through [2] or tele-operated driving operations [3]) may leverage AI to predict possible degradation of the Quality of Experience (QoE) based on Quality of Service (QoS)

* Corresponding author.
E-mail addresses: joseluis.corcuera@phd.unipi.it (J.L. Corcuera Bárcena), pietro.ducange@unipi.it (P. Ducange), francesco.marcelloni@unipi.it (F. Marcelloni), giovanni.nardini@unipi.it (G. Nardini), alessandro.noferi@phd.unipi.it (A. Noferi), alessandro.renda@unipi.it (A. Renda), fabrizio.ruffini@ing.unipi.it (F. Ruffini), a.schiavo2@studenti.unipi.it (A. Schiavo), giovanni.stea@unipi.it (G. Stea), antonio.virdis@unipi.it (A. Virdis).

[1] https://dedicat6~g.eu/, accessed November 2022
[2] https://6g-brains.eu/, accessed November 2022.
[3] https://hexa-x.eu, accessed November 2022.

https://doi.org/10.1016/j.comcom.2023.07.039
Received 7 November 2022; Received in revised form 17 March 2023; Accepted 27 July 2023
Available online 31 July 2023

metrics, in order to take countermeasures before the degradation occurs (e.g., avoiding dangerous maneuvers). The suitability of machine learning (ML) approaches to address the QoS-QoE prediction task has been widely investigated in the last few years [4–6]. However, just optimizing the accuracy of these systems is no longer enough and most of the efforts nowadays aim to enhance their trustworthiness.

B5G/6G networks will provide service to a very large number of devices, each one equipped with advanced sensing capabilities. This will generate a massive volume of valuable data, that can be exploited to build accurate AI models through ML. Data generated or sensed by users can be conveyed to the network and used as input to algorithms employed for learning AI models. These algorithms may possibly be executed within the network itself, which is foreseen to have in-built edge computing capabilities. However, the exploitation of user data to feed training algorithms at the network side poses concerns about privacy. For this reason, decentralized approaches that consider privacy as a requirement are rapidly emerging as potential candidates to be used in B5G/6G networks. Federated Learning (FL) [7] represents a major technological enabler in this regard, as it consists in a learning paradigm that allows multiple parties (i.e., data owners) to collaboratively learn an AI model without any disclosure of private raw data. The key intuition behind FL is that the training phase of AI models takes place locally, where data is generated, and models, rather than private data, are shared with a central orchestrator whose role is to aggregate them to produce a *global* model. As a consequence, FL paradigm becomes also particularly valuable in cases where data centralization for AI model training is ruled out due to intolerable communication and computation overhead.

While FL is meant to fulfill the privacy requirement, which is crucial for enhancing user's trust in AI services, making AI trustworthy requires one to explore other avenues at the same time [8]: indeed, the *explainability* of such AI systems is equally relevant and is the focus of the broad discipline of eXplainable AI (XAI) [9]. In a nutshell, an AI model can be defined explainable if it produces details or reasons to make its functioning clear and easy to understand; this property is inherent in so-called *transparent models*, such as decision trees or rule-based systems, and can be pursued for the so-called opaque models, such as deep neural networks and ensembles, through the adoption of *post-hoc explainability techniques*.

Notably, most renowned applications of FL exploit opaque models [10] as their mainstream guise is based on the collaborative optimization of a global differentiable objective function through adequate variants of stochastic gradient descent. As an example, FL of Deep Neural Networks is exploited for query suggestion improvement on Google Keyboard. However, it is often the case that other classes of models are preferable due their transparency and/or because they can still achieve competitive performance, e.g., when data comes in tabular form [11]. Thus, devising FL approaches for such models, and specifically XAI models, is currently gaining increasing attention [12–15]; notably, the concept of Federated XAI (Fed-XAI in short) has been recently awarded as key innovation by the EU Innovation Radar[4] and its applicability to an automated-vehicle networking use case in B5G/6G systems has been recently envisioned [16].

One of our previous papers [17] presented a preliminary study on how XAI models can be leveraged to address the problem of QoE forecasting in B5G/6G networks. Such work was carried out by exploiting a realistic dataset including QoS and QoE metrics for a video-streaming application, generated by using end-to-end, system-level simulations of a 5G network. Moreover, the basic concepts required for running FL within the B5G/6G network were briefly introduced.

This work takes a step forward by presenting a complete framework enabling FL-related services within future B5G/6G networks, according to the *as-a-service* paradigm. In B5G/6G networks, applications run by different mobile users have different quality requirements, and may exploit AI in different ways to improve the features they offer. In this context, FL combines the benefits of a potentially large number of participants (i.e., the mobile users) generating useful data to train AI models, while preserving the privacy of such data at the same time. Since different applications may require AI models with different objectives and requirements at possibly different times, the network should be able to provide flexible FL services to be instantiated when they are needed. The proposed *Federated Learning as-a-service* (FLaaS) framework allows mobile users to discover AI models made available by the network for several types of applications (e.g., QoE prediction for video-streaming applications), obtain them from the network and use them for their inference tasks. To this aim, the framework provides the mobile users with the protocols to join (and leave) a federation, and participate in the training of the corresponding (X)AI model, by exchanging local and global versions of the latter with the aggregation entity within the network.

Moreover, B5G/6G networks will provide connectivity to devices with different computing capabilities, ranging from high-end computers to resource-rich connected vehicles, to low-power IoT devices. The latters would not be able to use FL due to their computing constraints. Enabling FLaaS, some degrees of freedom exist in terms of the *placement* of its components within a network architecture. The central coordination infrastructure can be deployed *at the edge* of the network, i.e. exploiting the ETSI Multi-access Edge Computing (MEC) architecture. Moreover, distributed functions (e.g., user training modules) can be placed either at the user device itself, or offloaded to the edge as a dedicated application, hence allowing users with limited computation capabilities (e.g., IoT devices) to participate in FL operations.

There are many different applications that may be run on a 6G network enabled with our FLaaS framework. Beside the ones related to the automotive world – such as QoS prediction for cooperative or teleoperated driving, we can mention enhanced living for impaired people, where image acquisition via wearable cameras and sensors can be processed to identify obstacles or hazards, or assistance to individuals – e.g., identification of specific individuals (e.g., an unattended child) in a crowd. More applications and use cases are described, for instance, in [18].

The proposed approach is also validated through an extensive experimental analysis: we evaluate the performance of an XAI model learned in a federated fashion on a publicly available QoS-QoE dataset. Furthermore, the FL scheme is compared with two baselines: centralized learning and local learning. The former violates the requirement of privacy, but it allows us to quantify how much preserving it costs in terms of model accuracy; the latter, instead, guarantees privacy, but does not exploit any form of shared knowledge among participants. Accordingly, comparing against it allows us to shed light on the modeling benefit induced by the federated approach compared to local learning.

To summarize, the contributions of this paper are the following:

- we employ a state-of-the-art approach for FL of XAI models to tackle the problem of forecasting QoE of video streaming applications run by UEs in an automotive use case;
- we propose the FLaaS framework that provides the B5G/6G network with flexible mechanisms to allow mobile users to exploit FL services, and discuss the issues related to the placement of its components;
- we present results on the accuracy and the explainability of the XAI models obtained by our proposed FL algorithm on the QoE forecasting dataset;
- we evaluate the proposed FLaaS framework, and specifically the running time of the learning process, using a realistic end-to-end simulation environment that accurately models both communication and computation.

[4] https://www.innoradar.eu/innovation/45988

The rest of this paper is organized as follows: Section 2 provides the basic concepts about FL and XAI, and reviews the related work. Section 3 presents the algorithms we designed to enable the FL of XAI models, whereas Section 4 describes the FLaaS framework. We describe the case study and the dataset we produced to carry out our experiments in Section 5. The algorithms and the impact of the network on the proposed framework are evaluated in Section 6 and Section 7, respectively. Section 8 concludes our work.

2. Related work

Synergy between AI and wireless networks has been widely investigated [19], even in terms of challenges and opportunities for future B5G/6G networks [20,21]. In this section, we review the most relevant works in two fields that are deemed crucial for achieving trustworthiness in AI-empowered next generation wireless networks: FL and XAI. Moreover, we discuss existing works related to network protocols for FL.

2.1. Federated learning for wireless networks: applications and protocols

The adoption of FL in wireless networks has recently gained a significant momentum. In a recent review, Niknam et al. [22] discussed some key potential applications of FL in 5G networks: for example, FL may play a crucial role in enabling dynamic access to the spectrum, avoiding the disclosure of privacy-sensitive data, such as spectrum occupancy data, device nonlinearity information, and detection of abnormal signals (e.g., interference). Also, authors envision applicability of FL in the 5G core (5GC) network: FL may come into play in supporting collaborative learning over vertically partitioned data (i.e., different parties hold information about different features of the same instances). Specifically, each entity of the 5GC structure handles a subset of the features of a dataset related to the overall users in the network. For example, session management function handles session establishment (i.e., IP address allocation, traffic routing), while access mobility management function handles mobility information. The adoption of FL paradigm for collaborative data analysis by the network data analytics function (NWDAF) can mitigate privacy and security issues, compared to raw data sharing.

To the best of our knowledge, however, most of these visions have not yet found application in real-world scenarios. A recent proposal in this direction is represented by the work described in [23]: authors propose an approach based on Federated Support Vector Machine for mobile packet classification and, specifically, to predict personally identifiable information exposure and requests. Their approach is validated using three real-world datasets. In [24], FL of neural networks is exploited for Reference Signal Received Power (RSRP) estimation, and is enhanced with a sophisticated privacy-preserving mechanism to withstand membership inference attacks by malicious users.

Two challenges of FL in edge networking scenarios are discussed in [25]: model-weighting, i.e., selecting the weights given to different local models in the averaging phase, and node-dropping, i.e., the strategy for excluding some nodes from the FL process. Experiments on benchmark datasets, based on a 4-layer NN, reveal that decision making strategies cannot disregard the evaluation of the quality, and not only the quantity, of local data. An orthogonal challenge has been investigated in [26], stemming from the observation that edge ML has communication and on-device resource constraints: authors proposed a dynamic resource allocation strategy based on stochastic Lyapunov optimization, enabling adaptive FL at the network edge.

Within the growing interest in FL as a paradigm for privacy preservation, however, the aspect of explainability is generally neglected: most solutions revolve around FL of opaque or black-box models and thus cannot be considered fully trustworthy.

Another aspect to be taken into consideration relates to the communication between the distributed participants and the central aggregator to build an AI model. Communication protocols enabling the exchange of the information between such entities play a crucial role in determining the effectiveness and performance of the FL approach.

The authors of [27] provide a survey summarizing the main research efforts in defining the hardware and software platforms required to deploy FL, as well as the network protocols that have been proposed to enable the information exchange between entities. In this regard, the existing works consider wireless networks as the main usage scenario for FL, and they mainly focus on the procedures that are needed to send and receive local and global models, as well as the algorithms to perform the so-called client selection, i.e. choosing the most suitable clients that will participate in the training process. Protocols addressing security and privacy aspects are also discussed. Moreover, the survey lists the main use cases and applications that can benefit from the employment of FL.

In [28], authors provide an overview about the interplay between FL and Edge Computing. In particular, some challenges that can be encountered when implementing FL at the edge of the network are discussed, such as the communication cost and resource allocation. Although the above works are informative about the challenges related to the communication protocols required to enable FL, it can also be noted that some aspects have not been widely investigated by the state of the art. For example, the problem of discovering available FL services and the procedures required to join them have not been discussed. In general, to the best of our knowledge, the idea of considering the network as a provider of FL services – e.g., according to the as-a-service paradigm – has not been proposed so far. Likewise, quantifying the impact of the radio access network on FL and the benefits of using MEC are still open research issues.

Our work aims to provide a comprehensive and flexible framework that covers all the aspects required by the entities involved in an FL process, from service discovery to training and service termination, by also taking into account different deployment options, such as exploiting applications at the edge of the network to perform the local training of the models.

2.2. Explainable artificial intelligence for wireless networks

Some recent works [29,30] have reviewed the current status of AI-enabled cellular networks. Specifically, authors in [29] provide an overview of the key thrusts in AI for wireless communication from both an industry and a research perspective. The former category of thrusts include NWDAF specification for data collection and analytics defined by 3GPP, and remarkable initiatives such as the O-RAN alliance[5] and the ETSI Industry Specification Group on Experiential Networked Intelligence[6]. The latter category includes several examples of successful adoption of AI techniques to the PHY, MAC and Network layers, with applications ranging from channel estimation and prediction, to dynamic spectrum access and resource management and scheduling. Interestingly, the overview of the authors highlights that, in the last few years, the most prominent tools adopted in this context rely on opaque models based on neural networks. A similar overview is presented in [30], with a specific focus on the trustworthiness of AI/ML techniques. This survey reports some representative examples of the adoption of AI methods in wireless communication along with the degree of explainability that is generally accorded to the methods themselves. Most of the employed approaches feature a level of explainability that is defined as none, very low and low. Among the solutions with a medium degree of explainability, authors in [31] propose a Belief Network for passive in-situ diagnosis of Wireless Sensor Networks based on a lightweight packet marking scheme; authors in [32] propose a Bayesian Learning approach (Expectation Propagation) within

[5] https://www.o-ran.org/, accessed November 2022.

[6] https://www.etsi.org/technologies/experiential-networked-intelligence, accessed November 2022.

Table 1
Overview of related works.

Ref	Contribution	FL	XAI	QoE
[22]	Motivation, opportunities, challenges, potential applications of FL in 5G	✓		
[23]	FL of support vector machine for mobile packet classification	✓		
[24]	FL of NNs for Reference Signal Received Power (RSRP) estimation	✓		
[25]	Challenges of FL in edge networking scenarios; experiments on benchmark datasets	✓		
[26]	Dynamic resource allocation strategy enabling adaptive FL at the network edge; experiments on benchmark datasets	✓		
[27]	A survey on FL enabling technologies, protocols, and applications	✓		
[28]	Survey of FL in mobile edge networks	✓		
[31]	Passive in-situ diagnosis of Wireless Sensor Networks based on a lightweight packet marking scheme		✓	
[32]	Active learning of interference channels in cognitive radio networks		✓	
[33]	Multimedia QoE classification with decision trees		✓	✓
[34]	Multimedia QoE classification with random forests		✓	✓
[35]	Multimedia QoE classification with support vector machine		✓	✓
[4]	Multimedia QoE classification with bayesian networks		✓	✓
[5]	Multimedia QoE classification with tree-based models		✓	✓
[15]	Multimedia QoE forecasting with regression tree		✓	✓
[36]	Multimedia QoE forecasting with Hoeffding regression tree		✓	✓
Our	Multimedia QoE forecasting with Federated TSK FRBS	✓	✓	✓

a constrained dynamic programming framework for active learning of interference channels in cognitive radio networks. The only approach that features a high degree of explainability is described in [33]: a decision tree model is used to predict in real-time and with low complexity the QoE in wireless multimedia communications; a forward error correction scheme leverages the prediction to prevent quality degradation.

