This is the author's accepted version of the article. The final version published by IEEE is S. Duttagupta, M. Jabbari, C. Fiandrino, M. Fiore, and J. Widmer, "SymbXRL: Symbolic Explainable Deep Reinforcement Learning for Mobile Networks," IEEE INFOCOM 2025 - IEEE Conference on Computer Communications, doi: TBD.

SymbXRL: Symbolic Explainable Deep Reinforcement Learning for Mobile Networks

Abhishek Duttagupta*†◊, MohammadErfan Jabbari*◊, Claudio Fiandrino*, Marco Fiore* and Joerg Widmer*
*IMDEA Networks Institute, Spain, †Universidad Carlos III de Madrid, Spain
Email: {name.surname}@imdea.org

***Abstract*—The operation of future 6th-generation (6G) mobile networks will increasingly rely on the ability of Deep Reinforcement Learning (DRL) to optimize network decisions in real-time. DRL yields demonstrated efficacy in various resource allocation problems, such as joint decisions on user scheduling and antenna allocation or simultaneous control of computing resources and modulation. However, trained DRL agents are closed-boxes and inherently difficult to explain, which hinders their adoption in production settings. In this paper, we make a step towards removing this critical barrier by presenting SymbXRL, a novel technique for EXplainable Reinforcement Learning (XRL) that synthesizes human-interpretable explanations for DRL agents. SymbXRL leverages symbolic AI to produce explanations where key concepts and their relationships are described via intuitive symbols and rules; coupling such a representation with logical reasoning exposes the decision process of DRL agents and offers more comprehensible descriptions of their behaviors compared to existing approaches. We validate SymbXRL in practical network management use cases supported by DRL, proving that it not only improves the semantics of the explanations but also paves the way for explicit agent control: for instance, it enables intent-based programmatic action steering that improves by 12% the median cumulative reward over a pure DRL solution.**

I. Introduction

The future 6th generation (6G) of mobile networks promises unprecedented connectivity with ultra-high data rates, minimal latency, and massive device support [1]. Recent reports predict global mobile network data traffic will reach 466 exabytes per month by 2029, driven by the proliferation of connected devices and data-intensive applications [2]. As network complexity increases, Artificial Intelligence (AI) emerges as a crucial enabler for network management and optimization [3]. The integration of AI is expected to enhance network efficiency and enable new services and applications [4].

A specific AI paradigm that shows significant promise for 6G is DRL, which combines Deep Learning (DL) with Reinforcement Learning (RL) to tackle complex mobile network challenges [5]. DRL enables agents to learn optimal policies through interaction with their environment and has been successfully applied to resource allocation [6] and network slicing [7] in 5G systems. More advanced DRL applications are emerging for 6G, including dynamic spectrum management [8] and multi-agent coordination for network slicing [9], showcasing potential throughout the network ecosystem [10].

However, these DRL applications operate as closed boxes that inherently lack interpretability, which makes debugging and troubleshooting hard [11], may compromise performance [12] and generally curbs adoption by network providers [13]. Recent advances in EXplainable Artificial Intelligence (XAI) address this lack of transparency, yet current XAI solutions for DRL agents, such as *EXPLORA* [14], METIS [15] and post-hoc interpretation methods like SHAP [16] and LIME [17] often fall short in providing meaningful, human-understandable explanations for complex network management systems [18], as we will also detail in our study.

In this paper, we propose SymbXRL, a novel XRL technique leveraging symbolic AI, specifically First-Order Logic (FOL), to synthesize comprehensible explanations with rich semantics from symbolic representations of the states and actions of DRL agents. SymbXRL employs FOL to formalize the agent's behavior and decision-making process, allowing for more intuitive and interpretable explanations. This approach serves three primary objectives: *(1)* providing simple, logically structured explanations to understand and compare DRL agents; *(2)* enabling Intent-based Action Steering (IAS) through logical rules in the form of FOL representation; and *(3)* leveraging symbolic knowledge to identify flaws in the design process of new agents by analyzing logical inconsistencies or unexpected patterns in the agent's behavior. By translating complex numerical states and actions into symbolic logic statements, SymbXRL offers a unique perspective on DRL agent behavior, bridging the gap between low-level operations and high-level, human-interpretable concepts and addressing current limitations of XRL solutions.

We validate our approach with two distinct DRL use cases addressing critical 5G and 6G challenges. In the first application, a DRL agent [19] controls Radio Access Network (RAN) slicing and scheduling on a next Generation Node B (gNB) as an O-RAN compliant xApp for three network slices serving different traffic categories. The second use case employs a DRL agent [20] for resource scheduling in Massive MIMO. These agents offer diverse decision-making contexts: the first has a multi-modal action space with both continuous and discrete factors, where the actions affect all of its next state input Key Performance Indicator (KPI)s; the second agent has a discrete action space that affects a subset of the input KPIs. This diversity allows showcasing the flexibility of SymbXRL across different scenarios.

The key contributions ("C") and findings ("F") of our study are summarized as follows.

C1. We propose SymbXRL, a novel explainer for DRL agents using symbolic representations with FOL to synthesize human-interpretable explanations.

◊These authors contributed equally to this work.

C2. We validate SYMBXRL in two diverse use cases that rely on DRL agents for network slicing and Massive MIMO scheduling, and demonstrate the superior intelligibility and detailed level of our solution compared to state-of-the-art approaches.

C3. For reproducibility and to further stimulate the research in the field, we release the artifacts of our study (code of RL agents and SYMBXRL) at:https://github.com/RAINet-Lab/symbxrl.

F1. We prove that SYMBXRL provides human-readable and comprehensible symbolic explanations, which improve explanations of state-of-the-art methods and bridge the gap between DRL agent behavior and human understanding.

F2. We show that SYMBXRL's symbolic representation enables flexible IAS policies, which (i) improve the cumulative reward of DRL agents and (ii) enable enforcement of operational constraints on the agent actions. Our experiments demonstrate that these capabilities result in a 12% median improvement in cumulative reward over baseline performance, which outperforms existing XRL methods like METIS.

II. BACKGROUND AND TECHNICAL FOUNDATIONS

This section provides the necessary background and technical foundations for understanding our proposed approach. We cover the key concepts of RL and DRL, explainability in AI, and Symbolic AI with a focus on FOL.

A. Reinforcement Learning and Deep Reinforcement Learning

Reinforcement Learning (RL) is a computational approach to learn from interaction with an environment. RL agents learn optimal policies by taking actions in an environment to maximize cumulative reward signal. The RL process is typically modeled as a Markov Decision Process (MDP), defined by the tuple (S, A, P, R, γ), where S is the set of states, A is the set of actions, $P(s_{t+1}|s_t, a_t)$ are state transition probabilities, $R(s_t, a_t)$ is the reward, and $\gamma \in [0, 1]$ is the discount factor.

At each time step t, the agent observes the current state $s_t \in S$, takes an action $a_t \in A$ following a policy $\pi : S \to P(A)$, receives a reward r_t, and transitions to the next state s_{t+1}. The agent's goal is to learn an optimal policy π^* that maximizes the expected cumulative discounted reward:

$$\pi^* = \arg\max_{\pi} \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t r_t | \pi\right]. \quad (1)$$

DRL extends RL by utilizing Deep Neural Networks (DNN) to approximate value functions or policies, enabling RL to scale to high-dimensional state and action spaces [21]. DRL has been successfully applied to various domains, including playing complex games [22] and robotic control [23].

B. Explainability and XAI

XAI refers to techniques and methods that make the behavior of AI systems comprehensible to humans. The primary goal of XAI is to create models that can provide clear and understandable explanations for their decisions, facilitating trust and adoption in critical applications [24].

