Mälardalen University
School of Innovation Design and Engineering
Västerås, Sweden

Master of Science in Computer Science DVA504 - 120 credits

EXPLAINABLE AI FOR MULTI-AGENT CONTROL PROBLEM

Hanna Prokopova
hpa22002@student.mdh.se

Examiner: Shahina Begum
Mälardalen University, Västerås, Sweden

Supervisor: Ning Xiong
Mälardalen University, Västerås, Sweden

Company supervisors: Ahmad Terra, Franco Ruggeri
Ericsson, Stockholm

May 17, 2023

Abstract

This report presents research on the application of policy explanation techniques in the context of coordinated reinforcement learning (CRL) for mobile network optimization. The goal was to improve the interpretability and comprehensibility of decision-making processes in multi-agent environments, with a particular focus on the Remote Antenna Tilt (RET) problem.

The task has been formulated as providing insight into the extension of policy explanation in a cooperative multi-agent reinforcement learning (MARL) environment, evaluating its applicability to a network use case, and revealing interactions between agents in such a setting. The results contribute to a better understanding of decision-making processes, dynamics of coordination, and aspects of interpretation in complex multi-agent systems, in particular in the context of mobile network optimization.

This research is motivated by the need for transparency, accountability, and trust in AI-driven decision-making processes, especially in critical applications such as mobile networks. The study aimed to bridge the gap between the confusing behavior of many agents and the need for human-understandable explanations.

The approach involved training a CRL agent and using a policy explanation method to generate explanations based on the observations and actions taken by the agent. The outcomes demonstrated the effectiveness of the policy explanation method in providing clear and robust interpretations in both single-agent and multi-agent environments.

Furthermore, analysis of CRL Q-value functions revealed consistent patterns in some agents' preferences and avoidance of certain interactions with neighboring agents. This insight allows for a better understanding of coordination dynamics in mobile network optimization.

In conclusion, this study demonstrates the successful application of policy explanation methods in CRL to optimize mobile networks. Combining CRL and policy explanations improves the interpretation of agent behavior and increases accountability. The study contributes to the expansion of the explainable AI field and lays the foundation for future research on the optimization of complex multi-agent systems.

Table of Contents

1. Introduction

Artificial intelligence (AI) continuously increases in importance for users as well as for the industry. Machine learning and its subtype reinforcement learning (RL) are used for numerous applications:

- in games like chess, Go, and Dota, where agents can outperform human world champions;
- in autonomous driving – RL is used in autonomous vehicles to learn how to navigate complex environments and make decisions in real-time.
- in recommendation systems – RL is used by companies to provide tailored recommendations to maximize click rate and relevancy.
- in mobile connectivity - RL can be adopted to solve the remote antenna tilt problem (RET) and safely and effectively improve signal coverage and quality for mobile network users.

For some of these approaches, it is beneficial to employ more than one decision-making agent. For example, when the area of interest is distributed spatially, or agents' tasks differ.However, since transparent models are not always scalable and accurate AI is a "black box" in most cases, which complicates troubleshooting, and introduces human bias from the datasets. The use of RL in multi-agent systems introduces challenges related to transparency, scalability, and ethical decision-making.

To address these challenges, there is a growing need for explainable reinforcement learning (XRL) techniques. XRL aims to provide insights into the decision-making process of RL agents, enabling better understanding, troubleshooting, and testing of multi-agent systems. By explaining the strategies and policies employed by collaborative agents, XRL can narrow the gap between computational performance and comprehensibility, facilitating the development of more transparent and ethically justified strategies.

One goal of the XAI methods in cooperative RL tasks is to understand the decision-making process, cooperation and possible interference or obstruction among the agents. It can be applied for troubleshooting and testing of multi-agent systems in different cooperation scenarios.

The research focuses on solving the multi-agent control problem in the context of mobile connectivity using the means of XRL. Specifically, the research aims to summarize the policies of collaborative agents and evaluate the policy explanation methods for describing the strategies the agents apply to solve the problem of maximal mobile connectivity. In particular, perform the assessment of the role each agent plays in reaching the common goal. The policy explanation is tested on the mobile connectivity use case where each antenna on the grid corresponds to one RL agent intending to maximize the quality of signal and minimize interference for the user devices.

The remainder of this report presents an in-depth analysis of the research conducted, including an overview of previous work in the field, the methodology employed, the motivation behind the study, and the most important results and contributions. This work aims to provide a comprehensive understanding of the study and its implications in the context of multi-agent control and explainable reinforcement learning.

2. Background

Artificial Intelligence (AI) - is intelligence (e.g. the capacity to perceive, process, transform, produce information, learn from experience, etc.) displayed by non-living structures like programs and machines developed by humans [1]. AI applications include but are not limited to autonomous vehicles, recommendation systems, automated control systems, natural language processing, and generative models.

Machine Learning (ML) is one of the essential AI components. It involves the development of models and algorithms that enable computers to learn from data, without being explicitly programmed. Computer programs can learn patterns, and relationships, and make predictions based on provided examples, instead of following specific instructions. ML includes supervised learning, unsupervised learning, and reinforcement learning.

Reinforcement learning (RL) is a field of ML where the agent learns by trying to maximize the utility of the action to reach the goal. Agent in RL is an entity that learns from the environment by observing its state, taking actions, and receiving rewards or punishments depending on whether the actions taken will lead it closer to the goal and to which degree. The goal of a single agent is to learn a policy – a strategy that emerges from agent interaction with the environment. RL policy is used to increase the sum of expected rewards as much as possible over time. The policy maps the current state of the environment to a probability distribution over possible actions. Once the optimal policy is learned, the agent can use it to select actions that maximize the expected reward in a given state of the environment.

Off/On policy learning Off-policy and on-policy learning are two approaches to reinforcement learning. Off-policy learning involves learning from a policy that is different from the one that is being improved, while on-policy learning involves improving the same policy that is being used to interact with the environment. Off-policy learning is often used in situations where it is difficult or risky to use the current policy to collect data.

Q learning Q-learning is a type of off-policy learning algorithm used in reinforcement learning. The algorithm uses an exploration policy, such as epsilon-greedy, to choose actions during training, while estimating the value function for the optimal policy. It relies on a Q-table to keep track of the expected reward for each action the agent can take in each state of the environment. Q learning is a model-free approach, meaning that it does not require knowledge of the underlying dynamics of the environment.

The Bellman equation is used to iteratively update the values of states in dynamic programming algorithms like value iteration and policy iteration, and it provides the foundation for estimating optimal value functions in Q-learning. It relates the value of a state to the expected sum of future rewards that can be obtained by following a particular policy. The equation expresses the principle of optimality, stating that the value of a state is equal to the immediate reward obtained in that state plus the discounted value of the expected future rewards from the next state. It can be represented as:

$$V(s) = R(s) + \gamma * \sum [P(s, a, s') * V(s')]$$

Where:

- $V(s)$ is the value of state s,
- $R(s)$ is the immediate reward obtained in state s,
- γ (gamma) is the discount factor that determines the importance of future rewards,
- $P(s, a, s')$ is the transition probability from state s to state s' given action a,
- $V(s')$ is the value of the next state s' after taking action a.

Markov Decision Process (MDP) is a mathematical framework used to model decision-making problems in the field of reinforcement learning. It is a discrete-time stochastic control process where an agent interacts with an environment by taking actions and receiving rewards. An MDP is characterized by a set of states, a set of actions, transition probabilities that describe the

likelihood of transitioning from one state to another based on the chosen action, and immediate rewards associated with state-action pairs.

Deep Q-Network (DQN) DQN is a deep reinforcement learning algorithm that combines the power of neural networks with the Q-learning algorithm to approximate the optimal action-value function in a Markov decision process (MDP). In DQN, the goal is to learn an optimal policy that maximizes the expected cumulative reward over time. The algorithm leverages a deep neural network, known as the Q-network, to approximate the action-value function (Q-function). The Q-function estimates the expected cumulative reward for taking a specific action in a given state.

The algorithm updates the Q-values based on the Bellman equation and uses a target network to enhance stability during training. DQN has been successful in various domains and is known for its effective learning in high-dimensional state spaces.

In **multi-agent reinforcement learning (MARL)** two or more agents share the same environment. MARL interaction types include cooperative, competitive, and mixed strategies (including self-interested). Credit assignment problem occurs in partially observable MARL setting with joint reward, where the agent cannot know the impact of its actions towards reaching the common goal.

Credit assignment is often used in machine learning to determine which actions led to a positive or negative reward signal. This information is then used to update the agent's policy, which informs its future actions. It may be difficult to accurately assign credit or punishment for a particular outcome because the actions of the agents are influenced by a complex sequence of their own and their neighbors' decisions and environmental events.

Self-interested RL agents are designed to maximize their own rewards, without considering the collective consequences of taken actions. Such selfish behavior can diversify the system's policy but it also can result in suboptimal outcomes if the agents hinder overall progress. Self-interested RL agents are commonly used in game theory, economics, and social simulation to study the behavior of individuals in competitive or strategic settings.

Cooperative agents, on the other hand, work together to achieve a common goal. These agents learn to select actions that not only maximize their own rewards but also contribute to the collective reward of the group. Cooperative agents may use communication and coordination to achieve their objectives, and they may share rewards equally among the group members. Examples of their usage include multi-agent robotics, traffic control systems, networks, etc.

Competitive agents, like self-interested agents, are motivated by maximizing their own rewards. However, they also take into account the impact of their actions on other agents, and they actively try to minimize the rewards of other agents. Competitive agents may use adversarial strategies, such as deception and bluffing, to gain an advantage over their opponents. Competitive agents often find applications in the game industry and cybersecurity.

2.1. Explainable AI

Explainable AI (XAI) has emerged with the development of AI as a result of the learning structures becoming more complex and thus less clear. XAI is concerned with eliminating bias and possible errors in decision criteria, boosting model performance, and understanding as well as accountability. In particular, Reinforcement Learning algorithms are often viewed as black boxes. However, since they become more and more relevant in making important decisions, explainable reinforcement learning (XRL) approach is needed for people to identify the underlying reasons for actions taken by such agents together with any implicit biases or judgment errors. Most XRL approaches focus on machine learning experts, leaving a gap in explainability for users and professionals in other fields [2]. Multi-agent environments where agents act cooperatively are even harder to access for the lack of certainty which of the many agents have influenced the final results and the way they interact with each other and the environment in the process.

2.2. Multi-agent control problem

Multi-agent control problems are situations in which multiple agents, each with their own objectives and decision-making processes, must cooperate or compete to achieve a common goal. These problems arise in many fields, including robotics, economics, and social sciences. Solutions include:

- Reinforcement Learning. Some of the popular RL algorithms for multi-agent control problems include Q-Learning, Actor-Critic methods, and Deep RL [3].
- Game Theory. Game theory provides a useful framework for modeling and analyzing strategic interactions between decision-makers. It can be used to study competitive and cooperative scenarios and is often used to analyze the behavior of rational agents in multi-agent control problems[4].
- Distributed Optimization: Distributed optimization is a technique for solving problems where the objective function is decomposable into several smaller functions or subproblems which are then solved locally by the agents[5].
- Hybrid Approaches. Work by employing multiple techniques to solve multi-agent control problems. For example, a hybrid approach can combine RL and game theory to solve a multi-agent control problem where the agents interact in a dynamic environment [6].

Markov Games A Markov game is a type of game in which multiple agents make decisions in a shared environment that has a Markov property, meaning that the current state of the environment is sufficient to determine the probability distribution of future states. Markov games are commonly used in multi-agent reinforcement learning.

Cooperative Games Cooperation games are a type of game in which the players' interests are aligned, such that the players can achieve higher payoffs by working together than by acting selfishly. Cooperation games are an important area of study in game theory and have applications in fields such as economics, political science, and psychology.

