O-RAN xApps: Survey and Research Challenges

Arman Elyasi, Andrew Ashdown, K M Rumman, Francesco Restuccia

[a]*Institute for the Wireless Internet of Things, Northeastern University, United States*

Abstract

As the Open Radio Access Network (O-RAN) paradigm is transforming the cellular landscape, third-party Extensible Applications (xApps) deployed in the Near-Real-Time RAN Intelligent Controller will play a critical role in facilitating next-generation cellular networks. Such networks will rely heavily upon robust data-driven Artificial Intelligence and Machine Learning (AI/ML) solutions that can perform fine-grained RAN control at the millisecond timescale. As xApps represent the point of contact between these advanced AI/ML solutions and the RAN itself, understanding these important O-RAN applications is essential for researchers in the wireless domain and will be critical to the full-scale integration of AI/ML into the next generation of cellular networks. Motivated by a lack of relevant and up-to-date surveys on this topic, we provide a comprehensive overview of xApps, including important background information, current xApp development techniques, a discussion of xApps and AI/ML, state-of-the-art use cases, vulnerabilities in the xApp architecture, and other critical research challenges. We summarize the plethora of available information on this topic by highlighting the key issues and presenting a cohesive message that provides the necessary background to the reader and isolates the critical research issues that currently remain unaddressed. As such, this survey is a one-stop-shop for researchers seeking a solid grasp of the state-of-the-art and a clear outline of the central research challenges relevant to xApps in cellular networks.

Keywords: O-RAN, Cellular Networks, xApps, Artificial Intelligence, Machine Learning

1. Introduction

The rapid evolution of Fifth Generation (5G) cellular technology has significantly transformed the telecommunications landscape, which enables a

Email addresses: elyasi.a@northeastern.edu (Arman Elyasi), ashdown.a@northeastern.edu (Andrew Ashdown), rumman.k@northeastern.edu (K M Rumman), frestuc@northeastern.edu (Francesco Restuccia)

myriad of Future Generation (FutureG) applications that demand low latency and high performance [1]. This transformation is driven by growing user expectations that escalate the need for network operators to support not only Enhanced Mobile Broadband (eMBB) services, but also Ultra Reliable Low Latency Communications (URLLCs) and Massive Machine Type Communications (mMTCs), which are critical for emerging technologies like Augmented Reality (AR), Virtual Reality (VR), and Internet of Things (IoT) devices [2]. Traditional network architectures often struggle with scalability, interoperability, and the dynamic allocation of resources required in complex wireless environments [1]. In response to these challenges, paradigms such as the Open Radio Access Network (O-RAN) initiative have emerged, promoting openness and interoperability in the cellular ecosystem [3].

The main idea behind O-RAN is to break up the traditional ran functions into modularized software components called Virtual Network Functions (VNFs), which are connected through open and standardized interfaces [4]. This shift toward an open architecture where companies can focus on developing a single ran functionality and where networks can be composed of equipment coming from various vendors will foster a highly competitive yet extremely innovative marketplace, thus fueling network performance and increasing the user experience. Two central components of the O-RAN architecture are the RAN Intelligent Controllers (RICs), which perform management and control of the network in Near-Real-Time (Near-RT RIC) and Non-Real-Time (Non-RT RIC), respectively [5]. The near-RIC focuses on latency-sensitive tasks, enabling network optimization on a timescale between 10 milliseconds and 1 second, while the Non-RT RIC-RIC handles long-term analytics, policy management, and Artificial Intelligence and Machine Learning (AI/ML) model training on a timescale greater than 1 second [1].

Within this framework, intelligence can be embedded in the O-RAN in the form of Extensible Applications (xApps) deployed inside the near-RIC, enabling on-demand, closed-loop control of the ran resources and functionalities [5, 6, 7]. Given their centrality to the incorporation of AI/ML into the network, xApps represent a transformative step in achieving intelligent and programmable Radio Access Networks (RANs). Existing xApps have already been shown to enable dynamic optimization of network functionalities like Traffic Steering (TS) [8, 9], spectrum sensing [10], and resource allocation [11], etc. Their modular and open design, along with their strategic placement inside the near-RIC, makes them a critical intersection between third-party software applications and commercial network functionalities. Thus, as 5G networks are being deployed at a rapid pace, xApps will provide the agility and intelligence required to effectively address the challenges that arise and ensure that network performance is maximized.

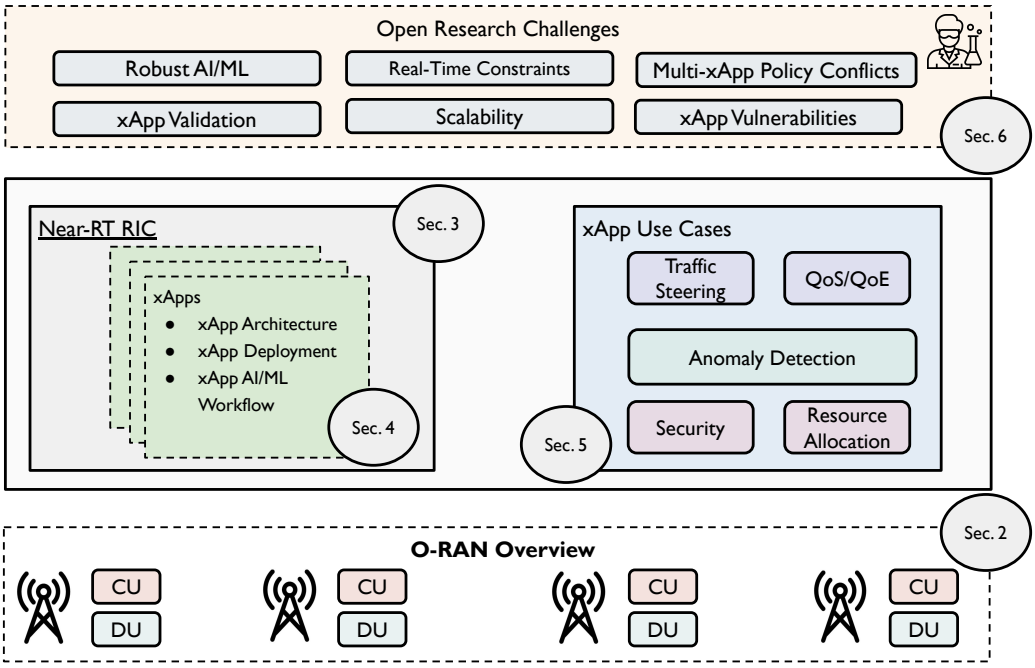

Figure 1: Survey Structure

1.1. Summary of Novel Contributions

Many papers have been published to explore O-RAN and its foundational principles. Existing surveys provide valuable insights into O-RAN architecture and Near-RT RIC functionalities [1, 5, 12, 13]. For instance, [1], examines RAN components, AI/ML workflows, and future research challenges or [12] which covers the development and deployment of xApps. Bonati et al. in [5] provides information about the close control loop within O-RAN. [13] presents a detailed overview of the O-RAN architecture. With respect to the previous surveys, we provide a comprehensive overview of xApps within the O-RAN architecture, including important background information, a discussion of the xApp development and deployment processes, AI/ML architectures and security, the interaction between xApp and AI/ML as well as current xApp use cases. We also outline vital open research challenges that currently remain unaddressed and highlight avenues for potential solutions to these pressing questions. A high-level overview of the topics we address and the respective sections in which we address them can be seen in Figure 1. All in all, this paper represents a one-stop-shop for researchers who are seeking a solid grasp of the state-of-the-art and a clear outline of the central research challenges relevant to xApps in next generation cellular networks. In section 2, we overview the O-RAN architecture, the components. Section 3 provides a detailed view of Near-RT RIC's architecture, services and APIs. xApp's architecture, deployment and the AI/ML workflow has been

presented in Section 4. Section 5 provides a detailed and comprehensive overview of xApp's applications within O-RAN and finally in section 6, we outline the important research challenges which should be addressed in the future.

2. O-RAN Overview

In this section, we review the O-RAN architecture in detail. Moreover, we outline the role of each component in the ran and highlight the open interfaces that connect the various disaggregated units.

2.1. RAN Evolution

Due to increasing network demands, traditional, black-box RAN have become inadequate as a result of their inability to address the issues of network flexibility and customization [9]. Particularly, with the rise of 5G technology and its promise of low latency and high throughput, these limitations have become more apparent; hence the need for O-RAN, which enables unprecedented flexibility via RANdisaggregated. Below we describe the development of the RAN as it progressed from the black-box, vendor-specific deployments of previous generations, to the customizable white-box O-RAN paradigm that will enable FutureG.

In traditional RAN, Radio Units (RUs) are connected to a Baseband Unit (BBU) which in turn forwards the signals to a Centralized Unit (CU). Each CU is connected to multiple BBUs and are tasked with forwarding messages between the lower-level ran and the core network. The problem with this RAN is that the architecture is proprietary; this means that an RU from one vendors won't be compatible with a BBU from another vendor. These limitations result in a network that is inflexible and expensive - this is known as vendor lock-in, where a single company provides an end-to-end, black-box RAN solution, making network customization virtually impossible.

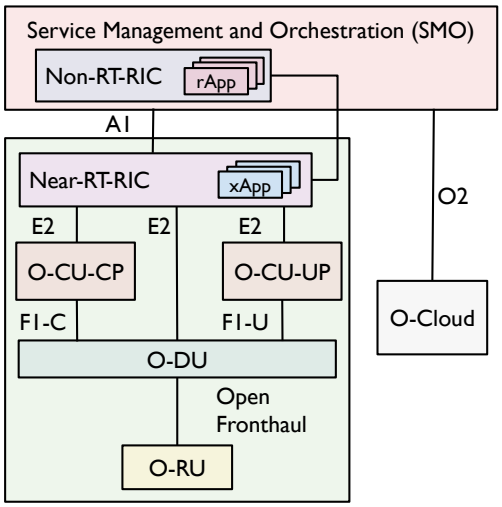

Figure 2: O-RAN Architecture

Cloud-RAN (C-RAN) was proposed to overcome some of the limitations imposed by traditional black-box RANs. In C-RAN, the BBU is moved to a centralized location instead of co-locating a BBU at each cell site. While

this helps to reduce the number of BBUs per RU in comparison to traditional RANs, it still results in vendor lock-in. Thus, researchers proposed another architecture known as Virtualized-RAN (vRAN).

In RAN, instead of a BBU, there is Virtual Distributed Unit (VDU) and rather than being connected to a CU, it is connected to a Virtual Centralized Unit (VC). The VDU and VC run on Commercial-off-the-Shelf (COTS) hardware and are known as VNFs. These provide the same RAN functionalities as their black-box counterparts, but because they run on COTS hardware vendors could focus on developing software solutions instead of purpose-built hardware. In this architecture, the RU remains a proprietary solution. Thus, taking RAN to the next step led to the proposal of the O-RAN architecture. By introducing open and standardized interfaces, both the RU and BBU could be implemented by different network vendors and the and the vendor lock-in dilemma could be eliminated.

To fully address these constraints, the O-RAN Alliance, an industry consortium comprising major operators, vendors, and research organizations, introduced the O-RAN architecture. By defining open and standardized interfaces, O-RAN enables multi-vendor interoperability, allowing different vendors to implement RUs, Distributed Unit (DU)s, and CUs independently, effectively eliminating vendor lock-in. Additionally, O-RAN promotes the integration of AI/ML-driven intelligence to enhance network efficiency and automation. The O-RAN Alliance has established multiple working groups to define and standardize the architecture:

- *Working Group 1 (WG1*) focuses on O-RAN use cases and architecture.
- *Working group 2 (WG2)* oversees the Non-RT RIC and the A1 interface, R1 interface and AI/ML workflow
- *Working group 3 (WG3)* defines the Near-RT RIC and the E2 interface, E2 Application Protocol and E2 Services
- *Working group 4 (WG4)* defines the open fronthaul interfaces and Cooperative Transport Interface Transport Management Procedures.
- *Working group 5 (WG5)* is responsible for defining the Open F1, W1, E1, X2 interfaces.
- *Working group 6 (WG6)* works on O-Cloud and Orchestration and virtualization of O-RAN.
- *Working group 7 (WG7)*, which focuses on designing open and standardized hardware platforms for O-RAN-compliant deployments.
- *Working group 8 (WG8)* is responsible for designing O-DU and O-CU architecture.
- *Working group 9 (WG9)* manages interfaces for Transport Network Elements and X-haul Transport.
- *Working group 10 (WG10)* deals with O1 Interface Operations, Administration and Maintenance (OAM) specifications.

• *Working group 11 (WG11)* is responsible for security and risk assesment of O-RAN.

These working groups collectively form the foundation of O-RAN's key principles: *Disaggregation*, *Openness*, *Virtualization*, and *Programmability*y, which fundamentally distinguish it from traditional, proprietary RAN architectures. By using these principles, O-RAN provides an open, multi-vendor ecosystem, where intelligent automation and cloud-native virtualization replace vendor-specific implementations. The following sections will explore how these principles drive the transformation of modern RAN systems.

2.2. Disaggregation

O-RAN offers a more flexible and multi-vendor network for FutureG networks. In this regard, O-RAN introduced *disaggregation* in the new architecture, which splits the functionalities of the base station of the network between different nodes [14].

As depicted in Figure 2, O-RAN is divided by Centralized Unit Control Plane (CU-CP), Centralized Unit User Plane (CU-UP), Distributed Unit (DU), Radio Unit (RU), Near-RT RIC, Non-RT RIC, Service Management and Orchestration (SMO). These logical nodes are connected to each other using different interfaces (E2, A1, O1, etc). CU-CP is responsible for hosting Radio Resource Control (RRC) and the control plane part of the Packet Data Convergence Control (PDCP), which handles data packets exchanged between User Equipment (UE) and the core network [15]. CU-UP hosts the user plane part of the PDCP protocol and the SDAP protocol [15]. DU is a logical node that is responsible for radio resources [16] and contains High-PHY layers based on a functional split of the lower layer and may be implemented using virtualized or nonvirtualized methods [17, 18]. RU features lower physical layer such as signal processing and Fast Fourier Transform (FFT) [19] and Radio Frequency (RF) processing based on a functional split of the lower layer [17]. O-Cloud is a cloud platform, which accelerates signal processing operations and employs supporting software components to virtualized the O-RAN architecture [18, 20].