In the context of DRL, explainability is particularly challenging due to the complex interactions between the agent and the environment over time [12]. To address these challenges, Explainable Reinforcement Learning (XRL) techniques have been developed to provide insights into the decision-making process of DRL agents [25]. XRL methods can be either *intrinsic* when they modify the RL algorithm itself to be explainable [26] or *post-hoc* if the explanations are produced without altering the original model [14]–[17].

Existing XRL methods, however, often fall short in providing meaningful, human-understandable explanations for complex network management systems [27]. METIS [15], an explainer using a mixture of Decision Tree (DT) and hypergraphs, requires multiple rollouts of the whole RL environment to improve the DT efficiency, with a time complexity that makes this approach unsuitable for continuous, complex, multi-modal environments. *EXPLORA* [14], an attribute graph-based explainer, uses low-level agent numerical data that complicates the synthesis of compact and easy-to-understand insights.

To address the shortcomings of the state of the art, we leverage FOL, a form of symbolic AI, to explain the behavior and decision-making process of DRL agents.

C. Symbolic AI and First-Order Logic

Symbolic AI, in contrast to Statistical AI approaches like DL, uses human-readable symbols and rules to represent knowledge and reason about it. First-Order Logic (FOL) is a powerful formalism within Symbolic AI that allows for the representation of complex relationships and reasoning.

FOL consists of the following key components:

- **Constants:** Represent specific objects in the domain.
- **Variables:** Stand for arbitrary objects.
- **Predicates:** Express properties of objects or relationships between objects.
- **Quantifiers:** "For all" ($\forall$) and "There exists" ($\exists$).
- **Logical connectives:** AND ($\wedge$), OR ($\vee$), NOT ($\neg$), IMPLIES ($\Rightarrow$).

To illustrate how FOL can represent policies in a networking context, consider the example: "If a user's data consumption exceeds the plan's limit at 10 GB, throttle their connection speed." We can formalize this policy in FOL as follows.

- **Predicates:**
  - Exceeds(x, y): "user x's data usage exceeds y GB."
  - Throttle(x): "throttle user x's connection speed."
- **Constants and Variables:**
  - u: a user.
  - L: the plan's limit, which is a constant value of 10 GB.
- **FOL Statement:** $\forall u\,(\text{Exceeds}(u, L) \Rightarrow \text{Throttle}(u))$.

This FOL statement reads as: "For all users u, if user u's data usage exceeds the plan's limit L (which the operator has set at 10 GB), then throttle user u's connection speed."

We chose FOL for our work due to its balance between expressiveness and simplicity. Unlike propositional logic, FOL

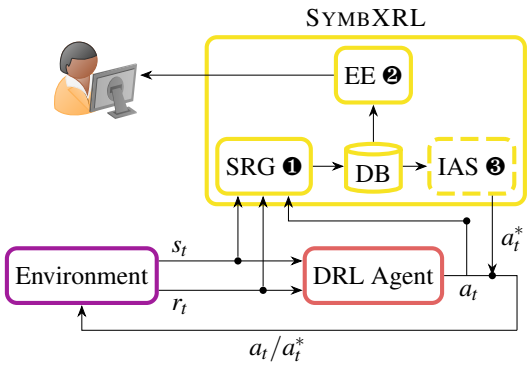


Fig. 1. SymbXRL's architecture and interaction with a DRL agent operating on a target environment.

can define predicates, variables, and constants, capturing the complexity of our agent's behavior. At the same time, it avoids the advanced features of higher-order logic that are beyond our current requirements [28].

By leveraging FOL, we can create a symbolic representation of DRL agents' behavior, enabling more intuitive and interpretable explanations of their decision-making processes [29]. This approach bridges the gap between the low-level numerical operations of DRL and high-level, human-understandable concepts, addressing limitations of current XRL techniques.

III. SymbXRL

In this section, we describe the details of SymbXRL and provide a comprehensive overview of its framework. This includes creating symbolic representations for the state and action space of the agent, generating explanations, and integrating these representations into the explainability pipeline. We also describe how we leverage this symbolic representation to provide IAS for the agents.

A. Overview of SymbXRL

SymbXRL leverages FOL to produce explanations for DRL agents. The symbolic representation offers the following three key advantages:

- *Enhanced Interpretability:* By representing agent behavior in logical terms, SymbXRL enables formal reasoning techniques for analyzing and verifying agent behavior. This approach produces explanations that are more intuitive compared to previous works [15].
- *Concise Representation:* The symbolic representation provides a more concise and compact representation of the state-action space, facilitating easier analysis and visualization compared to state-of-the-art explainability tools [14].
- *Flexible Action Steering:* This symbolic framework allows the definition of high-level rules to guide agent behavior, enhancing performance tuning and ensuring better compliance with operational constraints.

Fig. 1 illustrates SymbXRL's architecture and integration with DRL pipelines. The key components are:

1) **Symbolic Representation Generator (SRG ❶):** Transforms numerical states and actions into FOL terms and stores this information in a database.

2) **Explanation Engine (EE ❷):** Utilizes symbolic representations to generate human-readable explanations of agent behavior in the form of a Knowledge Graph (KG).

3) **Intent-based Action Steering (IAS ❸):** This optional module enables real-time fine-tuning of agent behavior by translating high-level intents via FOL representation.

B. Symbolic Representation Generator (❶)

The cornerstone of SymbXRL is creating symbolic representations for the DRL agent's state and action spaces, mapping numerical or categorical values into logical terms in FOL. The process begins by defining the explanation goal that guides the selection of relevant variables and the level of abstraction.

For each selected variable, we define FOL terms that capture essential information:

- For continuous variables: **predicate**(variable, quartile), where the predicate indicates the direction of change (increase, decrease, or constant).
- For categorical variables: **toCategory**(variable).

The quartiles (Q1, Q2, Q3, Q4) represent the relative magnitude of the value, calculated efficiently using the P2 algorithm [30] for online computation of quartile markers. These symbolic representations are stored in a database (DB) for use by other components and services.

An Example. To clarify, consider a DRL agent performing resource allocation on a gNB. Its state might include the number of connected users (integer), average user throughput (float, Mbps), and interference level (categorical: Low, Medium, High). The action space could involve adjusting transmission power (float, dBm) and changing frequency band (categorical: Low, Mid, High). In this context, **inc**(users, Q4) represents a significant increase in user numbers, while **toHigh**(interf) indicates high interference levels. The choice of predicates and granularity balances expressiveness and complexity, optimized for the specific application and explanatory goal.

C. Explanation Engine (❷)

The Explanation Engine (EE) leverages the symbolic representations created by the SRG module (❶) to generate insights into the agent's behavior. This approach enables two analytical methods: *probabilistic analysis* and *KG analysis*.

In *probabilistic analysis*, the EE performs four main steps: collects symbolic representations of states and actions from the DB, counts occurrences of each unique symbolic state and action, calculates probabilities or frequencies of these occurrences, and visualizes the results through probability distributions, correlation density maps, or KGs. This provides insights into both input state distributions and the correlation of agent's actions and their effects effects on the environment.

In *KG analysis*, EE constructs a graph where nodes represent symbolic actions and edges are transitions between them. The weight of each node and edge corresponds to the frequency of that action and transition. KG reveals the agent's learned decision-making strategies and overall behavior patterns.

An Example. Applying these analyses to our toy example: The correlation density map reveals that the agent maintains average

throughput in Q3 (**const**(throughput, Q3)) by keeping transmission power in Q4 (**const**(tx_power, Q4)). The KG shows the agent frequently switches between mid and high frequency bands, represented as transitions between **toMid**(freq_band) and **toHigh**(freq_band) nodes.

These analytical approaches offer several advantages over state-of-the-art methods [14], [15]: (i) direct insights into the agent's decision-making process, (ii) revealing patterns not apparent from reward analysis, and (iii) enabling comparison between different agents or versions. While our framework generalizes to various DRL agents and environments, the choice of symbolic representations and FOL definitions is use-case specific. Though discretization may lead to some loss of detail, a well-defined explanation goal can mitigate this limitation.

