6GSNS

Co-funded by the European Union

SEamless integratioN of efficient 6G WirelesS tEchnologies for Communication and Sensing

D4.3 AI/ML techniques for enhanced network performance

May 2026

6G-SENSES project has received funding from the Smart Networks and Services Joint Undertaking (SNS JU) under the European Union's Horizon Europe research and innovation programme under Grant Agreement 101139282

Project Start Date: 2024-01-01 **Duration**: 30 months

Call: HORIZON-JU-SNS-2023 **Date of delivery**: 2026-05-13

Topic: HORIZON-JU-SNS-2023-STREAM-B-01-02 **Version**: 1.0

Type: Report (R)

Grant Agreement Number:	101139282
Project Name:	SEamless integratioN of efficient 6G WirelesS tEchnologies for Communication and Sensing
Project Acronym:	6G-SENSES
Document Number:	D4.3
Document Title:	AI/ML techniques for enhanced network performance
Version:	1.0
Delivery Date:	2026-03-31 (**2026-05-13**)
Responsible:	Institute of Accelerating Systems and Applications (IASA)
Editor(s):	Pavlos Doanis (IASA)
Authors:	Markos Anastasopoulos, Anna Tzanakaki, Pavlos Doanis (IASA), Ramón Agüero, Luis Diez (UC), Novella Bartolini, Federico Trombetti, Salvatore Pontarelli (UNIROMA1), (BR), Germán Castellanos (ACC), Valerio Frascolla, Jessica Sanson (INT), Muhammad Saadi, Shahid Mumtaz (NTU).
Keywords:	Wireless Access Technologies (WATs), Integrated Sensing and Communication (ISAC), Intelligent Plane, Multi-access Edge Computing, Programmable Platforms, Open RAN (O-RAN), Reconfigurable Intelligent Surfaces (RIS).
Status:	Final
Dissemination Level	Public (PU)
Project URL:	https://www.6g-senses.eu

Revision History

Rev. N	Description	Author	Date
0.0	Draft Table of Contents (ToC)	Pavlos Doanis (IASA)	2026-01-21
0.1	First round of partner contributions	All partners	2026-03-19
0.2	Second round of partner contributions and initial editing	Contributions: All partners, Initial Editing: Pavlos Doanis (IASA)	2026-04-07
0.3	Third round of partner contributions and editing	Contributions: All partners, Editing: Pavlos Doanis (IASA)	2026-04-16
0.4	First revision of the document	Ramón Agüero, Luis Diez (UC)	2026-04-28
0.5	Second revision of the document	Anna Tzanakaki (IASA)	2026-05-04
0.6	Third revision of the document	Navid Nikaein (BR), Valerio Frascolla (INT)	2026-05-11
1.0	Last revision and submission of the document	Jesús Gutiérrez (IHP)	2025-05-13

Table of Contents

List of Figures

List of Tables

Executive Summary

This deliverable presents the work carried out in Task 4.2 of the 6G-SENSES project, focusing on the design and evaluation of Artificial Intelligence (AI)/Machine Learning (ML)-based algorithms for enhanced network performance in 6G systems. In increasingly complex and dynamic environments, driven by technologies such as Integrated Sensing and Communication (ISAC), Reconfigurable Intelligent Surfaces (RIS), and Multi-access Edge Computing (MEC), AI/ML techniques provide the means to enable adaptive, data-driven optimisation of network operation. Leveraging the 6G-SENSES architecture defined in deliverables **D2.2** [1] and **D2.3** [2], which enables the integration of intelligence across network domains, the proposed solutions exploit rich sensing and network data to support intelligent decision-making.

The deliverable introduces a set of algorithms addressing key challenges at different levels of the network. At the *resource management level* in the Radio Access Network (RAN) and Edge domains, it presents methods for efficient control and management of radio and edge resources, including Medium Access Control (MAC) scheduling, RIS configuration, and MEC caching, exploiting sensing-aware information such as user location and mobility. At a *higher level*, involving end-to-end (E2E), cross-domain, and application-level functionalities, it investigates AI-driven approaches for E2E performance prediction within the RAN, E2E slice orchestration for ISAC services across multiple network domains, intent-driven network automation, adaptive coordination of centralized sensing and Federated Learning (FL)-based distributed sensing for trajectory prediction and resource-efficient ISAC operation, as well as gesture recognition.

A key contribution of this document is the comprehensive evaluation of the proposed algorithms, considering not only system-level performance metrics related to Quality of Service (QoS), resource utilisation, and energy efficiency, but also algorithmic aspects including convergence behavior and sample efficiency. In addition, the deliverable documents the datasets used for training and validation.

Overall, this deliverable demonstrates the potential of AI/ML techniques to enhance network performance and enable intelligent, adaptive operation in ISAC-enabled 5G/6G environments, providing practical algorithmic solutions for the considered sensing-aware use case and insights for their effective deployment.

1 Introduction

The evolution towards 6G networks will require the support of heterogeneous services with widely diverse and stringent performance requirements. This can be achieved in a scalable and sustainable manner only through efficient utilisation of network resources, minimising energy consumption. Therefore, appropriate control, management, and orchestration algorithms are required to simultaneously optimise service performance and overall network efficiency. The complexity of these already challenging problems is further increased by the integration of advanced technologies such as Integrated Sensing and Communication (ISAC), Reconfigurable Intelligent Surfaces (RISs), and Multi-access Edge Computing (MEC). Considering also the dynamic nature of traffic demands, traditional model-based optimisation approaches are often not enough to cope with the large-scale, dynamic, and highly uncertain nature of such environments.

To address these challenges, Task 4.2 of the 6G-SENSES project investigates the adoption of Artificial Intelligence (AI)/Machine Learning (ML) techniques as key enablers for enhanced network performance. AI/ML provides the capability to extract patterns from complex system-level data, including network and user-related features (e.g. user mobility), and to learn adaptive control policies that optimize system behavior over time. By reducing computational complexity and enabling data-driven decision making, these techniques can support efficient and scalable network operation.

A critical aspect of this approach is the integration of intelligence within the network architecture. In this context, the Open-Radio Access Network (O-RAN) paradigm and its RAN Intelligent Controller (RIC) are considered fundamental building blocks for the deployment of AI/ML functionalities. The 6G-SENSES architecture, as defined in deliverables **D2.2** [1] and **D2.3** [2], extends the scope of control beyond traditional RAN components to incorporate sensing data from multiple Wireless Access Technologies (WATs), as well as external elements such as RIS. In addition, it enables the collection and processing of multi-domain information, spanning RAN, edge, and core network (CN) segments. Feeding this rich set of perceptive and contextual data into AI/ML agents, enhances decision-making capabilities and supports management and control across heterogeneous technologies and network domains.

Within this context, Task 4.2 has focused on the development of algorithmic solutions supporting functionalities that span both resource-level optimisation and end-to-end (E2E) intelligence. At the resource management level in the RAN and Edge domains, AI/ML and model-driven approaches are applied to the control and management of radio and storage resources, including tasks such as Medium Access Control (MAC) scheduling, RIS configuration, and MEC caching. These mechanisms exploit sensing information enabled by ISAC (e.g., user location and mobility) to enhance decision-making. At a higher level of abstraction, adopting an E2E perspective, AI/ML techniques enable predictive modeling and optimisation of network behavior, including E2E performance prediction within the RAN, E2E slice orchestration for ISAC services across multiple network domains, intent-driven network automation, as well as an adaptive framework that dynamically switches between centralised and Federated Learning (FL)-based distributed sensing for trajectory prediction and resource-efficient ISAC operation. In addition, sensing-enabled applications, such as gesture recognition, demonstrate how wireless signals can be leveraged to extract high-level contextual information from the environment.

This document reports the algorithms developed in Task 4.2 and evaluates them both in terms of system performance – e.g., Quality of Service (QoS) and energy efficiency – and algorithmic efficiency (e.g. sample efficiency and convergence behavior). While architectural integration and initial feasibility analysis of some algorithms have been reported in previous project deliverables (**D2.2** [1], **D2.3** [2], **D4.1** [3], **D4.2** [4]), this report extends them along several dimensions, including the introduction of new algorithms, algorithmic

improvements, evaluation of algorithmic efficiency, more comprehensive system-level performance analysis, and documentation of datasets used for AI/ML model training.

Organisation of the document

This document comprises six chapters. Following the Executive Summary and Introduction sections:

- Chapter 2 provides an overview of the 6G-SENSES Intelligent Plane. Specifically, it describes how the 6G-SENSES architecture enables AI-driven automation, outlining the types of sensing information available to AI/ML algorithms, the architectural enablers supporting these capabilities, and the different deployment alternatives for AI/ML models. These options are discussed with respect to key factors such as the location of training and inference, the timescale of decision-making, and the availability of data across network domains.
- Chapter 3 documents several Datasets used for the training and evaluation of AI/ML models developed in Task 4.2 and presented in chapters 4 and 5.
- Chapters 4 and 5 describe the developed AI/ML algorithms. First, Chapter 4 introduces the proposed algorithms for resource management-level optimisation in the RAN and Edge domains. These include both learning-based and rule-based approaches targeting the management and control of radio and storage resources, such as MAC scheduling, RIS configuration, and MEC caching. Next, Chapter 5 presents AI-driven approaches for E2E network intelligence and sensing. The chapter covers solutions for end-to-end performance prediction, end-to-end slice orchestration for ISAC, intent-driven RAN slicing, adaptive sensing and trajectory prediction through coordinated centralised and FL-based processing, as well as sensing-driven intelligence in the application-level for gesture recognition.
- Finally, Chapter 6 summarises the document.

2 6G-SENSES Intelligent Plane

This chapter provides an overview of the 6G-SENSES Intelligent Plane and its role in enabling AI-driven automation in next-generation networks. It describes how the 6G-SENSES architecture, aligned with 3GPP and Open-Radio Access Network (O-RAN) principles, integrates communication, sensing, and computing functionalities to support data-driven and adaptive network operation.

Section 2.1 first provides an overview of the 6G-SENSES Intelligent Plane, based on the architecture defined in deliverables **D2.2** [1] and **D2.3** [2], highlighting the key architectural components and enablers that support AI/ML-driven network operation. Then, Section 2.2 provides more details on different types of sensing data available and the mechanisms for their efficient provision to AI/ML agents.

2.1 Overview

Future 6G networks will support advanced applications with diverse requirements, integrating communications, sensing, and computing capabilities enabled by the ISAC and MEC paradigms. By leveraging radio signals not only for data transmission but also for environment perception, networks will be able to support advanced sensing-driven services, such as localisation, tracking, activity recognition, eXtended Reality (XR) and Digital Twins (DTs). This convergence introduces new opportunities but also increases complexity, as sensing services generate additional data that must be transported, processed, and stored, increasing the load on both network and edge communication and computing resources. As a result, network operation must jointly account for communication performance and sensing requirements. In such dynamic and information-rich environments, AI-driven automation becomes a fundamental enabler, allowing the network to continuously exploit sensing information, adapt to changing conditions, and optimize resource allocation across multiple domains.

The 6G-SENSES architecture introduces a unified framework, aligned with 3GPP and O-RAN principles, that integrates multi-technology RAN systems, including both 3GPP and non-3GPP technologies, to jointly support communication and sensing services. The baseline architecture, illustrated in Figure 2-1 and described in detail in **D2.2** [1], combines Sub-6 GHz, Wi-Fi, and millimetre wave (mmWave) (non-3GPP) technologies with 5G NR (3GPP) within an ISAC framework. In this context, 3GPP provides the foundation for standardised network functions and interfaces, while O-RAN enables openness, programmability, and AI-native control. The system follows a multi-layer, disaggregated design comprising virtualised RAN functions – Distributed Units (DUs)/Centralised Units (CUs) – a cloud–edge infrastructure, a core network (CN), and a Software Defined Networking (SDN)-controlled multi-technology transport network supporting fronthaul (FH), midhaul (MH), and backhaul (BH) connectivity. These components are controlled by hierarchical control entities, namely Near-Real Time RIC (Near-RT RIC) and Non-RT RIC/Service Management and Orchestration (SMO), with control logic implemented through xApps and rApps, altogether enabling intelligence across multiple time scales.

Non-3GPP sensing data plays a key role in enriching the perception of the network by incorporating information from heterogeneous sources, such as Wi-Fi-based sensing and other external sensors. In the 6G-SENSES architecture, Wi-Fi networks can operate as sensing entities (e.g., monostatic or multistatic radars), with their sensing data securely exposed to the RAN through extensions of the O-RAN interfaces. This is enabled through enhancements of the Non-3GPP InterWorking Function (N3IWF), allowing such systems to authenticate and deliver sensing information to the RIC with low latency. In addition, external sensing data from third-party platforms or domain-specific systems can be ingested at the SMO level through a gateway and processed within the Non-RT RIC, where it is combined with historical and configuration data to derive higher-level context, such as environment states or predictive indicators.

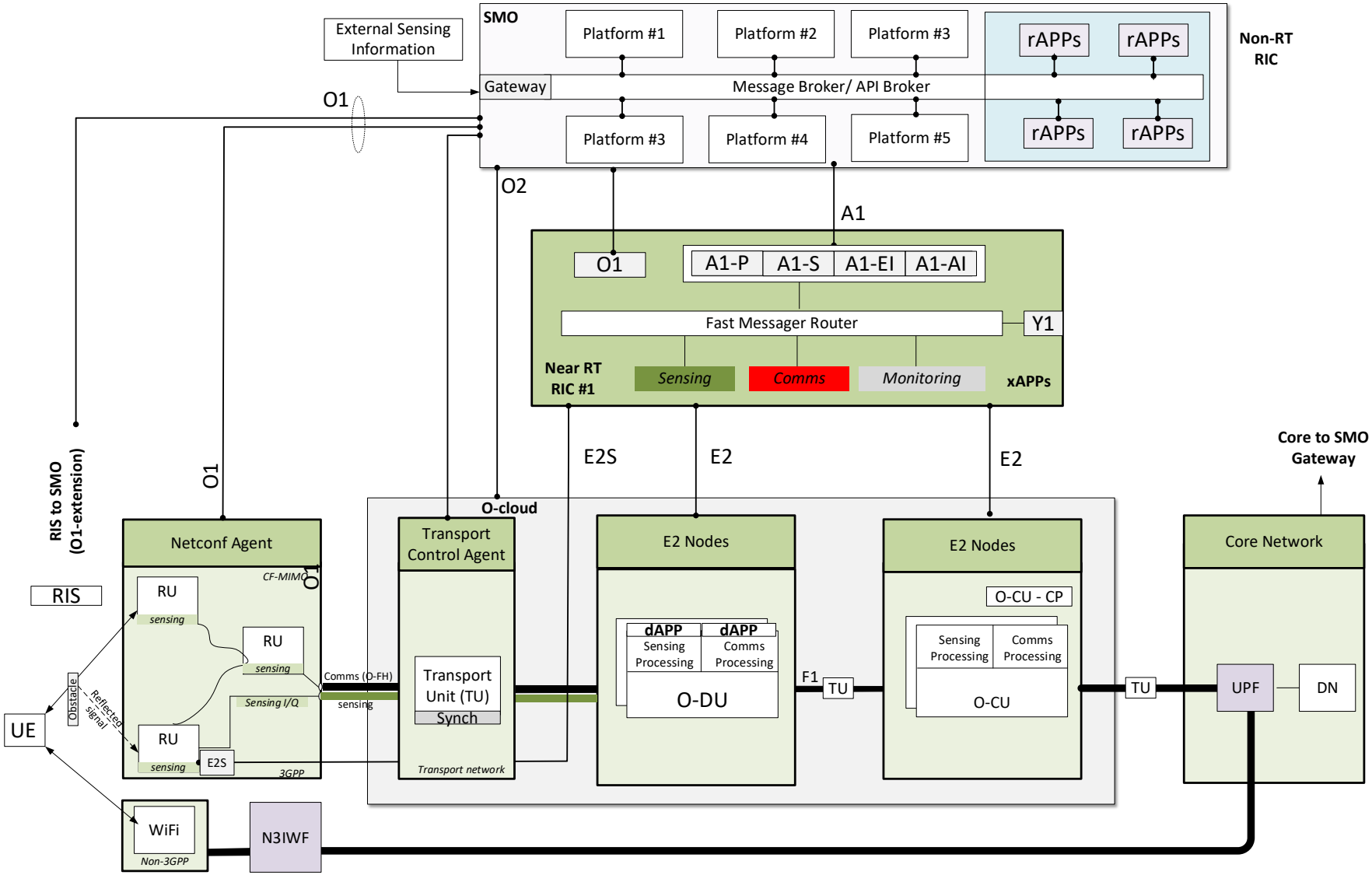


Figure 2-1 Generic 6G-SENSES architecture [2].

This abstracted sensing information is then forwarded to the Near-RT RIC, where it is consumed by xApps, complementing real-time measurements. Through this pipeline, non-3GPP sensing data is securely integrated, processed across different time scales, and made available to AI-driven control functions, enabling more informed and context-aware network optimisation. The capability to acquire sensing data from diverse external sensor types and deliver fused results securely to third parties is further described in deliverable **D2.3** [2], extending the architecture to support additional integration options for such sensors, including indirect connectivity via UEs that aggregate or relay data, as well as via Wi-Fi access points forming part of a non-3GPP access network.

3GPP-based sensing exploits the communication signals of the cellular network to enable environment perception based on distributed passive radar principles. In this approach, advanced base stations generate communication signals whose reflections from surrounding objects produce In-Phase and Quadrature (I/Q) echo streams, which are collected and forwarded to the O-RAN DUs for processing and compression. These sensing measurements are then delivered to the Near-RT RIC through the E2 interface, where specialised sensing xApps process and combine them with other data sources, including non-3GPP sensing inputs. The processed sensing data can be exploited to optimize RAN operations, such as beam steering and power control, or be exposed to external applications through the Y1 interface. In addition, sensing outputs are forwarded to the SMO, which leverages this information to orchestrate network resources and dynamically adapt end-to-end slices, ensuring that both communication and sensing requirements are jointly satisfied.

To provide a unified and Radio Access Technology (RAT)-independent interface for sensing data across heterogeneous sources, **D2.3** [2] further extended the O-RAN E2 interface through the introduction of a sensing-oriented Service Model (SM). This extension enables both 3GPP and non-3GPP nodes to be abstracted as sensing Radio Units (sRUs), which can directly expose sensing information to the Near-RT RIC. It supports multiple sensing data representations, e.g., I/Q samples, Channel State Information (CSI), or heatmaps, allowing flexible trade-offs between data granularity and communication overhead. In addition, it enables control functionalities, allowing xApps to dynamically configure sensing parameters. Through this extension, sensing capabilities are seamlessly integrated into the O-RAN control framework.

Building on this unified sensing data exposure, intelligence in the 6G-SENSES architecture is enabled in the near real time domain through the Near-RT RIC, which operates at time scales ranging from 10 milliseconds to 1 second. The Near-RT RIC hosts xApps that leverage both real-time RAN measurements (collected via the E2 interface) and enriched sensing information to perform closed-loop control of the network. By combining communication and sensing inputs, xApps can make context-aware decisions and dynamically adjust RAN parameters, such as beamforming configurations, power allocation and sensing-aware MAC scheduling, ensuring that both communication performance and sensing requirements are met. The resulting control decisions are enforced through the E2 interface towards the RAN nodes, forming a closed control loop where network actions continuously influence subsequent measurements.

At longer time scales, intelligence is supported by the SMO and the Non-RT RIC, providing centralised network management, analytics, configuration, and orchestration functions. The Non-RT RIC hosts rApps and AI/ML models that process large volumes of historical, contextual, and sensing data. Data is collected from multiple sources and network domains enabling holistic intelligence across the entire system. These functions enable tasks such as model training, traffic and sensing prediction, and long-term optimisation of network configurations. The outcomes of these processes are translated into policies, context information, or model updates, which are delivered to the Near-RT RIC through the A1 interface, or even translated into concrete actions such as RAN reconfiguration via O1. In this way, the SMO and Non-RT RIC provide guidance to near-real-time control loops, enabling coordinated, cross-layer optimisation across RAN, transport, and CN domains.

AI/ML models can be deployed across different parts of the system depending on latency requirements, data availability, and computational constraints, as described in **D2.2** [1]. The 6G-SENSES architecture supports

multiple deployment options, enabling flexible distribution of training and inference functionalities across the system:

- Option 1: Offline Training and Offline Inference. In this case, both training and inference are performed in a non-real-time manner, typically within the SMO or domain-specific orchestrators (e.g., Non-RT RIC). This setup supports long-term network optimisation and reconfiguration based on aggregated sensing and monitoring data. For example, AI/ML models implemented as rApps can analyze historical data and update network configurations accordingly via management interfaces.
- Option 2: Offline Training and Online Inference. Training is performed centrally (e.g., at the SMO/Non-RT RIC using rApps), leveraging large datasets and significant computational resources, while inference is executed closer to the network edge (e.g., in the Near-RT RIC using xApps). This approach enables real-time decision-making using models trained on global network knowledge.
- Option 3: Online Training and Online Inference (Edge-based Learning). In this deployment, both training and inference are performed locally at the edge, typically within the Near-RT RIC. This approach relies on locally available data and supports fully autonomous and adaptive decision-making in real time. It is particularly suitable for scenarios with strict latency requirements or limited data sharing capabilities, while also reducing privacy risks.
- Option 4: Distributed Training and Online Inference (FL). This approach enables collaborative model training across multiple distributed entities without sharing raw data. Each node performs local training using its own data, and model updates are aggregated to form a global model. Inference is then performed locally (e.g., at the edge). This setup combines the benefits of centralised and distributed learning, while addressing privacy and data distribution constraints.

Overall, the 6G-SENSES architecture provides a comprehensive framework to support AI-driven algorithms through flexible deployment, rich sensing data provision from multi-RAT and external sources, and fine-grained control across different network domains and time scales.

2.2 Sensing Data provision

This section presents further details and/or extensions regarding the support of AI agents during this reporting period, focusing on the different types of sensing data available to the agents and the mechanisms for their efficient provision within the 6G-SENSES architecture.

2.2.1 Near-RT-RIC

This section outlines two implementation options to make sensing data accessible to the Near-RT RIC, which, as commented above, serves as the fast-control tier in the O-RAN architecture, actively capturing and processing the data. The first implementation option considers the case when the sensing source cannot process I/Q samples and such processing is offloaded to an xApp at the near-RT RIC. On the other hand, the second implementation option assumes that I/Q samples can be locally processed, so that only processed sensing information is forwarded to the RIC. Figure 2-2 showcases the location of the E2 interface according to the implementation option.

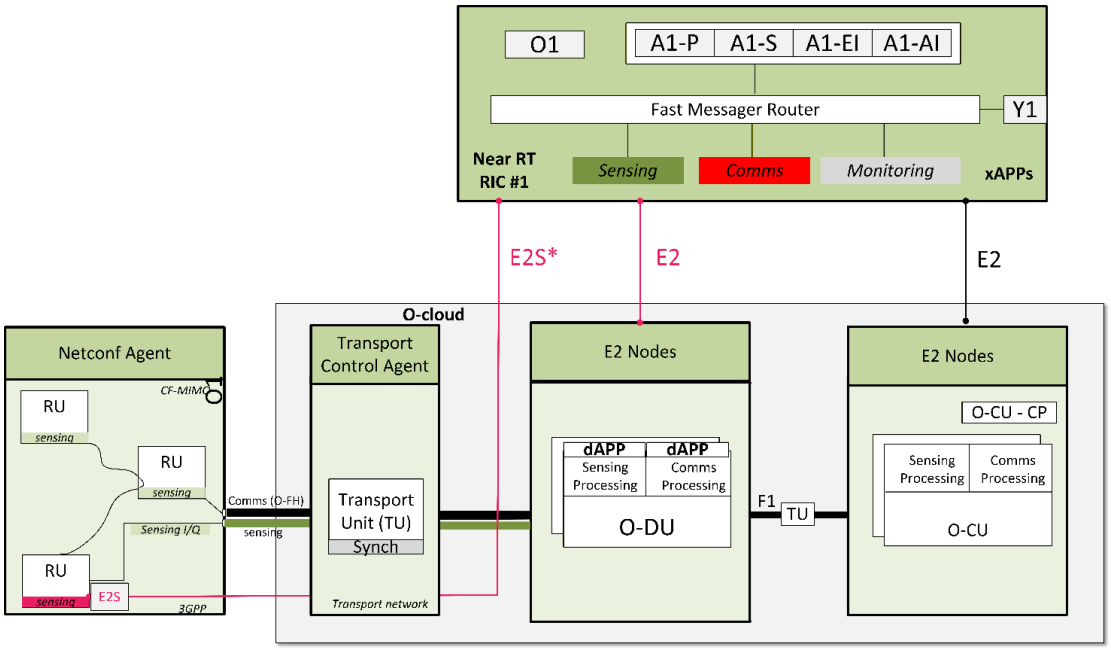


Figure 2-2 Generic 6G-SENSES architecture with particular focus on Service Models

For the first case (E2 option in Figure 2-2) an O-RAN compliant SM is adopted that bears raw I/Q samples. By utilising O-RAN standard measurement SMs, specialised xApps gather real-time network metrics to trigger immediate, dynamic control logic. To guarantee that data is kept for subsequent AI model training, the BubbleRAN MX-RIC integrates robust third-party databases, enabling the stable collection of continuous, multi-hour datasets.

- **Service Models (SMs):** Standard Key Performance Measurements (KPMs) and Logical Link Control (LLC) are used to collect 3GPP sensing data.
- **Pipeline:** xApp logic continuously gathers measurement data to trigger dynamic, fast-loop control responses.
- **Storage Integration:** Native support for third-party databases (SQLite, MySQL, VictoriaMetrics) to ensure stable dataset collection over extended periods.
- **Captured Parameters:** Records RAN configuration parameters, including Physical Resource Blocks (PRBs), gains, center frequency, Time Division Duplex (TDD) pattern, along with vital MAC-layer Key Performance Indicators (KPIs), such as Acknowledgement/Negative Acknowledgement (ACK/NACK), bitrate, Modulation and Coding Scheme (MCS), Signal-to-Noise Ratio (SNR), and Low-Density Parity-Check (LDPC) iterations.
- **Dataset repository:** An example of how the RIC collects data to configure RAN-KPIs is available in the BubbleRAN TelcoFabric Portal[1].

On the other hand, when the sensing processing unit is capable of processing these I/Q samples, a second SM has been designed to provide higher-level, processed sensing information, as illustrated in Figure 2-1 (denoted as E2S*). Additionally, this model enables control over aspects such as the format of the processed data and the sensing range.

[1] BubbleRAN TelcoFabric Portal: https://telcofabric.co/

Table 2-1 Summary of the different components, key functionalities and details regarding the E2SM

Component	Functionality	Details
Sensing SM (E2SM)	Data encapsulation	Encodes sensing data into Indication messages using heatmaps (angle, range, resolution, signal strength).
Indication Message	Data delivery to RIC/xApps	Transmits sensing data as heatmaps; signal strength is sent as a binary buffer.
RIC Control Message	sRU control	Supports multiple sensing control commands (*select_angle*, *select_radio*, *select_refresh_rate*, *select_resolution*) using TLV format.
sRU (Sensing Radio Unit)	Sensing configuration	Receives control commands from xApps to adjust sensing parameters dynamically.
E2 Node / Agent	Data processing & adaptation	Collects data from the sensing unit, performs optional local processing, and formats it according to the SM.
xApps	Data collection & control logic	Receives data via Indication callbacks, stores it in a MySQL database, and issues control commands to the sRU based on decision logic.
Database (MySQL)	Data storage	Stores collected sensing data for offline analysis and validation.

The sensing information is encapsulated within the new SM and included in the Indication message. It is transmitted as heatmaps, characterised by angular and range dimensions, resolution, and signal strength, with the latter conveyed as a binary buffer until fully received at the xApp.

Furthermore, several sensing control commands targeting the sRU, including angle, range, refresh rate and resolution selection (*select_angle*, *select_radio*, *select_refresh_rate* and *select_resolution* commands, respectively), are defined following a Type-Length-Value (TLV) format, which enables their integration within a single RIC Control message. These commands allow the xApps to dynamically configure and control the sensing behavior of the sRU. These functionalities are supported by specific internal logic adaptations in each entity. At the E2 entity, data from the sensing processing unit can be received and locally processed to generate lightweight sensing information, thereby minimising network load when the processing unit is hosted remotely. This way, the sRU is logically split between the radio device and the E2 Agent responsible for gathering the raw data and parsing it into the corresponding format defined by the SM.

Finally, the xApps collect the data through Indication callbacks and store it in a database for subsequent analysis. In addition, the xApps have been enhanced with the capability to issue sensing control commands toward the sRU based on predefined logic or decision criteria. Table 2-1 summarises the key components, their functionalities and details.

A comparative analysis, based on the second SM is done in terms of traffic load injected to the network. To estimate the sensing data rate over the E2 interface, two approaches are considered. In the first one the transmitted information corresponds to CSI, while in the second case the information is first processed and heatmaps are transmitted. In both cases an active monostatic radar function is assumed, where one channel estimate is generated for each angle using Orthogonal Frequency Division Multiplexing (OFDM) signals. As for the E2SM, besides the corresponding data, it bears information about: range of angles, coded in 3 integers (4 bytes each) that correspond to first angle, last angle and step; range of distances (only for processed information), coded in 3 integers (4 bytes each), minimum, maximum and step distance; and data precision

(only for processed information). Control commands from xApps to the radar devices are also considered, but the corresponding rate is negligible. For both types of information (CSI and heatmaps) the same reference solution [5] is used, which has 63 angles and 1024 subcarriers, from which only 828 are used for channel estimation, and the sensing rate is 100 Hz.

Table 2-2 summarises the corresponding parameters, as well as the two possible configurations: (i) raw data and (ii) heatmaps, along with their configurability. Option 1 corresponds to the data rate when CSI is sent by the E2 terminal. In this case, the channel is encoded using 30 bits (18 bits for amplitude and 12 bits for phase). Therefore, the data rate over the E2 interface, when transmitting CSI, denoted as R_{CSI}, can be expressed as:

$$R_{CSI}=\frac{N_{SC}\cdot N_{angles}\cdot 30+8\cdot R_A}{T_{sensing}},$$

where N_{SC} and N_{angles} represent the number of subcarriers (828) and angles (configurable), respectively, R_A denotes the angle range overhead (12 bytes), and T_{sensing} is the sensing period. Assuming a sensing rate of 100 Hz, the resulting traffic ranges from 2.49 Mbps (for a single angle) up to 156.5 Mbps (for 63 angles).

For the heatmap-based approach, up to 128 distance bins are considered, determined by the radar range resolution, which depends on system parameters, such as signal bandwidth. The resulting data rate, denoted as R_{heatmap}, can be expressed as:

$$R_{heat}=\frac{8\cdot\left(N_{dist}\cdot N_{\text{angles}}\cdot Res\right)+8\cdot(R_A+R_D+\text{Prec})}{T_{sensing}},$$

where $N_{\text{dist}}$and N_{angles} denote the number of distances and angles, respectively, Res represents the data resolution (e.g., float or double), R_A and R_D correspond to the angle and distance range overheads (12 bytes each), Prec is the configured data precision (4 bytes), and T_{sensing} is the sensing period.

Overall, the sensing traffic ranges from 0.432 Mbps (for a single angle and all distances) up to 25.83 Mbps (for the maximum number of angles and distances).

2.2.2 SMO and Non-RT-RIC

Operating at a higher abstraction level, the SMO and Non-RT RIC govern the slow control loops required for long-term network intelligence. This tier leverages aggregated historical sensing data to apply comprehensive policies and make strategic decisions via AI-powered rApps. An example application of this framework is the AI-driven sensing trajectory prediction where an rApp orchestrates the operation of multiple xApps performing sensing at Near-RT RIC level to estimate the trajectory of a moving target.