2.3. Programmability and Intelligence

O-RAN has enabled intelligence by introducing two main components called as Non-RT RIC and Near-RT RIC. RICs are designed to enable greater flexibility, efficiency and adaptability in managing the radio access network by utilizing real and non-real-time analytics. These two are connected to each other via A1 interface and mainly responsible for implementing the AI/ML models for managing and optimizing the network in different timescales (from 10 mili seconds to minutes). These AI/ML models are trained on the data, which are gathered from different Key Performance Measurements (KPMs)

for handover management, spectrum sensing, slicing, etc. In the following we provide some details about the Non-RT RIC and Near-RT RIC.

2.3.1. ***Non-real-time RIC***

As shown in figure 2, the Non-RT RIC is located inside the SMO, which is responsible for orchestrating and implementing the AI/ML models for optimization and to perform for a timescale larger than 1s [21]. The Non-RT RIC uses advanced analytics and AI/ML techniques to identify and implement actions that optimize the RAN. It relies on SMO functionalities, such as data collection and configuration of O-RAN nodes, while interacting through the O1 and O2 interfaces to support these processes [21].

The Non-RT RIC is defined by two main components: the *rApps* and *Non-RT RIC framework* [22].

rApps are the third-party applications defined by the O-RAN alliance and hosted by Non-RT RIC for implementing intelligent RAN operation. rApps are connected to the Non-RT RIC through R1 interface and are responsible for policy guidance, information enrichment, configuration management, AI/ML implementation, data analysis, providing training services and generating A1 policies [22, 23].

The Non-RT RIC framework is composed of various Non-RT RIC functional components. Non-RT RIC framework functions include rApp supporting functions, A1 functions, AI/ML Monitoring and Workflow functions [22]. The R1 interface provides access to a comprehensive set of R1 services, which include service management and exposure services for handling service registration, discovery, authentication. Additionally, data management and exposure services allow data producers to share network telemetry, performance statistics, and other operational insights with data consumers. The R1 interface also supports A1-related services, which enables communication with the Near-RT RIC via the A1 interface for policy-based control. Moreover, it provides access to AI/ML workflow services for rApps to integrate with ML pipelines for tasks such as model training, inference, and continuous learning. The R1 interface also supports RAN OAM-related services for having access to O1 and open fronthaul M-plane interfaces. Furthermore, O2-related services establish interactions with the O2 interface, supporting cloud-based O-RAN infrastructure management. Lastly, rApp management services facilitate the deployment, execution, and life-cycle management of rApps within the Non-RT RIC environment [23]. Through these capabilities, the Non-RT RIC, in conjunction with the R1 interface, enable open, programmable, and intelligent RAN management.

2.3.2. ***Near-real-time RIC***

The other component which adds closed-loop and intelligence control to the network is the Near-RT RIC to perform in time-scale less than 10ms. The

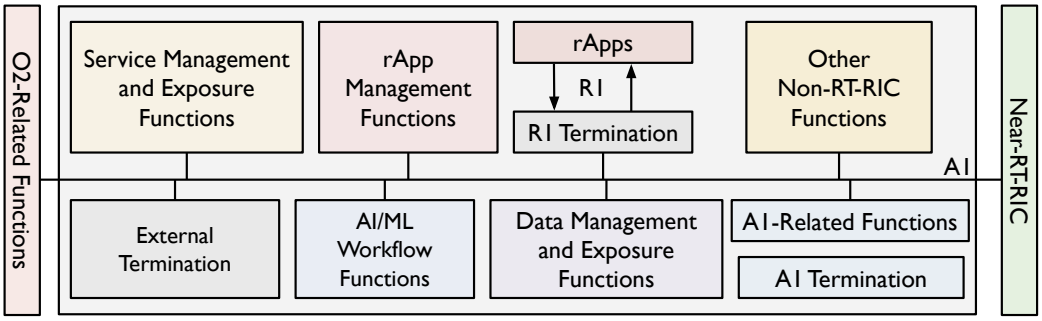

Figure 3: Non-RT RIC Architecture [23]

Near-RT RIC is connected to CU-UP,CU-UP and DU via the E2 Interface. As depicted in Figure 4, the Near-RT RIC consists of different components to provide the appropriate mechanisms for the implementation of AI/ML models in xApps. The components are connected together via different interfaces, which we shall discuss in the next section.

The Near-RT RIC hosts xApps. Specialized applications designed to enhance RAN performance through AI/ML-driven optimizations, which are trained by large datasets gathered from different nodes of the network. To support xApps, Near-RT RIC consists of differnet components such as: Database, Shared Data Layer (SDL), E2 Termination, xApp Subscription Management. A detailed discussion of these components and their respective interfaces will be provided in Section 4.3.

2.4. Interfaces

The O-RAN Alliance introduces open interfaces to replace proprietary protocols and to have a more intelligent and open ecosystem. These open interfaces, such as E2, A1, O1 and O2, enable the different components from different vendors to be deployed in RAN. They also enable the real-time optimization, dynamic resource allocation and AI/ML implementation by facilitating the interaction between E2 nodes and the intelligent controllers. In the following, we will discuss them in detail.

2.4.1. A1 Interface

A1 interface is an open interface that connects Non-RT RIC and Near-RT RIC in the O-RAN architecture and enables A1 policies and Enrichment Information (EI). The Non-RT RIC can define and manage policies that are provided to the Near-RT RIC over the A1 interface to achieve the overall goal of RAN intent.

The Non-RT RIC and SMO have access to internal and external sources of additional information which is not available for the other RAN nodes and components. This information can be delivered continuously or needed

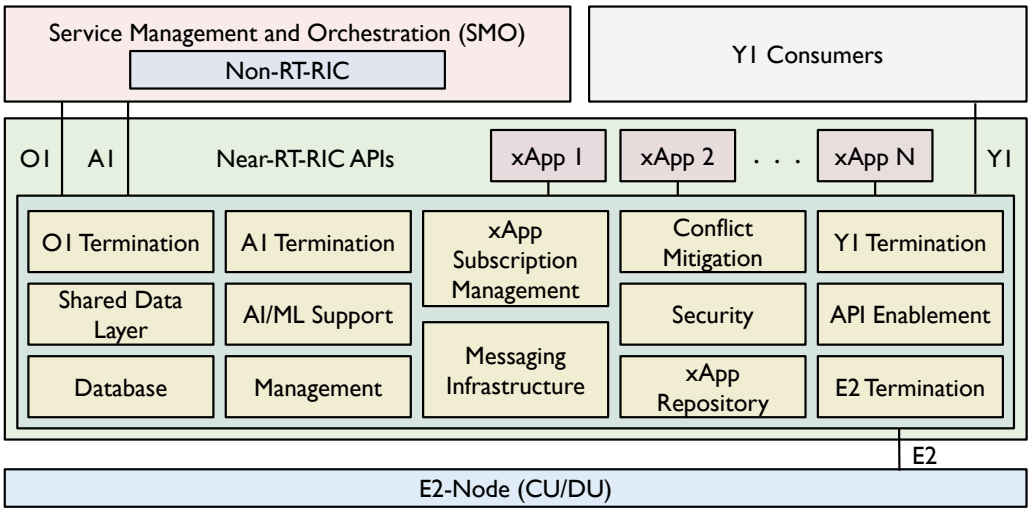

Figure 4: Near-RT RIC Architecture [28]

based by the A1 interface to the Near-RT RIC and xApps to improve the RAN optimization process [24, 25].

2.4.2. E2 Interface

E2 interface is a logical interface between the Near-RT RIC and E2 Nodes such as O-CU-CP, O-CU-UP, O-DU and O-eNB [26]. It accelerates the process of sending data from the E2 nodes to the Near-RT RIC and xApps for optimization and close-loop control of RAN [27] and enables the Near-RT RIC to control the functionalities of the E2 nodes [1]. Unlike the A1 interface, which focuses on policy-driven control from the Non-RT RIC, the E2 interface is designed for near-real-time monitoring and control of RAN components. It provides the Near-RT RIC with direct access to critical RAN metrics and allows it to issue control actions in response to dynamic network conditions.

Through the E2 interface, xApps can collect performance data, subscribe to events, and execute optimization actions. This interface is one of the pivotal components in enabling closed-loop control within the RAN by ensuring low-latency responses to changing network requirements while maintaining seamless connectivity and performance.

2.4.3. O2 Interface

O2 interface is the other key components of the RAN. As shown in Figure 2, it provides the connection between SMO and O-Cloud [29]. It enables management of O-Cloud and life cycle management of the Cloud-Native Network Functions (CNFs) running on the O-Cloud. CNFs are containerized network functions designed to operate in a cloud-native environment to leverage mi-

croservices and Kubernetes for deploying xApps. Overall, managing infrastructure, abstract resource NFs could be achieved via the O2 interface [29].

2.4.4. Y1 Interface

The Y1 interface is another key component of the O-RAN. It enables the Near-RT RIC to be connected to the Y1 consumers. Y1 consumers receive the RAN Analytics Information (RAI) from the Near-RT RIC via the Y1 interface [28, 30].

By implementing a scalabe and modular architecture, O-RAN has been able to manage and optimize the modern networks by deploying intelligent solutions. The key component to accomplishing this goal is the Near-RT RIC. In the following section, we provide a detailed examination of the Near-RT RIC, exploring its architecture, services, and framework, and its role in enhancing network intelligence and optimizing RAN performance.

3. Near-RT RIC

As mentioned above, the Near-RT RIC is the key component to optimize and manage the O-RAN within 10ms to 1s. It hosts xApps, extensible applications to perform close-loop controlling of the RAN ecosystem. Near-RT RIC is connected to the Non-RT RIC via the A1 interface. A1 interface allows Near-RT RIC to continuously refine its AI-driven optimization by receiving feedback from Non-RT RIC. As shown in Figure 4, The Near-RT RIC consists of several components to achieve this goal, which will be discussed in the following:

3.1. Near-RT RIC Components

• *O1 Termination*: The Near-RT RIC platform communicates with the SMO via the O1 interface and exposes O1-related management services. Within this framework, the Near-RT RIC acts as a Management Service (MnS) producer, while the SMO serves as the MnS consumer [28].
• *Shared Data Layer (SDL):* SDL services could be used to modify and expose the database for xApps by this functionality.
• *Database:* Database contains all the information related to UEs (UE-NIB) and RAN components (R-NIB) and provides them for the Near-RT RIC and xApps [28].
• *A1 Termination*: A1 termination enables the Near-RT RIC and the Non-RT RIC to receive and send messages regarding the A1 policies and EI.
• *AI/ML Support*: This component enables the Near-RT RIC and xApps to manage and use the AI/ML models based on the specific use case. Further details will be discussed the next section.
• *xApp Subscription Management*: This component provides additional support from the Near-RT RIC for the xApps. It manages xApp subscriptions to

E2 Nodes, enforcing access control policies, merging identical subscriptions, and auditing with corrective actions.

- *Messaging Infrastructure:* Messaging Infrastructure enables low-latency message delivery within the Near-RT RIC platform by supporting endpoint registration, discovery, and deletion, providing APIs for sending and receiving messages, supporting multiple messaging modes (e.g., point-to-point and publish/subscribe), enabling message routing. This component ensures to delete the outdated messages and not to lose any information delivering to other nodes.
- *Conflict Mitigation:* Within the Near-RT RIC framework, Conflict Mitigation is essential for identifying and resolving overlapping or contradictory requests from multiple xApps. Since xApps operate independently to optimize specific network performance metrics by modifying system parameters, their actions can sometimes interfere with one another. To address this, the platform continuously monitors interactions, detects potential conflicts, and implements resolution strategies to ensure seamless coordination, preventing network instability and optimizing overall performance.
- *Security:* This functionality is designed to prevent adversarial xApps from manipulating RAN functions or having access to the network information.
- *xApp Repository Function* : The xApp Repository Function enables the Near-RT RIC by overseeing a pool of eligible xApps for A1 Termination, ensuring that A1 policies and updates are assigned to the appropriate xApps based on policy classification and operator-defined guidelines. Furthermore, it implements access control measures for xApps requesting A1-EI types, ensuring adherence to operator policies and security standards.
- *Y1 Termination* : It facilitates the exposure of RAN analytics from the Near-RT RIC to Y1 consumers.
- *API Enablement*: According to [28], the Near-RT RIC APIs can be related to E2-related services, A1-related services, Management related services, and SDL services. In particular, the API enablement services include: maintaining a repository and registry for APIs, facilitating the discovery of registered APIs, and ensuring secure access through xApp authentication. Additionally, they support generic subscription and event notification mechanisms while implementing measures to prevent compatibility conflicts between xApps and the services they interact with, ensuring seamless integration and interoperability within the platform.
- *Management:* This functionality ensures overall Near-RT RIC operations, including xApps lifecycle management and resource allocation.
- *E2 Termination:* This functionality facilitates the termination of the E2 interface by managing Stream Control Transmission Protocol (SCTP) connections from E2 Nodes and routing messages between xApps and E2 Nodes. It decodes incoming Abstract Syntax Notation One (ASN.1) messages (a

standardized data representation format used for encoding and decoding messages) to extract relevant information, determine message types, and manage E2 connectivity-related communications efficiently. Additionally, it manages E2 Setup Requests, informs xApps about the RAN functions supported by an E2 Node based on E2 Setup and RIC Service Update procedures, and notifies newly connected E2 Nodes of the accepted functions.

E2 termination also serves as the foundational layer for enabling higher-level E2 services. E2 services leverage the established connection to facilitate real-time monitoring, control, and optimization of RAN behavior.

