XAInomaly: Explainable, Interpretable and Trustworthy AI for xURLLC in 6G Open-RAN

Osman Tugay Basaran and Falko Dressler
School of Electrical Engineering and Computer Science, TU Berlin, Germany
Email: {basaran, dressler}@ccs-labs.org

***Abstract*—Artificial intelligence (AI) has already been incorporated into wide range applications of the fifth generation (5G) networks. The AI-native design of 6G network is serving as cornerstone for intelligent, autonomous, and dynamic network operations. AI-driven techniques, such as machine learning (ML) and Deep Learning (DL), facilitate real-time data analytics, predictive modeling, and decision-making processes to optimize resource utilization, enhance network performance, and ensure seamless connectivity for a multitude of devices and services. However, it is crucial in many respects that these AI algorithms are reliable, trustworthy, and explainable. In this direction, Explainable AI (XAI) will ensure transparent and secure operation at different layers of 6G networks. With the integration of XAI, 6G networks can achieve transparent dynamic self-configuration, self-optimization, and self-healing capabilities, enabling the network to adapt to fluctuating demands, mitigate potential issues proactively. To ensure that the AI/ML algorithms used in 6G Next-generation URLLC (xURLLC) use case are trustable and reliable, we proposed a XAInomaly framework that use our novel fastSHAP-C XAI method which handle real-time XAI layer operations on Open-RAN (O-RAN). Our performance results show that fastSHAP-C provides a 25% advance over its competitors in terms of resource utilization.**

***Index Terms*—6G, Explainable and Trustworthy AI, xURRLC, XAI, SHAP Values, fastSHAP-C, O-RAN, xApps**

I. Introduction

AI empowers 6G networks with intelligent service orchestration, dynamic spectrum management, and advanced security mechanisms, paving the way for innovative applications and services, such as large language models (LLMs), the metaverse including augmented and virtual reality, and holographic communications [1, 2]. Furthermore, next generation AI-driven edge computing and intelligence facilitate low-latency data processing and decision-making at the network edge, reducing the reliance on centralized data centers and enhancing the responsiveness and efficiency of 6G networks [3, 4].

In the landscape of 6G networks, the utilization of Explainable AI (XAI) is paramount due to its multifaceted benefits [5, 6]. Transparency and trustworthiness are vital pillars in the deployment of AI-driven functionalities within dynamic and complex network environments. In Figure 1, XAI techniques provide stakeholders with clear insights into the decision-making processes of AI models, fostering transparency and enhancing trust. Moreover, accountability is ensured as XAI enables the traceability and auditability of AI decisions, mitigating risks associated with opaque systems.

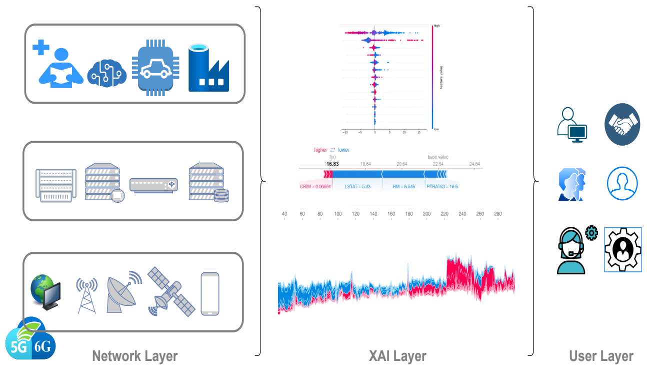

Figure 1. XAI Interaction Between User and Network Layers

The robustness and reliability of 6G networks are bolstered by XAI, facilitating rapid error detection and mitigation, thus improving overall network performance. Additionally, XAI supports ethical and regulatory compliance by identifying and mitigating biases, promoting fairness and inclusivity in AI-driven decision-making. Ultimately, the integration of explainable AI in 6G networks facilitates effective human-AI collaboration, empowering operators to leverage AI capabilities while retaining control and oversight, thus paving the way for the successful deployment and operation of next-generation wireless communication systems [7].

Especially with 6G towards Open Radio Access Networks (O-RAN) [8], black-box AI systems are inadequate in terms of reliability and trustworthiness [9, 10]. White-box AI systems are inevitable for users who handle the network from different perspectives, such as service providers, legal auditors, and end-users. In the near future, with 6G O-RAN deployments where network elements are disaggregated and decentralized, XAI enables local interpretation of AI decisions at the network edge, facilitating real-time adaptability and troubleshooting [10]. Furthermore, XAI techniques help mitigate biases and ensure fairness in resource allocation and network management, promoting inclusivity and equity in O-RAN operations as can be seen in Figure 2. The near-real-time RIC (Near-RT RIC) in the figure is responsible for the operation of the control loops work periodically between 10 ms and 1s. It enables AI/ML algorithms to intelligently manage the network via eXtended applications (xApps) running on top.

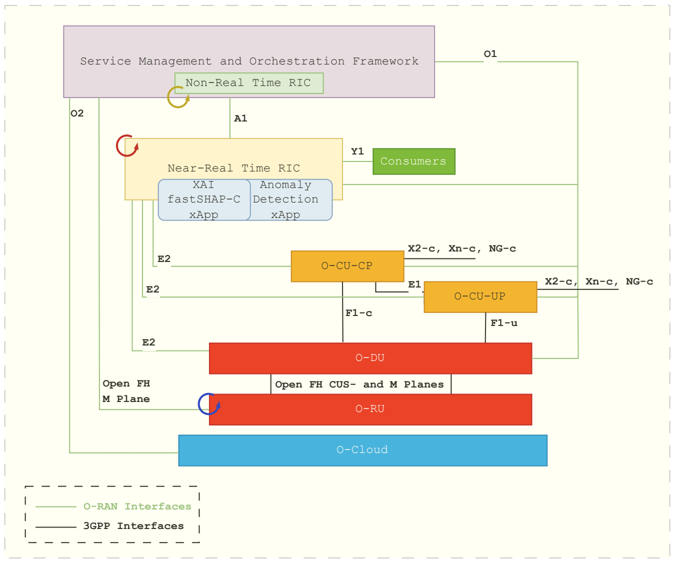

Figure 2. O-RAN Reference Architecture with XAInomaly

Enhanced Mobile Broadband (eMBB), ultra reliable low latency communications (URLLC), massive machine type communications (mMTC) advanced service categories have found their place in the literature with 5G. In 6G networks, advanced hybrid use cases such as URLLC-eMBB [11] and URLLC-mMTC are designed in which these services required together. In this case, Quality of Experience (QoE) should be provided in addition to the existing Quality of Service (QoS) metrics. Besides, more innovative technologies are required to achieve the QoE and KPIs required by next generation URLLC use case in 6G Networks [12].

