POLITECNICO DI TORINO
Repository ISTITUZIONALE

XAI4C: An XAI-powered Conflict Detection Framework in O-RAN

Original
XAI4C: An XAI-powered Conflict Detection Framework in O-RAN / Varshney, N., Mungari, F., Puligheddu, C., Badawy, A., Chiasserini, C.F.. - ELETTRONICO. - (2025), pp. 45-50. (The 22nd IEEE International Conference on Mobile Ad-Hoc and Smart Systems (MASS 2025) Chicago (USA) 6-8 ottobre 2025) [10.1109/MASS66014.2025.00020].

Availability:
This version is available at: 11583/3002330 since: 2025-08-05T07:05:22Z

Publisher:
IEEE

Published
DOI:10.1109/MASS66014.2025.00020

Terms of use:

This article is made available under terms and conditions as specified in the corresponding bibliographic description in the repository

Publisher copyright
IEEE postprint/Author's Accepted Manuscript

©2025 IEEE. Personal use of this material is permitted. Permission from IEEE must be obtained for all other uses, in any current or future media, including reprinting/republishing this material for advertising or promotional purposes, creating new collecting works, for resale or lists, or reuse of any copyrighted component of this work in other works.

(Article begins on next page)

17 November 2026

XAI4C: An XAI-powered Conflict Detection Framework in O-RAN

Nancy Varshney[†], Federico Mungari[*], Corrado Puligheddu[†*], Ahmed Badawy[‡], Carla Fabiana Chiasserini[†*§]

[†]Politecnico di Torino, Italy [*]CNIT, Italy [‡]Qatar University, Qatar [§]Chalmers University of Technology, Sweden

***Abstract*—The Open Radio Access Network (O-RAN) architecture is key to enabling AI-driven dynamic network management. However, the complexity of this architecture introduces challenges, especially in managing conflicts between different AI-driven applications that operate concurrently within the network. These conflicts, if left unchecked, can lead to degraded network performance and service disruptions. To address this issue, we propose XAI4C (Explainable AI for Conflict Detection), a framework that leverages the SHAP (SHapley Additive exPlanations) explainable AI technique. XAI4C enhances transparency and interpretability in AI decision-making by helping network operators understand the factors driving AI decisions across different network components thereby allowing for early detection of conflicts between applications. In this paper, we first present the architecture and operation of the XAI4C framework. We then demonstrate its effectiveness in conflict detection through two case studies related to network slicing. Our results demonstrate that XAI4C outperforms the state-of-the-art PACIFISTA providing a detection accuracy increase up to 30%, while reducing the number of samples required for conflict detection by 41.17%.**

***Index Terms*—O-RAN, Explainable AI, conflict detection, SHAP, network slicer**

I. Introduction

Recent research and standardization efforts have highlighted Open Radio Access Network (O-RAN) as the transformative shift in the RAN domain for a seamless expansion into new 5G, and beyond 5G, services. The O-RAN architecture, promoted by the Open RAN Alliance, introduces RAN Intelligent Controllers (RICs) where mobile network operators (MNO) deploy custom logic to manage and control the network across various time scales. While the Non-real-time (RT) RIC manages and supports via rApps on a timescale longer than 1 s, the near-RT RIC enables dynamic control policies through xApps operating on a 10 ms–1 s timescale. Artificial intelligence (AI) and machine learning (ML) are the cornerstones enabling the optimization of cellular networks through closed-loop, automated, and agile network control policies.

However, the inherent complexity of the O-RAN environment introduces significant challenges, particularly regarding conflict management. As multiple applications, or xApps and rApps, operate concurrently across the network, the chances for conflicting control policies arise, which can compromise performance and service reliability. For instance, different xApps targeting, respectively, the antenna's electrical tilt and the cell individual offset configuration parameters to optimize the cell handover boundaries may pursue their different respective objectives and have an uncontrollable or inadvertent impact on the system performance [1]. Effective conflict detection is thus critical to identifying and addressing these issues before the key performance indicators (KPIs) performance degradation escalates [2]. Timely detection mechanisms are indeed essential for maintaining operational efficiency, optimizing resource utilization, and ensuring seamless service delivery. It follows that addressing the challenges of conflict detection in O-RAN is vital for harnessing the full potential of this innovative architecture, ultimately enabling more resilient and adaptive network management.