3.2. E2 Service Models (E2SM)

E2SMs define the format and meaning of information exchanged between the Near-RT RIC and E2 Nodes. O-RAN Alliance WG3 has standardized E2SM Key Performance Measurements (KPM), E2SM RAN Control (RC), E2SM Cell Configuration and Control (CCC) and E2SM Network Interfaces (NI).

• **E2SM-KPM**: This service model allows the Near-RT RIC to periodically access performance measurements from DU and CU [31].

• **E2SM-RC**: It facilitates control operations within xApps using E2 control procedures. E2SM-RC can control radio bearer, radio resource allocation, mobility, radio access, dual connectivity, carrier aggregation (CA), idle mode mobility and multiple action control [32].

• **E2SM-CCC**: According to [33], E2SM-CCC supports Node level and Cell level configuration in E2 nodes.

• **E2SM-NI**: E2SM-NI defines the framework for managing NIs within an E2 Node that terminates the E2 interface, It enables the Near-RT RIC to access and interact with network interfaces, modify incoming and outgoing messages for optimization, and execute policies that dynamically influence network behavior [34]. To provide a reliable transport mechanism, E2 Application Protocol (E2AP) has been introduced by the O-RAN Alliance. E2AP ensures reliable transmission of E2SM-defined message, session management, and control exchanges. In the following we will delve in to its services and functionalities.

3.3. E2 Application Protocol (E2AP)

E2AP is responsible for facilitating message exchange between the Near-RT RIC and E2 Nodes over the E2 interface [35]. E2AP acts as the transport mechanism to ensure reliable communication by managing the delivery of signaling and control messages between these entities. To enable the use of various E2SMs, E2AP provides key services, including **REPORT**, **INSERT**, **CONTROL**, **POLICY**, and **QUERY**, each supporting different aspects of RAN monitoring, control, and policy enforcement [1].

• *REPORT*: The REPORT service starts with the Near-RT RIC setting up a RIC Subscription in the E2 Node to specify the type of data or events the E2 Node should monitor and report. This subscription defines the RIC Trigger Events, which are conditions that determine when the E2 Node must send information to the Near-RT RIC. . When such an event occurs, the E2 Node prepares an Report, which contains the requested data, such as performance metrics, network state, or event notifications. Before sending this Report, the E2 Node ensures that any ongoing RIC actions are completed. It then transmits the RIC INDICATION message to the Near-RT RIC, including the requested REPORT information and the originating Request ID. [26].
• *INSERT*: This service in E2AP allows the Near-RT RIC to pause certain procedures in an E2 Node and decide what happens next based on specific conditions. It starts when the Near-RT RIC sets up a RIC Subscription on the E2 Node, defining an INSERT action along with a Time to Wait timer and instructions on what to do after the timer expires. If necessary, the Near-RT RIC can modify this subscription later. When an event matching the trigger conditions occurs in the E2 Node, it pauses the related procedure for the specified Time to Wait period after completing any previous RIC actions. The E2 Node then notifies the Near-RT RIC by sending a RIC INDICATION message about the paused procedure and the original request.

The next steps depend on whether the Near-RT RIC sends a RIC CONTROL REQUEST before the timer runs out. If the request arrives in time, the E2 Node follows the new control instructions. If the Time to Wait timer expires before receiving any instructions, the E2 Node follows the predefined action. If set to Continue, the paused procedure resumes as normal. If set to Halt, the procedure is terminated, and no further actions are taken. In both cases, if the Near-RT RIC tries to send a RIC CONTROL REQUEST after the timer has expired, the E2 Node rejects it with a RIC CONTROL FAILURE message.
• *CONTROL*: The CONTROL service in E2AP allows the Near-RT RIC to send control commands to an E2 Node based on detected events, either from a previous RIC Indication or an internal trigger. Upon detection, the Near-RT RIC sends a RIC Control Request, optionally requiring acknowledgment and activating a TRICcontrol timer if needed. The E2 Node cancels any active Time to Wait timer, then either initiates or resumes the procedure. If successful, it responds with a RIC Control Acknowledge message; if unsuccessful, it sends a RIC Control Failure message explaining the reason. Finally, the Near-RT RIC cancels the TRICcontrol timer upon receiving a response, ensuring efficient execution and feedback for network control.
• *POLICY*: The E2 POLICY enables the Near-RT RIC to enforce rules on E2 Nodes, which influences various network functions without continuously intervention.

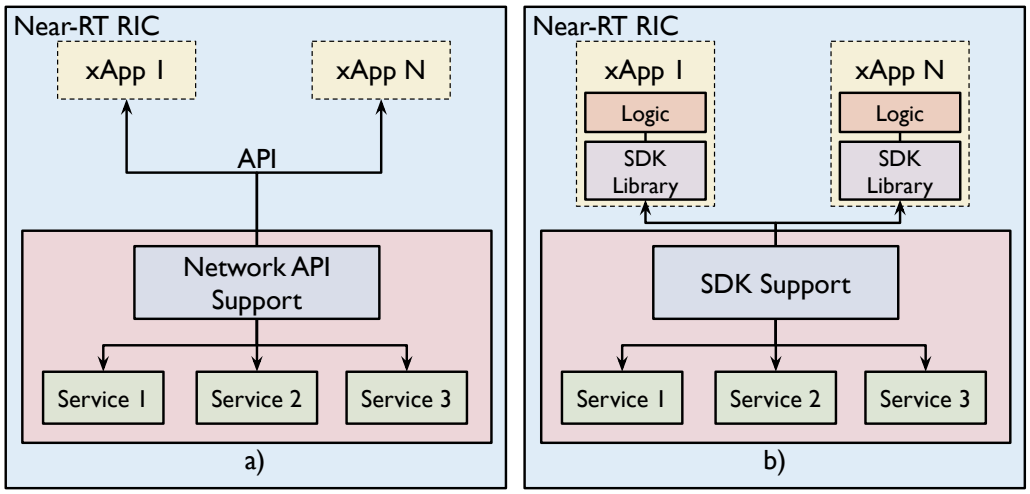

Figure 5: Near-RT RIC APIs: a) Network API Approach b) SDK Approach

• *QUERY*: This service, enables the Near-RT RIC to send a request to an E2 Node specifying the required information. The E2 Node processes the request by obtaining the necessary data. If successful, it responds with a response containing the requested details. If the retrieval fails, the E2 Node sends a failure message, indicating the reason for the failure [26].

3.4. Near-RT RIC APIs

Near-RT RIC APIs are interfaces that enable the Near-RT RIC to communicate with xApps. These APIs are categorized based on different operational needs: A1-related APIs (for accessing A1 services), E2-related APIs (for E2 interface interactions), Management APIs (for platform and xApp management), Shared Data Layer (SDL) APIs (for accessing stored network data), Enablement APIs (for service discovery and orchestration), and AI/ML Workflow APIs (for AI model integration and execution). In order to implement these APIs there are 2 different approaches: (i) the Network API approach and (ii) the SDK (software development kit) approach [28].

In the Network API approach, each Near-RT RIC component exposes a network endpoint with a defined data encoding and transport protocol to communicate with xApps directly. The protocol stack consists of an application layer protocol, a data encoding format (ASN.1), and a network transport protocol (such as SCTP, HTTPS, or gRPC). Additionally, the Network API approach establishes a secure communication by including authentication, encryption, and endpoint discovery. Each Near-RT RIC API may require a customized protocol stack based on specific trade-offs between service complexity, platform scalability, and xApp implementation overhead.

In contrast, the SDK approach abstracts low-level communication details to provide the xApp developers with a simplified programming interface for

interacting with the Near-RT RIC. Instead of directly implementing network protocols, developers utilize an SDK library that manages connection handling, authentication, encryption, and failover mechanisms. Additionally, the SDK provides debugging, building, and testing tools for xApp development. The actual protocol stack implementation within the SDK may be either based on standardized O-RAN Network APIs or left vendor-specific for a flexible integration.

Both API approaches are not mutually exclusive, and an xApp may leverage both methods depending on its requirements. For example, an xApp designed for Network API integration may either implement a direct interface or utilize an internal SDK library that abstracts network complexity. This flexibility allows developers to balance between direct control over network interactions and simplified development via SDKs.

With the mentioned components and services, the Near-RT RIC can effectively ensures efficient RAN management. However, the true intelligence of the Near-RT RIC lies in its ability to leverage xApps. The Near-RT RIC hosts xApps and by using the information provided by E2 Nodes enable xApps to optimize the RAN functions. The following section explores the role of xApps, their architecture, and how they contribute to the overall efficiency and adaptability of the O-RAN ecosystem.

4. xApps

xApps are applications deployed within the Near-RT RIC to enhance RAN real-time optimization and decision-making. xApps process the live network data and perform the executing control actions in near-real-time. xApps dynamically adjust the network parameters to perform various tasks such as: Handover Management, Resource Allocation, Spectrum Utilization, etc. To achieve this, xApps continuously monitor RAN nodes through high frequency telemetry and by utilizing AI/ML-driven techniques, they are able to detect traffic patterns and predict network behavior.

xApps interact with other Near-RT RIC components via the Near-RT RIC's internal messaging system, utilizing KPIs, mobility reports, and interference measurements, etc for dynamic network orchestration. Once an xApp determines an action, it generates the related commands that are relayed back to the RAN via the Near-RT RIC's control mechanisms. After executing a control action, an xApp may continue monitoring network conditions by collecting performance feedback to evaluate the impact of the decision. xApps are broadly categorized into two overarching fields; namely, *Reactive xApps* and *General xApps* [12].

Reactive xApps operate in an event-driven manner. They continuously monitor the network and respond to network triggers generated by telemetry

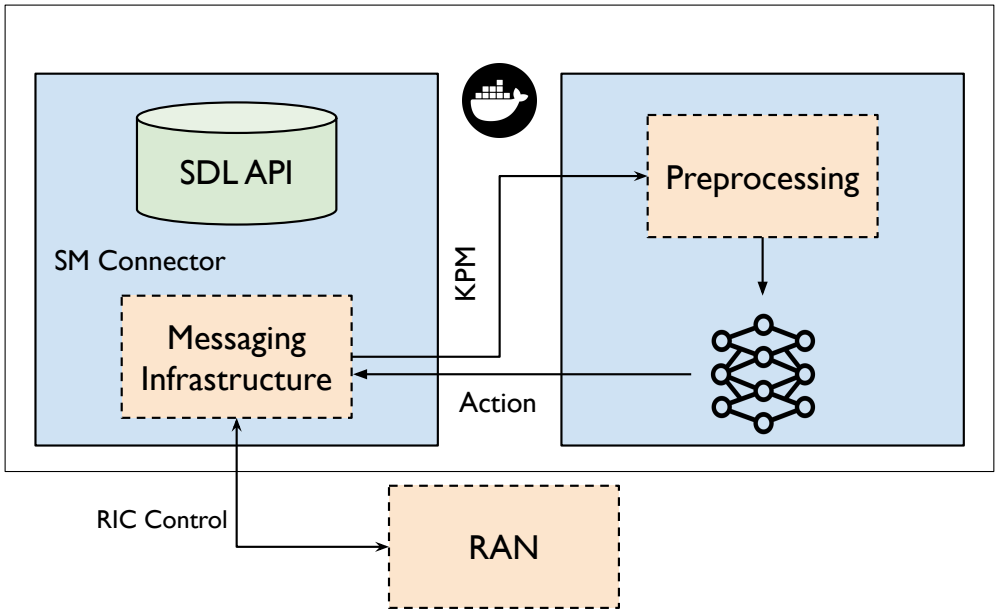

Figure 6: xApp Structure [36]

data from E2 Nodes; they subscribe to telemetry streams via the messaging infrastructure, process time-sensitive data, and execute rapid control adjustments. Reactive xApps prioritize low-latency execution where network state changes directly influence the control logic, making them suitable for high-frequency, localized optimizations within RAN subsystems.

General xApps analyze both real-time and historical data to optimize network behavior over longer time periods. Unlike reactive xApps, general xApps focus on sustained performance improvements, often integrating insights from multiple xApps to enforce multi-layer optimizations across the RAN.

4.1. xApp's Architecture

Figure 6 illustrates the architecture of an xApp. xApps are instantiated as a Docker container within the Near-RT RIC. At the core of this architecture is the service model (SM) Connector, which handles message parsing, API interactions, and data forwarding. It relays RAN KPMs to the data-driven logic unit and transmits xApp-generated control actions back to the RAN. Additionally, the SDL API allow the xApp to query the database to retrieve relevant information.

The other crucial component is the messaging infrastructure (ASN.1) used to establish a standard communication between the xApp, E2 Manager, and E2 Termination. This module supports the RIC Subscription message, the

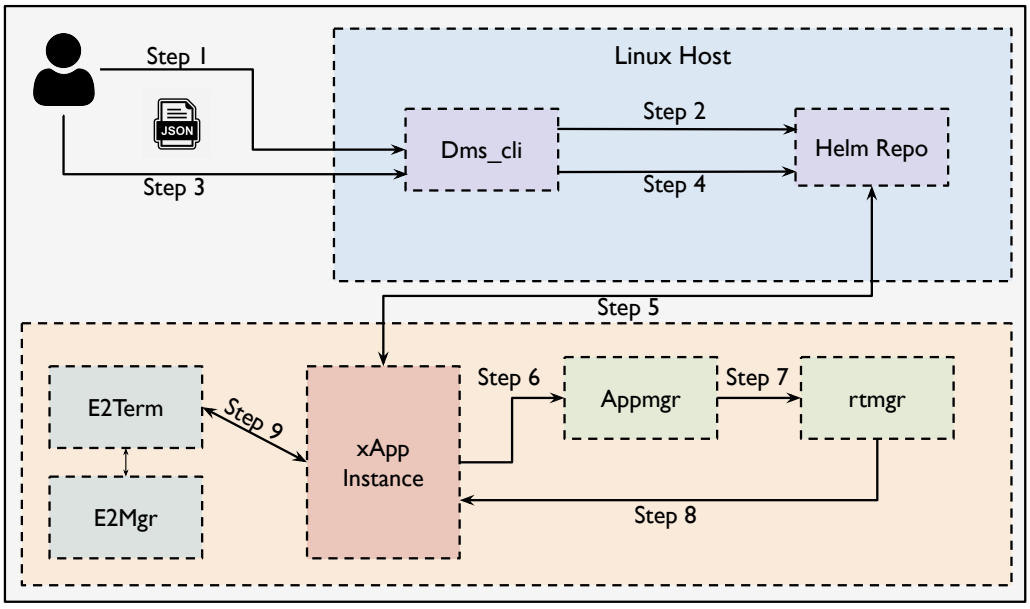

Figure 7: Deployment of xApp in O-RAN [37]

RIC Indication message (which delivers event-triggered KPM reports from base stations), and the RIC Control message (through which the xApp sends control actions to optimize network functions). In RAN, these messages are processed by the E2 Termination, which is responsible for interpreting and executing the received control actions.