Table 2-2 Baseline Radar Configuration and E2 Sensing SM Traffic Load.

Parameter/Metric	Option 1: CSI (Raw Data)	Option 2: Heatmaps (Processed)
Subcarriers (N_{SC})	828	N/A (Processed)
Distances (N_{dist})	N/A	Up to 128
Data resolution	30 bits (18 Amp / 12 Phase)	4 bytes
E2SM Overhead	Angle range (R_A: 12 bytes)	R_A (12 B) + R_D(12 B) + Prec(4 B)
Minimum rate (1 angle)	**2.49 Mbps**	**0.432 Mbps**
Maximum rate (63 angles)	**156.5 Mbps**	**25.83 Mbps**

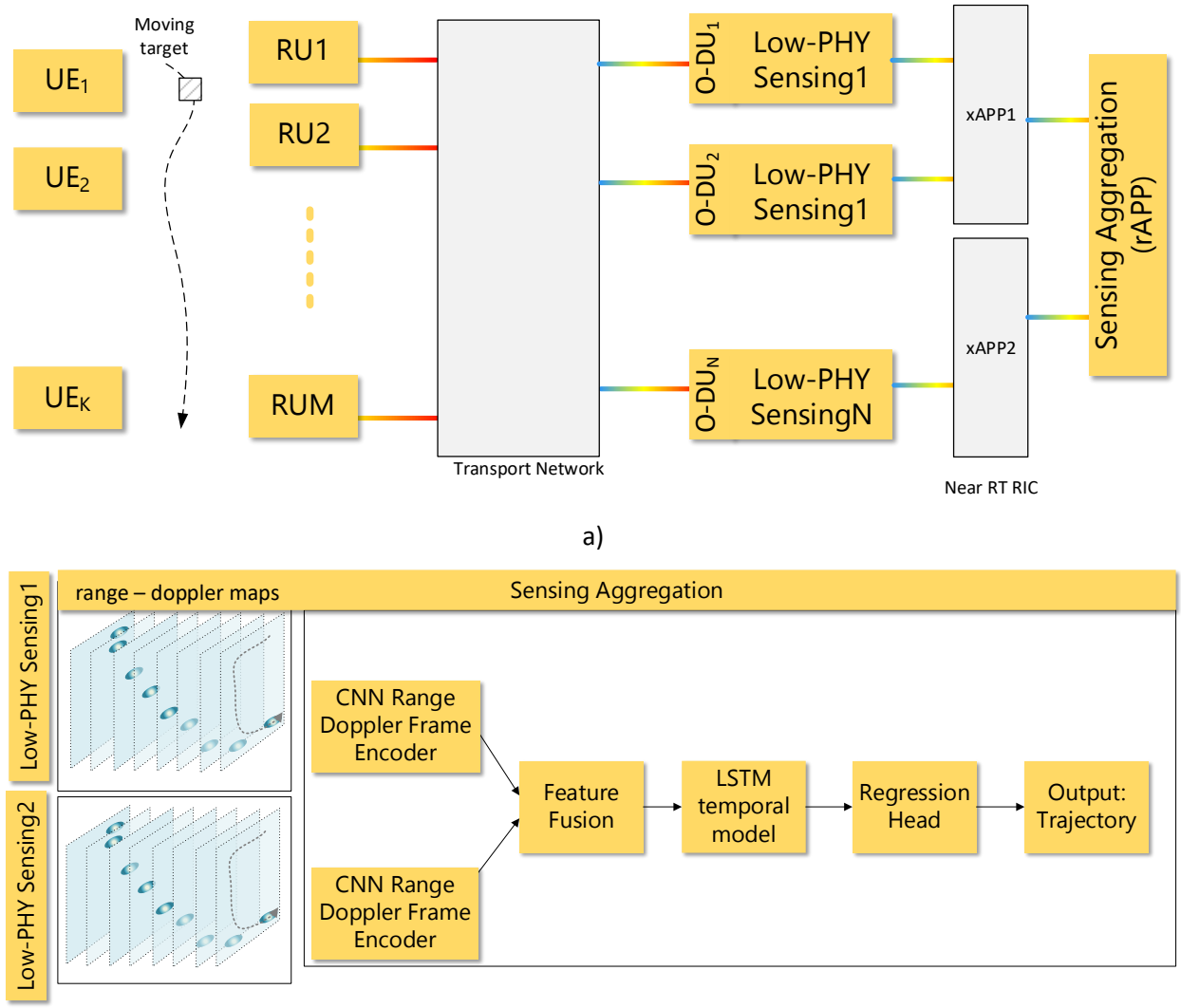


Figure 2-3 a) Hierarchical aggregation of sensing information, b) Fusion of sensing flows and processing by the rApp to perform trajectory prediction.

For the example shown in Figure 2-3, the rApp performs trajectory prediction by fusing sensing flows originating from multiple Low-PHY sensing nodes. After receiving features extracted from distributed xApp-controlled sensing pipelines, the rApp applies higher-layer AI models (e.g., temporal learning and regression networks) to estimate the future path of a moving target. Based on these predictions, it can further instruct the network to proactively steer sensing resources, activate additional nodes, optimise beam configurations, or hand over sensing responsibilities to neighbouring cells.

- **Scope:** Governs slow control loops for long-term policy generation and strategic decision-making via AI rApps.
- **Data Granularity:** Processes and analyzes 1-second 5G time-series measurements.
- **Example Use Case:** Trajectory prediction. rApps require sensing information exposure from xApps establish baselines and accurately identify trajectories for moving targets.
- **Dataset repository**: Range doppler radar images extracted from multiple O-RUs processed by different xApps.

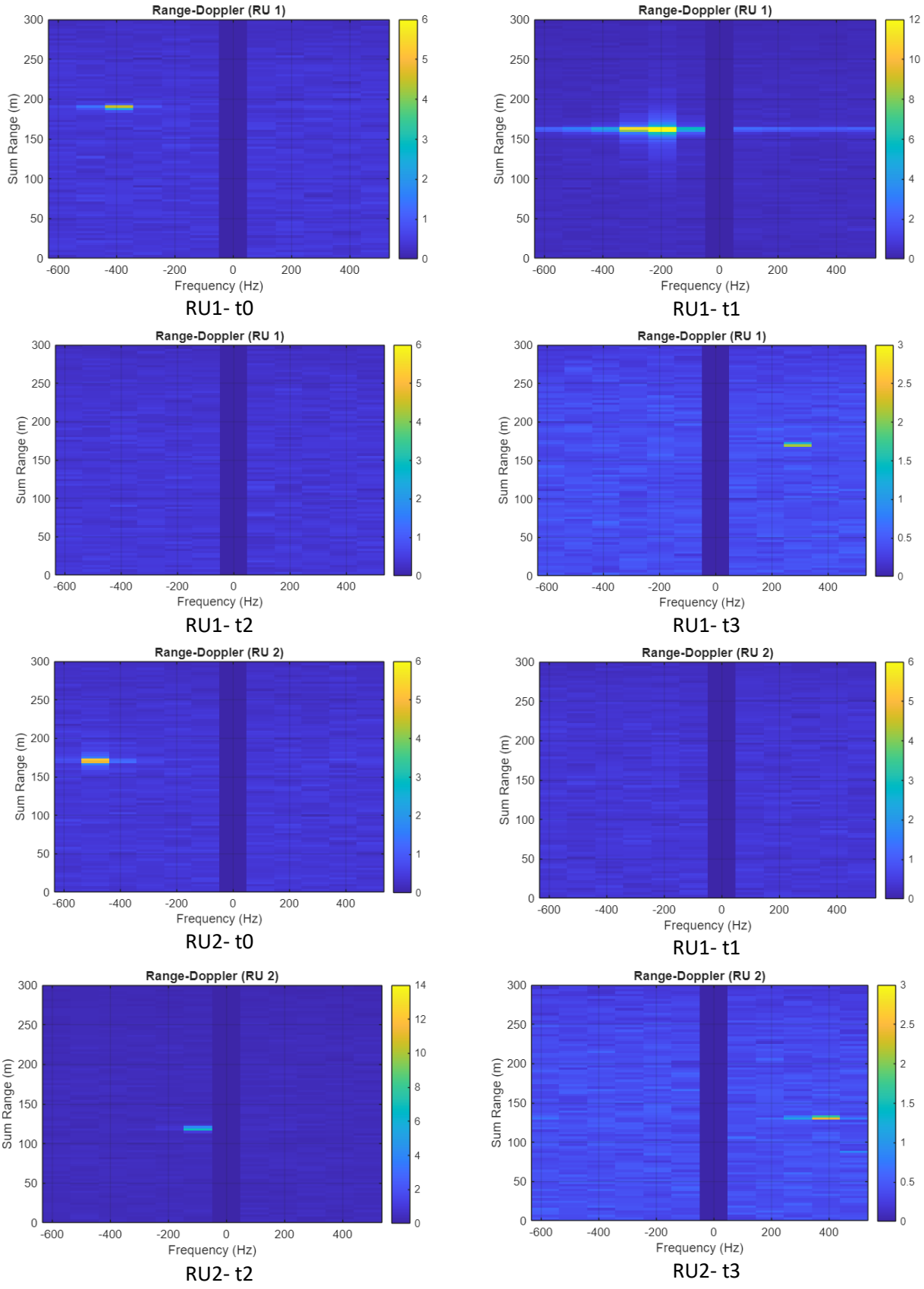


Figure 2-4 Frames of range doppler radar images monitoring the same moving target from two different RUs

In the O-RAN architecture depicted in Figure 2-2, sensing services are supported by extracting, at the O-DU, the lower layer RF signals that can then be processed to detect moving targets. In the context of 6G, RF signals can be processed using Multiple-Input Multiple-Output (MIMO) OFDM radar algorithms extracting range-Doppler radar images [1]. These images can be post-processed to estimate trajectories of moving targets. Based on these estimated trajectories suitable policies can be applied to ensure sensing service continuity, i.e., select the RU/DU configurations for sensing. Typical frames extracted from a sensing processing function implementing a MIMO-OFDM radar using 5G waveforms is shown in Figure 2-4. Frames correspond to range–Doppler map images obtained from two physically separated RUs. The horizontal axis represents the Doppler frequency (Hz) corresponding to the relative radial velocity of a moving target and the vertical axis represents the total distance (range in m) of that target from the RU. The color scale indicates the power of the received signals with brighter colors denoting stronger reflections. For RU1 at time t0 a peak is visible around a range of approximately 180–190 m and a negative Doppler frequency (around −400 Hz). This high-energy region corresponds to a moving target detected by RU 1 at that distance. The negative Doppler shift indicates that the target is moving towards the RU. The absence of other strong peaks implies that a single moving object has been detected. Combining the output of multiple range-doppler maps the trajectory of a moving target can be estimated.

Range–Doppler map images extracted by the Low-PHY sensing functions at different RUs are forwarded to a centralised sensing aggregation function, where they are jointly processed. The fusion of these multi-view sensing observations enables multi-static sensing and can significantly improve trajectory estimation accuracy compared to single-RU sensing. This improvement is achieved by exploiting spatial diversity and complementary Doppler and range perspectives from multiple RUs.

Figure 2-3 shows the processing pipeline that is executed at the sensing aggregation function (rApp) performing trajectory prediction for the moving target. The estimation is based on a hybrid Convolutional Neural Network - Long Short-Term Memory (CNN-LSTM) that predicts trajectories combining range–Doppler radar frames from two RUs. In the spatial domain, we leverage a Convolutional Neural Network (CNN) that transforms range–Doppler frames into a vector that captures target-related signatures. These vectors are then combined, providing a multi-RU perspective for target signatures. The fused feature sequence is fed into a Long Short-Term Memory (LSTM) temporal model, which learns motion dynamics across frames. Finally, a regression head maps the LSTM output to the desired trajectory, estimating the next location of the moving target in the map.

In this model, illustrated in Figure 2-5, sensing operates in a fully centralised manner, where a large volume of range–Doppler radar images generated at multiple RUs/DUs is transmitted over the transport network to a central sensing aggregation function (e.g., MEC or cloud). This central entity is responsible for aggregating all sensing measurements, training the corresponding AI models with the global dataset, and performing inference for tasks such as target detection, localisation, and trajectory estimation. The model can provide accurate trajectory predictions that are exploited by higher layer control functions to proactively allocate radio, transport and compute resources and ensure service continuity for sensing-enabled services. While this approach enables globally optimal model training and simplifies coordination across sensing nodes, it imposes significant requirements on fronthaul/backhaul capacity, introduces additional latency, and creates a potential computational and reliability bottleneck at the central aggregation point. Implementation details and numerical results for this model including a FL approach and comparisons are carried out in Subsection 5.4.

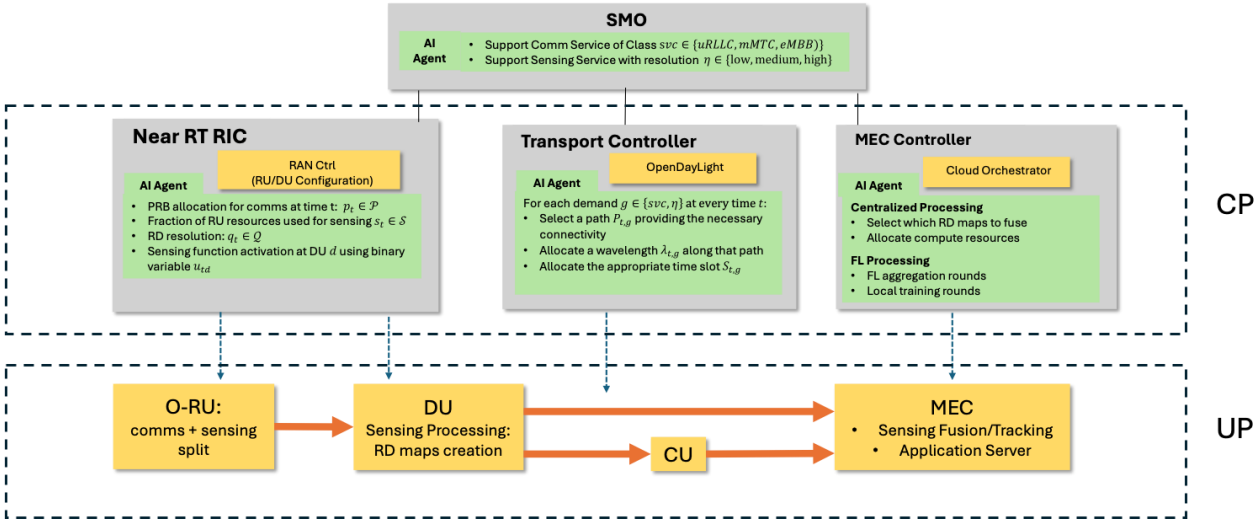


Figure 2-5 Hierarchical AI agents for sensing and communications

3 Datasets

This section describes the datasets generated for training and validation of the AI/ML-based solutions and algorithms presented in sections 4 and 5. The description of the datasets follows the "Common Metadata Template for Datasets emerge from the SNS-JU projects" generated by the SNS JU TMV data re-usability Sub-WG[2].

According to the document structure, datasets for RAN/Edge optimisation are presented in Section 3.1, while those focused on E2E network intelligence are introduced in Section 3.2. The dataset description is divided in four parts. The first one provides general information of the dataset, including its overall description and the related algorithm. The second part details the infrastructure or emulation/simulation solution adopted to generate the dataset. Afterwards, network segment specific information is provided. The network segments may correspond to the RAN, CN or transport network. Only those network segments that are considered in the corresponding dataset are described. Finally, the measurement points and generated metric are enumerated.

In all cases the categories indicated in each field follow the ones suggested in the aforementioned template. In this sense, it is worth noting that some measurements do not correspond to any of the types listed in Annex 2 of the template, which are in turn defined in 3GPP TS 28.552. In those cases, a measurement description is provided.

3.1 Datasets for resource allocation and optimisation in RAN/Edge

	Simulation dataset on sensing supported MAC scheduling
Identifier	https://10.5281/zenodo.20139852
List of Contributor(s)	**UC**
Dataset description	This dataset is related to the algorithm presented in Section 4.1. It contains temporal evolution of a communication scenario involving a single 5G gNB and multiple UEs running XR services. Over that communication scenario, sensing information becomes available to the RAN Intelligent controller. Then, such sensing information is used to adapt the MAC scheduling policy considering both QoS service requirements and radio resource utilisation. The dataset contains metrics used for MAC scheduling, including RLC buffer occupancies and MCS. Furthermore, traffic application metrics are also recorded, according to the kind of XR service. On top of that, sensing information used to optimize the MAC scheduling decisions are also kept, with the UE location and obstacle detection. The data has been generated using the ns-3 5G-LENA simulator, which implements the two-level MAC scheduling. An xApp gathers RAN metrics (including sensing data) and communicates with the MAC scheduler to support resource allocation decisions.
Dataset related publication	"DRL-Based Dynamic MAC Scheduler Reconfiguration in O-RAN", doi: 10.1109/ICC52391.2025.11160805 "A Software Architecture for Seamless Simulated to Real Testbed Transition for O-RAN AI", doi: 10.1109/NoF66640.2025.11223328
Keywords	MAC, scheduling, QoS, DRL, Lyapunov, ns-3
	Underlying network & compute infrastructure

[2] https://smart-networks.europa.eu/sns-ju-working-groups/

Underlay Platform	Base 5G-LENA (https://5g-lena.cttc.es/) Extended version in https://github.com/tlmat-unican/RBIS
Environment	Indoors (simulation).
Network Infrastructure domain(s)	RAN (5G-NR)
Compute infrastructure type	Private-cloud node
	RAN Focused
RAN 3GPP Release	Release 19
New Radio Type	5G-NR
RAN split	N/A
RAN focused technology	O-RAN, ISAC, RIC
Coverage Type	Single-cell
Frequency Band	4 GHz, and 6 GHz
Bandwidth (MHz)	20 MHz
Number of involved devices	N/A
Mobility Model	Pedestrian, Random walk
Other parameters	TDD pattern, numerology, gNB/UE transmission power, gNB/UE antenna elements, obstacle sizes, propagation model
	Experimentation Process and Metrics
Observation point	In the horizontal view it corresponds to the communication between end device and access node. Vertically, it corresponds to the radio level
Type of measurement	DRB (measurements related to Data Radio Bearer). RRU (measurements related to Radio Resource Utilisation) QF (measurements related to QoS Flow) L1M (measurements related to Layer 1 Measurement)
Measurement Tools used	Logging system of the adopted simulation framework.
Measured Indicators	Throughput (Mbps); delay (ms); packet size (B); RLC buffer occupancy(bytes); MCS (index); Location (m/m; cartesian coordinates); obstacle location (m/m; cartesian coordinates)

3.2 Datasets for AI-Driven E2E network intelligence and sensing

	O-RAN E2E performance metrics
Identifier	https://doi.org/10.5281/zenodo.20147311
List of Contributor(s)	**UC**
Dataset short description	This dataset is related to the solution presented in Section 5.1. It considers x-haul network traffic, generated by end-nodes: RAN, sensing and core elements, as well as configurable traffic forwarding nodes. The generated traffic flows belong to both communication and sensing services. In the case of communication flows, these are: fronthaul traffic according to the underlying RAN configuration; backhaul traffic following specific service features (i.e. URLLC, eMMB, mMTC); sensing traffic according to specific sensing capability configuration; in the following it will be referred as sensing

	backhaul. The forwarding nodes allow different labelling and traffic scheduling policies. The dataset records end-to-end (E2E) metrics of the different flows that can be used to compute per-packet delay, jitter, and throughput. In addition, sojourn time at intermediate nodes is also recorded.
Dataset related publication	N/A
Dataset related keywords	O-RAN; 6G; ISAC; Fronthaul; ns-3
	Underlying network & compute infrastructure
Underlay Platform	ns-3 network simulator (https://www.nsnam.org/). Modified version in https://github.com/tlmat-unican/GNN-Xhaul.git
Environment	Outdoors (due to the cell bandwidth, simulation)
Network Infrastructure domain(s)	Transport, E2E
Compute infrastructure type	Private-cloud node
	Transport Network Focused
Type of transport	Wired, fiber optics, packet switched
Transport segment	Fronthaul, backhaul, sensing backhaul.
Other parameters	RAN configuration to model fronthaul and sensing traffic flows. It includes bandwidth, TDD pattern and numerology of the 5G network.
	Experimentation Process and Metrics
Observation point	In the horizontal view it may correspond, according to the traffic type, to the RU to DU domain (fronthaul), access node to network core (backhaul) or access node to edge node (sensing traffic). From a vertical viewpoint, it corresponds to the Network Layer.
Type of measurement	IP (measurements related to IP Management) QF (measurements related to QoS Flow)
Measurement Tools used	Logging system of the adopted simulation framework.
Measured Indicators	*PacketUniqueID (integer); Timestamp transmission (ns); timestamp reception (ns); packet size(B); buffer occupancy (bytes); delay (ms); throughput (Mbps)*

	Monostatic Full Duplex Wi-Fi Sensing Walking person
Identifier	https://doi.org/10.5281/zenodo.20147649
List of Contributor(s)	**INT**
Dataset short description	This dataset is generated from a scenario with 2 commercial laptop devices and a user walking. The dataset comprises three different scenarios: approach/leave, static breathing, and distance zones. In the first scenario the user walks towards and away from the laptop across a range of 0.5 to 8m and back at a walking speed. Two full approach-leave sequences were recorded, each using one of the two laptops. In the second scenario (static breathing) the user sits still approximately 3m from one laptop and breathes normally. No macro-movement is present, only micro-Doppler from respiration is captured. The third scenario comprises records with distance zones annotated in 1-meter

	step (0-8m), synchronised with a ground truth video recorded simultaneously. The data is then labeled identifying the users' distance zone at each time frame. The dataset contains Wi-Fi CSI data collected during the experiments that can be used to develop different sensing solutions.
Dataset related publication	"Human Presence Detection via Wi-Fi Range-Filtered Doppler Spectrum on Commodity Laptops", doi: https://doi.org/10.48550/arXiv.2603.10845
Dataset related keywords	Wi-Fi, CSI, Human Presence estimation
	Content Information
Underlay Platform	HP laptop with W-iFi 7 and Lenovo ThinkPad with Wi-Fi 6E.
Environment	Indoors
Network Infrastructure domain(s)	RAN (Wi-Fi)
Compute infrastructure type	Private-edge node
	RAN Focused
RAN 3GPP Release	N/A
New Radio Type	Wi-Fi 7
RAN split	N/A
RAN focused technology	ISAC
Coverage Type	Single-cell
Frequency Band	6 GHz
Bandwidth (MHz)	160 MHz
Number of involved devices	2
Mobility Model	Pedestrian
	Experimentation Process and Metrics
Observation point	In the horizontal view it corresponds to the communcation link between the end device and the access node. From a vertical perspective the observation point is at the Radio Level.
Type of measurement	It does not fit into the categories in the template.

Measurement Tools used	Raw CSI including real and imaginary components with different Long Training Field (LTF) duration.
	HP laptop (Wi-Fi 7) and Lenovo ThinkPad (Wi-Fi 6E)
Measured Indicators	Time stamp (ms); Wi-Fi channel number (integer); target frame Interval (ms); Carrier frequency (MHz); number of subcarriers (integer); CSI (complex)

	Gesture recognition using Wi-Fi sensing on Commercial Off-The-Shelf (COTS) hardware
Identifier	https://doi.org/10.5281/zenodo.20147884
List of Contributor(s)	**INT**
Dataset short description	This dataset is related to the algorithm presented in Section 5.5. The data is generated in a scenario comprising a single laptop device equipped with a WiFi-7 card, and 5 users. During the data generation the users in turn do 5 different gestures multiple times. The dataset is generated from 55 different sessions, 50 of those corresponding to a lab environment and 5 to a café/public space, yielding a total of 722 gesture instances. The café subset provides an out-of-environment evaluation for cross location generalisation. The gestures include: forward/backward hand wave, up/down hand move, pulse gesture (push forward and back), clockwise circular motion and side-to-side wave. From these scenarios the raw CSI is recorded and time stamped.
Dataset related publication	"WIRD-GEST: Gesture Recognition in the Real World Using Active Range-Doppler Wi-Fi Sensing on COTS Hardware", Accepted in ICC 2026 Workshop (to be published)
Dataset related keywords	Wi-Fi, CSI, Human Presence estimation
	Content Information
Underlay Platform	Lenovo ThinkPad L14
Environment	Indoors
Network Infrastructure domain(s)	RAN (Wi-Fi)
Compute infrastructure type	Private-edge node
	RAN Focused
RAN 3GPP Release	N/A
New Radio Type	Wi-Fi 7
RAN split	N/A

RAN focused technology	ISAC
Coverage Type	Single-cell
Frequency Band	6 GHz
Bandwidth (MHz)	160 MHz
Number of involved devices	1
Mobility Model	Pedestrian
	Experimentation Process and Metrics
Observation point	In the horizontal view it corresponds to the communcation link between the end device and the access node. From a vertical perspective the observation point is at the Radio Level.
Type of measurement	It does not fit into the categories in the template. Raw CSI including real and imaginary components with different Long Training Field (LTF) duration.
Measurement Tools used	Intel Wi-Fi card X211 NIC,
Measured Indicators	Time stamp (ms); Wi-Fi channel number (integer); target frame Interval (ms); Carrier frequency (MHz); number of subcarriers (integer); CSI (complex)

	Multistatic sensing in FR1/FR3 bands
Identifier	https://doi.org/10.5281/zenodo.20148195
List of Contributor(s)	**IASA**
Dataset short description	Dataset containing multistatic sensing measurements in FR1 and FR3 frequency bands, represented as range–Doppler radar images. The dataset captures reflected signal characteristics from two distributed sensing nodes operating in the FR1 and FR3 bands, respectively, under different locations for targets. This dataset enables AI-driven target detection, tracking, classification, and integrated sensing and communications (ISAC) research for beyond-5G/6G systems
Dataset related publication	FR1 band covered through A. Tzanakaki et. al., “Optical Transport for Networked Integrated Sensing and Communication Services”, IEEE/Optica Journal of Optical Communications and Networking, 2026 in press.
Dataset related keywords	ISAC, 5G-NR, FR1, FR3, Multi-band ISAC, range doppler radar images
	Content Information

Underlay Platform	5G-NR platform
Environment	Indoor
Network Infrastructure domain(s)	RAN (5G-NR)
Compute infrastructure type	Private Cloud
	RAN Focused
RAN 3GPP Release	Rel.16
New Radio Type	5G-NR
RAN split	7.2
RAN focused technology	ISAC
Coverage Type	2-cell
Frequency Band	3.5 GHz, 10.5Ghz
Bandwidth (MHz)	20 MHz, 100MHz
Number of involved devices	2
Mobility Model	Pedestrian
	Experimentation Process and Metrics
Observation point	Point to-Point LoS connectivity for communication services Placement of reflectors emulating targets at different angles and positions relative to the O-RUs.
Type of measurement	RD heatmaps Detected target peaks
Measurement Tools used	Custom xApp collecting RD images
Measured Indicators	Timestamp (ms); reflector/target position coordinates; target angle relative to the O-RUs (degrees); carrier frequency (MHz); bandwidth (MHz); number of subcarriers; transmit power (dBm); received signal strength (RSSI/RSRP); SNR/SINR; CSI values (complex I/Q samples); propagation delay (ns); estimated range (m); Doppler shift (Hz); estimated target velocity (m/s); Range-Doppler map; angle-of-arrival (AoA); packet synchronisation accuracy (PTP offset/jitter).

4 Algorithms for Resource Allocation and Optimisation in RAN/Edge

This chapter presents a set of algorithms developed within the 6G-SENSES project for resource allocation and optimisation in the RAN and Edge domains, targeting the efficient management and control of radio and storage resources under dynamic and heterogeneous conditions. The solutions considered span both learning-based and rule-based techniques and operate at different layers and timescales, with the common objective of improving key performance metrics such as throughput, latency, energy efficiency, and reliability. In particular, they leverage key enabling technologies such as RIS for enhanced radio performance and MEC for adaptive caching at the network edge. Importantly, sensing capabilities enabled by ISAC and the 6G-SENSES architecture provide contextual information, which is exploited to support intelligent resource allocation across both radio (e.g., RIS configuration and MAC scheduling) and caching decisions.

Section 4.1 introduces a Deep Reinforcement Learning (DRL)-assisted sensing-aware MAC scheduling framework, where a learning agent dynamically tunes scheduler parameters based on the observed network state. By incorporating ISAC capabilities, the agent can leverage predictive environmental information, such as anticipated Line-of-Sight (LoS) blockages due to user mobility, enabling proactive rather than reactive scheduling decisions. The framework is evaluated using both Deep Q-Network (DQN) and Proximal Policy Optimisation (PPO), demonstrating that DQN achieves superior performance due to its higher sample efficiency and ability to learn from rare but critical events via experience replay. Importantly, the proposed approach maintains QoS requirements while significantly improving resource efficiency, achieving approximately 30% reduction in PRB usage compared to baseline schedulers, with results further validated in a real O-RAN testbed.

Section 4.2 focuses on resource optimisation in Active Simultaneously Transmitting and Reflecting RIS (ASRIS)-aided THz ISAC systems, proposing a distributed multi-agent DRL framework based on Multi Agent Deep Deterministic Policy Gradient (MADDPG). In this setting, RIS elements act as agents that jointly optimize communication and sensing parameters, including beamforming and radar processing variables, under realistic constraints such as mobility and synchronisation. The distributed formulation enables scalable and adaptive control, while experimental results demonstrate clear performance gains over baseline methods, such as Deep Deterministic Policy Gradient (DDPG), PPO, and conventional RIS solutions, achieving higher sum rates and improved robustness. The study also provides insights into key system design trade-offs, including RIS deployment, number of elements, and sensing-communication performance balance.

Section 4.3 examines Passive Simultaneously Transmitting and Reflecting RIS (STAR-RIS)-assisted full-duplex ISAC systems and introduces a meta-reinforcement learning (MRL) approach to improve learning efficiency and adaptability. The proposed single-agent framework focuses on optimising RIS phase shifts and energy splitting coefficients, achieving faster convergence and higher sum-rate performance compared to standard DRL baselines, like DDPG and Twin Delayed DDPG (TD3), even under imperfect CSI conditions.

Finally, a location-aware caching strategy is proposed in Section 4.4 to enhance data availability and system performance in both wireless sensing and MEC application scenarios. By leveraging context information such as user or target location and channel conditions, the approach enables adaptive location-aware caching decisions at the network edge. The results show significant improvements over conventional caching policies, including up to 30% reduction in energy consumption, lower end-to-end delay, and reduced cache MISS rates.

Overall, the presented works demonstrate that the combination of learning-based and rule-based techniques with sensing-aware optimisation brings efficient and adaptive resource management and control in the RAN and Edge domains.

4.1 DRL-Assisted Sensing-Aware MAC scheduling

4.1.1 Problem Description

5G and beyond networks must support heterogeneous services with diverse and often conflicting QoS requirements. In this context, the MAC scheduler at the gNB plays a critical role in determining how PRBs are allocated among UEs. Traditional MAC schedulers, e.g., Proportional Fairness (PF) or QoS-aware heuristics, rely on predefined mathematical rules and static weight tuning. Within the O-RAN architecture, the introduction of the Near-RT RIC enables the deployment of AI/ML models as xApps to optimise RAN behavior. This opens the possibility of augmenting or supervising MAC scheduling decisions using data-driven approaches.

