Leveraging LLMs to eXplain DRL Decisions for Transparent 6G Network Slicing

Mazene Ameur*, Bouziane Brik†, *Senior Member, IEEE*, and Adlen Ksentini *, *Senior Member, IEEE*
*EURECOM, Sophia-Antipolis, France, firstname.lastname@eurecom.fr
†Computer Science Department, College of Computing and Informatics,
Sharjah University, Sharjah, UAE, bbrik@sharjah.ac.ae

***Abstract*—The emergence of 6G networks heralds a transformative era in network slicing, facilitating tailored service delivery and optimal resource utilization. Despite its promise, network slice optimization heavily relies on Deep Reinforcement Learning (DRL) models, often criticized for their black-box decision-making processes. This paper introduces a novel Composable eXplainable Reinforcement Learning (XRL) framework customized for distributed systems like 6G Network Slicing. The proposed framework leverages Large Language Models (LLMs) and Prompt Engineering techniques to elucidate DRL algorithms' decision-making mechanisms, with a specific emphasis on user profiles. The latter transforms the inherently opaque nature of DRL into an interpretable textual format accessible not only to eXplainable AI (XAI) experts but also to diverse network slice provider stakeholders, engineers, leaders, and beyond. Experimental results underscore the efficacy of the proposed Composable XRL framework, showcasing substantial improvements in transparency and comprehensibility of DRL decisions within the context of 6G network slicing.**

***Index Terms*—Explainable Reinforcement Learning, Composable XRL, LLMs, Admission Control, 6G Network Slicing.**

I. Introduction

The emergence of 6G networks marks a significant milestone in the ongoing evolution of telecommunications technology. As we venture further into the digital age, the demand for faster, more reliable, and versatile communication networks continues to grow exponentially. 6G networks are poised to answer this call with their promise of unprecedented speeds, ultra-low latency, and the ability to support a multitude of emerging applications, such as augmented reality, virtual reality, and the Internet of Things. However, this incredible leap in connectivity comes with its own set of challenges, with one of the most critical being network slicing. Network slicing in 5G/6G is the process of dividing a single physical network into multiple virtual networks to cater to the diverse needs of various applications and services. Each network slice must be optimized for specific requirements, such as bandwidth, latency, and security. This constant pursuit of automation solutions has sparked extensive research into the applications of Artificial Intelligence (AI) and Machine learning (ML) in the context of 5G/6G [1].

With a growing interest in this topic, the regulations set forth by national and international authorities are continually evolving. For instance, Article 13 of the EU Regulatory Framework for AI[1] emphasizes the importance of AI systems being *explainable*, *transparent*, and *well-documented.* In scenarios where humans are directly involved, understanding the inner workings of complex models is essential for experts to conduct thorough root-cause analysis [2]. This principle extends to the majority of zero-touch network configuration and automation scenarios, currently under discussion within the ETSI ZSM[2] (Zero-touch Network and Service Management) group. Because they involve the use of AI and ML techniques in automating network and service management tasks. In such cases, ensuring the explainability of AI models used within the ZSM framework could be important to understand why certain automated decisions are made [1].

Current research in the application of Reinforcement Learning (RL) to network-related tasks (e.g. Network slicing, Load Balancing, etc.) relies on the use of Deep RL (DRL) algorithms [3]. These algorithms express their decision-making strategies through deep neural networks. DRL can handle a broad range of input types, not limited to finite or discrete sets, and these neural networks can effectively adapt to novel inputs. Furthermore, DRL can naturally adapt to changes in the system's environment, avoiding the need for explicit mechanisms to monitor such changes [4]. Nevertheless, a notable drawback of DRL is that the decision-making process is not explicitly revealed, it is instead concealed within the neural network's parameters. To service developers and users, DRL's decision-making may seem like a mysterious "black box". Consequently, there is a demand for methods that can elucidate and interpret the inner workings of these opaque systems, and shed light on how they reach their decisions [6].

Providing insights into the decision-making process of DRL can assist service developers in troubleshooting the reward function. This understanding helps them uncover the rationale behind DRL's specific choices. The effectiveness of DRL relies heavily on the quality of the problem definition, particularly in terms of how the reward function is specified. Additionally, the ability to offer explanations can play a crucial role in ensuring compliance with regulations. For instance, in the European Union, service providers are obligated to ensure that their services adhere to relevant legal frameworks, such as the

[1]https://digital-strategy.ec.europa.eu/en/library/proposal-regulation-laying-down-harmonised-rules-artificial-intelligence

[2]https://www.etsi.org/technologies/zero-touch-network-service-management

General Data Protection Regulation and the upcoming AI Act. Explanations also enable service users to establish trust. They can grasp the reasoning behind the service's outcomes and, consequently, decide whether to accept or reject those results [7].

We can categorize explanation formats into two major types [7], [8]: **(i) Visual explanations**, which encompass elements like graphical user interfaces, charts, data visualization, or heatmaps, and **(ii) Textual explanations**, which might involve a natural-language dialogue between the explainer and the person receiving the explanation. The choice of presentation method directly impacts how well users comprehend the information and, therefore, influences the overall effectiveness of the explanation. In comparison to visual explanations, verbal explanations offer several advantages as reported in the literature [3], [8], including (1) enhanced comprehensibility for individuals with diverse backgrounds and non-technical users, (2) increased user acceptance and trust, and finally, (3) more efficient explanations.