The next component is the pre-processing module, which is used to process the KPMs before they are fed to the model within ann xApp. The model will process it to perform the desired task, the action will be determined and converted to a RIC Control action.

4.2. Deployment

The deployment process of xApps involves defining xApp metadata, generating deployment configurations, registering xApps with the RIC platform, and enabling communication with E2 nodes.

This section provides a detailed breakdown of the xApp the onboarding, configuration, and deployment process, and explains the step-by-step interactions between the Linux-based deployment system, the RIC Cluster, and key components such as the Route Manager (rtmgr), Application Manager (Appmgr), and E2 interface.

Deployment begins with onboarding an xApp Descriptor Schema and xApp Descriptor file. These files define xApp metadata, operational requirements and deployment parameters, including resource allocation, networking configurations, and interactions with other components in the RIC cluster

(step 1). After onboarding, the DMS_cli (Deployment Management System Command Line Interface) on a Linux host is responsible for generating Helm charts based on the xApp descriptor. Helm charts are used for Kubernetes-based orchestration for ensuring automated deployment, management, and scaling of the xApp. Once generated, these Helm charts are uploaded to the Local Helm Repository for future retrieval and deployment (step 2). Once the necessary Helm charts and configuration files are in place, the xApp deployment process is triggered, which initiates its integration into the RIC Cluster (step 3). Afterwards, the system retrieves the Helm charts from the Local Helm Repository to fetch all necessary deployment configurations and scripts required for the xApp (step 4). Then the retrieved Helm charts are downloaded and prepared for deployment in the RIC Cluster (step 5). The xApp is then properly initialized by reading the configuration parameters (step 6). The rtmgr retrieves RMR information, which defines the communication pathways within the RIC Cluster. This routing information is essential for the xApp to send and receive messages effectively between different network components and services (step 7). Once the RMR route information is retrieved, the Appmgr updates the route table with the new xApp's details. The updated route table is then distributed across the RIC Cluster to inform other services and components about presence of the new xApp (step 8). Finally, the xApp is now fully deployed and ready to interact with the E2 interface. The E2 Terminator (E2Term) and E2 Manager (E2Mgr) enable the xApp to send and receive E2 messages (step 9).

4.3. AI/ML within xApp

As mentioned before, the integration of AI/ML models into RAN ecosystem, enables the network to be optimized in near or non-real-time timescales. These models could be deployed within xApps or rApps to perform various tasks such as scheduling, mobility management, resource allocation, and interference mitigation. In the following, we will discuss the workflow, deployment scenarios and model architectures.

4.3.1. Workflow

The AI/ML workflow within O-RAN follows a structured pipeline that consists of **data collection, pre-processing, model training, inference, assisted decision making,** and **continuous optimization**. xApps interact with this workflow by subscribing to high-frequency telemetry streams from E2 Nodes. Decisions generated by xApps are then enforced through the O1 Interface for network configuration updates, E2 Interface for real-time control actions and A1 Interface for policy-driven decision-making.

Data Collection : The first and important step is Data Collection. Data might be KPIs (e.g., throughput, jitter, packet loss, Signal-to-Interference-

plus-Noise Ratio (SINR)). It is collected via the O1, A1 or E2 Interfaces from different E2 Nodes such as CU, DU, to be served as the input of the model.

Data Pre-processing : After collecting the data, it goes through the preparation phase for noise reduction, data labeling and splitting for training and inference.

Training : After preparing data, the training phase begins. The model is trained on the dataset and continues to update the parameters for the real-time inference.

Inference : After AI/ML models have been trained and validated, they are deployed for real-time inference within xApps. The AI/ML Inference step involves applying trained models to live network data, allowing xApps to make intelligent decisions and optimize RAN performance in near-real-time. Then the models are integrated into xApps and initialized for inference.

AI/ML Assisted Solution: This step translates the AI-driven insights into RAN control messages. The resulting actions are categorized into configuration updates (O1), real-time control adjustments (E2), and policy recommendations (A1). To maintain efficiency, the impact of each action is continuously monitored, and feedback loops refine AI models for future improvements. This closed-loop automation framework allows xApps to enhance network reliability, performance, and scalability.

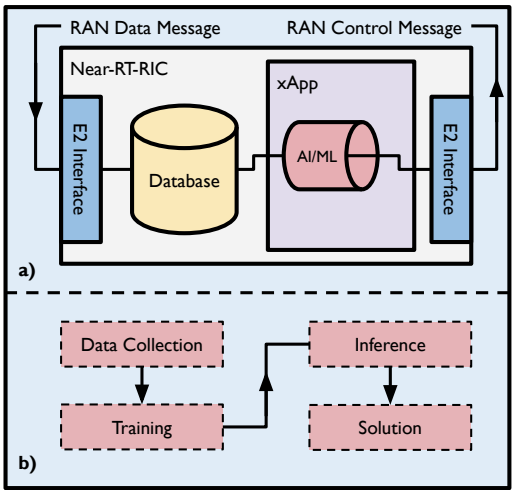

Figure 8: xApp AI/ML Workflow

Continuous Operation : This phase ensures that xApps remain adaptive and effective by continuously refining AI models based on real-time network feedback. If necessary, models undergo retraining, hyperparameter tuning, or redeployment to maintain accuracy and efficiency. Additionally, this phase incorporates automated validation against predefined KPIs and operator policies, ensuring compliance and stability.

4.3.2. Deployment Scenarios

Depending on the location of data preparation, model training, inference, and execution, different deployment scenarios can be implemented. These scenarios define how AI models are trained, managed, and deployed within the Non-RT RIC, Near-RT RIC, or external environments, and how AI-driven decisions are applied through interfaces. The scenarios are described as:

• *Scenario 1.1: Full AI/ML Processing in Non-RT RIC*

In this scenario, all AI/ML operations—including data preparation, model training, inference, model management, and continuous operation—occur in the Non-RT RIC. The trained models generate control actions that are sent to the Near-RT RIC via the A1 interface for policy enforcement or applied directly via O1 for configuration management. In this scenario, inference occurs outside the Near-RT RIC and is suitable for long-term RAN control.

• *Scenario 1.2: Hybrid Training in Non-RT RIC & Near-Real-Time Inference in Near-RT RIC*

This scenario introduces a hybrid AI/ML approach, where data preparation and model training occur in both the Non-RT RIC and Near-RT RIC, but AI/ML inference is executed in the Near-RT RIC. In this scenario model management still occurs outside Near-RT RIC.

• *Scenario 1.3: AI/ML Training Outside Non-RT RIC, Inference in Near-RT RIC*

In scenario 1.3, the Non-RT RIC does not handle model learning, therefore the AI/ML training is conducted externally. Trained models are deployed in the Near-RT RIC for inference and execution. Model management is still handled outside Non-RT RIC for ensuring model lifecycle updates.

• *Scenario 1.4: Dual AI/ML Training for Offline & Online Learning*

This deployment is a combination of offline and online learning, where SMO and the Non-RT RIC handle offline training and Near-RT RIC performs online learning and continuous adaptation.

4.3.3. Scenario 1.5: Future Feature Set (FFS) Deployment

This scenario represents a future AI/ML deployment model, where specific functionalities have yet to be standardized. It is expected to introduce advanced AI-driven RAN optimizations across multiple RIC layers.

In order to deploy AI/ML models in O-RAN, there are two different approaches: **Image** and **File-based** deployment [38]. In **Image-based** approach, the AI/ML model is encapsulated within containerized xApps. However, coupling the AI/ML models with xApps, means updates require full xApp redeployment, which can introduce additional computational overhead when upgrading individual models. The **File-based** approach decouples the AI/ML model from the xApp. This deployment approach is beneficial for architectures where the models need to be updated frequently based on different RAN conditions. Additionally, file-based deployment supports centralized model management through an ML model catalog, which enables automated deployment, rollback, and compatibility verification. However, this approach introduces dependencies on model format compatibility with the inference engine, necessitating standardization in model serialization, execution runtime environments, and optimization frameworks. The integration of

high-performance AI accelerators and optimized inference pipelines further enhances the efficiency of real-time AI-driven network optimizations.

4.3.4. Models

AI/ML models within O-RAN can be deployed using various architectural frameworks that define where model training, inference, and continuous learning take place. The choice of architecture impacts latency, scalability, AI model life cycle management, and overall system efficiency. This section explores the different AI/ML deployment architectures in O-RAN, analyzing how model training, inference, and real-time decision-making are structured to maximize performance.

The first model is **Reinforcement Learning (RL)**. RL plays a fundamental role in the O-RAN architecture [39, 40, 41, 42, 41] , which dynamically could be adapted to the evolving network conditions. RL models optimize control policies through direct interaction with the environment by utilizing feedback mechanisms in the form of rewards or penalties to maximize long-term network performance and operate based on a Markov Decision Process (MDP), which is defined by: **State (S)**, **Action (A)**, **Reward (R)**, and **State Transition Function (T)**. The state (S) represents the current network conditions which includes different KPIs such as signal strength, interference levels, traffic load, user mobility, and QoS metrics. The action (A) defines the set of control decisions that the RL agent (e.g., xApp/rApp) can take, such as adjusting handover thresholds, modifying scheduling priorities, reallocating spectrum resources, or dynamically tuning beamforming parameters. The reward function (R) quantifies the effectiveness of each action and provides feedback to improve network's performance. The state transition function (T) models the evolution of the system after an action is executed, capturing the stochastic nature of wireless environments, such as UE mobility, fluctuating interference, and traffic demand variations.

The RL-based xApp first collects real-time network state information from the E2 Nodes through the E2 interface using the E2SM-KPM service model. This data represents the current conditions of the network, forming the (S) that the RL agent will use to determine optimal actions. Once the xApp has observed the network state, it evaluates a set of possible control actions and selects the most suitable one to optimize RAN performance. The selected control actions are executed via the E2SM-RC service model, which applies these decisions to the underlying E2 Nodes.

Following action execution, the RL-based xApp evaluates its impact by computing a reward (R). If the action leads to better network conditions, the reward is positive; otherwise, a penalty is applied. This feedback mechanism allows the RL agent to iteratively refine its policy, ensuring that it progressively learns optimal strategies for different network scenarios. For

instance, as shown in Figure 10, the state is fed into the DQN, which is related to the UEs. Then the agent generates a random action and observe the corresponded reward. In this work, th goal is to maximize the number of underutilized radio cards (RC)s that are switched off while maintaining QoS.

As the RL agent interacts with the network over time, it continuously updates its policy, refining the mapping between states and actions to maximize long-term performance. Various RL techniques, such as Deep Q-Networks (DQN) [43, 44], Proximal Policy Optimization (PPO) [45, 46], and Multi-Agent RL (MARL) [47, 48], are employed to ensure efficient learning and adaptation. The use of neural network-based function approximation enables RL-based xApps to generalize across diverse network conditions, making them more resilient to unexpected changes in the wireless environment.

Beyond real-time decision-making, RL policies can be periodically updated through the A1 interface, which connects the Non-RT RIC and Near-RT RIC. The Non-RT RIC trains RL models offline using large-scale historical data to optimize strategies in a simulated environment before deploying them to xApps for real-time execution. This hybrid AI approach ensures that RL xApps benefit from both real-time learning and strategic offline training.

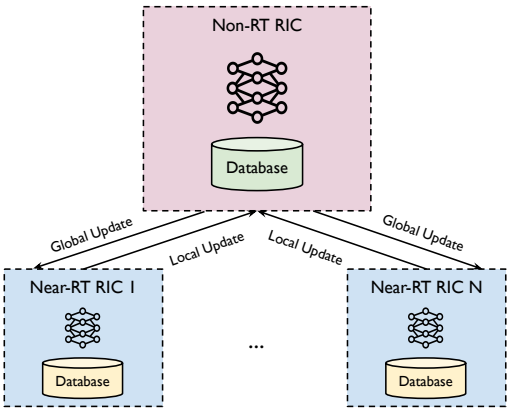

Figure 9: Example of FL Workflow within O-RAN [49] (sending the updates from xApps to rApps)

The next model is **Federated Learning (FL)** [50, 51, 52], which is an advanced distributed machine learning model that enables training across multiple network nodes without transferring raw data to a central location. FL allows individual RAN components such as xApps, rApps, and E2 Nodes to collaboratively learn from their local data and after gaining the weights or gradients, transmit it to the central node for orchestrating O-RAN. Therefore, excessive data transmission would be avoided.