The MAC scheduler considered in this work is based on the drift-plus-penalty algorithm derived from Lyapunov optimisation. This scheduler provides theoretical queue stability guarantees and enables explicit control, through a set of tunable parameters, over the trade-off between throughput, delay, and resource efficiency. The optimal configuration of these parameters is highly scenario-dependent and sensitive to traffic mix, load levels, and channel variability.

To address this challenge, an intelligent mechanism is required to adapt the scheduler configuration periodically in response to the evolving network state. Accordingly, a DRL-assisted MAC scheduling framework is proposed, in which a learning agent supervises and adjusts the scheduler parameters to optimize long-term network performance. Furthermore, this framework is enhanced by ISAC capabilities, allowing the DRL agent to leverage predictive environmental awareness, such as anticipating LoS blockages due to user mobility, to proactively optimise long-term network performance rather than merely reacting to channel degradation.

4.1.2 Algorithm Description

The MAC scheduling optimisation problem is formulated as a Markov Decision Process (MDP), in which the DRL agent interacts with the network environment by observing system states, selecting control actions, and receiving feedback through a reward signal.

The state representation captures relevant MAC-layer and radio information describing the current network conditions. This includes indicators of radio quality, such as MCS derived from Channel Quality Indicator (CQI) reports, as well as traffic-related metrics including buffer occupancy and achieved throughput per UE. In addition, QoS requirements, such as Guaranteed Flow Bit Rates (GFBRs) are incorporated, together with other contextual information, including sensing-based location when available. By combining these elements, the state provides the agent with a comprehensive view of both channel conditions and service demand. Formally, the observation space S is defined as:

$$S=\{N, M_n, \Gamma_n, \Lambda, \overline{Q}_n, \overline{G}_n, P_n | n=1, \dots, N\},$$

where N is the number of UEs, M_n is the MCS for each n-th UE, Γ_n is the GFBR requirement, Λ represents the total available PRBs. Furthermore, $\overline{Q}_n$ and $\overline{G}_n$ correspond to the time average occupancy of the RLC queues and virtual throughput queues, respectively, which keep track of the throughput deficit according to the defined GFBR. Finally, P_n is a channel degradation prediction indicator based on sensing data.

The action space is defined in terms of control parameters of the scheduler rather than direct PRB allocation decisions. This supervisory design enables a hierarchical control structure: the computationally efficient drift-plus-penalty scheduler operates at the slot level (<1 ms), while the DRL agent executes over longer control windows corresponding to the xApp subscription interval (10–1000 ms). The action space A dictates the values assigned to the scheduler weights:

$$A=\{V, \omega_{Q_n}, \omega_{G_n} | n=1, \dots, N\},$$

where V controls the importance of resource efficiency, ω_{Q_n} manages the stabilisation of the RLC queues, and ω_{G_n} prioritises the stabilisation of the virtual queues.

The reward function captures the primary performance objectives of the system. It includes metrics related to QoS satisfaction (compliance with throughput targets) and overall resource efficiency. To prevent the agent from taking defective actions, such as dropping traffic entirely to maximize resource savings, the resource efficiency bonus is only considered if the GFBR requirements are fully met. The reward function R is mathematically expressed as:

$$R\big(S(\tau), A(\tau)\big)=T \cdot \frac{1}{N}\sum_{n=1}^{N} min\left(1, \frac{\overline{\rho}_n}{\Gamma_n}\right) + \left[B+(1\text{-}T\text{-}B)\left(1\text{-}\frac{\sum_{n=1}^{N}\overline{\alpha}_n}{\Lambda}\right)\right] \cdot 1_{\{R_{thput}=1\}},$$

where T is a weighting factor for the throughput component, B is a bonus reward for satisfying the throughput requirement, ρ_n is the achieved throughput, α_n represents the allocated resources, and the indicator function $1_{\{R_{thput}=1\}}$ ensures the resource efficiency term is strictly evaluated, only when all throughput targets (i.e., GFBRs) are fulfilled.

To learn the optimal control policy, model-free DRL algorithms are employed. In particular, PPO, an on-policy actor-critic method, and DQN, an off-policy baseline, are considered to drive the weight-adaptation process.

4.1.3 Evaluation

The evaluation scenario consists of a single gNB serving four UEs located within the same beam. Initially, simulations of the RAN are carried out using the ns-3 5G-LENA simulator, which provides detailed modeling of the 5G NR PHY and MAC layers. All UEs generate XR traffic, including two UEs configured with Virtual Reality (VR) and two with Cloud Gaming (CG) traffic. The VR traffic corresponds to high-bandwidth, low-latency video streams for immersive environments, while the CG traffic models workloads involving computation offloading to a remote server. Both traffic types present different QoS requirements and statistical characteristics, including packet sizes, inter-arrival times, and burstiness, creating heterogeneous and demanding conditions for the RAN. Furthermore, to evaluate the impact of sensing information on scheduling, the simulation environment incorporates UE mobility following a random-walk model, and physical obstacles using the ns-3 buildings module. These obstacles may cause LoS blockages that can significantly degrade channel quality. Although the simulator tracks the exact positions of all nodes, the framework abstracts this information into a predictive binary indicator that signals whether a UE is likely to enter a high-probability blockage region based on its recent trajectory.

The MAC scheduler performs RB allocation, while a DRL agent operates as an external control entity that periodically tunes the scheduler configuration. The DRL agent is deployed as an xApp and trained online, updating the scheduler configuration with a control interval of 1 second. Communication between the simulated RAN (5G-LENA) and the DRL agent is implemented through a REST API that emulates a real RIC interface.

The proposed DRL-assisted MAC scheduling framework, namely Reinforcement-Based Intelligent Scheduling (RBIS), is evaluated by comparing two DRL approaches: the value-based algorithm DQN and the policy-gradient algorithm PPO. The objective is to analyze their convergence behavior and the quality of the scheduler parameter configurations learned by the agents.

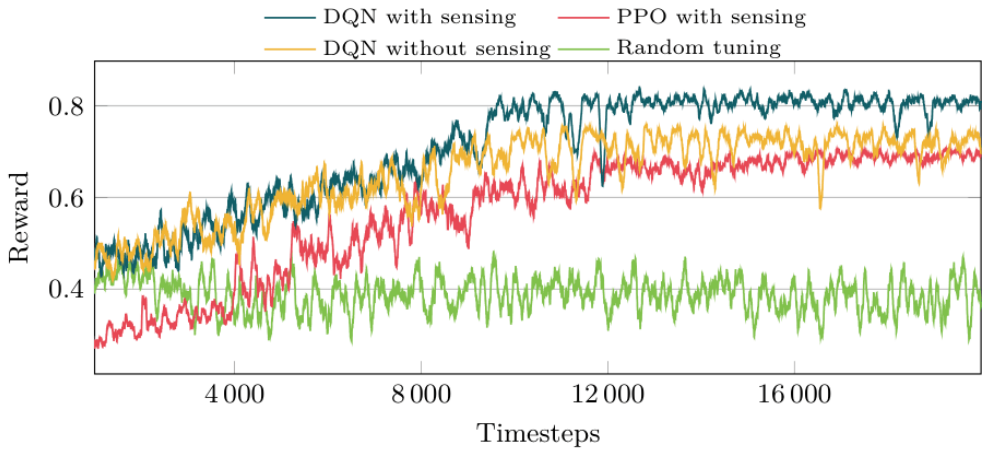


Figure 4-1 Comparison of the reward obtained over time

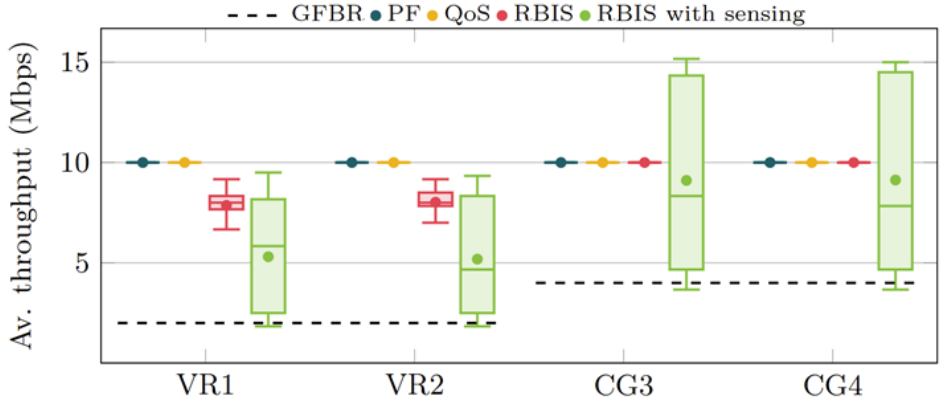


Figure 4-2 Average throughput achieved by each UE in the simulation environment

To this end, Figure 4-1 presents the evolution of the reward obtained during training for the considered agents. For DQN, configurations with and without sensing information are evaluated. Sensing provides additional contextual awareness of the environment, allowing the agent to better anticipate channel variations and adjust the scheduler parameters accordingly. For PPO, only the configuration including sensing information is shown in the figure, as the differences observed when excluding sensing were negligible. The curves correspond to a moving average computed over 100 training steps. As shown in the figure, all DRL agents significantly outperform the random tuning baseline. After convergence (around 10,000 steps), DQN with sensing achieves the highest performance with an average reward of 0.81, compared to 0.71 for DQN without sensing and 0.69 for PPO. This performance disparity highlights a fundamental difference between off-policy and on-policy learning paradigms in this specific environment. The PPO agent, being on-policy, exhibits premature convergence to a safe but suboptimal strategy. Because it discards data after policy updates, it fails to adequately learn from rare but critical events, such as sudden channel fading caused by user mobility around obstacles. Conversely, the DQN agent is highly sample-efficient and utilises an Experience Replay Buffer. By storing and repeatedly sampling past transitions, DQN successfully amplifies the learning signal of these infrequent but crucial channel degradation events, allowing it to discover a more globally optimal policy.

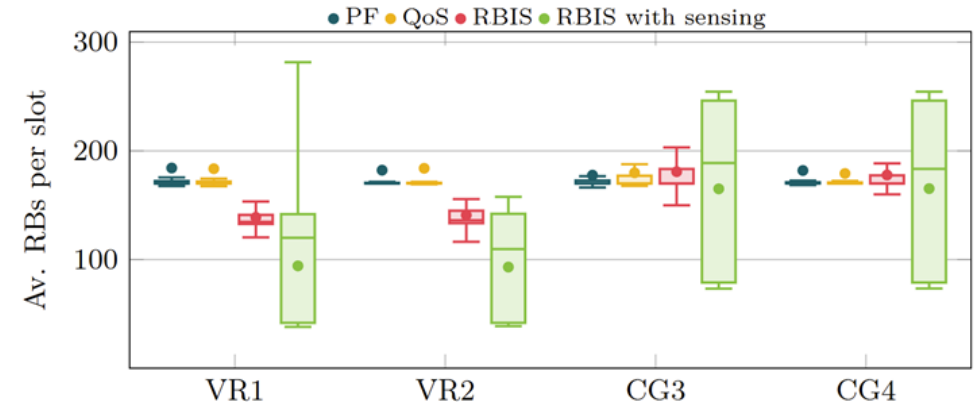


Figure 4-3 Average number of PRBs allocated per slot for each UE in the simulation environment

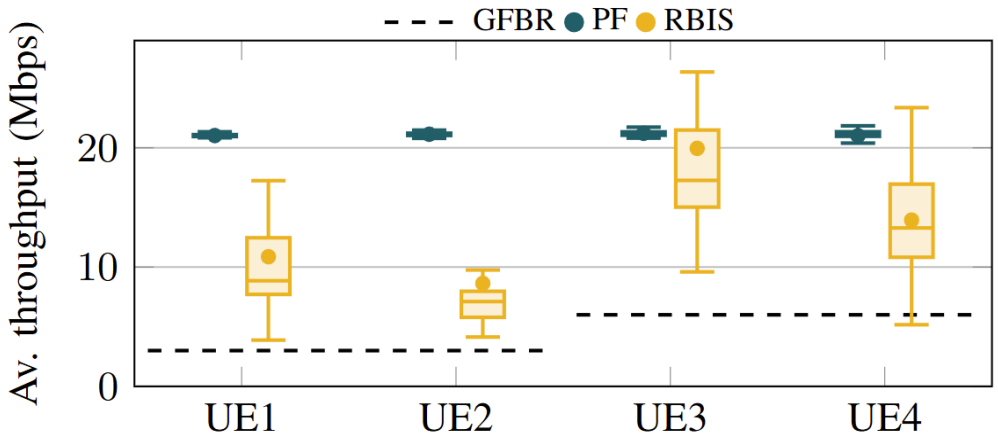


Figure 4-4 Average throughput achieved by each UE in the real testbed.

The performance of the DRL-driven scheduler was benchmarked against traditional PF and QoS-aware heuristic schedulers. As for throughput, Figure 4-2 indicates that all evaluated schedulers generally succeed in meeting the minimum GFBR requirements for both VR and CG traffic flows. However, the key advantage of the DRL-driven approach lies in its resource efficiency. As illustrated in Figure 4-3, which reports the average number of PRBs allocated per slot, RBIS aligns throughput with the required GFBR, thereby avoiding unnecessary resource allocation. In this simulated scenario, the RBIS approach achieves a significant 29.29% reduction in average PRB usage per slot compared to the baseline schedulers.

To demonstrate the feasibility of transitioning the DRL xApp from a simulated environment to operational hardware, the trained policy was deployed in a real-world O-RAN testbed. The physical testbed features a 3GPP-compliant OpenAirInterface (OAI) gNB and a FlexRIC instance as the Near-RT RIC, both running on an Intel NUC, paired with a USRP B210 Software Defined Radio (SDR). To manage subscriber authentication and session handling, a remote instance of Open5GS is employed as the 5G Core. The users consist of four Commercial Off-The-Shelf (COTS) Raspberry Pi 5 units equipped with Quectel RM520N-GL 5G modems.

The real testbed validates the simulation results under real-world conditions. The DRL agent continuously processes E2 SM telemetry and dynamically updates scheduler parameters. As shown in Figure 4-4, the scheduler consistently meets the GFBR requirements for COTS UEs. Figure 4-5 depicts the distribution of PRB usage per slot, with RBIS achieving a 30% reduction in the total PRBs allocated across all UEs compared to the baseline PF scheduler.

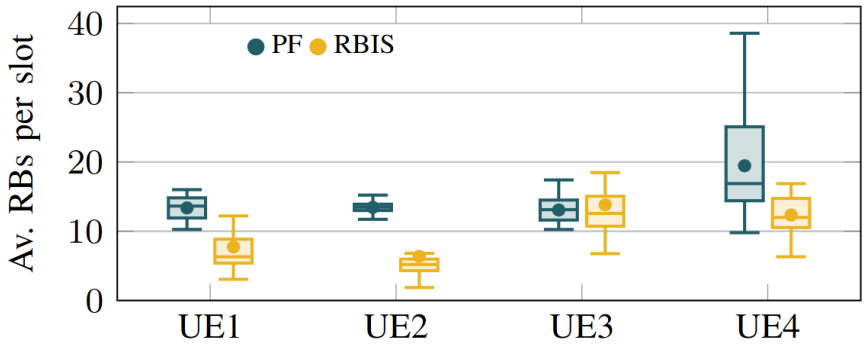


Figure 4-5 Average number of PRBs allocated per slot for each UE in the real testbed.

4.2 Multi-Agent DRL for Resource Optimisation in ASRIS-Aided THz ISAC Systems

4.2.1 Problem Description

In the era of advanced wireless communication technologies, the integration of sensing and data transmission has become a pivotal research area to enhance resource utilisation and efficiency [12]. One promising technology in this domain is ISAC, which seeks to share hardware and spectrum resources for data transmission and target sensing [13]. The adoption of terahertz (THz) frequencies in ISAC systems, owing to their ultra-wide bandwidth, has the potential to revolutionize the capabilities of ISAC systems [14]. Nevertheless, THz signals confront substantial challenges, including pathloss and poor diffraction, which curtail their coverage range.

To address these issues, RIS have become a viable solution, offering low-power and low-complexity hardware elements that can redirect THz signals, thereby not only expanding their coverage but also enhancing the range of ISAC applications [15]. However, conventional passive RIS systems suffer from multiplicative fading i.e., the product of two independent path losses (from the transmitter to the RIS and from the RIS to the receiver), which severely attenuates the reflected signal. This limitation is especially pronounced when a direct link already exists between the transmitter and receiver, as the RIS-assisted path often becomes negligibly weak compared to the direct path. To overcome this limitation, Active RIS (ARIS) was proposed, where each reflecting element amplifies signals to mitigate multiplicative fading, all while maintaining low power consumption and hardware simplicity. STAR-RIS, which stands for "simultaneously transmitting and reflecting," adds an interesting dynamic solution by combining the reflection abilities of ARIS with the extra ability to transmit signals to cover a large area. Recognising the complementary benefits of these two approaches, the study of active STAR-RIS (ASRIS) has garnered significant interest [16].

The existing body of work in ISAC systems predominantly revolves around passive RIS, limiting improvements in communication or sensing performance due to multiplicative pathloss effects or double fading. While ARIS has found application in ISAC systems, the utilisation of ASRIS in ISAC systems remains an unexplored frontier. Moreover, most existing studies focus on joint beamforming optimisation under two fundamental assumptions: (i) perfect timing synchronisation across various signal paths, and (ii) the assumption of static vehicular units or ignoring mobility parameters in the optimisation problem. Regardless of their paths, signals are assumed to always arrive at their intended destinations simultaneously under the assumption of perfect timing synchronisation. While this assumption simplifies the analysis and design of ISAC systems, it might not remain applicable in practice, especially in dynamic settings where timing variations are inevitable [17]. Moreover, when employing traditional channel equalisation or multicarrier processing, the system can become complex. Neglecting mobility parameters in the optimisation framework overlooks the dynamic nature of communication scenarios. There is currently no existing solution that combines both of these practical considerations in ASRIS-aided ISAC systems.

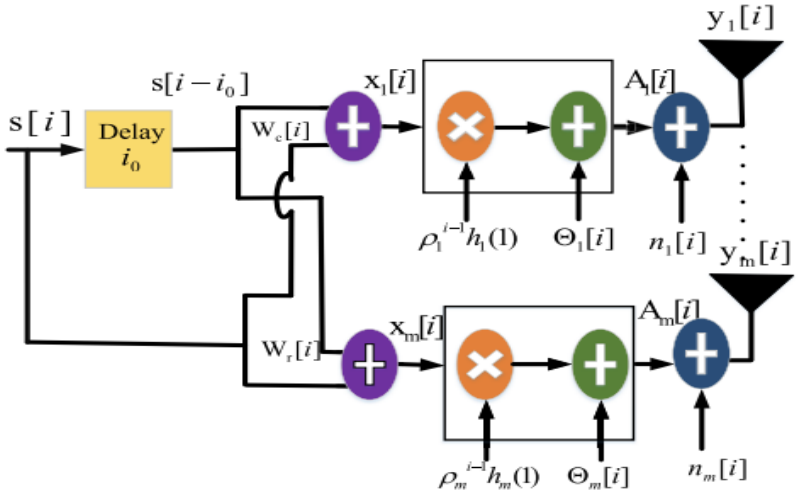


Figure 4-6 Block diagram of the proposed DDA modulation

The optimisation problem formulated in this work aims to jointly optimize Base Station (BS) transmit beamforming, ASRIS reflection and transmission beamforming matrices, vehicular unit (VU) mobility correlation parameters, and radar receive filter to maximize the sum rate while meeting the worst radar SNR requirement, ASRIS hardware constraints, and power budget constraints [18]. It is evident that the optimisation problem presents non-convex characteristics. This non-convexity arises from the intricate nature of the objective function, which involves logarithmic and fractional terms, the interdependencies between variables within both the objective function and total power constraints for BS and ASRIS.

4.2.2 Algorithm Description

To address this problem, an efficient machine learning-based MADDPG algorithm is introduced. The MADDPG algorithm offers unique characteristics that make it highly suitable for optimising system parameters and maximising the sum rate in the proposed THz ISAC system. MADDPG facilitates collaborative learning among agents, allowing them to make coordinated decisions in dynamic THz ISAC environment. It demonstrates adaptability to dynamic changes in system parameters, enabling efficient resource allocation and improved performance in dynamic scenarios. Moreover, MADDPG effectively navigates complex parameter spaces, converging to high-quality solutions within the non-linear optimisation landscape of the THz ISAC system.

The proposed approach implements a novel dynamic delay alignment (DDA) modulation method for the ASRIS-assisted ISAC system. The block diagram of the proposed system is shown in Figure 4-6. The elements of ASRIS in transmission space and reflection space are implemented as agents in the multi-agent scenario to handle the resource allocation tasks for the set of VUs in the transmission space and reflection space, respectively, each responsible for its decision-making relying on local observations for sum rate optimisation. To determine the best joint resource allocation strategies, the agents interact with the environment at distinct-time increments. The training of the multi-agents is carried out at the BS during the offline training phase. This offers the chance to share additional information to streamline the training process.

The state space consists of the real and imaginary components of the BS–VU, RIS–VU, and BS–RIS channel coefficients, along with their combined representations, for all agents. Since current ML platforms do not support complex inputs, the complex values are split into real and imaginary components. Each agent utilises its action space at each time step to decide an action in accordance with the current resource allocation policy, based on local observation of the environment state. The action space at each time step includes actions for each agent, comprising beamforming vectors, radar receive filter coefficients, and ASRIS beamforming matrices.

The two main deep neural networks (DNNs) used by the MADDPG agent are an actor-network, which approximates a joint resource allocation policy, and a critic network, which approximates a state-value function. The target networks are implemented as time-delayed copies of the actor and critic networks to improve training stability. The actor deterministically maps local observations to continuous actions at each time step, after which an exploratory policy is applied to encourage exploration. The environment collects the joint actions of all agents and returns the corresponding rewards along with the next observations. The reward function is defined based on the system sum rate to guide the learning process. To enhance training stability and sample efficiency, the agents' transitions are stored in replay buffers. During training, mini-batches of samples are randomly drawn from these buffers to update the networks.

The agents intend to continuously enhance their individual policies until they reach the best resource allocation policy and maximize the estimated cumulative reward. The policy network modifies its settings in order to optimize the projected long-term discounted reward according to the gradient of the multi-agent deterministic policy. Soft updates are used to update the target parameters in the actor and critic networks.

4.2.3 Evaluation

The simulation results of the proposed system's performance are discussed using a three-dimensional coordinate system with the BS positioned at the origin and the ASRIS positioned at (0, 15, 0) meters. On both sides of the ASRIS, VUs are randomly distributed within an elliptical area with a semimajor radius of 30 meters and semiminor radius of 5 meters centered around the ASRIS. The number of BS antennas and ASRIS elements are modeled as $M = 6$ and $Nr = 32$ respectively. A conventional path-loss model, accounting for distance-dependent attenuation, is employed, and all channels involved are assumed to be Rician fading. A uniform Doppler power spectra (DPS) model is used for mobility analysis of the VUs, considering the velocity of the VU to be 30 miles per hour unless specified otherwise.

The convergence of the system's sum rate is illustrated employing both the MADDPG algorithm and the DDPG algorithm across two distinct scenarios with V = 5 and V = 10 VUs as shown in Figure 4-7 a. Notably, MADDPG outperforms DDPG, attributable to two key reasons. Firstly, MADDPG facilitates collaborative information sharing among agents representing the ASRIS elements in the transmission and reflection regions, leading to more informed and coordinated decision-making. This collaborative approach enhances the exploitation of available resources, contributing to improved sum rates. Secondly, MADDPG's adaptive nature in dynamic environments provides a significant advantage over DDPG, as it leverages a centralised critic with access to joint actions and observations, enabling more effective learning in the presence of non-stationarity introduced by multiple interacting agents. The observed trend reveals that the sum rate is higher when fewer VUs are present and decreases as the number of VUs increases, justified by reduced contention and interference when fewer VUs exist.

The interplay of DDA modulation, VU mobility, and power budget in the system demonstrates that MADDPG consistently outperforms DDPG. When DDA modulation is employed without mobility, the system experiences a high sum rate, primarily attributed to DDA modulation's capability to optimize signal alignment, ensuring that the signal from the ASRIS is treated as the desired signal rather than interference. With mobility at 30 mph, DDA modulation remains effective and outperforms the scenario without DDA modulation, especially at high power budgets, credited to DDA modulation's dynamic adjustment feature compensating for mobility-induced channel variations as shown in Figure 4-7 b. In the absence of DDA modulation and mobility, the sum rate performance starts degrading as the power budget rises because the system erroneously treats the RIS signal as interference.

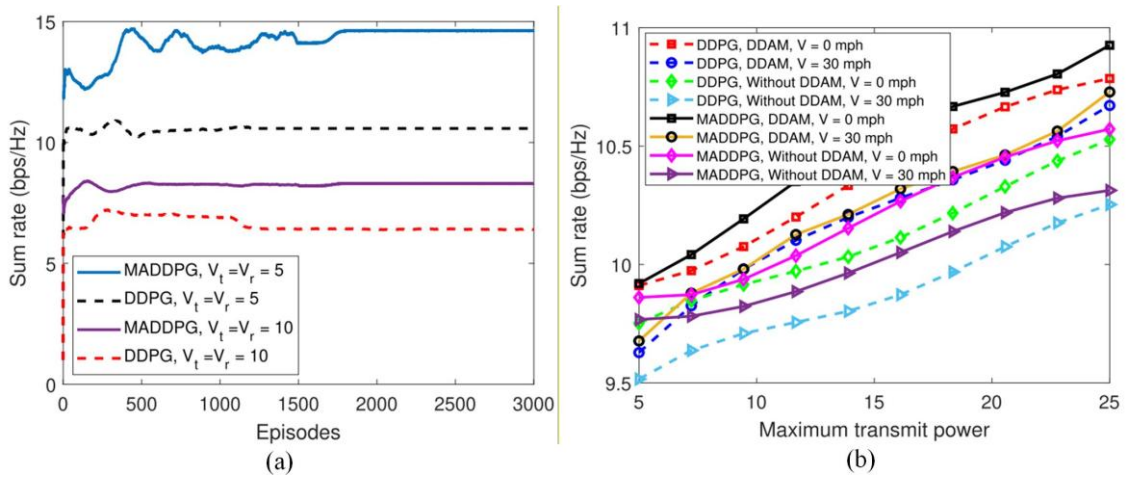


Figure 4-7 (a) Convergence of MADDPG and DDPG algorithms with different VU distribution. (b) Impact of maximum transmit power, DDAM and mobility

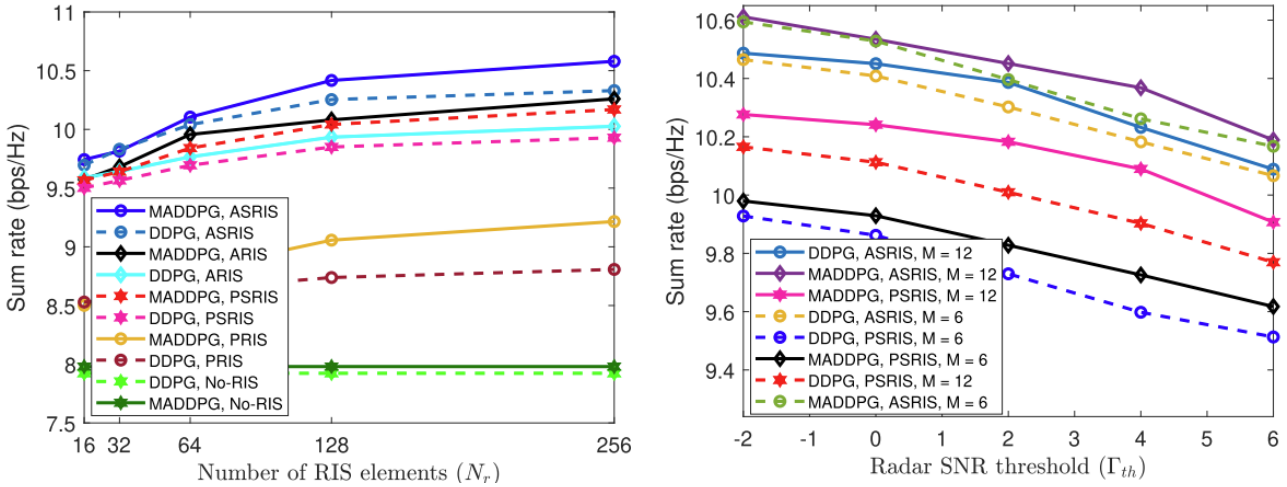


Figure 4-8 (left) Effect of number of RIS elements for various RIS configurations (right) Impact of radar SNR threshold and number of antennas at BS

The sum rate performance concerning the number of VUs in both transmission and reflection spaces demonstrates the efficiency of the system in managing increasing VU demands, with the addition of reflecting elements notably enhancing sum rate performance. When the number of users in transmission and reflection spaces are equal, the observed direct relationship between increasing users and sum rate is indicative of the system's efficient resource allocation mechanism. When the user count in the transmission space surpasses that of the reflection space, with reflection space users remaining constant, the system begins to show signs of strain, manifesting as a decline in sum rate attributable to increased demand for direct communication resources exceeding the capacity of the system. In contrast, increasing the number of users in reflection space, while keeping transmission space constant, illustrates the capability of the system to leverage the reflective properties of ASRIS more effectively, with enhancement in sum rate underscoring the efficiency of ASRIS in redirecting and optimising signal paths.

The intricate relationship among DDA modulation, VU mobility, and maximum RIS power comparing MADDPG with benchmark algorithms DDPG, PPO, and MPPO reveals that MADDPG and DDPG algorithms outperform both MPPO and standard PPO in terms of achieving higher overall sum rate (as evident from results presented in Figure 4-7 and Figure 4-8). This superior performance is primarily attributed to the exploration-exploitation balance fundamental to these algorithms. As an off-policy algorithm, DDPG inclines

towards exploitation, leveraging the learned policy more effectively to enhance the sum rate. Conversely, PPO and MPPO, which are on-policy algorithms, place greater emphasis on exploration. The implementation of an adaptive clipping strategy within MPPO facilitates an improvement over PPO through dynamic modification of the clipping range, enhancing exploration while mitigating policy divergence risk.

Investigation of ASRIS deployment at seven distinct positions along the Y-axis from -75 to 75 reveals that the system attains its peak sum rate performance when the RIS is positioned at (0, 0, 0), essentially aligning it precisely with the center of the operational area. This optimal performance underscores the significance of ASRIS deployment location in relation to path loss characteristics of the reflection link. By situating the ASRIS at the center, the system benefits from reduced path loss and enhanced signal reflection capabilities, thereby maximising effective utilisation of ASRIS's advanced signal processing and directional modulation features. The proximity of ASRIS to the center minimises the average distance to users, reducing signal attenuation and improving quality of reflected signals, while allowing for more uniform distribution of signal coverage.

4.3 STAR-RIS-Aided Full-Duplex ISAC Systems: A Novel MRL Approach

4.3.1 Problem Description

As commented above, STAR-RIS is an advanced meta-surface technology that splits incoming signals into transmitted and reflected components, enabling 360-degree coverage — a major improvement over traditional RIS [19]. Integrating STAR-RIS with ISAC technology creates a promising architecture for next-generation wireless networks, enhancing communication, sensing, and spectral efficiency. While prior work has explored half-duplex RIS-assisted ISAC systems using conventional optimisation methods (e.g., block coordinate descent), these approaches are computationally intensive and yield sub-optimal solutions [20]. Full-duplex STAR-RIS can double spectral efficiency but remains underexplored in ISAC contexts. DRL offers a model-free alternative to address these limitations. The solution presented in this section examines full-duplex STAR-RIS-assisted ISAC systems using DRL and the system model is shown in Figure 4-9.