However, when AI/ML algorithms are used in highly dynamic heterogeneous networks such as 6G towards O-RAN, decision-making processes of AI/ML models must be interpretable and reliable. To better interpret AI/ML algorithms that will work on performance demanding 6G networks, we aim to increase transparency by implementing the model-agnostic reactive XAI framework elucidate the Deep Autoencoder model of the Anomaly Detection xApp microservice running on O-RAN infrastructure as can be seen in Figure 3. To the best of our knowledge, we are the first to propose a reactive XAI framework on O-RAN RIC (RAN Intelligent Controller) considering 6G xURLLC use case real-time data processing and low latency requirements.

Our key contributions can be summarized as follows:

- We design, for the first time, a reactive XAI framework on O-RAN considering xURLLC use case.
- We introduce novel fastSHAP-C XAI method that obtain real-time SHAP values regarding O-RAN intelligence orchestration operations.
- To explain Deep Autoencoder model running in Anomaly Detection xApp, we proposed performance efficient fastSHAP-C for the real-time scenarios.
- We are the first to implement kernelSHAP, fastSHAP baseline models in O-RAN.

II. Related Work

The emergence of 6G networks brings forth unprecedented opportunities and challenges [11], necessitating the integration of advanced AI techniques to optimize performance, resource management, and user experience. In this context, XAI has garnered significant attention as a means to enhance transparency, interoperability, and trustworthiness in AI-driven decision-making processes within 6G ecosystems [7]. By synthesizing insights from diverse disciplinary perspectives, these works contribute to a comprehensive understanding of the opportunities and challenges associated with leveraging XAI in the evolution of 6G networks.

Continuing this exploration, recent research has emphasized the critical role of AI explainability in various facets of 6G network operations, including resource management [13]. In this study, the authors focused on the use of XAI through separate 6G use cases. Particularly, they discussed the deep reinforcement learning based solution used in the Vehicular network scenario. In this regard, they examined the contribution of features through SHAP values. However, one major challenge is the inherent complexity and high dimensionality of DRL models, which often consist of numerous layers and parameters, making them difficult to interpret. This complexity can lead to explanations that are either too simplified, missing critical nuances, or overly detailed, making them hard to understand for end-users. On the other hand, in a use case with vehicles, the XAI model is expected to perform real-time evaluation. Achieving this real-time evaluation capability with the conventional SHAP algorithm is quite challenging in terms of real life scenario. By addressing these multifaceted challenges and opportunities, researchers aim to establish a foundation for the development of transparent and accountable AI systems that underpin the next generation of wireless communication networks.

In [14], researchers introduce EXPLORA, a novel framework aimed at enhancing the explainability of AI/ML models in the context of Open RAN. This paper addresses the pressing need for transparency and interpretability in decentralized network architectures like Open RAN, where complex decision-making processes occur at the network edge. Fiandrino *et al.* [5], explore the integration of XAI and robustness in the context of 6G networks. Despite the progress made, the paper acknowledges that significant improvements are still needed, particularly in balancing the explainability-robustness trade-off and ensuring that explanations are both accurate and comprehensible to various stakeholders, from network operators to end-users. A use case examined on the 4G network with LRP and SHAP algorithms. And the execution time and resource utilization results of these algorithms have been shared. According to the authors' results, CPU usage constitutes a serious bottleneck, so the development of more innovative domain-aware XAI algorithms is inevitable. This is why it motivated us to research and develop XAI algorithms that can handle the 6G xURLLC real-time data communication.

Although XAI studies are found in different domains in the literature [15, 16], they are especially prevalent in 5G+/6G O-RAN research. Especially when we evaluate the deficiencies in the aforementioned studies, there is still room for improvement on the 6G towards O-RAN.

III. Explainability and Trustworthiness in 6G Open-RAN

XAI is the interdisciplinary field at the intersection of artificial intelligence and cognitive science, focused on developing techniques and methodologies to enable AI systems to provide transparent and interpretable explanations of their decision-making processes to humans, fostering trust, understanding, and collaboration between humans and machines. Model-agnostic methods in XAI are techniques that can be applied to a wide range of machine learning models without relying on knowledge of their internal structures. These methods aim to provide interpretable explanations for individual predictions or the overall behavior of a model, regardless of its complexity or type. One common type of model-agnostic method is the SHAP (SHapley Additive exPlanations) framework, which computes feature attributions for each prediction based on game-theoretic principles. Other model-agnostic methods include LIME (Local Interpretable Model-agnostic Explanations) and Anchors, which generate explanations at the local level by approximating the behavior of the underlying model in the vicinity of a specific data point.

A. *Local vs Global Model-Agnostic XAI Methods*

The difference between local and global model-agnostic methods lies in the scope of their explanations. Local model-agnostic methods, such as LIME [17], focus on explaining individual predictions by approximating the behavior of the model in the local neighborhood of a data point. These methods provide insights into why a particular prediction was made for a specific instance, making them useful for understanding model behavior at the instance level. In contrast, global model-agnostic methods, like SHAP, provide explanations that apply across the entire dataset or model [6]. They offer insights into the overall importance of features and how they contribute to predictions on average, providing a broader understanding of the model's behavior across different instances. While local methods offer fine-grained explanations for individual predictions, global methods provide a more comprehensive view of the model's decision-making process across the entire dataset. Both types of methods are valuable in different contexts and can be used together to gain a holistic understanding of a machine learning model's behavior.