State of the art and research gap. Current conflict management frameworks in O-RAN primarily focus on conflict avoidance [1]. While this approach may be effective in smaller deployments, it significantly limits the advantages of RICs by failing to account for scenarios where conflicting applications can coexist and operate under specific conditions. The impracticality of a centralized entity managing the entire network exacerbates this limitation, given the combinatorial complexity of network states and actions, which can lead to inefficiencies and suboptimal resource allocation. A more promising direction for conflict management is adopting a distributed approach. In this case, multiple xApps and rApps collaboratively manage distinct network segments, enhancing scalability and programmability while enabling tailored solutions to meet the diverse requirements of varying network environments. However, ensuring that these AI-driven, multi-vendor components do not inadvertently generate conflicting control policies poses a significant challenge. Conflicts may arise from several sources, like divergent objectives among applications, temporal dependencies, and varying impacts on multiple KPIs. Thus, conflict detection mechanisms that can work effectively in a distributed system become essential.

Explainable AI (XAI) has emerged as a possible approach to proactively enhance AI decision processes in resource allocation, anomaly detection, and adaptive policy management [3]. However, these approaches focus on ensuring the transparency of AI models for different network management tasks rather than on detecting conflicts among the xApps and rApps as specific use case. Though proactive methods aim to prevent conflicts, these struggle with the complexities of real-time O-RAN environments. In contrast, a reactive approach to conflict detection in Near-RT and Non-RT RIC provides clearer

This work was supported by the Qatar Research Development and Innovation Council ARG01-0525-230339. The content is solely the responsibility of the authors and does not necessarily represent the official views of Qatar Research Development and Innovation Council. The work was also supposed by the EU under the Italian NRRP of NextGenerationEU, partnership on PE00000001, program "RESTART".

insights into existing conflicts, adapts to real-time conditions, and reduces computational overhead and false positives.

Technical challenges. Integrating XAI into the complex O-RAN architecture requires seamless communication to facilitate real-time decision-making. Executing complex XAI analyses can be resource-intensive, potentially causing delays in conflict detection. As the number of xApps and rApps increases, the framework must scale effectively, allowing XAI analysis to handle a growing number of parameters while efficiently managing diverse and dynamic network conditions. Further, it is crucial to maintain an efficient feedback loop between conflict mitigation outcomes and the AI/ML catalog for continuous learning without introducing extra overhead. Finally, careful evaluation of XAI outputs is necessary, as noise in output values can hinder conflict detection. Establishing appropriate thresholds for conflict scores is thus vital to guarantee accuracy and reliability of the detection process.

Our solution: XAI for conflict detection. We leverage XAI to develop a solution framework that enhances O-RAN network management through *transparency* and *scalability*, while effectively addressing the above technical challenges. Our framework, named XAI for Conflict detection (XAI4C), provides the following advantages. *(i)* It exploits SHAP (SHapley Additive exPlanations), an XAI technique that uses Shapley values from cooperative game theory to quantify each feature's contribution to a model outputs. The understanding provided by SHAP aids in identifying the factors driving decisions, which is particularly useful when using ML-driven xApps and rApps in O-RAN. *(ii)* XAI4C's modular architecture supports scalability across diverse deployment scenarios, from small to large multi-region implementations. This allows conflict management mechanisms to adapt as O-RAN deployments grow, ensuring efficiency in complex environments. *(iii)* Thanks to SHAP, XAI4C provides detailed explanations of AI-driven actions, helping operators justify decisions, identify conflicts, and adjust policies accordingly. Through SHAP analysis across layers, including Non-RT RIC and xApps, XAI4C ensures that AI-driven decisions are interpretable and trustworthy. *(iv)* XAI4C acts swiftly, enabling rapid and seamless operations in complex, dynamic environments.

Summary of novel contributions.

- We envision the XAI4C framework's architecture as integrated within an O-RAN environment, and outline the comprehensive workflow for conflict management;
- We present the XAI4C algorithmic solution for conflict detection in O-RAN. The proposed solution integrates the SHAP XAI technique to improve transparency and interpretability in decision-making. Specifically, XAI4C effectively interprets AI model outputs, allowing for the identification of potential conflicts through the analysis of shifts in KPIs;
- We validate the effectiveness of XAI4C in detecting conflicts among different xApps in O-RAN for two case studies related to network slicing. Besides demonstrating XAI4C's applicability to real-world scenarios, our results show that, when compared to the state-of-the-art solution [4], XAI4C increases detection accuracy by up to 30% while reducing the time required for conflict detection by over 40%.