By combining these analytical approaches with symbolic representation, the EE ❷ module generates intelligible explanations of the agent's behavior, bridging the gap between closed-box DRL agents and human understanding.

D. Intent-based Action Steering (❸)

We introduce Intent-based Action Steering (IAS), a mechanism integrating symbolic representation to guide agent behavior towards network operators' specific intents. Inspired by decision shielding in reinforcement learning [31], SYMBXRL's IAS leverages symbolic knowledge to surpass traditional methods. Unlike approaches that may reduce rewards or introduce unfamiliar states or actions, IAS selects actions from the agent's prior experiences, ensuring both constraint satisfaction and performance optimization.

SYMBXRL's unique IAS approach operates on discretized and concise state and action spaces, unlike previous methods such as *EXPLORA* [14]. This design offers two key advantages: it *(i)* enables efficient and accurate matching between current and past states and actions, reducing computational complexity; *(ii)* allows operators to define intents using the same FOL format as agent explanations, seamlessly integrating intents without compromising learned behavior.

We demonstrate the versatility of SYMBXRL's IAS through three distinct use cases:

1) *Reward maximization:* enhances the cumulative reward of the agent by maximizing each step's reward through targeted action steering (a_t^*), achieving:

$$a_t^* = \underset{a_1,\ldots,a_T}{\arg\max} \sum_{t=1}^{T} r_t(s_t, a_t) \text{ s.t. } a_t \in \mathcal{A}(s_t). \tag{2}$$

where $\mathcal{A}(s_t)$ is the set of actions the agent has previously taken for state s_t plus the current timestep action. a_t^* is the optimal action chosen to maximize the reward at each step.

2) *Decision conditioning:* applies constraints to the agent's actions to enforce operational limits or policy requirements without excluding reward maximization. For example:

- Schedule a specific user: **Schedule**(a_t, UserID)
- Do not schedule users in a group: $\forall$ UserID $\in$ Group : **notSchedule**(a_t, UserID)

These examples illustrate the flexibility in defining policies beyond simple reward or throughput maximization [14]. This mode allows us to improve the agent's cumulative reward while applying operational constraints.

3) *Accelerated learning:* Reducing training time for DRL agents is crucial for faster deployment and resource efficiency [32]. By effectively leveraging the agent's acquired knowledge, IAS allows agents trained for fewer episodes to achieve competitive cumulative rewards compared to those trained for longer periods.

An Example. Applying IAS for reward maximization to our toy example: Given the environment's input state and agent's action, IAS checks the KG for a potentially better action (transmission power and frequency band) that could yield a higher expected reward. This replacement action is chosen based on the similarity between the current state and the state where the action was previously applied.

IV. APPLICATION TO PRACTICAL USE CASES

The operation and advantages of SYMBXRL are best demonstrated through practical applications. This section presents a comprehensive empirical evaluation of SYMBXRL using two RL use cases for network management. We detail the evaluation framework and experimental setup (§IV-A and §IV-B), followed by our findings on DRL agent behavior (§IV-C) and performance improvements achieved through IAS (§IV-D).

A. Evaluation Framework

Our evaluation of SYMBXRL focuses on two main objectives. First, we assess the explanation quality and interpretability through a qualitative analysis of compactness and clarity of the generated explanations. Second, we quantify performance improvements achieved with IAS by measuring changes in cumulative reward and training efficiency.

B. Symbolic Representations for the Agents

To validate SYMBXRL, we employ two distinct DRL agents addressing challenging resource allocation and scheduling problems in mobile networks. By selecting these agents, we demonstrate the versatility of SYMBXRL. *Table I* summarizes state-action space and the FOL representations for each agent.

A1 Network Slicing and Scheduling Agent [19]: This agent jointly controls RAN slicing and scheduling policies for three slices $\mathcal{L} = \{\text{eMBB, mMTC, URLLC}\}$ in an OpenAI Gym environment with O-RAN compliant xApps. The agent is trained and evaluated in the Colosseum emulator using two traffic profiles: TRF1 (slice-based) and TRF2 (uniform). It operates in two modes, each favoring one slice (i.e., eMBB or URLLC) by adjusting the weight of each slice's KPI effect in the reward function (`tx_brate` for eMBB and `dl_buffer` for URLLC).

For FOL-based symbolic representation, we use the format **predicate**(variable, Quartile) for continuous, unbounded variables (e.g., KPIs). For bounded discrete variables (e.g., PRB allocation), we define categories and use **predicate**(variable, starting-category, final-category). Categorical variables are represented as **toCategory**(variable). Quartiles are not used

TABLE I
STATE AND ACTION SPACES AND THEIR SYMBOLIC REPRESENTATIONS OF THE TWO DRL AGENTS USED FOR VALIDATION

Agents	**A1: Network Slicing and Scheduling Agent**	**A2: Massive MIMO Scheduling Agent**
State Space	$s_t \in \mathcal{S} = \mathbb{R}^{M \times K \times \lvert\mathcal{L}\rvert}$ where: • $\mathcal{M}$: Measurements, $M = 10$ • $\mathcal{L}$: Slices, $\mathcal{L} = \{$eMBB, mMTC, URLLC$\}$ • $\mathcal{K}$: KPIs, $\mathcal{K} = \{$tx_brate, tx_pkts, dl_buffer$\}$	$s_t \in \mathcal{S} = \mathbb{R}^{K \times \lvert\mathcal{N}\rvert}$ where: • $\mathcal{N}$: Number of Users, $N = 7$ • $\mathcal{K}$: KPIs, $\mathcal{K} = \{$MSE, DTU, G$\}$
Symbolic Representation of State Space	For each $k \in \mathcal{K}$ and $l \in \mathcal{L}$: • $\bar{k}_l \rightarrow$ Pred$(\bar{k}_l, Q)$, where $\bar{k}_l = \frac{1}{M}\sum_{m=1}^{M} k_{l,m}$ • Pred $\in$ Predicate $= \{$inc, dec, const$\}$ • Q $\in$ Quartile $= \{$Q1, Q2, Q3, Q4$\}$ • $M = 10$: Number of measurements	For each $k \in \mathcal{K}$ and $g \in \mathcal{G}$: • $\bar{k}_g \rightarrow$ Pred$(\bar{k}_g, Q)$, where $\bar{k}_g = \frac{1}{\lvert U_g\rvert}\sum_{u \in U_g} k_u$ • Pred $\in$ Predicate $= \{$inc, dec, const$\}$ • Q $\in$ Quartile $= \{$Q1, Q2, Q3, Q4, MAX$\}$ • U_g: Set of users in group g
Action Space	$a_t \in \mathcal{A} =$ PRB$^{\lvert\mathcal{L}\rvert} \times$ SP$^{\lvert\mathcal{L}\rvert}$ where: • PRB: Physical Resource Block $= \{1, 2, \ldots, 50\}$ • SP: Scheduling Policy $= \{$WF, RR, PF$\}$ • $\mathcal{L}$: Slices	$a_t \in \mathcal{A} = \{0, 1\}^N$ where: • N: Number of users, $N = 7$ • 0: Do not schedule a user • 1: Schedule a user
Symbolic Representation of Action Space	For each $l \in \mathcal{L}$: • PRB$_l \rightarrow$ Pred(PRB, C$_{\text{start}}$, C$_{\text{final}}$) • SP$_l \rightarrow$ toPolicy(sched) • Pred: Predicate $\in \{$inc, dec, const$\}$ • C: PRB Category $\in \{$C1, C2, C3, C4, C5$\}$ • toPolicy $\in$ Scheduling Policy $= \{$toWF, toRR, toPF$\}$	For each $g \in \mathcal{G}$: • $a_g \rightarrow$ Pred$(g, Q, Percentage)$ • Pred: Predicate $\in \{$sched$\}$ • g: Group number, $g \in \mathcal{G}$ • Q: Quartile $\in \{$Q1, Q2, Q3, Q4, MAX$\}$ • Percentage: Round$_{\{0,25,\ldots,100\}}\left(\frac{\lvert\text{Sched. users in } g\rvert}{\lvert\text{Total users in } g\rvert} \times 100\right)$

for PRB allocation as categories offer adjustable intervals for operator goals. Here, we define categories such that each category covers 10% of the available PRB range. Mathematically, the interval for category C_i is defined as $\left[\frac{(i-1)\times\text{PRB}}{10}, \frac{i\times\text{PRB}}{10}\right)$, where i ranges from 1 to 10 and PRB is 50. For space reasons, we only present results for TRF1 in §IV.