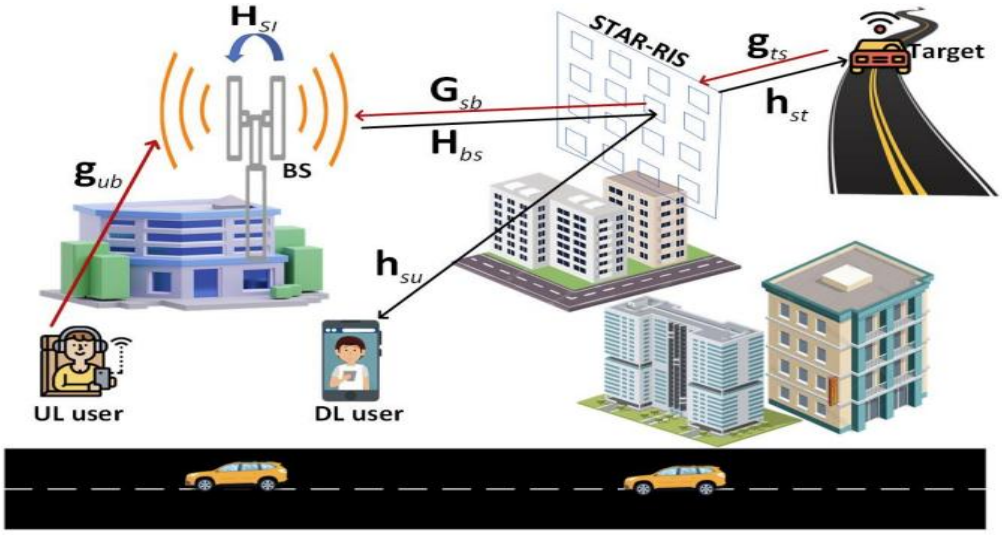


Figure 4-9 System Model

The use of STAR-RIS in wireless communications has inspired researchers to explore the integration of STAR-RIS and ISAC to leverage the benefits of both technologies. Specifically, incorporating STAR-RIS in ISAC systems can create a virtual LoS path to overcome the effect of obstacles and introduce more degrees of freedom to enhance system performance. STAR-RIS can enhance the sensing capabilities of ISAC systems while simultaneously boosting data transmission rates using multiple spatially separated streams. Additionally, STAR-RIS can divide a signal into two distinct signals, which is a good match for ISAC's dual

function. Consequently, using full-duplex mode in ISAC systems allows improved spectral efficiency compared to half-duplex mode. The communication and sensing scenario under consideration is realistic, posing a significant challenge in enabling efficient communication. Specifically, designing appropriate phase-shift matrices for STAR-RIS is an arduous task.

The optimisation problem formulated in this section aims to maximize the sum rate by jointly optimising the phase-shifts at the STAR-RIS for energy splitting mode. The constraints denote the limitations of phase and amplitude of STAR-RIS elements. The phase shifts must lie within a specified range, and the amplitude coefficients for reflection and transmission for each element must sum to unity while remaining between zero and one. The optimisation problem is non-convex due to the presence of these constraints and thus challenging to solve.

4.3.2 Algorithm Description

To solve the non-convex sum-rate maximisation problem, a MRL approach is adopted, which utilises the MDP framework. Before describing the MRL formulation, it is important to clarify what constitutes a *task* in this context. Each task corresponds to a distinct channel realisation, that is, a specific instantiation of the channels between the BS, STAR-RIS, users, and sensing target, as defined by the state space. Because channel conditions vary across episodes, the agent must adapt its phase-shift policy rapidly to each new realisation. DNNs are used by MRL to learn reinforcement learning (RL) tasks, and after updating parameters with just a few samples, the MRL method seeks to learn an initial parameterised control policy with the best performance on a new task. The parameters of a model for fast learning are trained using a meta-learning technique based on a first-order gradient, which saves computational resources because it does not require back-propagation in the outer loop of training, where the outer loop refers to the meta-level update of the exploration policy, as distinct from the inner-loop DDPG-based updates of the exploitation policy.

Standard RL optimises a policy for a fixed task. MRL extends this by introducing a task distribution from which different channel realisations are drawn across episodes. A key distinction from conventional RL lies in the composition of the agent's input: although RL and MRL are similar in structure, the distinction resides in the agent's most recent actions, rewards, and current state all being incorporated as input to the policy. Specifically, the agent follows a policy $\pi(a^{t-1}, r^{t-1}, s^t)$, that is, the policy takes as input not only the current state but also the previous action and previous reward. The primary aim of incorporating these previous rewards and actions is to facilitate the policy's comprehension of the connections and dynamics between the present state, the agent's latest actions, and rewards in the current MDP.

The MDP is formulated as a communication between an exploration agent (π_e) and an exploitation agent (π). The three components of the MDP are defined as follows.

- **State:** The state at each time step comprises the full set of channel gains required to evaluate the system, defined as:

$$s^t = \{ h_{st}, H_{bs}, h_{su}, g_{ts}, g_{ub}, G_{sb} \}.$$

  This includes the BS-to-STAR-RIS channel (H_{bs}), the STAR-RIS-to-DL-user channel (h_{su}), the UL-user-to-BS channel (g_{ub}), the STAR-RIS-to-BS channel (G_{sb}), and the channels between the target and STAR-RIS for both downlink (h_{st}) and uplink (g_{ts}) directions.

- **Action:** The action at each time step is defined as the reflection and transmission phase-shift matrices of the STAR-RIS:

$$a^t = \{ \theta_r, \theta_t \}.$$

These constitute the passive beamforming coefficients, specifically, θ_r and θ_t denote the phase shifts for the reflection and transmission matrices, respectively.

- **Reward:** The reward is defined as the instantaneous sum-rate $\tilde{R} \triangleq R_{tot_u} + R_{tot_d}$ encouraging the agent to maximise total system throughput across both uplink and downlink directions.

The working mechanism proceeds as follows. The exploration policy π_e interacts with the STAR-RIS-assisted ISAC environment in discrete time steps, generating rollout data B_o by executing π_e, where the rollouts refer to the process of simulating the agent's behaviour in the environment. This data is used to update the exploitation policy π via DDPG, yielding an updated policy $\pi' = D(\pi, B_o)$. Additional rollout data B_1 is then generated from π' to estimate its reward $\hat{R}^{\pi'}$. The meta-reward is estimated as:

$$\hat{R}(B_0) = \hat{R}_{\pi'} - \hat{R}_{\pi}.$$

The exploration policy is then updated according to the gradient of the meta-policy using learning rate η.

4.3.3 Evaluation

The utility of the STAR-RIS for full-duplex aided ISAC systems is illustrated while considering the case of perfect CSI as well as imperfect CSI. The effectiveness and convergence of the proposed algorithm are numerically evaluated and compared with other RL algorithms like DDPG and TD3. The simulation parameters include a BS with 4 transmit and 4 receive antennas, noise power of -70 dBm, carrier frequency of 24.2 GHz, transmit power at BS of 20 dBm, residual self-interference factor of 1,00,000, 500 episodes, discount factor of 0.99, replay buffer size of 10000, batch size of 30, 300 steps per episode, and 200 rollout steps.

The accuracy of the CSI greatly impacts the performance of the system under consideration. Obtaining perfect CSI is difficult due to the simultaneous reflection and transmission through STAR-RIS, resulting in some error in the estimated CSI, which exhibits a circular symmetric Gaussian distribution following the statistical CSI error model. The received signals and SINR expressions are modified accordingly to account for the imperfect channel estimates.

To show the superiority of the proposed scheme with optimal phase shifts, it is compared with two baseline schemes. DDPG is an off-policy, model-free algorithm that learns policies in continuous action spaces. It combines DNNs and the actor-critic method to learn policies where both networks are trained simultaneously using gradient descent such that the expected cumulative reward is maximised. TD3 addresses the issue of Q-function overestimation common in DDPG by utilising two Q-functions, delayed policy update, and target policy smoothing.

The rewards performance versus the number of episodes for the proposed MRL algorithm as well as the benchmark schemes show that for the MRL algorithm with perfect CSI, rewards increase significantly in initial training episodes and converge approximately after 200 episodes. The proposed algorithm achieves the highest reward at approximately 15 bits/s/Hz, demonstrating its effectiveness in providing optimum phase shift matrices. Similar observations are seen for imperfect CSI with convergence achieved at 13.4 bits/s/Hz. Due to the imperfection in CSI, there is a decrease in net signal strength, resulting in performance degradation of approximately 1.5 bits/s/Hz compared to the perfect CSI case. Additionally, the MRL algorithm outperforms DDPG and TD3 algorithms in both full-duplex and half-duplex modes. All these results are shown in Figure 4-10.

The comparative sum-rate performance of the proposed STAR-RIS framework with baseline schemes by varying the number of STAR-RIS elements (Figure 4-10 left) shows that the required sum-rate increases with increasing number of elements. This is due to the rise in reflecting gain at the RIS with more elements, which

increases SINR at each user. The performance of MRL with perfect CSI is 26% better compared to MRL with imperfect CSI when the number of elements is 30.

The performance of downlink rate and uplink rate for two different self-interference cancellation factors versus transmit power at the BS for the proposed algorithm (Figure 4-10 middle) demonstrates that as the SNR at the downlink user increases with higher transmit power, the downlink rate increases consistently. However, at the uplink transmission, the signal must deal with self-interference from the BS, resulting in decreased uplink rate as transmit power increases. This declining trend is due to incomplete cancellation of interference caused by inadequacy in self-interference cancellation. When the self-interference factor increases, uplink rate decreases from 10 to 7.8 bits/s/Hz at 10 dBm.

The impact of S-RIS location on the performance of both uplink and downlink data rates (Figure 4-10 right) demonstrates that placing the S-RIS and users close to each other achieves better performance. The performance of both systems exhibits an initial increase, reaching a peak value, and subsequently declining beyond a specific deployment location. This behavior is attributed to the reduction in signal strength between the BS and S-RIS link when the S-RIS and users are in close proximity to each other, and vice versa.

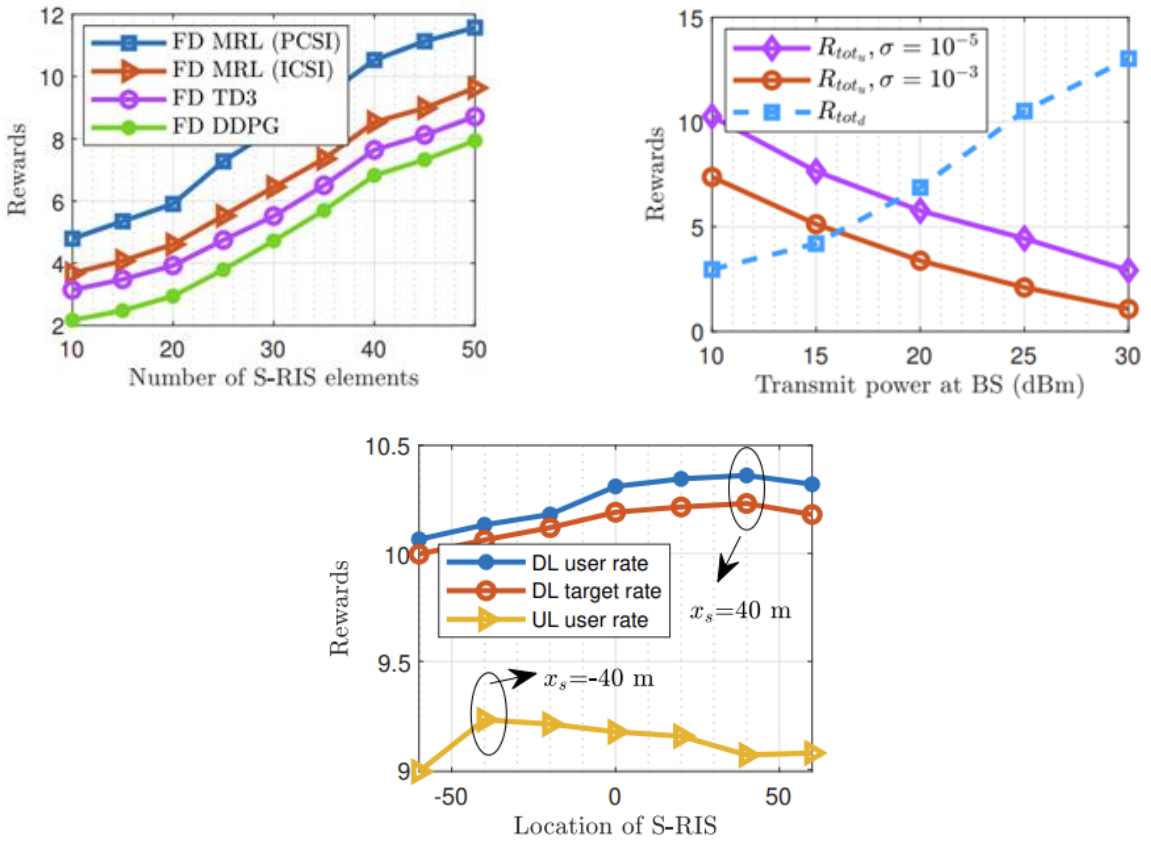


Figure 4-10 (left) Rewards versus number of elements in the STAR-RIS (middle) Trade-off between UL rate and DL rate (right) Rewards versus location of S-RIS.

4.4 Location-Aware Caching Strategies

As mentioned earlier, besides optimising RAN resources, those of the Edge/MEC domain are also considered. In this section, we discuss location aware caching strategies, which are used at the application layer to perform eviction decisions of sensing data.

4.4.1 Problem Description

The location-aware caching strategy is to be employed in mainly two use cases, one pertaining object tracking in the context of wireless sensing, and one for MEC application data related to ISAC applications.

4.4.1.1 Wireless Sensing Use Case

The objective of the sensing use case is the continuous monitoring of a set of targets moving within a given area. This monitoring paradigm applies to a wide range of tracking applications, including wildlife monitoring, asset tracking, and mobile object surveillance, where information about target location or status must be collected and made available to a central entity.

In this context, sensing devices distributed over the field generate data associated with one or more targets, while a BS periodically issues queries to retrieve the most recent information available. Due to the mobility of the targets and the limited storage and communication capabilities of edge devices, it is not always possible to guarantee that the requested data is locally available at the time of the query.

We consider an optimisation problem in terms of HIT and MISS rates. A HIT event occurs when a request from the BS for a data from a particular sensor is satisfied, otherwise, it is considered a MISS. In addition, the following KPIs are considered:

1. **Energy Consumption**: it is measured as the amount of energy consumed in terms of sensor activations, i.e., how many times sensors are activated by a given caching policy.
2. **End-to-end Delay**: it measures the average time it takes to satisfy BS requests, given standardised times on tasks such as data retrieval and cache interrogation.

4.4.1.2 MEC Application Use Case

In this use case, we aim to enhance the performance experienced by end-user devices operating on a field, by leveraging edge computing caching mechanisms. With this objective, we not only aim to reduce latency through edge computing, but to improve the cache eviction strategies at the MEC layer under dynamic wireless and mobility conditions. While conventional MEC solutions rely mostly on content popularity, they often overlook the real-time state of wireless channels and the mobility of user devices.

We demonstrate how location-aware caching, enabled by sensing information exposed through the ISAC libraries, can be exploited to adapt cache contents to the current communication context. Specifically, the proposed approach leverages channel-quality indicators and proximity information to prioritize the storage of data that is more likely to be accessed by nearby users, thereby reducing cache MISS events.

The primary performance metric considered in this use case is the MISS RATE, defined as the fraction of requests that cannot be satisfied by cached data at the edge. The evaluation focuses on analyzing how the proposed Closest In, Farthest Out (CIFO) caching strategy performs under different system configurations and operating conditions by varying multiple parameters. A lower MISS rate directly reflects improved caching efficiency and better application-level performance.

Algorithm 1: CIFO Caching Algorithm

1 $C_e \leftarrow$ cache table of device e;
2 $r^t \leftarrow$ latest request from BS;
3 $S \leftarrow$ set of sensors on the field;
4 Event: Data received e_s^t
5 **if** $s \in C_e$ **then**
6 update entry for sensor s;
7 **else**
8 **if** C_e *is full* **then**
9 $s_{farthest} \leftarrow \arg\max_{s \in C_e} \|\mathbf{s} - \mathbf{r}^t\|$;
10 $C_e = C_e \setminus s_{farthest}$;
11 $C_e = C_e \cup e_s^t$;
12 Event: Request received $r^{t_{last}}$
13 $r^t \leftarrow r^{t_{last}}$;
14 $s_{closest} \leftarrow \arg\min_{s \in S} \|\mathbf{s} - \mathbf{g}\|$;
15 **if** $s_{closest} \in C_e$ **then**
16 return $s_{closest}$;
17 **else**
18 request sensing task to $s_{closest}$;

4.4.2 Algorithm Description

We present the CIFO strategy in Algorithm 1. The algorithm is run by a caching device $\boldsymbol{e}$, which keeps in memory its caching table C_e, the latest request from the BS r^t, and a view of the sensors on the field $\boldsymbol{S}$. The execution of the policy is driven mainly by two events:

- The first event is triggered when a sensor broadcasts new data (line 4). This broadcast could either be triggered by a passive sensing activity by the sensor or actively by an edge device previously requesting the sensing data. When the data is received, the edge device checks if it already has old sensing data for that sensor (line 5) and updates it accordingly. If the data is not present, it means that a new entry for it must be generated. If the cache is full, the device first finds the farthest sensing data from the latest queried position r^t (line 9) and evicts it.
- The second event is triggered when the edge device receives a query request from the BS (line 12). When it happens, the edge device first updates its last queried position and then verifies which sensor data is needed to satisfy the BS's request, which is given by the closest sensor (line 14). If the data for that sensor is present in its memory C_e, then it is promptly returned; otherwise, a sensing task is requested to that sensor.

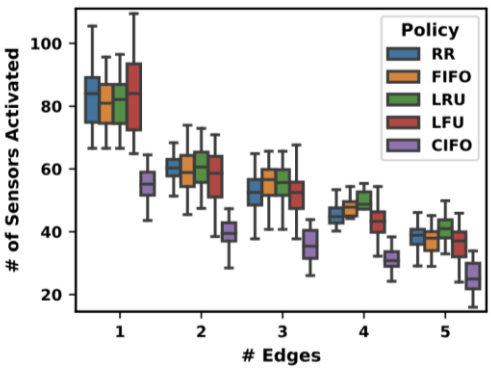
4.4.3 Evaluation

We analyze in this section the performance of our proposed strategy, both for the sensing use case and the MEC application use case. We tested the CIFO strategy against traditional caching strategies, including Round Robin (RR), First-In First-Out (FIFO), Least Recently Used (LRU), and Least Frequently Used (LFU).

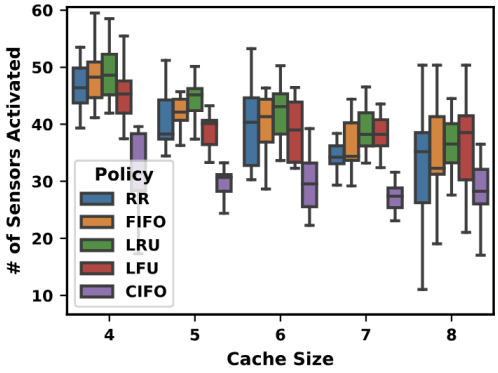
4.4.3.1 Sensing Use Case

We analyze the performance of all strategies first in terms of energy consumption. Energy is computed based on the number of times sensing devices are activated, either through passive sensing or active requests triggered by a cache MISS from the base station.

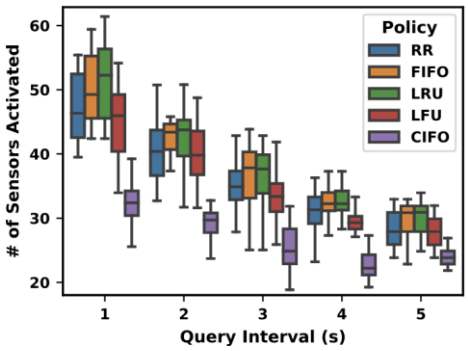
The results in Figure 4-11 show that the CIFO caching strategy consistently achieves lower energy consumption across all evaluation settings, reducing sensor activation by roughly 30% compared to other policies. The average end-to-end delay is evaluated by assigning a 1-sec delay to cache HIT events and a 10-sec delay to MISS events, reflecting the higher latency of MISS events. Figure 4-12 shows that, across all configurations, CIFO consistently achieves the lowest end-to-end delay, mirroring the trends observed in the energy consumption evaluation.



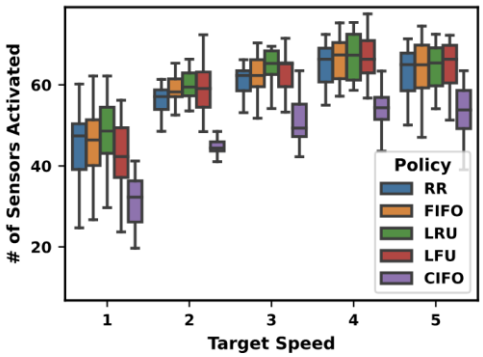
(a) Number of Edge Servers



(b) Cache Size

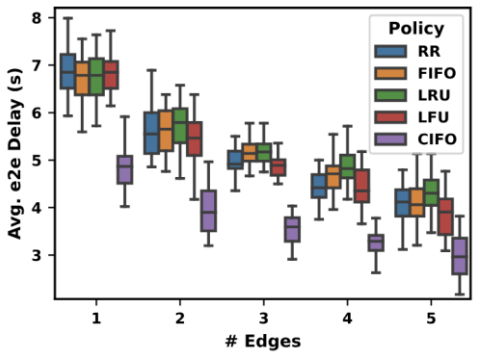


(c) Query Interval

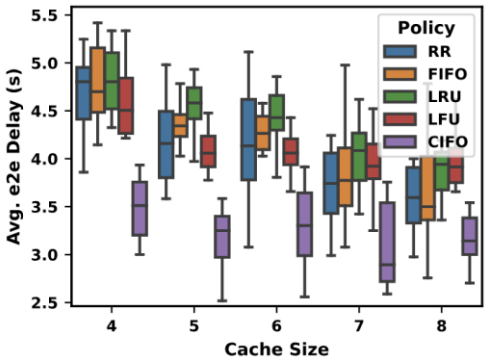


(d) Target Speed

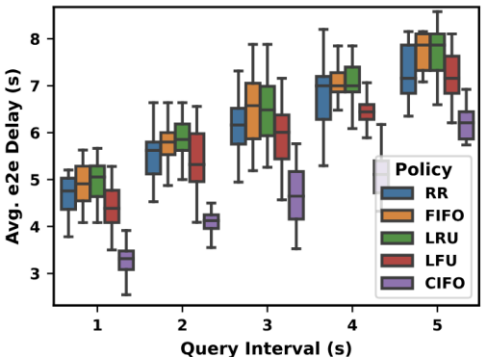
Figure 4-11 Sensing Application, Evaluation Results – Energy Consumption.



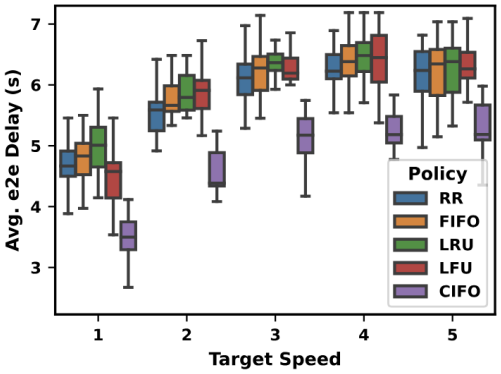
(a) Number of Edge Servers



(b) Cache Size



(c) Query Interval



(d) Target Speed

Figure 4-12 Sensing Application, Evaluation Results – End-to-end Delay.

4.4.3.2 MEC Application Use Case

In this section, we present the results of the simulation experiments for the MEC application Use Case.

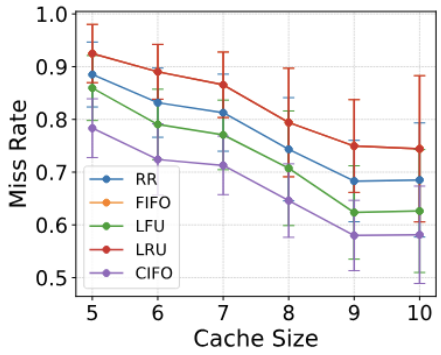
For each test, we deployed a baseline scenario consisting of 3 edge devices, 10 antenna relays, and 30 users, randomly placed in a 2-by-2 km field. Starting from this baseline, we varied the following parameters for each set of tests: cache size, number of users, number of edge devices, number of antenna relays, user speed, and field size. The results of these tests are shown in Figure 4-13.

Across all settings, CIFO outperforms the traditional caching strategies, particularly in scenarios where the amount of storage is limited relative to the number of users in the field. For example, setups with a small number of cache edge devices (.c) or small cache size (.a) demonstrate the most significant performance improvements.

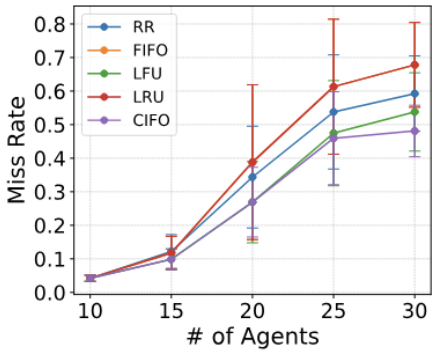
From Figure 4-13(c), it is noticeable that increasing the number of edge devices slightly decreases performance. This occurs because having multiple edge devices increases the likelihood that users move from one assigned edge device to another, which can trigger a MISS event. Such situations could be mitigated by predicting user movement and proactively moving data to the new device, but current caching strategies do not implement this behavior. This may be subject to future research leveraging the work carried out on the ISAC front.

Increasing the number of antenna relays Figure 4-13(d) leads to results with smaller standard deviations and more predictable performance. This is likely due to improved coverage of the field and a more balanced assignment of relays to edge devices.

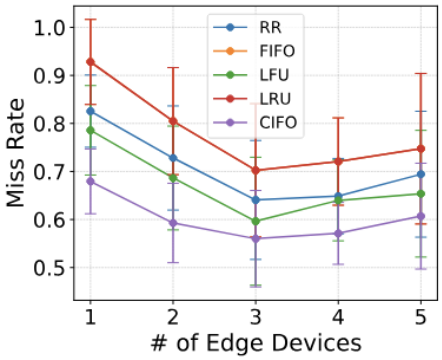
Interestingly, changing the size of the covered field or the speed of the users does not significantly affect the results. Only small variations are observed, with slightly worse performance for faster-moving users. This is explained by the fact that, even in these scenarios, users tend to switch more frequently between relays, often changing their designated edge device.



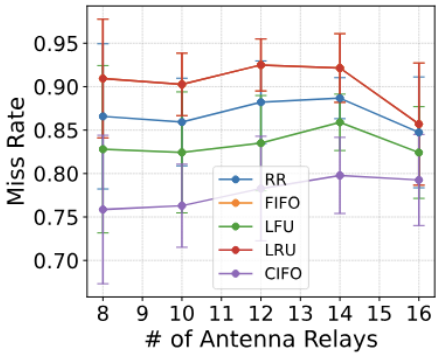
a) Variable Cache Size



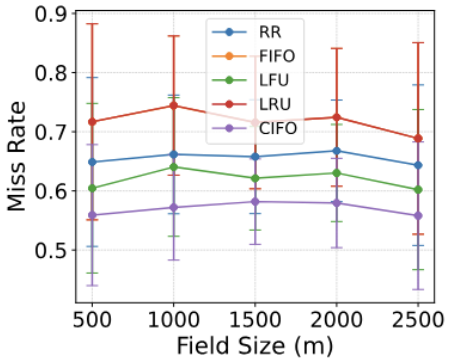
b) Variable Number of Agents



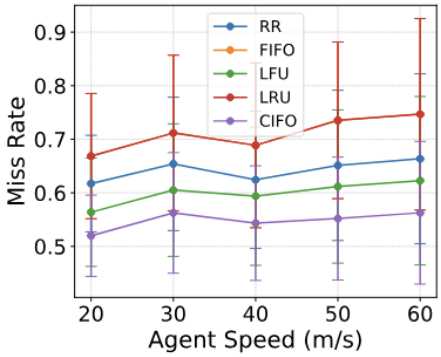
c) Variable Number of Edge Devices



d) Variable Number of Antenna Relays



e) Variable Field Size



f) Variable Agent Speed

Figure 4-13 MEC Application, Evaluation Results in terms of MISS rate.

5 AI-Driven End-to-End Network Intelligence and Sensing

This chapter presents AI/ML-based approaches that enable system-level intelligence within the 6G-SENSES architecture through predictive modeling, learning-based decision-making, and sensing capabilities. In contrast to the previous chapter, which focused on resource-level management and control within the RAN and Edge domains, the contributions presented here operate at a higher level of abstraction, addressing the modeling and optimisation of E2E network behavior, as well as the extraction of contextual information from wireless signals. These solutions support data-driven and adaptive end-to-end network operation.

Section 5.1 proposes a Graph Neural Network (GNN)-based framework for E2E performance prediction in dynamic RAN environments, where static configurations fail to ensure QoS compliance. By capturing the topological dependencies of the network, the model accurately estimates key performance metrics such as delay, jitter, and packet loss under dynamic and heterogeneous traffic conditions. The framework is evaluated using ns-3 under diverse scenarios, including heterogeneous traffic (sensing, fronthaul, backhaul), multiple scheduling configurations, and varying network load conditions. The results demonstrate that the proposed approach provides a scalable and efficient alternative to simulation-based methods, making it suitable for supporting network optimisation and orchestration mechanisms.

Section 5.2 addresses E2E slice orchestration for ISAC services using DRL. Building upon prior work (**D2.3** [2]), a hierarchical action decomposition scheme is introduced to improve sample efficiency by exploiting the structure of the action space. The evaluation shows that DQN-based approaches significantly outperform both tabular methods and myopic baselines in terms of convergence speed and quality of obtained policy. These results confirm the effectiveness of DRL in optimising both slice-level and network-level performance. Moreover, the proposed Hierarchical DQN achieves approximately 33% higher sample efficiency compared to vanilla DQN. Overall, this contribution demonstrates the benefits of combining deep function approximation with hierarchical decision-making for efficient slice orchestration.

Section 5.3 introduces a Generative AI (GenAI)-driven approach for automated RAN slicing, leveraging a Large Language Model (LLM)-powered AutoRAN rApp to enable intent-based network control. In this framework, high-level operator intents are translated into actionable network policies, which are deployed through the O-RAN architecture via the SMO, Non-RT RIC, and Near-RT RIC, enabling end-to-end slice configuration and real-time closed-loop control. The solution supports heterogeneous slice requirements (e.g., low-latency and high-throughput services), significantly reducing operational complexity and manual intervention.

Section 5.4 presents an adaptive sensing and orchestration framework that combines centralised sensing and Federated Learning (FL)-based distributed processing for trajectory prediction using range–Doppler radar data. The proposed approach dynamically selects between centralised and FL-based sensing modes based on network conditions, sensing requirements, and available resources. In centralised mode, multi-node sensing data are fused to achieve high trajectory-prediction accuracy, while in FL mode, local models are trained at distributed sensing nodes and only model updates are exchanged, reducing fronthaul and backhaul load and improving scalability and privacy. The framework operates across the RAN, transport, and compute domains, enabling coordinated resource allocation and proactive, sensing-aware optimisation of ISAC services. Finally, the WiRD-Gest framework presented in Section 5.5 demonstrates the potential of wireless sensing for application-level inference. The proposed framework is a Wi-Fi-based gesture recognition system that operates on COTS hardware using a monostatic full-duplex sensing pipeline. By transforming raw Wi-Fi signals into high-quality Range-Doppler representations and leveraging Deep Learning models, it enables accurate gesture classification. Multiple Deep Learning architectures were trained and evaluated, with the results showing that hybrid Convolutional Neural Network (CNN)-Recurrent Neural Network (RNN) architectures achieve the best performance, reaching an accuracy of 95.18%. This contribution highlights

how wireless signals can be exploited not only for communication and network optimisation, but also for extracting high-level contextual information about the environment.