B. *Reactive vs Post-hoc XAI*

Reactive-XAI refers to a subset of XAI techniques that provide real-time explanations and interventions in response to model predictions or user queries. Unlike traditional XAI methods that generate explanations post-hoc or offline, reactive XAI methods operate in real-time, offering immediate insights into the decision-making process of machine learning models as predictions are made. In contrast, post-hoc XAI involves generating explanations after the decision-making process has occurred. This is typically done by applying separate techniques to interpret and explain the decisions of already trained models, often using methods such as feature importance analysis, visualization, or surrogate models. Reactive XAI and post-hoc XAI are related but distinct concepts. While reactive XAI focuses on built-in transparency, post-hoc XAI addresses interpretability retrospectively, aiming to shed light on the workings of complex models that were not inherently designed to be interpretable.

Since the xURLLC use case requires high reactiveness and responsivity, it is more appropriate to move towards reactive-XAI design.

C. *XAI Metrics*

XAI metrics are quantitative measures used to evaluate the performance and effectiveness of explainability techniques applied to machine learning models. These metrics help assess the quality of explanations provided by XAI methods, as well as their impact on user understanding, trust, and decision-making. Metrics of Reactive-XAI models are as follows:

- *Confidence score* is particularly important when assessing the reliability and robustness of XAI techniques, as it directly influences user perception and acceptance of the explanations. It typically measures the consistency between the model's predictions and the explanations generated by an XAI method. One way to compute the CS is by evaluating the discrepancy between the model's predicted output and the output reconstructed using the explanations. Let $f(x)$ be the prediction of the model for input x and $\phi(x)$ be the vector of feature importances provided by the XAI method for input x. The explanation can be used to reconstruct the model's prediction as:

$$\hat{f}(x) = \sum_{i=1}^{d} \phi_i(x) \tag{1}$$

where d is the number of features. The Confidence Score can be defined as the average absolute difference between the model's actual prediction and the reconstructed prediction from the explanations:

$$\text{CS} = \frac{1}{n} \sum_{i=1}^{n} \left| f(x_i) - \sum_{j=1}^{d} \phi_j(x_i) \right| \tag{2}$$

where n is the number of samples, x_i is the i-th sample, and $\phi_j(x_i)$ is the explanation for the j-th feature of the i-th sample.
A high confidence level indicates that the explanations are reliable, accurate, and consistent with the underlying model's behavior, instilling trust in the XAI method and

the model's predictions. Conversely, a low confidence level suggests that the explanations may be uncertain or unreliable.

- *Sensitivity* refers to the degree to which the explanations provided by an XAI method are sensitive to changes in the input data or model parameters. It quantifies how the explanations vary in response to perturbations or variations in the input features, helping assess the robustness and reliability of the XAI technique. A highly sensitive XAI method exhibits significant changes in the explanations when small modifications are made to the input data or model parameters, indicating potential instability or uncertainty in the insights provided.

$$\lambda(x_i) = argmax_{x_j \in B_{\in}(x_i)} \frac{||\Phi_(x_i) - \Phi_(x_j)||_2}{||x_i - x_j||_2} \quad (3)$$

- *Log-odds* is used to assess the importance or relevance of each feature in influencing the model's predictions. It measures how much the log-odds of the predicted outcome change when a particular feature is present or absent, providing insights into the feature's impact on the model's decision-making process. A positive log-odds value indicates that the presence of the feature increases the likelihood of the positive outcome, while a negative log-odds value indicates that the presence of the feature decreases the likelihood of the positive outcome.

$$log - odds(p) = -\frac{1}{L}\sum_{i=1}^{L} log\frac{Pr(\hat{y}|x_i^{(p)})}{Pr(\hat{y}|x_i)} \quad (4)$$

D. SHapley Additive exPlanations (SHAP)

SHAP [18] is a method in Explainable AI (XAI) that aims to provide interpretable explanations for the predictions made by machine learning, deep learning models. It is based on cooperative game theory, specifically the concept of Shapley values, which originated from economics and political science. SHAP emerges as a powerful tool in the realm of complex networks due to its unique advantages. First and foremost, SHAP provides transparent and interpretable explanations for individual predictions, enabling users to comprehend the reasoning behind model decisions. Its model-agnostic nature allows for seamless integration with a wide range of machine learning algorithms, ensuring applicability across diverse domains and applications. Additionally, SHAP offers the flexibility to generate both local and global explanations, providing insights into model behavior at different levels of granularity.

The core idea behind SHAP is the concept of Shapley values, derived from cooperative game theory. Shapley values provide a way to fairly distribute the "payout" among players based on their contributions to the total payout. For a machine learning model, the "players" are the features, and the "payout" is the prediction of the model. The Shapley value for a feature represents its average contribution to the prediction across all possible subsets of features.

Mathematically, the Shapley value ϕ_i for a feature i is given by:

$$\phi_i = \sum_{S \subseteq N \setminus \{i\}} \frac{|S|! \cdot (|N| - |S| - 1)!}{|N|!} \left(v(S \cup \{i\}) - v(S)\right) \quad (5)$$

where N is the set of all features, S is a subset of N that does not include feature i, and $v(S)$ is the value function that gives the model prediction using only the features in subset S.

SHAP values adapt the Shapley values to the context of machine learning by defining a value function $v(S)$ that represents the model's prediction given the features in subset S. A common choice for $v(S)$ is:

$$v(S) = \mathbb{E}[f(x) \mid x_S] \quad (6)$$

where $f(x)$ is the model prediction, and x_S are the features in subset S. In practice, computing exact Shapley values is computationally expensive due to the combinatorial number of subsets. SHAP uses several approximations to make this feasible, such as KernelSHAP, TreeSHAP, and DeepSHAP, tailored to specific model types.

1) kernelSHAP: It is a method for interpreting the predictions of machine learning models based on Shapley values, a concept from cooperative game theory [18] . The goal is to fairly attribute the contribution of each feature to the prediction of a machine learning model. Given a set of players $N = \{1, 2, \ldots, n\}$, the Shapley value for player i is defined as

$$\phi_i(v) = \sum_{S \subseteq N \setminus \{i\}} \frac{|S|!(|N| - |S| - 1)!}{|N|!}[v(S \cup \{i\}) - v(S)] \quad (7)$$

Here, v is a characteristic function that assigns a real number to each subset of players, representing the value or payoff of that subset. S is any subset of N that does not include player i. Algorithm 1 basically completes the following steps:

- **Sample subsets**: Randomly sample subsets of features.
- **Compute characteristic function**: For each subset, compute $v(S)$.
- **Weight subsets**: Apply the kernel weighting to each subset.
- **Fit linear model**: Solve the weighted least squares problem to obtain the Shapley values.