A2 Massive MIMO Scheduling Agent [33]: This agent focuses on resource scheduling in Massive MIMO networks to maximize spectral efficiency while maintaining fairness among users. We evaluate two DRL agents: the SMART scheduler based on Soft Actor-Critic (SAC) [34] proposed by the authors [33], and a Deep Q-Network (DQN)-based implementation [35]. The evaluation uses the Indoor Mobility Channel Measurements dataset [33], providing channel state information for Line of Sight (LoS) and Non-Line of Sight (NLoS) scenarios, as well as low-speed and high-speed mobility profiles. The agent's state space includes KPIs such as Maximum Available Spectral Efficiency (MSE), Data Transmitted of User (DTU), and User Group Label (G). User's G are assigned based on their channel state correlation. For FOL-based symbolic representation, we focus on groups rather than individual users to assess if the agent learns to avoid interference from joint scheduling of different group's users. Continuous variables (MSE, DTU) use **predicate**(variable, Quartile). The agent's decision is represented as **sched**(group_number, quartile, percentage), where percentage is the proportion of scheduled users in the group and quartile is calculated for the number of scheduled users in the group with respect to the total number of users. For DTU and user scheduling, we also use the MAX quartile to represent the highest observed value up to the current timestep, provided by the P2 algorithm [30]. This approach enables comparison of decisions across groups and assesses interference mitigation.

The diverse nature and purpose of these agents allow us to assess SYMBXRL's capabilities across different network management scenarios and DRL architectures.

Table II demonstrates the efficiency of SYMBXRL in converting the agent's action space to symbolic representations, compared to *EXPLORA* [14], which operates on low-level numerical representations. The FOL definition of SYMBXRL requires approximately 99.5% fewer nodes to model the behavior of Agent **A1** and 40% fewer nodes for Agent **A2**. Unlike *EXPLORA*, where the node count increases as the agent discovers new states and actions, SYMBXRL's KG is bounded by the FOL definition. This enables SYMBXRL to generate bounded KGs, regardless of action space size.

C. Understanding Agents' Behavior

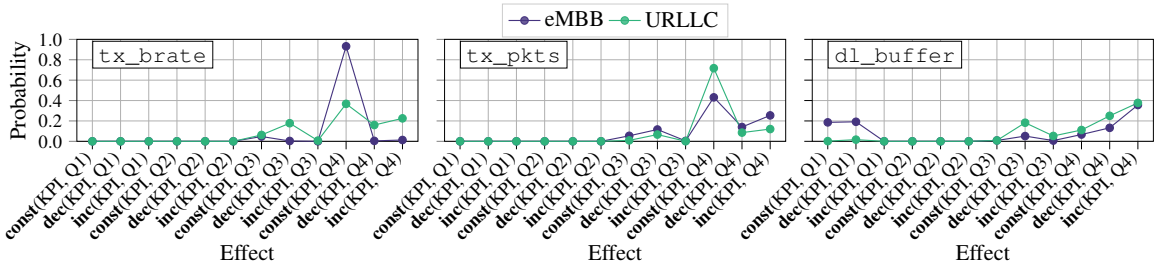
We analyze the explanations generated by SYMBXRL for agents **A1** and **A2**.

For agent **A1**, Fig. 2(a) and 2(b) illustrate the probability distribution of symbolic effects (changes in input KPIs due to the agent's decisions) for both eMBB and URLLC slices under TRF1 traffic. The figures provide insights into how each variant of the agent manages different slices under the same traffic profile. Key observations are:

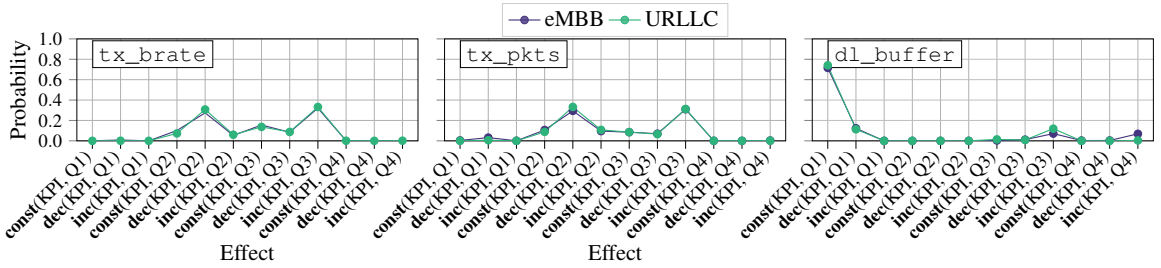
O1: High Throughput Maintenance for eMBB Slice: The eMBB agent effectively maintains high throughput (`tx_brate`) for the eMBB slice, as shown by the high probability of `tx_brate` remaining constant in Q4 (const(KPI, Q4)) and

TABLE II
COMPARING ACTION SPACE SIZES

Agents	**SYMBXRL**	***EXPLORA***
A1	$\lvert\mathcal{L}\rvert \cdot \lvert\text{Pred}_{\text{PRB}}\rvert \cdot \lvert\text{Cat}_{\text{PRB}}\rvert \cdot \lvert\text{toPolicy}\rvert$	$\binom{\text{PRB}-1}{2} \cdot \lvert\text{SP}\rvert^{\lvert\mathcal{L}\rvert}$
	Node Count = 180	*Node Count* = 31,752
A2	$\lvert\mathcal{G}\rvert \cdot \lvert\mathcal{Q}\rvert \cdot \lvert\text{Percentage}\rvert$	2^N
	Node Count = 75	*Node Count* = 128



(a) Slice eMBB



(b) Slice URLLC

Fig. 2. Probabilistic analysis of SymbXRL showing the probability distribution of decision effects for agent A1's variants under TRF1.