We also report that several works recently proposed to address the QoE prediction problem with AI tools: prediction of video stalling or starvation under various (simulated) network conditions has been tackled mainly with "gray-" or "black-box" models, i.e. Random Forest [34], Support Vector Machine [35] and Bayesian Networks [4].

In one of our recent works, [5], we describe an experimental campaign on the dataset presented in [4], aimed at investigating the trade-off between accuracy and explainability in tree-based models: results showed that multi-way Decision Trees, enhanced with concepts from fuzzy set theory, ensure high level of interpretability and achieve competitive levels of performance compared to Random Forests. The analyses performed in [4,5], however, are based on aggregate QoS and QoE metrics, and neglect the streaming nature of such measurements.

The present work stems from a preliminary investigation [17], which revolves around a novel dataset of QoS-QoE timeseries obtained through realistic 5G network simulations. The preliminary experimental analysis showed that an inherently explainable model, i.e., Regression Tree (RT), can be regarded as a suitable solution to address even the forecasting task on QoE timeseries. Later, in [36], we resorted to the same dataset to investigate the performance of the Hoeffding RT, a variant of RT tailored for incremental learning over streaming data.

However, to the best of our knowledge, none of the previous works has addressed the challenge of privacy preservation: conversely, all of them rely on the strong assumption that a global training set can be built, possibly by collecting data produced by different peripheral nodes.

Table 1 summarizes the related works discussed in this section, emphasizing their relevance to the three main areas covered in our work (FL, XAI, QoE): our proposal to adopt FL of XAI models to tackle the QoE forecasting problem, unprecedented in the literature, represents indeed a leap forward towards trustworthy AI in next generation wireless networks.

3. Background: An approach for federated learning of XAI models

A possible approach to achieve explainability is through the adoption of inherently interpretable models. In this regard, Decision Trees (DTs) and Rule-Based Systems (RBSs) are generally considered among the most *transparent* models, as their operation and characteristics can be easily understood by a human. For both models, in fact, the inference strategy is traced back to the evaluation of *if-then* rules, which is quite akin to some human rational processes. In this paper we exploit a particular type of RBSs, namely Takagi–Sugeno–Kang Fuzzy RBSs (TSK-FRBSs) [37], which are particularly useful in dealing with regression tasks for two reasons: on one hand they have proven to achieve high modeling capability of complex systems, on the other hand the adoption of concepts from fuzzy set theory may further enhance both model performance in scenarios with some degree of noise, and interpretability, thanks to a natural linguistic representation of numeric variables.

In this section we first provide some preliminaries about TSK-FRBSs, and then we recall a recently proposed approach for FL of TSK-FRBSs, which is employed in the experimental analysis discussed in this paper.

3.1. Takagi–Sugeno–Kang Fuzzy rule based systems: Preliminaries

For the purpose of describing the key concepts of the model, we refer to the centralized scenario, in which the learning stage is performed based on a dataset available at a single location.

Let $\mathbf{X} = \{X_1, X_2, \dots, X_F\}$ be a set of input variables and Y the output variable. A generic instance of the dataset is in the form $\mathbf{x} = [x_1, x_2, \dots, x_F]^T$ and has an associated target value y. Let U_f, $f = 1, 2, \dots, F$, be the universe of discourse of variable X_f. Let $P_f = \{A_{f,1}, A_{f,2}, \dots, A_{f,T_f}\}$ be a fuzzy partition over U_f with T_f fuzzy sets, each labeled with a linguistic term. Finally, let K be the number of rules in the rule base. The generic kth rule is in the form:

$$R_k: \textbf{IF } X_1 \textit{ is } A_{1,j_{k,1}} \textbf{ AND } \dots \textbf{ AND } X_F \textit{ is } A_{F,j_{k,F}} \quad \textbf{THEN } y_k = f(\mathbf{x}) \tag{1}$$

where $j_{k,n} \in [1, T_f]$ identifies the index of the fuzzy set of partition P_f. In zero order TSK-FRBS the consequent function of the generic rule R_k is a constant value, namely $y_k = \gamma_{k,0}$. In first order TSK-FRBS the consequent function of the generic rule R_k is a linear combination of the elements of $\mathbf{x}$, parameterized by the vector of coefficients $\boldsymbol{\gamma_k} = \{\gamma_{k,0}, \gamma_{k,1}, \dots \gamma_{k,F}\}$: in this case the output is evaluated as $y_k = \gamma_{k,0} + \sum_{i=1}^{F} \gamma_{k,i} \cdot x_i$.

When an input pattern $\mathbf{x}_i$ is fed to a TSK-FRBS, the strength of activation of each rule is computed as follows:

$$w_k(\mathbf{x}) = \prod_{f=1}^{F} \mu_{f,j_{k,f}}(x_f) \quad \text{for} \quad k = 1, \dots, K \tag{2}$$

where K is the number of rules in the rule base and $\mu_{f,j_{k,f}}(x_f)$ is the membership degree of x_f to the fuzzy set $A_{f,j_{k,f}}$. Once the activated rules are identified, two inference strategies are viable, namely weighted average and maximum matching. In the *weighted average*

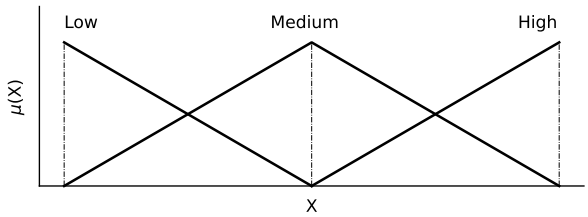


Fig. 1. An example of strong triangular uniform fuzzy partition with three fuzzy sets.

strategy, the output depends on the K outputs obtained from as many rules. Formally:

$$\hat{y}(\mathbf{x}) = \sum_{k=1}^{K} \left(\frac{w_k(\mathbf{x})}{\sum_{h=1}^{K} w_h(\mathbf{x})} \right) \cdot y_k(\mathbf{x}) \tag{3}$$

In the *maximum matching* strategy, the output is determined by using the rule with the highest strength of activation. Evidently, this strategy entails a higher level of interpretability, compared to the weighted average one, as the output can be motivated by looking at a single rule.

The most popular approach to learn TSK-FRBS consists of two stages: structure identification and model parameter identification. In the former stage, the number of rules and the conditional part of the rules are determined; this is typically done either with grid-partitioning of the input space or exploiting fuzzy clustering methods [38]. In the latter stage, with fixed antecedents, parameters of the local linear models are learned by pseudo-inversion or by applying the recursive least square method. Alternative approaches have been proposed for optimizing TSK-FRBSs, including genetic algorithms [39] or mini batch gradient descent [40,41].

In the following, we describe the procedure proposed in [14] and adopted in our experimental analysis. The antecedent parameters are determined as follows. First, a strong triangular uniform fuzzy partition is defined on each normalized input variable: an example of such partition with three fuzzy sets, respectively labeled as *low*, *medium* and *high*, is shown in Fig. 1.

Then, for each variable value x_f of each training sample, we compute the membership degree to the fuzzy sets of the uniform fuzzy partition defined on X_f: a condition "X_f *is* $A_{f,j_{k,f}}$" is added to the antecedent, where $A_{f,j_{k,f}}$ is the fuzzy set with the maximum membership degree. The antecedent of a rule is finally obtained as conjunction of the conditions associated with a sample. The same antecedent can be generated by a number of samples. In this case, only one antecedent is considered in the following steps. Once rule antecedent parameters are determined, the corresponding consequent parameters are estimated through the weighted least-squared method applied to all the samples which activate the antecedent. More details on the method adopted for generating single local models can be found in [14].

3.2. A federated approach for learning Takagi–Sugeno–Kang Fuzzy rule based systems

An approach for FL of interpretable TSK-FRBSs has been recently presented in [14]. A schematic overview is given in Fig. 2.

The process involves the following actors: N FL Local Managers (FLLMs), each associated with a given data owner, and one FL Global Manager (FLGM), that is the central entity responsible for model aggregation. Such a setting is consistent with *standard* FL over centralized communication topology: unlike *standard* FL, however, we consider FL of XAI models. The proposed Fed-XAI approach is supported by the novel FLaaS framework: a detailed description of the framework and the involved entities will be provided in Section 4.

The one-shot procedure for generating the Federated TSK-FRBS encompasses the following four steps:

(1) the FLGM transmits the configuration parameters to the learning modules;
(2) the learning module of each FLLM generates the local TSK-FRBS based on private data;
(3) each FLLM transmits the local TSK-FRBS to the FLGM;
(4) the FLGM generates the Federated TSK-FRBS by aggregating the local models.

In the first step, the configuration parameters include all the relevant information for generating consistent local models (e.g., the domain of definition of the attributes for data normalization and the number of fuzzy sets for fuzzy grid partitioning of the input space). In the second step, a TSK-FRBS is generated by each local learning module, according to the procedure reported in Section 3.1. Once local rule bases are transmitted to the FLGM (third step), the aggregation procedure is performed (fourth step): first, the FLGM generates a preliminary global rule base as the juxtaposition of the rules collected from the clients. Then the rule base is refined by solving rule conflicts (rules with the same antecedent but different consequents). A conflict is handled by replacing the set of conflicting rules with a single rule with the same antecedent and a novel consequent, obtained by averaging the coefficients of the consequents of the conflicting rules. This averaging operation is weighted, as each rule has an associated *rule weight* computed as the harmonic mean of its support (i.e., how much a rule is activated by the sample of the training set), and confidence (i.e., average quality of the prediction of the rule measured on the training set).

4. A framework for federated learning as a service

B5G/6G networks are envisaged to provide User Equipments (UEs) with procedures to take advantage of intelligent services (e.g., forecasting of QoE) by exploiting AI models that are built according to the FL paradigm. This can be realized by endowing the B5G/6G network with an FLaaS framework that allows UEs to discover FL services made available by the network, obtain the corresponding "federated" AI model and, possibly, participate in training it. This section describes the FLaaS framework, including its modules, protocols and functions. This framework has been designed in the context of the Hexa-X EU project [42], and a proof-of-concept implementation has been developed on top of the Simu5G OMNeT++-based public simulator [43]. The FLaaS framework will be evaluated in Section 7.

4.1. Description of the framework

In this section, we first present the main components of the framework, also providing some taxonomy. Then, we describe the protocols that define the main interactions among the different entities of the FLaaS framework.

4.1.1. Main components of the framework

In a production context, the FLaaS framework should be used by two classes of users:

- Network operators or third parties, which define and onboard FL services to be adopted by users, using ad hoc management interfaces;
- UEs, which discover existing FL services, join/leave FL processes (i.e., running instances of the above services), possibly participate in the FL process by contributing to learning a global model, or just use whatever is available for their own inferences.

We refer to an *FL service* as a collaborative learning task dedicated to a specific application (e.g., QoE prediction for automotive applications). The FLaaS framework provides a collection of FL services that can be instantiated when needed, i.e., when an AI model for that specific application is requested by some UE. We refer to a running

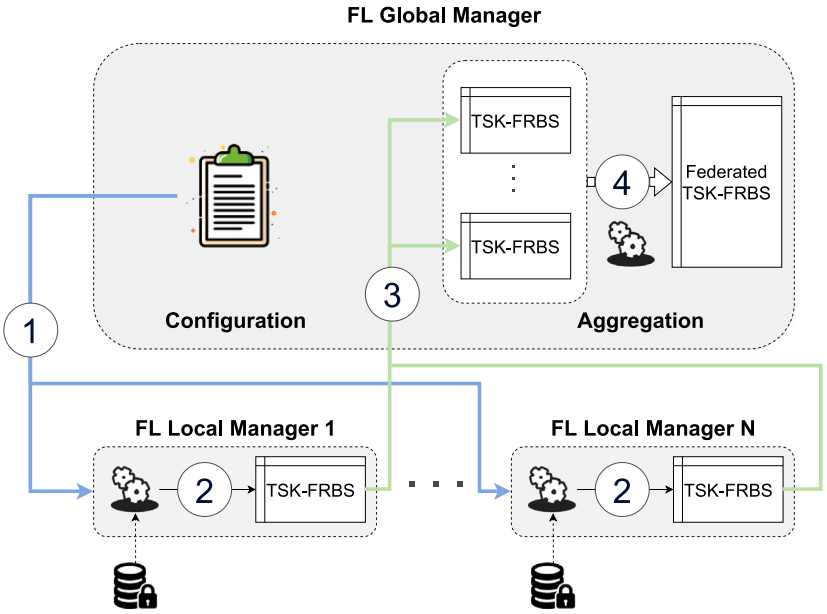


Fig. 2. Overview of the proposed approach for FL of TSK-FRBSs.

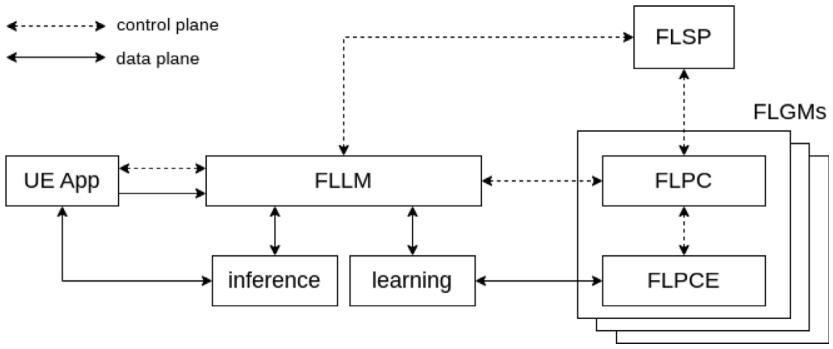


Fig. 3. Overview of the components involved in the FLaaS framework.

instance of an FL service as *FL process*. Multiple FL processes referring to the same FL service may be instantiated simultaneously in the network, for example handling disjoint sets of UEs in different geographical areas.

With reference to Fig. 3, the main component of the framework is the *FL service provider* (FLSP), which is located in the network and maintains both a library of available FL services and a list of active (running) FL processes in the system. Each UE is supported by an FLLM that interacts with the network side of the FLaaS framework on behalf of the UE application. It manages both the learning and inferencing process of the UE. When the UE wants to discover an FL process, its FLLM queries the FLSP. The latter is also responsible for coordinating and triggering the activation of the entities that will actually execute the FL processes. For this purpose, the FLSP will interact with the system orchestrator, whose type will depend on the actual deployment, e.g., the MEC Orchestrator in an ETSI MEC environment or a virtual infrastructure manager, such as kubernetes, in a dedicated deployment. Each active FL process is handled by its FLGM, which is in turn composed of two modules, namely the *FL process controller* (FLPC) and the *FL process computation engine* (FLPCE). The former manages control-plane interactions with the FLSP (e.g., authorization grants) and the FLLMs, whereas the latter acts as the aggregator of the FL process, i.e., the entity that actually builds the global AI model. To do this, the FLPCE must exchange local and global AI model updates with the learning module of the FLLMs, which, in their turn, act as collaborators in the process of training the global AI model.