A. Motivation

Numerous initial investigations have been conducted to develop eXplainable RL (XRL) models and have made notable progress in generating explanations. Nevertheless, the challenge of elucidating the decision-making process of DRL using natural language remains unaddressed effectively. In the broader field of XAI, methods for furnishing natural-language explanations for ML are available [9], [10]. However, it is essential to note that these XAI approaches are tailored to supervised learning and not specifically designed for reinforcement learning except the work of Metzger et al. in [12], where they have introduced a framework designed to enhance the comprehension of DRL decision-making within general service-oriented systems. Despite the efforts showcased in the previous work, there is currently a notable gap in the literature as there are no existing studies that specifically address networking services in a general context and, more specifically, the intricacies of 5G/6G slicing. Moreover, XAI methods like SHapley Additive exPlanations (SHAP) and Local Interpretable Model-Agnostic (LIME) often yield outputs that require a high level of technical expertise for interpretation. Unfortunately, this expertise is not always readily available among various stakeholders within companies. In many instances, human intervention is necessary to elucidate decisions made by DRL models, particularly in strategic decisions. This is where Large Language Models (LLMs) prove invaluable. LLMs can offer personalized explanations to all company stakeholders, including leaders and experts, in the form of intelligent chatbots. These chatbots take into account the diverse backgrounds and profiles of individuals, ensuring that explanations are tailored to the specific needs and understanding levels of each user. The end goal is to augment transparency and streamline automation, while minimizing the need for constant human intervention in these systems.

In a broader sense, a significant research gap emerges in the domain of networking when considering the provision of natural language processing-based explanations. This gap becomes particularly pronounced when we recognize the necessity for **user-friendly explanations tailored to individual user profiles**. To instill trust in DRL decisions, especially those involving strategic choices that demand critical thinking, it is imperative to bridge this gap. The networking field, with its intricate technical intricacies and high-stakes decisions, calls for the development of interpretable and personalized explanations that can empower users to comprehend and trust the decisions made by DRL systems. Therefore, *"the ultimate goal of this paper is to motivate the use of natural language explanations of DRL black-box models in the 5G/6G networking realm"*.

B. Novelty

In contrast to the approach taken in [12], our framework uses a different explainable RL method than XRL-DINE to generate pseudo-explanations that will serve as input data to the prompt engineering and LLM modules as detailed in Section III-A3. Furthermore, the authors used in their framework a "Question Analyzer" module that interacts with the LLM to get information about the explanation type (either local or global through the timestep), which sometimes can be misleading to LLM if the user input is not clear enough. Conversely in our approach, we use different analogy that interact with the end user to get the accurate explanation type through the query templates bank.

To the best of our knowledge, this research represents the first effort to pave the road for a powerful approach named *"Composable XRL"* that tries to shed light on producing a human-friendly textual explanation of DRL decisions in the context of 5G/6G network slicing through leveraging three distinct modules including LLMs, Prompt engineering, and XRL. These explanations are designed in the format of a chatbot that considers the user profile regardless of his type (AI expert, Network Slice engineer, shareholder, etc.). The main goal is to enhance the transparency and trustworthiness of such DRL systems.

C. Key Contribution

The key contributions of this paper can be succinctly summarized as follows:

- We collect a dataset during the runtime of a DRL "Black-Box" model for network slice admission control.
- We adopt a novel XRL technique that employs reward decomposition to identify the features influencing the agent's decision-making process.
- We propose a new approach that harnesses the capabilities of XRL, LLMs, and Prompt Engineering to provide personalized user-aware explanations in the form of a Chatbot that takes into consideration different backgrounds and profiles of company actors (technical/non-technical AI/XAI experts).
- Through a proof-of-concept, we test the effectiveness of the proposed framework to explain DRL decisions

concerning the admission control problem of network slices in 5G/6G networks.

The remainder of the paper is organized as follows: Section II introduces the black-box DRL solution for the admission control problem. Section III provides an overview of our Composable XRL framework design. Section IV, with a dual focus, first details the experiment setup, and subsequently, presents and discusses the obtained results. Finally, Section V concludes the paper.

II. Explainable Deep Reinforcement Learning for Network Slicing

In the context of 6G, Network Slice Providers (NSPs) are tasked with handling network slice Requests (NSLRs) for NSL instances designed to support various use cases, including the next-generation Enhanced Mobile Broadband (eMBB+), Massive Machine Type Communication (mMTC+), and Ultra-Reliable Low Latency Communication (URLLC+) [5]. Each instance comes with its unique Quality of Service (QoS) requirements. To efficiently manage these requests, NSPs must establish an Admission Control mechanism to determine whether to accept an NSLR, factoring in the ability to meet QoS demands and the availability of physical resources, as illustrated in Fig. 1. This adaptive and self-optimizing approach ensures that network slices meet their service-level agreements while reducing the need for constant human intervention, which contributes to the efforts aimed at achieving the ZSM automation vision.

In response to the challenge of understanding DRL decisions, some efforts have been made to provide explanations for these intricate algorithms [6]. However, a significant issue persists: the existing solutions are often far from user-friendly. While they may offer some insight into the decision-making process, they lack consideration for the end user's profile. This oversight results in explanations that are not tailored to individual needs and preferences, rendering the systems less transparent and less effective in building trust.

The overarching goal of these developments is to enhance transparency and trustworthiness among users across the spectrum, not just limiting it to AI experts. When users can interact with and comprehend AI decisions, it fosters a sense of trust and empowers them to make informed choices. Such explainable systems have the potential to revolutionize the way individuals from various backgrounds engage with and benefit from AI-driven technologies, contributing to a more inclusive and transparent AI ecosystem.

A. Admission Control Problem

In this section, we expound on the formulation and modeling of the control admission problem from [11] as a case study, in order to analyze and validate our framework.

1) System Model: The 5G core network is modeled as an undirected graph with labels and weights, denoted as $SN = \{N, L\}$, where N represents the set of nodes $N = \{n_1, n_2 ... n_m\}$, and L represents the set of links

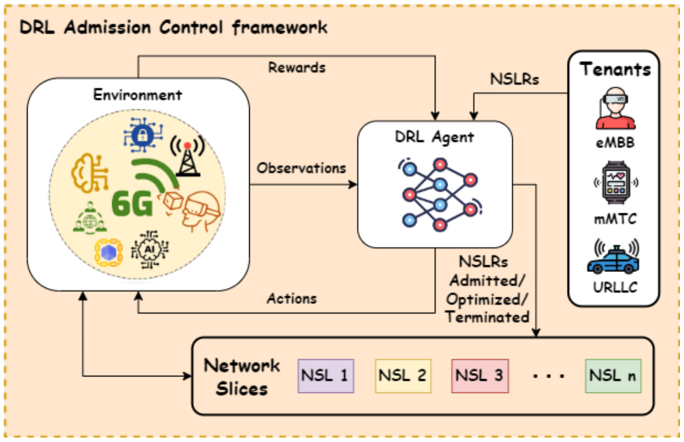


Fig. 1: DRL framework for Admission Control in 5G/6G Network Slicing

$L = \{(n_1, n_2), (n_1, n_3) ... (n_l, n_m)\}$. Each node, $n_i \in N$ is characterized by a processing capacity denoted as $CPU(n_i)$. The bandwidth of a link between nodes n_i and n_j is expressed as $BW(n_i, N_J)$.