Remote Electrical Tilt (RET) problem is a multi-agent control problem from the telecom domain. It focuses on optimizing the direction of signal transmission of cellular tower antenna. The solution must maximize the quality of user service (SINR, RSRP, bandwidth, throughput, etc.) and minimize coverage expenses and interferences. RET can be described as a cooperative problem where the goal is to optimize the holictic performance of telecom services.

SINR stands for Signal-to-Interference-plus-Noise Ratio and is a metric used in telecommunications to measure the quality of a wireless signal. It quantifies the ratio of the desired signal power to the combined interference and background noise power in a given communication channel.

In a wireless communication system, multiple signals are present, including the desired signal, interference from other sources, and background noise. The SINR provides an indication of the signal strength relative to the interfering signals and noise in the environment.

The SINR is typically expressed in decibels (dB) and is calculated using the following formula:

$$SINR = 10 * log_{10}(\frac{P_{signal}}{P_{interference} + P_{noise}})$$

Where:

- P_{signal} is the power of the desired signal,
- $P_{interference}$ is the combined power of all interfering signals,
- P_{noise} is the power of background noise.

A higher SINR value indicates a stronger desired signal relative to interference and noise, resulting in better signal quality and improved communication performance.

RSRP stands for Reference Signal Received Power. It is a measurement used to assess the power level of the reference signals transmitted by a base station (eNodeB or eNB) in a cellular network. RSRP is typically reported by User Equipment (UE) devices, such as mobile phones, to indicate the strength of the signal received from the serving base station.

RSRP is an important metric in evaluating the signal quality and coverage of a cellular network. It represents the received power of the reference signal and provides an indication of the signal strength at the UE. A higher RSRP value generally implies a stronger and more reliable connection between the UE and the base station.

RSRP is measured in dBm (decibel-milliwatts), which is a logarithmic unit used to express power levels. The value is typically reported as a negative number, with a higher absolute value

indicating a stronger signal. For example, an RSRP value of -70 dBm represents a stronger signal than an RSRP value of -90 dBm.

2.3. Coordinated RL

Coordinated RL approach for mobile networks proposed by [7] introduces a coordination graph representation of the cellular network, where each cell is a node and edges are assigned between cells that can influence each other's signal reception. It proposes using deep Q-learning to train agents that control the antenna tilt angles to maximize the Signal-to-Interference-plus-Noise Ratio (SINR) for user devices.

The research presents a custom telecom environment developed using a simulation program, which incorporates propagation models and traffic calculations. Multiple experiments are conducted to evaluate different antenna configurations and observe the impact on network performance metrics such as SINR and throughput.

In Coordinated RL, coordination graphs are employed to represent the relationships between agents. Each agent is treated as a node in the graph, and edges are created to connect agents that can influence each other's actions or observations. These coordination graphs capture the dependencies and interactions among the agents, forming a structured framework for the model to use. The communication between interconnected agent nodes is facilitated by message passing mechanism. Agents exchange information and updates through messages sent along the edges of the coordination graph. This communication enables agents to share observations, coordinate actions, and update the environmental knowledge or information about other agents' states.

Value functions play a crucial role in the context of agent pairs connected by edges in the coordination graph. They capture the expected return or utility of taking specific actions in a given state. Edge value functions represent the value of joint actions taken by a pair of connected agents. These value functions guide the agents' decision-making process and help them coordinate their actions to achieve optimal outcomes.

The study [7] demonstrates that the trained coordinated RL agents effectively optimize the antenna tilt angles, leading to improved network performance, highlighting its potential in optimizing mobile networks, showcasing its ability to tackle complex optimization problems and improve user experience in wireless communication systems

2.4. Policy Explanation

Markov Decision Process (MDP) framework is employed as a ground for communicating or justifying important behavioral patterns in Autonomous Policy Explanation. Within this framework, (S, A, T, R) definition is used. Here, S describes a set of environment states or domains and A stands for a set of actions the agent can take. After choosing an action a $\in$ A from the action set, the agent transitions from state s0 $\in$ S to s1 $\in$ S. Function T (s0, a, s1) = [0,1] expresses the s0 to s1 transition probability and thus is called a transition function. Reward function R(s0, a) determines the agent's reward after the transition. Policy in MDP terms is π: S $\times$ A $\rightarrow$ [0, 1]. An optimal policy maximizes total reward from every state of the MDP.

Authors of [8] propose to use communicable predicates to convert the states to Boolean minterms (prime implicants). Communicable predicates mean a set of Boolean classifiers that provide an abstraction over the feature vectors. They are encoding the most important characteristics of every state. Each of the classifiers contains a natural language (NL) description for the True clause. Communicable predicates and their descriptions are provided by the user for each of the environments in question.

Quine McCluskey Method is used to minimize the Boolean functions that map features encoded in predicates over the agent's states. It simplifies Boolean expression into the essential form using prime implicants. Unnecessary implicants are eliminated by combining those which differ at exactly one variable, according to the rule:

$$XY + X\overline{Y} = X$$

First, all the minterms are sorted by the number of ones. Then, literals from adjacent groups are combined. The process repeats iteratively until it's not possible to combine any of the implicants

Figure 1: Minterm generation

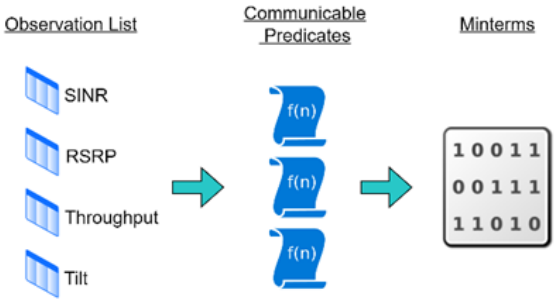
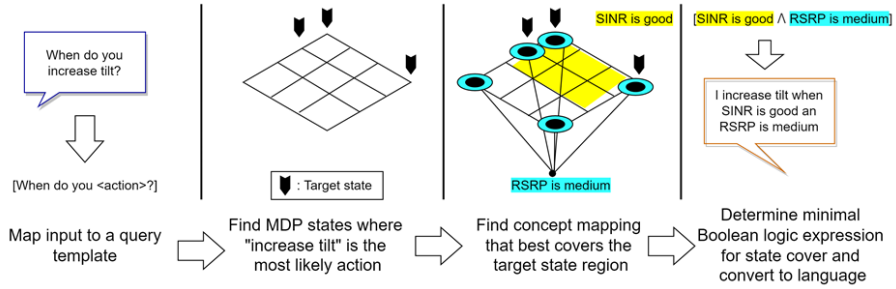


Figure 2: Minimal Set Cover



anymore. The result is the minimum possible list of prime implicants that can cover the function – canonical Disjunctive Normal Form (DNF).

An example of a question template is "When do you do {action}?". I.e. in RET case, we can ask an agent "When do you increase tilt?" in 2. First, Minimal Set Cover is constructed from natural language (NL) questions to discover states where the action "Increase tilt" has been selected. Then, the states are projected onto hypercube vertices using predicates resulting in DNF. After that, the Quine McCluskey method is applied to the minterms in order to infer the essential prime implicants. Implicants are then converted to NL explanations using descriptive clauses of communicable predicates.

3. Related Work

RL methods include numerous approaches to the explainability problem in a single-agent environment. According to existing taxonomies of explainable methods for RL in general and in the field of deep RL [2, 9], there is a lack of understanding of how to apply the XRL methods in a more practical setting, especially in telecommunication [10]. Experimental evaluations in the industry outside of narrow use cases in robotics (often for grid-based environments) or gaming are rare [2]. Though some works [11, 12] investigate the application of Explainable AI in a wireless communication setting, they concentrate on 5G/6G use cases, without highlighting the RET problem.

Some methods are focused on studying how features affect the resulting agent's actions in the current state [13, 14, 15, 16, 17]. Here one approach is to create easily interpretable policies using decision trees (DTs) [13] sometimes augmented with Markov Decision Processes (MDPs) [14], fuzzy RL [15], off-policy training [16], genetic programming [17], etc. Another approach is to match a policy to its interpretable counterpart using imitation learning [18] or its augmented versions [19]. In such a manner, an agent can learn from humans e.g. assembly robot being able to repeat the movements of the worker [20]. Finally, imitation of human motions, accompanied by an explanation in natural language, helps to understand the agent's actions better [21, 8]. Policy explanations also include saliency maps - images depicting the parts of the model responsible for the current decision [22, 23]. They are used in the game industry though are not reliable enough to produce a complete explanation [24].

Other methods go in the direction of finding the training points, the change of which will influence the resulting Q-values [25], use characteristics of the domain to aid the explainable structure of the model [7] or approximating the model to explainable one [26] which is echoed in [2]. For example, the Coordinated RL approach [7] is leveraging the graph structure based on the mobile network environment topology to train agents' value functions pairwise which would aid in clarifying the agents' interactions.

Finally, some methods are focused on determining the long-term policy. For instance, by recording the history of chosen actions with a Recurrent Neural Network (RNN), converted to save space and keep the most important points intact e.g. using finite state machines [27] or by investigating how transition tuples change over time [28, 29] or by estimating the number of conditions agent needs to satisfy in order to reach the goal [30, 31].

A survey done in [2] focuses on explainability in deep RL. Authors divide these approaches into two main categories: transparent and post hoc explainable models. Here, transparent models represent algorithms that are explainable by design. For example, the agent orders goals by the degree of importance using hindsight experience replay [32] or its variations [33]. Another way is to learn the explanation model at the same time as the policy, for example, by the means of reward decomposition in Q-learning and minimal sufficient explanation to identify why the specific action was chosen [34, 35]. This approach echoes the category in the Learning Process and Markov Decision process from [9]. Shapley Q-values were proposed as a means of measuring reward distribution by [36] in the MARL field specifically for cooperative agents. Cooperation can influence the final agent's contributions and thus should influence the rewards. Some approaches employ feature importance to reduce the data dimensionality and can greatly improve learning time as mentioned in [9, 2] and proved in [35, 13, 14, 15, 16, 17].

Post-hoc means that explainability is enhanced after the model has been trained. This type of explainability corresponds to generating explanations from models without intrinsically interpretable policies in [9] and includes saliency maps [37, 22, 23] and interpretation of policy from the agent-environment cooperation.

Despite the existing variety of the XRL approaches, there is a lack of techniques focusing on more practical areas of application such as antenna tilt regulation. Multi-agent environments are not as widely covered as single-agent ones leaving space for exploration.

4. Problem Formulation

MARL studies agents in a shared environment. Decision-makers may be dispersed spatially (for instance, cell towers in a mobile network), functionally, or both (for example, in railroads there is a division of responsibility in the management of locomotives, crews, and boxcars). Multi-agent reinforcement learning is often employed for the automation of the control problem. Multi-agent control problems include cooperative (traffic signal control, antenna tilt regulation, collaborative robotics), competitive (games), mixed (team sports), and adversarial (cyber-security).

However, there is no clear understanding of the underlying cause of the individual agent's actions. XRL methods which are designed to combat uncertainty and ensure the trustworthiness of the applied reinforcement learning algorithms can be used to solve this problem for multi-agent environments. XRL can be applied to many cases where MARL is implemented and ensure transparency, trustworthiness, and replicability of the results of a model. The goal of the study is to train a reinforcement learning policy and apply policy explanation methods based on the data recorded during the training and internal explainability features of the algorithm's structure.

Specifically, XRL approach is well applicable to the Remote tilt control (RET problem) which is concerned with controlling the tilt of an antenna remotely instead of its manual adjustment. XRL can improve the performance of the whole network by identifying possible underlying reasons for agents' actions as well as potential sources of conflict. MARL environment in question consists of seven base stations which incorporate three antennas each pointing in different directions that are distributed throughout the terrain. Optimization of the key KPI parameters (such as connection quality for the end user) by incrementally changing the antenna tilt is achieved with reinforcement learning algorithms (DQN, CRL).