It is worth mentioning that the deployment of the above components is immaterial: the FLSP may reside either in the core cloud or at the edge of the B5G/6G network. Likewise, the FLLM may reside at either the UE or the network edge. This last option may be necessary with resource-constrained UEs, such as IoT devices.

The issue of function placement in a communication/computation architecture and its implications will be discussed at the end of this section. The description that follows is general, and can be matched to different function placements.

4.1.2. Description of the protocols involved in the framework

In the following, we describe the functionalities that enable the FLaaS framework. For each operation, we describe the necessary interactions among the components and, where applicable, we define alternative options that may present different trade-offs. Where it is relevant, we also report sequence diagrams for specific interactions.

Onboarding of an FL service. The FLSP exports a management interface to network operators or third parties to allow them to interact with the FLaaS framework. This interface provides functions to register/unregister an FL service. Registration messages include information

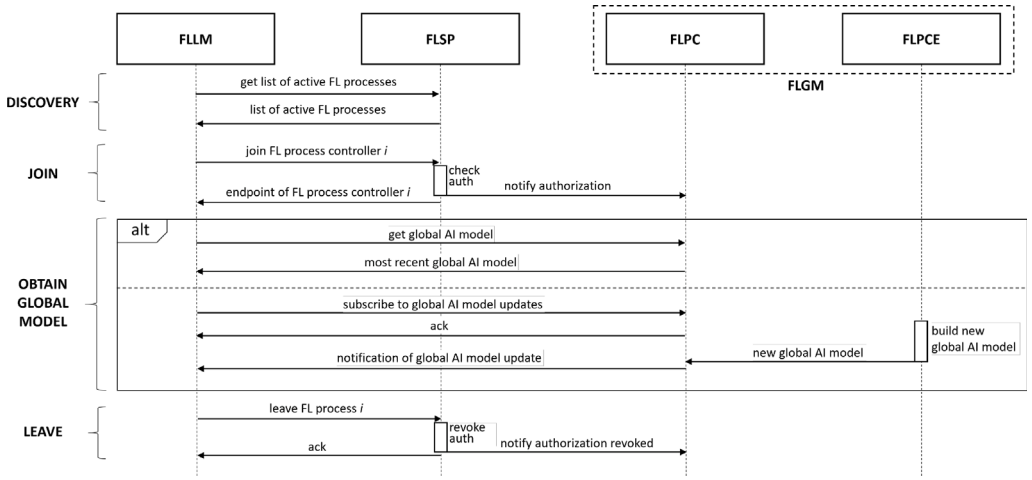


Fig. 4. Procedures to discover and join an FL process, obtain the global model, and leave the FL process.

such as the description of the service (i.e., its objective), the application image and configuration files, possible constraints (e.g., minimum capabilities of the UEs participating in the training) and type of training (e.g., synchronous or asynchronous, see the part on training operation below). This creates a library of FL services, that users can then query. The interface also provides mechanisms to allow the operator or third party to instantiate an FL process. When this is requested, the FLSP requests the deployment of the entities – e.g., virtual machines or containers – acting as FLPC and FLPCE, specifying the application image that was previously onboarded within the FLSP.

Service Discovery. This phase is initiated by UEs to obtain the list of available FL services and/or running FL processes. The FLLM queries the FLSP to obtain the list of available FL services. The response may include conditions that need to be met for the FLSP to start a new instance of that FL service, i.e., an FL process. Alternatively, the FLLM can obtain the list of the active FL processes in the system as depicted in Fig. 4. The request messages may include fields for filtering the required FL service the FLLM may be interested in (e.g., services available in a given geographical area, or services relevant for specific use cases only). Responses include attributes such as the current status of a service/process, and the type of training (synchronous, asynchronous). If there are no running FL processes for an FL service yet, the FLLM can signal the FLSP that it is *available* for using an FL process. The decision to start a new FL process is made by the FLSP, possibly when some conditions hold (e.g., a minimum number of interested FLLMs) and also for security reasons. Accordingly, the FLSP acknowledges the FLLM's intent and saves the information in a data structure for later use. When said condition is verified later on, the FLSP actually starts the FL process by deploying its FLGM (i.e., its FLPC and FLPCE) and notifies all the FLLMs that have registered so far. This is exemplified in Fig. 5, where FP_i is a generic name for an FL service for which no FL processes are running.

Joining an FL process. Once the UE is aware of the available FL services and processes, it may be interested in obtaining the corresponding global AI model. Note that it is not strictly required that the UE participates in the construction of the model itself. With reference to Fig. 4, the FLLM contacts the FLSP, which checks whether the authorization to the UE can be granted and, in the affirmative case, returns the endpoint of the FLPC (e.g., IP address/port) to the FLLM. Note that this allows for *arbitrary service orchestration* policies to be implemented at the FLSP. For instance, the FLSP may want to partition FLLMs requesting the same service based on some user characteristics (possibly connected to the type of data in their possession).

Once the FLLM can connect to an FL process, it can either (i) obtain the global AI model available for the selected FL process, if any, or (ii) take part in the construction of the global AI model. Below, the involved procedures for both cases are described.

Obtaining the global AI model. The FLLM can request the global model from the FLPC according to either a request–response or a subscribe-notification paradigm. Both alternatives are reported in Fig. 4. The first option allows the FLLM to obtain the model whenever it needs it. Timestamping is used to avoid sending the same global AI model twice to a requesting FLLM, thus wasting network resources (we recall that global AI models can easily be in the order of megabytes). With a subscribe-notification pattern, instead, the FLPC sends global AI model updates when the FLPCE produces a new model. A subscription message from the FLLM may specify filter conditions that regulate the number of updates it expects to receive (e.g., one model update per day).

Joining the training of the global AI model. With reference to Fig. 6, the FLLM notifies the FLPC of its intention to participate in the building of the global AI model. The FLPC must either accept or reject the request. Motivations for rejection may include, for instance, insufficient computational resources available at the UE for training a local model, or position of the UE outside the geographical area of interest for the given FL process. Note that an FLLM is not allowed to join the training without first completing the join operation described above, since the FLSP must first authorize the FLLM itself. This can prevent some type of training poisoning done by malicious users that bypass the authorization of the FLSP.

Training phase. This phase starts after the FLLM has joined the training process, as shown in Fig. 6. First, the FLPC and the FLLM exchange control-plane information. In particular, the FLPC checks the current availability of FLLMs that joined the training by querying them. If an FLLM replies negatively (e.g., it has not enough computing resources at the moment), or the FLPC does not receive a reply from it within a certain deadline (e.g., it has no connectivity at the moment), then the FLLM is excluded from the current training phase. Then, data-plane information flow occurs between the FLPCE and the FLLM. This entails sending the global model from the FLPCE to the local learning module and sending the local model in the opposite direction. In both cases, models can be sent either as a whole, or by incremental updates. Incremental updates can be used with well-structured models, e.g. rules in a fuzzy set, and can lead to considerable savings of network resources. The FLPCE can be configured to adopt both a *synchronous* and an *asynchronous* training strategy:

- In synchronous mode (shown in Fig. 6), training takes place in multiple *rounds*. For each round, the FLPCE selects the participants, i.e., a subset of the UEs that have declared themselves available to train the model, and notifies their learning module when a round has started. If some UE becomes unavailable after

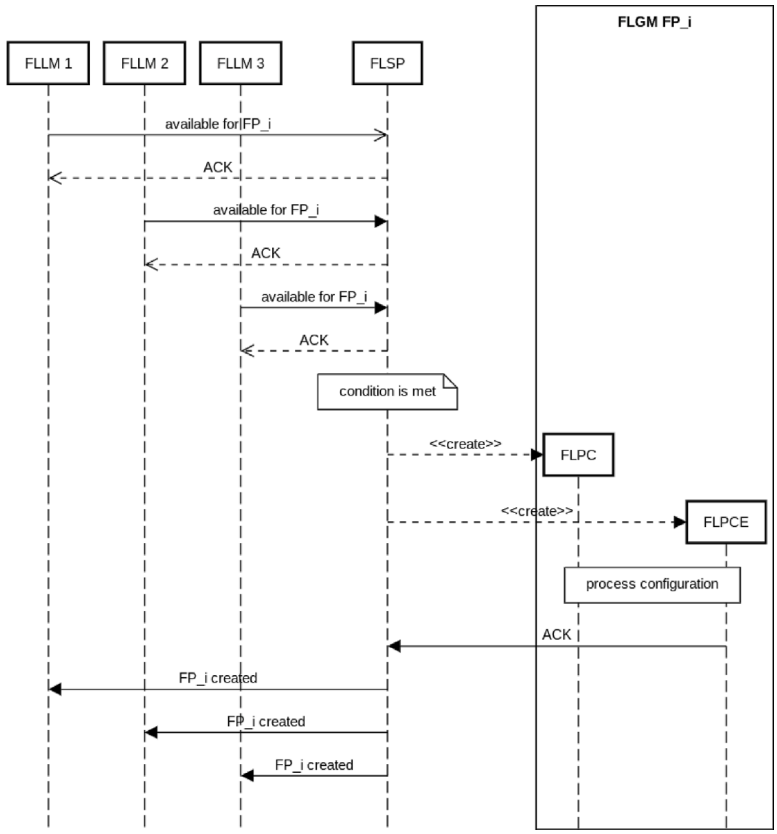


Fig. 5. Procedures to start a new FLGM handling a new process FP_i.

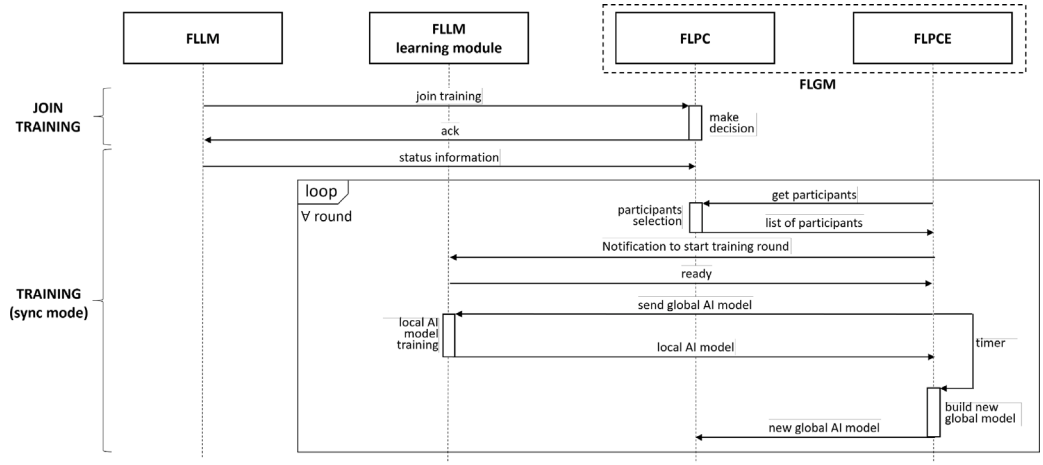


Fig. 6. Procedures to join the training of a global AI model, and to train it.

being selected, it sends a NACK back to the FLPCE, which may then select other UEs before starting the round. Upon reception of an ACK from the selected participants, the FLPCE sends them the global model (or the parameters configuration thereof). In turn, each FLLM starts its local training and sends its updated local model to the FLPCE. The FLPCE aggregates the received local models and builds a new version of the global one. This may happen, depending on the configuration, when a predefined percentage of the participants (or possibly all) have sent their update, or after a predefined deadline.

- In asynchronous mode, whenever a UE joins the training, its learning module receives the global model from the FLPCE (or the parameters configuration thereof), performs the local training and sends the model back to the FLPCE. The latter aggregates such models according to a configurable policy. For instance, it can update the new global model as soon as it receives a local model from any UE, or when it receives the local models from a predefined number of UEs, or when a timer expires.

Once the global AI model has been produced, the FLPCE sends it to the FLPC, which stores it and provides it to the FLLMs requesting it.

Leaving the training of the global AI model. A UE may decide to stop participating in the construction of the global model. For instance, this may occur when its computational resources become too scarce, or when it has not been selected for participating in a round for a long time. In this case, its FLLM sends a leave request to the FLPC, which revokes previous authorizations and responds with an ACK. On

the other hand, the FLPC itself may decide to remove a UE from the training process. For instance, this may occur because the UE moved outside the geographical area of interest for the FL process, or it is too slow to run its local training. When such an event occurs, the FLPC sends a notification to the UE's FLLM, which in turn replies with an ACK.

Leaving an FL process. A UE may decide to leave an FL process, e.g., because it is no longer interested in using a global AI model for the given FL service. In this case, a message is sent from the UE's FLLM to the FLSP as shown in Fig. 4, which takes care of withdrawing the authorization to contact the corresponding FLPC. If the operation is done before the UE left the training (see above), the FLSP is also responsible for informing the FLPC to remove the UE from the set of potential participants to the training of the global AI model.

4.2. Deployment and function placement

B5G/6G networks will benefit from an integrated computation infrastructure, possibly based on ETSI MEC. The latter is an industrial standard that defines entities, functions and control-plane interactions that allow UEs to request the activation/deactivation of services, called MEC apps, running on the MEC infrastructure [44]. FLaaS can exploit the above infrastructure, leveraging high-performance on-demand computation capabilities as well as value-added services (e.g., location of UEs, up-to-date information on radio conditions of single UEs, etc.). FLaaS function placement can have an impact on the type and amount of traffic that flows through the Radio Access Network (RAN). We describe the possible options below.

FLaaS service provisioning infrastructure (i.e., the FLSP and – for each service – the FLPC and FLPCE) is meant to run in the MEC.

As far as the modules logically connected to the UE are concerned, i.e., the FLLM, the inferencing module and the learning module, there are two options:

(a) They might reside at the UE itself, if it has enough computational resources to run them;
(b) they might reside on the MEC infrastructure, running as MEC apps on behalf of the UE.

Options (a) and (b) are not mutually exclusive: they can be used by different UEs in the same FLaaS process, and the same UE may want to switch from one to the other over time, depending on the circumstances. We envisage that option (b) will be preferred by resource-constrained UEs. In fact, it will act as an enabler, allowing such range of UEs to exploit FLaaS benefits at an affordable cost.