The FL workflow in O-RAN follows a structured process. First, each FL-enabled xApp or rApp trains an AI/ML model using its locally available RAN performance metrics, to eliminate the need for raw data transmission to a central server. After the training phase , the FL framework aggregates only the locally computed model parameters rather than actual datasets, which significantly reducing the risk of privacy breaches. The Non-RT RIC (or SMO) then performs a global aggregation process in order to refine the AI model while ensuring that no

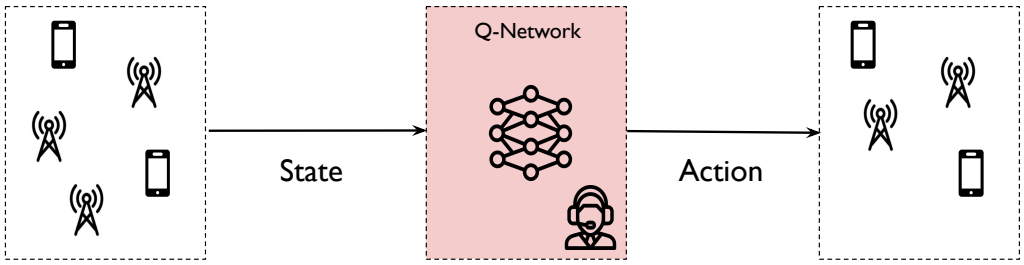

Figure 10: Example of RL Workflow within O-RAN [58]

single dataset dominates the learning process. After the aggregation phase, the global model is redistributed to all participating RAN nodes.

Federated Learning in O-RAN is particularly beneficial in multi-operator deployments, where data-sharing constraints and vendor-specific network management policies make centralized AI training impractical. By enabling each operator or vendor to independently contribute to AI training while preserving data sovereignty, FL fosters collaborative intelligence across the entire O-RAN ecosystem. Moreover, FL supports a hierarchical AI model training framework, where rApps in the Non-RT RIC perform global model training on historical datasets, while xApps in the Near-RT RIC fine-tune models in real-time for near real-time decision-making to make the network proactive for various conditions.

From a security perspective [50], FL significantly reduces the risk of data breaches and cyber attacks by localizing the data. Additionally, since FL minimizes the need for high-bandwidth data transfers, it significantly reduces network congestion and computation overhead, which is ideal for AI/ML approaches within resource-constrained RAN environments.

While FL effectively enables distributed model training for tasks such as anomaly detection and traffic forecasting, many real-time network control and optimization problems require sequential decision-making rather than static predictions. Extension of FL to federated reinforcement learning (FRL) addresses this by enabling different xApps to collaboratively refine the network's performance in a decentralized manner [53, 54, 55, 56, 57]. Unlike FL, which relies on static, batch-based model training using pre-collected data, FRL enables continuous policy optimization, where the Near-RT RIC locally trains RL-based xApp(s) through real-time interactions with the RAN environment.

4.3.5. AI/ML Security

Security of AI/ML models is one of the key points in implementing them. These models could be vulnerable to various attacks such as adversarial attacks, model poisoning and data privacy breaches which results in performance degradation. In this section we provide detail about the various

attacks and their affect in the O-RAN ecosystem.

These attacks could be defined as *Input Manipulation Attack, Data Poisoning Attack, Model Inversion Attack, Model Stealing, AI Supply Chain Attacks, Transfer Learning, Model Skewing* [59].

• *Input Manipulation Attack*: In this attack, the adversary has access to the training dataset which is collected by the E2 nodes and can manipulate the labels in order to fool the model.

• *Data Poisoning*: The adversary can poison xApps and E2 nodes dataset to affect the performance of the model which could be categorized into black box, gray box and white box attack.

• *Model Inversion*: One significant threat to AI/ML models is inversion attacks, where an adversary leverages access to a model's input-output pairs to approximate its original training data. Without knowledge of the model's architecture and dataset, the attacker systematically generates inputs, analyzes the corresponding predictions, and refines their approach iteratively until they reconstruct sensitive data. In an O-RAN environment, this could lead to the exposure of user mobility patterns, Quality of Service (QoS) metrics, or network performance statistics, which can be exploited for adversarial manipulation or privacy breaches. To mitigate model inversion, adversarial defense mechanisms, such as query limitation, response perturbation, and model watermarking, can help detect and counter unauthorized model probing.

• *Model Stealing*: Stolen models enable adversaries to replicate proprietary functionalities without incurring the significant costs of model development and training. If a competitor gains unauthorized access to a trained AI model, they can leverage it for similar or competing purposes to undermine the original developer's competitive advantage. In an O-RAN environment, stolen models could lead to unauthorized deployment of proprietary xApps, which weakens the innovation incentives and exposes networks to further security risks.

To mitigate model theft, encrypted model storage and access control mechanisms should be integrated into AI pipelines to ensure model ownership verification and restrict unauthorized access.

• *AI Supply Chain Attacks*: Beyond software and data vulnerabilities, hardware-based attacks pose a significant risk to xApps and Near-RT RIC intelligence in O-RAN. Malicious actors may target AI processing hardware, such as GPUs, ASICs, and FPGAs by utilizing vulnerabilities at the hardware level. Such manipulations allows the attackers to degrade the model's performance. Even if the AI software and training data remain secure, hardware-based exploits can create hidden vulnerabilities that are difficult to detect and mitigate.

Additionally, AI services in O-RAN are often exposed through APIs for

model deployment, inference and maintenance. If these APIs are not properly secured, attackers can exploit them to launch supply chain attacks or inject malicious code.

To mitigate these risks, hardware-based security solutions such as trusted execution environments (TEEs) could be implemented. Additionally, strong API security practices, which includes authentication and encrypted communications are essential to protect AI services from external threats.

• *Model Skewing*: AI/ML models in O-RAN continuously adapt based on feedback mechanisms to improve performance. However, adversaries can exploit these feedback loops through model skewing attacks, where manipulated feedback causing the model to learn incorrect patterns. This can significantly disrupt network operations and degrade service quality. Attackers can tamper with feedback data by introducing biases that gradually corrupt model predictions and influence the behavior of the entire ML system.

These attacks are particularly dangerous due to vulnerabilities in weak feedback mechanisms, inadequate adversarial training and insufficient monitoring of ML performance for detecting anomalies or manipulations.

To mitigate feedback manipulation risks, O-RAN AI models should incorporate adversarial training and real-time anomaly detection to ensure feedback integrity.

5. Applications of xApps in O-RAN

This section delves into practical implementations and real-world scenarios where xApps facilitate the smooth functioning and advancement of O-RAN networks.

5.1. Traffic Steering

Ensuring the optimal user experience in today's wireless networks is crucial when performance is impacted by local load imbalance or bandwidth fluctuations. To mitigate these challenges, traffic management policies dynamically allocate users to cells by priority in the control and user planes. The Non-RT RIC constantly monitors user equipment (UE)-level performance and cell resource utilization, comparing real-time conditions to expected service-level requirements. When there are performance deviations, it detects congested zones and, if necessary, redirects users to less congested cells to enhance radio resources, bandwidth availability, and overall network performance. In multi-access networks, traffic steering is essential to smartly distribute traffic loads over different access technologies based on radio conditions and application demands. The Non-RT RIC defines the policies for the prioritization of the UE connection and the best cell selections for the control and user planes. The policies are propagated via the A1 interface to

the Near-RT RIC, where radio resources are adjusted in real time. With the integration of Artificial Intelligence (AI)-aided traffic management, O-RAN enhances network flexibility, which enables dynamic policy enforcement that ensures seamless performance in complex and rapidly changing network environments.

In [60], the authors proposed an xApp for TS and interference mitigation in multi-cell, multi-frequency 5G O-RAN networks. The approach utilizes unsupervised learning-based anomaly detection and dynamically steers user connections to minimize interference. System-level simulations show an improvement in throughput of 8. 9% and an improvement in detection accuracy and latency. However, xApp relies heavily on accurate interference classification and may be limited by real-time adaptability in high-mobility scenarios. In [61], the paper describes multi-vendor interoperability for facilitating energy-aware traffic steering by an xApp for dynamic user offloading. The proposed TS-xApp interfaces with an Energy Saving rApp (ES-rApp), which aids in switching on / off the idle cells by dynamic load levels of the network. The test findings validate effective energy savings and load leveling, yet operation is dependent on coordinated xApp/rApp functionality without interference, and conflicting control actions may lead to poor performance. Habib et al. [62] introduced a hierarchical Deep Q-Network (DQN) xApp to improve TS decisions in the O-RAN framework. In contrast to conventional single-layer machine learning models, h-DQN splits TS into a two-layer decision-making process, where a meta-controller determines high-level policies and a local controller implements real-time steering actions. The case study verifies significant performance gains compared to baseline algorithms, but demands a considerable amount of training data and may experience slow adaptation to rapidly varying network states.

This paper [9] presented an AI-driven TS xApp using reinforcement learning (RL) for per-user handover optimization The xApp, trained with 40 million data points, utilizes Conservative Q-learning (CQL) and convolutional neural network (CNN)-based models to maximize the choice of the base station (BS) on a per-user basis, improving throughput and spectral efficiency by 50%. Training is computationally expensive, however, and it remains uncertain whether the model works in ultra-dense environments. In [63], the article proposed a modular TS xApp for future 6G networks centered on adaptive traffic management through AI-driven decision-making. xApp adaptively redistributes traffic based on network load and QoS requirements in real time, improving latency and congestion control. However, the research remains largely theoretical, with minimal real-world testing and performance validation. This article [64] introduced a TS and load balancing xApp, which groups the UEs using K-means and predicts cell throughput using long short-term memory (LSTM) models. The xApp identifies poor QoS

users and dynamically requests handover to enhance load balancing, as well as network efficiency. Performance shows improved throughput distribution, but the process relies on data updates repeatedly, and K-means clustering would not work in highly dynamic UE movement.

5.2. Quality of Service (QoS) Optimization

xApps are essential for AI/ML-based decision making to improve RAN resource optimization based on QoS. This scenario combines O-CU, O-DU, Non-RT RIC, and Near-RT RIC, where xApps in Near-RT RIC facilitate real-time resource management based on QoS. Policies are created by Non-RT RIC that inform Near-RT RIC, while xApps in the latter interpret the policies and enforce them by dynamically tweaking RAN parameters for optimal performance. Further, the Non-RT RIC is always watching QoS-driven measurements derived from both network and SMO procedures, thereby giving beneficial information upon which xApps rely for predictive and adaptive optimization. The O-CU and O-DU collect extensive UE performance measurements, which are transferred to the SMO through O1, and enable xApps to optimize scheduling and congestion management strategies. Moreover, facilitation of external data sources can offer a great boost to optimize RAN resources, thereby complementing the insights generated by the network.

In [65], this paper addresses QoS conflicts between xApps in Near-RT RIC, and introduces a QoS-Aware Conflict Mitigation (QACM) solution. The solution optimally resolves conflicting control decisions while meeting multiple xApps' QoS requirements simultaneously. The solution stabilizes the network, but with high computational overhead and low scalability, which are issues for large-scale deployment. In [66], the authors proposed a machine learning (ML)-driven QoS xApp profiling in Near-RT RIC to optimize RAN and the core network resources. The xApp periodically monitors real-time network data and assigns QoS profiles to users as estimated by an ML model. The result is efficient QoS-aware resource allocation, leading to better jitter management and bit rate constancy. However, the approach depends on precise user profiling and real-time data processing, which might be problematic in large-scale networks. Samorzewski et al. [67] presented a QoS-aware Resource Management (QRA) xApp for physical resource block (PRB) allocation in the SD-RAN platform. The xApp dynamically allocates PRBs to users based on real-time QoS requirements while ensuring compliance with Service Level Agreements (SLAs). Simulations confirm improved resource allocation efficiency, yet the system lacks full integration of the A1 policy, and control messages are not sent over the E2 interface, which removes real-world applicability. In the article [68], an AI-driven xApp for 5G and beyond intra-slice QoS-aware resource management is presented. Using DQN, the xApp dynamically adapts resources between eMBB and URLLC

slices and increases throughput by 11.5% and reduces latency by up to 45.5%. Mungari et al. [69] introduced OREO, an xApp orchestrator that optimizes xApp deployment on RAN services with minimum use of computational resources. OREO enables xApp sharing with a reduction of up to 66. 7% computational overhead while maintaining QoS. The approach is NP-hard and, therefore, requires heuristic-based approximations, which may not be scalable in highly dynamic environments O-RAN.

5.3. Quality of Experience (QoE)

Several studies in the literature have explored ML-driven approaches to enhance Quality of Experience (QoE) in O-RAN. Anand et al. [70] introduced MLCIMO, a ML-enabled interference mitigation and user offloading technique to improveQoS and QoE in 5G heterogeneous networks (HetNets) with O-RAN. The scheme employs a multi-binary classification model to categorize users based on their interference levels and then reassigns them to the most appropriate base station (MBS or FBS) for improved resource utilization. The simulation results reveal that MLCIMO improves throughput while reducing delay and packet loss, leading to high QoE scores in video streaming scenarios. However, its performance is highly dependent on accurate interference classification, and there is a limitation with regard to real-world testing. In [71], the authors proposed an ML-driven xApp with a focus on interference classification and user offloading in multi-tier 5G HetNets for enhancing QoE of different services. The xApp incorporates a multi-binary classifier embedded in Near-RT RIC for load balancing and dynamic interference management. Experiments demonstrate that xApp improves QoE metrics such as VMAF, R-Factor, and RUMSI over state-of-the-art solutions but requires continuously retraining the model. In another study [72], the authors explored the role of QoE-aware xApps in next-generation wireless networks based on O-RAN. It highlights the need to incorporate AI-based QoE optimization models into xApps to improve the performance of the network for latency-sensitive applications. The proposed framework makes use of deep learning (DL)-based QoE estimation to adjust network parameters dynamically in real time. Although this method provides significant improvements in user experience, the study is conceptual and lacks implementation details.

In [73], the article presents QoE2F, an xApp for adaptive resource allocation for high-resolution video streaming in O-RAN HetNets. The xApp applies a genetic algorithm-based optimization method to the allocation of power and spectrum resources that is based on a prediction of QoE in real time. The simulation results show that QoE2F performs better than the baseline models, with VMAF, MoS scores, and load balancing, but the use of sophisticated optimization methods imposes high computational overhead

that poses challenges for real-time deployment. In [74], the paper focused on the optimization driven by QoE for VR services in O-RAN with the focus being on ultra-low latency and high bandwidth requirements for VR streaming. The paper proposes an AI driven xApp that predicted network congestion and tailor-made resource allocation to prevent VR experience degradation. Although the result affirms that the optimization driven by AI QoE greatly improves the performance of the VR streaming, the paper does not analyze the complete scope of the influence of the cutting of the network on VR QoE.