A NSL Request is defined by $nslr = \{s_{type}, T_o, G\}$. The s_{type} identifies the 5G use case, such as eMBB, URLLC, or mMTC. T_o specifies the requested operational time, indicating the duration of a network slice. $G = \{F, V\}$ forms a labeled and weighted undirected graph representing an NSL (Network Slice). Here, F is the set of Virtual Network Functions (VNFs), and V is the set of virtual links connecting them. Nodes in this graph are labeled to signify the amount and type of resources demanded by a VNF, with edges weighted to represent the bandwidth requested by the virtual link. The processing capacity required by a VNF is denoted as $cpu(vnf_i)$, and the node type a VNF requests is represented by $type(vnf_i)$. Similarly, $bw(vnf_i, vnf_j)$ designates the bandwidth demanded by the virtual link connecting VNFs vnf_i and vnf_j.

2) DRL-based Solution: The previously mentioned optimization challenge can be addressed by employing the DRL framework, wherein the configuration of state and action spaces, along with the reward, is outlined below (see Table I):

- **State Space:** Characterizes resources in the 5G core network, In the context provided, $cpu(E)$ signifies the processing capacity available within the collection of edge nodes (E), while $cpu(C)$ denotes the available processing capacity within the core node set (C). Additionally, $bw(L)$ represents the available bandwidth within the group of links (L).
- **Action Space:** DRL agent selects weights $(W_{s_{type}})$ for different use cases to maximize profit.
- **Reward:** The reward value represents the monetary profit achieved by the DRL agent's action on a given state. The DRL agent aims to maximize the NSP profit while optimizing resource utilization. The final reward is the sum of profits $p(nsli)$ obtained for each accepted NSL,

normalized by the maximum profit i.e. $\max P(SN,T)$ achievable when utilizing all resources in the substrate within a specified period.

TABLE I: DRL Framework Parameters

Parameter	Type
State Space	$S = \{(cpu(E), cpu(C), bw(L)\}$
Action Space	$A = \{W_{URLLC}, W_{eMBB}, W_{mMTC}\}$
Reward	$p(nsl_i) = (rev_i - cst_i) \times T_o$ $csti = \sum_{j=0}^{m} cpu(vnf_j) \times fcpu_j + \sum_{j=0}^{n} bw(v_j) \times fbw \times h$ $R = \frac{\sum_{i=0}^{k} p(nsli)}{\max P(SN,T)}$

- rev: The income generated by the NSP for provisioning the $nsli$.
- cst : The operating expenses incurred by the NSP for running the $nsli$ on the underlying infrastructure.
- m,n: The number of VNFs, and virtual links within the $nsli$, respectively.
- $fcpu_j$: The processing cost of the VNF instance j.
- fbw: The bandwidth cost of the virtual link j.
- h: The number of hops in the path where the virtual link j is allocated.
- $cpu(vnf_j)$: is the cpu need of $vnfj$ in $nsli$.
- $bw(v_j)$: is the bandwidth need of virtual link v_j in $nsli$.

The Deep Q-Network (DQN) serves as a DRL algorithm utilizing a neural network to estimate the Q-value associated with each action in a specific state. The agent selects the action a_t with the highest estimated Q-value, receives a reward R_t for the action taken, and observes the ensuing state s_{t+1}. The objective is to admit NSLRs that yield the highest profit. Time is discretized, and the algorithm takes a set of NSLRs within a designated time window as input. The algorithm's output comprises the accepted NSLRs that maximize profitability and minimize the running cost.

III. Composable AI framework for eXplainable RL (Composable XRL)

Composable AI is a pioneering approach to Artificial Intelligence (AI) that emphasizes modularity and collaboration within an AI system. This framework integrates diverse AI subsystems to work together harmoniously, with each subsystem specializing in a specific task or domain. By combining the strengths of various components, Composable AI has the capacity to tackle complex problems in a holistic and flexible manner. This approach not only enhances the adaptability and problem-solving capabilities of AI systems but also allows for the seamless integration of new AI technologies and components as they emerge. Composable AI represents a versatile and forward-thinking paradigm that promises to revolutionize AI by making it more agile, effective, and capable of addressing a wide range of challenges in today's rapidly evolving technological landscape.

In our case, the proposed framework regroups three cutting-edge AI techniques: LLM, Prompt Engineering, and an eXplainable RL. These components work collaboratively to provide clear and comprehensible explanations of DRL decisions regarding the *"Admission Control of network slices where the end goal is to optimize the 5G/6G service provider's profit"*.

The LLM in our framework serves as an AI chatbot, capable of delivering natural-language explanations for DRL decisions in response to user inquiries. The unique advantage of AI chatbots is their ability to provide responses in a conversational, human-like manner. However, they are not without their challenges. One of these challenges is the potential for the LLM to *"hallucinate"*, which means generating nonsensical text that may not accurately reflect the provided source input. This can result in explanations that exhibit low fidelity, undermining the trustworthiness of the AI system.

To overcome these challenges and ensure that the explanations are both faithful and stable, our framework incorporates a dedicated prompt engineering module for the AI chatbot. Our mechanism involves providing a set of targeted, initial questions (prompts) before the actual user question. This approach is designed to increase the correctness of the answers generated by the chatbot, resulting in explanations with higher fidelity and stability. Additionally, we carefully select and optimize the hyper-parameters of the underlying LLM to further enhance the quality and accuracy of the explanations. This combination ensures that users receive natural-language explanations that are not only understandable but also reliable, promoting transparency and trust in the decision-making processes of DRL.