As outlined, the above options impact communications as well. Under option (a), the RAN transmits *models* only (global in the downlink, local in the uplink). A model is a relatively large file, which needs to be delivered fast (delaying it – in either direction – slows down the entire learning process). This requires an ad-hoc resource scheduling strategy at the MAC layer, which can grant large amounts of resources to a single user for bursty transmissions. A cross-layer approach, in which application-level UE selection (at the FLPC) is made also considering UE radio conditions or location would be beneficial. Such an approach is made possible in a MEC environment by the provision of *MEC services*. A MEC application such as the FLPC can query the UE radio conditions using the Radio Network Information Service (RNIS), or the Location Service, both standardized in the ETSI MEC, and acquire in real time the necessary knowledge to schedule updates efficiently.

Under option (b), instead, *raw data*, and these alone, flow in the uplink. All *model* exchanges occur among MEC apps, and hence do not affect the RAN. It can be assumed that intra-MEC communications, occurring on a well-provisioned wired infrastructure, do not constitute a bottleneck or require clever scheduling strategies. On the other hand, the flow of raw data in the uplink does require some scheduling: raw data are sensed at the UE. They can be periodic (e.g., a sensor acquisition) or sporadic (e.g., the occurrence of anomalous events), and they are unlikely to be more than a few bytes each. Well-known MAC-level scheduling strategies, such as periodic grants, are already available and can be used for this type of data. We observe that the MEC infrastructure may indeed play a role in *reducing* the amount of such data traversing the RAN: for instance, radio conditions can be acquired by the UE MEC app directly from the RNIS MEC service, without involving the UE itself and RAN-based communications.

Finally, we add a few words to dispel the notion that option b) may defy the whole purpose of FL, i.e. to preserve privacy by preventing UE data to fall in the hands of third parties. Under option (b), raw data are *not* shared with third parties. The UE MEC app(s) running in the MEC do not send them anywhere, and only exchange *models* with the service provisioning side of the FLaaS infrastructure. MEC itself can be expected to be a trusted infrastructure, with virtual machine/container isolation ensured through standard OS security means [45]. Likewise, transmission over the RAN and the core network of the mobile operator is expected to be secure and private through authentication and ciphering mechanisms natively provided by the network itself [46]. The issue of UE mobility and MEC App migration requires some considerations too. MEC Apps may in fact migrate to ensure geographical proximity (hence short latency) to a mobile user. MEC App migration policies are not under the control of the UE, and anyway App migration times are normally larger than handover times. Accordingly, there will be time intervals when the UEs and MEC Apps cannot communicate directly via the RAN alone, and UE data must instead traverse the core network to get to its destination. However, this will not represent a privacy problem, as long as the routing to the MEC app does not leave the operator's network.

5. An example of FL service: the quality of experience forecasting case study

In this paper we focus on a specific FL service, designed for addressing the QoE forecasting task in an automotive case study. Our experimental analysis stems from the one presented in [17]: we tackle the problem of QoE forecasting in B5G/6G networks exploiting the same simulated, publicly available, QoE dataset[7].

In this section, we recall the key concepts regarding the investigated use case, the simulation environment, the generation of the dataset and the main related preprocessing steps.

5.1. Description of the use case

We consider a scenario where vehicles are connected to the mobile network, hence acting as UEs, and play real-time video streams whose perceived quality is relevant to determine the availability of some advanced driving assistance system. For example, in a see-through application, a vehicle receives the live video acquired by the camera of the vehicle in front of it, which helps the driver perform a safer overtake maneuver. Intuitively, such a service is only safe if one can rely on the video to be displayed continuously and with high quality for the entire duration of the maneuver, which may require several seconds. Thus, in order to decide whether to start such maneuver at all, we need to predict the QoE perceived by UEs in the next future by leveraging real-time QoS and QoE data generated by the UEs themselves. Similar applications are investigated in [47–52].

Considering the see-through application, we assume that the video streaming does not occur directly between UEs (i.e., using device-to-device communications). Instead, we consider a scenario in which the sending UE sends its live video stream to an application running at the edge of the network (e.g., a MEC application in a MEC host). In turn, the

[7] http://docenti.ing.unipi.it/g.nardini/ai6g_qoe_dataset.html, accessed November 2022.

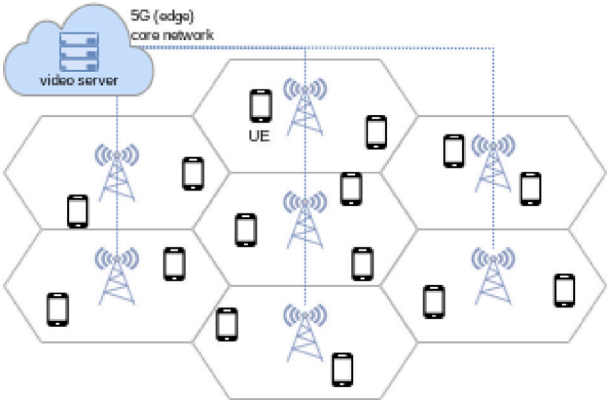


Fig. 7. Simulation scenario.

edge application forwards the video stream to the destination UE. This is done because the edge application may (or may not) perform some processing on the incoming video stream. For example, it may augment the video with some context information – e.g., indicating the position of traffic lights – or it may try to recover possible losses occurred in the uplink transmission.

In this scenario, each UE receiving a video stream instantiates its FLLM (either locally or at the MEC), since it is interested in knowing the future QoE of the video. The FLSP is deployed at the MEC and, upon the reception of the first discovery request from an FLLM, it instantiates the FLGM that handles the requested FL process. We assume that a UE receiving the video stream collects both QoS (e.g., inter-arrival time and loss rate of application packets) and QoE (e.g., percentage of frames that are correctly displayed) metrics while receiving the video stream, and makes such data available to its FLLM. Context information such as the UE's position and speed can also be part of the dataset to train an ML model. Each UE collects the above metrics at discrete time periods, hence the UE produces a vector for each metric, whose ith element is the value of the metric collected at time i. If the FLLM performing the local model training is deployed at the MEC, network-wise metrics such as the average utilization of the cell can also be leveraged to provide more meaningful predictions. This is the scenario we consider for the QoE forecasting use case in this section. All the above metrics will be exploited to predict the value of a *target QoE metric* (or a set thereof) at a time in the future.

To support the above prediction, we need to train an ML model with a dataset that includes realistic QoS and QoE metrics obtained from the mobile network. However, mobile network data are notoriously hard to get from operators, for confidentiality reasons, and their level of detail is often coarse (e.g., delays and losses of individual flows tend not to be logged at sub-second timescales, if at all). For this reason we relied on network simulations to generate the relevant dataset. Since the definition of the 6G RAN architecture is still underway at the time of writing, using a 5G RAN simulator – namely Simu5G – is the only viable choice. The next section will present the features of such simulation software.

5.2. Simu5G

Simu5G [43] is an open-source *model library* for the OMNeT++ discrete-event simulation framework [53], and provides the modules for the simulation of the data plane of 4G and 5G mobile networks. In particular, Simu5G provides the modules that represent the UEs and the base stations (gNBs) of a 4G/5G network. They are modeled by a device equipped with a NewRadio (NR) Network Interface Card (NIC) submodule, alongside the submodules for the other layers of the NR protocol stack. The NR NIC, in turn, implements the data-plane functionalities of all the sublayers of the NR protocol stack, from Packet Data Convergence Protocol (PDCP) to the physical layer. Since the scope of Simu5G is not to evaluate the performance of link-level transmissions, the physical layer is implemented via realistic channel models (i.e., taken from 3GPP standard documents) that abstract the transmissions of radio symbols over the wireless medium and only simulate the effects on their correct reception. This also allows Simu5G to reduce the simulation complexity and simulate large-scale network scenarios (e.g., composed of tens of gNBs and hundreds of UEs). Specific NR features are also supported by Simu5G, such as dual connectivity, carrier aggregation, multiple numerologies, frequency- and (flexible) time-division duplexing. Moreover, Simu5G provides a realistic modeling of MEC compliant with the ETSI specifications [54]. The latter is implemented using standard-compliant RESTful interfaces to the applications running on the UE and to the MEC apps running on MEC hosts, so that real applications can be interfaced with it, also in real time (e.g., to emulate a network scenario for demonstration purposes [55]).

Being part of the OMNeT++ ecosystem allows Simu5G to exploit and integrate many other libraries made available by the community and the OMNeT++ developers themselves, like INET that provides the modules for Internet protocols (e.g., the whole TCP/IP stack) and devices (e.g., IP-based routers) [56]. This allows Simu5G users to simulate arbitrarily complex network scenarios to evaluate end-to-end performance of applications and services exploiting the 5G radio access network, as well as the performance of the 5G network itself.

In the following, we describe how we exploited Simu5G to evaluate our proposed FLaaS framework and to generate a realistic dataset related to video-streaming application in B5G/6G networks, which will be used to assess the performance of the algorithms for FL of XAI models.

5.3. Generation of the dataset

In order to generate a realistic dataset for the use case presented in Section 5.1, we implemented a client–server video-streaming application within Simu5G. The server sends a video stream to the client following a trace-based approach: sending rate, size and type of video frames are read from a trace file generated from real videos via a dedicated command of the FFmpeg library.[8] Traces were obtained from three dash-camera videos, so as to reproduce a see-through scenario.[9] The video is transmitted via the Real-time Transport Protocol (*RTP*), hence frames are fragmented at the application layer before being sent. RTP packets received by the client are then played out at their corresponding playout time. We configured the client with 100ms-playout delay: this is a tradeoff between the responsiveness of the video streaming and the buffering time required to prevent stalls.

Fig. 7 illustrates the network topology in our simulation scenario, which is composed by a regular hexagonal grid of seven gNBs, with inter-gNB distance of 500 m. 15 UEs are deployed randomly over the floorplan, and they dynamically connect to the gNB they receive the highest power from. Each UE runs the client side of the video-streaming application, whereas their server-side counterparts reside on an edge host connected to the 5G core network. Each client receives a different video trace, obtained by starting one of the three aforementioned dash-cam videos at different times. In order to simulate realistic load conditions, each gNB is configured to send 50 KB/s downlink traffic to 30 *background UEs*. We also simulate an additional tier of *background cells*, each serving 30 background UEs, with the aim of adding interference to the UEs attached to the seven central gNBs [55]. Each gNB is configured with 20 MHz bandwidth, resulting in 100 Resource Blocks (RBs). We run 24 independent replicas of a 120-second simulation, collecting time-tagged metrics from the 15 UEs in the seven central cells. The description of the produced metrics is reported in Table 2.

[8] `ffprobe -show_frames`: online documentation https://ffmpeg.org/ffprobe.html, accessed November 2022.

[9] https://bit.ly/3iN651q, https://bit.ly/35n9elO, https://bit.ly/3IT5g24, accessed November 2022.

Table 2
Description of the metrics included in the dataset.

Name	Level	Description	Missing values handling strategy
Context			
UE position	Application	(x, y, z) coordinates of the UE in the floorplan	Forward filling
UE speed	Application	Speed of the UE in $\frac{m}{s}$	Forward filling
QoS metrics			
avgServedBlocksDl	Network	Number of RBs occupied in downlink	Default value: -1
averageCqiDl	Network	CQI values reported in DL	Default value: 0
rcvdSinrDl	Network	SINR value measured at packet reception	Default value: -20
servingCell	Network	ID of the new serving cell after the handover	Forward filling
frameSize	Application	Size of the displayed frame (Byte)	Default value: 0
rtpPacketSize	Application	Size of the RTP packet (Byte)	Default value: 0
end2EndDelay	Application	Time between transmission and reception of an RTP packet	Default value: -1
interArrivalTimeRtp	Application	Interarrival time between two RTP packets	Default value: 0
rtpLoss	Application	RTP packets of frame lost	Default value: 0
QoE metrics			
framesDisplayed	Application	Frame percentage arrived at the time of its display	Default value: 0
playoutBufferLength	Application	Frame buffer size	Default value: 0
firstFrameElapsedTime	Application	3 values: (1) timestamp of the UE request, (2) timestamp of the sender ACK, (3) time between the request and the first frame displayed	NA

The resulting dataset consists of 5568 rows, each being a tuple with six fields:

run is the ID of the replica; *network_parameters* include the variables describing the simulation configuration (e.g., the scheduling algorithm); *module* is the network entity (e.g., ue[0]) that recorded the metric; *statistic* is the name of the recorded metric; *values* is a vector including the values recorded for the above metric; *timestamp* is a vector whose elements are the timestamps of the corresponding elements of the *values* vector.

5.4. Qoe prediction as a regression problem: preprocessing and design choices

We formulate the QoE forecasting problem as a regression problem as in [17]. In the following, we first recall the preprocessing steps designed to transform the original raw dataset into a regression dataset, suitable for the downstream adoption of traditional ML approaches. Then, we describe the design choices, detailing about the parameter configuration adopted for the generation of the TSK-FRBSs.

The preprocessing steps can be summarized as follows:

1. identification of the timeseries available at each UEs;
2. windowing on the timeseries (Fig. 8);
3. handling missing values;
4. statistics computation on each window;
5. feature selection.

In the first step, the timeseries available at each UE have been determined. The *avgServedBlocksDl* statistic (related to gNBs) can be retrieved by the UEs, based on the known value of the current *ServingCell*, through the services available in a MEC-enabled architecture. Similarly, since the position of each UE and gNB is known, the distance between an UE and its serving gNB can be computed (*UE_Distance_to_gNB*).

Fig. 8 shows the windowing operation (step 2). As an example, the first ten seconds of the timeseries from three metrics are shown, namely *UE_Distance_to_gNB*, *rcvdSinrDL* and *framesDisplayed* (QoE target metric).

At this point, differently from [17], missing values are handled by replacement with default values or by using the forward filling approach (step 3). The strategy used for each metric is dictated by the reason behind the absence of the measurement and is reported in Table 2.

Any record of the preprocessed dataset is obtained as follows: we compute statistics within a window W over historical data of each variable (step 4). Such statistics consist in mean, median, max, min, variance, standard deviation, kurtosis, skewness, Q1 and Q3: the number of actual samples are used for the estimates. The associated target value is the mean of the *frameDisplayed* variable over the time horizon of size H (one step ahead forecasting). The subsequent record is obtained by sliding the two windows with a step H. Each instance is thus represented in $\mathbb{R}^{132}$ (11 statistics evaluated over window of size W on 12 timeseries) and is associated with the target QoE (average value of *frameDisplayed* over window of size H). In this analysis we focus on timeseries metrics, therefore we do not include the values of *firstFrameElapsedTime*.