Research Questions

- How can the applied RL algorithm extend policy explanation?

  For example the chosen model can influence the interpretability by providing counterfactual explanations or visualisations of certain agents' parameters.

- Is the policy explanation method applicable to the coordinated multi-agent reinforcement learning method in the networking use case (remote electrical tilt problem)?

- How do agents influence each others' policies and actions in such a setting?

5. Method

The case study has been selected as a research methodology in order to explore and investigate policy explanation strategies for solving the multi-agent control problem in reinforcement learning for individual and cooperative agents. A case study is suitable for this project since the research is conducted in an industrial setting (provided by a company), studies a particular phenomenon in the real world within its context, and requires an in-depth and thorough exploration of the possible use of XRL for the telecom problem (RET). The case study addresses a complex observable problem like XRL for the multi-agent cooperative environment in its setting and is building upon the existing body of knowledge.

The study is conducted from the viewpoint of the informed user since this stance is under-represented in the existing studies. The use case of this research project involves applying policy explanation techniques for trained agents in an industrial setting. The unit of analysis is a set of RL agents, each controlling the tilt of a corresponding antenna in the mobile network simulation.

The objective of the study is to investigate the application of policy explanation methods for the multi-agent environments in the example of RET problem. Understanding the inner logic of the algorithm can be very important especially in the industrial setting since it allows humans to gain a better grasp of the decision-making process of the agents. This understanding is crucial for experts and stakeholders to assess and validate the behavior of the model, identify patterns, and make informed decisions. Another reason for developing explanation techniques is the identification of unwanted or undesired actions exhibited by the agents. Professionals can pinpoint specific actions or interactions that may lead to suboptimal or problematic outcomes just by using policy interpretations. This enables them to diagnose and rectify abnormal behavior, ensuring the agents adhere to desired performance criteria. Policy explanations provide valuable insights, enabling specialists to ensure that the agents' actions do not violate any constraints, ethical guidelines, or other rules that are critical for the application domain. By examining the decision-making criteria and observed patterns, experts can evaluate the policy's effectiveness in various scenarios, identify its limitations, and make necessary adjustments to improve robustness and generalization.

Investigated training and explainability approaches are based on existing methods from earlier studies that affected the designs of the current case study. In order to measure the efficiency of the explanations, their complexity, time to produce policy explanations together with their fidelity, redundancy, and accuracy was taken into account.

The use case includes a simulation of 21 antennas that can be controlled remotely. To regulate the tilt, 21 RL agents were trained to optimize antenna configuration by adjusting the direction of radiation. The policy explanation method was implemented and evaluated based on the cooperative policy.

6. Ethical and Societal Considerations

Providing transparency to black-box models is ethically justified due to the potential consequences of using these models in critical situations without fully understanding their inner workings. Failure to comprehend the decision-making processes of complex algorithms can result in unwanted and potentially dangerous outcomes. For instance, reinforcement learning is a promising avenue for implementing industry automation, but the need to avoid environmental damage and ensure safe and sustainable operations highlights the importance of comprehending how the agent functions and what influences its decisions.

Moreover, it is crucial to consider the environmental impact of training deep learning models. The computations and data processing efforts required to train them demand significant energy consumption, leading to harmful carbon emissions that can have lasting environmental effects.

Additionally, while regulations such as the EU General Data Protection Regulation may require companies to provide explanations for their decision-making processes when using black box models, it is important to note that the ethical implications of explainability go beyond mere compliance. The ability to understand and interpret agent decisions is necessary for ensuring accountability, fairness, and trust in the deployment of these models, particularly in critical applications. The policy explanation methods applied for the RET problem are focused on understanding the decision-making process and behavior of the agents, not requiring data of any real user. By utilizing a simulated environment and synthetic data, the study ensures the privacy and confidentiality of individuals.

7. Description of the work

After the prestudy has been conducted, a CRL has been identified as a training method [7]. CRL was chosen as an appropriate technique to train large numbers of agents (21) simultaneously in a custom environment. Its employment of the coordination graph leverages the structure of a mobile network and allows for possible computational distribution thus potentially simplifying scaling efforts. This algorithm has already proven to be successful for a similar RET use case [7]. CRL also records value functions for each pair of the connected agents which allows us to measure how agents' interactions influence their payoff. This method is well suited for the mobile network topology and has proven to be effective in a given environment.

The policy explanation method is implemented for single and multi-agent environments.

7.1. Simulation

The data used in this study originate from an internal simulation program developed by Ericsson, which incorporates propagation models described in [38]. To facilitate reinforcement learning tasks, the simulation program is integrated into a Python program and wrapped within an OpenAI Gym environment — a widely used open-source interface for reinforcement learning. The simulation program relies on default inputs for this particular study. It is important to note that the simulation code has been exclusively developed by Ericsson and is protected by a non-disclosure agreement. Hence, specific implementation details of the simulation cannot be provided in this report. With the simulation's flexibility, it is possible to set different configurations of action and observation spaces as well as the reward. Through multiple experiments, the most effective choices for the agent's observations and rewards have been determined.

7.2. Environment

The policy explanation method proposed in [8] was implemented to be universal for single-agent environments. Agents can be interchangeable provided the user supplies instruction sets for observation/action space and predicates. As a demonstration, the technique was implemented for CartPole, Lunar Lander single-agent environments, and a proprietary multi-agent environment.

CartPole is a single-agent environment from OpenAI gym. It was based on an inverted pendulum with a center of gravity above its pivot point as illustrated by Figure 3. Here, a pole is attached to a cart that can move +1 or -1 (left or right) along the horizontal track. After the initial state, when a pole is positioned straight, reward +1 is given for each step the pole is remaining upright. Observation space is continuous and includes Cart Position (-4.8 to 4.8), Cart Velocity (-Inf to Inf), Pole Angle (-24° to 24°), and Pole Angular Velocity (-Inf to Inf).

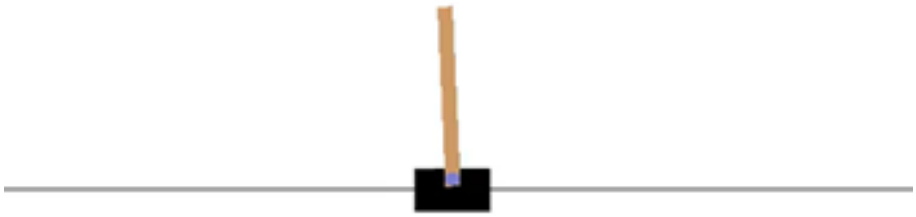
Lunar Lander is another OpenAI gym environment illustrated by Figure 4. The goal of the agent (lunar lander) is to land in between the two flags. There are four possible discrete actions that the agent can take:

1. Do Nothing
2. Fire Left Orientation Engine
3. Fire Main Engine
4. Fire Right Orientation Engine

Lunar Lander has a mixed observation space:

1. X distance from the target site (Continuous, typically ranging from -1.2 to 1.2.)
2. Y distance from target site (Continuous, -1.2 to 0.6)
3. X velocity (Continuous, -0.07 to 0.07)
4. Y velocity (Continuous, -0.5 to 0.5)
5. The angle of a ship (Continuous, $-\pi$ to π)

Figure 3: Illustration of a Cartpole environment (OpenAI). A brown pole in the middle balances on a black, wheeled cart underneath.



6. Angular velocity of a ship (Continuous, -1.0 to 1.0)
7. The left leg is grounded (Binary, 0 = not in contact, 1 = in contact)
8. The right leg is grounded (Binary, 0 = not in contact, 1 = in contact)

The proposed XRL method was then expanded to include multi-agent environments: RET environment.

RET environment which is a base experiment environment consists of a mobile network with 7 base stations and 21 antennas over a set terrain as illustrated by Figure 5. Each base station deploys 3 antennas positioned in opposite directions so that together they construct a hexagonal structure. This simulated setup is very similar to the real-life mobile network which provides for a more realistic scenario and would allow easier field deployment if needed.

In the context of cellular networks, the association of user equipment (users) with a specific cell is determined based on the strength of the signal they receive from the antenna of that particular cell (RSRP). If a user receives a signal that is sufficiently strong, they are associated with that cell. The strength of the signal experienced by users is influenced by various factors, including tunable parameters such as antenna tilt and power. Environmental aspects also play a role in affecting signal strength. Ultimately, the quality of experience of users is directly impacted by the signal strength they receive, which is determined by set parameters. In this work, SINR, RSRP, and antenna tilt are used as observations in continuous observation space and each antenna chooses one of three actions from the action space: increase tilt, decrease tilt, or do not change tilt.

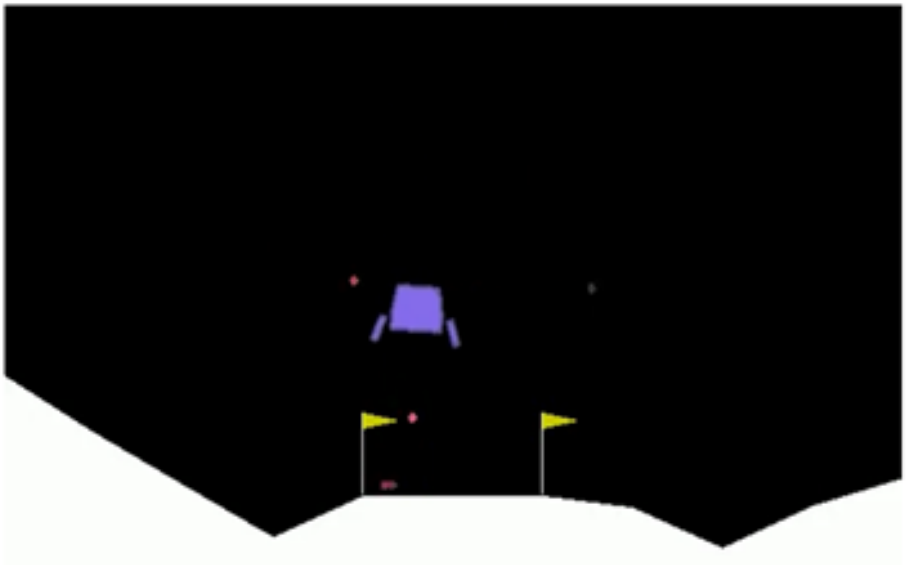
7.3. Implementation

CRL methods have been already implemented for the telecom use case [39, 7]. CRL configuration was updated to fit the environment requirements, action, observation, and reward structures and it was employed as a collaborative intrinsically explainable multi-agent algorithm- in the experiments.

Policy explanation initially has been implemented for a toy example of a Grid World-like environment with a very limited state space. Current work included broadening and updating it to be used interchangeably with most OpenAI Gym-compatible environments and Ericssons RET simulation.

The implementation consists of two parts. The first part involves configuring CRL for the given environment and interpreting the agent interactions from the observation data recorded during

Figure 4: Illustration of a Lunar Lander environment (OpenAI). A purple lander in the middle fires propulsion engines to land in between the two yellow flags.



training. The second is using the gathered data to use the policy explanation for interpreting agents' actions in a cooperative multi-agent environment.