A. System Design

As prerequisites for the successful implementation of this framework, two fundamental components are paramount. Firstly, the dataset collection phase stands as an essential foundation. Secondly, an equally critical requirement is a clear and comprehensive description of the network slicing environment carefully prepared by domain experts. This description serves as a *"context meta-data"* that guides the Prompt Generator and eventually the LLM.

As illustrated in Fig. 2, the workflow of the proposed framework can be summarized as follows:

*1) **Dataset Collection and Pre-processing**:* We train the DRL black-box model which is in our case Deep Q-Networks (DQN) algorithm on the admission control management of NSLs where we capture and collect a dataset during runtime. To begin, the dataset should encompass diverse and representative scenarios, that the DRL agent is expected to encounter during its training process. In our scenario where a Network Slice Provider (NSP) seeks to optimize the admission control for network slicing, it is imperative to cover various states, actions, and outcomes such as operational time, cost associated with running the slice, and the number of VNFs to ensure the model's robustness. Moreover, it is essential to ensure that the dataset encompasses a range of policy variations, including both successful and less successful ones. Once the DRL training is complete, the collected dataset can be utilized in the eXplainable RL model, where the reward decomposition method can help disentangle the influence of different factors

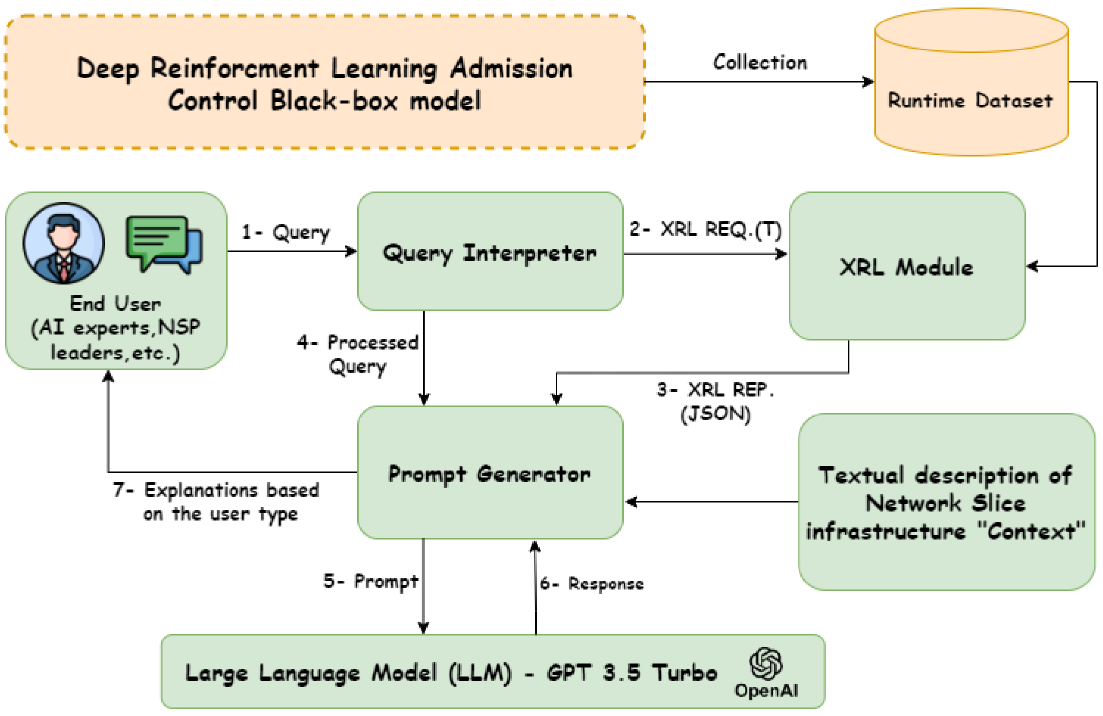

Deep Reinforcment Learning Admission Control Black-box model
Collection
Runtime Dataset
End User (AI experts,NSP leaders,etc.)
1- Query
Query Interpreter
2- XRL REQ.(T)
XRL Module
4- Processed Query
3- XRL REP. (JSON)
Prompt Generator
7- Explanations based on the user type
Textual description of Network Slice infrastructure "Context"
5- Prompt
6- Response
Large Language Model (LLM) - GPT 3.5 Turbo
OpenAI

Fig. 2: The Proposed Composable XRL framework that leverages LLMs, Prompt Engineering, and Explainable RL to produce personalized natural language explanations of DRL decisions about the admission control of 6G slices.

(e.g. state-action-reward, Cst, OT, nbr of VNF, etc.) on the agent's decision-making. Careful dataset curation is pivotal to the effectiveness of this process, enabling meaningful insights into the DRL agent's behavior and facilitating informed adjustments and optimizations.

*2) **Query Interpreter**:* The "Query Interpreter" plays a pivotal role in the initial phase of the framework. Its primary function is to decipher critical information from the user query, specifically identifying the user's profile and the desired explanation type. To streamline this process, end users are guided to follow a designated query template. This template instructs the user to start by specifying their profile, such as "AI expert" or "NSL leader," and then to indicate their preferred explanation type (for technical profiles), whether it's a "local explanation" (providing insights into a single decision within a single timestep) or a "global explanation" (covering a sequence of decisions across several timesteps). Once the user has inputted this essential information, they can proceed to submit their query to the framework.

Upon handling the user input, the *"Query Interpreter"* forwards the processed query to the *Prompt Generator* module and dispatches an XRL Request to the XRL module, including the relevant timestep (T) information.

*3) **eXplainable RL (XRL) Module**:* After receiving the XRL Request, the collected dataset is used as input for our Explainable RL model. This model employs a well-known method called "Reward Decomposition" to analyze the dataset using the timestep specified in the XRL Request.