Finally, a feature selection based on decision tree [57] is performed to reduce the dimensionality of the problem. A preliminary experimental analysis, carried out by inducing a regression on the whole training set, allowed us to select the following 15 most relevant features: *framesDisplayed_Q3_W, framesDisplayed_mean_W, playoutBufferLength_mean_W, interArrivalTimeRtp_max_W, framesDisplayed_median_W, playoutBufferLength_counter_W, distanceGNB_variance_W, distance- GNB_stdev_W, interArrivalTimeRtp_counter_W, interArrivalTimeRtp_skew_W, framesDisplayed_kurtosis_W, end2endDelay_mean_W, rcvdSinrDl_Q3_W, end2endDelay_counter_W, distanceGNB_skew_W*. For this step, we resorted to the decision tree for regression available in scikit-learn[10].

For the purpose of TSK-FRBSs generation, the following design choices are performed. Each feature is clipped in the range $[0, 1]$ after robust scaling using quantiles $(0.025, 0.975)$; this is to ensure that all participants have the features defined in the same range, and that scaling does not depend on the 5% most extreme values of each distribution. Then, the number of fuzzy sets in which each input attribute is partitioned is set equal to three: this guarantees a high level of semantic interpretability [58] thanks to the adoption of just three linguistic labels (*Low*, *Medium* and *High*). Finally, we set the windows size $W = 3$ and $H = 1$.

6. Experimental analysis of Fed-XAI models for QoE forecasting

In this section, we first describe our experimental setup, in terms of learning settings and evaluation metrics, and then report and discuss the results of our experiments on FL of XAI models for QoE forecasting from a twofold perspective: model performances and interpretability aspects.

[10] https://scikit-learn.org/stable/, accessed November 2022.

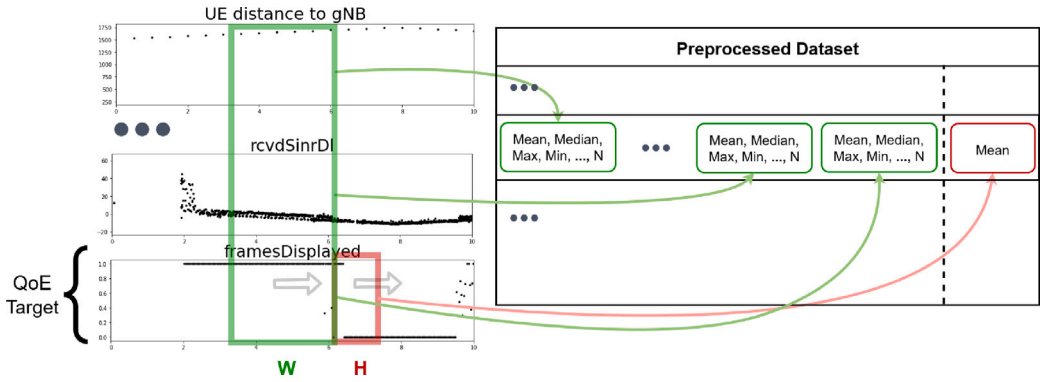


Fig. 8. Preprocessing steps: the QoE prediction task as a regression problem.
Source: Figure from [17].

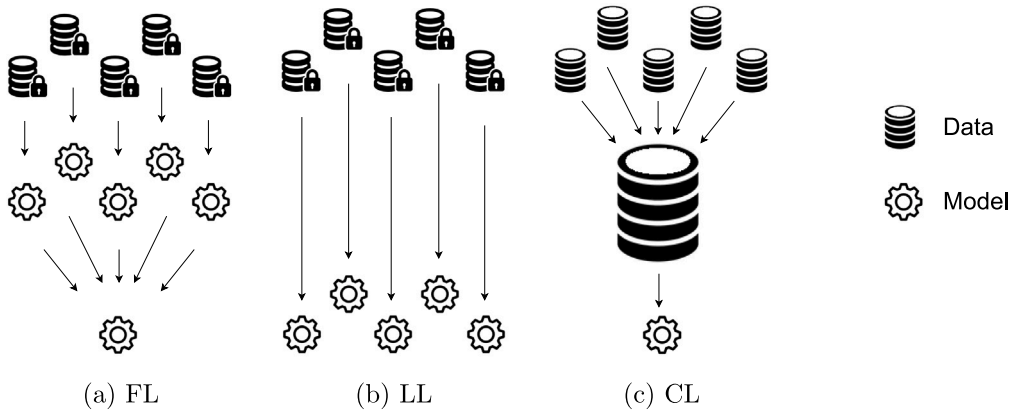


Fig. 9. Schematized representation of the three experimental learning settings: (a) federated learning, (b) local learning; (c) centralized learning.

6.1. Experimental setup

The QoS-QoE dataset is naturally spread over the fifteen UEs. To assess the generalization capability of our proposed ML models, we split the local dataset of each UE into training and test sets according to the same rationale described in [17]: the union of the first 20 runs out of the 24 independent runs are used as training set, whereas the remaining 4 runs are used as test set. Hereafter, we refer to the four test sets as Run 1, Run 2, Run 3, and Run 4. In addition to being consistent with the established experimental setup, this splitting ensures a quite high number of instances both in the training set (35487) and in the test set (7213).

We consider the following learning settings:

Federated Learning (FL) setting: The approach described in Section 3.2 for FL of TSK-FRBS is adopted. This setting entails a form of collaboration among UEs without any disclosure of private raw data.

Local Learning (LL) setting: Each client locally learns its own TSK-FRBS. This setting is privacy preserving, but entails no collaboration among UEs.

Centralized Learning (CL) setting: The union of the local training sets is used to train a global TSK-FRBS. Evidently, this setting represents the utmost form of collaborative training, but implies the violation of the users' privacy due to the collection of local raw data at a central location.

A schematized representation of the three settings is reported in Fig. 9. The goal of the experimental analysis is threefold: first, we evaluate the modeling capability of the TSK-FRBS learned in a federated fashion. Second, we compare the performance obtained in the FL setting with those in the LL setting to assess whether the FL paradigm brings any benefit to the participants. Third, we compare the performance obtained in the FL setting with those in the CL setting to quantify the performance degradation with respect to the (generally unfeasible) scenario in which all data can be gathered for training on a central server.

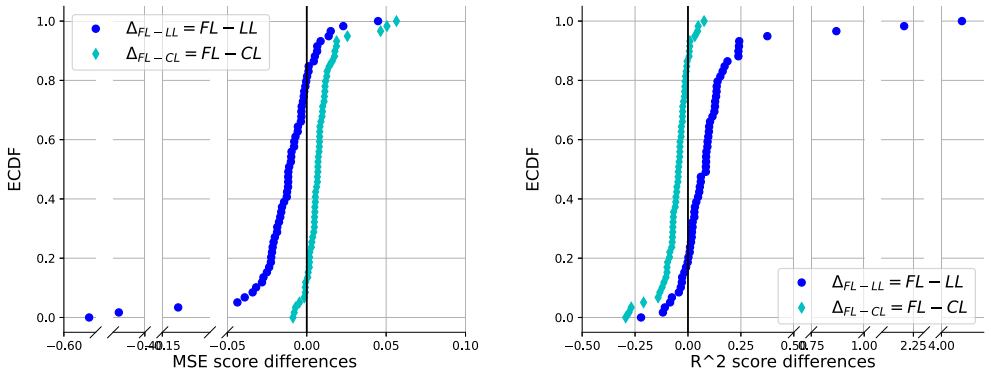
The quality of prediction of the TSK-FRBSs is evaluated through the *Mean Squared Error* (MSE) and the *coefficient of determination* (R^2), formally defined in the following Eqs. (4) and (5), respectively:

$$MSE = \frac{1}{N_{test}} \sum_{i=1}^{N_{test}} (y_i - \hat{y}_i)^2 \tag{4}$$

$$R^2 = 1 - \frac{\sum_{i=1}^{N_{test}} (y_i - \hat{y}_i)^2}{\sum_{i=1}^{N_{test}} (y_i - \bar{y})^2} \tag{5}$$

where N_{test} is the number of samples considered for the evaluation, y_i and $\hat{y}_i$ are the ground truth value and the predicted value associated with the ith instance of the test set, respectively, and $\bar{y}$ is the mean of ground truth values. When all the predictions match the true values, we obtain $MSE = 0$ and $R^2 = 1$. Thus, the goal is to minimize MSE and maximize R^2.

For the purpose of performance assessment, we evaluate the metrics as follows: regardless of the learning setting, we consider the actual partition of the dataset across UEs; specifically, for CL and FL the global model is evaluated on each run of the local test sets; for LL, each local model is tested on the dedicated, private, test set.



(a) MSE (b) R^2

Fig. 10. Empirical cumulative distribution function (ECDF) of the differences of MSE scores (a) and R^2 scores (b) between FL and LL (Δ_{FL-LL}, dark blue circles) and between FL and CL (Δ_{FL-LL}, light blue diamonds).

Table 3
Average values of MSE scores and R^2 scores on the training set and on the test set (highlighted in italic), obtained with the three learning settings.

	FL		LL		CL	
	Train	*Test*	Train	*Test*	Train	*Test*
MSE	0.052	*0.066*	0.030	*0.094*	0.045	*0.057*
R^2	0.614	*0.559*	0.799	*0.376*	0.692	*0.617*

6.2. Experimental results

Table 3 shows the average values of MSE and R^2 obtained with the three learning settings on the training set (20 runs) and on the test set (4 runs).

Results on the test set suggest that FL model outperforms, on average, the LL models. The indications given by the two metrics are in agreement: FL obtain lower values of MSE compared to LL (0.066 vs 0.094) and higher values of R^2 (0.559 vs 0.376). Interestingly, local models are particularly prone to overfitting: in fact, the results obtained on the training set from the LL models are the best ones, but the wide gap with respect to the results on test set is an indication of poor generalization capability.

In the rightmost column of Table 3 we also report the results obtained in the CL setting: evidently, the ability to train the model on the complete set of training data leads to the best average results on the test set, both in terms of MSE and R^2.

The average values, however, provide only a rough indication of the performances, but do not accurately reflect the actual situation experienced on each UE. To provide a fine grained picture of the performance of FL and LL models, we report the values of MSE and R^2 in Table 4 and 5, respectively, as computed on each run of the test set of each UE. For the sake of visualization, we omit to report the results for the CL setting, merely pointing out that it leads, in general, to better performance.

Also for the purposes of this analysis, we find that the two metrics give consistent indications. Although for some UEs and some runs, the locally learned model leads to better modeling of the regression problem, in most cases the FL model outperforms the LL one. In other words, UEs generally achieve a benefit in participating in the federation in terms of enhanced modeling capability, without any disclosure of private raw data.

The fine-grained information reported in Tables 4 and 5 is elaborated in a compact visualization shown in Fig. 10.

Fig. 10(a) shows the empirical cumulative distribution function (ECDF) of the difference between the MSE score of FL setting and the MSE score of LL setting (Δ_{FL-LL}, dark blue circles) and between the MSE score of FL setting and the MSE score of CL setting (Δ_{FL-CL}, light blue diamonds) for each of the 60 experiments (i.e., 4 runs for each of the 15 UEs). The plot can be interpreted as follows: a curve lies in the negative half-plane if and only if the MSE values of the FL model are lower (and therefore better) than those of the model with which it is compared. It can thus be noticed that the discrepancy of MSE values between the different settings is generally in the range of $[-0.05, +0.05]$. Furthermore, the dark blue curve crosses the y-axis in around 0.8, indicating that in about 80% of cases the FL model obtains better performance than its local counterpart on the corresponding test run. The light blue curve, conversely, lies in the positive half-plane starting from around the value of 0.15, testifying the better modeling capability of the CL model compared to the FL one in about 85% of cases. Fig. 10(b) shows the ECDF analysis related to the values of R^2: the considerations are similar to those reported for the MSE, taking into account that R^2 has to be maximized and therefore the problem is symmetrical with respect to the y-axis.

We performed the pairwise Wilcoxon signed-rank test [59] to assess possible statistical differences in performances (in terms of MSE and R^2) between the FL setting and the other ones: specifically, the *FL* setting is selected as the control one and is separately compared with *LL* and *CL*. For each setting, the distribution consists of 60 values of the metric measured on the test sets (4 runs for 15 UEs). Table 6 reports the results of the test.

$\mathbf{R}^+$ and $\mathbf{R}^-$ denote, respectively, the sum of ranks for the evaluations in which the federated model outperformed the other one, and the sum of ranks for the opposite outcome. The statistical hypothesis of equivalence can be rejected whenever the p-value is lower than the level of significance α. Results suggest that, with $\alpha = 0.05$, the *FL* model statistically outperforms the *LL* ones, regardless of the metric considered. On the other hand it is outperformed by the *CL* model.

It is worth analyzing the experimental results also from the perspective of the actual timeseries. Fig. 11 shows a QoE timeseries for an example test run of an example UE (Run-1, UE-04): specifically, the true values of QoE (*y_true*) are shown, along with the values predicted with the three learning settings (FL, LL, CL).

It can be noticed (Fig. 11(a)) that the gray curve (CL predictions) traces the black curve (*y_true*) more closely, whereas the red one (LL predictions) often exhibits incorrect predictions. In the magnified portion of the timeseries depicted in Fig. 11(b) we can appreciate further details: around second 41, a minimal increase in QoE is improperly predicted by the LL model, whereas FL and CL models predict low QoE. Moreover, at around second 47, it can be observed that all the learning settings correctly model the actual increase but with different delays: response time is shorter for CL and FL and longer for LL, confirming the general considerations about the performance of the different settings.

Table 4
MSE scores: fine-grained results on the test set for FL and LL models. Best values are highlighted in bold.

	Run 1		Run 2		Run 3		Run 4	
	FL	LL	FL	LL	FL	LL	FL	LL
UE-01	0.049	**0.047**	**0.087**	0.098	**0.061**	0.073	**0.070**	0.085
UE-02	**0.037**	0.042	**0.075**	0.093	**0.156**	0.160	**0.059**	0.067
UE-03	0.066	**0.051**	**0.054**	0.056	**0.027**	0.054	**0.100**	0.104
UE-04	**0.062**	0.083	**0.097**	0.118	**0.088**	0.105	0.072	**0.068**
UE-05	0.039	0.039	0.027	**0.020**	**0.051**	0.056	**0.073**	0.076
UE-06	**0.078**	0.112	**0.063**	0.102	0.068	**0.063**	0.100	**0.091**
UE-07	**0.043**	0.053	**0.046**	0.630	**0.060**	0.075	**0.042**	0.061
UE-08	**0.066**	0.076	0.086	**0.072**	0.138	**0.093**	0.060	**0.037**
UE-09	0.029	0.029	0.044	**0.043**	**0.080**	0.101	**0.057**	0.068
UE-10	**0.112**	0.124	**0.073**	0.089	**0.055**	0.080	**0.064**	0.096
UE-11	**0.073**	0.090	**0.058**	0.061	**0.065**	0.067	**0.131**	0.548
UE-12	**0.053**	0.075	**0.032**	0.055	**0.038**	0.046	**0.030**	0.031
UE-13	**0.038**	0.060	**0.104**	0.244	0.036	**0.029**	**0.068**	0.080
UE-14	**0.050**	0.062	**0.037**	0.047	**0.110**	0.134	**0.073**	0.084
UE-15	**0.056**	0.100	**0.048**	0.076	**0.076**	0.079	**0.061**	0.069

Table 5
R^2 scores: fine grained results on the test set for FL and LL models. Best values are highlighted in bold.