II. The XAI4C Framework

This section introduces the XAI4C framework for conflict detection in O-RAN systems. We first describe the framework architecture for conflict management (Sec. II-A), and then our solution for conflict detection (Sec. II-B). We remark that XAI4C is specifically designed to ensure transparent and interpretable decision-making across various network components, leveraging on SHAP analysis for conflict detection.

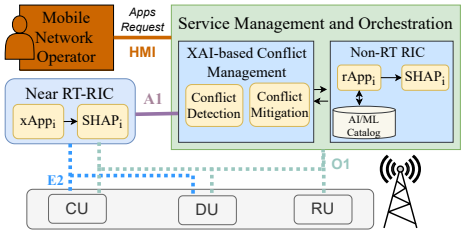


Fig. 1: XAI-based conflict management architecture.

A. XAI4C system architecture

Fig. 1 shows the different components of XAI4C, which are detailed below. The central unit (CU), distributed unit (DU), and radio unit (RU) collaboratively manage processing, transmission, and signal operations in the RAN. The CU and DU, working closely with the Near-RT RIC, execute low-latency actions driven by real-time optimization decisions from xApps, which continuously monitor and adjust the network. The Near-RT RIC is responsible for real-time network optimization and control and it includes xApps, specialized applications that perform localized tasks. Each xApp is integrated with SHAP-based explainability tools to provide real-time insights into their decisions, elucidating how conflicts may arise from misaligned objectives or policy clashes. This integration enables the xApps to compute SHAP values directly, which helps them understand and justify their actions with minimal communication overhead. The Near-RT RIC also interfaces with the CU and DU via the E2 interface for immediate control over RAN resources. The RU, responsible for signal transmission and reception, connects to the Near-RT RIC through the O1 interface, allowing it to receive real-time updates for dynamic network performance optimization.

At the core of the architecture is the service management and orchestration (SMO) block, which plays a crucial role in coordinating both long-term and real-time network operations. It includes the Non-RT RIC coupled with its own SHAP engine and the XAI-based conflict management system. The Non-RT RIC is tasked with strategic decision-making, like policy control, resource planning, and optimization of ML models. It also houses the AI/ML catalog, which manages various models and algorithms guiding network operations. Notably, it employs SHAP analysis to interpret AI model outputs and analyze shifts in KPIs on a longer time horizon. Within the SMO, the XAI-based conflict management system is pivotal for detecting and resolving conflicts. The SMO

connects to other components through key interfaces, such as the A1 interface to the Near-RT RIC, which translates strategic goals into real-time operations, and the human-machine interface (HMI), which allows mobile network operators to submit service requests and manage high-level intents.

In instances where a conflict is identified among xApps, such as contradictory service requirements, the Near-RT RIC communicates with the SMO through the A1 interface, and the mitigation module within the XAI-based system activates appropriate mechanisms. To handle conflict between xApps and rApps, which operate on different time scales, it is important to define thresholds, priorities, and timing windows for each application. Setting these parameters allows the SMO to identify potential conflicts in real-time or at scheduled intervals, ensuring smooth coordination between xApps and rApps. Synchronization mechanisms can then be used to maintain system stability and prevent conflicts from affecting network performance. Additionally, the conflict mitigation block in the SMO updates the AI/ML catalog for continuous learning and informs xApps of any new policies. Thus, the framework incorporates a continuous feedback loop between conflict detection and mitigation outcomes and the AI/ML catalog, promoting continuous learning. This feature enables the system to anticipate and prevent similar conflicts in future operations. Within this framework, our primary focus is on the design of the detection mechanism, ensuring it accurately identifies conflicts among xApps.

B. XAI4C: Conflict detection

The process begins with the deployment and activation of application or service requests submitted by the MNO via the HMI. Such requests are also processed by the near-RT RIC as well as SMO which evaluates the new requirements against existing network policies using the AI/ML models in the Non-RT RIC. Once the configurations are updated, the SHAP analysis engine in both the conflict detection block within the Near-RT RIC and the SMO assesses how these changes impact the KPIs, offering interpretable insights into potential conflicts.