Reward Decomposition, initially introduced with the goal of enhancing learning effectiveness. Then, it was employed by Juozapaitis et al. to enhance explainability [13]. It is a method that aims to break down the overall reward signal received by an agent into its constituent parts, enabling a deeper understanding of what factors contribute to the agent's performance. In essence, it dissects the reward into individual components, shedding light on which actions or states are responsible for a particular outcome. What makes this method particularly useful is its versatility across various DRL algorithms.

In our case, we employ one of these reward decomposition methods, specifically *the Q-value decomposition method*. This approach plays a pivotal role in elucidating the workings of the Deep Q-Network (DQN) black-box model. To achieve this, we leverage the runtime dataset that has been meticulously collected during the training of the DQN model. The Q-value decomposition method breaks down the reward signal into various components, allowing us to discern how specific state-action pairs influence the agent's decision-making process and its overall performance. This in-depth understanding of the Q-value decomposition is a crucial step in making the DQN model more transparent and explainable, ultimately enhancing its utility in the context of managing admission control of network slices.

In our framework, the outcome of the XRL module is a JSON file that contains information on the contribution of various factors, such as income cost (cst), operational time, CPU-related computational costs, and bandwidth-related costs, to the overall reward received by the DRL agent.

*4) **Prompt Generator**:* The Prompt Generator assumes a crucial role in this framework by processing three vital pieces of information:

- The Prompt Generator receives the descriptive context of the network slicing environment, which serves as the foundation for subsequent interactions with the LLM.
- It takes in the processed user query, incorporating the user's input into the prompt generation process.
- It receives the XRL reply, comprising explanations of NSL decisions in a JSON file format tailored to the specified timestep, as previously discussed.

With all this information at its disposal, the Prompt Generator goes on to craft a well-structured prompt designed to elicit the desired information from the LLM. This prompt serves as the bridge between the user's query, the network context, and the LLM's capabilities. Once the prompt is generated, it is transmitted to the LLM, where this module awaits the LLM's response.

Upon receiving the LLM's response, the Prompt Generator module finalizes its task by compiling a user-aware explanation that integrates the LLM's insights, the network context, and the user's original query. This synthesized explanation is then delivered to the end user in a textual format, ensuring that the user receives a comprehensible and contextually relevant response to their query.

B. XRL Query Templates (Bank)

The diversity in query templates holds a crucial significance in the interaction between users and AI systems. One of the key factors influencing the choice of query templates is the user type. Depending on their role and objectives, different users may seek distinct types of explanations from the AI. For instance, an AI expert may have different information needs compared to an NSL leader. Leaders often prefer relevant and non/few technical explanations that can assist them while making strategic decisions to improve the company's profitability and ensure a positive user experience. In contrast, NSL domain experts may require more technical explanations to enhance their NSL system and implement the instructions provided by the leaders. This disparity in user preferences necessitates a dynamic approach in generating query templates.

One more substantial element that influences query templates is the type of explanation required, whether local or global taking into consideration the timestep if relevant, as shown in the examples below:

- **Single Timestep:** " **At timestamp 11**, why did the model choose to terminate the URLLC slice? "
- **Sequence of Timesteps:** "**For timesteps 10,00—10,500,** how many slices have been admitted?"

This distinction is especially important in the context of technical explanations, where the choice of explanation type can significantly impact the accuracy of the results obtained from the AI model.

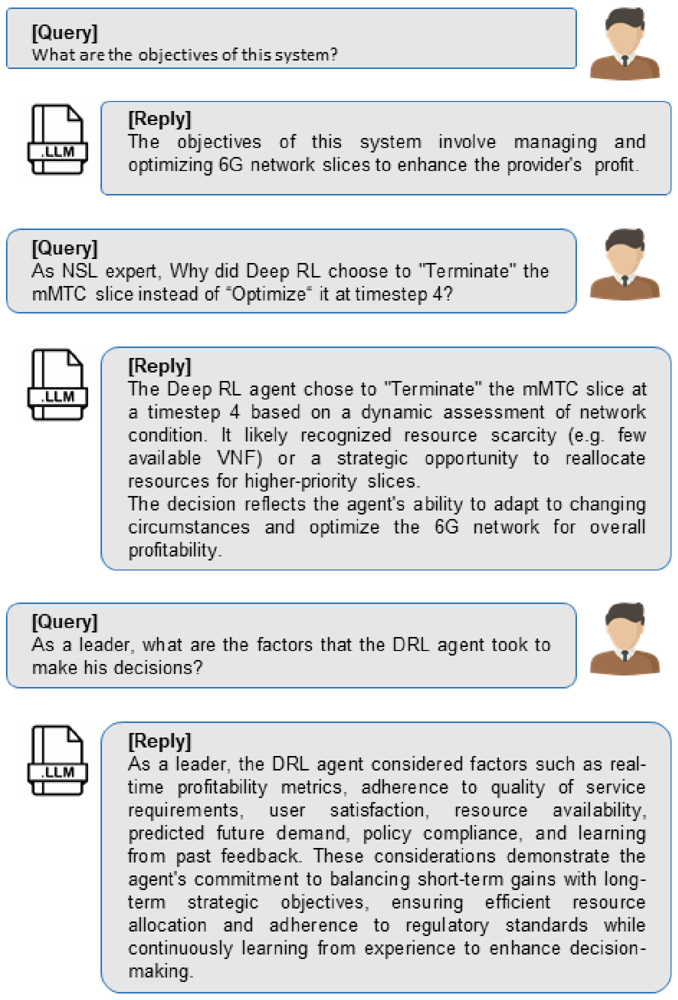
The dynamic flexibility inherent in LLM serves as a cornerstone of our system's capabilities, enabling it to respond effectively to a diverse array of query types. To address this variability and cater to diverse user intentions, Table II presents a comprehensive set of query templates that serve as a query bank.