	Run 1		Run 2		Run 3		Run 4	
	FL	LL	FL	LL	FL	LL	FL	LL
UE-01	0.595	**0.604**	**0.318**	0.226	**0.684**	0.623	**0.612**	0.525
UE-02	**0.808**	0.778	**0.392**	0.239	**0.011**	−0.011	**0.372**	0.287
UE-03	0.633	**0.717**	**0.715**	0.706	**0.627**	0.252	**0.435**	0.417
UE-04	**0.631**	0.502	**0.420**	0.295	**0.362**	0.235	0.534	**0.563**
UE-05	**0.770**	0.768	0.550	**0.659**	**0.727**	0.696	**0.557**	0.542
UE-06	**0.450**	0.209	**0.614**	0.376	0.640	**0.667**	0.509	**0.552**
UE-07	**0.583**	0.490	**0.675**	−3.422	**0.642**	0.558	**0.577**	0.390
UE-08	**0.594**	0.533	0.525	**0.600**	0.318	**0.540**	0.691	**0.809**
UE-09	0.869	**0.870**	0.395	**0.412**	**0.536**	0.420	**0.743**	0.690
UE-10	**0.229**	0.144	**0.546**	0.446	**0.622**	0.449	**0.526**	0.288
UE-11	**0.546**	0.441	**0.652**	0.630	**0.653**	0.645	**0.305**	−1.901
UE-12	**0.675**	0.540	**0.802**	0.662	**0.735**	0.679	**−0.134**	−0.166
UE-13	**0.772**	0.636	**0.355**	−0.514	0.816	**0.850**	**0.417**	0.318
UE-14	**0.622**	0.526	**0.825**	0.777	**0.381**	0.248	**0.435**	0.349
UE-15	**0.690**	0.448	**0.724**	0.560	**0.580**	0.563	**0.662**	0.623

Table 6
Results of the Wilcoxon Signed-Rank test on the performance metrics measured on the test sets (MSE and R^2). The learning setting that statistically outperforms the other (with significance level $\alpha = 0.05$) is highlighted in italic in the comparison.

MSE				
Comparison	**R⁺**	**R⁻**	***p*-value**	**Hypothesis** ($\alpha = 0.05$)
FL vs LL	1563	267	0.0000	Rejected
FL vs *CL*	135.5	1694.5	0.0000	Rejected
R^2				
Comparison	**R⁺**	**R⁻**	***p*-value**	**Hypothesis** ($\alpha = 0.05$)
FL vs LL	1575	255	0.0000	Rejected
FL vs *CL*	144	1686	0.0000	Rejected

Table 7
Model complexity. Number of rules of the TSK-FRBSs (size of the rule base).

FL	LL	CL
997	289.1 ± 21.6	997

6.3. Interpretability analysis

The model complexity can be regarded as a proxy for assessing the global interpretability of an FRBS: less complex models (with fewer rules) are generally deemed more interpretable.

Table 7 reports the number of rules of the FL, LL and CL models. For the LL setting we report the mean and standard deviation of the number of rules of the local models generated by the fifteen UEs.

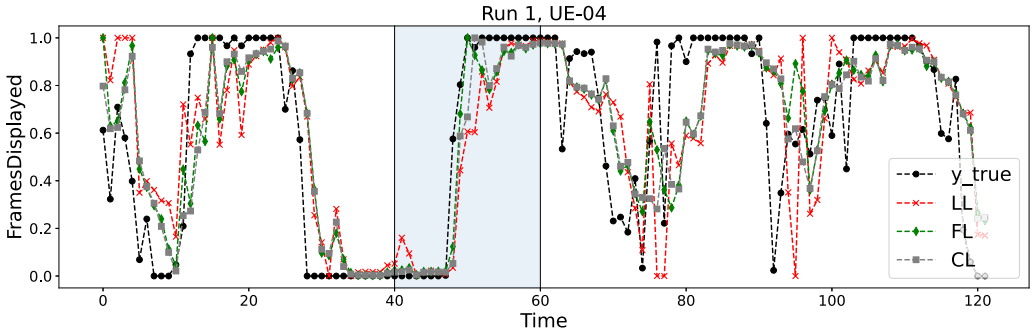
Expectedly, the number of rules of the FL approach coincides with that of CL approach: both approaches in fact use, directly or indirectly, all the training instances spread on the various UEs for model generation. FL (and CL) model, on the other hand, is more complex than the LL models by a factor of around 3.4, which can still be considered limited given the fact that the data originate from 15 UEs.

Interpretability is not only related to the structural properties of the model, but also to the inference strategy. In our TSK-FRBSs the predicted output depends on a single rule and an interpretation can be easily provided: the antecedent part of a rule isolates a region of the search space, whereas its consequent part describes a local linear model therein. Fig. 12 reports a graphic representation of two rules extracted from the FL TSK-FRBS, with an indication of the importance of each feature f in the linear model.

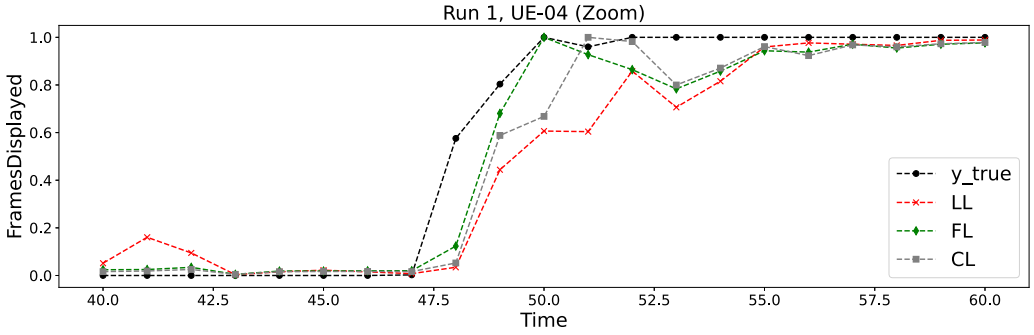
The color of each bar denotes the relevant fuzzy set, as specified in the antecedent, whereas its width indicates the actual value of the coefficient γ_f in the linear model. As an example, Fig. 12(a) suggests that when the values of all features is *Low* (except for *end2endDelay_counter_W*), the predicted output strongly decreases with the value of *framesDisplayed_mean_W*. Conversely, when the conditions of Fig. 12(b) hold, the output strongly increases with the value of *rcvdSinrDl_Q3_W*.

Finally, the inference process associated with two example test instances is explicated in Tables 8 and 9. The value y_{pred} is computed as follows:

$$y_{pred} = \gamma_0 + \sum_{f=1}^{F} \gamma_f \cdot x_f \tag{6}$$



(a) Complete timeseries. The shaded region is magnified in (b).



(b) Zoom of a portion of the timeseries.

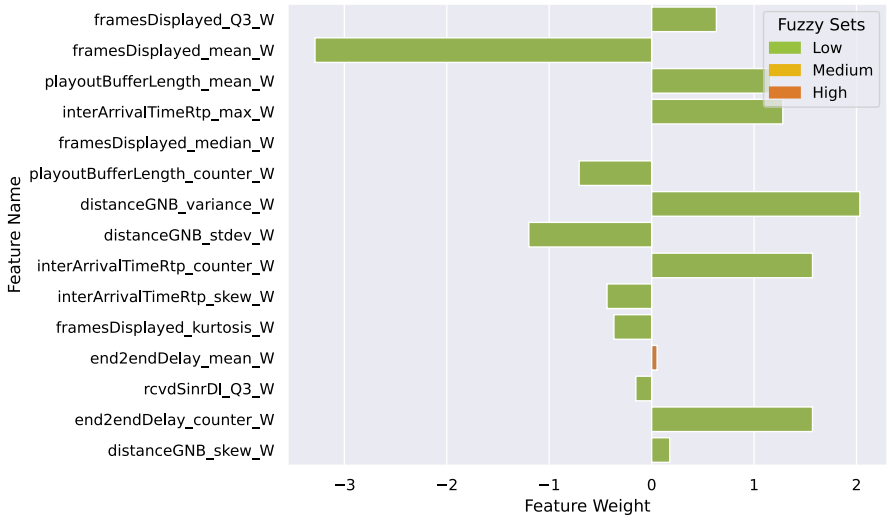
Fig. 11. Comparison of real and predicted values of QoE for an example test run (Run 1) from an example UE (UE-04). The predicted output is shown with different colors and markers for the three learning settings (FL, LL, CL).

Table 8
Inference process on Rule 1. $\gamma_0 = -0.084$.

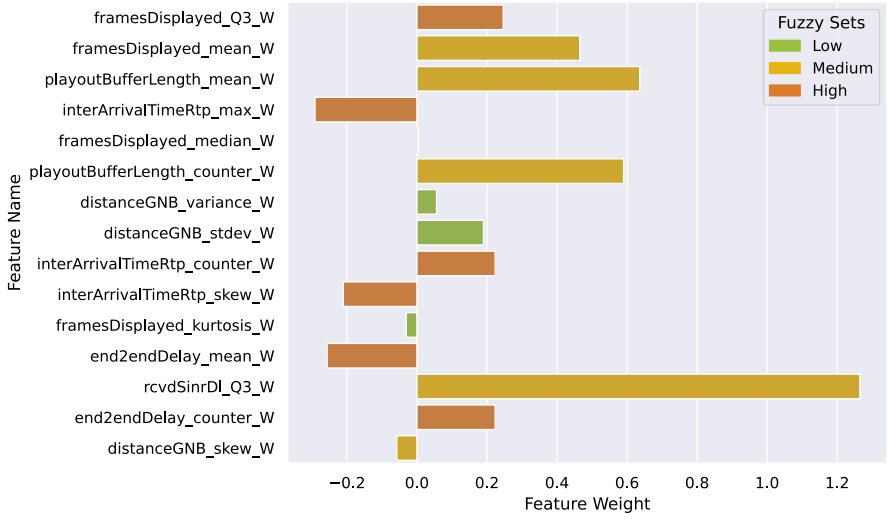
Feature f	A_f	$\mu_{A_f}(x_f)$	x_f	γ_f	$x_f \cdot \gamma_f$	y_{pred}	y_{true}
framesDisplayed_Q3_W	low	1.000	0.000	0.632	0.000		
framesDisplayed_mean_W	low	0.986	0.007	−3.287	−0.023		
playoutBufferLength_mean_W	low	0.584	0.208	1.165	0.242		
interArrivalTimeRtp_max_W	low	0.866	0.067	1.280	0.086		
framesDisplayed_median_W	low	1.000	0.000	0.000	0.000		
playoutBufferLength_counter_W	low	0.960	0.020	−0.708	−0.014		
distanceGNB_variance_W	low	0.904	0.048	2.033	0.097		
distanceGNB_stdev_W	low	0.571	0.215	−1.200	−0.257	0.009	0.019
interArrivalTimeRtp_counter_W	low	0.991	0.005	1.570	0.007		
interArrivalTimeRtp_skew_W	low	0.827	0.086	−0.436	−0.038		
framesDisplayed_kurtosis_W	low	0.612	0.194	−0.369	−0.071		
end2endDelay_mean_W	high	1.000	1.000	0.053	0.053		
rcvdSinrDl_Q3_W	low	0.751	0.124	−0.155	−0.019		
end2endDelay_counter_W	low	0.991	0.005	1.570	0.007		
distanceGNB_skew_W	low	0.738	0.131	0.177	0.023		

Table 9
Inference process on Rule 2. $\gamma_0 = -0.210$.

Feature f	A_f	$\mu_{A_f}(x_f)$	x_f	γ_f	$x_f \cdot \gamma_f$	y_{pred}	y_{true}
framesDisplayed_Q3_W	high	1.000	1.000	0.246	0.246		
framesDisplayed_mean_W	medium	0.956	0.478	0.465	0.222		
playoutBufferLength_mean_W	medium	0.785	0.608	0.636	0.387		
interArrivalTimeRtp_max_W	high	1.000	1.000	−0.291	−0.291		
framesDisplayed_median_W	low	1.000	0.000	0.003	0.000		
playoutBufferLength_counter_W	medium	0.993	0.497	0.590	0.293		
distanceGNB_variance_W	low	0.979	0.010	0.056	0.001		
distanceGNB_stdev_W	low	0.805	0.098	0.190	0.019	1.068	1.000
interArrivalTimeRtp_counter_W	high	1.000	1.000	0.223	0.223		
interArrivalTimeRtp_skew_W	high	1.000	1.000	−0.210	−0.210		
framesDisplayed_kurtosis_W	low	0.986	0.007	−0.031	0.000		
end2endDelay_mean_W	high	1.000	1.000	−0.257	−0.257		
rcvdSinrDl_Q3_W	medium	0.718	0.359	1.265	0.454		
end2endDelay_counter_W	high	1.000	1.000	0.223	0.223		
distanceGNB_skew_W	medium	0.908	0.546	−0.058	−0.031		



(a) Rule 1



(b) Rule 2.

Fig. 12. Feature importance: vector of coefficients of the linear model for two rules (a) Rule 1 (b) Rule 2. Color code indicates the antecedent part of the rule: for each input feature, the bar is green for fuzzy set *Low*, orange for fuzzy set *Medium* and red for fuzzy set *High*.

7. Evaluation of the network impact on the flaas framework

In this section we evaluate the learning process times in the FLaaS framework, under different network load conditions and different deployments of FLaaS functions. This evaluation, carried out via detailed end-to-end, system-level simulations, takes into account both the communication and the computation aspects.