5.4. Anomaly Detection

Anomaly detection systems are essential to protect beyond 5G networks against emerging cyber threats, ensuring that these networks meet their performance and reliability goals. Among these threats, identifying Distributed Denial of Service (DDoS) attacks is particularly critical. DDoS attacks can quickly overwhelm the network infrastructure or deplete the computational and memory resources of a server, causing significant service interruptions and potential data loss. The increasing sophistication of DDoS strategies poses challenges to traditional rule-based detection methods, prompting the adoption of ML and DL techniques. These advanced approaches excel in identifying subtle changes in network behavior caused by malicious activities, thus improving detection accuracy and generalization [75] in network anomaly detection.

Benazaid et al. [75] introduce TenaxDoS, an innovative framework that integrates Federated Learning (FL) with a replay memory-based continual learning (CL) strategy for the detection of sustainable network anomalies in O-RAN. TenaxDoS enables privacy-preserving and cooperative anomaly detection while ensuring 98.8% knowledge retention in dynamic environments beyond 5G. However, the approach relies on centralized FL aggregation, which limits scalability, and focuses only on DDoS attacks, leaving other threats unaddressed. It also requires high memory, which can be challenging for resource-limited devices. Another study [76] proposes a peer-to-peer (P2P) FL-based anomaly detection within the O-RAN architecture, eliminating single points of failure and enhancing privacy. The authors evaluated four P2P FL variants using the UNSW-NB15 data set. The normal P2P FL model achieved a precision of 90.8%, comparable to the centralized FL. However, it incurs higher communication costs and lacks the evaluation of various threats to the network. Basaran et al. [77] formulate a deep autoencoder-based RF anomaly detection system for 5G O-RAN Near-RT RIC xapps to improve handover efficiency and network performance. The deep autoencoder outperforms the Isolation Forest algorithm by 10% in accuracy which demonstrates its effectiveness in detecting RF anomalies. As the approach depends on pre-trained models, it limits the adaptability.

5.5. Security

xApps improve network performance while ensuring secure communication across multiple interfaces. Using real-time analytics, these applications monitor network activity to detect security threats and anomalies. When unusual traffic patterns or suspicious data are detected, xApps can trigger alerts and implement preventive measures to address potential risks. They help maintain strong security by implementing access controls, encryption protocols, and authentication [78].

The study [79] explores security vulnerabilities in the xApp access control and the E2 interface within O-RAN. The authors simulate three attack scenarios, demonstrating how malicious xApps can exploit weak access controls to shut down E2 nodes and disrupt RAN services. revealing significant risks due to the lack of defined access control permissions for xApps. However, the study focuses mainly on API security, leaving ML-based threats and adversarial attacks unaddressed.

Atalay et al. [80] introduces the xApp Repository Function (XRF), an authentication and authorization framework to secure xApp interactions in O-RAN. XRF ensures secure xApp doscovery and access control with minimal overhead. Although it secures xApp authentication, it does not address other emerging threats, such as adversarial ML attacks or API-level vulnerabilities. Meanwhile, [81] provides a comprehensive security assessment of O-RAN, focusing on open interfaces, closed-loop control driven by AI and cloud-based platforms. The study evaluates the encryption overhead on the E2 interface and identifies three key security principles for the cloud O-RAN. In another study, [82] highlights how ML-based xApps in O-RAN can be compromised by adversarial attacks. The authors deploy a ML-based interference classifier xApp in a real-world O-RAN testbed and demonstrate how small manipulation in input data can significantly degrade the accuracy of classification. The results show that even minor adversarial attacks can disrupt network performance, leading to reduced capacity and data loss. However, the study does not propose a defense mechanism, leaving ML-based xApps exposed to adversarial threats. In addition, it focuses only on interference classification, limiting insights into broader xApp security challenges.

Chiejina et al. [83] investigate adversarial attacks on xApps based on ML in Near-RT RIC and evaluate a distillation-based defense to improve security. The authors deploy a malicious xApp that alters ML test data, causing significant classification errors and network degradation. Experimental results show that adversarial attacks can reduce ML accuracy to 0% in some cases, but the distillation-based defense restores accuracy to 98.3%, significantly improving resilience. However, the approach relies on centralized training, limiting scalability in distributed O-RAN setups. Although these studies address key security challenges in O-RAN, they also reveal common limi-

tations. Many introduce computational overhead, making them difficult to implement in real-time. Authentication frameworks such as XRF improve security but depend on centralized control, which limits scalability. Moreover, most solutions focus on specific attack types, leaving broader security risks such as zero-day exploits, cross-xApp vulnerabilities, and API manipulation unaddressed.

5.6. Resource Allocation

xApps can optimize spectrum allocation by identifying idle spectrum and analyzing real-time RAN data to uncover usage trends. Morfa et al. [84] proposes deep reinforcement learning (DRL)-based xApps for joint RAN and Multi-Access Edge Computing (MEC) resource allocation. Using DQN, xApps jointly optimize PRB allocation and computing resources, balancing spectral and edge computing capacity. Simulation results confirm that DRL-based xApps improve QoS and achieve efficient spectrum utilization in 5G and 6G networks. However, training complexity and inference latency pose challenges for real-time decision making in large-scale O-RAN deployments. In [85], the authors present an AI-based xApp for dynamic spectrum allocation, optimizing the assignment of PRB in O-RAN. The xApp employs a Random Forest Classifier, which dynamically allocates resources according to real-time traffic and QoS requirements. The model attains 85% accuracy in choosing optimal resource policies, considerably enhancing scheduling efficiency. Nevertheless, its dependence on historical training data renders it less responsive to sudden traffic fluctuations. The study [86] proposes a conflict-free resource allocation team learning algorithm for xApps in O-RAN. xApps for power and spectrum allocation collaborate through a shared DQN-based decision-making mechanism, preventing resource conflicts in multi-vendor O-RAN systems. Simulations show 8.8% higher throughput and 64.8% lower packet drop rate in high-mobility environments (20 m/s). Although the approach stabilizes the network, it involves additional signaling between xApps, which causes overhead in large-scale networks. In another study [87], the authors introduce an xApp-based AI/ML-powered resource allocation framework in a software defined radio (SDR)-based O-RAN testbed. xApps running on the Near-RT RIC monitor traffic demand in real time and dynamically assign radio resources to improve spectral efficiency. Tests in an open source Software Radio Systems (SRS)RAN deployment confirm that AI-enhanced xApps reduce latency and boost spectrum utilization. However, inference in real time AI in xApps incurs computational overhead, which slows decision making in time-critical applications.

In [68], the authors outline O-RAN-Sense, a novel xApp-based crowdsensing framework for noncooperative transmitter localization and detection in O-RAN. xApps deployed in the Near-RT RIC run Time Difference of

Arrival (TDOA) intelligence from low-cost IoT spectrum sensors to locally estimate the location of transmitters. City-scale experiments show great localization accuracy at tens of meters, which is misleading in the use of bounding spectrum. Synchronization error and sensor installation limitations can affect performance in extensive deployments. In [88], the article explores how xApps can enhance O-RAN spectrum sharing by replacing traditional external spectrum sensors with AI powered spectrum sensing. The proposed xApps leverage real-time RAN data to predict spectrum availability and dynamically allocate it using the Near-RT RIC. Using closed-loop control mechanisms within the xApps, interference is reduced and spectrum utilization is optimized without additional hardware requirements. The findings demonstrate reduced deployment expense and enhanced spectral efficiency; however, the dependence on precise machine learning training creates difficulties in conforming to real-world radio frequency fluctuations.

6. Open Research Challenges

While some progress has been made up to this point, several important research challenges currently remain unaddressed on the xApp frontier. If xApps and the O-RAN system model as a whole are to have the revolutionary impact they promise, these crucial research challenges must be successfully tackled and overcome. Thus, in this section we highlight the primary issues and outline avenues for potential solutions.

6.1. Robust AI/ML

As we have discussed in this work already, AI/ML is central to the O-RAN paradigm and xApps represent the locus point where AI/ML interacts with the physical network. As such, the ubiquitous need for robust AI/ML solutions that are reliable in complex environments is essential in the xApp domain. In fact, the cellular landscape presents a host of unique and multi-faceted challenges that make the development of dependable AI/ML techniques very challenging. Particularly, the nature of wireless data itself introduces a high level of complexity into the model development.

• *Data Volume*: Firstly, the sheer *volume* of data in the wireless domain can become unmanageable very quickly. For instance, capturing IQ data at a sampling rate of 100Mbps for 10 seconds will result in a file size around 4GB. Thus, for AI/ML models housed inside xApps to be effective they must handle vast swaths of heterogeneous data quickly without becoming overwhelmed.

• *Data Integrity*: The *integrity* or *legitimacy* of the wireless data is also of utmost importance when considering AI/ML techniques for xApps. This has to do with whether or not the data being fed to AI/ML solutions has been

compromised in some way so as to negatively affect the model performance. The question of data integrity is relevant in the AI/ML training, testing, deployment, and fine-tuning phases since the data that is driving the AI/ML in each of these phases will affect the model behavior and could result in network performance degradation if the data is compromised. The main challenge here is determining whether data is legitimate or not. This aspect of the robust AI/ML research challenges is related to the xApp vulnerabilities subsection which we discuss in greater detail in the respective subsection below.

• *Data Relevance and Training*: The *relevance* of the data is also an important question to consider for AI/ML development, especially in the training phase. Creating a realistic dataset with which to train a particular AI/ML application is a critical component of model robustness. This is especially challenging in the wireless domain where every wireless scenario is unique and highly dynamic. AI/ML solutions will need to be portable from one cellular environment to another without needing to be re-trained from scratch. This will only be possible if they are trained using carefully crafted datasets that capture the complexities of the wireless domain. Unfortunately, commercial network operators do not generally make their cellular datasets publicly available which means that realistic wireless testbeds [89] must be utilized to create the necessary datasets.

• *Testing and Benchmarks*: Before AI/ML applications can be placed inside third-party xApps, sold in an xApp marketplace, and utilized for commercial-grade ran surveillance and control etc., they must first be thoroughly tested to validate their performance. This is not a trivial task given the nature of the wireless landscape. For one, commercial network operators do not allow their public networks to be used for testing purposes (for obvious reasons) which makes it challenging for developers to experimentally validate their models in real-world cellular scenarios. Not only this, but there is also a lack of standardized benchmarks or benchmarking platforms where the performance of xApp-related AI/ML techniques can be proven in relation to existing solutions. This creates a serious challenge for developers who are seeking to create a robust product that can withstand the complexities of the wireless landscape. In order to overcome the testing roadblock, realistic cellular testbeds like [89] must be developed and made publicly available for researcher's in both industry and academia to utilize; and a standardized benchmarking system must be developed as well to bring cohesion to the xApp AI/ML testing world. Only then will realistic AI/ML solutions be successfully incorporated into the O-RAN framework.

• *Fine-Tuning and Dynamic Environments*: The highly dynamic nature of the wireless domain will necessitate fine-tuning of AI/ML models in many cases. In these instances an AI/ML model will be tweaked as the environment

changes overtime to maximize the application's effectiveness. This is an essential process in the creation of robust xApp AI/ML solutions that remain effective as wireless conditions change dynamically overtime. The question remains as to the frequency of the fine-tuning process. How often should models be updated and how can it be known whether or not a model has become stale given its current wireless environment? These are important research questions that must be addressed.

6.2. xApp Vulnerabilities

A major research issue that is almost entirely unaddressed in the literature is the question of xApp vulnerability. This is a vital concept that must be taken into account when considering the broader O-RAN paradigm; this is particularly true since the O-RAN system model is based upon the idea of openness and disaggregation which greatly expands the vulnerability surface in comparison to the black-box cellular networks of previous generations. While open interfaces, virtualization, and interoperable third-party solutions have the potential to support unprecedented ran flexibility and customization, it is also true that they present a serious security hazard since hackers will have greater access to the network. Given this fact, there are a few vitally important vulnerabilities to consider when dealing with xApp development and deployment for FutureG RANs.

• *Third-Party xApp Certification*: Given that xApps will be created by third-party developers before being distributed on some form of xApp marketplace, the question arises regarding how to tell if a given xApp is legitimate or not. By what means will network operators know if a given xApp was developed by a hacker in order to steal user information or degrade network performance using a suboptimal policy, etc? There is currently no public xApp certification process by which network operators can determine such things, which means that hackers have free reign to develop malicious xApps that compromise the network. Thus, it will be absolutely essential that some form of xApp certification process is instantiated, but the question remains as to what form this will take.

• *Interface Vulnerabilities*: The open interfaces through which the xApps communicate with the Near-RT-RIC and in turn through which the Near-RT-RIC interacts with the ran (namely, the RIC internal messaging system and the E2 Interface, respectively) pose a serious security challenge as well. Adversaries can use these interfaces to intercept ran-related messages and either sniff private user data or introduce small perturbations into the ran control/data messages that cause the Near-RT-RIC to make suboptimal control decisions (for instance, slicing the network inefficiently or implementing a sub-ideal scheduling algorithm). This means that detection and prevention techniques must be developed to protect against such security breaches.

However, this topic is largely unexplored and much work still remains to be done in this field.

• *Adversarial ML*: Because xApps are publicly available, hackers can purchase the same xApps that network operators utilize in their commercial implementations. This means that hackers also have access to the AI/ML models located inside the xApps which opens up the realistic possibility of them performing white-box attacks on the commercial AI/ML models (for instance, using perturbations injected via the open interfaces). This is obviously related to the *data integrity* problem we mentioned in the subsection on robust AI/ML, but we have included it here for completeness since this also represents a serious xApp vulnerability. Up to this point, very little work has been done to analyze the effects of such attacks on the network performance; however, these types of studies are a necessary prerequisite for increased xApp security.