7.4. CRL

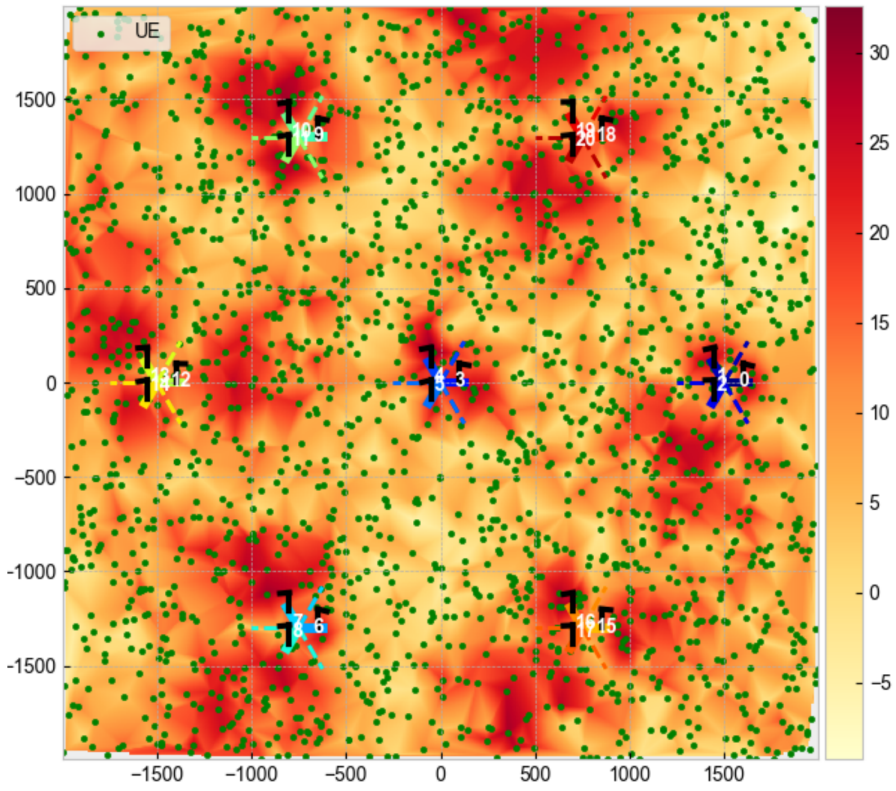
- **Coordination graph**
  In order to leverage the topology of cellular networks effectively, a coordination graph is used to represent the network. This graph construction involves considering the deployment of base stations, where each individual cell becomes a node in the graph. The connections between nodes are based on their potential to influence the received signal of each other's users. To establish these relationships, an automated procedure is employed, combining domain knowledge and heuristics. Factors such as the geographic distance between cells and the radiation patterns of the antennas [40] are taken into account to determine the connections between nodes in the coordination graph. This approach enables the representation of the network's structure and interdependencies, which can be further utilized in coordinated reinforcement learning algorithms for optimizing mobile networks. The graph for this work has been constructed with the knowledge of antenna radiation patterns, and edges connecting antennas that can cause interference to each other's signals.

- **Message Passing** Max plus message passing algorithm is employed for finding the assignment maximizing a payoff function in a factored graph [41]. It operates by sending messages to optimize the local payoff structures over the connected graph edges. This method computes an optimal joint action for every agent throughout the coordination graph. Every agent sends repeated messages to each of its neighbors one by one. Each message combines the agent's individual payoff for taking a certain action, the payoff for joint action with this specific neighbor with messages received from every other neighbor except the current recipient. Message passing occurs until messages converge or a certain number of steps is reached. It is used in CRL to compute action pairs by exchanging messages between neighboring antennas:

$$\mu_{ij}(a_j) = max_{a_i}[Q_{ij}(s_i, s_j, a_i, a_j) + \sum_{k \in N(i), k \neq j} \mu_{ki}(a_i)] + c_{ij}$$

  for every $a_j \in A_j$, where i is the agent constructing the message μ_{ij} to its neighboring agent j. $N(j)$ is the set of neighbors of j, c_{ij} - normalization term. $Q_{ij}(s_i, s_j, a_i, a_j)$ is a value function for the two neighbors represented by a neural network.

Figure 5: Experiment environment. The environment features 3 base stations with seven antennas each. Heatmap denotes SINR values for the user equipment (UE) represented by the green dots.



- **Edge value factors explainability** In order to describe the relationship between the neighboring antennas, information in the form of value functions is gathered for each corresponding pair of connected agents. Action pairs are then evaluated based on levels of interference, the proximity of the agents to each other, and their directions of radiation.

7.5. Policy explanation

Implementation includes settings files for each environment action and observation spaces, labels of observation values, and split range for creating a custom set-up. The user should also provide a set of communicable predicates discussed in the Background section for algorithms to correctly classify the state space.

State records are collected in a list which is constructed as follows: each record consists of a vector of observations that are recorded during the agent training, action, and state number. After training is finished, the recorded observations are fed to the trained agent to infer the list of actions that will be used for generating policy explanations.

A user provides the list of predicates, consisting of the boolean classifiers splitting each observation range into meaningful clusters and natural language labels for each classifier. Four algorithms which correspond to three question templates then make use of the given predicates and state record list to identify the behavioral model of the reinforcement learning system.

The first algorithm (Minimal Set Cover) describes the target states in NL. We project the states onto the hypercube vertices using predicates resulting in DNF. Quine McCluskey method is applied to the minterms in order to infer the essential prime implicants. Implicants are then converted to NL explanations using descriptive clauses of communicable predicates.

Algorithm 2 corresponds to the question template “When do you do A?”, where A is an action from the action space. This module specifies the conditions under which a given action is chosen. First, it searches for all the states where action A has been selected, and after that target states

are passed to the Minimal Set Cover.

Algorithm 3 corresponds to the question template "Why did not you do A in state S?", where A is an action from the action space, and S is the state from the state space. This module identifies the difference in expected and observed agent behaviors. First, we identify a set of states where action A was chosen. Then, Minimal Set Cover is applied to state S as the target state and set of states where action A was chosen (S') as non-target states. Thus, we identify the difference between state S and set S'. Finally, the states where A was preferred are passed to Algorithm 1 to find the condition for its selection.

Algorithm 4 corresponds to the question template "What will you do when C?", where C is a set of conditions. This module summarizes policy under specific circumstances. First, we map C to the communicable predicates. Then, states matching the description are pinpointed. After this, the most popular action A taken in the identified states is found. Conditions, provided in the description, are then discarded. The Minimal Set Cover is used on states matching the description to detect other standard features.

7.6. Training

Ray RLlib [42], an open-source library for RL was used to train CRL and Deep Q-Network (DQN) algorithms. RLlib features built-in support for distributed training, algorithm customization, and scalability, which can be advantageous in complex RL environments such as telecom scenarios. Training parameters include:

- Learning rate: 0.001
- Gamma: 0.99
- Epsilon: 0.02
- Batch size: 32
- Size of replay buffer: 50000
- Training timesteps: 10 000

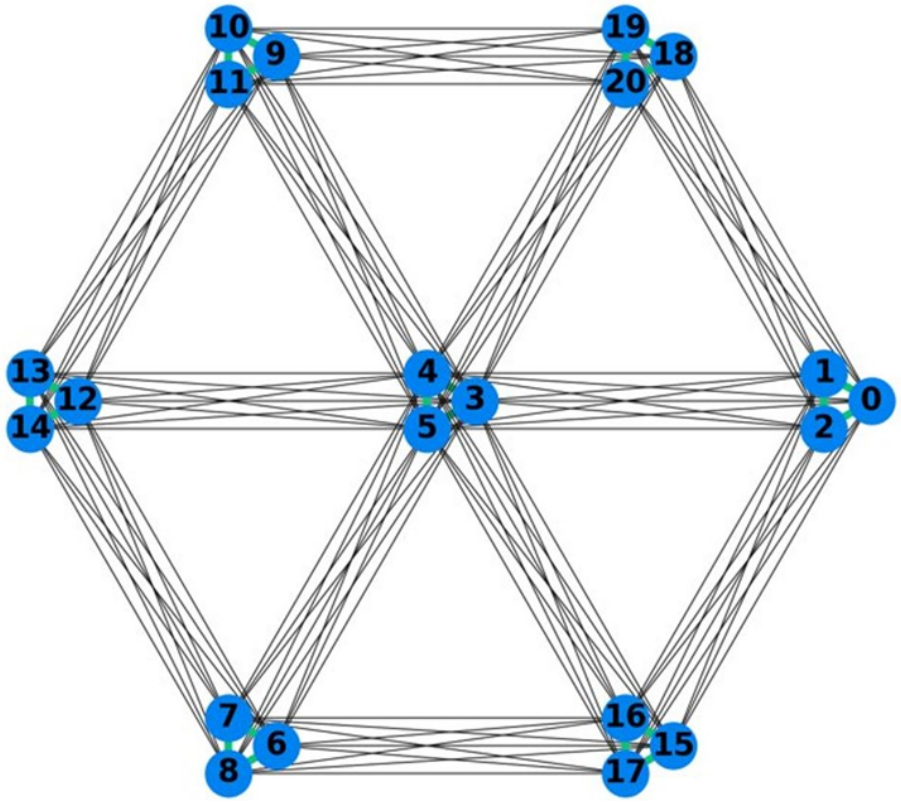
Observation batches were recorded during the training in JSON format which were then decoded during runtime. Checkpoints are used to save the model's progress during training. When policy explanation algorithm is run, it restores the previously trained agent from the saved checkpoint. After that, fully trained and restored agent is used to produce a set of actions corresponding to the set of training observations (uploaded from JSON). A set of states is created by combining the newly computed actions with the previously recorded observations for each agent.

Generating Explanations By examining the learned value functions across the coordination graph, correlations are identified between antennas to store and later leverage this information as a network planning tool. Heatmaps are then built for each pair of the connected nodes in the graph as shown in Figure 6. Data to produce heatmaps is generated by the value function of two connected agents. Heatmaps combine average value information over multiple timesteps for each possible action combination between these agents. Received data such as overall value intensity throughout action combinations as well as higher/lower intensity for certain actions of certain neighbouring agents can then be used together with information about agent's position and direction (towards or opposite each other) to estimate the risk of interference. This allows user to identify the most and the least likely action combinations diagnosing antenna configurations and optimizing coordination strategies based on the information of learned value functions from CRL.

7.7. Experiments

A custom environment mentioned before has been set up to train the CRL algorithm. It has been compared to Deep Q-Network (DQN) as a standard approach using reward per equal number of steps to measure performance. Reward metric is based on Reference Signal Received Power (RSRP) and Signal-to-Interference-plus-Noise Ratio (SINR) as one of the most important signifiers of the quality of signal [43].

Figure 6: Graph structure. The graph consists of 21 interconnected nodes, representing 21 antennas in the environment.



The RSRP is used in telecommunications to evaluate the performance of wireless networks. It specifically measures the power level of the reference signal transmitted by a base station and received by User Equipment (UE).

RSRP is a fundamental metric that is used for cell association, thus it can also be used to indicate the coverage of the antenna. It represents the power level of the primary signal transmitted by the serving base station, without considering interference or noise.

$$RSRP = \frac{P}{N}$$

where:

(P) – the power of the reference signal received by the UE,

(N) – background noise.

However, RSRP alone may not provide a complete picture of network performance. Other metrics such as SINR are also considered to evaluate the overall network quality.

SINR is defined as the ratio of the power of the desired signal (S) to the combined power of interfering signals (I) and background noise (N). Mathematically, it is expressed as:

$$SINR = \frac{S}{(I + N)}$$

In this equation, a higher SINR value indicates a stronger desired signal relative to the combined interference and noise levels.

Observations - In the simulation, each antenna observes the SINR of the User Equipments (UEs) it covers, measured in dB. RSRP of UEs is another observed metric, measured in dBm. Additionally, the current tilt of the antenna is recorded, which is normalized between 0 and 1. Observation for each agent is recorded as 3 element array.

Actions - Every antenna's tilt angle (θ), can be adjusted within a range of $\theta \in \{0, 1, ..., 15\}$. An antenna is positioned in a range of no tilt (pointing straight up) and 15 (full tilting down).