We implemented the FLaaS framework described in Section 4 within Simu5G. FL entities have been developed as ETSI MEC applications running on a MEC host with the exception of the FLLM, that can instead be deployed on the UE according to its needs. This choice allows us to evaluate the framework when model exchanges occur via the 5G mobile network, under different network conditions. More in detail, we simulated a scenario composed of seven gNBs, deployed in a regular hexagonal grid as shown in Fig. 7, 500 meters apart from each other. A second tier of background cells, serving different background UEs, has been added in order to generate interference to the UEs connected to the seven central gNBs, which are the ones involved in the training. A MEC system, composed by only one MEC host, is connected to the network. First, each UE deploys its FLLM as either a local application or a MEC application. Then it instructs the FLLM about which FL service to take part in, i.e., QoE_forecasting. The FLGM handling the latter runs on the MEC host, and as soon as the required number of FLLMs is reached – i.e., the required number of FLLMs notified their interests in participating to the training process – the FLPCE starts the training phase. The configuration information of the FL process (possibly including the latest version of the global model) is sent to all the available FLLMs, which start the training with their local data after receiving it. After the training is completed, the new generated local models are sent back to the FLPCE. The latter aggregates the received local models only when *all* of them have been received. When the FLLM is deployed on the MEC host, the UE periodically sends the data representing the training dataset to it. In our simulations we assumed that the FLLM already has the data when it receives the global model to train.

The above use case has been evaluated in four different scenarios, consisting of the cross product of two deployments of the FLLM (on the UE, on the MEC), and two network loads (light, heavy).

The load of the network is quantified by the number of background UEs served by each cell, i.e., 10 for light load and 30 for heavy load.

Table 10
Main scenario parameters.

Network parameters	Value
Carrier frequency	2 GHz
Bandwidth	10 MHz
Numerology index	0
Downlink/Uplink duplexing	Frequency Division Duplexing
Path loss model	Urban Macro (UMa)[60]
Number of gNBs	7
Number of background gNBs	12
Number of UEs	[20, 40, 60, 80]
Number of background UEs per cell	[10, 30]
Background traffic Downlink	50 KB/s CBR
Background traffic Uplink	20 KB/s CBR
UE speed	uniform(13.8 mps, 41.7 mps)
FLaaS parameters	**Value**
FLLM deployment	[UE, MEC host]
Size of the global configuration	240 kB
Size of the local model	Uniform(70 kB, 80 kB)
Training duration of the local model on UE	Exponential(50 s) with 0.9 probability Exponential(85 s) with 0.1 probability
Training duration of the local model on the MEC host	Normal(15 s, 2 s)
Model aggregation time	500 ms * #local models received
Dataset chunk size and period	140B, 1 s

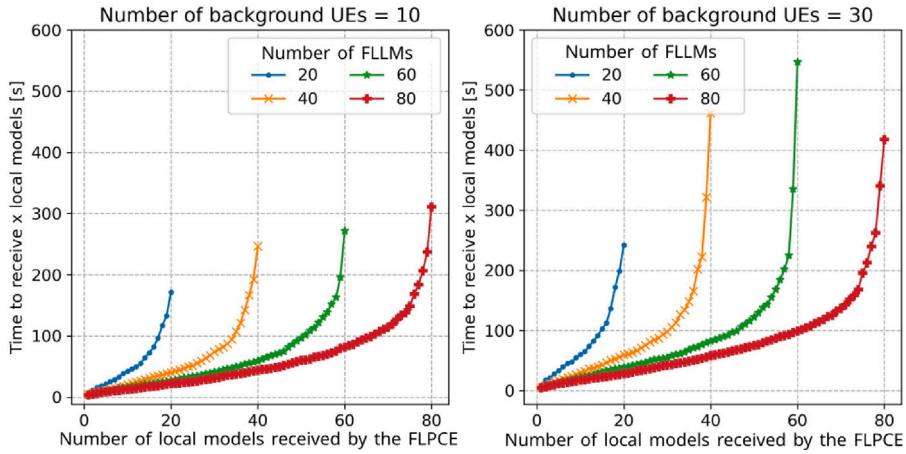


Fig. 13. Time to receive local models, with FLLMs on the UEs, light (left) and heavy (right) network load.

Each scenario has been run with an increasing number of UEs involved in the training. UEs can be heterogeneous devices, such as smartphones or laptops, hence the time to train the model may vary considerably. For this reason we generated the training times according to different distributions, depending on whether the FLLMs run on the UEs or on the MEC. For the latter we assumed shorter training times, since a MEC host has more computational resources. The dimension of the model, the duration of the training and other network parameters are reported in Table 10.

Our aim is to assess the time it takes for the FLPCE to retrieve all the local models from the FLLMs. Times are computed starting from when the FLPCE selects the FLLM for the training to when it receives the trained local model: hence they include uplink and downlink communications and training times, as well as the time needed to traverse all the protocol stack from/to the application layer (e.g., including the setup times for TCP connections).

Fig. 13 shows the time needed to retrieve a given number of local models, in the two load conditions described above. Since the training times are extracted from the same distributions in both load conditions, the difference between the two graphs is given by the communication overhead (in both the downlink and the uplink), i.e. the time it takes to send the models over the air. The charts allow us to observe how many local models the FLPCE should expect to receive at any given time. For instance, this information can be used to set meaningful deadlines on the training process, i.e., a maximum waiting time between the start of the training process and the start of the aggregation phase. For example, if the FLPCE performs the aggregation after 100 s and the number of FLLMs is 80, the FLPCE will have 65–70 models to aggregate when the network is lightly loaded, whereas it will aggregate around 60 models when the network is heavily loaded. Such information can be useful for tuning the parameters of the FL process.

The Empirical Cumulative Distribution Functions (ECDFs) of the time needed to exchange the models over the air in Figs. 14 and 15 confirm the above considerations and show how the load of the network affects such times. In particular, Fig. 14 shows that when the network is lightly loaded the time to send the global model in the downlink also depends on the number of the FLLMs, while with a heavily-loaded network the times do not depend anymore on the number of the FLLMs. This is because the data traffic generated by background UEs is prevalent with respect to the one needed to send global models to the FLLMs. Fig. 15 shows that the number of FLLMs does not significantly affect the time needed to send their trained local models, but even in this case the load of the network is relevant. In particular, the probability that a model is sent in less than five seconds is 0.95 when only 10 background UEs per cell are present, while in the other case the probability is below 0.8.

Fig. 16 shows the time needed to receive the local models as a function of the number of FLLMs, when these are deployed on the

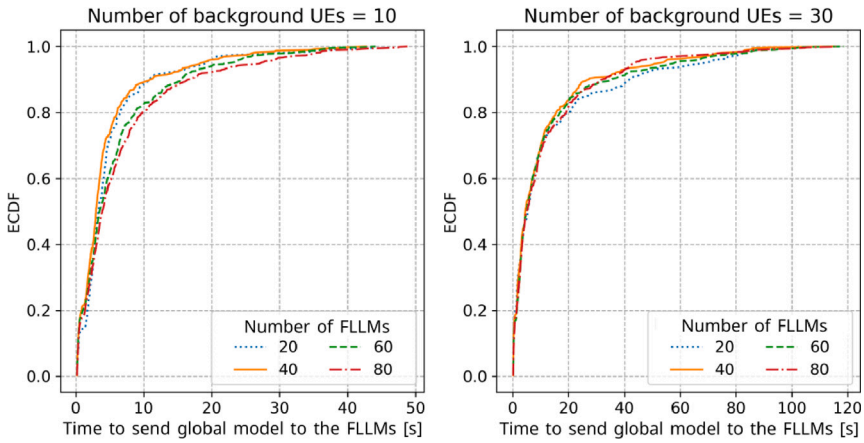


Fig. 14. ECDF of the time to send the global model from the FLPCE to the FLLM, light (left) and heavy (right) network load.

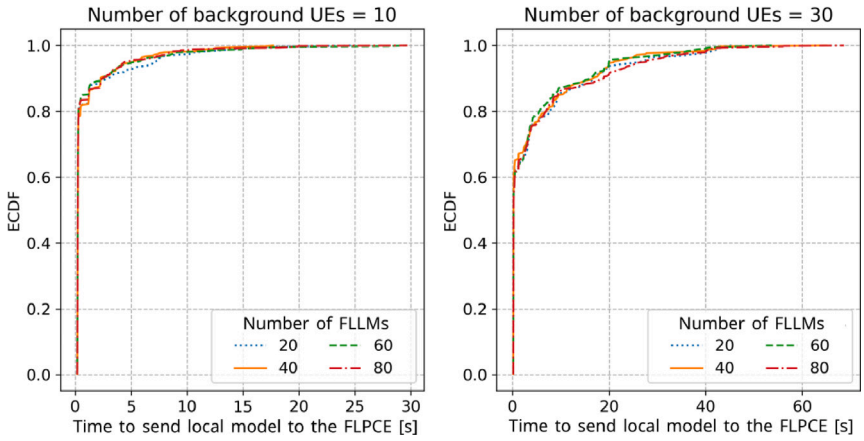


Fig. 15. ECDF of time to send the local model from the FLLM to the FLPCE, light (left) and heavy (right) network load.

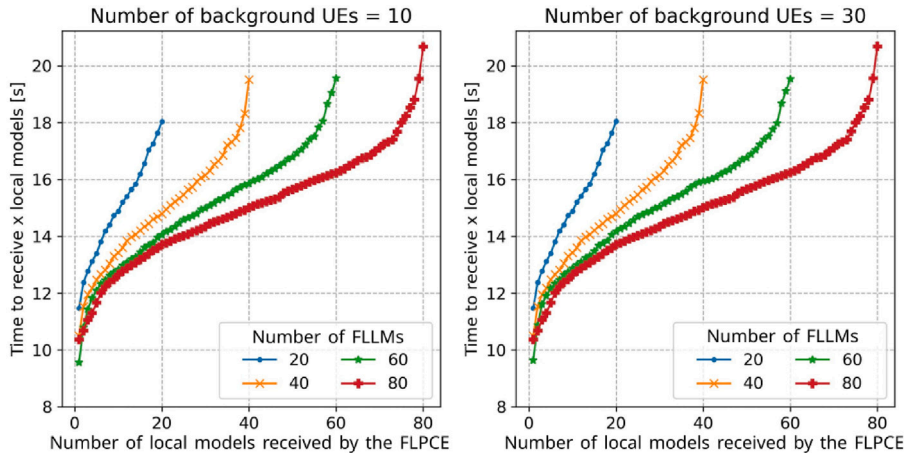


Fig. 16. Time to receive local models, with FLLMs on the MEC host, light (left) and heavy (right) network load.

MEC host. Comparing the charts with the ones in Fig. 13, we observe that the time needed to receive a given number of local models is greatly reduced, hence the benefit of deploying the FLLMs on the MEC is evident. Moreover, such time is independent of the mobile network load, since models are not exchanged over the air anymore in this case. The data used for the training is transmitted as a data stream over the air (in the uplink), but it consists of comparatively fewer data, which occupy few RBs and do not suffer from large delays — unlike models, which are bulky.

8. Conclusion

In this work, we have proposed a novel framework for Federated Learning (FL) of eXplainable AI (XAI) models. Our framework is envisioned to empower B5G/6G networks with AI services and stems from the urge for *trustworthiness* in intelligent systems. The FL paradigm, in fact, ensures raw data privacy preservation while enabling collaborative learning of AI models. Furthermore, in our case, such models are

explainable by-design so that any entitled stakeholder can monitor and oversee a transparent decision-making process.

Our proposal targets the synergic interface between trustworthy AI and next generation wireless networks and encompasses the following contributions. First, a novel FL-as-a-Service (FLaaS) framework is proposed and a thorough description of its components is provided along with their interaction and deployment. Second, an example FL service pertaining the automotive field is investigated: multiple instances of connected User Equipments consume a video stream and the goal of the AI model is to forecast the related Quality of Experience (QoE). Third, a comprehensive experimental analysis is carried out. Our approach for FL of XAI models is employed to address the problem of QoE forecasting and compared with (i) a local learning approach, which rules out collaboration among UEs, and (ii) a centralized learning approach, which entails centralization of raw data and privacy violation. Results are evaluated in terms of widely acknowledged regression metrics and show that our approach outperforms the local one, thus testifying for the benefit of the federation; in turn, it is outperformed by the centralized approach, which however is unfeasible when privacy preservation is a mandatory constraint. The adoption of inherently explainable models also allowed us to elaborate on the aspect of interpretability. Finally, the impact of communication and computation delays on the FL process times has been assessed through system-level simulations: it has been observed that leveraging edge-based environment for the deployments of FLaaS components can significantly reduce the overall time for the generation of a federated model.

The adoption of the proposed framework on other case studies and services represents, in our opinion, the most interesting future development of this work.

CRediT authorship contribution statement

José Luis Corcuera Bárcena: Software, Data curation, Investigation. **Pietro Ducange:** Conceptualization, Methodology, Writing – original draft, Writing – review & editing, Supervision. **Francesco Marcelloni:** Conceptualization, Writing – review & editing, Supervision. **Giovanni Nardini:** Conceptualization, Methodology, Investigation, Writing – original draft, Writing – review & editing, Supervision. **Alessandro Noferi:** Methodology, Software, Data curation, Investigation. **Alessandro Renda:** Conceptualization, Methodology, Investigation, Writing – original draft, Writing – review & editing. **Fabrizio Ruffini:** Software, Data curation, Investigation. **Alessio Schiavo:** Software, Data curation, Investigation. **Giovanni Stea:** Conceptualization, Methodology, Writing – review & editing, Supervision. **Antonio Virdis:** Writing – review & editing.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

Data will be made available on request.

Acknowledgments

This work was partially supported by: the EU Commission through the H2020 projects Hexa-X (Grant no. 101015956); by the PNRR - M4C2 - Investimento 1.3, Partenariato Esteso PE00000013 - “FAIR - Future Artificial Intelligence Research” - Spoke 1 “Human-centered AI”; by the PNRR “Tuscany Health Ecosystem” (THE) (Ecosistemi dell’Innovazione) - Spoke 6 - Precision Medicine & Personalized Healthcare (CUP I53C22000780001) under the NextGeneration EU programme; by the Italian Ministry of University and Research (MIUR), in the framework of the FoReLab project (Departments of Excellence); and by the Center for Logistic Systems of Livorno.

References

[1] F. Miltiadis, L. Vasiliki, M. Jafar, M. Mattia, E.U. Soykan, B. Tamas, R. Nandana, R. Nuwanthika, L. Le Magoarou, P. Pietro, et al., Pervasive artificial intelligence in next generation wireless: The Hexa-X project perspective, in: First International Workshop on Artificial Intelligence in beyond 5G and 6G Wireless Networks, (AI6G 2022), 2022.

[2] C-V2X Use Cases and Service Level Requirements Vol. I, Tech. rep., 5GAA, 2020.

[3] C-V2X Use Cases and Service Level Requirements Vol. II, Tech. rep., 5GAA, 2021.

[4] V. Vasilev, J. Leguay, S. Paris, L. Maggi, M. Debbah, Predicting QoE factors with machine learning, in: 2018 IEEE International Conference on Communications, (ICC), 2018, pp. 1–6, http://dx.doi.org/10.1109/ICC.2018.8422609.