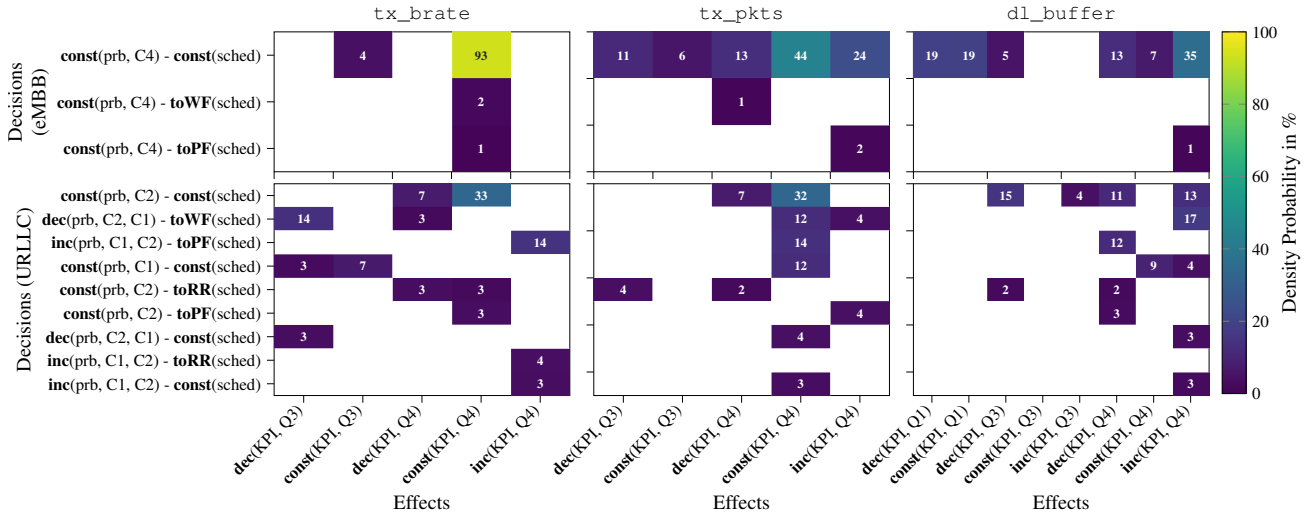


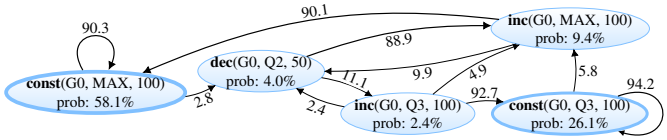
Fig. 3. Correlation density map of input KPIs and decisions for two variants of agent **A1** favoring eMBB (first row) or URLLC (second row) slices.

low probability of other changes.

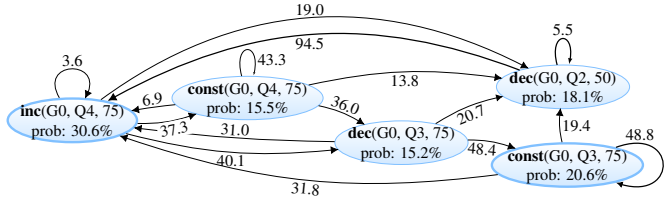
O2: Buffer Management in URLLC Slice: Both agents aim to keep buffer occupancy (`dl_buffer`) low in the URLLC slice, with the URLLC agent performing slightly better, indicated by a higher probability of const(KPI, Q1) for this agent versus the eMBB agent's counterpart.

O3: Takeaway: The performance of the URLLC agent is not significantly superior to the eMBB agent for the URLLC slice, suggesting that merely adjusting the reward function weights is insufficient for optimal agent action tuning.

To further analyze the behavior of the two variants of agent **A1**, we examine the correlation density map of its decisions and their effects on the eMBB slice (Fig. 3). In eMBB mode, the agent consistently allocates high PRB resources (const(PRB, C4), 93% density) and adapts to traffic variations mainly through scheduling policy adjustments. Conversely, in URLLC mode, the agent exhibits more diverse decision frequencies across PRB allocations and scheduling policies,



(a) Strategy: DQN

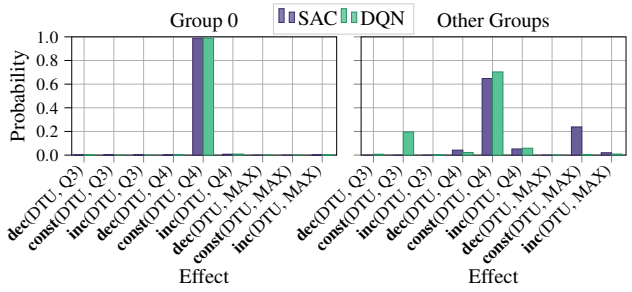


(b) Strategy: SAC

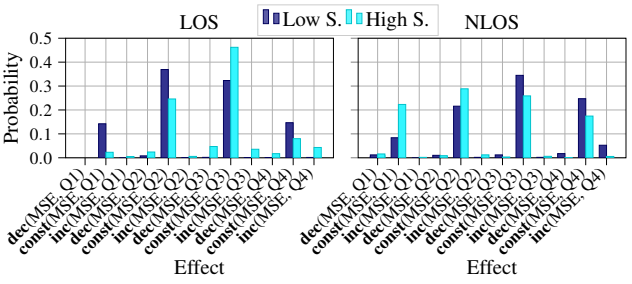
Fig. 4. KGs produced by SymbXRL for each implementation of agent **A2**.

indicating a less stable approach than the eMBB counterpart.

Let us now move to agent **A2**. Fig. 4 compares the KGs for SAC and DQN models in Group 0, revealing their learned



(a) Mean DTU



(b) Mean MSE

Fig. 5. Agent **A2**: Probability distribution of input KPIs (DTU and MSE)

strategies. The SAC agent (Fig. 4(b)) shows a stochastic decision-making pattern by distributing actions across various quartiles. One frequent transition path (Q2 $\rightarrow$ Q4 $\rightarrow$ Q3) indicates fluctuations between scheduling fewer and more users. In contrast, the DQN agent (Fig. 4(a)) demonstrates a more deterministic approach, with a dominant node representing a 58% probability of scheduling users in the MAX quartile and a 90% likelihood of maintaining this action. This suggests the DQN agent's deterministic policy is more efficient for Group 0 scheduling, consistently maximizing the number of scheduled users. SYMBXRL's KGs provide valuable insights into agent decision trends, highlighting the effectiveness of deterministic policies in this scenario.

Fig. 5 shows the symbolic effect analysis of agent **A2** using SYMBXRL, focusing on DTU and MSE. Fig. 5(a) depicts the probability distribution of mean DTU for SAC and DQN models. For Group 0, both agents consistently achieve mean DTU in Q4, indicating efficient scheduling. For other groups, the distribution varies: SAC favors Q4 or MAX quartiles, while DQN leans towards Q3 or Q4. Similarly, Fig. 5(b) presents the mean MSE distribution across different channel conditions (LoS and NLoS) and mobility profiles (low and high speed). In LoS scenarios, MSE is concentrated in Q2 and Q3 for both speed profiles. NLoS conditions show a broader distribution. High-speed scenarios exhibit mean MSE spread across all quartiles, indicating higher variability, while low-speed scenarios concentrate in Q3 and Q4. This analysis reveals the impact of user speed on MSE distribution, especially in NLoS conditions. While this may seem obvious to experts, SYMBXRL is crucial for explaining how these effects unfold within the DRL agent's decision process.

D. Improving Agent's Behavior

This section analyzes the performance of IAS (module ❸ in Fig. 1), focusing on agent **A2**. We present quantitative analyses for three key use cases described in §III-D to demonstrate SYMBXRL's capability to enhance agent performance in complex network scenarios.

1) Reward Maximization

As discussed in Section III-D, SYMBXRL IAS can automatically improve agent behavior to maximize cumulative reward (2). We apply reward maximization IAS to agent **A2** using the NLoS dataset with high-speed mobility over 10 test episodes. Fig. 6 illustrates how this functionality affects the probability distribution of symbolic actions for the target agent. We observe that:

- *Group 0*: Before IAS, the agent predominantly scheduled either 25% or 75% of users. After IAS, there is a notable increase in scheduling 75% or 100% of users, with a substantial reduction in scheduling 25% of users.
- *Group 1*: After IAS, the agent shows a 13.2% increase in not scheduling any users from this group, indicating a strategy to avoid inter-group interference.

Analysis of changes before and after applying IAS suggests that IAS's reward maximization mode alters agent behavior towards more efficient resource use, reducing inter-group interference while maximizing utilization for Group 0.

Fig. 7 shows how these changes translate to cumulative reward (2) improvement. Here, we benchmark SYMBXRL against METIS [15], which uses a decision tree-based approach to improve RL agent performance. The figure reports the cumulative reward attained by the baseline **A2** agent, by the same **A2** agent improved by Metis, and by agent **A2** improved by SYMBXRL's IAS.