The action space for the agent is discrete and consists of three possible actions:

- Increase tilt: The agent can increase the tilt of the antenna by one step.
- Decrease tilt: The agent can decrease the tilt of the antenna by one step.
- Do not change tilt: The agent can choose to keep the antenna's tilt unchanged.

During each step of the RL training or decision-making process, the agent is allowed to select one of these three actions to modify the tilt of the antenna. Discrete action space allows the agent to increment or decrement the tilt gradually or maintain the current tilt if deemed appropriate based on its learned policy. This way every agent has the flexibility to modify the antenna tilt to different angles within the specified range, enabling it to explore various tilt configurations and learn optimal strategies that maximize system performance or achieve specific objectives in the telecom environment.

Shared Policy - All the agents share the same policy parameters.

Graph - In the context of creating coordination graphs sparse graphs, dense graphs, and trees were explored to determine the structure of the graph. Each method has its own characteristics and considerations.

- Sparse Graphs: Characterized by having a limited number of connections or edges between nodes. In the context of coordination graphs for antennas, this would mean that only a subset of antennas is directly connected to each other. Sparse graphs can be advantageous when there are constraints on the number of connections or when the underlying network topology suggests a sparse connectivity pattern.
- Dense Graphs: Dense graphs, on the other hand, have a higher number of connections or edges between nodes. In the case of antenna coordination graphs, this would mean that most antennas are directly connected to each other. Dense graphs can provide more opportunities for coordination and communication between antennas, allowing for more comprehensive information exchange and collaboration.
- Trees: Tree-like structures are characterized by a hierarchical arrangement of nodes, where each node has a single parent except for the root node. In the context of antenna coordination, a tree structure could represent a hierarchical organization of antennas, with some antennas acting as central hubs and others as peripheral nodes. Tree structures can provide a clear hierarchy and facilitate information flow from higher-level nodes to lower-level nodes.

In the given scenario, based on the overall distance between antennas, the dense graph structure was chosen as the preferred option. This implies that most antennas are directly connected to each other, allowing for a high degree of coordination among the antennas in the network. The decision was influenced by the need for extensive information exchange, close proximity between antennas, and the desire to enable efficient collaboration and coordination between neighboring antennas.

By selecting the dense graph structure, the coordination graph aims to facilitate effective communication and coordination among antennas, contributing to improved performance and optimization.

Reward - The reward is computed based on the percentage of UEs (between 0 and 1) with good coverage and good quality. Good coverage is a threshold-based check of the RSRP. Good quality is a threshold-based check of the SINR.

Validation - Explanations are evaluated by their cohesion and internal consistency as well as how they correspond to the agent's position in the graph structure.

8. Results

Outcomes of the work include training the MARL model and applying the policy explanation method to measure its efficiency, inference time, and robustness of approximate agents' policies. Another goal lay in evaluating the influence of each agent on the final decision and other agents' actions.

8.1. Generating Policy Explanations

Policy explanation types include single and multi-agent environments as well as three possible question templates for each of the environments:

1. When do you do {action}?
2. Why didn't you do {action} in state {state #}?
3. What will you do when {conditions}?

Single-agent environments

As a result of the study, standard question templates were implemented for single-agent environments Cartpole (Figures 9, 10) and Lunar Lander (Figures 8, 10). They show internal consistency and robustness (policy explanation results for one agent stay consistent with different data and different question question templates). For example, in Figures 8, 10 one of the reasons to fire left orientation engine for Lunar Lander is to stabilize lander moving to the left. For the next question, reason for a lander not to fire left orientation engine in the given state is that the lander is moving to the right and right side stabilisation is needed. Similarly, for Cartpole, the agent didn't push the cart to the right in Figure 9 because the pole was falling to the left and push to the right was needed. In Figure 10 policy explanation supports this notion by answering "push cart to the right" for stabilisation if the pole is falling to the right.

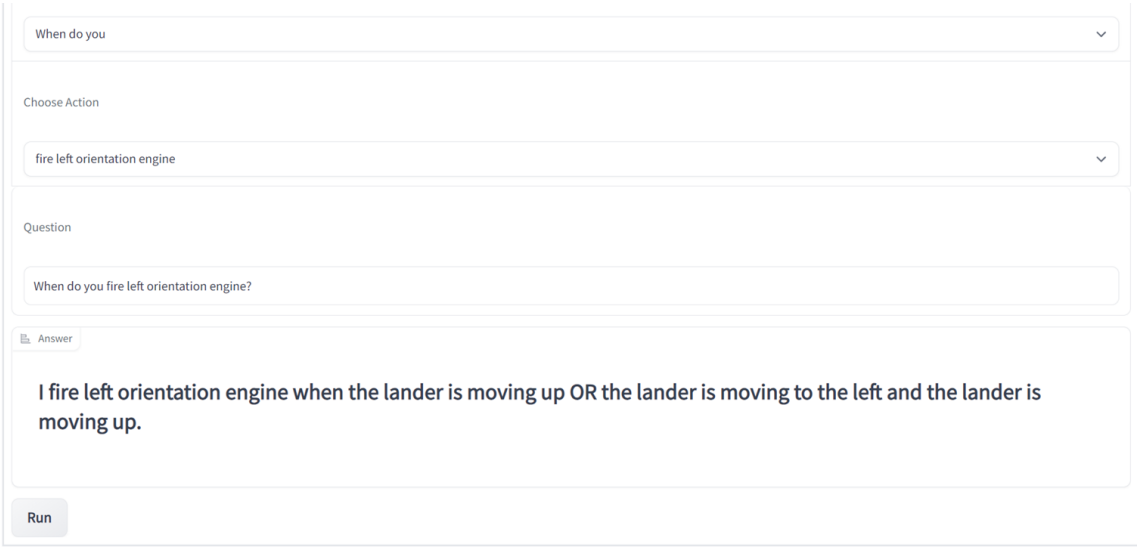
Multi-agent environments For multi-agent environment pool is represented by the proprietary telecom environment described previously in the Environment section. Three observation parameters were used to construct predicates for MARL policy explanation: RSRP, SINR, and tilt values. In the experiments, RSRP ranges from -2 to 5 (Figure 11) and was split into five groups: excellent, good, fair, poor and no signal; SINR which ranges from -3 to 7 (Figure 12) was split into four groups: stong, good, fair, and no signal. Antenna can tilt in range from 0 to 15 normalized on 0 to 1 interval and was split into three groups by ranges 0 to 0.2 to 0.8 to 1 (Figure 13). In experiment setting, state data was plotted to check if there are any clustering patterns to derive the split of the observation features into groups from (Figure 14). Unfortunately, no discernable patters or action were recognised, so the groups were split based on the prestudy of mobile signal strength recommendations from professionals in the field [44].

Policy explanation in this case includes similar templates as for a single-agent environment with the addition of the possibility of choosing agents to consider for policy explanation illustrated in Figure 15. Similar notions of internal consistency and robustness (policy explanation results for one agent stay consistent with different data from the same agent and different question question templates). For instance in Figures 16 and 17, when agent 0 is asked when does it increase tilt, one of the conditions is strong signal, same holds true in reverse - if signal is strong, agent 0 would increase tilt.

8.2. Generating Coordinated Explanations

Coordinated explanations were produced by using value functions data from every pair of connected agents recorded over 100 timesteps. Q values for every possible action combination of neighboring nodes were then plotted as heatmaps. The heatmaps' highest (red) and lowest (blue) values correspond to the most common actions chosen by the agent pairs as evident from the Figures 18 and 17. Agents 1 and 3 are located next to each other on the map which can cause their relative interference, agents 3 and 5 are located on the same base side by side, so their radiation perimeters might overlap causing them to choose different tilt angles to produce higher signal quality (Figure6)

Figure 7: Example of asking a question from a first template to an agent from a single agent environment (Lunar Lander). Question: When do you fire left orientation engine?



8.3. Performance

Current study has proven that policy explanation method can be applied to the coordinated multi-agent reinforcement learning and provide similarly clear and robust results as with single-agent environments. Conducted research has shown on practice how observations influence agents decisions for both single- and multi-agent environments as well as proven based on CRL Q-value functions analysis that some agents have consistently preferred certain interactions with neighbouring agents and avoided choosing certain type of actions with certain neighbours as described in 8.2.

Implementing CRL algorithm instead of simple DQN was more effective in terms of reward during the training process and allowed to generate more complete explanations. Using intrinsic CRL allowed to generate counterfactual explanations by exploring all possible combinations of agents' actions and not only ones that were already taken (like in policy explanation) as well as add visual information to the text question-answer format supported by the policy explanation method.

Overall, the utilization of CRL and the policy explanation method has proven to be effective in enhancing the interpretability of the agents' behaviors and decision-making processes. The combination of these techniques offers insights into the factors influencing agent actions and contributes to a better understanding of the multi-agent control problem in the context of optimizing mobile networks.

Figure 8: Example of asking a question from a second template to an agent from a single agent environment (Lunar Lander). Question: Why didn't you fire left orientation engine in state 4?

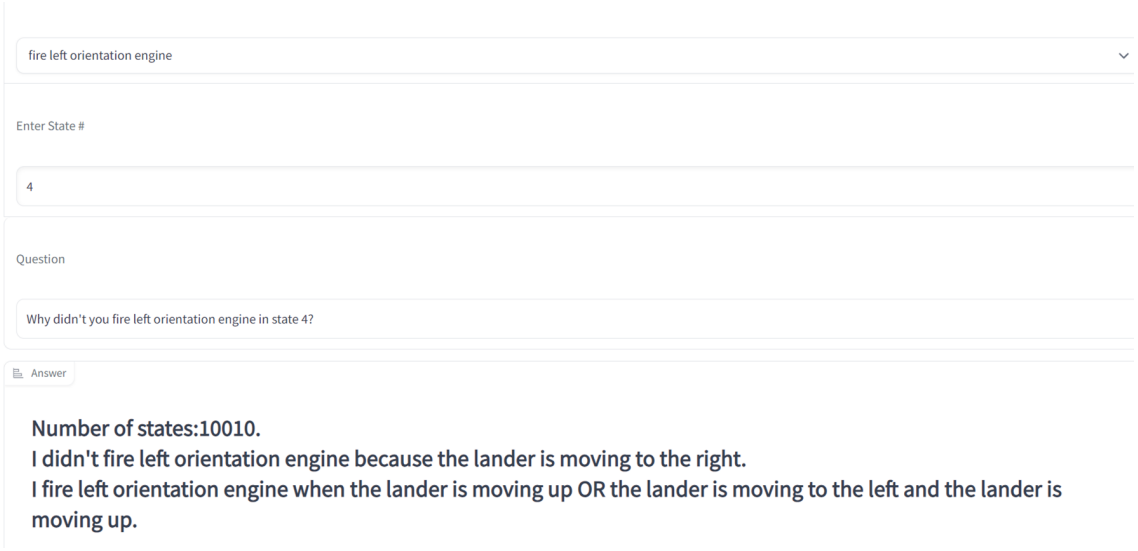


Figure 9: Example of asking a question from a second template to an agent from a single agent environment (Cartpole). Question: Why didn't you push the cart to the left in state 10 (from the state list)?