The methodology begins by defining the following key sets. $\mathcal{S}$ for network operational condition parameters ($s{\in}\mathcal{S}$), such as number of users and network load, average signal-to-noise ratio (SNR), and active xApps. $\mathcal{K}$ is the set of KPIs ($k{\in}\mathcal{K}$), such as throughput and latency. $\mathcal{P}$ is the set of controllable network configuration parameters ($p{\in}\mathcal{P}$), like radio resource allocation and modulation coding scheme (MCS). $\mathcal{A}$ is the set of active xApps ($a{\in}\mathcal{A}$), with $\mathcal{P}_a{\subset}\mathcal{P}$ being the set of network configuration parameters controlled by a. The system observes and stores SHAP value distributions $x(s,a)$ for each xAapp a and each condition s.

SHAP values are calculated using a perturbation-based method [5]. Each feature is perturbed while keeping the other features fixed at their baseline values. The model's response to these perturbations is examined by assessing the resulting predictions. The differences between these predictions and the baseline prediction are then calculated to measure the impact of each feature when it diverges from its baseline value.

When KPI degradation is detected under condition $\tilde{s}$, the conflict detection module in the SMO is triggered. This module identifies a set of network conditions, $\{\hat{s}\}$ that are operationally similar to $\tilde{s}$. The SHAP values, denoted by $x(\hat{s},a)\,\forall\{\hat{s}\}$, are then used to train a classifier, which determines if the SHAP values $x(\tilde{s},a)$ for KPIs $k{\in}\mathcal{K}$ indicate that the KPI degradation is due to conflicts among xApps or external environmental factors, based on whether the SHAP values exceed a specified threshold.

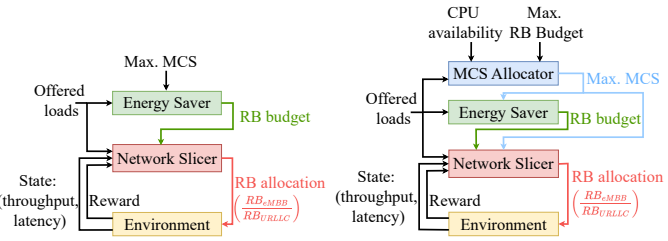


Fig. 2: System model of energy saver and network slicer interaction with the environment (left); system model with MCS Allocator added at the top (right).

III. XAI4C Effectiveness: Results and Discussion

In this section, we first present a general system model of conflict detection in O-RAN using the XAI4C framework, simulation environment, and the selected state-of-the-art benchmark. Then, we present the numerical analysis for conflict detection in two relevant case studies in O-RAN.

System Overview. Network slicing allows creating customized virtual networks for different services. Notably, enhanced Mobile Broadband (eMBB) aims for high data speeds, while ultra-Reliable Low Latency Communications (uRLLC) focuses on minimal delays. This study explores how XAI4C manages conflicts between network slicing and other strategies like energy-saving and MCS control strategies. As shown in Fig. 2, the system consists of a network slicing xApp alongside others like an energy saver xApp and MCS Allocator xApp. The network slicing xApp manages virtual networks, ensuring resources are allocated to meet the specific requirements of each slice, such as eMBB or uRLLC, optimizing performance metrics like throughput and latency. Conflicts arise when other xApps, such as the energy saver, reduce resource availability, affecting the performance of network slices. The network slicer uses autoencoders to process input metrics, extracting essential features for decision-making on resource allocation. When an autoencoder is introduced, the input data is first compressed into a latent space before being passed to the decision-making model. The compressed metrics inform a Deep Reinforcement Learning (DRL) mechanism based on an actor-critic architecture. This DRL agent learns to optimize resource allocation using historical data. The agent considers metrics like latency, load, throughput, and the available resource budget to decide how many RBs to assign to each

slice. The reward system evaluates performance by combining throughput and latency, encouraging efficient resource use.

Simulation environment. We analyze SHAP values in three network operation scenarios $\mathcal{S}=\{s, \tilde{s}, s'\}$. Here, s: refers to a stable condition with no KPI degradation, where only the network slicer is active; $\tilde{s}$: represents the scenario where KPI degradation is caused by the activation of additional xApps; and, s': denotes the scenario where the energy saver and MCS Allocator xApps are inactive but there is a KPI degradation due to external factors such as high variability in the SNR values. To ensure consistent conditions for our study, we stabilize key factors like the load for each slice and the average SNR value. First, we evaluate the network slicer's performance using SHAP values $x(s, a)$ to establish a baseline for latency, load, throughput, and RB budget using condition s for high SNR scenarios. The SHAP values are calculated over sliding windows of length N samples each, i.e., over N observations. The network slicer is a DRL-based xApp trained for two slices using the Colosseum O-RAN data set available on [6]. We perturb the raw input features and observe how these changes propagate through the autoencoder to affect the latent representation and, eventually, the DRL's output.