We observe that SYMBXRL achieves a median 11.76% improvement in cumulative reward compared to the basic agent, whereas Metis only provides a 0.07% gain. This highlights SYMBXRL's ability to leverage symbolic knowledge effectively to identifying suboptimal decisions and replacing them with better alternatives from similar past states. Metis's marginal improvement suggests limitations in fully characterizing and exploiting agent knowledge.

The right portion of Fig. 7 shows the impact of initiating IAS at different points during the test episode. Starting late yields diminishing performance gains compared to starting early. While a late start can leverage more knowledge, it can only enact changes for a limited time, ultimately determining the lower cumulative reward. However, a late start yields reduced variance in performance improvements, due to more refined action replacements resulting from more knowledge gathered. These results show SYMBXRL's significant potential to enhance RL agent performance through knowledge-based action steering, outperforming other existing explainers and being effective even after a minimal amount of data has been collected about the target agent's behavior.

2) Decision Conditioning

IAS can enforce high-level intents for flexible control of agent behavior without excluding reward maximization, as discussed in Section III-D. This capability is crucial for network management scenarios with specific operational requirements like prioritization. We use SYMBXRL to prohibit

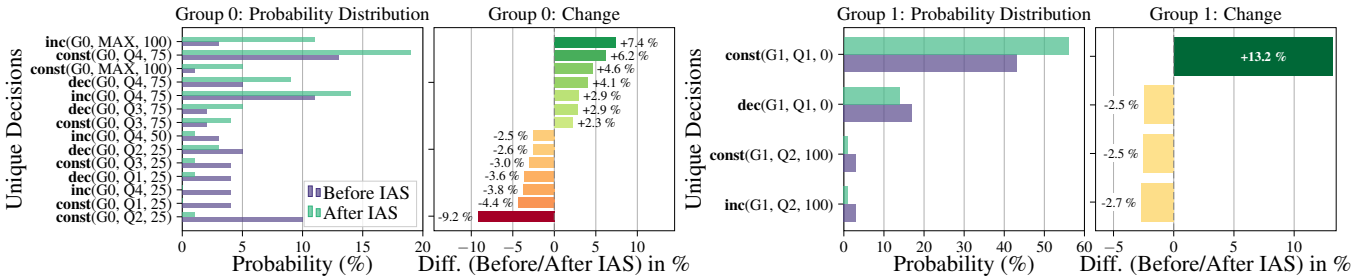


Fig. 6. Comparison of agent **A2** decisions before and after applying Action Steering IAS, showing the effect of IAS on agent behavior.

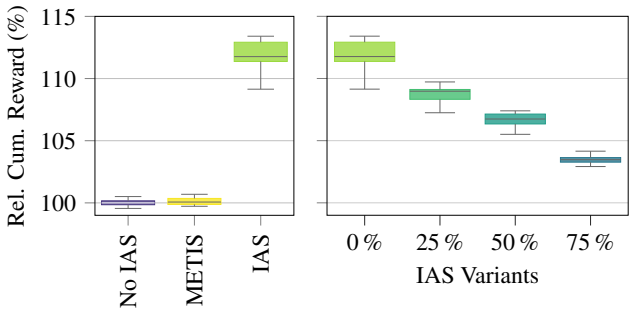
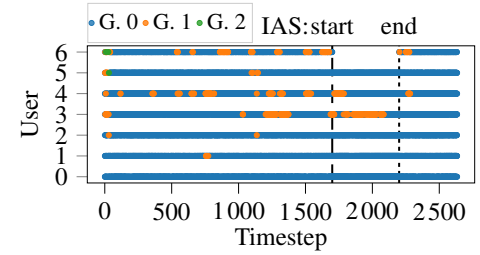
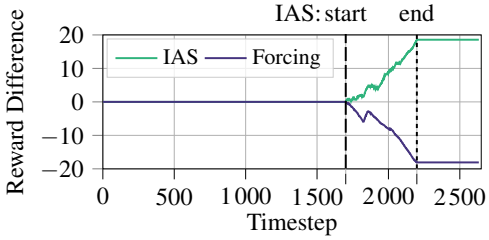


Fig. 7. Comparing relative cumulative reward of different benchmarks for Agent **A2**. "No IAS" indicates the basic agent, METIS shows improvement with [15], and the other boxplots show improvements with SYMBXRL by starting IAS at different fractions of the test set (0%, 25%, 50%, and 75%).



(a) Decision Conditioning: **notSchedule**$(a_t, 6)$



(b) Reward improvement with IAS versus direct forcing

Fig. 8. Intent-based operation of enforcing not scheduling a specific user with **A2**. IAS allows the agent to achieve high reward compared to blindly forcing actions.

agent **A2** from scheduling user 6 for about 500 timesteps ($t \in [1700, 2200]$), achieved by conditioning the agent's behavior using the FOL term **notSchedule**$(a_t, 6)$. To apply this constraint, IAS uses both KG and DB to find previously seen states similar to the current state where the agent's decision did not include scheduling user 6. If multiple timesteps satisfy the condition, we choose the action that provided the best reward when applied previously.

The outcome is shown in Fig. 8(a), where user 6 is not scheduled in any group (denoted by different colors) in the target interval. Fig. 8(b) compares the performance of IAS and direct action forcing i.e., explicitly removing user 6 from the agent's decision without considering any consequences of this change.

The cumulative reward differences attained by each technique with respect to the original agent **A2** show that:

- IAS enhances the reward over time while applying constraints which indicates that IAS can choose actions with the highest reward from the KG for replacement.
- Direct forcing leads to a rapid decline in cumulative reward compared to the baseline, indicating performance degradation. This is because removing one of the scheduled users reduces the spectral efficiency factor of the reward function.

These results demonstrate SYMBXRL's superiority in applying constraint. By leveraging symbolic knowledge, IAS adjusts the agent's actions to accommodate constraints while optimizing overall performance. This shows SYMBXRL's potential to enhance the flexibility and efficiency of RL agents in real-world telecommunication systems, balancing operational requirements with system optimization.

3) Accelerated Learning

Fig. 9 demonstrates IAS's ability to reduce training time for RL agents. We compare agent **A2**'s performance at three training Checkpoints (CHKPs): the final model, used until now in the evaluation, is trained for 205 episodes (CHKP 205, the baseline). Next, we create variants with fewer training episodes, stopping the training earlier at episodes 68 and 115. Note that all these CHKPs are in the stable region of cumulative reward upon training.

The left portion of Fig. 9 shows the cumulative reward difference of CHKP 68 and CHKP 115 relative to the baseline CHKP 205 without IAS. As expected, the cumulative reward without IAS is lower for CHKP 68 and CHKP 115 compared to CHKP 205. The right portion of the figure illustrates the impact of applying IAS after the first 25% of the testing episode (denoted by the dashed line in the figure):

- Before IAS for accelerated learning is enabled, the agents trained up to CHKPs 68 and 115 yield a lower cumulative reward compared to the baseline, similar to the left plot.
- Upon IAS activation, the agents trained up to CHKPs 68 and 115 start leveraging the KG through IAS to replace low-reward actions with higher-reward ones, as detailed in Section IV-D1. This effectively boosts cumulative reward

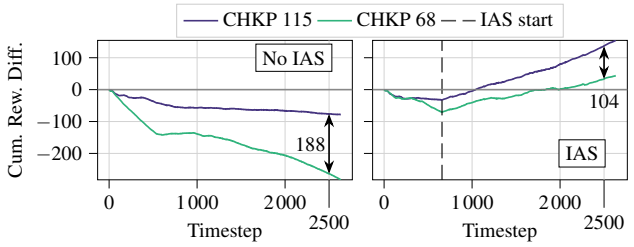


Fig. 9. Relative cumulative reward of CHKPs 115 and 68 compared to the final model (CHKP 205) without IAS (left) and with IAS enabled after the first 25% of the test episode (right).

with respect to the baseline agent and eventually surpasses it by the end of the episode.