5.1 GNN-based E2E Performance Prediction in RAN

5.1.1 Problem Description

Future RAN deployments will support heterogeneous traffic flows over a shared transport infrastructure, including sensing traffic, Open Fronthaul (O-FH), and backhaul services. These traffic classes exhibit diverse QoS requirements in terms of latency, jitter, packet loss, reliability, and timeliness.

In dynamic environments where traffic profiles evolve over time, static network configurations are insufficient to guarantee continuous QoS compliance. This limitation is particularly critical for sensing-related services, where data freshness constraints impose strict upper bounds on E2E delay.

To address this challenge, accurate and scalable performance prediction mechanisms are required to estimate network behavior under varying and previously unseen traffic conditions. The objective we seek herewith is to assess the capability of a GNN-based model to generalize and provide reliable E2E performance predictions across diverse RAN scenarios to be afterwards leveraged by different controllers, which might include techniques to optimize network performance.

5.1.2 Algorithm Description

The proposed solution is based on a data-driven performance modeling framework built upon a GNN architecture, specifically RouteNet-Fermi [11]. The model is designed to estimate key E2E network performance metrics, including delay, jitter, and packet loss ratio. By leveraging the inherent graph structure of network systems, the GNN captures topological dependencies and interactions among network elements, enabling scalable inference without the need for computationally expensive simulations during runtime.

To ensure accuracy and robustness, the training dataset is generated using *ns-3* as a ground-truth network simulator. Besides, an additional framework has been integrated as a middleware layer to accurately capture the behavior of the open fronthaul interface. This framework allows configuration of each flow according to 5G New Radio (NR) parameters, including duplexing mode TDD/Frequency Division Duplex (FDD), Subcarrier Spacing (SCS), bandwidth (BW), and Radio Resource Control (RRC) configuration, as can be seen in Figure 5-1.

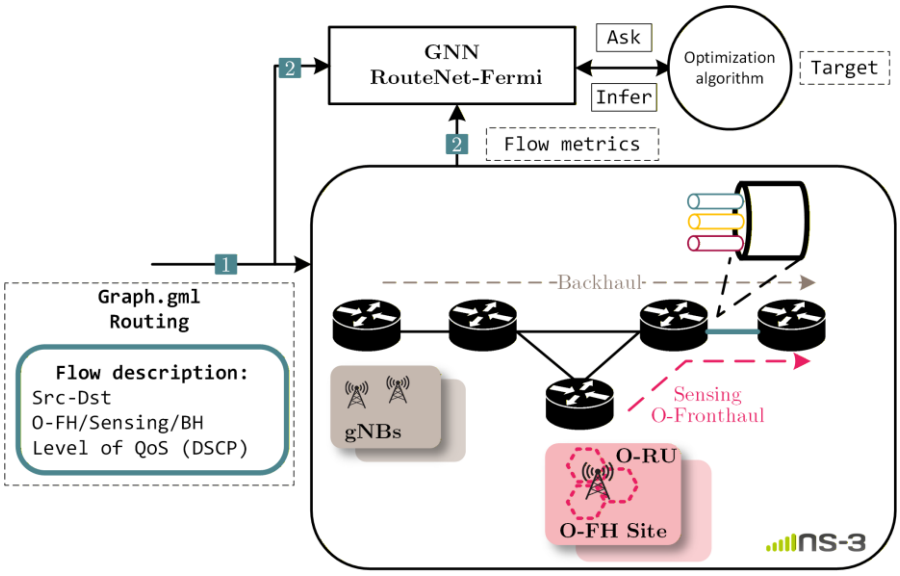


Figure 5-1 Workflow of the GNN-based network modeling architecture

The process initiates with the network representation as a graph (1), defined by the topology, the routing configuration, and the flow descriptions, which are specified using a JSON format. This input is integrated with real-time flow metrics from the *ns-3* simulator (2) to feed the RouteNet-Fermi GNN, enabling performance inference.

For this purpose, we assume that sensing traffic is generated by O-RUs operating in TDD mode. This sensing scheme adopts an active monostatic solution where the O-RU gathers reflections of its own transmissions and forwards the raw I/Q samples over the O-FH for centralised processing. The generation of this traffic is strictly coupled with the 5G NR frame structure, as reflections are obtained during dedicated downlink slots or symbols using standard waveforms, or during flexible symbols using tailored waveforms. These parameters, together with flow metrics from the *ns-3* simulator (2), feed the GNN to enable performance inference and iterative optimisation against defined targets.

To align the simulation with realistic 3GPP RRC configurations, the sensing throughput is calculated based on the subcarrier spacing (SCS) $\Delta f=15\cdot 2^{\mu}kHz$ and the total number of physical resource blocks N_{RB} corresponding to the configured bandwidth. Specifically, our implementation uses a RRC mapping where a specific number of OFDM symbols (up to 14 per slot) within the transmission periodicity (i.e., `dl-UL-TransmissionPeriodicity`) are dedicated to sensing tasks. On the O-FH interface, the packet length is determined by the PRB size, $PRB_{\text{size}}=2\cdot data_w\cdot N_{subcs}+8$, which, under the Block Floating Point (BFP9) compression scheme, accounts for 9-bit data resolution ($data_w$) across 12 subcarriers. Packets of size $L_{pkt}=N_{RB}\cdot PRB_{\text{size}}+H_{\text{overhead}}$) are generated at the symbol rate, while the packet arrival rate can be expressed as $\lambda=\frac{1}{T_{symbol}}$. Therefore, the resulting throughput can be computed as $R_{bound}=\lambda\cdot L_{pkt}\cdot 8$, expressed in bits per second.

To precisely capture the `dl-UL-TransmissionPeriodicity`, the total active time within a repeating periodicity (T_{period}) is calculated by aggregating the duration of all individual symbols allocated for sensing across all active slots, as can be seen in Figure 5-2. This is formally defined as:

$$T_{\text{ON}}=\frac{\sum_{j=1}^{N_{\text{slots}}}\left(\sum_{i\in S_j}T_{\text{symb},i}\right)}{T_{\text{period}}}.$$

Finally, the average sensing data rate is calculated as $R_{sensing}=R_{bound}\cdot T_{ON}$. Furthermore, we consider that the network capacity has been dimensioned to accommodate all flows associated with each slice, under the assumption of full resource utilisation.

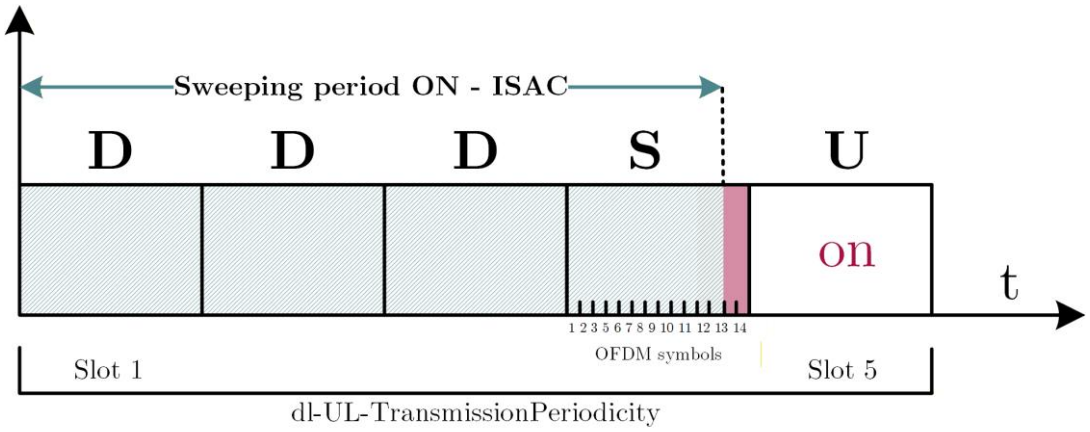


Figure 5-2 Slot-level TDD configuration highlighting the ISAC Sweeping Period and Symbol-level Special Slot (S) Allocation.

To realistically model network behavior, the system performance is assessed under heterogeneous and competing traffic conditions, including 3GPP 5G NR O-FH U-Plane traffic (in both TDD and FDD modes), sensing traffic, and backhaul traffic, particularly URLLC. In the case of sensing flows, it is assumed that the DL communication signal is reused for sensing purposes, and the resulting data is transported across the network as an additional traffic flow. Each traffic flow belonging to a QoS profile will be marked in the Type of Service (ToS) field so that a priority level can be assigned using scheduling policies at the intermediate nodes of the network.

The final stage of the framework maps the time-varying traffic dynamics into a structured representation suitable for GNN processing. This step ensures that the RouteNet-Fermi model captures the impact of 5G NR physical layer constraints on E2E network performance.

For each flow f in the set of flows F, a feature vector is constructed to represent its key attributes, including the offered traffic load, packet generation rate, and traffic model parameters. On the other hand, link and queue elements are also represented through dedicated feature vectors. Link features do not only capture network capacity, but also the scheduling policy, while queue features encode information such as buffer size, priority and weight under the corresponding scheduling policy. In addition, the relation $f \rightarrow Q$ is done by mapping the ToS to a queue priority level.

The ToS value, derived from the combination of the Differentiated Services Code Point (DSCP) and Explicit Congestion Notification (ECN) fields at the IP layer, is used to assign each flow to a corresponding queue under a Strict Priority (SP) scheduler. The mapping from flow priorities to queues is formally defined by the function:

$$\pi: d_f \rightarrow q_f, q_f \in \{1,2,3\},$$

where d_f denotes the DSCP label associated with flow f and q_f represents the queue to which the flow is assigned under a strict-priority (SP) scheduling policy. The combination of all flow-to-queue assignments across the network defines a scheduling configuration: $s \in S, |S|=11$, where S denotes the set of all admissible mappings of flows to the three SP queues, including the FIFO configuration as a particular case.

The GNN is designed to model the complex behavior of network infrastructures by means of a three-part graph approach. This allows the model to distinguish between the various functional elements of a network within the graph, $G=(V, E)$. The set of vertices V is not homogeneous, but it is categorised into three distinct types of entities, each representing a specific part of the network stack: (i) v_l represents the capacity (or load, in an indirect way) of an interface, but also the scheduling policy; (ii) v_q represents the buffer size, the weights assigned to each queue (if a weighted scheduling policy is applied) and the ToS assigned to each queue to define the flow-to-queue mapping; finally (iii) v_p corresponds to the traffic model, incorporating all features regarding traffic behavior, specifically inter-arrival distribution. The set of edges E defines the relationships among these entities, reflecting the structural dependencies of the network itself.

The architecture follows a Message Passing Neural Network (MPNN), where V nodes represent different entities of the network: flows, links and queues; and E represents the relationship among them. Furthermore, the interaction of the different layers has been done by means of Gated Recurrent Units (GRU), which act as the update function to evolve the hidden states of each entity during the message-passing phases. Specifically, the model is configured with $T=8$, where T represents both the number of iterations in the MPNN algorithm and the depth (layers) of the model. During the message-passing phase, each node $v \in G$ is associated with a hidden state h_v^t, represented as a fixed-size vector that encodes relevant information about the node at iteration t. These hidden states are iteratively refined by exchanging messages with neighboring

nodes. At each iteration, node v receives messages from its neighbors $w \in N(v)$, which are combined through an aggregation function. The resulting message is computed as:

$$m_v^{t+1} = \sum_{w \in N(v)} M_t(h_v^t, h_w^t, e_{vw}),$$

where $M_t(\cdot)$ denotes the message function at iteration$t$, and e_{vw} represents the features associated with the edge between nodes v and w.

The hidden state of node v is then updated using an update function $U_t(\cdot)$, which incorporates both the previous state and the aggregated message:

$$h_v^{t+1} = U_t(h_v^t, m_v^{t+1}).$$

This process is repeated for a fixed number of iterations, allowing information to propagate across the graph and enabling each node to capture both local and global structural dependencies.

Finally, in the readout phase, the output of the model is computed by applying a readout function $R(\cdot)$to the final hidden states after T iterations:

$$\hat{y} = R(\{h_v^T \mid v \in G\}).$$

To sum up, in the model we can distinguish three phases: initialisation (embedding phase), message-passing (as previously depicted), and readout. Each node type (path, link, queue) is initialised with a feature vector derived from input parameters (e.g., traffic statistics, link capacity, queue size, scheduling policy), as can be seen in Figure 5-3. These features are processed using a Multilayer Perceptron (MLP) composed of two dense layers with Rectified Linear Unit (ReLU) activation. This produces an initial hidden state of 32 dimensions for each node. Subsequently, a message passing phase starts. During T iterations nodes exchange information with their neighbors, following the graph structure: (i) paths receive information from links and queues; (ii) queues aggregate information from paths, and finally (iii) links update their state based on queues. These interactions are explicitly defined through the relationships encoded in the adjacency lists shown in Figure Figure 5-3, which determine how information flows across the graph. During this phase, message computation, aggregation and state update operations are applied, as defined by the expressions above.

The aggregation of incoming messages is performed either explicitly, via summation in the case of queues, or implicitly, through Gated Recurrent Units (GRUs), which process sequences of incoming messages for paths and links. These GRU-based updates enable the model to capture complex dependencies and iteratively enhance node representations.

Finally, in the readout phase, the refined hidden states of the path nodes are processed through an additional MLP (readout function) to estimate the target metric. Specifically, the model predicts the end-to-end delay, which includes both queueing delay and transmission delay, computed based on the inferred network state.

This end-to-end integration enables the proposed solution to replace computationally expensive ns-3 simulations with millisecond-scale GNN inference, while maintaining high accuracy across various SCS, BW and RRC periodicity configurations, under different network setups. Furthermore, this approach facilitates seamless integration with optimisation frameworks, enabling efficient exploration of the configuration space and quick identification of optimal system parameters.

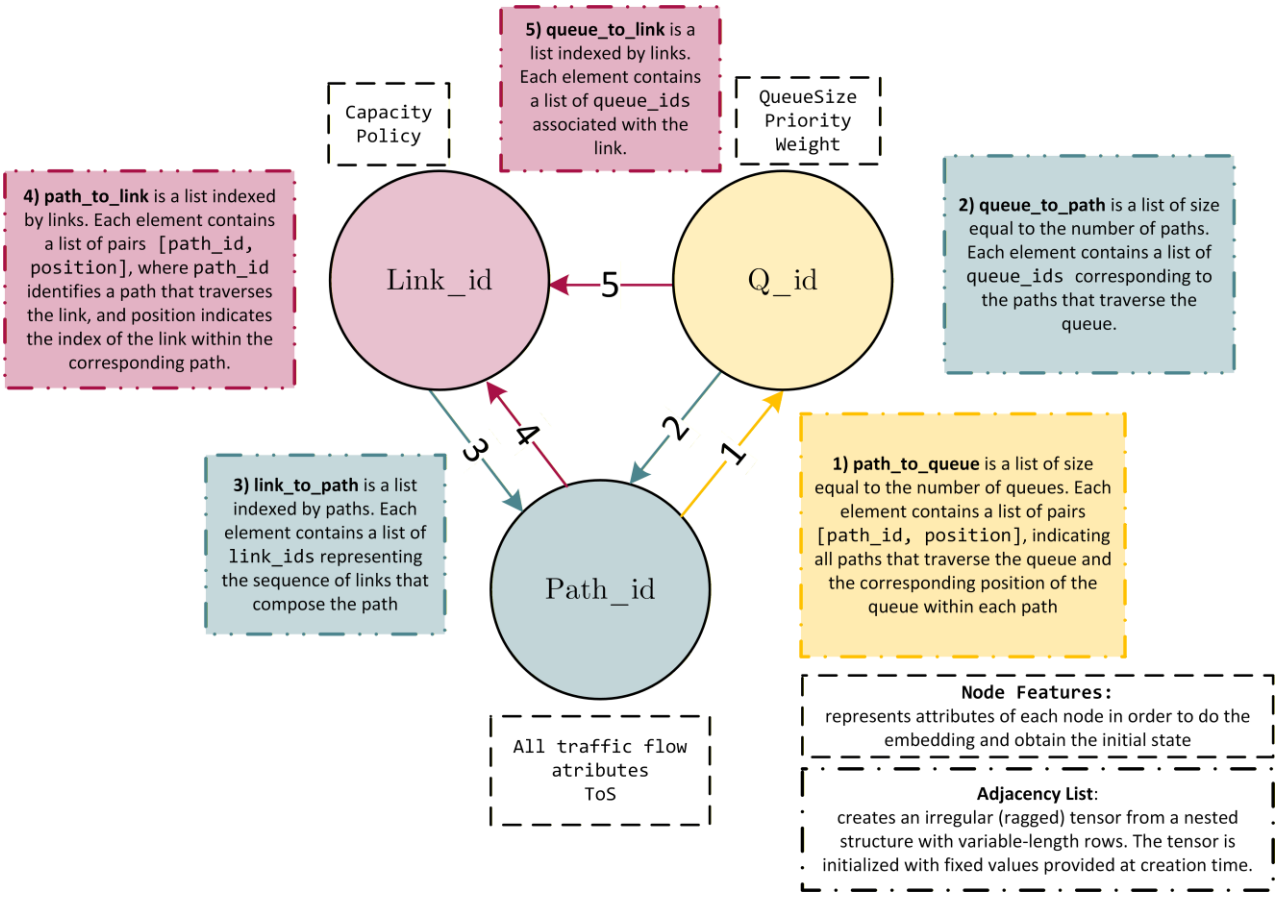


Figure 5-3 Architecture of the Heterogeneous Tripartite GNN: Integration of node-specific features (Paths, Queues, Links) and topological dependencies via adjacency lists

5.1.3 Evaluation

The framework is evaluated through a simulation-based study using *ns-3* as the reference network simulator. The evaluation considers heterogeneous traffic scenarios combining sensing, O-FH, and backhaul flows over a shared transport topology.

The assessment focuses on the prediction accuracy of the GNN model under both seen and unseen conditions. In particular, the analysis assesses the accuracy of end-to-end delay predictions, as well as the robustness of the model across different scheduling configurations and link capacity assignments.

The network topology considered during the evaluation, is illustrated in Figure 5-4. Link capacities are assigned based on the cell site deployment corresponding to split 7.2x and monolithic architectures, assuming full-capacity operation of the sites. The data rates associated with each slice are summarised in Table 5-1 and are evaluated considering UL traffic streams.

In particular, the TDD configuration is evaluated using a DL-UL transmission periodicity of 2.5ms, following a slot pattern of DDDSU. This allows modeling realistic time-domain resource allocation between downlink (DL), uplink (UL) and switching intervals.

Furthermore, the sensing slice is evaluated under different resource allocations by sweeping the number of sensing slots from 1 to 3 full slots. In the latter case, the allocation also includes the special symbols corresponding to downlink transmission and guard periods. Overall, this parameterisation results in a total of 4 distinct configurations.

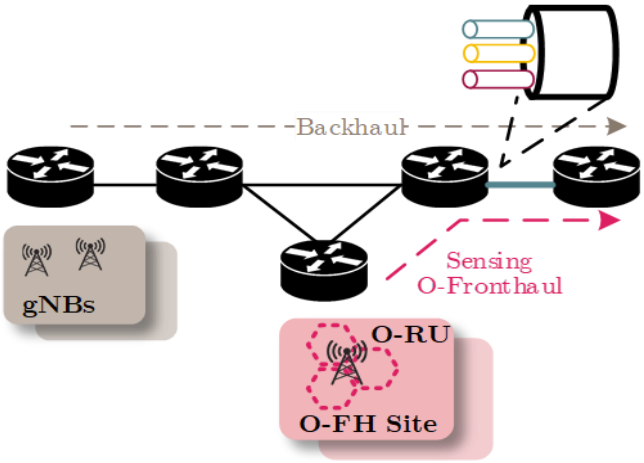


Figure 5-4 Network topology used for the evaluation of the proposed framework.

On the other hand, network performance is assessed under different load conditions, with a link utilisation ranging from 10% to 95%. This allows studying the model performance across both underutilised and near-saturation regimes.

In addition, different ToS flow-to-queue mappings are considered to represent heterogeneous service requirements. In particular, three traffic slices are defined, namely O-FH traffic, sensing traffic and backhaul traffic. These slices are mapped to a set of strict-priority (SP) queues, resulting in a total of 11 possible scheduling configurations. This setup enables the evaluation of the model under diverse priority assignments and service differentiation policies.

Table 5-1 Radio Configuration Parameters and Traffic Profile Characterisation.

Site Deployment	Features	Datarate (Mbps) / Traffic Model
O-FH Site	Band 1: FDD BW 20 MHz, SCS 15 kHz	288.88 ON-OFF
	Band 2: TDD BW 40 MHz, SCS 30 kHz (**ISAC**)	576.80 Deterministic
Backhaul	URLLC – Discrete Automation	833.33 Poisson Arrival Process

Figure 5-5 shows the Mean Absolute Error (MAE) under different network utilisation levels, focusing on the shared link. The vertical axis represents the different policy mappings evaluated, while the horizontal axis corresponds to the utilisation regimes (low, medium, and high). The color intensity in the heatmap encodes the MAE values, enabling a visual comparison of the model's prediction accuracy across different configurations. In addition, the queue mapping is defined by identifying three traffic classes, which are associated with distinct DSCP labels. Each incoming flow is classified according to its DSCP marking and mapped to the corresponding queue within the scheduler, ensuring differentiated treatment according to the underlying scheduling policy. As can be seen in the figure, when the network operates under high utilisation (>85%), the model yields a more relevant deviation from the ground-truth values, resulting in errors of up to 11 µs.

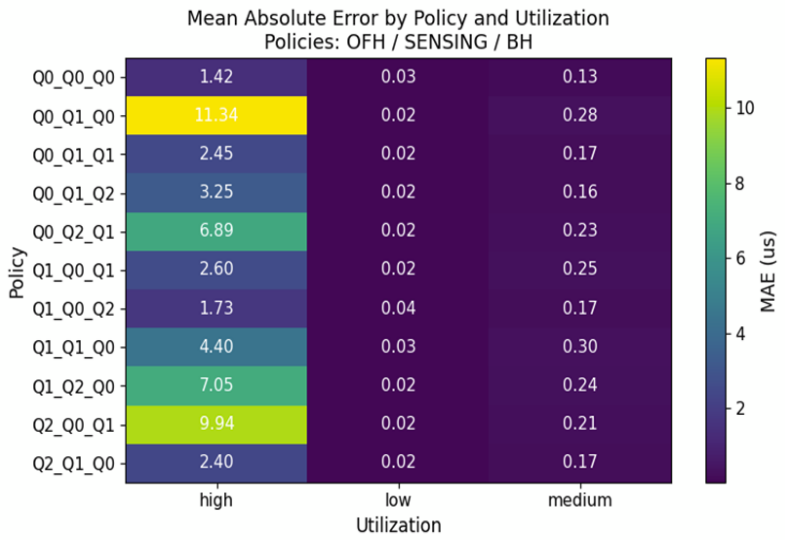


Figure 5-5 Heatmap of MAE for E2E delay prediction across different scheduling policies and network utilisation levels

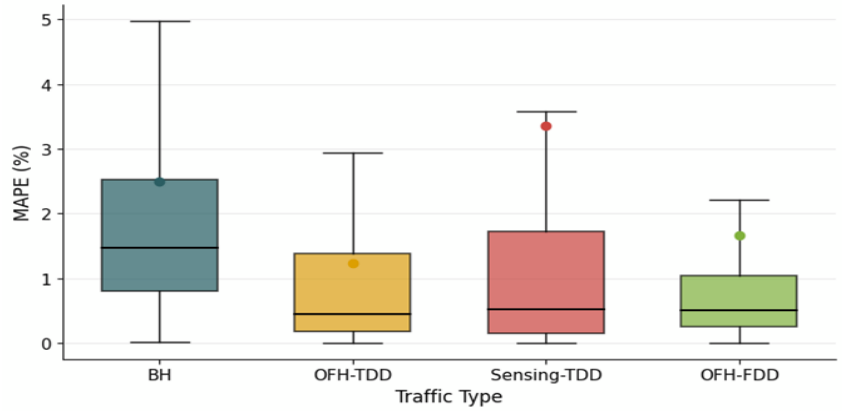


Figure 5-6 Mean Absolute Percentage Error (MAPE) distribution per traffic type.

Figure 5-6 illustrates the Mean Absolute Percentage Error (MAPE) per traffic flow for different traffic types, namely BH, OFH-TDD, Sensing-TDD, and OFH-FDD. The boxplots summarize the distribution of the prediction error across multiple scenarios, considering variations in network utilisation, scheduling policies, and sensing traffic load.

As can be seen, BH traffic exhibits the highest variability, as well as generally higher error values, evincing more challenging prediction conditions. OFH-TDD shows lower median errors with a relatively tighter distribution, suggesting a more stable performance. Sensing-TDD shows a moderate error level, but with wider spread and higher upper whiskers, indicating that certain configurations might yield more relevant deviations. Finally, OFH-FDD achieves the lowest median error among the considered traffic types, with a tighter distribution, corresponding to more consistent and accurate predictions. Overall, the results highlight how the model's accuracy varies depending on the traffic type and its associated network configuration.

To sum up, the results demonstrate the effectiveness of the GNN-based approach in providing reliable performance estimates, enabling its use as a scalable alternative to traditional simulation-based methods.

5.2 DRL-Based E2E Network Slicing in support of ISAC services

The network slicing paradigm enables flexible deployment of services tailored to the diverse QoS requirements of different verticals [7]. It allows multiple logically independent end-to-end virtual networks, known as slices, to coexist on top of a shared physical infrastructure. Each slice is associated with specific Service Level Agreements (SLAs).

In deliverable D2.3 [2], a DRL approach for the orchestration of E2E communication and sensing slices was proposed and evaluated. Specifically, a Deep Q-Network (DQN) agent was employed and its performance was compared against a set of carefully designed static baseline policies, demonstrating significant performance gains.

In this section, we extend this work along two main directions. First, to improve sample efficiency, we introduce a hierarchical action decomposition scheme that exploits the structure of the action space, effectively reducing its dimensionality and enhancing exploration. Second, we provide a more comprehensive evaluation by comparing DQN-based approaches (both hierarchical and vanilla) against tabular Q-learning, as well as against a myopic baseline that selects the optimal action assuming static traffic conditions in the next time slot. This comprehensive evaluation enables the assessment of (i) the benefits of function approximation via DNNs, and (ii) the performance gains achieved through learning-based decision-making under dynamic traffic conditions.

5.2.1 Problem Description

We consider dynamic orchestration of end-to-end network slices supporting communication and sensing services over a shared physical network infrastructure. A detailed description of the underlying system model, including the physical network, traffic demands, and KPI computation, is provided in D2.3 [2], while Figure 5-7 depicts an overview. In this section, we briefly recap the key elements of the RL problem formulation that are essential to motivate the algorithmic choices presented in the following subsection.

A network slice is modeled as a Virtual Network Function (VNF) chain associated with specific E2E SLAs. The throughput requirements of each slice are time-varying, depending on user-generated traffic and sensing conditions, while slices compete for shared compute and transport resources. To configure a slice, the orchestrator must:

- Embed its VNFs onto available physical servers across the network domains,
- Set any slice-specific parameters, such as the sensing Signal to Noise Ratio (SNR) threshold in the case of sensing slices, which directly affects both processing load and sensing accuracy.

Since slices share limited resources, configuration decisions for one slice affect the performance of others. Moreover, traffic demands and sensing conditions evolve stochastically and are not known in advance. The orchestrator must therefore continuously configure slices to balance energy efficiency, migration overhead, and the fulfilment of SLAs.

Considering an RL formulation of the dynamic slice orchestration problem, an RL agent interacts with the environment in discrete time slots. At each slot, the agent observes the current system state, selects an action (new configuration), and receives a reward reflecting network and slice performance. The objective is to learn a policy that minimises the expected long-term cumulative cost. The formulation of the RL problem relies in defining the State space, the Action space, and the Reward function.

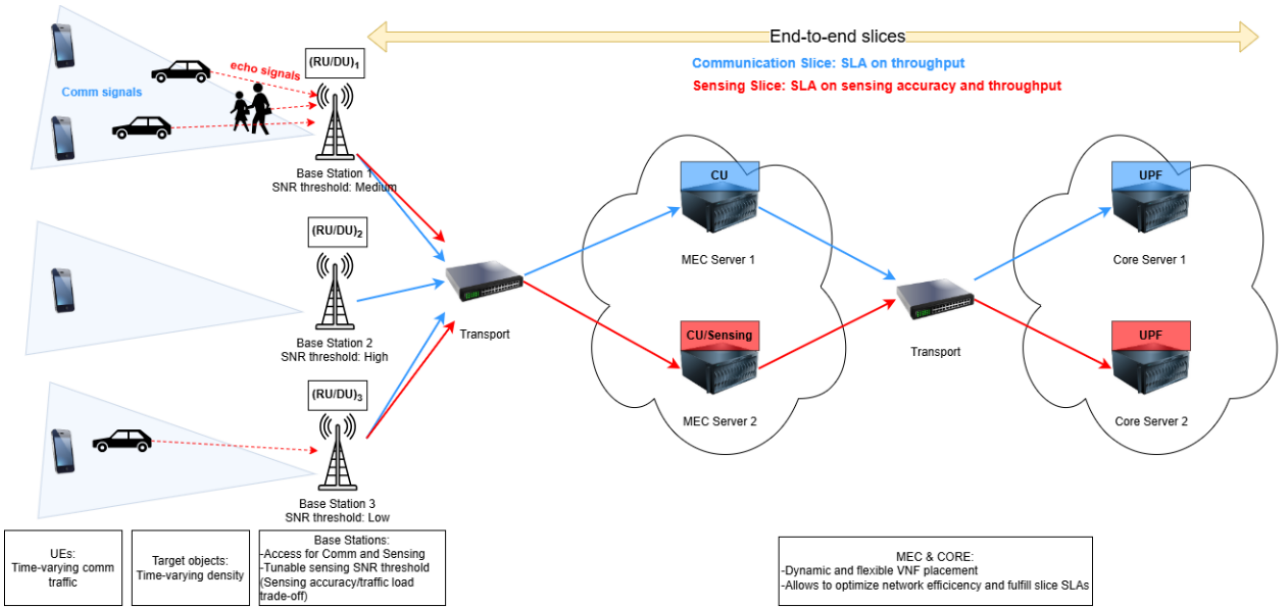


Figure 5-7 System model overview [2].

5.2.1.1 State Space (S)

At the beginning of each time slot t, the agent observes the system state s_t, which captures both load conditions and the current network configuration. More specifically, the state includes:

- the current throughput demand per RAN VNF of the communication slice, and the respective sensing-related indicators (e.g., object activity) that determine the sensing slice load (optionally, a short window of recent throughput and sensing measurements may be included),
- the current embedding of all VNFs onto physical servers across the MEC, and CN domains.