Next, we introduce additional xApps, $a\in\mathcal{A}$, such as an energy saver and an MCS Allocator, which control the parameter $p\in\mathcal{P}$, where $\mathcal{P}=\{\text{RB budget}, \text{maximum MCS}\}$. This is done to create a conflict and to re-evaluate the performance of the network slicer, allowing us to observe the effects of limited resources under conditions $\tilde{s}$ and s'.

Subsequently, we intentionally introduce KPI degradation unrelated to conflicts by increasing the variability in the channel conditions by 20%, to yield condition s'. By using the *shap.KernelExplainer*, we compute the SHAP values again and the SHAP distance, $x(s, a)-x(\tilde{s}, a)$ between s and $\tilde{s}$. This metric aims to reveal if conflicts introduce significant behavioral changes in the network slicer across KPIs $k\in\mathcal{K}$. This tool assesses the contribution of each feature by varying them while keeping others constant, allowing us to calculate SHAP values that highlight significant factors affecting conflict detection and performance changes. The shap.KernelExplainer is trained on the same dataset used to train the network slicer.

Benchmark. We compare XAI4C's conflict detection capabilities to those of PACIFISTA [4]. PACIFISTA uses statistical data from O-RAN xApps and organizes them into hierarchical graphs to predict when conflicts may happen and how serious they could be. It utilizes the Kolmogorov-Smirnov (K-S) distance to quantify the differences between two Empirical Cumulative Distribution Functions (ECDFs). By comparing the ECDFs of KPIs under conflict scenarios to those from non-conflict ones, it evaluates the conflicts severity. For the $\tilde{s}$ and s' scenarios, PACIFISTA treats s (i.e., a stable condition with no KPI degradation where only the network slicer is active) as a baseline. For each KPI $k\in\mathcal{K}$, it calculates the ECDFs $F_s(k), F_{\tilde{s}}(k)$, and $F_{s'}(k)$, and the K-S distance between s and $\tilde{s}$ (or s') as $D_{\tilde{s},s}=\max|F_s(k)-F_{\tilde{s}}(k)|$. In the following case studies, we provide comparative analysis of the SHAP values, used in XAI4C, and the K-S distances, used in PACIFISTA.

A. Case study with Energy Saver and Network Slicer xApps

As shown in Fig. 2(left), in this case study, the system includes two key applications: the network slicer xApp and the energy saver xApp. The network slicer xApp manages virtual networks for eMBB and uRLLC, ensuring resources are allocated to meet specific performance requirements such as throughput and latency. Conflicts arise when the energy saver xApp reduces the resource block (RB) budget to save energy, limiting the resources for the network slicer. The energy saver xApp predicts the necessary RB budget by analyzing historical throughput data, with a total RB budget capped at 50, considering a maximum available MCS. It dynamically adjusts allocations based on demand, defaulting to the maximum budget when necessary to meet network requirements. If no allocations suffice, it defaults to the maximum RB budget value, allowing for dynamic adjustments based on demand.

Numerical analysis. Fig. 3(left) and Fig. 3(center-left) display the SHAP distances for the key features latency, offered load, throughput, and RB budget. Fig. 3(left) depicts the SHAP distances when the energy saver xApp restricts the RB budget to the minimum number of RBs needed to meet the combined mean load requirements of eMBB and uRLLC slices over the observing window (which depends on the current traffic demand). Fig. 3(center-left) shows the SHAP distances for KPI degradation caused by external factors (e.g., high variability in the SNR values) also in comparison to the baseline scenario s. In Fig. 3(left), a high SHAP distance across all features suggests resource competition, compelling the system to prioritize certain features to mitigate negative impacts, thereby elevating their importance. Similarly, in Fig. 3(center-left), the decreased reliability of predictions regarding future states (e.g., load or network conditions), due to high variability in the SNR values, necessitates greater reliance on real-time indicators such as latency and throughput for resource allocation, rather than on the RB budget. Moreover, the RB budget in scenario s' is fixed as in the baseline scenario, and only the slicing policy distributes the RBs, leading to nearly zero SHAP distance for this feature (Fig. 3(center-left)). Conversely, in scenario $\tilde{s}$, the RB budget is dynamically adjusted by activation of the energy saver xApp, making it more influential in the model's decision-making process, hence higher SHAP distance (Fig. 3(left)).