- Using IAS by the end of the test episode reduces the performance gap between CHKP 115 and 68 by 45% (from 188 to 104), indicating more consistent performance improvement across different training stages.

These results show SYMBXRL's ability to leverage an agent's acquired knowledge during its operation effectively, even at earlier training stages. IAS's accelerated learning makes it possible for an agent trained up to CHKP 68 to perform as well as an agent trained up to CHKP 205, with a 66.7% training time reduction, equivalent to 13 hours and 41 minutes. Both are trained on an NVIDIA A100 40 GB GPU cluster.

This suggests that DRL agents accumulate valuable knowledge early in training but may not fully exploit it without further assistance until later episodes. The IAS's *accelerated learning* strategy provides a mechanism to fully exploit this latent knowledge, thereby providing high performance.

4) Summary of Insights

The analyses of *reward maximization*, *decision conditioning*, and *accelerated learning* collectively demonstrate the flexibility and effectiveness of SYMBXRL's IAS approach. By leveraging symbolic knowledge, IAS not only improves overall agent performance but also enables precise control over agent behavior with flexible intents and makes early stopping of the training process viable. These capabilities are particularly valuable in dynamic network environments, where adaptive management and efficient resource utilization are crucial.

V. RELATED WORK

Relevant to our work are studies on the use of DRL for mobile networking, XAI at large, and XRL techniques, especially those applicable to mobile networking.

DRL in Mobile Networking. **DRL in Mobile Networking**. DRL models have gained prominence for handling high-dimensional data in dynamic environments, excelling at rapid parameter reconfiguration during exploitation [36]. Unlike DTs which suit rule-based configurations using historical data (e.g., Auric for Base Stations (BSs) [13], Configanator for content-delivery [37]), DRL applications span diverse areas: dynamic spectrum access [8], joint user scheduling with antenna allocation [38] and mmWave configuration [39], resource management [40], network slicing [19], [41], mobility management [42], [43], service coverage [44], and anomaly detection [43], [45].

XAI For Mobile Networks. Future 6G networks embrace the vision for native, explainable network intelligence. Seminal works [46], [47] motivate the need for XAI and stress that the lack of explainability may lead to poor AI/ML model design. This has been proved detrimental in the presence of adversarial attacks [48]. All areas where AI is applied to mobile networking tasks can benefit from explainability. These include physical and MAC layer design, network security, mobility management, and localization [49].

Related Works on XRL. The analysis of transitions between actions is not new [50], [51]. HIGHLIGHTS [51] produces summaries of agent behavior for general audience, while [50] focuses on expert explanations by highlighting influential transitions whose removal significantly affects rewards. However, this approach fails with autoencoders masking real input as in Agent **A1**. DeepSynth [52] reveals patterns in reward sequences to reduce exploration time. PIRL [53] generates interpretable and verifiable policies using domain-specific languages, but designing primitives per RAN scenario is inefficient unlike SYMBXRL's symbolic representation. The work [54] uses reward decomposition with Large Language Model (LLM) for text explanations. EXPLORA [14] is closest to our work but synthesizes network-aware explanations using attributed graphs rather than symbolic AI like SYMBXRL.

VI. CONCLUSIONS

In this paper, we proposed SYMBXRL, a new XRL technique that generates explanations for DRL agents. SYMBXRL leverages symbolic AI to represent concepts and their relationships, coupled with logical reasoning. In this way, SYMBXRL provides a competitive advantage over existing explainers because it clarifies how DRL agents arrive at their decisions in an understandable manner to the human observer.

We validated our approach extensively with agents trained on both real-world datasets and datasets generated through Colosseum. We demonstrated that SYMBXRL not only improves the clarity of the explanations but also enables performance improvements for the agents by using the knowledge generated by the explanations. Specifically, we show that intent-based action steering IAS achieves a median 12% improvement in cumulative reward over the baseline DRL solution. The authors have provided public access to their code and/or data at https://github.com/RAINet-Lab/symbxrl

ACKNOWLEDGMENT

This work is partially supported by bRAIN project PID2021-128250NB-I00 funded by MCIN/ AEI /10.13039/501100011033/ and the European Union ERDF "A way of making Europe"; by Spanish Ministry of Economic Affairs and Digital Transformation, European Union NextGeneration-EU/PRTR projects MAP-6G TSI-063000-2021-63, RISC-6G TSI-063000-2021-59 and AEON-ZERO TSI-063000-2021-52; C. Fiandrino is a Ramón y Cajal awardee (RYC2022-036375-I), funded by MCIU/AEI/10.13039/501100011033 and the ESF+.

REFERENCES

[1] W. Saad, M. Bennis *et al.*, "A vision of 6G wireless systems: Applications, trends, technologies, and open research problems," *IEEE network*, vol. 34, no. 3, pp. 134–142, 2019.

[2] Ericsson, "Mobility Report, June 2024. Technical Report." 2024, Accessed on 07/08/2024: https://rb.gy/hzpqoi.

[3] K. B. Letaief, W. Chen *et al.*, "The roadmap to 6G: AI empowered wireless networks," *IEEE communications magazine*, vol. 57, no. 8, pp. 84–90, 2019.

[4] G. Wikström, J. Peisa *et al.*, "Challenges and technologies for 6G," in *2020 2nd 6G wireless summit (6G SUMMIT)*. IEEE, 2020, pp. 1–5.

[5] N. C. Luong, D. T. Hoang *et al.*, "Applications of deep reinforcement learning in communications and networking: A survey," *IEEE communications surveys & Tutorials*, vol. 21, no. 4, pp. 3133–3174, 2019.

[6] Y. Wei, F. R. Yu *et al.*, "Joint optimization of caching, computing, and radio resources for fog-enabled iot using natural actor–critic deep reinforcement learning," *IEEE Internet of Things Journal*, vol. 6, no. 2, pp. 2061–2073, 2018.

[7] D. Bega, M. Gramaglia *et al.*, "AI-based autonomous control, management, and orchestration in 5G: From standards to algorithms," *IEEE Network*, vol. 34, no. 6, pp. 14–20, 2020.

[8] O. Naparstek and K. Cohen, "Deep multi-user reinforcement learning for distributed dynamic spectrum access," *IEEE transactions on wireless communications*, vol. 18, no. 1, pp. 310–323, 2018.

[9] Y. Kim and H. Lim, "Multi-agent reinforcement learning-based resource management for end-to-end network slicing," *IEEE Access*, vol. 9, pp. 56 178–56 190, 2021.

[10] Y. Sun, J. Liu *et al.*, "When machine learning meets privacy in 6G: A survey," *IEEE Communications Surveys & Tutorials*, vol. 22, no. 4, pp. 2694–2724, 2020.

[11] M. M. Morovati, F. Tambon *et al.*, "Common challenges of deep reinforcement learning applications development: an empirical study," *Empirical Software Engineering*, vol. 29, no. 4, p. 95, 2024.

[12] A. Heuillet, F. Couthouis *et al.*, "Explainability in deep reinforcement learning," *Knowledge-Based Systems*, vol. 214, p. 106685, 2021.

[13] A. Mahimkar, A. Sivakumar *et al.*, "Auric: Using data-driven recommendation to automatically generate cellular configuration," in *Proc. of the ACM SIGCOMM*, 2021, p. 807–820.

[14] C. Fiandrino, L. Bonati *et al.*, "EXPLORA: AI/ML explainability for the open RAN," *Proc. of the ACM on Networking*, vol. 1, no. CoNEXT3, pp. 1–26, 2023.

[15] Z. Meng, M. Wang *et al.*, "Interpreting deep learning-based networking systems," in *Proc. of ACM SIGCOMM*, 2020, pp. 154–171.

[16] S. M. Lundberg and S.-I. Lee, "A unified approach to interpreting model predictions," in *Proc. of NIPS*, 2017, pp. 4768–4777.