2) fastSHAP: It is an efficient approximation method for computing Shapley values [19]. As can be seen in Algorithm 2, it aims to accelerate the computation of Shapley values, which can be computationally expensive for complex models and large datasets. fastSHAP leverages a differentiable surrogate model trained to approximate Shapley values, significantly reducing computation time.

IV. Proposed Design: XAInomaly Framework

In 5G+/6G networks are expected to be highly dynamic and heterogeneous. With these next generation networks, traditional anomaly detection methods may struggle to adapt to the

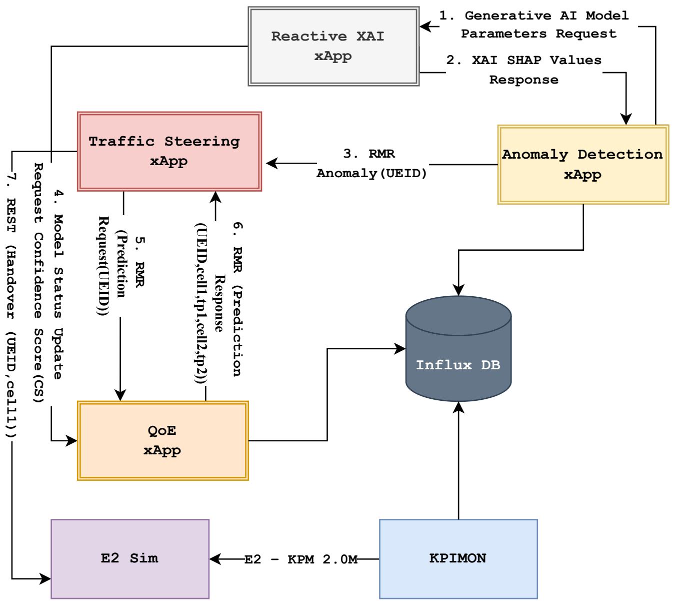
Reactive XAI xApp
1. Generative AI Model Parameters Request
2. XAI SHAP Values Response
Traffic Steering xApp
3. RMR Anomaly(UEID)
Anomaly Detection xApp
4. Model Status Update Request Confidence Score(CS)
5. RMR (Prediction Request(UEID))
6. RMR (Prediction Response (UEID,cell1,tp1,cell2,tp2))
7. REST (Handover (UEID,cell1))
Influx DB
QoE xApp
E2 Sim
E2 - KPM 2.0M
KPIMON

Figure 3. Integration of XAInomaly Framework to O-RAN

evolving network conditions and detect subtle anomalies in the vast amount of network data generated. Recently, it has come to light in 5G research that Deep autoencoders leverage deep learning techniques to learn hierarchical representations of the network data, allowing them to model intricate relationships and detect anomalies with high accuracy. However, even though deep autoencoders excel at capturing complex patterns and anomalies in high-dimensional data, they often operate as black-box models, providing limited insights into the underlying reasons for anomaly detection. So, using XAI algorithms provides transparent and interpretable explanations for individual anomalies, quantifying the contribution of each feature to the anomaly detection decision. In this way, the AI/ML Execution and Inference and Continuous Operations steps of the AI/ML Workflow [20] published by the O-RAN Alliance (WG2) are correctly constructed.

Figure 3 shows the workflow of the XAInomaly framework integrated top of the existing O-RAN anomaly detection use case. In addition to the existing anomaly detection use case flow, the performance of the deep autoencoder model running in AD xApp is checked via Reactive XAI xApp. The confidence level of the anomaly detection classification is determined thanks to the SHAP Values calculated by the model parameters coming from AD xApp. In the next step, the model status update information is shared with QoE xApp. After the Confidence Score (CS) is calculated through the metrics specified in Section III-C, CS is fed to QoE Predictor xApp. Thus, in an application that requires high real-time computing such as xURLLC, control loops can be operated while ensuring the reliability of the DL model.

```
Input: $X_{i..j}$ samples that kernelSHAP uses to calculate
  SHAP values, $ErrorList$ an ordered list for calculate
  error per feature, $m$ is autoencoder model
Output: $SHAPbestMfeatures$, calculated SHAP
  values for each feature in $bestMfeatures$
$bestMfeatures \leftarrow XbestfeaturesfromErrorList$
for $each\ \ i \in bestMfeatures$ do
    $explainer \leftarrow shap.KernelExplainer(m, x_{i..j})$
    $SHAPbestMfeatures[i] \leftarrow$
      $explainer.shapvalues(x_i)$
end
return $SHAPbestMfeatures$
```

Algorithm 1: kernelSHAP Training

```
Input: Value function $f_{x,y}$, learning rate $\alpha$
Output: $fastSHAP$ explainer $\phi_{fast}(x, y; \theta)$
initialize $\phi_{fast}(x, y; \theta)$
while $not\ \ converged$ do
    $sample\ \ x \sim p(x), y \sim Unif(y), s \sim p(s)$
    $predict\ \ \hat{\phi} \leftarrow \phi_{fast}(x, y; \theta)$
    if $normalize$ then
        $set\ \ \hat{\phi} \leftarrow \hat{\phi} + d^{-1}(f_{x,y}(1) - f_{x,y}(0) - 1^T\hat{\phi})$
    end
    $calculate$
    $\mathcal{L} \leftarrow (f_{x,y}(s) - f_{x,y}(0) - s^T\hat{\phi})^2$
    $update\ \ \theta \leftarrow \theta - \alpha\nabla_\theta\mathcal{L}$
end
```

Algorithm 2: fastSHAP Training

A. Anomaly Detection xApp: Dataset and Model Parameters

As Anomaly Detection (AD) xApp, we directly used existing model designed by Basaran et al. [21]. Model based on the Deep Autoencoder model. UE data streams coming through InfluxDB are classified with a semi-supervised learning perspective via Deep Autoencoder based xApp. To reveal which features carry more valuable information regarding Deep Autoencoder based anomaly detection xApp, we also implemented traditional SHAP algorithm as shown in Figure 4 and Table I.

The dataset consists of samples from different user equipment (UEs) such as cars, train passengers, pedestrians, and waiting passengers.