We stress here that if traffic demands, object activity, and sensing parameters are real-valued quantities, the state space is continuous. Even if we quantize the above quantities, the state space is high dimensional due to its combinatorial nature. More concretely, if we denote the number of nodes in domain d with N_d, the number of respective VNFs of slice k with M_d^k, and the number of quantisation levels of quantity q with L_q, then the number of states can be expressed as:

$$|S|=L_{object}{}^{M_{RAN}^{sens}} \times L_{throughput}{}^{M_{RAN}^{comm}} \times N_{MEC}{}^{M_{MEC}^{comm}+M_{MEC}^{sens}} \times N_{CORE}{}^{M_{CORE}^{comm}+M_{CORE}^{sens}}.$$

As an example, for the setup considered in this work

$$(N_{MEC}=N_{CORE}=2, M_{MEC}^{comm}=M_{MEC}^{sens}=M_{CORE}^{comm}=M_{CORE}^{sens}=1, M_{RAN}^{comm}=M_{RAN}^{sens}=3),$$

if we quantize the all the continuous quantities to 5 levels then the number of states is: $|S|=5^3 \times 5^3 \times 2^2 \times 2^2=250000$. Further increasing the number of quantisation levels to 10, leads to 16000000 possible states, which is already prohibitively large for tabular RL approaches.

5.2.1.2 Action Space (A)

At each time slot t, based on the observed state s_t, the agent selects an action a_t that specifies: i) a new placement of all VNFs onto physical servers (across MEC and CORE domains) at t+1, and ii) the sensing-related parameters, e.g. the per RAN VNF sensing SNR thresholds at t+1. Using the same notation as above, the total number of actions can be expressed by:

$$|A|=L_{SNR}{}^{M_{RAN}^{sens}} \times N_{MEC}{}^{M_{MEC}^{comm}+M_{MEC}^{sens}} \times N_{CORE}{}^{M_{CORE}^{comm}+M_{CORE}^{sens}}.$$

For the setup considered in this work and assuming 3 quantisation levels for the sensing SNR thresholds (low, medium, high), the total number of actions is:

$$|A|=3^3 \times 2^2 \times 2^2=432.$$

5.2.1.3 Reward

After applying action a_t, the traffic and sensing conditions at t+1 are revealed by the environment, and the agent receives a reward r_{t+1}. The reward is a feedback signal that shows how good the action was. Note that in our problem the reward is determined by the traffic and sensing conditions at t+1 that were unknown when the agent decided what action to take.

The reward is defined as the negative weighted sum of the following individual costs:

- **energy/resource cost (c_{energy})**, proportional to the number of active servers and transport elements,
- **VNF migration cost (c_{mig})**, penalising VNF relocations between consecutive time slots,
- **communication slice SLA violation penalties (c_{SLA}^{comm})**, capturing violations of the respective KPIs (e.g. throughput or delay constraints),
- **sensing slice SLA violation penalties (c_{SLA}^{sens})**, capturing violations of the respective KPIs (e.g. sensing accuracy, throughput, or delay, constraints.

The energy cost at timeslot t is given by:

$$\boldsymbol{c_{energy}} = \sum_{v \in V} \mathbf{1}_{\{v \in c(t)\}},$$

where V the set of servers, $c(t) = (c_n^k(t) | \forall k \in K, n \in N_k)$ denotes the placement of all VNFs to servers, with $c_n^k(t)$ indicating the host server of VNF n of slice k, and $\mathbf{1}_{\{.\}}$ is an indicator function equal to 1 when the condition is satisfied and 0 otherwise.

The migration cost at time t is given by:

$$\boldsymbol{c_{mig}} = \sum_{k \in K} \sum_{n \in N_k} \mathbf{1}_{\{c_n^k(t) \neq c_n^k(t+1)\}}.$$

The SLA violation cost if given by:

$$\boldsymbol{c_{SLA}^k} = \sum_{e \in KPIs} (\boldsymbol{\sigma_{k,e}} + |\boldsymbol{f_{k,e}}(\boldsymbol{s_t}) - \boldsymbol{q_{k,e}}|) \cdot \mathbf{1}_{\{\boldsymbol{f_{k,e}}(\boldsymbol{s_t}) < \boldsymbol{q_{k,e}}\}},$$

where $\boldsymbol{f_{k,e}}(\boldsymbol{s_t})$ a function that outputs the KPI value e of slice k based on the current system state and modeling assumptions (more details on the modeling of SLAs can be found in D2.3), $\boldsymbol{q_{k,e}}$ is the minimum allowed value of KPI e of slice k (i.e. minimum sensing accuracy or minimum throughput) based on the corresponding SLA, and $\boldsymbol{\sigma_{k,e}}$ is a fixed penalty paid whenever an SLA violation occurs.

Considering all the above, the reward at $t + 1$ is defined as:

$$r_{t+1} = -(w_1 \boldsymbol{c_{energy}} + w_2 \boldsymbol{c_{mig}} + w_3 \sum_{k \in \{comm,sens\}} \boldsymbol{c_{SLA}^k}),$$

where w_1 , w_2, w_3, are positive weighting factors that determine the relative importance of each cost component.

By structuring the reward in this way, the agent is encouraged to balance resource utilisation efficiency and SLA satisfaction for both communication and sensing services, avoiding unnecessary reconfigurations.

5.2.1.4 Objective

The goal of the agent is to learn a policy $\pi(a|s)$ that maximises the expected discounted cumulative reward:

$$max_{\pi} E_{\pi}\left[\sum_{t=0}^{\infty} \gamma^{t} r_{t+1}\right],$$

which is equivalent to minimising the long-term expected discounted cumulative cost. Importantly, future traffic demands and sensing conditions are unknown at decision time. Therefore, the problem constitutes a sequential stochastic control problem under uncertainty, where the agent must implicitly learn traffic patterns and system dynamics through interaction with the environment.

5.2.2 Algorithm Description

Having described the dynamic slice orchestration RL problem formulation, this section focuses on briefly outlining the most important characteristics of the (D)RL baselines (Q-Learning and DQN), as well as introducing the proposed hierarchical action decomposition mechanism on top of both approaches.

5.2.2.1 Q-Learning (QL)

One of the simplest and most widely applied approaches for solving an RL problem is "tabular" Q-learning (QL) [6]. A Q-learning agent maintains a table, called the Q-table, which contains one entry for each possible state-action pair. Each entry corresponds to an estimate of the action-value $Q(s,a)$ of the respective state and action (s,a). The table is initialised (e.g., to random values or to all zeros), and the agent iteratively refines these estimates through interaction with the environment.

At each time slot, the agent selects an action based on its current Q-table using an ϵ-greedy policy, meaning that, with probability 1-ϵ the agent exploits its current Q-value estimates (a_t=$argmax_a Q(s,a)$), and with probability ϵ, the agent explores (selects an action uniformly at random).

After applying the selected action a_t and observing the next state s_{t+1} and reward r_{t+1} received by the environment, the agent updates Q-estimate of the visited state-action pair using the standard temporal-difference update rule derived from the Bellman optimality equation:

$$Q(s_t,a_t) \leftarrow (1\text{-}\eta)Q(s_t,a_t)+\eta[r_{t+1}+\gamma \max_a Q(s_{t+1},a)],$$

where η is the learning rate and γ the discount factor.

While Q-learning is attractive due to its conceptual simplicity and theoretical convergence guarantees under ideal conditions, its application in the considered slice-orchestration problem faces several important limitations. We outline these limitations below.

Memory Requirements: The size of the Q-table scales as $|S| \times |A|$. From the state and action space complexity analysis of the previous section, it can be derived that even in the case we do a coarse quantisation of the continues-valued parameters in our setup (3-level quantisation of the sensing SNR threshold and 5-level quantisation of traffic demands and object densities), the Q-table size explodes: $|S| \times |A|$=$6750000 \times 432 \approx 10^8$.

Sample Efficiency: As discussed earlier, in QL only the Q-value corresponding to the visited s, a is updated at each time slot. Moreover, each state–action pair must be visited many times to obtain an accurate estimate and exploration of such a large state-action space is extremely slow. Therefore, convergence time also explodes with $|S| \times |A|$.

Quantisation of Continuous Traffic: Since traffic demands and object activity (or even sensing-related parameters) may be continuous-valued, while Q-learning is defined over discrete state spaces, manual quantisation of these quantities is required. This introduces the following challenges: i) the choice of the number of quantisation levels and quantisation type (uniform vs. non-uniform) is non-trivial; ii) fine-grained

quantisation is infeasible due to the exponential growth of $|S|$. Coarse quantisation may result to suboptimal policies and learning instabilities due to aliasing (different traffic conditions are mapped to the same state).

Lack of Convergence Guarantees in Practice: Although tabular Q-learning is theoretically guaranteed to converge to an optimal policy under certain assumptions, these conditions are not strictly satisfied in our setup. In particular: i) real traffic traces are not purely Markovian; ii) manual quantisation introduces approximation errors. Therefore, even if we did not have any memory, computing, and convergence time limitations, convergence to a truly optimal policy would not be guaranteed in our setup.

5.2.2.2 Deep Q-Network (DQN)

Many of the limitations of QL can be effectively addressed by employing a Deep Q-Network (DQN) [10], which was the approach evaluated in **D2.3** [2]. Instead of maintaining a Q-table, the DQN approximates the action-value function using a DNN $Q_\theta(s,a)$ with trainable parameters θ. The DNN takes as input the state and outputs the Q-value estimates of all possible actions.

The DQN agent follows an ϵ-greedy policy, similarly to the QL agent. An important difference to QL is that the agent now employs an experience replay buffer, where it stores each visited transition i (called experience) as a tuple (s_i, a_i, r_i, s_i'). Therefore, the network parameters are updated by randomly sampling a mini-batch from the replay buffer and applying a gradient step:

$$\theta \leftarrow \theta\text{-}\eta\nabla_\theta E_{\iota\sim U(B)}[\delta_i^2],$$

$$\delta_i = Q_\theta(s_i,a_i)\text{-}(r_i+\gamma \underset{a_i'\epsilon A}{max} Q_{\theta'}(s_i',a_i')),$$

where δ_i is the temporal-difference error, $\iota\sim U(B)$ denotes the experiences sampled uniformly at random from the replay buffer, and $Q_{\theta'}$(s,a) is the target network (an older copy of the network Q_θ(s,a) that is only periodically updated).

The main advantages over QL are summarised below.

Memory Requirements: In DQN, the Q-function is represented by a neural network with a fixed number of parameters, independent of $|S|$. This eliminates the need to store $|S| \times |A|$entries and makes the approach scalable to high-dimensional and continuous state spaces.

Sample Efficiency: Through function approximation, updates performed for a given state generalize to similar states. Moreover, experience replay allows multiple gradient updates from the same transition, significantly improving data efficiency compared to tabular Q-learning.

Automated Feature Extraction: Unlike tabular Q-learning, DQN can directly operate on real-valued traffic demands and object densities. This eliminates the need for manual state quantisation and avoids information loss and aliasing effects introduced by coarse discretisation.

Despite its clear advantages over tabular Q-learning, DQN still faces important challenges in the slice-orchestration problem considered.

First, although function approximation improves generalisation across similar states, sample efficiency may remain limited when the action space is large. As the number of feasible configurations grows combinatorially, both exploration and learning become increasingly inefficient. Especially when actions correspond to complex joint configuration decisions (e.g., simultaneous placement and sensing-parameter selection), learning accurate Q-values becomes more difficult due to the increased dimensionality of the decision space and the resulting credit-assignment challenges. Moreover, at each update step the computation of $max_a\, Q(s,a)$ requires evaluating all candidate actions, which becomes increasingly costly as the action space grows.

Second, the use of an ϵ-greedy exploration strategy may become ineffective in large action spaces. Random exploration over a vast number of configurations has a very low probability of selecting meaningful or near-optimal actions, leading to slow policy improvement.

Therefore, while DQN removes the prohibitive memory requirements and the need for manual quantisation, its exploration and sample efficiency may still degrade significantly as the action space grows. Note that typically DQN is applied to relatively small action spaces (e.g. fewer than 20 actions in the seminal work [10]). Therefore, even the 432 actions in our setup when quantising the sensing SNR thresholds to 3 levels is already quite large and can become significantly larger under more fine-grained quantisation. This motivates the use of structured action representations and hierarchical decomposition mechanisms to further improve sample efficiency.

5.2.2.3 Hierarchical Q-Learning (HQL) and Hierarchical Deep Q-Network (HDQN)

This section introduces a hierarchical approach to tackle the sample efficiency and exploration limitations of Q-learning and DQN by exploiting the structure of the slice orchestration problem. The proposed approach decomposes the original joint (combinatorial) action into sub-actions to reduce the effective action complexity and guide exploration toward promising regions of the search space.

As discussed previously, in the considered formulation, an action is essentially a joint decision that combines multiple VNF placement decisions and the selection of sensing SNR thresholds. However, based on the structure of the reward function, we can decompose it into lower-dimensional components that not only reduce the effective branching factor but also facilitate credit assignment.

Hierarchical Action Decomposition

We hierarchically decompose the action space into two levels:

- High-level agent: selects the subset of physical nodes that will be active in the next timeslot.
- Low-level agent: selects the specific placement of VNFs to nodes, within the subset of active nodes provided by the high-level agent, and the sensing SNR thresholds.

Therefore, on the one hand, the high-level agent is responsible for capacity provisioning, as the subset of active nodes determines the total available capacity to be shared by the slices. On the other hand, the low-level agent is responsible for micromanagement, deciding how the available capacity will be utilised.

This decomposition encodes prior knowledge about the system structure and reshapes the learning problem. For example, the number of active nodes directly determines energy consumption and strongly affects SLA violations: activating too few nodes under high traffic leads to congestion and SLA penalties, while activating too many nodes increases energy cost.

Improved Credit Assignment

In the flat action space, the reward must be attributed to a large number of tightly coupled decision variables (individual VNF placements and sensing parameters). Each variable influences the reward only indirectly and through strong interactions with others. For example, to reduce energy consumption, multiple VNFs may need to migrate simultaneously from a specific node so that it can be unused and switched to sleep mode. However, if these migrations are not perfectly coordinated, the node may remain partially active, preventing any energy savings. In such cases, it also becomes unclear which specific decisions led to the low reward, making credit assignment particularly challenging (high variance rewards and poor sample efficiency). In contrast, in the hierarchical formulation the high-level decision (node activation) determines the overall capacity regime energy consumption of the system. If the selected set of active nodes systematically leads to poor rewards across multiple low-level refinements, this can be reliably attributed to inadequate capacity

provisioning. Consequently, the high-level agent receives more consistent and lower-variance feedback, which improves sample efficiency.

Action Space Complexity Reduction

As discussed in the problem description section earlier, for the setup considered in this work and assuming 3 quantisation levels for the sensing SNR thresholds, the total number of actions in the flat action space is 432. In the proposed hierarchical approach, the high-level agent first selects the active nodes subset among $(2^{N_{MEC}}\text{-}1) \times (2^{N_{CORE}}\text{-}1)$ subsets and the low-level agent then decides for the VNF placements and sensing parameters within the chosen subset. For the considered setup, the high-level action space comprises only $(2^2\text{-}1) \times (2^2\text{-}1)=9$ actions. While the total number of feasible joint configurations remains unchanged, the hierarchical structure reshapes the decision process reducing the effective branching factor and enabling more efficient exploration of fundamentally different operational regimes.

Hierarchical ε-Greedy exploration

In flat ε-greedy exploration, uniform random sampling over a high-dimensional combinatorial space exhibits strong bias toward configurations that activate many nodes and assign heterogeneous SNR thresholds. Energy-efficient configurations involving few active nodes have extremely low sampling probability.

In contrast, hierarchical ε-greedy decouples exploration at different abstraction levels: i) the high-level agent explores node-set configurations; ii) the low-level agent explores placement and sensing configurations conditioned on the selected node set. This structure dramatically increases the probability of sampling energy-efficient and structurally distinct regimes. Moreover, this decoupled exploration allows for example to explore different micromanagement actions (low-level) within a greedy high-level action or alternatively explore completely different regimes.

Flexible exploration is also enabled since each agent has a different exploration rate ε_H and ε_L, corresponding to the high and low-level agents respectively. The exploration rates can be tuned individually using different schedulers for the two levels. Note that in hierarchical (parameterised) settings, alternate optimisation schemes benefit from stationarity at one level while the other is being optimised [8]. Therefore, learning stability can be improved by using a smaller exploration rate that decays faster for the low-level agent (more stable micromanagement policy while the high-level agent explores different regimes). Once promising regimes are identified, the low-level agent can gradually refine its policy within that structure.

HQL details

In HQL, the implementation of the proposed hierarchical scheme requires the use of two Q-tables instead of the single Q-table of vanilla QL. More concretely, the high-level agent stores $Q_H(s,u)$, where u is the subset of physical nodes, while the low-level agent stores $Q_H(s,u,a)$.

At each timeslot: i) the high-level agent observes the current state s_t and selects the subset of active nodes u_t using an ε-greedy policy with exploration rate ε_H over $Q_H(s,u)$; ii) the low-level agent selects the final action $a_t \in A(u_t)$, which is a specific placement of VNFs to nodes within the subset u_t, using an ε-greedy policy with exploration rate ε_L over $Q_H(s,u,a)$; iii) the reward r_{t+1} and next state s_{t+1} are revealed and both agents update their Q-tables using the typical temporal-difference rule:

$$u'=\underset{u}{argmax}\, Q_H(s_{t+1},u),$$

$$Q_H(s_t,u_t) \leftarrow (1\text{-}\eta)Q(s_t,u_t)+\eta[r_{t+1}+\gamma Q_H(s_{t+1},u')],$$

$$Q_L(s_t,u_t,a_t) \leftarrow (1\text{-}\eta)Q_L(s_t,u_t,a_t)+\eta[r_{t+1}+\gamma \underset{a}{max}\, Q_L(s_{t+1},u',a)].$$

In the above expressions, the high-level action u' (subset of active nodes) that maximises $Q_H(s_{t+1}, u)$ is also a constraint in the low-level update, i.e., it restricts the feasible action space over which the maximisation in the low-level Q-function is performed.

While Hierarchical QL demonstrates the advantages of improved sample efficiency described above, compared to the vanilla QL of a flat action space, the scalability bottlenecks imposed by the tabular representation of the Q-function still remain.

HDQN details

The proposed hierarchical approach can be readily applied to DQN in the same way as in QL, replacing the two Q-tables with DNNs respectively.

More specifically, the high-level agent approximates the Q-function using a $Q_{\theta_H}(s, u)$ DNN with trainable parameters θ_H that takes as input the state and outputs the Q-value estimates for all possible physical node subsets. The low-level agent similarly uses a $Q_{\theta_L}(s, u, a)$ that takes as input the state and the selected subset of active nodes and outputs the Q-value estimates for all possible configurations. The conditioning of the low-level network output based on u is implemented by masking the invalid actions, where the actions involving nodes outside the selected subset are excluded by assigning them a value of $-\infty$, ensuring they are never selected during action maximisation.

Action selection follows the same hierarchical ε-greedy procedure described for HQL, using separate exploration rates $\epsilon_H$and ϵ_L.

In our implementation, a single replay buffer is used to store transitions, and both networks are updated using the standard update rule of DQN described in the respective section. The maximisation in the update is performed in the same way as described in HQL, meaning that the maximisation in the high-level update is over regimes (u'=$\underset{u}{argmax} Q_{\theta_H}(s_{t+1}, u)$), while the maximisation in the low-level update is only over actions compatible with u'.

Therefore, HDQN preserves the hierarchical bootstrapping structure introduced in HQL, while replacing tabular representations with DNN-based function approximation. This enables scalability to continuous and high-dimensional state spaces, without altering the hierarchical decomposition of action space.

5.2.3 Evaluation

Having introduced the proposed hierarchical DRL scheme (HDQN) and the (D)RL baselines (QL, HQL, DQN) in the previous section, here we will evaluate its performance to confirm the expected sample efficiency and cost gains against these baselines. The cost performance of the proposed scheme will be also compared to the policy of a dynamic agent that selects the best action based on current traffic demands assuming that traffic will remain static in the next timeslot. The latter will confirm the need to accurately anticipate traffic in an environment where traffic demands are time-varying and unknown.

System Setup

We use the same system setup with the respective section in deliverable D2.3 (see Figure 5-7). Therefore, in what follows we only recap its main characteristics.

The RAN consists of three BSs, while the MEC and Core networks each comprise two servers. The sensing SNR threshold at each BS is quantised and may take one of three discrete levels (low, medium, high). The network hosts two slices, one for communication and one for sensing. Each slice has one VNF that can be flexibly placed at one of the MEC servers (CU/sensing VNFs) and one VNF that can be flexibly placed at one

of the Core servers (UPF VNFs). Accounting for all possible VNF placements and sensing configurations, the agent has a total of 432 feasible actions as described in Section 5.2.1.2.

Traffic demands for the communication slice as well as object-activity traces that drive the sensing slice are imported from the real-traffic Milano dataset [9]. Since sensing activity is often correlated with communication demand, we select correlated traces from the above dataset. For the "tabular" methods, traffic is quantised uniformly into two levels (high-low).

Regarding service requirements, the sensing slice is associated with SLAs on throughput and sensing accuracy (considering both local per-BS and global sensing accuracy), while the communication slice is associated with an SLA on throughput. The required throughput for each slice varies dynamically according to the traffic demand. Whenever the delivered throughput or sensing accuracy falls below the required level, an SLA violation penalty is incurred.

Training

We train the DQN agent over 10 individual runs with different random seeds. Each run consists of 200 episodes, and all reported cost-related results are averaged across the runs. Regarding exploration, we apply an exponentially decaying exploration rate.

Results

The convergence behavior during training for all evaluated (D)RL agents is illustrated in Figure 5-8, which shows the evolution of the cost as a function of the training episode. The cost is defined as a weighted combination of multiple components, including SLA violation penalties, the number of active nodes, and VNF migrations, as discussed in Section 5.2.1.3, and thus serves as an overall indicator of system performance.

All agents start from a random initial policy, resulting in high cost at the early stages of training (left side of the plot). As training progresses, they gradually improve their policies through interaction with the environment, leading to reduced cost (right side of the plot). The best-performing algorithm is the one that achieves the lowest cost with the fewest training samples, i.e., the fastest convergence. The plot also includes the Opt. Dynamic Policy under Static-Traffic Assumption baseline, where the agent selects the optimal action based on current traffic demands while assuming that these demands remain unchanged in the next time slot. Since this scheme does not involve learning, it is used solely as a cost performance baseline.

The obtained results lead to the following key observations. First, both tabular approaches (QL and HQL) exhibit significantly slower convergence compared to the DQN-based methods, demonstrating orders-of-magnitude lower sample efficiency. This behavior is expected given the large state-action space, which makes exploration and learning considerably slower when using tabular representations.

Second, due to the manual (and coarse) quantisation of traffic demands required by tabular QL-based methods, the final policies obtained by these methods result in higher cost compared to those learned by the DQN-based agents. In addition, the tabular approaches demonstrate less stable training behavior due to the coarse traffic quantisation that leads to aliasing, meaning that different traffic conditions are mapped to the same state. In contrast, the DQN-based methods operate directly on real-valued inputs and benefit from experience replay and mini-batch updates, which further improves training stability. We note here that more fine-grained quantisation of traffic was also attempted for the tabular methods (up to 4 levels), leading to more stable but extremely slow learning without any gains in terms of quality of the obtained policy.

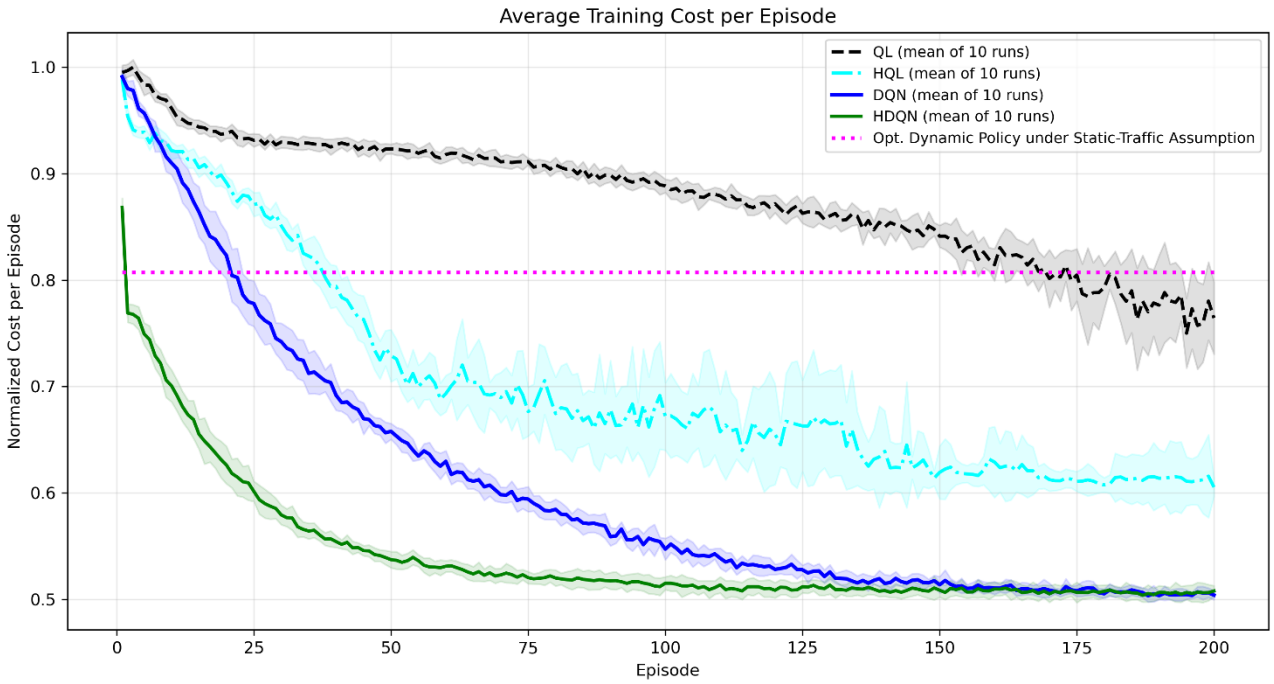


Figure 5-8 Convergence plot.

Third, even within the tabular setting, the proposed hierarchical decomposition offers significant improvements compared to the flat action-space QL. More specifically, after roughly 25 episodes HQL reaches the same cost level that vanilla QL achieves at the end of training (episode 200), demonstrating a substantial improvement in learning speed.

Fourth, both DQN-based approaches outperform the dynamic policy that selects the best configuration under the assumption that traffic remains unchanged in the next time slot. This highlights the importance of learning adaptive policies that can implicitly anticipate traffic variations in environments with time-varying and uncertain demands. In contrast, the optimal dynamic policy under the static traffic assumption often selects configurations that are optimal for the current conditions but become suboptimal once the actual traffic demands are revealed in the following time slot. This mismatch leads to frequent SLA violations and increased overall cost. This observation builds on top of the evaluation results of **D2.3** [2], where we showed that dynamic policies significantly outperform static baselines in our setup.

Finally, comparing the two DQN-based approaches, HDQN demonstrates faster convergence than vanilla DQN as expected. In the evaluated scenario, HDQN reaches its lowest cost at approximately time slot 100, while DQN requires roughly 150 time slots to reach a similar performance level. This corresponds to an improvement of approximately 33% in sample efficiency for HDQN, achieved without any degradation in the quality of the learned policy.

5.3 GenAI-Enabled Intend-Driven RAN Slicing

5.3.1 Problem Description

Managing RAN slicing in 5G/6G environments is highly complex. Networks must concurrently support diverse, conflicting workloads (e.g., low-latency VPNs vs. high-bandwidth 4K streaming) under strict QoS requirements. Traditional static templates and manual configurations lack the agility needed to adapt to real-time network dynamics. This section describes the implementation of an LLM for the automation of RAN slice configuration. In particular, the following is considered:

- **Scenario:** 2 User equipment have 2 different use cases (low-latency VPNs vs high-bandwidth 4K streaming).
- **Objective**: Create a slice optimisation for watching YouTube in 4K using LLM-powered AutoRAN rApp.

5.3.2 Algorithm Description

The flowchart of Figure 5-9 describes the step-by-step process of managing RAN slicing, from the initial brainstorming idea to applying changes to the gNB via O-RAN services. Moreover, this transformation stage is driven by the AutoRAN rApp, which seamlessly translates ideas into actionable network policies. These policies are deployed across the management and control planes and autonomously executed at the radio node level for continuous, dynamic resource optimisation.

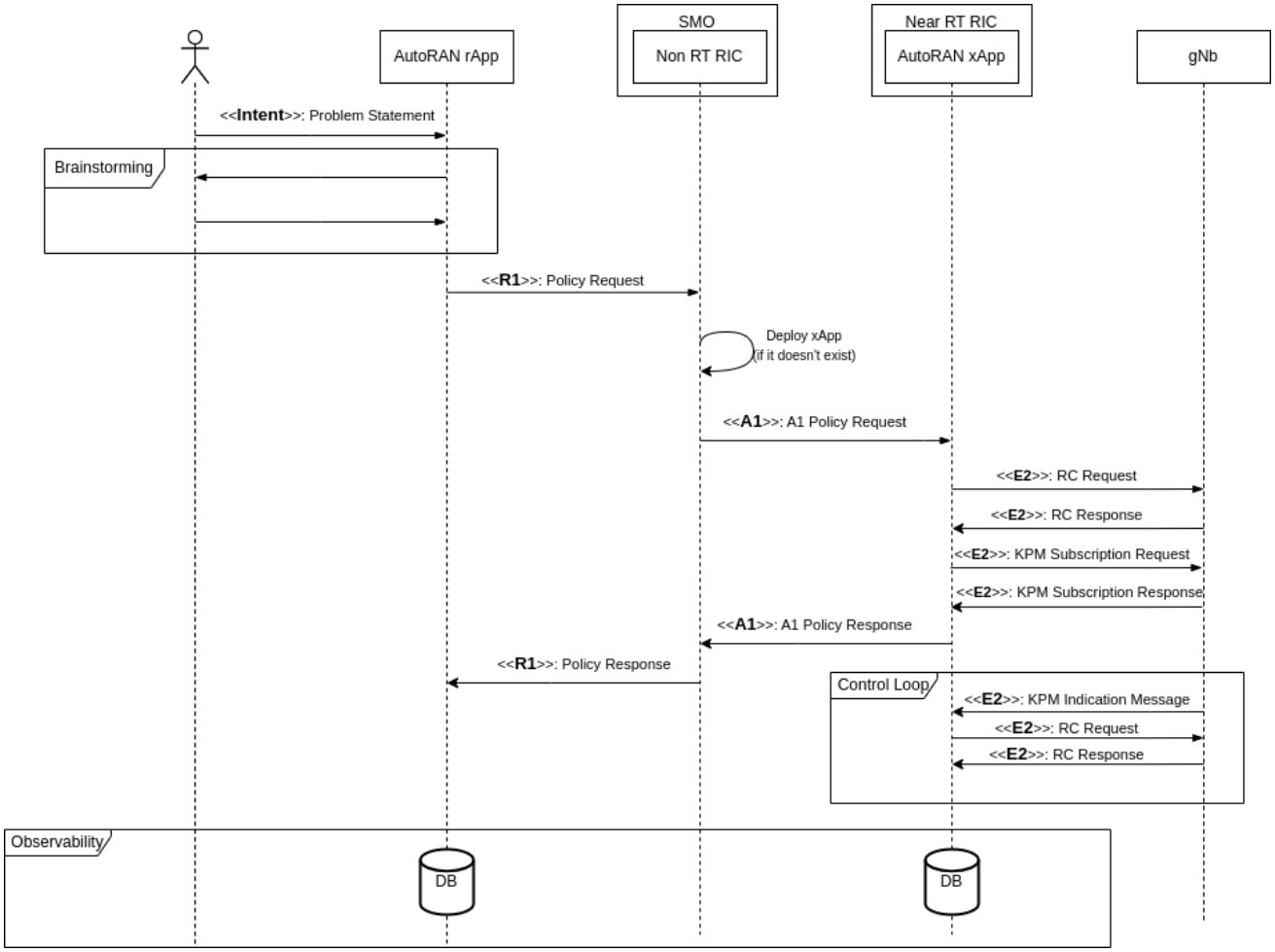


Figure 5-9 End-to-end automation workflow.