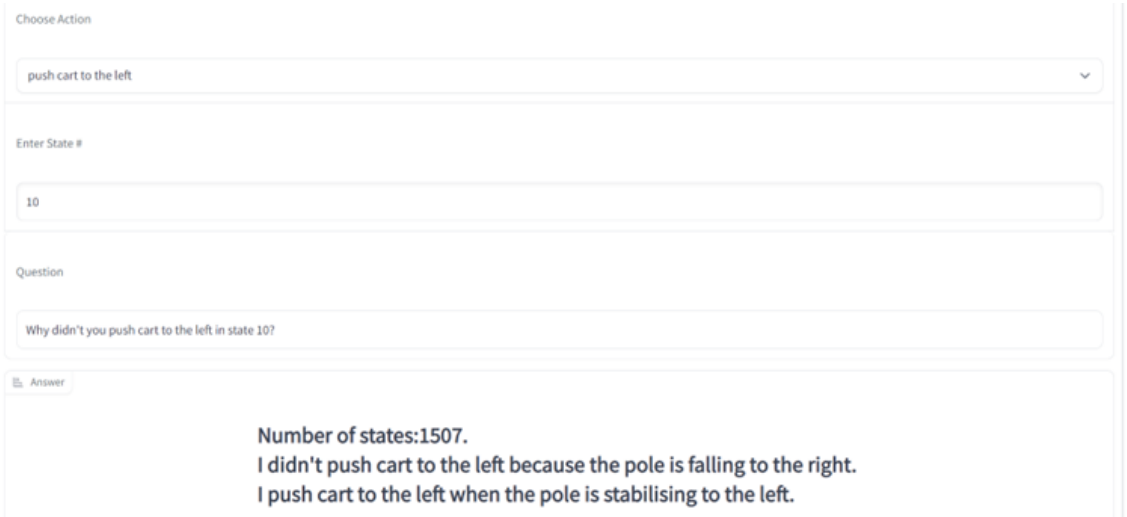


Figure 10: Example of asking a question from a third template to an agent from a single agent environment (Cartpole). Question: What will you do when the pole is falling to the right AND the pole is falling fast?

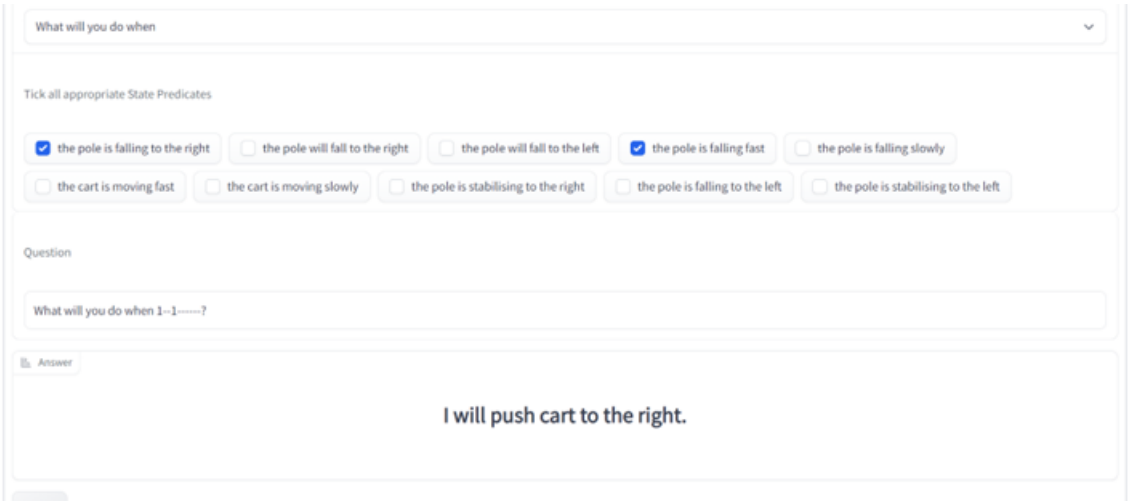


Figure 11: Range of the recorded Reference Signals Received Power (RSRP) [dBm=decibels per milliwatt] values over 5000 episodes.

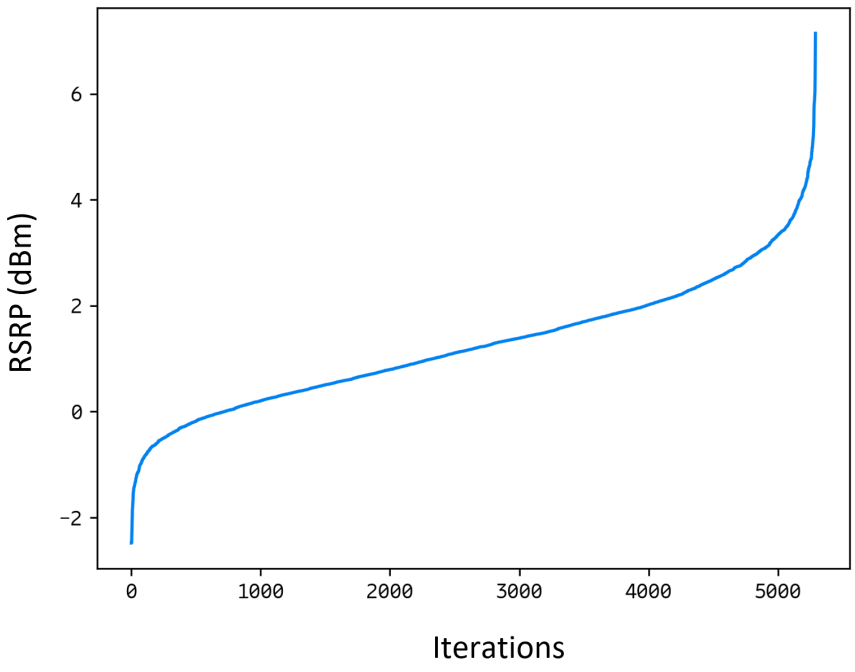


Figure 12: Range of the recorded Signal to Interference and Noise Ratio (SINR) [dB] values over 5000 episodes.

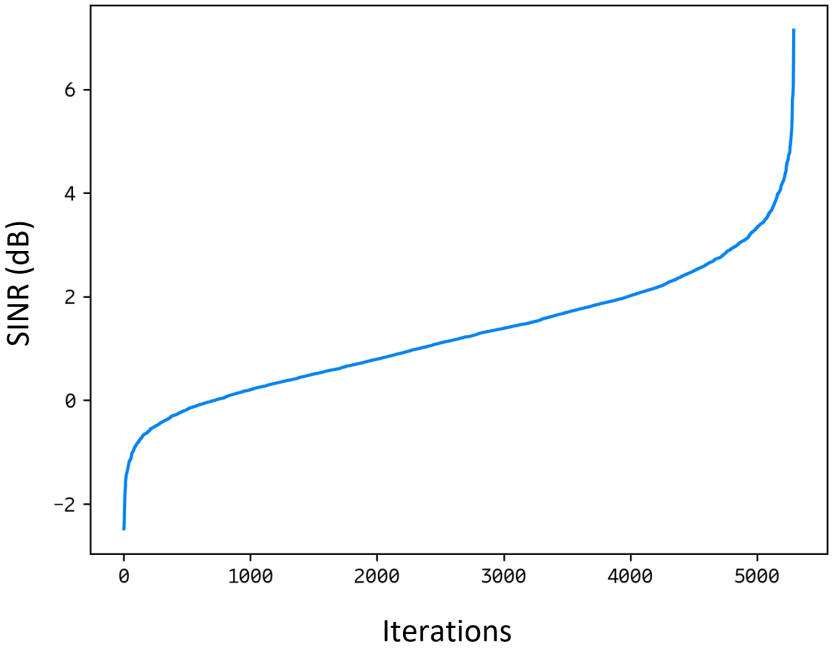
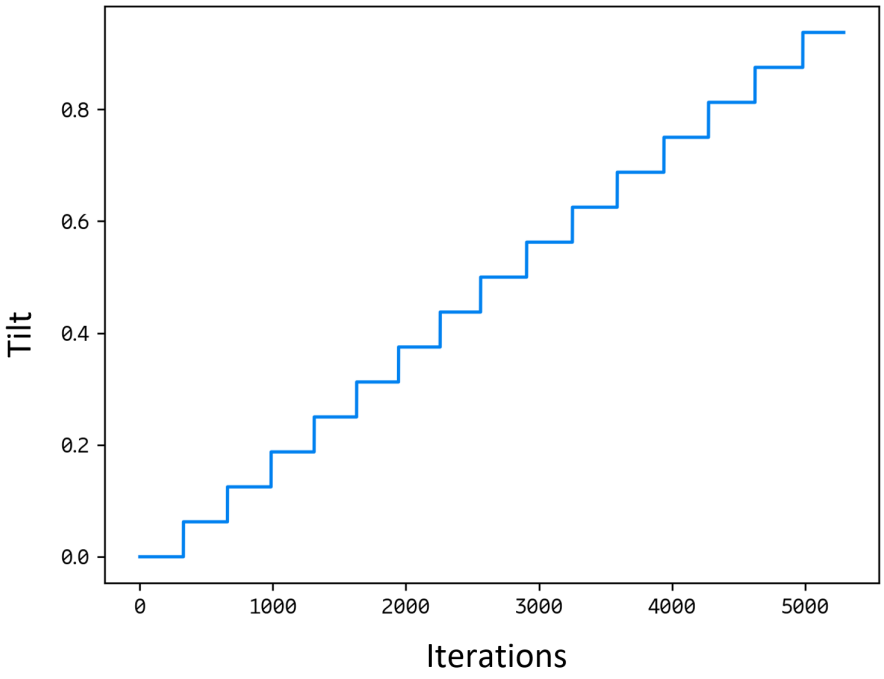
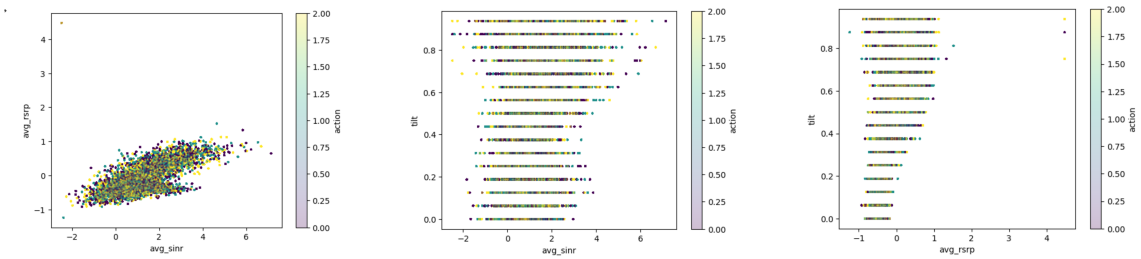


Figure 13: Range of the recorded tilt values normalized [0...1] over 5000 episodes.





(a) Actions heatmap for RSRP vs SINR.

(b) Actions heatmap for tilt vs SINR.

(c) Actions heatmap for tilt vs RSRP.

Figure 14: Action heatmaps for the observation array.

Figure 15: Example of selecting the antennas to pass to the policy explanations algorithm. Selection by ticking the number corresponding to the antenna on the environment map

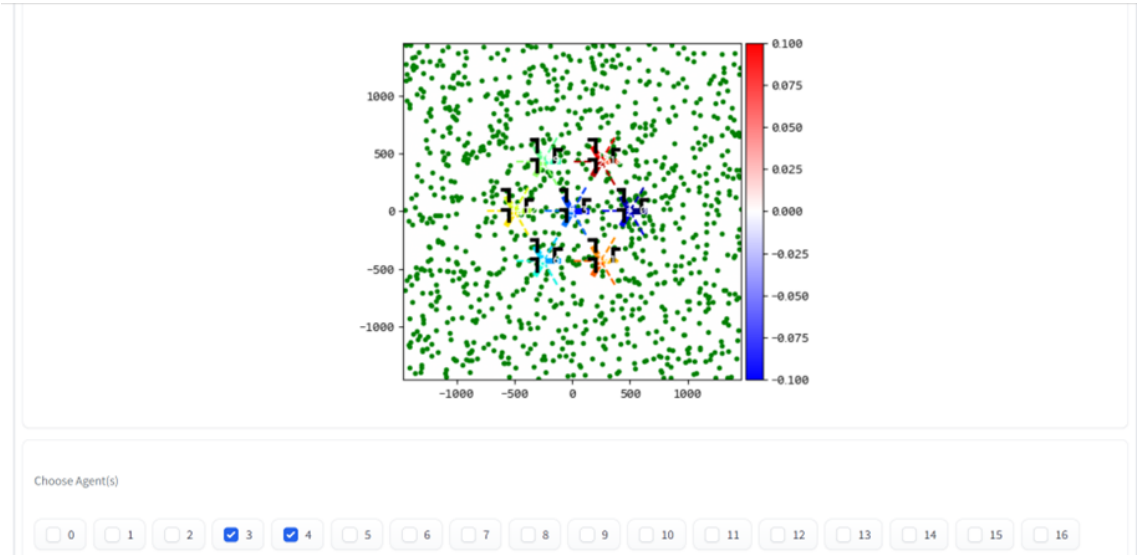


Figure 16: Example of asking a question from a first template to chosen coordinated agent from a multi-agent environment. Question: When do you increase tilt?