[5] A. Renda, P. Ducange, G. Gallo, F. Marcelloni, XAI models for quality of experience prediction in wireless networks, in: 2021 IEEE International Conference on Fuzzy Systems, (FUZZ-IEEE), 2021, pp. 1–6, http://dx.doi.org/10.1109/FUZZ45933.2021.9494509.

[6] H.E. Dinaki, S. Shirmohammadi, E. Janulewicz, D. Côté, Forecasting video QoE with deep learning from multivariate time-series, IEEE Open J. Signal Process. 2 (2021) 512–521, http://dx.doi.org/10.1109/OJSP.2021.3099065.

[7] Q. Yang, Y. Liu, T. Chen, Y. Tong, Federated machine learning: Concept and applications, ACM Trans. Intell. Syst. Tech. 10 (2) (2019) 1–19.

[8] Ethics Guidelines for Trustworthy AI, Technical Report, European Commission. High Level Expert Group on AI., 2019, https://ec.europa.eu/digital-single-market/en/news/ethics-guidelines-trustworthy-ai.

[9] A. Barredo Arrieta, N. Díaz-Rodríguez, J. Del Ser, A. Bennetot, S. Tabik, A. Barbado, S. Garcia, S. Gil-Lopez, D. Molina, R. Benjamins, R. Chatila, F. Herrera, Explainable artificial intelligence (XAI): Concepts, taxonomies, opportunities and challenges toward responsible AI, Inf. Fusion 58 (2020) 82–115.

[10] B. McMahan, E. Moore, D. Ramage, S. Hampson, B.A.y. Arcas, Communication-efficient learning of deep networks from decentralized data, in: A. Singh, J. Zhu (Eds.), Proceedings of the 20th International Conference on Artificial Intelligence and Statistics, in: Proceedings of Machine Learning Research, vol. 54, PMLR, 2017, pp. 1273–1282.

[11] M. Polato, R. Esposito, M. Aldinucci, Boosting the federation: Cross-silo federated learning without gradient descent, in: 2022 International Joint Conference on Neural Networks, (IJCNN), 2022, pp. 1–10, http://dx.doi.org/10.1109/IJCNN55064.2022.9892284.

[12] X. Zhu, D. Wang, W. Pedrycz, Z. Li, Horizontal federated learning of Takagi–Sugeno fuzzy rule-based models, IEEE Trans. Fuzzy Syst. 30 (9) (2022) 3537–3547, http://dx.doi.org/10.1109/TFUZZ.2021.3118733.

[13] A. Wilbik, P. Grefen, Towards a federated fuzzy learning system, in: 2021 IEEE International Conference on Fuzzy Systems, (FUZZ-IEEE), 2021, pp. 1–6, http://dx.doi.org/10.1109/FUZZ45933.2021.9494392.

[14] J.L. Corcuera Bárcena, P. Ducange, A. Ercolani, F. Marcelloni, A. Renda, An approach to federated learning of explainable fuzzy regression models, in: IEEE WCCI 2022 (World Congress on Computational Intelligence), IEEE, 2022, (accepted).

[15] J.L.C. Bárcena, F. Marcelloni, A. Renda, A. Bechini, P. Ducange, A federated fuzzy c-means clustering algorithm., in: Proceedings of the 13th International Workshop on Fuzzy Logic and Applications 2021, (WILF), 2021.

[16] A. Renda, P. Ducange, F. Marcelloni, D. Sabella, M.C. Filippou, G. Nardini, G. Stea, A. Virdis, D. Micheli, D. Rapone, et al., Federated learning of explainable AI models in 6G systems: Towards secure and automated vehicle networking, Information 13 (8) (2022) 395.

[17] J.L.C. Bárcena, P. Ducange, F. Marcelloni, G. Nardini, A. Noferi, A. Renda, G. Stea, A. Virdis, Towards trustworthy AI for QoE prediction in B5G/6G networks, in: First International Workshop on Artificial Intelligence in beyond 5G and 6G Wireless Networks - AI6G2022, Vol. 3189, 2022, pp. 1–9, http://dx.doi.org/10.5281/zenodo.7024525, URL http://ceur-ws.org/Vol-3189/paper_07.pdf.

[18] M. Hoffmann, M. Uusitalo, M.-H. Hamon, B. Richerzhagen, G. D’Aria, A. Gati, D. Lopez (Eds.), Expanded 6G Vision, Use Cases and Societal Values, Tech. rep., Hexa-X EU Project deliverable D-1.2, 2021.

[19] R. Li, Z. Zhao, X. Zhou, G. Ding, Y. Chen, Z. Wang, H. Zhang, Intelligent 5G: When cellular networks meet artificial intelligence, IEEE Wirel. Commun. 24 (5) (2017) 175–183, http://dx.doi.org/10.1109/MWC.2017.1600304WC.

[20] S. Zhang, D. Zhu, Towards artificial intelligence enabled 6G: State of the art, challenges, and opportunities, Comput. Netw. 183 (2020) 107556, http://dx.doi.org/10.1016/j.comnet.2020.107556, URL https://www.sciencedirect.com/science/article/pii/S138912862031207X.

[21] H. Yang, A. Alphones, Z. Xiong, D. Niyato, J. Zhao, K. Wu, Artificial-intelligence-enabled intelligent 6G networks, IEEE Netw. 34 (6) (2020) 272–280, http://dx.doi.org/10.1109/MNET.011.2000195.

[22] S. Niknam, H.S. Dhillon, J.H. Reed, Federated learning for wireless communications: Motivation, opportunities, and challenges, IEEE Commun. Mag. 58 (6) (2020) 46–51, http://dx.doi.org/10.1109/MCOM.001.1900461.

[23] E. Bakopoulou, B. Tillman, A. Markopoulou, Fedpacket: A federated learning approach to mobile packet classification, IEEE Trans. Mob. Comput. 21 (10) (2022) 3609–3628, http://dx.doi.org/10.1109/TMC.2021.3058627.

[24] O. Haliloglu, E.U. Soykan, A. Alabbasi, Privacy preserving federated RSRP estimation for future mobile networks, in: 2021 IEEE Globecom Workshops, (GC Wkshps), 2021, pp. 1–6, http://dx.doi.org/10.1109/GCWkshps52748.2021.9682084.
[25] F. Malandrino, C.F. Chiasserini, Federated learning at the network edge: When not all nodes are created equal, IEEE Commun. Mag. 59 (7) (2021) 68–73, http://dx.doi.org/10.1109/MCOM.001.2001016.
[26] C. Battiloro, P.D. Lorenzo, M. Merluzzi, S. Barbarossa, Lyapunov-based optimization of edge resources for energy-efficient adaptive federated learning, IEEE Trans. Green Commun. Netw. (2022) 1, http://dx.doi.org/10.1109/TGCN.2022.3186879.
[27] M. Aledhari, R. Razzak, R.M. Parizi, F. Saeed, Federated learning: A survey on enabling technologies, protocols, and applications, IEEE Access 8 (2020) 140699–140725.
[28] W.Y.B. Lim, N.C. Luong, D.T. Hoang, Y. Jiao, Y.-C. Liang, Q. Yang, D. Niyato, C. Miao, Federated learning in mobile edge networks: A comprehensive survey, IEEE Commun. Surv. Tutor. 22 (3) (2020) 2031–2063, http://dx.doi.org/10.1109/COMST.2020.2986024.
[29] R. Shafin, L. Liu, V. Chandrasekhar, H. Chen, J. Reed, J.C. Zhang, Artificial intelligence-enabled cellular networks: A critical path to beyond-5G and 6G, IEEE Wirel. Commun. 27 (2) (2020) 212–217, http://dx.doi.org/10.1109/MWC.001.1900323.
[30] W. Guo, Explainable artificial intelligence for 6G: Improving trust between human and machine, IEEE Commun. Mag. 58 (6) (2020) 39–45, http://dx.doi.org/10.1109/MCOM.001.2000050.
[31] Y. Liu, K. Liu, M. Li, Passive diagnosis for wireless sensor networks, IEEE/ACM Trans. Netw. 18 (4) (2010) 1132–1144, http://dx.doi.org/10.1109/TNET.2009.2037497.
[32] A. Tsakmalis, S. Chatzinotas, B. Ottersten, Constrained Bayesian active learning of interference channels in cognitive radio networks, IEEE J. Sel. Top. Sign. Proces. 12 (1) (2018) 6–19, http://dx.doi.org/10.1109/JSTSP.2017.2785826.
[33] A. Hameed, R. Dai, B. Balas, A decision-tree-based perceptual video quality prediction model and its application in FEC for wireless multimedia communications, IEEE Trans. Multimed. 18 (4) (2016) 764–774, http://dx.doi.org/10.1109/TMM.2016.2525862.
[34] G. Dimopoulos, I. Leontiadis, P. Barlet-Ros, K. Papagiannaki, Measuring video QoE from encrypted traffic, in: Proc. of the 2016 Internet Measurement Conf., 2016, pp. 513–526.
[35] Y.-T. Lin, E.M.R. Oliveira, S.B. Jemaa, S.E. Elayoubi, Machine learning for predicting QoE of video streaming in mobile networks, in: 2017 IEEE Int'L Conf. Communications, (ICC), IEEE, 2017, pp. 1–6.
[36] J.C. Bárcena, P. Ducange, F. Marcelloni, A. Renda, F. Ruffini, Hoeffding regression trees for forecasting quality of experience in B5G/6G networks, in: First Workshop on Online Learning from Uncertain Data Streams, (OLUD 2022), 2022, http://dx.doi.org/10.5281/zenodo.7024541.
[37] T. Takagi, M. Sugeno, Fuzzy identification of systems and its applications to modeling and control, IEEE Trans. Syst. Man Cybern. (1) (1985) 116–132.
[38] D. Kukolj, Design of adaptive Takagi–Sugeno–Kang fuzzy models, Appl. Soft Comput. 2 (2) (2002) 89–103.
[39] O. Cord, et al., Genetic Fuzzy Systems: Evolutionary Tuning and Learning of Fuzzy Knowledge Bases, Vol. 19, World Scientific, 2001.
[40] Y. Cui, D. Wu, J. Huang, Optimize TSK fuzzy systems for classification problems: Minibatch gradient descent with uniform regularization and batch normalization, IEEE Trans. Fuzzy Syst. 28 (12) (2020) 3065–3075.
[41] D. Wu, Y. Yuan, J. Huang, Y. Tan, Optimize TSK fuzzy systems for regression problems: Minibatch gradient descent with regularization, DropRule, and AdaBound (MBGD-RDA), IEEE Trans. Fuzzy Syst. 28 (5) (2020) 1003–1015, http://dx.doi.org/10.1109/TFUZZ.2019.2958559.
[42] Hexa-X project, 2022, Accessed: October 2022, https://hexa-x.eu/.
[43] G. Nardini, D. Sabella, G. Stea, P. Thakkar, A. Virdis, Simu5G–An OMNeT++ library for end-to-end performance evaluation of 5G networks, IEEE Access 8 (2020) 181176–181191, http://dx.doi.org/10.1109/ACCESS.2020.3028550.
[44] ETSI, Multi-access edge computing (MEC); framework and reference architecture, 2022, etsi.org/deliver/etsi_gs/MEC/001_099/003/03.01.01_60/gs_MEC003v030101p.pdf.
[45] G. Akman, P. Ginzboorg, V. Niemi, Privacy-aware access protocols for MEC applications in 5G, Network 2 (2) (2022) 203–224.
[46] B. Ali, M.A. Gregory, S. Li, Multi-access edge computing architecture, data security and privacy: A review, IEEE Access 9 (2021) 18706–18721.
[47] G. Kakkavas, K.N. Nyarko, C. Lahoud, D. Kühnert, P. Küffner, M. Gabriel, S. Ehsanfar, M. Diamanti, V. Karyotis, K. Möß ner, S. Papavassiliou, Teleoperated support for remote driving over 5G mobile communications, in: 2022 IEEE International Mediterranean Conference on Communications and Networking, (MeditCom), 2022, pp. 280–285, http://dx.doi.org/10.1109/MeditCom55741.2022.9928745.
[48] G. Kakkavas, M. Diamanti, K. Nseboah Nyarko, M. Gabriel, V. Karyotis, K. Möß ner, S. Papavassiliou, Realistic field trial evaluation of a tele-operated support service for remote driving over 5G, in: 2022 IEEE Conference on Standards for Communications and Networking, (CSCN), 2022, pp. 58–63, http://dx.doi.org/10.1109/CSCN57023.2022.10051034.
[49] H. Schippers, C. Schüler, B. Sliwa, C. Wietfeld, System modeling and performance evaluation of predictive QoS for future tele-operated driving, in: 2022 IEEE International Systems Conference, (SysCon), 2022, pp. 1–8, http://dx.doi.org/10.1109/SysCon53536.2022.9773810.
[50] R. Lopes, F. Rocha, S. Sargento, M. Luís, R. Leitão, E. Marques, B. Antunes, A multi-layer probing approach for video over 5G in vehicular scenarios, Veh. Commun. 38 (2022) 100534.
[51] J. Qiao, Y. He, X.S. Shen, Improving video streaming quality in 5G enabled vehicular networks, IEEE Wirel. Commun. 25 (2) (2018) 133–139.
[52] M. Uitto, A. Heikkinen, Evaluating 5G uplink performance in low latency video streaming, in: 2022 Joint European Conference on Networks and Communications & 6G Summit, (EuCNC/6G Summit), IEEE, 2022, pp. 393–398.
[53] OMNeT++ simulation framework website, 2022, Accessed: October 2022, http://omnetpp.org.
[54] A. Noferi, G. Nardini, G. Stea, A. Virdis, Deployment and configuration of MEC apps with Simu5G, CoRR (2021) arXiv:2109.12048, URL https://arxiv.org/abs/2109.12048.
[55] G. Nardini, G. Stea, A. Virdis, Scalable real-time emulation of 5G networks with Simu5G, IEEE Access 9 (2021) 148504–148520.
[56] INET library website, 2022, Accessed: October 2022, https://inet.omnetpp.org.
[57] K. Grabczewski, N. Jankowski, Feature selection with decision tree criterion, in: Fifth International Conference on Hybrid Intelligent Systems, (HIS'05), 2005, p. 6, http://dx.doi.org/10.1109/ICHIS.2005.43.
[58] M.J. Gacto, R. Alcalá, F. Herrera, Interpretability of linguistic fuzzy rule-based systems: An overview of interpretability measures, Inform. Sci. 181 (20) (2011) 4340–4360.
[59] F. Wilcoxon, Individual comparisons by ranking methods, in: Breakthroughs in Statistics, Springer, 1992, pp. 196–202.
[60] 3GPP TR 38.901 v16.1.0, "Study on Channel Model for Frequencies from 0.5 to 100 GHz", Tech. rep., 2020, January 2020.