1. **Intent Ingestion and Brainstorming**
   - **Input:** The operator submits a natural language problem statement or slicing requirement (<<Intent>>).
   - **Processing:** The LLM-powered AutoRAN rApp engages in a "Brainstorming" phase. During this step, the generative model analyzes the intent, assesses feasibility, and synthesises a high-level network policy strategy without requiring manual rule-coding from the operator.
2. **Policy Formulation and Deployment**

- **Request initiation:** The AutoRAN rApp formulates a precise policy configuration and transmits an R1 Policy Request to the Service Management and Orchestration (SMO) framework / Non-RT RIC.
- **Validation**: The SMO verifies the request and checks the Near-RT RIC environment. If the necessary control logic (the AutoRAN xApp) is not already active, the SMO executes a deployment loop (Deploy xApp) to instantiate it.

3. **Policy Translation and Initial Configuration**
   - **A1 Policy Enforcement:** The SMO sends an A1 Policy Request down to the Near-RT RIC, effectively passing the translated GenAI policy parameters to the fast-control tier**.**
   - **RAN Configuration:** The Near-RT RIC translates this A1 policy into low-level node commands, sending an E2 RC (RAN Control) Request and E2 KPM (Key value measurement report) Request to the target gNB to configure the specific slicing parameters (e.g., PRB allocation). The gNB confirms execution via an E2 RC Response and an E2 KPM Subscription response**.**
4. **Telemetry Subscription and Acknowledgment**
   - **Monitoring Setup:** To enable real-time adaptation, the Near-RT RIC sends an E2 KPM (Key Performance Measurement) Subscription Request to the gNB. The gNB acknowledges this with a KPM Subscription Response**.**
   - **Upstream Confirmation:** With the slice configured and monitoring active, the Near-RT RIC sends an A1 Policy Response back to the SMO, which propagates an R1 Policy Response to the AutoRAN rApp, confirming successful deployment**.**
5. **Real-Time Closed-Loop Control**
   - **Execution:** Once initialised, the system enters a continuous "Control Loop" managed by the Near-RT RIC**.**
   - **Dynamic Adaptation:** The gNB streams telemetry via E2 KPM Indication Messages. The AutoRAN xApp evaluates this real-time data against the GenAI-defined policy boundaries and autonomously issues subsequent E2 RC Requests to adjust radio resources dynamically, receiving E2 RC Responses to close the loop**.**
6. **E2E Observability**
   - **Logging:** Throughout the entire lifecycle, both the AutoRAN rApp (in the management plane) and the Near-RT RIC (in the control plane) log system states, actions, and performance metrics to their respective Databases (DB). This observability layer feeds historical data back into the system for future GenAI model refinement and anomaly detection**.**

5.3.3 Evaluation

The GenAI-enabled slicing algorithm is evaluated within an end-to-end 5G network architecture to assess its autonomous management capabilities and its impact on operational efficiency. Early results indicate that the GenAI framework effectively balances resources across competing slices using real-time data, drastically reducing the need for manual oversight.

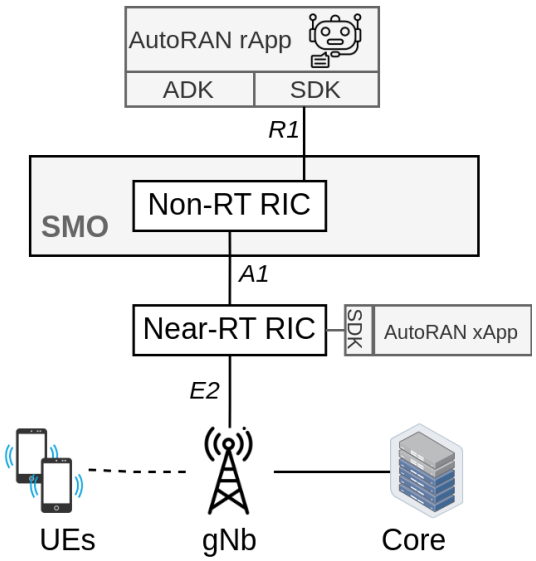


Figure 5-10 Evaluation architecture overview.

To visualize how this autonomous management is operationalised, Figure 5-10 above illustrates the integration of the GenAI framework within the O-RAN architecture. It depicts the hierarchical control structure, showing how the AutoRAN rApp in the management plane coordinates with the Near-RT RIC and its designated xApp to execute real-time resource adjustments at the gNB.

- **Testbed Setup:** it comprises a complete 5G architecture including a 5G Core, gNBs, Near-RT RIC, and User Equipment (UEs) configured for distinct slice profiles (e.g., VPN vs. YouTube streaming). Figure 5-11 depicts the deployed network in BubbleRAN's platform, consisting of a core network, RAN (Radio Access Network), and two simulated radio-frequency User Equipment (UEs). As observed in the Figure, both UEs exhibit similar bandwidth characteristics, indicating balanced resource distribution under the configured slicing strategy.
- **Primary Metrics:** the main goal of this evaluation is to assess the reduction in operational complexity (measured by time and manual steps saved) alongside Service Level Agreement (SLA) satisfaction rates. To achieve this objective, a prompt is sent to the agent, which creates two distinct network slices based on its recommendations. As shown in Figure 5-12 below, the slices are configured with different throughput limits: a Video slice with a maximum throughput of 80 Mbps, and a VPN slice with a maximum throughput of 20 Mbps.
- **Performance Results:** they evince a significant simplification of the slice orchestration process and robust, real-time resource balancing. To meet the user's intent, both the policy and xApp are deployed to dynamically adjust network behavior. As illustrated in Figure 5-13 below, the Grafana dashboard shows throughput variations across two different network slices. These changes are driven by prompts generated by the LLM model, enabling adaptive resource allocation. More details are available in the form of a video demonstrating the BubbleRAN MX-AI architecture and the LLM-powered AutoRAN rApp managing E2E slices[3]. Intelligent mobility management empowered by sensing is yet another use case where ISAC is beneficial[4].

[3] Demonstration video of MX-AI: BubbleRAN Ecosystem for Multi X Automation & Intelligence, available at: https://www.youtube.com/watch?v=CEIya7988Ug.

[4] https://www.youtube.com/watch?v=CGgvUCKy3M4

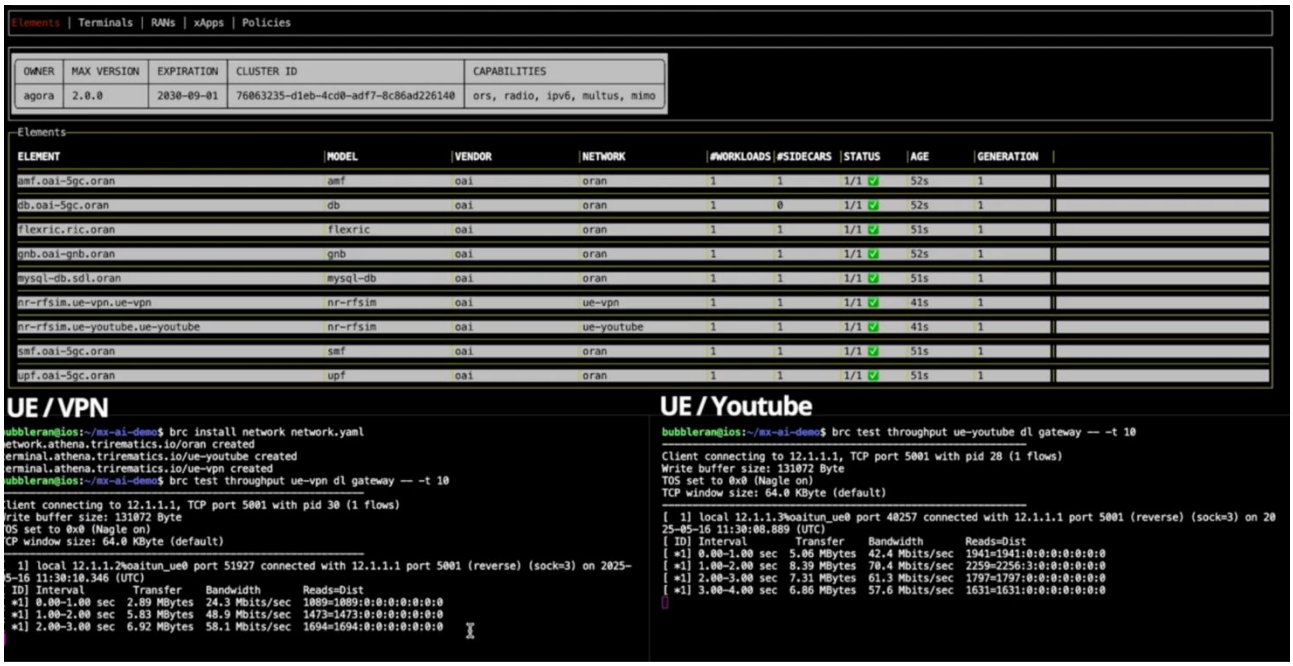


Figure 5-11 Screenshot of the deployed network in BubbleRAN's platform, consisting of a core network, RAN, and two simulated radio-frequency UEs

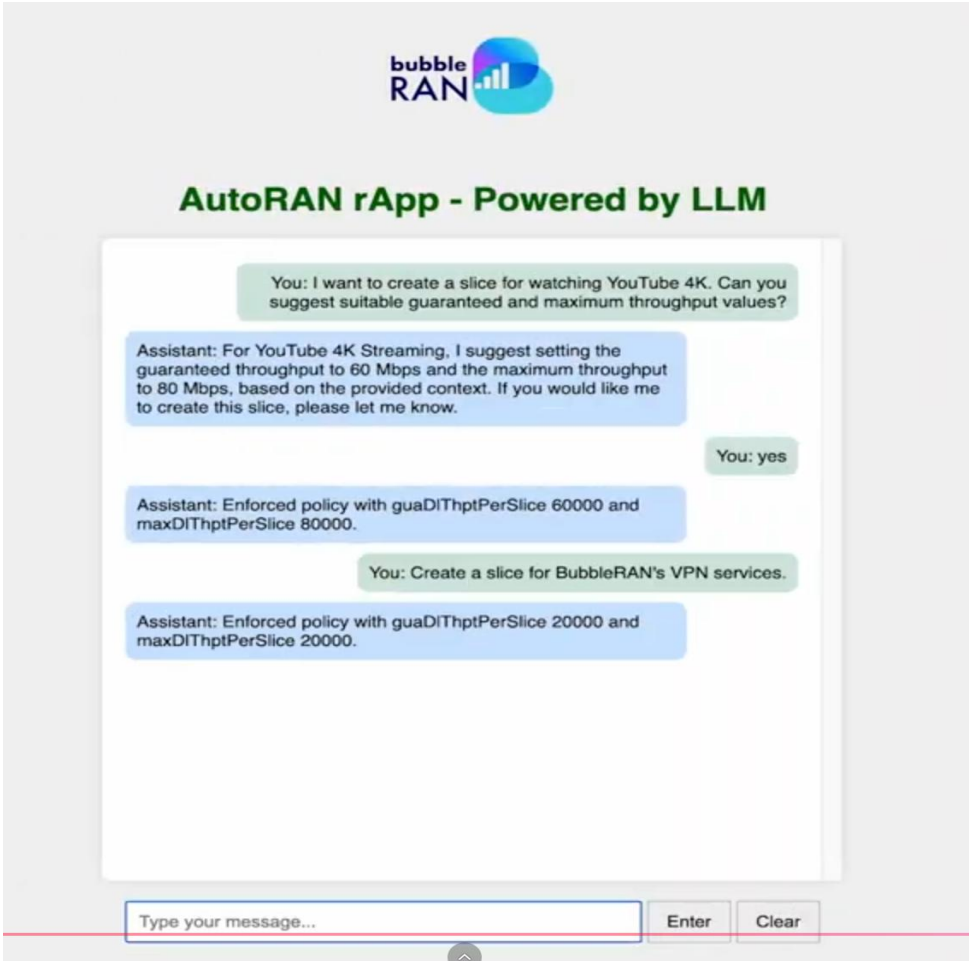


Figure 5-12 Prompts sent to the GenAI agent for slice creation

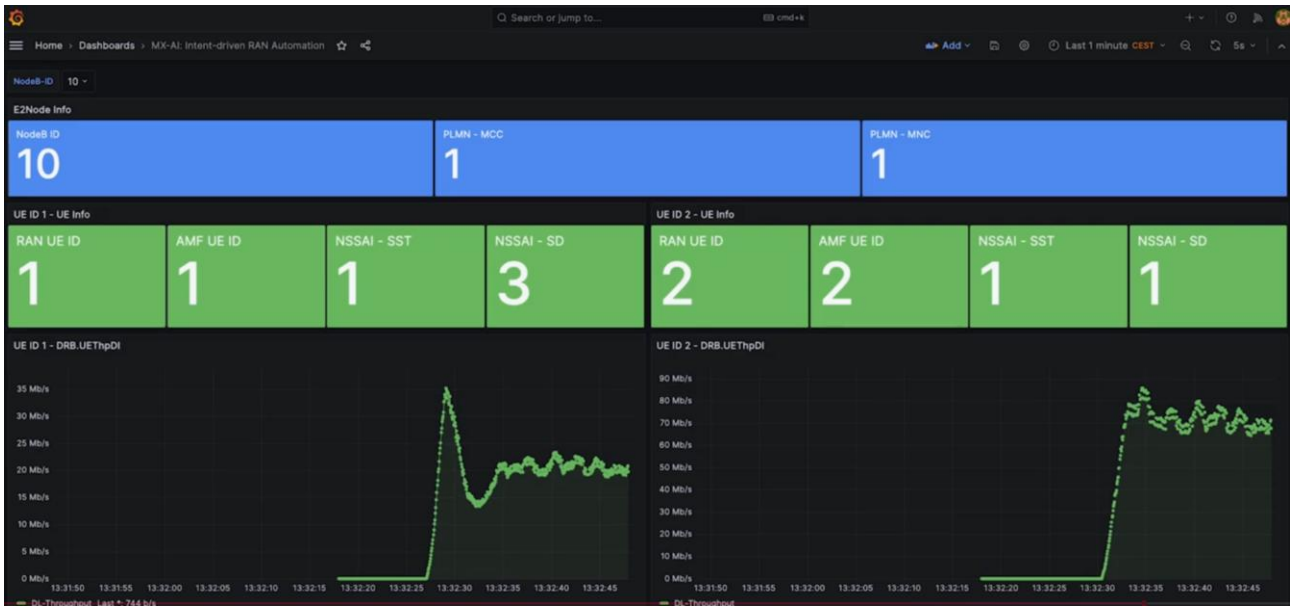


Figure 5-13 Grafana dashboard screenshot, depicting throughput variations across the two deployed network slices

5.4 Adaptive Federated Learning Framework for Sensing-Based Trajectory Prediction

5.4.1 Problem Description

We consider the problem of trajectory prediction for moving targets discussed in Section 2.2.2. As an alternative to the centralised sensing approach, a Federated Learning (FL) framework can be adopted, where AI models for sensing are trained locally at distributed sensing nodes, using locally generated range–Doppler radar images (see Figure 5-14).

In this model, each sensing node independently trains an AI model, such as a CNN, LSTM, or hybrid CNN–LSTM architecture, to support target detection, motion estimation, and trajectory prediction. Instead of transmitting high-volume raw sensing data over the fronthaul and backhaul networks, the nodes periodically exchange only model updates, such as weights or gradients, with a federated aggregation function. This function combines the received updates to construct a global sensing model, which is then redistributed to the participating nodes and used as the baseline for subsequent local training. This process is repeated iteratively until convergence or until a predefined performance criterion is satisfied.

By avoiding centralised data aggregation, the FL-based approach significantly reduces fronthaul and backhaul bandwidth requirements, enhances data privacy, and improves scalability and robustness under heterogeneous sensing conditions. The resulting global model can provide accurate trajectory predictions that are exploited by higher-layer control functions to proactively allocate radio, transport, and compute resources and ensure service continuity for sensing-enabled services.

In this context, AI agents observe the predicted target trajectories, sensing accuracy indicators, and the current state of radio, transport, and compute resources. Based on this information, they take decisions that balance sensing performance with communication service requirements. Specifically, the AI agent can decide which sensing nodes should participate in each FL round, adapt the sensing and training periodicity, and allocate radio and compute resources between sensing and communication functions. In addition, it can influence network-level actions such as traffic routing, scheduling, and resource provisioning across the RAN, transport, and edge/cloud domains. Through continuous interaction with the environment, the RL agent enables proactive, adaptive, and resource-efficient operation of ISAC services.

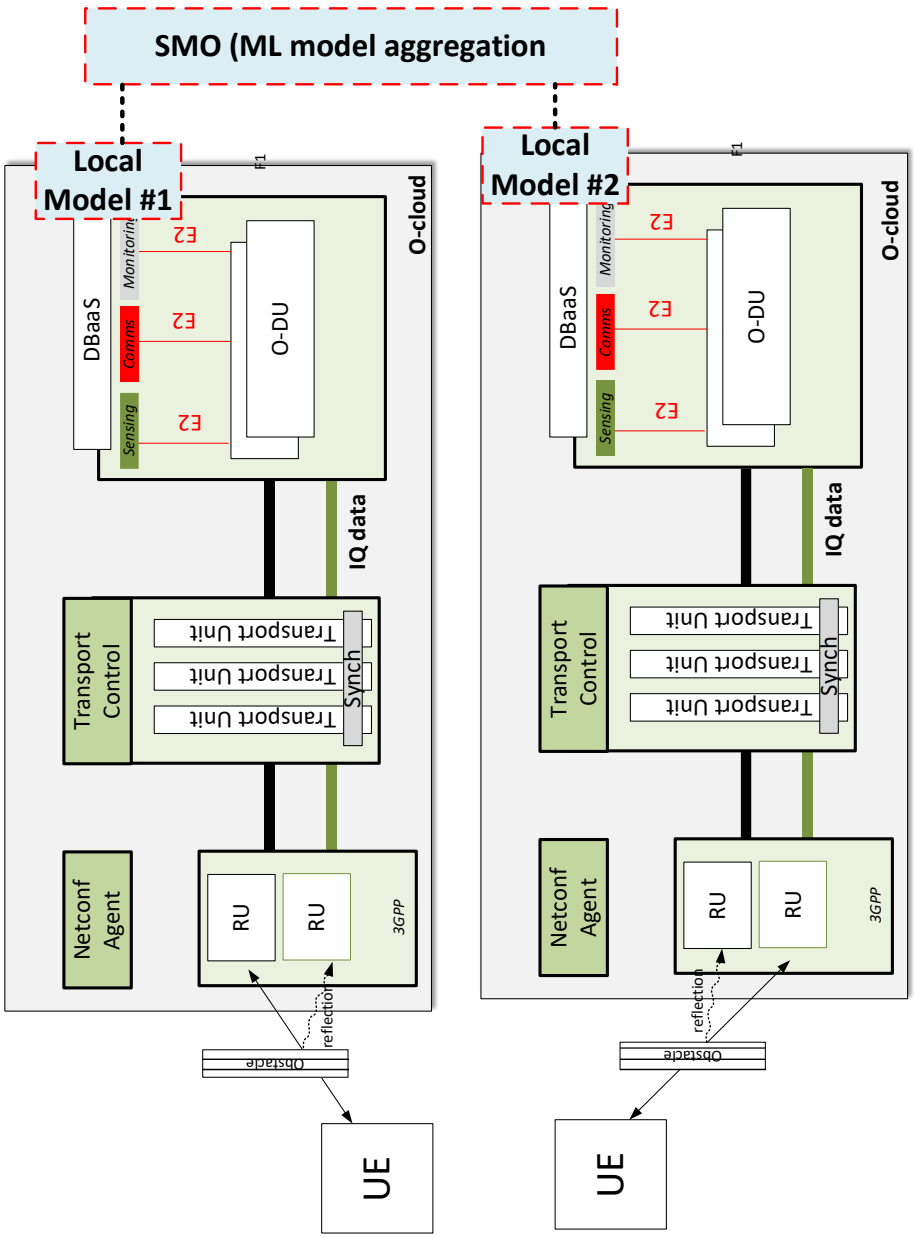


Figure 5-14 FL for ISAC services

5.4.2 Algorithm Description

The proposed rApp algorithm jointly optimises moving-target trajectory prediction and sensing-service orchestration by dynamically selecting between centralised sensing and FL-based distributed sensing according to current network conditions, sensing requirements, and available resources.

At each control interval, the rApp collects real-time information from the RAN, transport, and compute domains. This includes communication traffic demand, fronthaul and backhaul utilisation, radio resource occupancy, optical transport load, compute resource availability at MEC or cloud locations, current sensing accuracy, trajectory-prediction confidence, delay measurements across the RAN, transport, compute, and end-to-end paths, as well as the number, location, and mobility characteristics of active sensing nodes and detected targets. These measurements define the current system state and provide the basis for subsequent orchestration decisions.

Based on this state, the rApp receives high-level service intents describing the required sensing accuracy and communication performance. These intents are translated into operational targets for the lower-level agents controlling the RAN, transport, and compute domains. The rApp therefore acts as the higher-layer coordination entity that determines how sensing and communication objectives should be balanced under the current network conditions.

The next step is the selection of the sensing processing mode. When sufficient transport capacity is available, low-latency paths exist, and the highest possible trajectory-prediction accuracy is required, the algorithm selects the centralised sensing mode. In this case, distributed sensing nodes generate range–Doppler radar frames and transmit them to a central sensing fusion function. The multi-node sensing data are jointly processed, enabling spatial diversity across different sensing viewpoints. A hybrid CNN–LSTM model is then used for trajectory prediction: the CNN extracts spatial target signatures from the range–Doppler frames, while the LSTM captures the temporal evolution of the target movement and predicts the future trajectory.

When the transport network becomes congested, bandwidth consumption must be reduced, or distributed scalability is required, the algorithm selects the FL-based sensing mode. In this mode, each sensing node trains a local trajectory-prediction model using its own locally generated range–Doppler radar frames. Instead of transmitting raw sensing data to a central location, the nodes exchange only model parameters, gradients, or model updates with a federated aggregation function. The aggregator combines the received updates to construct a global trajectory-prediction model, which is then redistributed to the sensing nodes. The nodes continue local training iteratively using the updated global model as a baseline. This approach significantly reduces fronthaul and backhaul traffic, improves privacy, and avoids overloading the central sensing fusion function, while still enabling collaborative learning across multiple sensing nodes.

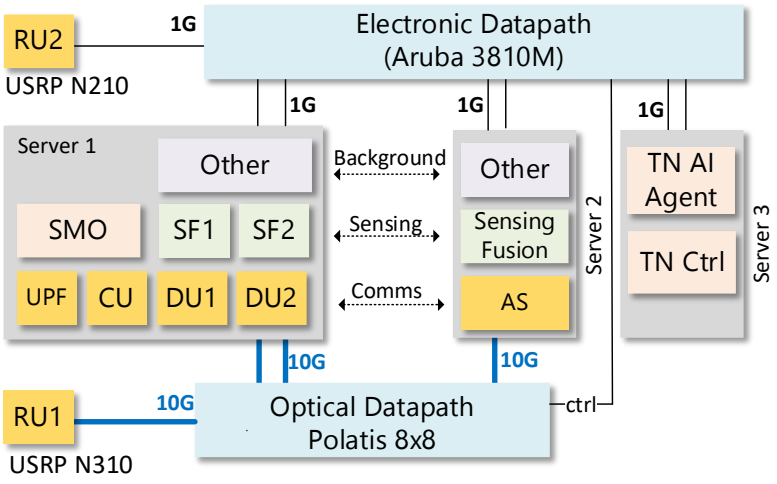
After the sensing mode has been selected, domain-specific resource optimisation is performed. The xApp-based RAN agents decide which O-RUs should participate in sensing, how radio resource blocks should be split between communication and sensing functions, which beam configurations and sensing waveform parameters should be applied, and what range–Doppler generation resolution should be used locally. The compute agent decides whether sensing processing should be placed at the MEC or cloud, how CPU/GPU resources should be allocated, whether sensing functions should be scaled, and whether centralised or FL-based processing should be maintained.

Finally, the rApp performs or coordinates trajectory prediction according to the selected sensing mode. In centralised mode, prediction is performed by fusing the range–Doppler information received from multiple sensing nodes and applying the CNN–LSTM trajectory-prediction model centrally. In FL mode, prediction relies on the collaboratively trained global model, while raw sensing data remain local to the sensing nodes. The predicted trajectories are then exposed to higher-layer control functions, which may proactively activate additional sensing nodes, adjust beam configurations, hand over sensing responsibilities to neighbouring cells, reserve transport capacity, or adapt compute resources in advance. In this way, the rApp algorithm enables adaptive, bandwidth-efficient, and sensing-aware orchestration of ISAC services across the RAN, transport, and compute domains.

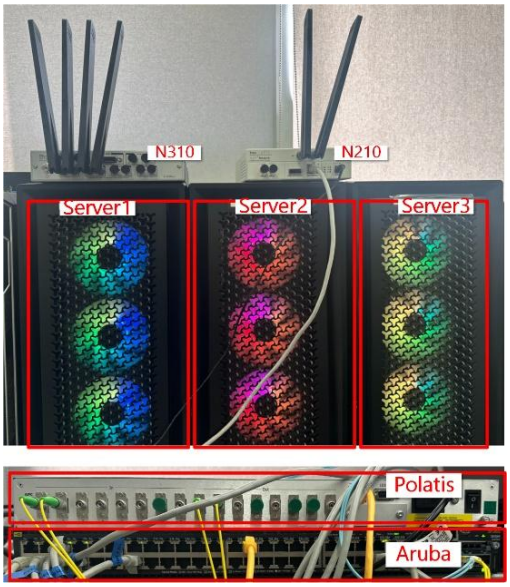
Details of the mathematical modeling framework are provided in [2].

5.4.3 Evaluation

We evaluated the performance of the proposed framework considering the small-scale testbed shown in Figure 5-15.



a)



b)

Figure 5-15 a) Lab testbed connectivity with electronic and optical data paths, b) physical nodes

The transport infrastructure in our analysis combines an optoelectronic packet-switching segment with an all-optical switching segment. The optoelectronic segment is implemented using OpenFlow-capable Aruba switches (3810M), while the all-optical segment is implemented using a Polatis 6000 optical switching fabric. Control of the optoelectronic segment is provided by the OpenDaylight (ODL) SDN controller, which maintains topology and state for the OpenFlow switch and exposes control through RESTCONF. The framework interacts with ODL exclusively through RESTCONF APIs, relying on ODL's operational datastore to retrieve link/port state and to program OpenFlow rules on the packet switches. Control of the all-optical segment is performed through direct interaction with the Polatis switch using Telnet and SCPI-style commands, enabling the framework to establish, validate, and tear down optical cross-connections on demand. For the mobile segment we rely on the O-RAN implementation provided by the O-RAN alliance which is controlled through the SMO/OAM. For the RUs we rely on a USRP N310 supporting 1 x SFP+ and an N210 with 1GbE interfaces. N310 is attached to the optical Datapath whereas N210 to the electronic.

The developed framework has been deployed across three servers as shown in Figure 5-15. The Polatis is an all optical MEMS based switch that provides the optical datapath to the servers, which are interconnected with 10G SFP+ transceivers whereas the optoelectronic switch a typical 1GbE has been used. For server 1 a Network Interface Card (NIC) with 2xSFP+ and 2xGbE transceivers has been used, whereas for Server 1xSFP+ and 2x1GbE ports have been activated. In Server 3, only 1GbE ports have been activated and they are used for control of the transport network. Server1 to Server 2 connections are established: backhaul and sensing connectivity, interconnecting the UPF with the application server and local sensing functions with the sensing fusion function. In addition to this, a dynamic background traffic is created between two VMs hosted in server1 and server 2.

A) Lab scale evaluation

To evaluate the performance of the RL scheme we initially benchmarked on the time needed to reconfigure RAN and network resources. RAN reconfiguration is performed through SMO/OAM O1/NETCONF protocol. The corresponding benchmarks are shown in Figure 5-16, testing possible actions taken by the RAN AI-Agent to "Create a Sensing RU" and "Activate a Sensing RU", to configure specific RU parameters i.e., "Change Gain" or "Change Bandwidth", to "Create VLAN" for sensing and comms flows. From these results we observe that the action to "Activate Sensing RU" exhibits the highest average latency. Activation transitions the RU context from a configured but inactive state into an operational state, enabling the generation of U-plane IQ samples. This step typically triggers multiple internal actions, including validation of all configured parameters, allocation of PHY and fronthaul resources, and synchronisation with timing and transmission subsystems. Because this operation directly impacts the runtime data plane, it is inherently more complex than pure configuration updates, resulting in higher latency compared to other steps.

The Cumulative Distribution Function (CDF) of the reconfiguration time for the optical Datapath is shown in Figure 5-17. The control for the optical switch is performed over an RJ45 1GbE port. The SDN controller is hosted in Server 3 and controls the optical switch over the electronic datapath. Reconfiguration delays for one optical datapath range between 10.32 and 11.23ms. The variance is attributed to queuing delays of the electronic switch as no QoS policy has been applied and the server hosting the controller.