TABLE II: Query Types and Templates

Query Type	Query Template Bank
Data Used for Training	"What data was used to train the system?"
	"Tell me about the training data."
	"What is the source of the training data?"
	"How were the labels/ground-truth produced?"
	"What is the sample size of the training data?"
	"What dataset(s) is the system NOT using?"
	"What are the potential limitations/biases of the data?"
System Outputs	"What kind of results does the system provide for DRL decisions?"
	"Explain the outcomes related to decision-making."
	"What is the significance of the system's output?"
	"How do other system components use the output?"
	"In what ways is the output employed by other system components?"
	"What's the most effective way to make use of the system's output?"
	"How can the system's output be integrated into my work process?"
System Performance	"How precise are the predictions made by the system?"
	"Tell me about the system's performance in terms of prediction accuracy."
	"How frequently does the system make errors?"
	"Under what circumstances is the system prone to accuracy/inaccuracy?"
	"What types of errors the system is inclined to commit?"
	"Is the system's performance sufficient for...?"
Counterfactual Explanations (What-ifs)	"What would have happened if we changed a certain input?"
	"What might the system forecast if this instance were to alter to...?"
	"What could the system anticipate if a specific feature were modified to...?"
	"What would be the system's prediction for a distinct instance?"
General Questions	"Why do instances A and B receive identical predictions?"
	"How does the system function?"
	"What leads the system to make particular predictions?"
	"Why does the system refrain from delivering predictions in certain scenarios?"

IV. Performance Evaluation

This section outlines the assessment of our Composable XRL framework. Initially, we introduce the metrics employed to evaluate its performance. Subsequently, we provide the specifics of the experimental setup, followed by a discussion of the results obtained in the experiments.


[Query]
What are the objectives of this system?

[Reply]
The objectives of this system involve managing and optimizing 6G network slices to enhance the provider's profit.

[Query]
As NSL expert, Why did Deep RL choose to "Terminate" the mMTC slice instead of "Optimize" it at timestep 4?

[Reply]
The Deep RL agent chose to "Terminate" the mMTC slice at a timestep 4 based on a dynamic assessment of network condition. It likely recognized resource scarcity (e.g. few available VNF) or a strategic opportunity to reallocate resources for higher-priority slices.
The decision reflects the agent's ability to adapt to changing circumstances and optimize the 6G network for overall profitability.

[Query]
As a leader, what are the factors that the DRL agent took to make his decisions?

[Reply]
As a leader, the DRL agent considered factors such as real-time profitability metrics, adherence to quality of service requirements, user satisfaction, resource availability, predicted future demand, policy compliance, and learning from past feedback. These considerations demonstrate the agent's commitment to balancing short-term gains with long-term strategic objectives, ensuring efficient resource allocation and adherence to regulatory standards while continuously learning from experience to enhance decision-making.

Fig. 3: Example of a User-LLM interactions

A. Metrics

The chosen metrics to evaluate the efficacy of our framework draw inspiration from state-of-the-art practices in explainable AI [15]. These metrics encompass:

*1) **Sensitivity**:* Sensitivity measures how changes in input features affect the explanations. It evaluates whether the XAI system appropriately adjusts the importance assigned to features when there are changes in the input. The correlation between changes in input features and changes in the importance scores assigned by the XRL system can be used as a sensitivity metric. A higher correlation indicates better sensitivity.

*2) **Stability**:* This metric assesses the consistency of explanations across different instances or perturbations of the input data. A stable explanation system should provide similar explanations for similar instances or when the input data is slightly modified. Stability is often tested by introducing small variations to the input data and observing whether the explanations remain consistent.

*3) **Comprehensibility**:* We measure the comprehensibility metric by evaluating the clarity of explanations, influenced by the intended audience (e.g., AI experts or leaders), and the concise nature of explanations (measured by factors like the number of rules, words, etc.). Additionally, the customization of explanations based on end users' profiles, providing either technical details or serving as a general decision assistant, is considered in the assessment.

B. Experiment setup

In our experimental setup, we implemented a proof-of-concept realization of our Composable XRL framework using Python and the LangChain[3] framework, integrating the OpenAI[4] API key to leverage the powerful GPT 3.5 Turbo (175B) as our LLM. The latter is built upon the robust foundation of the transformer architecture. Boasting an impressive 175 billion parameters, this language model has been meticulously trained on an extensive dataset comprising both text and code, enabling it to adeptly comprehend and generate human-like language across diverse domains. With a maximum output length of 4,096 tokens, the model exhibits a remarkable capacity for generating lengthy and coherent text [14]. This combination of immense parameter count, diverse training data, and rapid response times positions this LLM at the forefront of cutting-edge natural language processing technology. LangChain was chosen for its efficiency in streamlining the development of applications utilizing LLMs, ensuring faster response generation and the capacity to deliver more comprehensive answers [14].

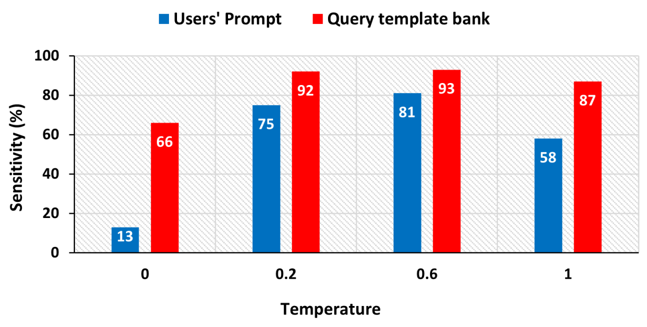
To assess Composable XRL's performance, we investigated three key independent variables. First, we examined the impact of prompting, comparing the prompt provided by the users (Zero-shot prompting) to the prompts available in the query bank. This analysis aimed to determine how much Composable XRL's effectiveness is influenced by prompt engineering and to assess its overall robustness independent of specific prompt modifications. Second, we evaluated Composable XRL's ability to generate explanations for diverse end-user profiles by varying the question form. Lastly, we scrutinized the framework's performance under different hyperparameter settings, specifically exploring temperature values within the range 0, 0.2, 0.6, 1. Additionally, we maintained a maximum token limit of 350 to strike a balance between answer length and processing efficiency.