Deep Autoncoder model was trained based on RSRP, RSRQ, RSSINR, Physical Resource Block (PRB) usage, and throughput features on a dataset of 10000 samples. 25% of the dataset constitutes the anomaly class. In other words, it is understood that the model was trained with an unbalanced dataset. Deep Autoencoder model is designed with a 10 layer symmetrical encoder and decoder with shallow layers. After different hyperparameter tuning stages, a model with 31297 parameters (Trainable Params: 30,491 and Nontrainable Params: 806). Learning capacity of the deep autoencoder model was monitored by calculating the mean squared error (MSE) between the real input and the reconstructed input. Accordingly,

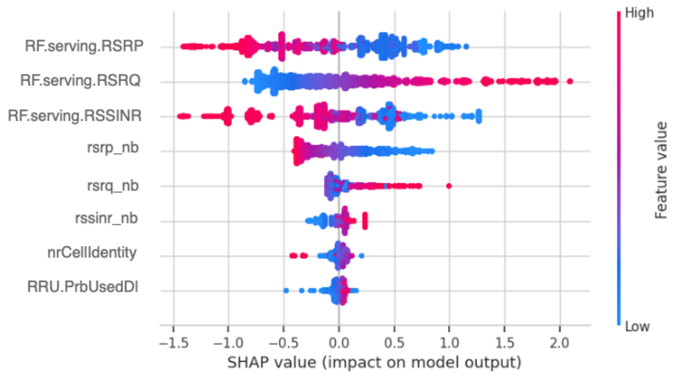

Figure 4. Explaining Feature Contributions with SHAP Values

Table I
EXPLAINABILITY WITH MODEL FEATURES

Features	
High Reconstruction Error	X_1
Explanatory	X_2, X_3
Explaining Anomaly	X_1, X_2, X_3

the anomaly classification accuracy of the model was obtained as 93%. Detailed model parameters in the training process of Deep Autoencoder based Anomaly Detection xApp are given in Table II.

B. Our Proposed fastSHAP-C Algorithm

KernelSHAP is one of the important SHAP implementations in the literature. It is also used on anomaly detection which works on autoencoder models [22]. Unlike traditional SHAP methods that rely on sampling or approximation techniques, KernelSHAP offers an exact solution by leveraging a kernel function to approximate the Shapley values as shown in Algorithm 1. KernelSHAP, while offering accurate and interpretable explanations for machine learning models, presents several challenges that must be considered. Especially considering the xURLLC scenario, kernelSHAP's computational complexity can be prohibitive, particularly for large datasets and high-dimensional feature spaces, leading to scalability issues and impracticality for real-time applications.

Therefore, the performance trade-off should be carefully investigated by using novel new designs. A more competent framework can be designed through the algorithm proposed by researchers for real-time Shapley value estimation [19]. Unlike traditional methods that require exponential time complexity to calculate Shapley values, fastSHAP leverages sampling techniques and optimization strategies to achieve significant speedups without sacrificing accuracy. The flow of fastSHAP is given in Algorithm 2. When the algorithm flow is examined, two important approaches stand out. First, fastSHAP does not need to optimize separately for each input to be explained. Second, it can obtain Shapley values from similar data points. This allows fastSHAP to keep its computational complexity at a low level compared to existing SHAP methods.For this reason, we first ran the fastSHAP algorithm on O-RAN anomaly detection xApp. However, this had to be able to truly give accurate feedback to other xApps within the O-RAN. For this, we added the $\mathcal{CS}$ implementation and proposed fastSHAP-C, shown in Algorithm 3. The addition of a Confidence Score (CS) and an Error Metric (EM) allows for assessing the reliability and accuracy of the explanations provided by fastSHAP which is critical for 6G towards O-RAN. The steps of fastSHAP-C algorithm are as follows:

1) *Input and Initialization:*
   - Model f
   - Data sample x
   - Background dataset X
   - Number of samples M

   Initialize:

$$\phi_i(x) = 0 \quad \forall i$$
$$f(x) = f(x)$$
$$\hat{f}(x) = 0$$

2) *Sampling:* For each sample m from 1 to M: Sample a subset $S \subseteq \{1, 2, \ldots, d\}$ uniformly at random.
3) *Marginal Contribution:* For each feature i: Compute the marginal contribution of feature i given the subset S:

$$\Delta f_i(S) = f(S \cup \{i\}) - f(S) \tag{8}$$

   Update the Shapley value:

$$\phi_i(x) = \phi_i(x) + \frac{\Delta f_i(S)}{M} \tag{9}$$

4) *Calculate Confidence Score (CS)* Compute the reconstructed prediction:

$$\hat{f}(x) = \sum_{i=1}^{d} \phi_i(x) \tag{10}$$

   Calculate the Confidence Score:

$$CS = \frac{1}{n} \sum_{k=1}^{n} \left| f(x_k) - \hat{f}(x_k) \right| \tag{11}$$

5) *Calculate Error Metric (EM):* Compute the error between the actual and reconstructed predictions:

$$EM = \frac{1}{n} \sum_{k=1}^{n} \left(f(x_k) - \hat{f}(x_k) \right)^2 \tag{12}$$

6) *Output:* Return the Shapley values $\phi_i(x)$, Confidence Score CS, and Error Metric EM.

V. EVALUATION AND RESULTS

In this section, we will first interpret the SHAP values to understand the Deep Autoencoder model that works as anomaly detection xApp. Then, we will benchmark the performance results of our novel fastSHAP-C implementation in XAInomaly framework, against kernelSHAP and fastSHAP used by researchers in the literature.