Additionally, Fig. 3(left) and Fig. 3(center-left) compare the SHAP distance of XAI4C against the K-S distance metric used in PACIFISTA. The distribution of RB budget is a constant value of 50 when the energy saver xApp is inactive, while it exhibits a distribution when the energy saver is active. Therefore, in Fig. 3(left), the K-S distance for the RB budget is 1, indicating significant distribution shifts, whereas it is 0 in Fig. 3(center-left), suggesting no shifts in that scenario. For the rest of the KPIs, the distribution is non-constant in all scenarios hence, their K-S value is below 1. In summary, non-zero K-S distance indicates significant distribution shifts, whereas the non-zero SHAP distance reflects the features impact on predictions, emphasizing that K-S and SHAP capture different aspects of the xApp's behavior.

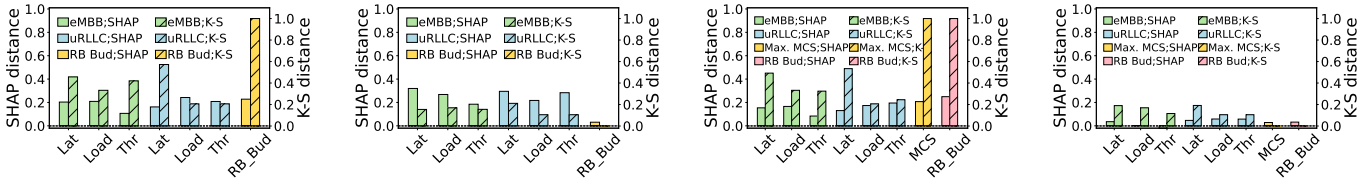


Fig. 3: SHAP distance (used by XAI4C) and K-S distance (used by our benchmark) for latency (Lat), offered load (Load), throughput (Thr), maximum MCS, and RB budget (RB Bud), for the case study without (left plots) and with (right plots) MCS Allocator. For each pair of plots, the first one refers to the distance between conflict scenario $\tilde{s}$ and baseline scenario s, while the second plot to the scenario with external KPI degradation factors s' and the baseline scenario s.

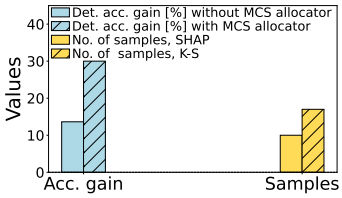


Fig. 4: Detection accuracy gain of XAI4C over the benchmark and required number of observations for each method.

In Fig. 3(left) and Fig. 3(center-left), we consider a baseline scenario s where the RB budget is fixed at 50. We now introduce an additional upper layer that controls the RB budget available to the network slicer based on the load profile, besides the energy saver xApp. The higher layer limits the RB budget to the minimum number of RBs that would be necessary to meet the combined offload demands of both eMBB and uRLLC, ensuring that the throughput can adequately support 1.6 times the total mean load requirement for the observation window. As the complexity of this scenario makes it challenging to determine the threshold for detecting conflicts, as described in Sec. II-B, we employ an ML-based classifier, namely, the *RandomForestClassifier*, along with the latency and throughput KPIs of both eMBB and uRLLC slices, to ascertain whether the KPI degradation raised from the conflict energy saver xApp or not.

Fig. 4 highlights that XAI4C outperforms our benchmark by 13.63% in detection accuracy. It is worth remarking that detection accuracy can be further enhanced by employing a more effective classifier, which we plan to explore in future work. Also, importantly, the K-S metric measures the distance between distributions and requires a larger number of samples, resulting in slower conflict detection. Notably, the accuracy achieved in Fig. 4 required an observing window of $N{=}10$ samples for XAI4C and of $N{=}17$ samples for our benchmark, leading to a 41.17% reduction in the detection time.