C. Results & Discussion

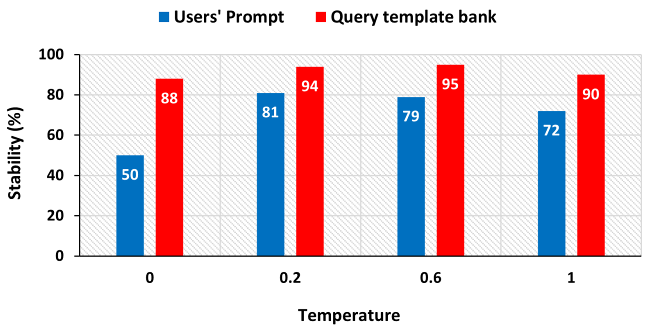
Figure 3 illustrates an example of User-LLM interactions where users from different backgrounds can ask for explanations based on their profiles. The system tries to clarify and simplify the DRL agent's actions and can go further by replying to follow-up queries about the user's needs due to the LLMs' general knowledge capabilities.

1) Prompt engineering effect: When comparing the Stability, Sensitivity, and Comprehensibility metrics between Users' Prompt (zero-shot prompting) versus Prompt from the Query bank, a clear trend emerges. User prompts show moderate Sensitivity initially (see Fig. 4a), peaking at 92 for Temperature

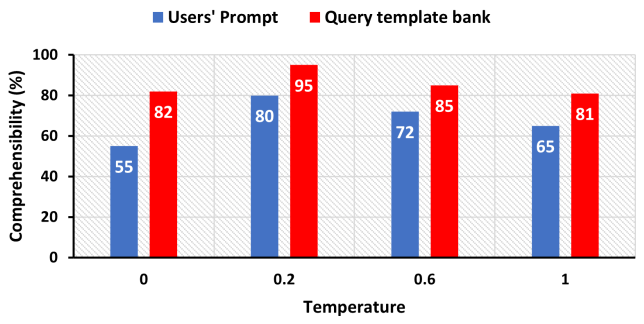
[3]https://python.langchain.com
[4]https://platform.openai.com/



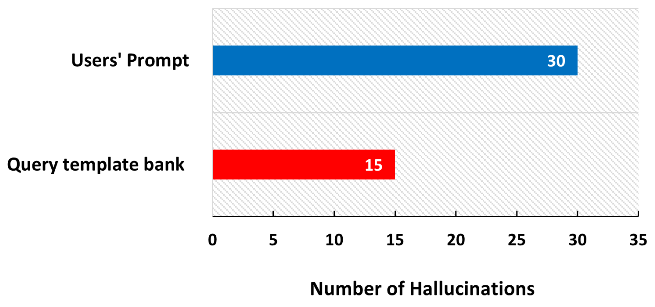
(a) Sensitivity using Users' Prompt (zero-shot) vs. Prompt from the Query bank at Different Temperatures.



(b) Stability using Users' Prompt (zero-shot) vs. Prompt from the Query bank at Different Temperatures.



(c) Comprehensibility using Users' Prompt (zero-shot) vs. Prompt from the Query bank at Different Temperatures.



(d) Number of Hallucinations when using queries from the Query Bank vs using Users' Prompt (zero-shot).

Fig. 4: Performance Evaluation of Four Metrics with varied Temperature values: Assessing the usefulness of the proposed Query Bank guiding system users (especially non-technical).

0.2, while query bank prompts consistently maintain high Sensitivity, reaching 94 at Temperature 0.2. Query prompts also demonstrate superior Stability, starting at 88 for Temperature 0 and staying above 90, as illustrated in Fig. 4b. Moreover, in the Comprehensibility case depicted in Fig. 4c, query prompts outperform user prompts across various temperatures.

This consistent superiority in all the metrics underscores the effectiveness of Prompt Engineering in providing precise and accurate explanations, outperforming the users' Prompting under various conditions.

Furthermore, clarifying the type of user in the prompt, whether a network slice leader or engineer, has a profound impact on the generation of personalized explanations and eventually its enhancement on comprehensibility levels. By specifying the user profile, the LLM can tailor its responses to cater to the distinct needs and expertise levels of these stakeholders. Network slice leaders may benefit from higher-level, strategic insights, while engineers might require more technical details. This customization not only enhances the relevance of generated explanations but also significantly contributes to overall comprehensibility for end-users. The clarity achieved through personalized prompts ensures that the LLM's responses align with the specific expectations and requirements of different user roles, thereby optimizing the utility of generated explanations in decision-making processes within the complex domain of next-generation network slicing.

2) *Impact of Temperature Variations:* This hyperparameter controls the randomness of the text generated by the model. Higher values introduce more creativity but may increase the risk of hallucinatory outputs, while a temperature of 0 ensures deterministic text generation[5]. The impact of temperature variations, as observed in Fig. 4a, Fig. 4b, and Fig. 4c provides valuable insights into the behavior of LLMs. In the context of regular Prompting, the positive correlation between temperature increases and improved Sensitivity and Stability suggests that higher temperatures enhance the model's adaptability and responsiveness. This aligns with the nature of LLMs, known for their ability to capture intricate patterns and variations in data. The observed improvements indicate that, under elevated temperatures, the model explores a broader and more diverse set of responses, contributing to heightened sensitivity and stability in explaining decisions related to network slicing tasks.

Conversely, the impact of temperature on Prompt Engineering reveals a different dynamic. The consistently high Sensitivity and Stability values across all temperature variations suggest that the effectiveness of Prompt Engineering is less

[5]https://community.openai.com/t/openai-temperature-parameter/287485

contingent on temperature changes. This resilience aligns with the inherent nature of LLMs, particularly when guided by tailored prompts. Prompt Engineering, being a deliberate and customized approach, leverages the model's capabilities effectively, demonstrating robust and stable performance irrespective of temperature fluctuations. This nuanced understanding of the impact of temperature variations underscores the intricate interplay between model architecture, training, and the nature of input queries in shaping the behavior of LLMs in explaining decisions critical to network slicing and profit optimization.