Table II
DEEP AUTOENCODER MODEL SUMMARY WITH PARAMETERS

Autoencoder Layers	Output Shape	Parameters
input_1 (InputLayer)	(None, 20)	0
dense (Dense)	(None, 100)	2100
batch_normalization (BatchNorm)	(None, 100)	400
dense_1 (Dense)	(None, 75)	7575
batch_normalization_1 (BatchNorm)	(None, 75)	300
dense_2 (Dense)	(None, 50)	3800
batch_normalization_2 (BatchNorm)	(None, 50)	200
dense_3 (Dense)	(None, 25)	1275
batch_normalization_3 (BatchNorm)	(None, 25)	100
dense_4 (Dense)	(None, 3)	78
dense_5 (Dense)	(None, 3)	12
batch_normalization_4 (BatchNorm)	(None, 3)	12
dense_6 (Dense)	(None, 25)	100
batch_normalization_5 (BatchNorm)	(None, 25)	100
dense_7 (Dense)	(None, 50)	1300
batch_normalization_6 (BatchNorm)	(None, 25)	100
dense_8 (Dense)	(None, 75)	3825
batch_normalization_7 (BatchNorm)	(None, 75)	300
dense_9 (Dense)	(None, 100)	7600
dense_10 (Dense)	(None, 20)	2020
Total Params: 31,297		
Trainable Params: 30,491		
Nontrainable Params: 806		

```
Input: Value function $f_{x,y}$, learning rate $\alpha$
Output: $fastSHAP$ explainer $\phi_{fast}(x, y; \theta)$
initialize $\phi_{fast}(x, y; \theta)$
while $not\ converged$ do
    $sample\ \ x \sim p(x), y \sim Unif(y), s \sim p(s)$
    $predict\ \ \hat{\phi} \leftarrow \phi_{fast}(x, y; \theta)$
    if $normalize$ then
        $set\ \ \hat{\phi} \leftarrow \hat{\phi} + d^{-1}(f_{x,y}(1) - f_{x,y}(0) - 1^T \hat{\phi})$
    end
    $calculate$
    $\mathcal{L} \leftarrow (f_{x,y}(s) - f_{x,y}(0) - s^T \hat{\phi})^2$
    $update\ \ \theta \leftarrow \theta - \alpha \nabla_\theta \mathcal{L}$
end
$\mathcal{CS} \leftarrow \frac{1}{n} \sum_{i=1}^{n} \left| f_{x,y}(s_i) - f_{x,y}(0) - s_i^T \hat{\phi} \right|$
$\mathcal{EM} \leftarrow \frac{1}{n} \sum_{i=1}^{n} \left( f_{x,y}(s_i) - s_i^T \hat{\phi} \right)^2$
return $\phi_{\text{fast}}(x, y; \theta)$, $\mathcal{CS}$, $\mathcal{EM}$
```

Algorithm 3: fastSHAP-C Training

A. SHAP Values and Feature Contributions

The problem of calculating SHAP values is generally NP-hard, which may cause the high convergence time and the solution to be complex to reach. We implemented kernelSHAP, which is also our baseline model, to calculate SHAP values and get insight about feature contributions.

When the results of Figure 4 and Table I are examined, the most informative feature that can be used to explain the anomaly occurring in the UE is seen as RF.serving.RSRPV(X_1). RSRP is a metric used to quantify the strength of the radio signal received by a device from the serving cell's base station (eNodeB in LTE or gNodeB in 5G). RSRP is a crucial parameter that reflects the quality of the radio link between the user equipment (UE) and the base station. It measures the power level of the downlink reference signal (RS) from the serving cell, providing an indication of the signal strength experienced by the UE. Low RSRP values indicate weak signal strength, leading to dropped connections, packet loss, and degraded performance, while erratic fluctuations can disrupt handover processes and impact QoS metrics such as latency and jitter.

RF.serving.RSRQ and RF.serving.RSSINR (X_2 and X_3 respectively), which are classified as explanotary features by the XAI algorithm, also contribute to the formation of anomalies in a way that is dependent on the RF.serving.RSRP feature. Variations in RSRQ and RSSINR values can indicate changes in signal quality, interference levels, and network conditions experienced by the UE. Anomalies such as low RSRQ values may lead to degraded signal quality, increased interference, and connectivity issues, while fluctuations in RSSINR may impact data throughput, latency, and overall network performance.

B. fastSHAP-C Advantages

The performance trade-off between fastSHAP-C and other customized SHAP algorithms kernelSHAP, fastSHAP lies in their computational efficiency and accuracy. When Table III is examined, exclusion and inclusion metrics are calculated based on log-odds. Exclusion and inclusion AUC metrics provide nuanced insights into the performance of XAI framework by evaluating their discriminatory power across specific subsets of data. Considering metrics, it is clearly seen that fastSHAP-C outperformed existing kernelSHAP and fastSHAP models. Moreover, the results in Table IV have very novel results in terms of the reliability of 6G xURLLC scenario. When the runtime results are examined, it is seen that fastSHAP-C reduces the computational burden of Shapley value estimation, making it practical for large-scale datasets and complex ML/DL models while considering $\mathcal{CS}$ and $\mathcal{EM}$ metrics. Its computational efficiency allows for on-the-fly explanation generation, facilitating dynamic adaptation to changing network conditions and user requirements.

fastSHAP-C's accuracy, albeit slightly compromised compared to conventional methods, remains sufficient for providing actionable insights into model behavior, aiding in network optimization, fault diagnosis, and resource allocation. Under the same resources results given in Table IV show that, under the same resources (Intel(R) Core(TM) i7-6800K CPU @ 3.40GHz (12 cores)), the fastSHAP-C algorithm provides approximately 30% runtime advantage. On the other hand, by reducing CPU usage by 25%, it brings performance-frindly explainability to deep learning algorithms that require real-time processing.

C. Benchmarking with Recent Studies

We compared our results with literature studies in two different ways. First, we took as a basis the lowest and highest performance servers we had available and used by the authors

Table III
EXCLUSION AND INCLUSION AUCS $(log - odds(p))$

Algorithm	Exclusion AUC	Inclusion AUC
kernelSHAP	10.53 (10.17, 11.03)	4.96 (4.56, 5.47)
fastSHAP	6.89 (6.73,7.65)	5.79 (5.58,5.93)
fastSHAP-C	6.6 (5.73,4.65)	6.79 (5.53,5.96)

Table IV
RUNTIME AND RESOURCE UTILIZATION

Algorithm	Runtime (ms)	CPU Util. (%)	RAM Util. (%)
kernelSHAP	320400	0.88	0.17
fastSHAP	48007	0.67	0.11
fastSHAP-C	33627	0.52	0.8