Figure 17: Example of asking a question from a third template to chosen coordinated agent from a multi-agent environment. Question: What do you do if the signal is strong?

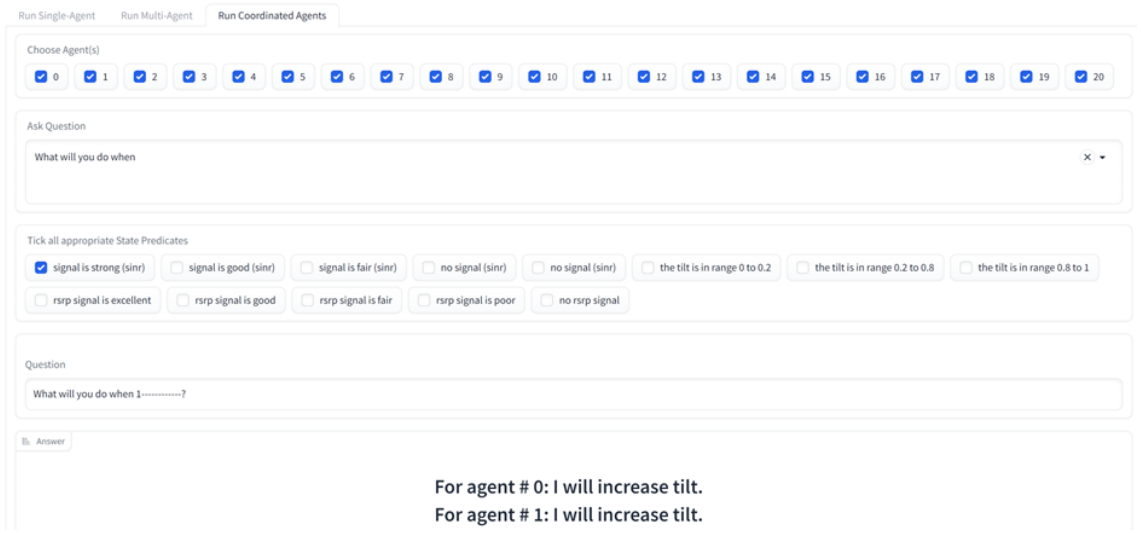


Figure 18: Q-values of pairs of actions for agents (1, 3). The most likely taken action pair (red) is: 'decrease tilt' (0) for agent number 3 and 'increase tilt' (2) for agent number 1. The least likely action pair (blue) is: 'increase tilt' (2) for agent number 3 and 'decrease tilt' (0) for agent number 1.

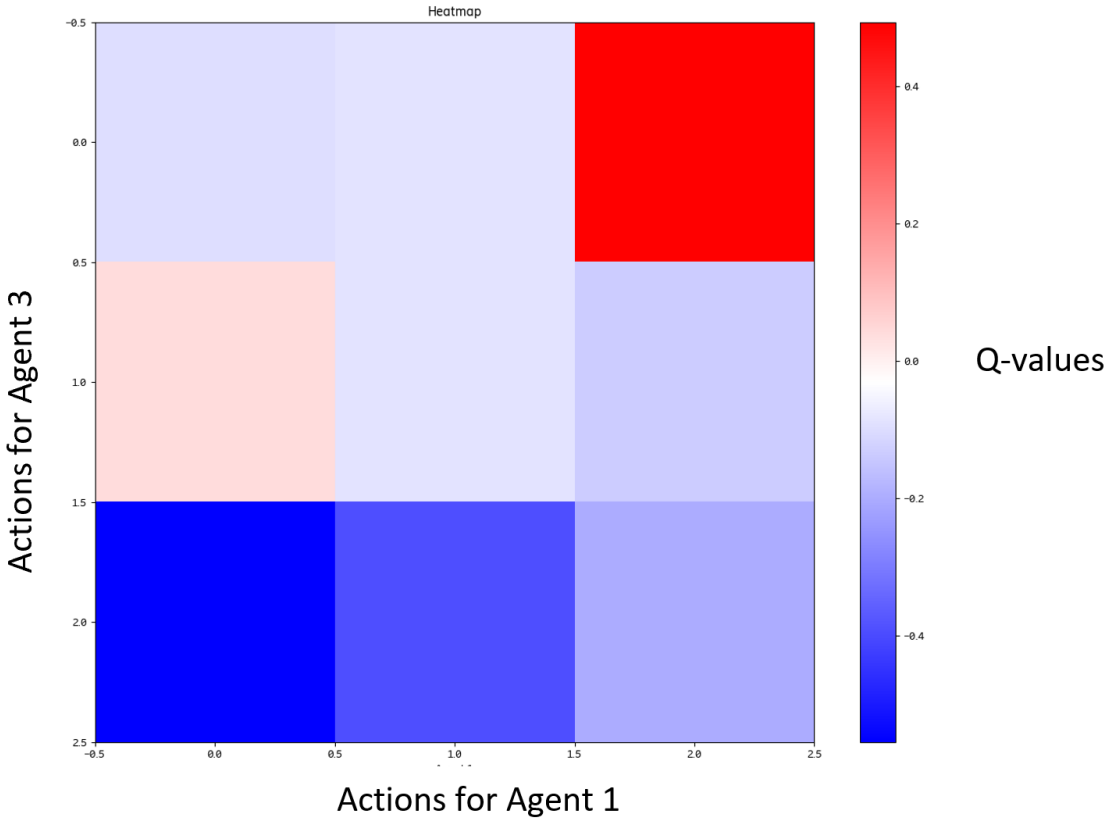
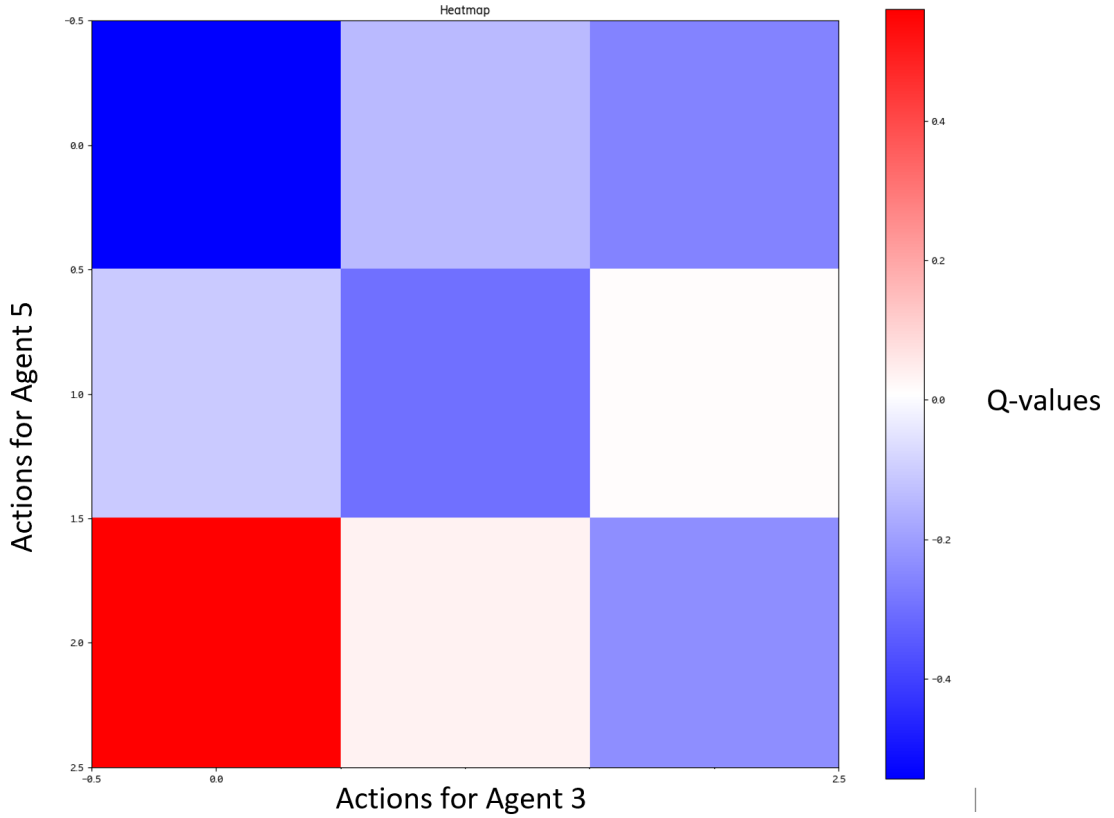


Figure 19: Q-values of pairs of actions for agents (3, 5). The most likely taken action pair (red) is: 'decrease tilt' (0) for agent number 3 and 'increase tilt' (2) for agent number 5. The least likely action pair (blue) is: 'decrease tilt' (0) for agent number 3 and 'decrease tilt' (0) for agent number 5.



9. Discussion

XAI is a complex problem in single-agent RL and transparency in multi-agent RL is even more crucial to address. In MARL agents can influence and interfere with each others' as well as overall progress, making it harder to assign the credit or the blame for their actions. As a result, each agent's role is more obscure and contribution is even harder to track.

Limitations of the policy explanation algorithm include computational resources, inference and processing time which is inversely proportional to the inference accuracy (influenced by the number of predicates and observations).

Another limitation is the interpretability of the explanations. While the model can generate explanations for the policies of individual agents and their interactions, the level of interpretability may vary depending on the complexity of the explanation and the specific task.

It is important to note that the results and conclusions drawn from this work are specific to the mobile network optimization task and the chosen CRL algorithm and coordination graph structure. Generalizability to other domains was implemented for cartpole and lunar lander environments and, by design, multi-agent setup is also well generalized. However, different domains and environment settings may have different challenges especially for defining the predicates and should be further investigated.

Nevertheless, the findings of this study highlight the potential of using policy explanation algorithms and coordinated XRL in multi-agent reinforcement learning for optimizing mobile networks.

In a broader context, the work presented in this study contributes to the growing field of explainable artificial intelligence and reinforcement learning. By providing insights into the decision-making strategies of cooperative agents, it offers transparency and interpretability, which are crucial for deploying AI systems in critical domains such as telecommunications.

Overall, while there are limitations and challenges to address, the application of MARL coordinated learning policy explanation techniques holds promise for optimizing mobile networks and can pave the way for future advancements in the field of multi-agent control.

10. Conclusions and future work

To sum up, the CRL agent has been trained for a custom RET environment consisting of 21 agents. Also, the policy explanation has been implemented and tested for single- and multi-agent environments.

Explanations have been extracted from the inner structure of the Coordinated agents and evaluated. The value functions for agent pairs, represented by the edge values, provided helpful information about the coordination and interaction between neighboring antennas.