3) Hallucinations: The number of hallucinations data reveals a distinct contrast between scenarios where end users employ a predefined template and those where users independently engage with the system without providing sufficient contextual information. It's important to highlight that for the assessment of this metric, we engaged technical experts from EURECOM 5G facility [16]. Their expertise was pivotal in conducting this experiment. As presented in Fig. 4d, when end users utilize one of the queries from the template bank, the system exhibits a notably lower number of hallucinations, with only 15 instances out of 200 queries. This outcome can be attributed to the structured and specific nature of the queries within the template bank, guiding the model toward focused and accurate responses and reducing the likelihood of generating hallucinations.

On the other hand, when end users decide to chat with the system independently, without leveraging the provided templates and without furnishing ample context for their queries, the number of hallucinations increases to 30. This escalation aligns with the inherent tendency of LLMs to produce responses that may not precisely align with the user's intended task, particularly when operating without specific cues or constraints. The observed correlation between increasing temperature values and a higher number of hallucinations in this scenario underscores the model's heightened creativity and flexibility under elevated temperatures, leading to a greater likelihood of generating responses that might be considered hallucinatory.

In summary, the lower number of hallucinations when utilizing the template bank emphasizes the effectiveness of structured and directed input queries in mitigating undesired outputs. However, it is noteworthy the LLM is able to satisfy all stakeholder users, thus reducing the need for specialized human intervention to understand the typical outputs of XRL.

V. Conclusion

This paper tackles the intricacies of 6G network management, with a particular emphasis on understanding and elucidating the decision-making processes of DRL models. In this light, it introduces the concept of Composable XRL, specifically exploring the application of XRL and LLMs, in tandem with Prompt Engineering to illuminate DRL algorithm decisions within the context of network slicing. The proposed approach renders human-understandable explanations for the decisions made by DRL agents, thereby transforming the inherent black-box nature of DRL into an interpretable format. This initiative is pivotal in fostering trust, accountability, and enhancing overall network performance. The case study demonstrates improved transparency and comprehensibility of DRL agents' actions, thereby providing valuable insights for network operators, regulators, and stakeholders. For future works, we aim to assess the efficacy of the proposed framework in addressing other 5G/6G challenges (green resource allocation, scheduling, etc.) with more advanced LLMs including GPT-4 and Llama 3 (8B, 70B). The overarching goal is to develop a comprehensive and globally applicable explainable RL chatbot specialized in networking.

Acknowledgment

This work is supported by the European Union's Horizon Program under the 6G-Bricks project (Grant No. 101096954).

References

[1] F. Rezazadeh, H. Chergui, and J. Mangues-Bafalluy, "Explanation-guided deep reinforcement learning for trustworthy 6G RAN slicing," arXiv [cs.NI], 2023. [Online]. Available: http://arxiv.org/abs/2303.15000

[2] C. Fiandrino, G. Attanasio, M. Fiore, and J. Widmer, "Toward native explainable and robust AI in 6G networks: Current state, challenges and road ahead," Comput. Commun., vol. 193, pp. 47–52, 2022, doi: 10.1016/j.comcom.2022.06.036.

[3] N. C. Luong et al., "Applications of deep reinforcement learning in communications and networking: A survey," IEEE Commun. Surv. Tutor., vol. 21, no. 4, pp. 3133–3174, 2019, doi: 10.1109/comst.2019.2916583.

[4] A. Palm, A. Metzger, and K. Pohl, "Online reinforcement learning for self-adaptive information systems," in Advanced Information Systems Engineering, Cham: Springer International Publishing, 2020, pp. 169–184.

[5] Vo, N.S., Masaracchia, A., Sheng, Z. and Nguyen, T.T., 2023. Towards 6G Networks: Technologies, Services and Applications. Mobile Networks and Applications, pp.1-3.

[6] Y. Qing, S. Liu, J. Song, H. Wang, and M. Song, "A survey on explainable reinforcement learning: Concepts, algorithms, challenges," arXiv [cs.LG], 2022. [Online]. Available: http://arxiv.org/abs/2211.06665

[7] Mohseni, S., Zarei, N. and Ragan, E.D., 2021. A multidisciplinary survey and framework for design and evaluation of explainable AI systems. ACM Transactions on Interactive Intelligent Systems (TiiS), 11(3-4), pp.1-45.

[8] E. Cambria, L. Malandri, F. Mercorio, M. Mezzanzanica, and N. Nobani, "A survey on XAI and natural language explanations," Inf. Process. Manag., vol. 60, no. 1, p. 103111, 2023, doi: 10.1016/j.ipm.2022.103111.

[9] M. Uzba and P. Biecek, "What would you ask the machine learning model? identification of user needs for model explanations based on human-model conversations," in ECML PKDD 2020 Workshops, I. Koprinska, Ed., Springer, 2020.

[10] Q. V. Iao, D. M. Gruen, and S. Miller, "Questioning the AI: informing design practices for explainable AI user experiences," in Conference on Human Factors in Computing Systems (CHI '20), ACM, 2020.

[11] W. F. Villota Jácome, O. M. Caicedo Rendon, and N. L. S. da Fonseca, "Admission control for 5G network Slicing based on (deep) reinforcement learning," 2021. doi: 10.36227/techrxiv.14498190.v1.

[12] A. Metzger, J. Bartel, and J. Laufer, "An AI chatbot for explaining Deep Reinforcement Learning decisions of service-oriented systems," arXiv [cs.LG], 2023. [Online]. Available: http://arxiv.org/abs/2309.14391

[13] Z. Juozapaitis, A. Koul, A. Fern, M. Erwig, and F. DoshiVelez, "Explainable reinforcement learning via reward decomposition," in IJCAI/ECAI Workshop on Explainable Artificial Intelligence, 2019

[14] J. Ye et al., "A comprehensive capability analysis of GPT-3 and GPT-3.5 series models," arXiv [cs.CL], 2023. [Online]. Available: http://arxiv.org/abs/2303.10420

[15] G. Vilone and L. Longo, "Notions of explainability and evaluation approaches for explainable artificial intelligence," Inf. Fusion, vol. 76, pp. 89–106, 2021.

[16] S. Arora, "A 5G Facility for Trialing and Testing Vertical Services and Applications"," IEEE Internet of Things Magazine, 2022.