6.3. Multi-xApp Policy Conflicts

Another important research question to consider is the possibility of multi-xApp policy conflicts [90][91]. These occur when two different xApps inside the same Near-RT-RIC generate contradictory control actions in accordance with their own unique policies (for instance, if the control message from *xApp A* indicates that 25 PRBs should be allocated to network *slice 1*, while the control message from *xApp B* simultaneously indicates that only 10 PRBs should be allocated to network *slice 1*; or perhaps *xApp C* and *xApp D* dictate that different UE scheduling algorithms should be used for network *slice 2*). These types of control conflicts could create a variety of issues and unfortunately, this topic has not been well studied up to this point; Giannopoulos et. al [90] represents one of the only (if not *the* only) conflict resolution technique for such scenarios. The primary question is what should be done in these multi-xApp conflict instances given that the Near-RT-RIC cannot implement contradictory control policies in the ran. There is obviously a need for some form of orchestration or xApp hierarchy, but how this should be implemented in situational cellular scenarios is not a trivial problem.

6.4. Near-RT Constraints

Another important xApp research challenge is related to the near requirements imposed on these applications given their placement inside the Near-RT-RIC. As previously discussed, the Near-RT-RIC operates on a timescale between 10 milliseconds and 1 second which means that the xApps housed within must operate with very low latency - especially those xApps that are responsible for sending control messages to the Near-RT-RIC which are then forwarded to the ran for dynamic network control. Thus, it will be of utmost importance that xApps function certifiably within the imposed near

constraints, which requires that all internal xApp operations function with strict timing guarantees.

6.5. Scalability

As the network size expands (e.g., as more devices are registered in the core network and/or the number of slices corresponding to different traffic types is increased), the complexity of the network increases greatly. Thus, the scalability of xApp solutions that deal with UE-specific Key Performance Metrics (KPMs) or slice-specific data becomes and important discussion. In order for xApps to function in commercial-grade network deployments, they must remain effective in scenarios where the network size changes dynamically and (potentially) increases over time. This not only means that the interfaces connecting xApps with the Near-RT-RIC must be capable of supporting higher data rates (e.g., if additional UE KPMs need to be forwarded from the Near-RT-RIC to the relevant xApps, but also that the AI/ML applications housed inside the xApps themselves must be flexible if the input size changes. Ideally, xApps should be agnostic to network size as this will ensure they remain effective with dynamic network changes. This however, is not a trivial task and will require xApp developers to craft and (even more challenging) *test* their products so as to ensure that they function properly as the network scales up.

6.6. xApp Validation and Reliability

Although we have mentioned this in passing in the proceeding subsections, it is worth highlighting that an overarching xApp development research challenge exists in the testing phase. Ideally, xApp developers would be running thorough experimental evaluations to verify the reliability of their products prior to commercialization. This however, is unrealistic in many cases since the commercial infrastructure is not publicly available for third-party research and development teams to test and hone their xApp products. Even the most realistic large-scale testbeds and development frameworks do not necessarily capture all the intricacies of real-world commercial-grade networks and (additionally) they may not be publicly available for experimentation in every case. This creates a genuine crisis when it comes to validating the reliability of xApps that are placed on the market for commercial use. Interestingly, despite the fact that commercial network operators will almost certainly not allow third parties to use their networks for testing or release their private network datasets for such purposes, the question of third-party xApp reliability and validation should nonetheless be very much of interest to commercial-grade network operators. Inefficient xApps will degrade network performance and could result in suboptimal user experience. So this is not just a problem for xApp developers, it is also an issue the network operators.

7. Conclusions

It's clear that next generation cellular networks will require a high degree of flexibility and will embed an unprecedented amount of intelligence into the network to bolster performance and support a wide range of demanding user applications. This shift in the cellular landscape will require an open and customizable ran architecture that can support the growing performance demands. In this survey we have highlighted the transformative potential of the O-RAN paradigm, particularly honing in on the locus of FutureG intelligence and flexibility - namely, xApps. We have shown how these vital applications fit into the broader O-RAN framework and demonstrated their centrality to the integration of AI/ML into the network. We have examined state-of-the-art xApp uses cases and outlined the critical research challenges that currently remain unaddressed in the field. All in all, we have provided a thorough *xApps* survey that will give any reader an accurate *lay of the land* so to speak and enable remaining challenges to be successfully tackled and overcome.

References

[1] M. Polese, L. Bonati, S. D'oro, S. Basagni, T. Melodia, Understanding o-ran: Architecture, interfaces, algorithms, security, and research challenges, IEEE Communications Surveys & Tutorials 25 (2023) 1376–1411.

[2] E. Bastug, M. Bennis, M. Medard, M. Debbah, Toward interconnected virtual reality: Opportunities, challenges, and enablers, IEEE Communications Magazine 55 (2017) 110–117. doi:10.1109/MCOM.2017.1601089.

[3] O-RAN Alliance, The O-RAN Alliance, `https://www.o-ran.org/`, 2024. Accessed: February 2024.

[4] B. Brik, H. Chergui, L. Zanzi, F. Devoti, A. Ksentini, M. S. Siddiqui, X. Costa-Pérez, C. Verikoukis, A survey on explainable ai for 6g o-ran: Architecture, use cases, challenges and research directions, arXiv preprint arXiv:2307.00319 (2023).

[5] L. Bonati, S. d'oro, M. Polese, S. Basagni, T. Melodia, Intelligence and learning in o-ran for data-driven nextg cellular networks, IEEE Communications Magazine 59 (2021) 21–27. doi:10.1109/MCOM.101.2001120.

[6] M. Kochaki, V. Marojevic, Actor-critic network for o-ran resource allocation: xapp design, deployment, and analysis, 2022. doi:10.48550/arXiv.2210.04604.

[7] U. M, G. Sadashivappa, R. Palepu, Integration of ric and xapps for open -radio access network for performance optimization, 2023, pp. 1–4. doi:10.1109/CSITSS60515.2023.10334159.

[8] F. Kavehmadavani, V.-D. Nguyen, T. X. Vu, S. Chatzinotas, Empowering traffic steering in 6g open ran with deep reinforcement learning, IEEE Transactions on Wireless Communications 23 (2024) 12782–12798. doi:10.1109/TWC.2024.3396273.

[9] A. Lacava, M. Polese, R. Sivaraj, R. Soundrarajan, B. S. Bhati, T. Singh, T. Zugno, F. Cuomo, T. Melodia, Programmable and customized intelligence for traffic steering in 5g networks using open ran architectures, IEEE Transactions on Mobile Computing 23 (2023) 2882–2897.

[10] I. Diógenes, L. Medeiros, P. Alves, M. Goldbarg, V. Lopes, D. Flor, W. Barros, V. Filho, V. Jr, E. Aranha, A. Martins, M. Fernandes, R. Fontes, A. Venancio Neto, Prototyping near-real time ric o-ran xapps for flexible ml-based spectrum sensing, 2022, pp. 137–142. doi:10.1109/NFV-SDN56302.2022.9974940.

[11] M. Martínez-Morfa, C. Ruiz de Mendoza, C. Cervelló-Pastor, S. Sallent, Drl-based xapps for dynamic ran and mec resource allocation and slicing in o-ran, 2024, pp. 106–114. doi:10.1109/NoF62948.2024.10741435.

[12] J. Santos, A. Huff, D. da Silva, K. Cardoso, C. Both, L. Dasilva, Managing o-ran networks: xapp development from zero to hero, 2024. doi:10.48550/arXiv.2407.09619.

[13] M. S. Wani, M. Kretschmer, B. Schröder, A. Grebe, M. Rademacher, Open ran: A concise overview, IEEE Open Journal of the Communications Society (2024).

[14] L. Bonati, M. Polese, S. D'Oro, S. Basagni, T. Melodia, Open, programmable, and virtualized 5g networks: State-of-the-art and the road ahead, Computer Networks 182 (2020) 107516.

[15] O-RAN Alliance Working Group 1, O-RAN.WG1.OAD-R003-v12.00, Technical Specification, 2024. URL: `https://www.o-ran.org`, o-RAN Architecture Description (OAD) Specification.

[16] O. Orhan, V. N. Swamy, T. Tetzlaff, M. Nassar, H. Nikopour, S. Talwar, Connection management xapp for o-ran ric: A graph neural network and reinforcement learning approach, in: 2021 20th IEEE international conference on machine learning and applications (ICMLA), IEEE, 2021, pp. 936–941.

[17] O-RAN Alliance Working Group 8, O-RAN.WG8.AAD.0-R004-v13.00, Technical Specification, 2024. URL: https://www.o-ran.org, o-RAN Distributed Unit (DU) Specification.

[18] O-ran working group 1. (2020). o-ran operations and maintenance interface 4.0. accessed: Dec. 11, 2022, ????

[19] J. X. Salvat, J. A. Ayala-Romero, L. Zanzi, A. Garcia-Saavedra, X. Costa-Perez, Open radio access networks (o-ran) experimentation platform: Design and datasets, IEEE Communications Magazine 61 (2023) 138–144.

[20] A. Garcia-Saavedra, X. Costa-Perez, O-ran: Disrupting the virtualized ran ecosystem, IEEE Communications Standards Magazine 5 (2021) 96–103.

[21] O-RAN Alliance Working Group 2, O-RAN.WG2.Non-RT-RIC-ARCH-TR-v01.01, Technical Report, 2021. URL: https://www.o-ran.org, o-RAN Non-RT RIC Architecture Technical Report.

[22] O-RAN Alliance Working Group 2, O-RAN.WG2.Non-RT-RIC-ARCH-TR-v01.01, Technical Report, 2024. URL: https://www.o-ran.org, o-RAN Non-RT RIC Architecture Technical Report.

[23] O-RAN Alliance Working Group 2, O-RAN.WG2.Non-RT-RIC-ARCH-R004-v06.00, Technical Specification, 2024. URL: https://www.o-ran.org, o-RAN Non-RT RIC Architecture Specification.

[24] O-RAN Alliance Working Group 2, O-RAN.WG2.A1GAP-R004-v04.00, Technical Specification, 2024. URL: https://www.o-ran.org, o-RAN A1 General Aspects and Principles (A1-GAP) Specification.

[25] O-RAN Alliance Working Group 1, O-RAN.WG1.TS.Slicing-Architecture-R004-v13.01, Technical Specification, 2024. URL: https://www.o-ran.org, o-RAN Slicing Architecture Specification.

[26] O-RAN Alliance Working Group 3, O-RAN.WG3.E2GAP-R004-v06.00, Technical Specification, 2024. URL: https://www.o-ran.org, o-RAN E2 General Aspects and Principles (E2-GAP) Specification.

[27] S. Marinova, A. Leon-Garcia, Intelligent o-ran beyond 5g: Architecture, use cases, challenges, and opportunities, IEEE Access 12 (2024) 27088–27114.

[28] O-RAN Alliance Working Group 3, O-RAN.WG3.RICARCH-R003-v06.00, Technical Specification, 2024. URL: https://www.o-ran.org, o-RAN RIC Architecture Specification.

[29] O-RAN Alliance Working Group 6, O-RAN.WG6.O2-GAP-R004-v08.00, Technical Specification, 2024. URL: https://www.o-ran.org, o-RAN O2 General Aspects and Principles (O2-GAP) Specification.

[30] O-RAN Alliance Working Group 3, O-RAN.WG3.Y1GAP-R004-v01.01, Technical Specification, 2024. URL: https://www.o-ran.org, o-RAN Y1 General Aspects and Principles (Y1GAP) Specification.

[31] O-RAN Alliance Working Group 3, O-RAN.WG3.E2SM-KPM-R003-v05.00, Technical Specification, 2024. URL: https://www.o-ran.org, o-RAN E2 Service Model for Key Performance Measurement (E2SM-KPM).

[32] O-RAN Alliance Working Group 3, ORAN.WG3.E2SM-RC-R003-v06.00, Technical Specification, 2024. URL: https://www.o-ran.org, o-RAN E2 Service Model for RAN Control (E2SM-RC).

[33] O-RAN Alliance Working Group 3, O-RAN.WG3.E2SM-CCC-R003-v04.00, Technical Specification, 2024. URL: https://www.o-ran.org, o-RAN E2 Service Model for Cell Capacity and Coverage Optimization (E2SM-CCC).

[34] O-RAN Alliance Working Group 3, ORAN-WG3.E2SM-NI-v01.00.00, Technical Specification, 2024. URL: https://www.o-ran.org, o-RAN E2 Service Model for Network Intelligence (E2SM-NI).

[35] O-RAN Alliance Working Group 3, O-RAN.WG3.E2AP-R004-v06.00, Technical Specification, 2024. URL: https://www.o-ran.org, o-RAN E2 Application Protocol (E2AP) Specification.

[36] L. Bonati, M. Polese, S. d'oro, S. Basagni, T. Melodia, Intelligent closed-loop ran control with xapps in openran gym, 2022. doi:10.48550/arXiv.2208.14877.

[37] Confluence, On-boarding and deploying xapps, URL https://lf-o-ran-sc.atlassian.net/wiki/spaces/RICA/pages/13566308/On-boarding+and+Deploying+xApps (2024).

[38] O-RAN Alliance Working Group 2, O-RAN.WG2.AIML-v01.03, Technical Specification, 2021. URL: https://www.o-ran.org, o-RAN AI/ML Specification.

[39] F. Mungari, An rl approach for radio resource management in the o-ran architecture, in: 2021 18th Annual IEEE International Conference on Sensing, Communication, and Networking (SECON), IEEE, 2021, pp. 1–2.

[40] N. Ghafouri, J. S. Vardakas, K. Ramantas, C. Verikoukis, A multi-level deep rl-based network slicing and resource management for o-ran-based 6g cell-free networks, IEEE Transactions on Vehicular Technology (2024).