B. Conflict detection with Energy Saver, Network Slicers, and MCS Allocator

For this case study, the system model (Fig. 2(right)) integrates three primary components: the Network Slicer xApp, the Energy Saver xApp, and the MCS Allocator. The model operates within an environment characterized by key performance states, including throughput and latency, along with offered loads, CPU availability, and RB budget constraints. The MCS Allocator xApp determines the maximum MCS based on CPU availability and an RB budget capped at 50. This component ensures that the network maintains optimal transmission quality while adhering to hardware resource limitations. The relationship between CPU usage, the available RB budget, and the MCS index is discussed in detail in [7]. For numerical analysis, we consider CPU availability for 50 RBs to be 70%. The Energy Saver xApp utilizes the output from the MCS Allocator to predict the required RB budget for the offered loads, similar to the methodology outlined in the first case study in III-B.

Numerical analysis. Fig. 3(center-right) and Fig. 3(right) display the SHAP distances for the key features latency, offered load, throughput, maximum MCS, and RB budget between the scenarios $\tilde{s} - s$ and $s'{-}s$. Fig. 3(center-right) shows the SHAP distances when both the energy saver xApp and MCS Allocator xApp are active, which restrict the RB budget and maximum MCS (resp.). This restriction leads to higher SHAP distances for all the features, reflecting their combined impact on maintaining system performance under constrained resources. In contrast, Fig. 3(right) presents the SHAP distances when the energy saver and MCS Allocator xApps are inactive. Here, KPI degradation arises from high variations in the SNR values adversely affecting achievable latency and throughput. Consequently, the SHAP distances for all features are smaller compared to the conflict scenario $\tilde{s}$, especially for maximum MCS and RB budget.

Furthermore, Fig. 3(center-right) and Fig. 3(right) compare the XAI4C SHAP distances against the PACIFISTA K-S distance. In this case study, for scenarios s' and s, both the RB budget and maximum MCS remain constant at 50 and 27, respectively, resulting in a K-S distance of zero (Fig. 3(right)). For scenario $\tilde{s}$, both maximum MCS and RB budget have distributions, yielding a K-S distance of 1. The K-S distances for other features fall between 0 and 1, with higher values in Fig. 3(center-right) compared to Fig. 3(center-right), indicating a greater shift in the distribution of each feature.

Similar to the analysis in III-A, we now consider an upper layer that controls both the maximum MCS and the RB budget available to the network slicer and sets the maximum MCS based on maintaining CPU availability at 80% for the maximum RB budget of 50. Given maximum MCS availability, it then decides the RB budget to support 1.6 times the total mean load requirement for the observation window. Again, using

the SHAP values of throughput and latency of both slices, we detect whether the KPI degradation is from the additional xApps or from the external environment. Remarkably, as shown in Fig. 4, this approach results in a 30% increase in detection accuracy compared to the benchmark. Additionally, XAI4C still allows for a 41.17% reduction in the number of samples that are necessary for detecting a conflict.

IV. Related Work

We discuss below the existing work on the three main areas related to our study.

Conflict mitigation in O-RAN. In this context, [8] proposes a Deep Q-learning-based team learning algorithm where two cooperating xApps work in an O-RAN setting. However, this method has limited scalability, as its decision logic must adapt to the number of cooperating xApps. [9] presents a conflict mitigation framework that relies on real-time monitoring and analysis of control messages and performance management data. By continuously tracking xApp control messages, the framework dynamically detects and resolves conflicts. While this approach improves xApp coordination and real-time decision-making, it faces challenges related to its reliance on timely, accurate data. Any lag or inaccuracies in the data could hinder performance, and increasing network complexity introduces computational overhead, raising scalability concerns. Moreover, the predefined conflict detection strategies may struggle to adapt to unforeseen conflict scenarios in evolving networks. The xApp distillation scheme in [10] offers a conflict mitigation approach by creating an enhanced xApp that learns from multiple conflicting xApps using policy distillation. However, the method has limitations when dealing with a large number of conflicting xApps, as distillation can become computationally demanding and may fail to capture subtle differences in decision-making policies.

XAI in O-RAN. [11] proposes SliceOps, which integrates deep RL with XAI to optimize slices resource allocation through rApps. The latter uses SHAP to analyze data, train models, and deploy them as xApps on the Near-RT RIC. However, SliceOps focuses on operational resource management and AI service deployment within an application, rather than on conflict resolution in O-RAN. Also, [12] makes DRL-based decision-making more transparent by providing contextual explanations of wireless behavior through attributed graphs.