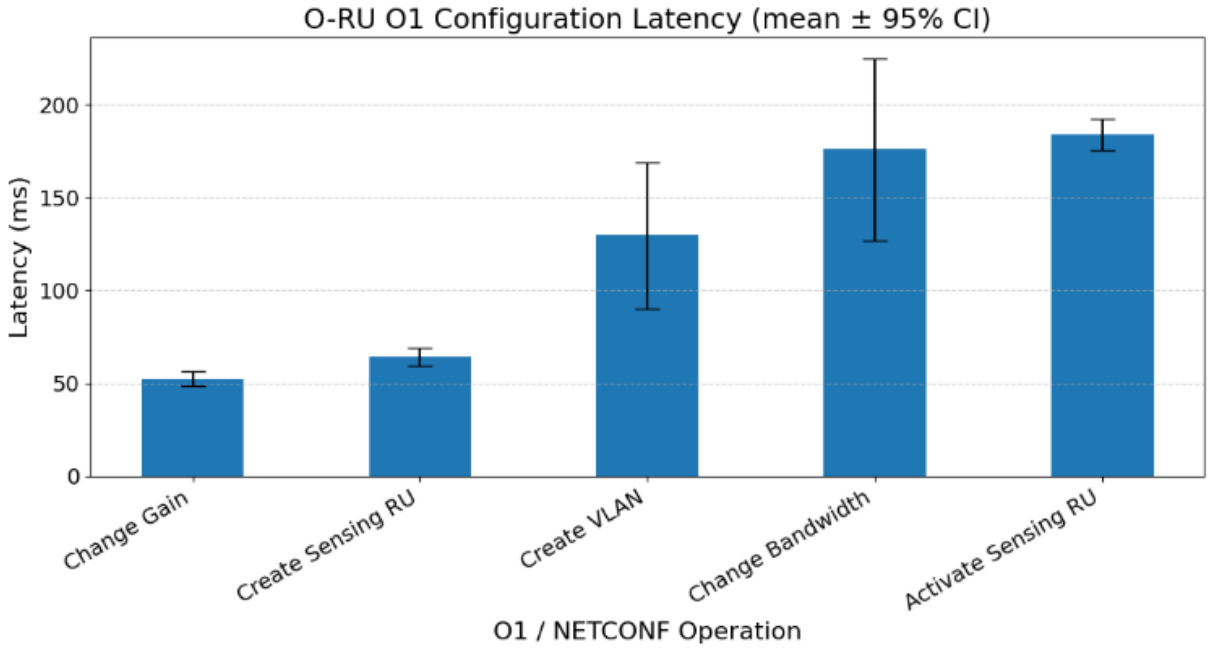


Figure 5-16 RAN reconfiguration time through the SMO/OAM using O1/NETCONF

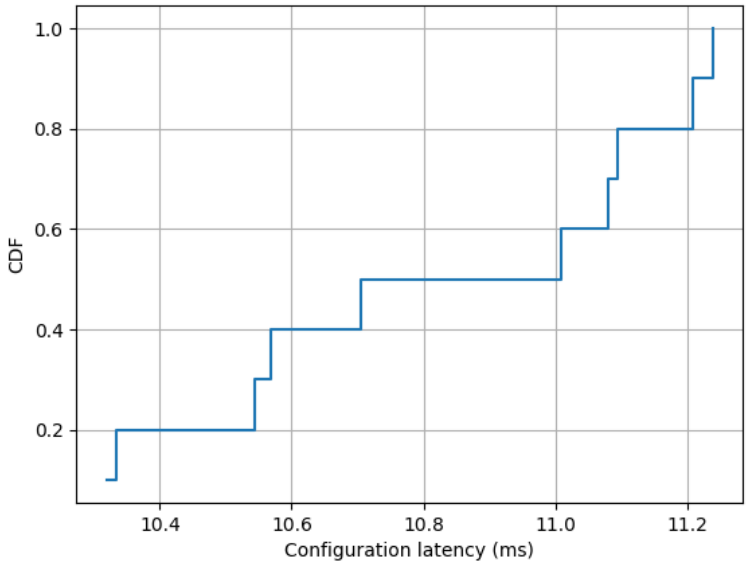
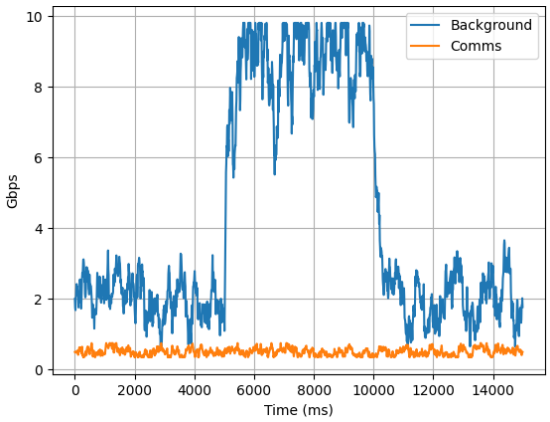
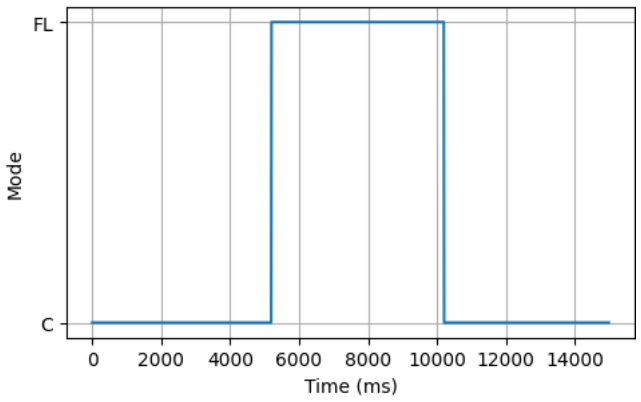


Figure 5-17 CDF Optical Datapath reconfiguration time

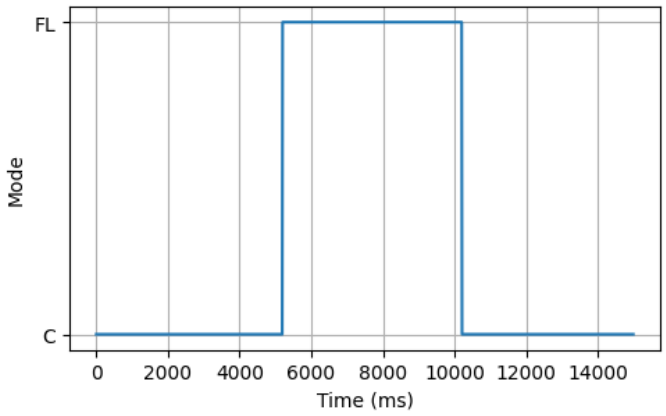
The ability of the proposed agentic AI framework to reconfigure RAN, transport network and compute resources is evaluated over testbed shown in Figure 5-15, for the scenario where the background traffic generated by a VM in server 1 destined to server 2 varies over time. In addition to background traffic, the transport network should also provide resources for user-plane (communication traffic) and sensing traffic connectivity. The servers are interconnected using an optical datapath with 10Gbps capacity and an electronic path with 1Gbps capacity. Figure 5-18a) Shows the egress traffic from server 1. During the initial period (0–5 s), the network experiences a total load of approximately 2–3 Gbps, while the backhaul communication traffic generated by the RAN fluctuates between 50‑300 Mbps. At t=5 s, the background traffic increases significantly, reaching values close to 9‑10 Gbps, which brings the total network load close to the transport capacity. After t ≈ 10 s, the background traffic decreases again to a moderate level. In Figure 5-18 (b) we observe that the proposed framework switches from centralised to FL learning significantly reducing the transport load by transmitting only model updates instead of high-volume sensing data Figure 5-18 (c). The use of FL prevents the sensing traffic from exceeding the available transport capacity. Instead of transmitting full RD maps, only model updates are exchanged, keeping the aggregate traffic below the network capacity limit. When the background traffic decreases again, the RL policy switches to centralised sensing.



a)



b)



c)

Figure 5-18 a) Generated Communication and background at Server b) Switch from Centralised to FL learning, c) Total network traffic

B) Large-scale evaluation

We then examine the performance of the complete considering the optical network topology shown in Figure 5-19(a). Each fiber link supports 8 wavelengths with 100 timeslots/wavelength. To evaluate the capability of the AI agents to adapt to dynamic traffic scenarios we consider, as input, communication traffic that varies over time as shown in the snapshot of in Figure 5-19 (b). Specifically, in the simulated scenario the traffic evolves in phases, starting with a period of high demand, followed by a lower demand period, and finally returning to a high-load regime. This dynamic pattern is used to emulate realistic network conditions where traffic varies over time.

Figure 5-19 (c) illustrates the sensing distribution mode selected by the controller. When the traffic input is high, the utilisation of the transport network increased leading to an increase of the resources needed to support ISAC services. The compute controller switches to the FL mode because it significantly reduces the amount of sensing data that needs to be transported through the network reducing transport cost. When the traffic load decreases, the system returns to centralised sensing, which provides higher sensing accuracy as the trajectory prediction algorithm has full knowledge of all sensing data. The switching behavior therefore reflects an adaptive strategy that balances sensing performance and transport network utilisation.

Figure 5-19 (d) shows the transport network cost over time. This cost reflects the amount of network resources that are used to carry both communication and sensing traffic. Centralised sensing requires continuous transmission of range–Doppler maps, which increases the amount of traffic carried by the transport network. FL requires significantly less continuous sensing traffic, since only model updates are transmitted. As a result, FL reduces the average transport cost. The spikes observed in the cost curve correspond to periodic model update transmissions in the FL process. These updates are transmitted at specific intervals, creating short bursts of additional traffic.

Figure 5-19 (e) compares the sensing accuracy obtained with the different approaches. Centralised sensing generally achieves the lowest trajectory estimation error because the full sensing information is available at the fusion node. However, under high transport load this approach becomes less efficient due to congestion effects. FL exhibits slightly higher sensing error because sensing data is processed locally and only model updates or summarised information are exchanged. The adaptive strategy combines the advantages of both approaches by using centralised sensing when the network load is low and switching to FL when the transport network becomes congested.

Finally, Figure 5-19 (f) presents the trade-off between sensing accuracy and transport network cost. Each point represents a particular operating condition of the system. Points corresponding to centralised sensing tend to achieve lower sensing error but require higher transport resources. Points corresponding to FL show lower transport usage but slightly higher sensing error. The adaptive strategy lies between these two extremes and achieves a better balance between sensing accuracy and network resource utilisation.

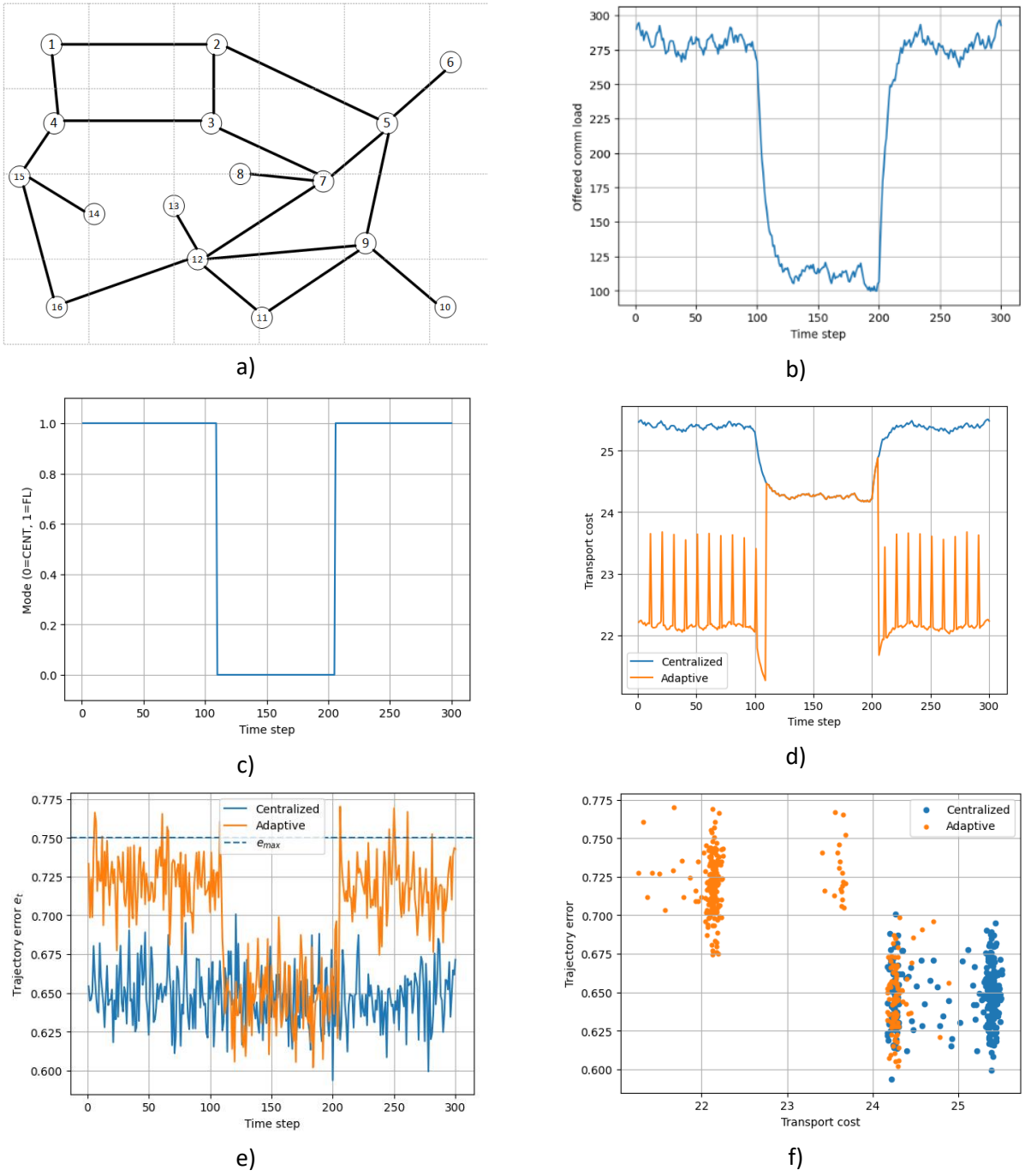


Figure 5-19 a) Optical network topology, b) Snapshot of incoming network traffic at the optical transport, c) Dynamic adaptation of compute policy, d) Impact of compute policy on transport resources, e) Impact on sensing accuracy, f) transport cost and sensing accuracy tradeoffs

5.5 DNN-Based Gesture Recognition

5.5.1 Problem Description

The Intel gesture recognition framework, known as WiRD-Gest, is designed to perform accurate gesture recognition using a single, unmodified Wi-Fi transceiver on COTS hardware. By leveraging a monostatic full-duplex sensing pipeline, the system extracts high-fidelity Range-Doppler (RD) information to distinguish human gestures from background interference.

5.5.2 Algorithm Description

The WiRD-Gest system utilises a monostatic full-duplex sensing pipeline to extract high-fidelity RD information, isolating human gestures from background interference. The pipeline processes raw Wi-Fi signals into actionable input tensors for deep learning models through the following steps:

- **Signal Synchronisation**: The system performs coarse and fine delay calibration, followed by phase synchronisation across frames, to compensate for hardware asynchronisation.
- **Self-Interference Cancellation**: To mitigate strong Transmitter-Receiver (Tx-Rx) coupling, the system subtracts the mean across all frames.
- **Input Representation**: Each RD frame is converted from raw Signal-to-Noise Ratio (SNR) values into a normalised tensor by clipping the values to [5, 40] dB and scaling them to a [0, 1] range.
- **Video Segmentation**: The frames are resised to a 64x64 target resolution and segmented into fixed-length clips using a sliding window. Each sequence consists of 32 RD frames, representing 3.2 seconds of temporal data containing one complete gesture.

Training Configuration To evaluate model performance, the dataset of 720 gesture instances was utilised, with 30% allocated for validation and testing. Deep learning architectures were trained using the following configurations to ensure robust generalisation:

- **Loss and Optimizer**: Models were trained using cross-entropy loss and the Adam optimizer.
- **Hyperparameters**: A learning rate and weight decay of 0.001 were applied, alongside a dropout rate of 0.2 and a batch size of 16.
- **Epochs**: Training was conducted for up to 100 epochs.

5.5.3 Evaluation

Hardware Setup: All data were collected using a single commercial laptop (Lenovo ThinkPad L14) equipped with an Intel Wi-Fi 6E AX211 NIC, operating in channel 79 (6.345 GHz center frequency), with a bandwidth of 160 MHz, 512 subcarriers, and a frame rate of 40 Hz ($\Rightarrow \pm 0.4$ m/s unambiguous velocity). The system provides range and Doppler resolutions of 0.93 cm (160 MHz bandwidth) and 0.03 m/s, respectively.

Scenario and Dataset: We collected a comprehensive dataset of 191000 frame samples (77500 gesture frames) to train and evaluate our models in controlled settings. Five distinct users participated in data collection, each instructed to perform five common gestures: (1) Push & Pull - hand toward and away from PC, (2) Slide - side to side motion, (3) Up-Down - up and down motion, (4) Double Pulse - two open hands in front of PC, and (5) Double Rotate - rotate twice clockwise in front of PC.

To ensure some environmental diversity, data were collected in two different indoor locations: data from two users were collected in a furnished office room (location A), while the other three users' data were collected in an open meeting room (location B). For each gesture, every user performed 20 repetitions for

the training set, plus 8 repetitions for the test set. This resulted in a total dataset of 720 gesture instances where 30% was used for validation and testing.

Unseen public-space validation. To assess robustness in the real world, one user also performed the gestures in a third location (location C) - a public space (cafe) characterized by dense foot traffic and monostatic Wi-Fi devices. This data is never used for training or model selection and serves exclusively as an out-of-domain test set. More details can be found in [21].

The framework serves as a benchmark for various deep learning architectures operating on Wi-Fi RD data, including 3D CNNs (e.g., ResNet3D), Video Transformers (e.g., ViViT, TimeSformer), and CNN-RNN hybrids.

- **Top Performer**: The **CNN-GRU** model consistently outperformed other architectures. By utilising 2D convolutions for spatial feature extraction and Gated Recurrent Units for temporal sequence modeling, it achieved the highest accuracy of 95.18%.
- **Architectural Comparisons**: In general, models combining CNN-based spatial feature extraction with temporal modeling (such as GRU and LSTM) consistently outperformed standalone 3D CNNs. This highlights the suitability of CNNs for spatial feature encoding and the effectiveness of recurrent units for tracking the temporal sequence of gestures in RD representations.
- **ResNet3D Models**: The ResNet3D family demonstrated highly competitive results (with ResNet3D-10 achieving 92.53% accuracy as the next best performer) and was the first to converge during training. This underlines the value of residual connections when adapting video models to Wi-Fi RD data.
- **Transformer Models**: Transformer-based models (TimeSformer and ViViT) delivered moderate performance (81.45% and 82.89% accuracy, respectively). This moderate accuracy is attributed to the limited size of the dataset; however, their compactness and efficiency significantly outperformed much larger 3D CNNs on a per-parameter basis.

Table 5-2 Evaluation results of various deep learning architectures on the Wird-Gest framework

Model	Accuracy (%)	F1-Macro (%)	Parameters	GFLOPS
3D CNN	78.80	78.99	344,581	3.77
CNN-GRU	95.18	95.18	537,669	1.89
CNN-LSTM	91.81	91.81	587,077	1.89
CNN-RNN	77.83	78.12	438,853	1.88
ResNet3D-10	92.53	92.54	14,358,085	3.75
ResNet3D-18	91.81	91.86	33,162,565	7.15
ResNet3D-50	89.64	89.71	46,165,317	9.49
TimeSformer	81.45	81.36	202,629	0.08
ViViT	82.89	83.04	144,901	0.08

The data collection and processing methodologies for this algorithm were designed with strict adherence to user privacy and ethical standards:

- **Data Minimisation**: No personal data or identifiable biometric traits were saved during the data collection process.
- **Camera Usage**: Camera data was strictly utilised on a temporary basis solely to process and establish the ground truth labels for the gestures.
- **Data Deletion**: All video recordings used for labeling were immediately deleted after the ground truth was verified and are not part of the dataset.
- **Privacy-by-Design**: By relying purely on Wi-Fi Channel State Information rather than optical sensors, this gesture recognition approach intrinsically protects user privacy, eliminating the need for continuous visual surveillance.

6 Conclusions

This deliverable has presented the design and comprehensive evaluation of AI/ML-based algorithms for enhanced network performance in the context of the 6G-SENSES project, concluding the work carried out in Task 4.2. Building on the architecture and preliminary evaluations introduced in deliverables **D2.2** [1] and **D2.3** [2], this document has focused on developing and evaluating AI-driven solutions that enable intelligent, adaptive network operation across multiple domains and time scales.

At the architectural level, the deliverable demonstrated in Chapter 2 how the 6G-SENSES framework, aligned with 3GPP and O-RAN principles, supports the integration of AI/ML functionalities through the availability of rich sensing data and flexible deployment options. The combination of multi-RAT sensing, edge computing capabilities, and hierarchical control mechanisms enables the exploitation of contextual information for data-driven decision-making.

In Chapter 4, the deliverable presented and evaluated a set of algorithms targeting the efficient management and control of radio and storage resources in the RAN and Edge domains. These include DRL-based sensing-aware MAC scheduling, multi-agent DRL for ASRIS-assisted ISAC systems, an MRL approach for STAR-RIS optimisation, and location-aware MEC caching strategies. The results demonstrate that incorporating sensing information and learning-based decision-making can significantly improve resource utilisation, energy efficiency, and QoS performance, with representative gains including substantial reductions in PRB usage in sensing-aware scheduling, improved sum-rate performance in RIS-assisted ISAC scenarios, and notable reductions in energy consumption and cache miss rates through adaptive MEC caching.

The deliverable also delves into AI-driven approaches with an end-to-end, system-wide scope, enabling predictive modeling, slice orchestration, and intent-based network automation (Chapter 5). The proposed solutions include a GNN-based framework for end-to-end performance prediction in the RAN, Hierarchical DRL-based E2E slice orchestration for ISAC services, GenAI-driven intent-based RAN slicing, and an adaptive AI-driven framework combining centralised and FL-based distributed processing for trajectory prediction using range–Doppler data. In addition, application-level use cases, such as Wi-Fi-based gesture recognition, demonstrate how sensing capabilities can be leveraged to extract high-level contextual information from the environment. The results highlight the effectiveness of the GNN-based approach in accurately predicting network performance under dynamic conditions, the improved sample-efficiency and quality of obtained policies of the Hierarchical DRL-based orchestration, as well as effective automation and reduced operational complexity enabled by intent-driven and GenAI-based automation. Furthermore, the adaptive AI-driven framework demonstrates the ability to dynamically switch between centralised and FL-based processing, balancing sensing accuracy and transport network utilisation by reducing the need to transmit high-volume sensing data while maintaining effective trajectory prediction performance. Finally, the gesture recognition framework demonstrates high classification accuracy, highlighting the potential of wireless sensing for application-level intelligence.

A key contribution of this deliverable is the comprehensive evaluation of the proposed algorithms, considering not only system-level performance metrics such as QoS, resource utilisation, and energy efficiency, but also algorithmic aspects including convergence behavior and sample efficiency. The inclusion of realistic datasets, documented in Chapter 3, further strengthens the validity of the results and supports reproducibility.

Overall, the results presented confirm that AI/ML-driven approaches, when combined with sensing-aware architectures and flexible deployment mechanisms, can significantly enhance network performance and enable intelligent, adaptive operation in 6G environments. The proposed solutions provide practical insights

into the design and deployment of AI-native network functionalities, contributing to the realisation of efficient, scalable, and context-aware 6G systems.

7 References

[1] 6G-SENSES deliverable D2.2, “System architecture and preliminary evaluations”, March 2025, https://6g-senses.eu/wp-content/uploads/2025/11/2025-03-05-6G-SENSES_Deliverable_2_2_vf_pending_EU-review.pdf

[2] 6G-SENSES deliverable D2.3, " 6G-SENSES architecture evaluation and benchmarking", January 2026.

[3] 6G-SENSES deliverable D4.1, “Initial SoTA and design of wireless edge caching solutions”, January 2025, http://6g-senses.eu/wp-content/uploads/2025/02/2025-01-15-6G-SENSES_Deliverable_4.1_MAIN_FINAL.pdf

[4] 6G-SENSES deliverable D4.2, “Final design and implementation of wireless edge caching solutions”.

[5] M. Petri and N. Maletic, "A mmWave JCAS System for Real-Time High-Data Rate Communication and RADAR Sensing," in IEEE Access, vol. 13, pp. 117700-117715, 2025, doi: 10.1109/ACCESS.2025.3586645.

[6] R. S. Sutton and A. G. Barto, Reinforcement Learning: An Introduction, 2nd ed. Cambridge, MA, USA: MIT Press, 2018.

[7] I. Afolabi, T. Taleb, K. Samdanis, A. Ksentini, and H. Flinck, ‘‘Network slicing and softwarisation: A survey on principles, enabling technologies, and solutions,’’ IEEE Commun. Surveys Tuts., vol. 20, no. 3, pp. 2429–2453, 3rd Quart., 2018.

[8] W. Masson, P. Ranchod, and G. Konidaris, “Reinforcement Learning with Parameterised Actions”, AAAI, vol. 30, no. 1, Feb. 2016.

[9] Telecom Italia, “Milano Grid,” 2015.

[10] V. Mnih et al., “Human-level control through deep reinforcement learning,” Nature, vol. 518, no. 7540, pp. 529–533, Feb. 2015.

[11] M. Ferriol-Galmés et al., "RouteNet-Fermi: Network Modeling With Graph Neural Networks," in IEEE/ACM Transactions on Networking, vol. 31, no. 6, pp. 3080-3095, Dec. 2023, doi: 10.1109/TNET.2023.3269983.

[12] Wen, D., Zhou, Y., Li, X., Shi, Y., Huang, K. and Letaief, K.B., 2024. A survey on integrated sensing, communication, and computation. IEEE Communications Surveys & Tutorials.

[13] Wang, J., Varshney, N., Gentile, C., Blandino, S., Chuang, J. and Golmie, N., 2022. Integrated sensing and communication: Enabling techniques, applications, tools and data sets, standardisation, and future directions. *IEEE Internet of Things Journal*, *9*(23), pp.23416-23440.

[14] Han, C., Wu, Y., Chen, Z., Chen, Y. and Wang, G., 2024. THz ISAC: A physical-layer perspective of terahertz integrated sensing and communication. *IEEE communications magazine*, *62*(2), pp.102-108.

[15] Zhu, X., Liu, J., Lu, L., Zhang, T., Qiu, T., Wang, C. and Liu, Y., 2024. Enabling intelligent connectivity: A survey of secure ISAC in 6G networks. *IEEE Communications Surveys & Tutorials*, *27*(2), pp.748-781.

[16] Megahed, A., Abd El-Haleem, A.M., Elmesalawy, M.M. and Ibrahim, I.I., 2025. Deep learning optimisation of STAR-RIS for enhanced data rate and energy efficiency in 6G wireless networks. *Scientific reports*, *15*(1), p.26311.

[17] Zhi, K., Pan, C., Ren, H., Chai, K.K. and Elkashlan, M., 2022. Active RIS versus passive RIS: Which is superior with the same power budget?", *IEEE Communications Letters*, *26*(5), pp.1150-1154.

[18] Kurma, S., Singh, K., Mumtaz, S., Tsiftsis, T.A. and Li, C.P., 2024. Resource optimisation in active-STAR-RIS-aided THz ISAC systems with DDA modulation: A machine-learning approach. IEEE Transactions on Wireless Communications, 23(10), pp.15291-15307.

[19] Ahmed, M., Wahid, A., Laique, S.S., Khan, W.U., Ihsan, A., Xu, F., Chatzinotas, S. and Han, Z., 2023. A survey on STAR-RIS: Use cases, recent advances, and future research challenges. IEEE Internet of Things Journal, 10(16), pp.14689-14711.

[20] Katwe, M., Singh, K., Clerckx, B. and Li, C.P., 2023. Improved spectral efficiency in STAR-RIS aided uplink communication using rate splitting multiple access. IEEE Transactions on Wireless Communications, 22(8), pp.5365-5382.

[21] J. Sanson, R. Shah, Y. Zhu, R. Rosales, V. Frascolla, "Wird-gest: gesture recognition in the real world using active range-doppler Wi-Fi sensing COTS hardware" in IEEE International Conference on Communications (ICC), 24–28.05.2026, Glasgow, UK.

8 Acronyms

Acronym	Description
3GPP	3rd Generation Partnership Project
5G	5th Generation Mobile Network
5G NR	5G New Radio
6G	6th Generation Mobile Network
ACK	Acknowledgement
AI	Artificial Intelligence
API	Application Programming Interface
ARIS	Active Reconfigurable Intelligent Surface
ASRIS	Active Simultaneously Transmitting and Reflecting Reconfigurable Intelligent Surface
BI	*Barkhausen Institut* (6G-SENSES Beneficiary)
BR	BubbleRAN (6G-SENSES Beneficiary)
BS	Base Station
BW	Bandwidth
CG	Cloud Gaming
CIFO	Closest In Farthest Out
CNN	Convolutional Neural Network
COTS	Commercial Off-The-Shelf
CQI	Channel Quality Indicator
CSI	Channel State Information
CU	Central Unit
dBaaS	Database as a Service
DDA	Dynamic Delay Alignment
DDPG	Deep Deterministic Policy Gradient
DL	Downlink
DNN	Deep Neural Network
DPS	Doppler power Spectra
DQN	Deep Q-Network
DRL	Deep Reinforcement Learning
DSCP	Differentiated Services Code Point
DT	Digital Twin
E2E	End-to-End
E2SM	E2 Service Model
E2SM- KPM	E2 Service Model – Key Performance Measurements
E2SM-RC	E2 Service Model – RAN Control
ECN	Explicit Congestion Notification

FDD	Frequency Division Duplex
FH	Fronthaul
FIFO	First In First Out
GenAI	Generative AI
GRU	Gated Recurrent Units
GenAI	Generative Artificial Intelligence
GFBR	Guaranteed Flow Bit Rates
gNB	Next Generation Node B
GNN	Graph Neural Network
HDQN	Hierarchical Deep Q-Network
HPD	Human Presence Detection
HQL	Hierarchical Q-learning
IASA	Institute of Accelerating Systems and Applications (6G-SENSES Beneficiary)
IHP	*IHP – Leibniz Institut für innovative Mikroelektronik* (6G-SENSES Beneficiary)
INT	*Intel Deutschland GmbH* (6G-SENSES Beneficiary)
IQ	In-Phase and Quadrature
ISAC	Integrated Sensing and Communication
KPI	Key Performance Indicator
LDPC	Low-Density Parity-Check
LFU	Least Frequently Used
LLC SM	Logical Link Control Service Model
LLM	Large Language Model
LOS	Line-of-Sight
LRU	Least Recently Used
LTF	Long Training Field
MAC	Medium Access Control
MADDPG	Multi-Agent Deep Deterministic Policy Gradient
MAE	Mean Absolute Error
MARL	Multi-Agent Reinforcement Learning
MCS	Modulation and Coding Scheme
MDP	Markov Decision Process
MEC	Multi-access Edge Computing
ML	Machine Learning
MLP	Multilayer Perceptron
mmWave	Millimeter Wave
MPNN	Message Passing Neural Network
MRL	Meta Reinforcement Learning
N3IWF	Non-3GPP InterWorking Function

NACK	Negative Acknowledgement
Near-RT RIC	Near Real Time Radio Intelligent Controller
Non-RT RIC	Non Real Time Radio Intelligent Controller
NR	New Radio
NTU	Nottingham Trent University (6G-SENSES Beneficiary)
OAI	OpenAirInterface
O-CU	O-RAN Central Unit
O-DU	O-RAN Distributed Unit
OFDM	Orthogonal Frequency-Division Multiplexing
O-FH	O-RAN Fronthaul
O-RAN	Open Radio Access Network
O-RU	O-RAN Radio Unit
PF	Proportional Fairness
PHY	Physical Layer
PPO	Proximal Policy Optimisation
PRB	Physical Resource Block
QL	Q-Learning
QoS	Quality of Service
RAN	Radio Access Network
rApp	non-real-time RAN Application
RAT	Radio Access Technology
RB	Resource Block
RBIS	Reinforcement-Based Intelligent Scheduling
RC	RAN Control
RD	Range-Doppler
ReLU	Rectified Linear Unit
RIC	RAN Intelligent Controller
RIS	Reconfigurable Intelligent Surface
RL	Reinforcement Learning
RLC	Radio Link Control
RNN	Recurrent Neural Network
RR	Round Robin
RRC	Radio Resource Control
RU	Radio Unit
SCS	Subcarrier Spacing
SDN	Software Defined Networking
SDR	Software Defined Radio
SINR	Signal-to-Interference-plus-Noise Ratio

SLA	Service Level Agreement
SM	Service Model
SMO	Service Management and Orchestration
SNR	Signal-to-Noise Ratio
SNS JU	Smart Networks and Services Joint Undertaking
SP	Strict Priority
sRU	Sensing Radio Unit
STAR-RIS	Simultaneously Transmitting and Reflecting RIS
TD3	Twin Delayed Deep Deterministic Policy Gradient
TDD	Time Division Duplex
THz	Terahertz
TLV	Type-Length-Value
ToC	Table of Contents
ToS	Type of Service
TUBS	*Technische Universität Braunschweig* (6G-SENSES Beneficiary)
UC	University of Cantabria (6G-SENSES Beneficiary)
UE	User Equipment
UL	Uplink
UPF	User Plane Function
U-Plane	User-Plane
URLLC	Ultra-Reliable Low Latency Communication
VU	Vehicular Unit
VNF	Virtual Network Function
WAT	Wireless Access Technology
WEC	Wireless Edge Caching
WG	Work Group
Wi-Fi	Wireless-Fidelity
WP	Work Package
xApp	eXtended Application
XR	Extended Reality