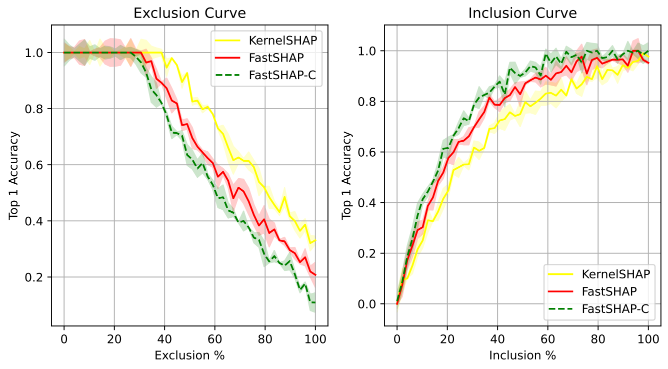

Figure 5. Exclusion and Inclusion Curves for *top-1* Accuracy

[5]. Server 1; Intel(R) Core(TM) i7-6800K CPU @ 3.40GHz (12 cores), equipped with 64 GB of RAM, while Server 2; Intel(R) Xeon(R) Gold 6240R CPU @ 2.40GHz (97 cores) was. While the first two rows show the Time, CPU and Memory results that the authors obtained using SHAP, the last two lines show the results of the fastSHAP-C algorithm that we ran on servers with the same resources as we mentioned. When the performance results in the last two rows of Table V are examined, the proposed fastSHAP-C algorithm is significantly faster in execution time compared to the standard SHAP algorithm on both standard and high-end servers.fastSHAP-C algorithm demonstrates significantly improved execution speed compared to the standard SHAP algorithm. This is evident in both short-term (1-hour) and long-term (6-hour) profiling. For instance, on Server 2, the execution time for fastSHAP-C is reduced to approximately 3.90 seconds for 1 hour and 10.05 seconds for 6 hours, compared to 10.52 seconds and 58.28 seconds for SHAP. This improvement is crucial for real-time applications and scenarios where quick turnaround is essential. fastSHAP-C uses the CPU more efficiently, leading to lower mean and standard deviation in CPU usage. For example, on Server 2 (fastSHAP-C), CPU usage is 3.44% (Mean) for 1 hour, compared to 7.66 % (Mean) for SHAP. This reduction in CPU usage allows for more concurrent tasks and less strain on the processing resources, making it suitable for environments with high computational demand. The performance gains with fastSHAP-C are more pronounced on high-end servers with more cores. This presents that fastSHAP-C is optimized to take full advantage of multi-core architectures, making it highly scalable for large datasets and complex models that require parallel processing.

We concluded second performance comparison with the algorithms we reviewed in the Section II and also the most used ones in recent studies. We calculated accuracy for kernelSHAP and fastSHAP. When Figure 5 is examined, it is seen that the exclusion curve on the left graph shows how the top-1 accuracy of the model decreases as an increasing percentage of the most important features (as identified by each method) are excluded from the model's input. kernelSHAP (yellow line) maintains high accuracy when a small percentage of features are excluded but shows a steady decline as more features are removed. fastSHAP (red line) shows a similar trend but with a slightly faster decline in accuracy compared to kernelSHAP, indicating that it might be less robust when critical features are excluded. fastSHAP-C (green dashed line) starts off with a slightly lower accuracy than kernelSHAP and fastSHAP but shows a comparable decline. However, fastSHAP-C performs better than fastSHAP in the mid-range of feature exclusion (around 20-70%). The inclusion curve on the right graph illustrates how the top-1 accuracy improves as an increasing percentage of the most important features are included in the model's input. kernelSHAP shows a steady increase in accuracy as more features are included, but it lags slightly behind fastSHAP and fastSHAP-C in the initial phase (0-20% inclusion). fastSHAP improves accuracy more rapidly than kernelSHAP in the early stages of feature inclusion, indicating a better initial selection of important features. fastSHAP-C (green dashed line) consistently outperforms both kernelSHAP and fastSHAP across the entire range of feature inclusion, achieving the highest accuracy as the inclusion percentage increases. kernelSHAP maintains higher accuracy under exclusion but starts slower in inclusion. fastSHAP shows faster initial improvement in inclusion but drops quicker in exclusion.

VI. CONCLUSION

5G+/6G and upcoming next-generation wireless networks prioritize user-centric QoE, especially in critical applications such as xURLLC use cases. In these scenarios, where the network must meet stringent requirements for reliability, real-time processing, latency and understanding the factors influencing QoE is paramount. Taking these requirements into consideration; in this study, we proposed novel XAI method called fastSHAP-C. fastSHAP-C method integrated into O-RAN in a domain-aware way and demonstrates the explainability of the deep autoencoder model used in anomaly detection applications. Besides, it provides significant reduction in computation time and resource usage which makes

Table V
RESOURCE UTILIZATION COMPARISON OF ALGORITHMS ON DIFFERENT SERVERS
(SHAP *vs* PROPOSED *fast*SHAP-C)

Server	**Time**		**CPU**				**Memory**			
	1 h	6 h	1 h		6 h		1 h		6 h	
	(s)	(s)	Mean	Std	Mean	Std	Mean	Std	Mean	Std
Server 1 (SHAP)	11.74	62.00	29.8%	8.3%	36.8%	16.7%	1.5%	0.1%	2.6%	0.2%
Server 2 (SHAP)	10.52	58.28	7.66%	3.82%	12.3%	3.8%	1.6%	0.2%	2.4%	0.1%
Server 1 (*fast*SHAP-C)	6.56	15.87	13.5%	11.3%	25.3%	4.7%	1.4%	0.2%	2.3%	0.1%
Server 2 (*fast*SHAP-C)	3.90	10.05	3.44%	2.51%	7.8%	2.5%	1.3%	0.1%	2.5%	0.1%

fastSHAP-C particularly suitable for real-time applications where decisions need to be made quickly. The results show that using the fastSHAP-C algorithm provides superiority compared the state-of-the-art XAI methods used in the literature where demanding 6G use cases are considered.

ACKNOWLEDGMENT

This work has been funded by the German Federal Ministry of Education and Research (BMBF, Germany) as part of the 6G Research and Innovation Cluster 6G-RIC under Grant 16KISK020K.

REFERENCES

[1] L. Bariah, Q. Zhao, H. Zou, Y. Tian, F. Bader, and M. Debbah, "Large Generative AI Models for Telecom: The Next Big Thing?" *IEEE Communications Magazine*, Jan. 2024, to appear.