Conflict detection in O-RAN. Two main methods exist: KPI-based monitoring and control message exchange. In the first one, conflicts are detected by monitoring KPIs. For example, PACIFISTA [4] leverages statistical data from xApps and rApps, combined with hierarchical graphs, to map relationships between control parameters and KPIs. This mapping facilitates identification and resolution of potential conflicts among applications, leading to a refined conflict mitigation process. The second method involves control message exchange between O-RAN components [9]. By monitoring the real-time decisions exchanged between components like xApps and the RIC, conflicts are detected dynamically as inconsistencies in control messages, providing more immediate conflict identification than KPI monitoring.

Novelty. Our XAI4C framework offers a novel approach to conflict detection in O-RAN by leveraging SHAP values to provide detailed insights into the factors contributing to conflicts. Unlike traditional methods that depend on statistical analysis or control message exchanges, XAI4C captures complex interactions among various KPIs, allowing for a comprehensive understanding of how these variables influence conflict scenarios. Its architecture enables real-time conflict detection while fostering continuous learning, ultimately enhancing decision interpretability and laying the groundwork for proactive conflict mitigation strategies.

V. Conclusions and Future Work

O-RAN plays a crucial role in enabling AI-driven network management, but it also presents challenges in handling conflicts among various applications. Our XAI4C framework tackles these challenges by using SHAP-based XAI for a scalable and reliable solution in conflict detection, ensuring efficient network operations. We showed that XAI4C effectively detects KPI degradation caused by conflicts between xApps, distinguishing it from that caused by external factors. Compared to the state-of-the-art, XAI4C shows better detection accuracy (up to 30%), with a reduction (by 41.17%) in the number of samples needed for conflict detection. The flexible design of XAI4C also allows it to scale across different network conditions. XAI4C's insights lay the groundwork for future conflict mitigation efforts, making it a valuable tool for managing resource conflicts in dynamic environments.

References

[1] O-RAN Working Group 3, "O-RAN Near-RT RIC Architecture 6.0." O-RAN.WG3.RICARCH-R003-v06.00 Technical Specification, June 2024.

[2] C. Adamczyk, "Challenges for Conflict Mitigation in O-RAN's RAN Intelligent Controllers," in *SoftCOM*, 2023.

[3] B. Brik, H. Chergui, L. Zanzi, F. Devoti, A. Ksentini, M. S. Siddiqui, X. Costa-Pérez, and C. Verikoukis, "A survey on explainable ai for 6G O-RAN: Architecture, use cases, challenges and research directions," *arXiv preprint arXiv:2307.00319*, 2023.

[4] P. B. del Prever, S. D'Oro, L. Bonati, M. Polese, M. Tsampazi, H. Lehmann, and T. Melodia, "PACIFISTA: Conflict Evaluation and Management in Open RAN," *arXiv preprint arXiv:2405.04395*, 2024.

[5] S. Lundberg, "A unified approach to interpreting model predictions," *arXiv preprint arXiv:1705.07874*, 2017.

[6] "Colosseum ORAN dataset." https://github.com/wineslab/colosseum-oran-commag-dataset, 2024.

[7] S. Pramanik, A. Ksentini, and C. F. Chiasserini, "Cost-efficient slicing in virtual radio access networks," *Comp.. Comm.*, vol. 209, 2023.

[8] H. Zhang, H. Zhou, and M. Erol-Kantarci, "Team learning-based resource allocation for open radio access network (O-RAN)," in *IEEE ICC*, 2022.

[9] C. Adamczyk and A. Kliks, "Conflict mitigation framework and conflict detection in O-RAN near-RT RIC," *IEEE Comm. Mag.*, vol. 61, no. 12, 2023.

[10] H. Erdol, X. Wang, R. Piechocki, G. Oikonomou, and A. Parekh, "xApp Distillation: AI-based Conflict Mitigation in B5G O-RAN," *arXiv preprint arXiv:2407.03068*, 2024.

[11] F. Rezazadeh, H. Chergui, L. Alonso, and C. Verikoukis, "SliceOps: Explainable MLOps for Streamlined Automation-Native 6G Networks," *IEEE Wir. Comm.*, 2024.

[12] C. Fiandrino, L. Bonati, S. D'Oro, M. Polese, T. Melodia, and J. Widmer, "EXPLORA: AI/ML explainability for the Open RAN," *ACM CoNEXT*, 2023.