[17] M. T. Ribeiro, S. Singh *et al.*, ""Why Should I Trust You?": Explaining the predictions of any classifier," in *Proc. of ACM SIGKDD*, 2016, p. 1135–1144.

[18] S. Milani, N. Topin *et al.*, "Explainable reinforcement learning: A survey and comparative review," *ACM Computing Surveys*, vol. 56, no. 7, pp. 1–36, 2024.

[19] M. Polese, L. Bonati *et al.*, "ColO-RAN: Developing machine learning-based xApps for open RAN closed-loop control on programmable experimental platforms," *IEEE Transactions on Mobile Computing*, vol. 22, no. 10, pp. 5787–5800, 2022.

[20] Q. An, S. Segarra *et al.*, "A deep reinforcement learning-based resource scheduler for massive mimo networks," *IEEE Transactions on Machine Learning in Communications and Networking*, 2023.

[21] V. Mnih, K. Kavukcuoglu *et al.*, "Human-level control through deep reinforcement learning," *nature*, vol. 518, no. 7540, pp. 529–533, 2015.

[22] O. Vinyals, I. Babuschkin *et al.*, "Grandmaster level in starcraft ii using multi-agent reinforcement learning," *Nature*, vol. 575, no. 7782, pp. 350–354, 2019.

[23] T. Haarnoja, B. Moran *et al.*, "Learning agile soccer skills for a bipedal robot with deep reinforcement learning," *arXiv:2304.13653*, 2023.

[24] D. Gunning and D. Aha, "Darpa's explainable artificial intelligence (xai) program," *AI magazine*, vol. 40, no. 2, pp. 44–58, 2019.

[25] E. Puiutta and E. M. S. P. Veith, "Explainable reinforcement learning: A survey," in *Machine Learning and Knowledge Extraction*, 2020, pp. 77–95.

[26] Z. Juozapaitis, A. Koul *et al.*, "Explainable reinforcement learning via reward decomposition," in *IJCAI/ECAI Workshop on XAI*, 2019.

[27] R. Dazeley and *et al.*, "Explainable reinforcement learning for broad-XAI: a conceptual framework and survey," *arXiv:2108.09003*, 2021.

[28] J. Y. Halpern and V. Weissman, "Using first-order logic to reason about policies," *ACM Transactions on Information and System Security*, vol. 11, no. 4, pp. 1–41, 2008.

[29] Z. Ma, Y. Zhuang *et al.*, "Learning symbolic rules for interpretable deep reinforcement learning," *arXiv:2103.08228*, 2021.

[30] R. Jain and I. Chlamtac, "The p2 algorithm for dynamic calculation of quantiles and histograms without storing observations," *Communications of the ACM*, vol. 28, no. 10, pp. 1076–1085, 1985.

[31] M. Alshiekh, R. Bloem *et al.*, "Safe reinforcement learning via shielding," *Proc. of AAAI*, vol. 32, no. 1, Apr. 2018.

[32] J. Queeney, Y. Paschalidis *et al.*, "Generalized proximal policy optimization with sample reuse," *Proc. of NeurIPS*, vol. 34, pp. 11 909–11 919, 2021.

[33] Q. An, S. Segarra *et al.*, "A deep reinforcement learning-based resource scheduler for massive MIMO networks," *IEEE Transactions on Machine Learning in Communications and Networking*, vol. 1, pp. 242–257, 2023.

[34] T. Haarnoja, A. Zhou *et al.*, "Soft actor-critic algorithms and applications," *arXiv:1812.05905*, 2018.

[35] V. Mnih, K. Kavukcuoglu *et al.*, "Playing atari with deep reinforcement learning," *arXiv:1312.5602*, 2013.

[36] C. Ge, Z. Ge *et al.*, "Chroma: Learning and using network contexts to reinforce performance improving configurations," in *Proc. of ACM MobiCom*, 2023.

[37] U. Naseer and T. A. Benson, "Configanator: A data-driven approach to improving CDN performance," in *Proc. of USENIX NSDI*, Apr 2022, pp. 1135–1158.

[38] M. Naeem, A. Coronato *et al.*, "Optimal user scheduling in multi antenna system using multi agent reinforcement learning," *Sensors*, vol. 22, no. 21, p. 8278, 2022.

[39] Y. Zhang and R. W. Heath, "Reinforcement learning-based joint user scheduling and link configuration in millimeter-wave networks," *IEEE Transactions on Wireless Communications*, 2022.

[40] J. A. Ayala-Romero, A. Garcia-Saavedra *et al.*, "VrAIn: A deep learning approach tailoring computing and radio resources in virtualized RANs," in *Proc. of ACM MobiCom*, 2019.

[41] J. J. Alcaraz, F. Losilla *et al.*, "Model-based reinforcement learning with kernels for resource allocation in RAN slices," *IEEE Transactions on Wireless Communications*, vol. 22, no. 1, pp. 486–501, 2023.

[42] T. M. Ho and K.-K. Nguyen, "Joint Server Selection, Cooperative Offloading and Handover in Multi-Access Edge Computing Wireless Network: A Deep Reinforcement Learning Approach," *IEEE Transactions on Mobile Computing*, vol. 21, no. 7, pp. 2421–2435, July 2022.

[43] X. Ma and W. Shi, "Aesmote: Adversarial reinforcement learning with smote for anomaly detection," *IEEE Transactions on Network Science and Engineering*, vol. 8, no. 2, pp. 943–956, 2020.

[44] Y. Yang, Y. Li *et al.*, "Decco: Deep-learning enabled coverage and capacity optimization for massive mimo systems," *IEEE Access*, vol. 6, pp. 23 361–23 371, 2018.

[45] G. Caminero, M. Lopez-Martin *et al.*, "Adversarial environment reinforcement learning algorithm for intrusion detection," *Computer Networks*, vol. 159, pp. 96–109, 2019.

[46] W. Guo, "Explainable artificial intelligence for 6G: Improving trust between human and machine," *IEEE Communications Magazine*, vol. 58, no. 6, pp. 39–45, 2020.

[47] C. Li, W. Guo *et al.*, "Trustworthy deep learning in 6G-enabled mass autonomy: From concept to quality-of-trust key performance indicators," *IEEE Vehicular Technology Magazine*, vol. 15, no. 4, pp. 112–121, 2020.

[48] S. Moghadas Gholian, C. Fiandrino *et al.*, "Spotting deep neural network vulnerabilities in mobile traffic forecasting with an explainable AI lens," in *IEEE INFOCOM*, 2023.

[49] U. Challita, H. Ryden *et al.*, "When machine learning meets wireless cellular networks: Deployment, challenges, and applications," *IEEE Communications Magazine*, vol. 58, no. 6, pp. 12–18, 2020.

[50] O. Gottesman, J. Futoma *et al.*, "Interpretable off-policy evaluation in reinforcement learning by highlighting influential transitions," in *Proc. of ICML*, vol. 119, Jul 2020, pp. 3658–3667.

[51] D. Amir and O. Amir, "Highlights: Summarizing agent behavior to people," in *Proc. of AAMS*, 2018, pp. 1168–1176.

[52] M. Hasanbeig, N. Yogananda Jeppu *et al.*, "DeepSynth: Automata synthesis for automatic task segmentation in deep reinforcement learning," *Proc. of AAAI*, vol. 35, no. 9, pp. 7647–7656, May 2021.

[53] A. Verma, V. Murali *et al.*, "Programmatically interpretable reinforcement learning," in *Proc. of ICML*, vol. 80, 2018, pp. 5045–5054.
[54] M. Ameur, B. Brik *et al.*, "Leveraging LLMs to explain DRL decisions for transparent 6G network slicing," in *Proc. of IEEE NetSoft*, 2024, pp. 204–212.