[2] K. Li, Y. Cui, W. Li, T. Lv, X. Yuan, S. Li, W. Ni, M. Simsek, and F. Dressler, "When Internet of Things meets Metaverse: Convergence of Physical and Cyber Worlds," *IEEE Internet of Things Journal*, vol. 10, no. 5, pp. 4148–4173, Mar. 2023.

[3] S. Garg, K. Kaur, G. S. Aujla, G. Kaddoum, P. Garigipati, and M. Guizani, "Trusted Explainable AI for 6G-Enabled Edge Cloud Ecosystem," *IEEE Wireless Communications*, vol. 30, no. 3, pp. 163–170, 2023.

[4] F. Dressler, C. F. Chiasserini, F. H. P. Fitzek, H. Karl, R. Lo Cigno, A. Capone, C. E. Casetti, F. Malandrino, V. Mancuso, F. Klingler, and G. A. Rizzo, "V-Edge: Virtual Edge Computing as an Enabler for Novel Microservices and Cooperative Computing," *IEEE Network*, vol. 36, no. 3, pp. 24–31, May 2022.

[5] C. Fiandrino, G. Attanasio, M. Fiore, and J. Widmer, "Toward native explainable and robust AI in 6G networks: Current state, challenges and road ahead," *Elsevier Computer Communications*, vol. 193, pp. 47–52, Sep. 2022.

[6] S. Ali, T. Abuhmed, S. El-Sappagh, K. Muhammad, J. M. Alonso-Moral, R. Confalonieri, R. Guidotti, J. Del Ser, N. Díaz-Rodríguez, and F. Herrera, "Explainable Artificial Intelligence (XAI): What we know and what is left to attain Trustworthy Artificial Intelligence," *Elsevier Information Fusion*, vol. 99, p. 101 805, Nov. 2023.

[7] W. Guo, "Explainable Artificial Intelligence for 6G: Improving Trust between Human and Machine," *IEEE Communications Magazine*, vol. 58, no. 6, pp. 39–45, Jun. 2020.

[8] M. Polese, M. Dohler, F. Dressler, M. Erol-Kantarci, R. Jana, R. Knopp, and T. Melodia, "Empowering the 6G Cellular Architecture with Open RAN," *IEEE Journal on Selected Areas in Communications*, vol. 42, no. 2, pp. 245–262, Feb. 2024.

[9] 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, cs.NI 2307.00319, Jul. 2023.

[10] L. Bonati, S. D'Oro, M. Polese, S. Basagni, and T. Melodia, "Intelligence and Learning in O-RAN for Data-Driven NextG Cellular Networks," *IEEE Communications Magazine*, vol. 59, no. 10, pp. 21–27, Oct. 2021.

[11] W. Saad, M. Bennis, and M. Chen, "A Vision of 6G Wireless Systems: Applications, Trends, Technologies, and Open Research Problems," *IEEE Network*, vol. 34, no. 3, pp. 134–142, May 2020.

[12] C. She, C. Pan, T. Q. Duong, T. Q. S. Quek, R. Schober, M. Simsek, and P. Zhu, "Guest Editorial xURLLC in 6G: Next Generation Ultra-Reliable and Low-Latency Communications," *IEEE Journal on Selected Areas in Communications*, vol. 41, no. 7, pp. 1963–1968, Jul. 2023.

[13] N. Khan, S. Coleri, A. Abdallah, A. Celik, and A. M. Eltawil, "Explainable and Robust Artificial Intelligence for Trustworthy Resource Management in 6G Networks," *IEEE Communications Magazine*, vol. 62, no. 4, pp. 50–56, Apr. 2024.

[14] C. Fiandrino, L. Bonati, S. D'Oro, M. Polese, T. Melodia, and J. Widmer, "EXPLORA: AI/ML EXPLainability for the Open RAN," *Proceedings of the ACM on Networking*, vol. 1, pp. 1–26, Nov. 2023.

[15] A. K. Gizzini, Y. Medjahdi, A. J. Ghandour, and L. Clavier, "Towards Explainable AI for Channel Estimation in Wireless Communications," *IEEE Transactions on Vehicular Technology*, vol. 73, no. 5, pp. 7389–7394, May 2024.

[16] M. Zolanvari, Z. Yang, K. Khan, R. Jain, and N. Meskin, "TRUST XAI: Model-Agnostic Explanations for AI With a Case Study on IIoT Security," *IEEE Internet of Things Journal*, vol. 10, no. 4, pp. 2967–2978, Feb. 2023.

[17] M. T. Ribeiro, S. Singh, and C. Guestrin, ""Why Should I Trust You?": Explaining the Predictions of Any Classifier," in *22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, San Francisco, CA: ACM, Aug. 2016.

[18] S. M. Lundberg and S.-I. Lee, "A Unified Approach to Interpreting Model Predictions," in *31st International Conference on Neural Information Processing Systems (NIPS 2017)*, Long Beach, CA: Curran Associates Inc., Dec. 2017, pp. 4768–4777.

[19] N. Jethani, M. Sudarshan, I. C. Covert, S.-I. Lee, and R. Ranganath, "FastSHAP: Real-Time Shapley Value Estimation," in *10th International Conference on Learning Representations (ICLR 2022)*, vol. V1, Virtual Conference: ICLR, Apr. 2022.

[20] "O-RAN AI/ML Workflow Description and Requirements 1.03," O-RAN Alliance, Technical Specification (TS) O-RAN.WG2.AIML-v01.03, Oct. 2021.

[21] O. T. Basaran, M. Basaran, D. Turan, H. G. Bayrak, and Y. S. Sandal, "Deep Autoencoder Design for RF Anomaly Detection in 5G O-RAN Near-RT RIC via xApps," in *IEEE International Conference on Communications (ICC 2023), 2nd Workshop on Industrial Private 5G-and-beyond Wireless Networks*, Rome, Italy: IEEE, May 2023, pp. 549–555.

[22] L. Antwarg, R. M. Miller, B. Shapira, and L. Rokach, "Explaining Anomalies Detected by Autoencoders using Shapley Additive Explanations," *Elsevier Expert Systems with Applications*, vol. 186, p. 115 736, Dec. 2021.