[41] N. F. Cheng, T. Pamuklu, M. Erol-Kantarci, Reinforcement learning based resource allocation for network slices in o-ran midhaul, in: 2023 IEEE 20th Consumer Communications & Networking Conference (CCNC), IEEE, 2023, pp. 140–145.

[42] S. Gopal, D. Griffith, R. A. Rouil, C. Liu, Adapshare: An rl-based dynamic spectrum sharing solution for o-ran, arXiv preprint arXiv:2408.16842 (2024).

[43] K. Boutiba, M. Bagaa, A. Ksentini, Radio resource management in multi-numerology 5g new radio featuring network slicing, in: ICC 2022-IEEE International Conference on Communications, IEEE, 2022, pp. 359–364.

[44] Q. Wang, S. Chetty, A. Al-Tahmeesschi, X. Liang, Y. Chu, H. Ahmadi, Energy saving in 6g o-ran using dqn-based xapp, 2024. doi:10.48550/arXiv.2409.15098.

[45] J. Dai, S. Mahboob, H. Wang, L. Liu, Intelligent handover management enabled by o-ran and deep reinforcement learning, in: 2024 IEEE 100th Vehicular Technology Conference (VTC2024-Fall), IEEE, 2024, pp. 1–6.

[46] R. Raftopoulos, S. D'Oro, T. Melodia, G. Schembra, Drl-based latency-aware network slicing in o-ran with time-varying slas, in: 2024 International Conference on Computing, Networking and Communications (ICNC), IEEE, 2024, pp. 737–743.

[47] M. Zangooei, M. Golkarifard, M. Rouili, N. Saha, R. Boutaba, Flexible ran slicing in open ran with constrained multi-agent reinforcement learning, IEEE Journal on Selected Areas in Communications 42 (2023) 280–294.

[48] M. L. Betalo, S. Leng, H. N. Abishu, F. A. Dharejo, A. M. Seid, A. Erbad, R. A. Naqvi, L. Zhou, M. Guizani, Multi-agent deep reinforcement learning-based task scheduling and resource sharing for o-ran-

empowered multi-uav-assisted wireless sensor networks, IEEE Transactions on Vehicular Technology 73 (2023) 9247–9261.

[49] A. K. Singh, K. K. Nguyen, Communication efficient compressed and accelerated federated learning in open ran intelligent controllers, IEEE/ACM Transactions on Networking (2024).

[50] N. Islam, F. Monir, M. M. Syeed, M. Hasan, M. F. Uddin, Federated learning integration in o-ran: A concise review, in: 2023 33rd International Telecommunication Networks and Applications Conference, IEEE, 2023, pp. 283–288.

[51] H. Erdol, X. Wang, P. Li, J. D. Thomas, R. Piechocki, G. Oikonomou, R. Inacio, A. Ahmad, K. Briggs, S. Kapoor, Federated meta-learning for traffic steering in o-ran, in: 2022 IEEE 96th Vehicular Technology Conference (VTC2022-Fall), IEEE, 2022, pp. 1–7.

[52] Y. Rumesh, D. Attanayaka, P. Porambage, J. Pinola, J. Groen, K. Chowdhury, Federated learning for anomaly detection in open ran: Security architecture within a digital twin, in: 2024 Joint European Conference on Networks and Communications & 6G Summit (EuCNC/6G Summit), IEEE, 2024, pp. 877–882.

[53] M. Kouchaki, A. S. Abdalla, V. Marojevic, Openai dapp: An open ai platform for distributed federated reinforcement learning apps in o-ran, in: 2023 IEEE Future Networks World Forum (FNWF), IEEE, 2023, pp. 1–6.

[54] H. Zhang, H. Zhou, M. Erol-Kantarci, Federated deep reinforcement learning for resource allocation in o-ran slicing, in: GLOBECOM 2022-2022 IEEE Global Communications Conference, IEEE, 2022, pp. 958–963.

[55] Z. Abou El Houda, H. Moudoud, B. Brik, Federated deep reinforcement learning for efficient jamming attack mitigation in o-ran, IEEE Transactions on Vehicular Technology 73 (2024) 9334–9343.

[56] A. Abouaomar, A. Taik, A. Filali, S. Cherkaoui, Federated deep reinforcement learning for open ran slicing in 6g networks, IEEE Communications Magazine 61 (2022) 126–132.

[57] F. Rezazadeh, L. Zanzi, F. Devoti, H. Chergui, X. Costa-Perez, C. Verikoukis, On the specialization of fdrl agents for scalable and distributed 6g ran slicing orchestration, IEEE Transactions on Vehicular Technology 72 (2022) 3473–3487.

[58] Q. Wang, S. Chetty, A. Al-Tahmeesschi, X. Liang, Y. Chu, H. Ahmadi, Energy saving in 6g o-ran using dqn-based xapp, arXiv preprint arXiv:2409.15098 (2024).

[59] O-RAN Alliance Working Group 11, O-RAN.WG11.TR.AIML-Security.O-R004-v02.00.02, Technical Report, 2024. URL: https://www.o-ran.org, o-RAN AI/ML Security Specification.

[60] H. Yang, C. Mcpeak, R. Nagampally, N. Tripathi, G. Anderson, J. H. Reed, Y. Yang, D. J. Jakubisin, Agile 5g networks: Advance traffic steering xapp for interference mitigation, in: MILCOM 2024-2024 IEEE Military Communications Conference (MILCOM), IEEE, 2024, pp. 300–305.

[61] A. Akman, P. Tehrani, P. Oliver, M. Hoffmann, M. Jones, J. Li, Energy saving and traffic steering use case and testing by o-ran ric xapp/rapp multi-vendor interoperability, 2024. doi:10.48550/arXiv.2409.19807.

[62] M. A. Habib, H. Zhou, P. E. Iturria-Rivera, Y. Ozcan, M. Elsayed, M. Bavand, R. Gaigalas, M. Erol-Kantarci, Machine learning-enabled traffic steering in o-ran: A case study on hierarchical learning approach, IEEE Communications Magazine (2024).

[63] M. Dryjański, Ł. Kułacz, A. Kliks, Toward modular and flexible open ran implementations in 6g networks: Traffic steering use case and o-ran xapps, Sensors 21 (2021) 8173.

[64] R. Ntassah, G. Dell'Aera, F. Granelli, xapp for traffic steering and load balancing in the o-ran architecture, 2023, pp. 5259–5264. doi:10.1109/ICC45041.2023.10278921.

[65] A. Wadud, F. Golpayegani, N. Afraz, Qacm: Qos-aware xapp conflict mitigation in open ran, IEEE Transactions on Green Communications and Networking PP (2024) 1–1. doi:10.1109/TGCN.2024.3431945.

[66] C. Valente, P. Valente, P. Rito, D. Raposo, M. LuÍs, S. Sargento, 5g ran and core orchestration with ml-driven qos profiling, in: IEEE INFOCOM 2024 - IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS), 2024, pp. 1–6. doi:10.1109/INFOCOMWKSHPS61880.2024.10620861.

[67] A. Samorzewski, A. Kliks, M. Dryjański, Qos-based rrm procedure for o-ran systems, ACM MobiCom '23, Association for Computing Machinery, New York, NY, USA, 2023. URL: https://doi.org/10.1145/3570361.3615737. doi:10.1145/3570361.3615737.

[68] S. Mhatre, F. Adelantado, K. Ramantas, C. Verikoukis, Intelligent qos-aware slice resource allocation with user association parameterization for beyond 5g o-ran-based architecture using drl, IEEE Transactions on Vehicular Technology (2024).

[69] F. Mungari, C. Puligheddu, A. Garcia-Saavedra, C. F. Chiasserini, O-ran intelligence orchestration framework for quality-driven xapp deployment and sharing, IEEE Transactions on Mobile Computing (2025).

[70] D. Anand, M. A. Togou, G.-M. Muntean, A machine learning-based approach for interference mitigation to enhance qos and qoe in 5g o-ran networks, in: 2024 IEEE 35th International Symposium on Personal, Indoor and Mobile Radio Communications (PIMRC), IEEE, 2024, pp. 1–6.

[71] D. Anand, M. A. Togou, G.-M. Muntean, Enhancing qoe diversity in hetnets through interference mitigation with ml-based xapp in a 5g o-ran architecture, 2024, pp. 5473–5478. doi:10.1109/ICC51166.2024.10623003.

[72] G. Kougioumtzidis, V. Poulkov, Z. D. Zaharis, P. I. Lazaridis, Intelligent and qoe-aware open radio access networks, in: 2022 3rd URSI Atlantic and Asia Pacific Radio Science Meeting (AT-AP-RASC), 2022, pp. 1–4. doi:10.23919/AT-AP-RASC54737.2022.9814435.

[73] B. Agarwal, M. A. Togou, M. Ruffini, G.-M. Muntean, Qoe-driven optimization in 5g o-ran-enabled hetnets for enhanced video service quality, IEEE Communications Magazine 61 (2023) 56–62. doi:10.1109/MCOM.003.2200229.

[74] G. Kougioumtzidis, A. Vlahov, V. Poulkov, Z. Zaharis, P. Lazaridis, Qoe-oriented open radio access networks for virtual reality applications, in: 2022 25th International Symposium on Wireless Personal Multimedia Communications (WPMC), IEEE, 2022, pp. 491–496.

[75] C. Benzaïd, F. M. Hossain, T. Taleb, P. M. Gómez, M. Dieudonne, A federated continual learning framework for sustainable network anomaly detection in o-ran, in: 2024 IEEE Wireless Communications and Networking Conference (WCNC), IEEE, 2024, pp. 1–6.

[76] D. Attanayaka, P. Porambage, M. Liyanage, M. Ylianttila, Peer-to-peer federated learning based anomaly detection for open radio access networks, in: ICC 2023-IEEE International Conference on Communications, IEEE, 2023, pp. 5464–5470.

[77] O. Başaran, M. Basaran, D. Turan, H. Bayrak, Y. Sabucu, Deep autoencoder design for rf anomaly detection in 5g o-ran near-rt ric via xapps, 2023. doi:10.1109/ICCWorkshops57953.2023.10283501.

[78] P. K. Kakani, H. Djuitcheu, H. D. Schotten, Evaluation of xapps and their security in next-generation open radio access networks (oran), Authorea Preprints (2024).

[79] C.-F. Hung, Y.-R. Chen, C.-H. Tseng, S.-M. Cheng, Security threats to xapps access control and e2 interface in o-ran, IEEE Open Journal of the Communications Society (2024).

[80] T. O. Atalay, S. Maitra, D. Stojadinovic, A. Stavrou, H. Wang, Securing 5g openran with a scalable authorization framework for xapps, in: IEEE INFOCOM 2023-IEEE Conference on Computer Communications, IEEE, 2023, pp. 1–10.

[81] J. Groen, S. DOro, U. Demir, L. Bonati, M. Polese, T. Melodia, K. Chowdhury, Implementing and evaluating security in o-ran: Interfaces, Intelligence, and Platforms. arXiv (2023).

[82] N. Sapavath, B. Kim, K. Chowdhury, V. Shah, Experimental study of adversarial attacks on ml-based xapps in o-ran, 2023, pp. 6352–6357. doi:10.1109/GLOBECOM54140.2023.10437125.

[83] A. Chiejina, B. Kim, K. Chowhdury, V. K. Shah, System-level analysis of adversarial attacks and defenses on intelligence in o-ran based cellular networks, in: Proceedings of the 17th ACM Conference on Security and Privacy in Wireless and Mobile Networks, WiSec '24, Association for Computing Machinery, New York, NY, USA, 2024, p. 237–247. URL: https://doi.org/10.1145/3643833.3656119. doi:10.1145/3643833.3656119.

[84] M. Martínez-Morfa, C. R. De Mendoza, C. Cervelló-Pastor, S. Sallent, Drl-based xapps for dynamic ran and mec resource allocation and slicing in o-ran, in: 2024 15th International Conference on Network of the Future (NoF), IEEE, 2024, pp. 106–114.

[85] M. M. H. Qazzaz, L. Kulacz, A. Kliks, S. A. R. Zaidi, M. Dryjanski, D. McLernon, Machine learning-based xapp for dynamic resource allocation in o-ran networks, 2024, pp. 492–497. doi:10.1109/ICMLCN59089.2024.10625184.

[86] H. Zhang, H. Zhou, M. Erol-Kantarci, Team learning-based resource allocation for open radio access network (o-ran), in: ICC 2022-IEEE

International Conference on Communications, IEEE, 2022, pp. 4938–4943.

[87] A. Tripathi, J. S. Mallu, M. H. Rahman, A. Sultana, A. Sathish, A. Huff, M. R. Chowdhury, A. P. Da Silva, End-to-end o-ran control-loop for radio resource allocation in sdr-based 5g network, in: MILCOM 2023-2023 IEEE Military Communications Conference (MILCOM), IEEE, 2023, pp. 253–254.

[88] R. Smith, M. Farshchian, C. Freeberg, T. Machacek, D. Li, Enabling technologies for spectrum sharing with oran, in: 2024 IEEE International Symposium on Dynamic Spectrum Access Networks (DySPAN), IEEE, 2024, pp. 102–107.

[89] OpenRAN Gym — openrangym.com, https://openrangym.com/, ???? [Accessed 10-02-2025].

[90] A. Giannopoulos, S. Spantideas, L. George, K. Alexandros, P. Trakadas, Comix: Generalized conflict management in o-ran xapps – architecture, workflow, and a power control case, 2025. URL: https://arxiv.org/abs/2501.14619. arXiv:2501.14619.

[91] M. A. Habib, H. Zhou, P. E. Iturria-Rivera, M. Elsayed, M. Bavand, R. Gaigalas, Y. Ozcan, M. Erol-Kantarci, Intent-driven intelligent control and orchestration in o-ran via hierarchical reinforcement learning, in: 2023 IEEE 20th International Conference on Mobile Ad Hoc and Smart Systems (MASS), 2023, pp. 55–61. doi:10.1109/MASS58611.2023.00015.