The implementation of the policy explanation module allowed for a deeper understanding of the agents' strategies and decision-making process. The extracted explanations provided insights into the role played by each agent in reaching the common goal of maximizing mobile connectivity. This information can be utilized for troubleshooting, network planning, and identifying areas for improvement in mobile network deployment.

Moving forward, additional techniques for improving the scalability of the CRL approach might be explored, as larger networks with a higher number of antennas and agents may pose computational challenges. Additionally, considering different types of coordination graphs and message-passing protocols could further enhance the performance and efficiency of the coordinated reinforcement learning system.

Further studies can include using value function insights from the value functions for agent pairs and comparing them to the most common actions taken by these agents under similar circumstances (similar observations) which are implemented in the policy explanation module.

In terms of future work, expanding the study to include larger networks with more agents and antennas would provide insights into the scalability of the CRL approach. Additionally, exploring different coordination graph structures and message-passing protocols could enhance the performance and efficiency of the system. The information provided by the explanations can help network operators and experts better understand the decision-making processes of the agents and identify potential issues or areas for improvement in the network deployment and operation.

Overall, this report has provided insight into the challenges and solutions of transparency in multi-agent RL and has demonstrated the potential of XRL in optimizing mobile networks.

11. Time Plan

1. Thesis application: January 8, 2023
2. Starting the course: January 16, 2023
3. Planning and status report writing: January 16, 2023 - February 12, 2023
4. Status and planning seminar: February 15, 2023
5. Thesis work: February 16, 2023 - April 16, 2023
   (a) Planning the research: February 16, 2023 - February 23, 2023
   (b) Identifying XAI methods to implement: February 24, 2023 - March 5, 2023
   (c) Testing the selected methods for RET case in MARL: March 6, 2023 - April 9, 2023
   (d) Summarising the findings: April 10, 2023 - April 16, 2023
6. Report writing: April 17, 2023 - May 16, 2023
7. Examination: May 17, 2023 - June 4, 2023
   (a) Submitting the report: May 17, 2023
   (b) Preparing the defence presentation: May 17, 2023 - May 23, 2023
   (c) Signing up for opponentship: May 23, 2023
   (d) Preparing the opponentship report: May 24, 2023 - May 30, 2023
   (e) Thesis defence: June 1-2, 2023

References

[1] R. J. Solomonoff, "An inductive inference machine," in *IRE Convention Record, Section on Information Theory*, vol. 2. Institute of Radio Engineers New York, 1957, pp. 56–62.

[2] A. Heuillet, F. Couthouis, and N. Díaz-Rodríguez, "Explainability in deep reinforcement learning," 2020. [Online]. Available: https://arxiv.org/abs/2008.06693

[3] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. A. Riedmiller, A. K. Fidjeland, G. Ostrovski, S. Petersen, C. Beattie, A. Sadik, I. Antonoglou, H. King, D. Kumaran, D. Wierstra, S. Legg, and D. Hassabis, "Human-level control through deep reinforcement learning," *Nature*, vol. 518, pp. 529–533, 2015.

[4] M. J. Osborne and A. Rubinstein, *A course in game theory*. MIT press, 1994, vol. 1.

[5] A. Nedic and A. Ozdaglar, "Distributed subgradient methods for multi-agent optimization," *IEEE Transactions on Automatic Control*, vol. 54, no. 1, pp. 48–61, 2009.

[6] M. Haliem, T. Bonjour, A. O. Alsalem, S. Thomas, H. Li, V. Aggarwal, B. K. Bhargava, and M. Kejriwal, "Learning monopoly gameplay: A hybrid model-free deep reinforcement learning and imitation learning approach," *ArXiv*, vol. abs/2103.00683, 2021.

[7] M. Bouton, H. Farooq, J. Forgeat, S. Bothe, M. Shirazipour, and P. Karlsson, "Coordinated reinforcement learning for optimizing mobile networks," *CoRR*, vol. abs/2109.15175, 2021. [Online]. Available: https://arxiv.org/abs/2109.15175

[8] B. Hayes and J. A. Shah, "Improving robot controller transparency through autonomous policy explanation," in *2017 12th ACM/IEEE International Conference on Human-Robot Interaction (HRI*, 2017, pp. 303–312.

[9] S. Milani, N. Topin, M. Veloso, and F. Fang, "A survey of explainable reinforcement learning," 2022. [Online]. Available: https://arxiv.org/abs/2202.08434

[10] L. Wells and T. Bednarz, "Explainable ai and reinforcement learning—a systematic review of current approaches and trends," *Frontiers in Artificial Intelligence*, vol. 4, 2021. [Online]. Available: https://www.frontiersin.org/articles/10.3389/frai.2021.550030

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

[12] S. Wang, M. A. Qureshi, L. Miralles-Pechuaán, T. Huynh-The, T. R. Gadekallu, and M. Liyanage, "Explainable AI for B5G/6G: technical aspects, use cases, and research challenges," *CoRR*, vol. abs/2112.04698, 2021. [Online]. Available: https://arxiv.org/abs/2112.04698

[13] A. Silva, T. Killian, I. D. J. Rodriguez, S.-H. Son, and M. Gombolay, "Optimization methods for interpretable differentiable decision trees in reinforcement learning," 2019. [Online]. Available: https://arxiv.org/abs/1903.09338

[14] N. Topin, S. Milani, F. Fang, and M. Veloso, "Iterative bounding mdps: Learning interpretable policies via non-interpretable methods," *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 35, no. 11, pp. 9923–9931, 5 2021. [Online]. Available: https://ojs.aaai.org/index.php/AAAI/article/view/17192

[15] D. Hein, A. Hentschel, T. Runkler, and S. Udluft, "Particle swarm optimization for generating interpretable fuzzy reinforcement learning policies," *Engineering Applications of Artificial Intelligence*, vol. 65C, pp. 87–98, 08 2017.

[16] L. Zhang, X. Li, M. Wang, and A. Tian, "Off-policy differentiable logic reinforcement learning," in *Machine Learning and Knowledge Discovery in Databases. Research Track: European Conference, ECML PKDD 2021, Bilbao, Spain, September 13–17, 2021, Proceedings, Part II*. Berlin, Heidelberg: Springer-Verlag, 2021, p. 617–632. [Online]. Available: https://doi.org/10.1007/978-3-030-86520-7_38

[17] D. Hein, S. Udluft, and T. Runkler, "Interpretable policies for reinforcement learning by genetic programming," *Engineering Applications of Artificial Intelligence*, vol. 76, pp. 158–169, 09 2018.

[18] A. Al-Yacoub, Y. Zhao, N. Lohse, M. Goh, P. Kinnell, P. Ferreira, and E.-M. Hubbard, "Symbolic-based recognition of contact states for learning assembly skills," *Frontiers in Robotics and AI*, vol. 6, 2019. [Online]. Available: https://www.frontiersin.org/articles/10.3389/frobt.2019.00099

[19] O. Bastani, Y. Pu, and A. Solar-Lezama, "Verifiable reinforcement learning via policy extraction," 2018. [Online]. Available: https://arxiv.org/abs/1805.08328

[20] P. Christiano, J. Leike, T. B. Brown, M. Martic, S. Legg, and D. Amodei, "Deep reinforcement learning from human preferences," 2017. [Online]. Available: https://arxiv.org/abs/1706.03741

[21] J. Kim, S. Moon, A. Rohrbach, T. Darrell, and J. Canny, "Advisable learning for self-driving vehicles by internalizing observation-to-action rules," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 7 2020.

[22] K. Simonyan, A. Vedaldi, and A. Zisserman, "Deep inside convolutional networks: Visualising image classification models and saliency maps," 2013. [Online]. Available: https://arxiv.org/abs/1312.6034

[23] S. Greydanus, A. Koul, J. Dodge, and A. Fern, "Visualizing and understanding atari agents," 2017. [Online]. Available: https://arxiv.org/abs/1711.00138

[24] A. Atrey, K. Clary, and D. Jensen, "Exploratory not explanatory: Counterfactual analysis of saliency maps for deep reinforcement learning," 2019. [Online]. Available: https://arxiv.org/abs/1912.05743

[25] O. Gottesman, J. Futoma, Y. Liu, S. Parbhoo, L. A. Celi, E. Brunskill, and F. Doshi-Velez, "Interpretable off-policy evaluation in reinforcement learning by highlighting influential transitions," *CoRR*, vol. abs/2002.03478, 2020. [Online]. Available: https://arxiv.org/abs/2002.03478

[26] T. Huber, D. Schiller, and E. André, "Enhancing explainability of deep reinforcement learning through selective layer-wise relevance propagation," in *KI 2019: Advances in Artificial Intelligence: 42nd German Conference on AI, Kassel, Germany, September 23–26, 2019, Proceedings*. Berlin, Heidelberg: Springer-Verlag, 2019, p. 188–202. [Online]. Available: https://doi.org/10.1007/978-3-030-30179-8_16

[27] M. H. Danesh, A. Koul, A. Fern, and S. Khorram, "Understanding finite-state representations of recurrent policy networks," *CoRR*, vol. abs/2006.03745, 2020. [Online]. Available: https://arxiv.org/abs/2006.03745

[28] S. H. Huang, K. Bhatia, P. Abbeel, and A. D. Dragan, "Establishing appropriate trust via critical states," *CoRR*, vol. abs/1810.08174, 2018. [Online]. Available: http://arxiv.org/abs/1810.08174

[29] I. Lage, D. Lifschitz, F. Doshi-Velez, and O. Amir, "Exploring computational user models for agent policy summarization," in *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI-19*. International Joint Conferences on Artificial Intelligence Organization, 7 2019, pp. 1401–1407. [Online]. Available: https://doi.org/10.24963/ijcai.2019/194

[30] S. Sreedharan, T. Chakraborti, Y. Rizk, and Y. Khazaeni, "Explainable composition of aggregated assistants," *CoRR*, vol. abs/2011.10707, 2020. [Online]. Available: https://arxiv.org/abs/2011.10707

[31] N. Topin and M. Veloso, "Generation of policy-level explanations for reinforcement learning," *CoRR*, vol. abs/1905.12044, 2019. [Online]. Available: http://arxiv.org/abs/1905.12044

[32] B. Beyret, A. Shafti, and A. A. Faisal, "Dot-to-dot: Achieving structured robotic manipulation through hierarchical reinforcement learning," *CoRR*, vol. abs/1904.06703, 2019. [Online]. Available: http://arxiv.org/abs/1904.06703

[33] G. Cideron, M. Seurin, F. Strub, and O. Pietquin, "Self-educated language agent with hindsight experience replay for instruction following," *CoRR*, vol. abs/1910.09451, 2019. [Online]. Available: http://arxiv.org/abs/1910.09451

[34] A. Anderson, J. Dodge, A. Sadarangani, Z. Juozapaitis, E. Newman, J. Irvine, S. Chattopadhyay, A. Fern, and M. Burnett, "Explaining reinforcement learning to mere mortals: An empirical study," in *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI-19*. International Joint Conferences on Artificial Intelligence Organization, 7 2019, pp. 1328–1334. [Online]. Available: https://doi.org/10.24963/ijcai.2019/184

[35] W. Guo, X. Wu, U. Khan, and X. Xing, "Edge: Explaining deep reinforcement learning policies," in *Advances in Neural Information Processing Systems*, M. Ranzato, A. Beygelzimer, Y. Dauphin, P. Liang, and J. W. Vaughan, Eds., vol. 34. Curran Associates, Inc., 2021, pp. 12 222–12 236. [Online]. Available: https://proceedings.neurips.cc/paper/2021/file/65c89f5a9501a04c073b354f03791b1f-Paper.pdf

[36] J. Wang, Y. Zhang, T. Kim, and Y. Gu, "Rethink global reward game and credit assignment in multi-agent reinforcement learning," *CoRR*, vol. abs/1907.05707, 2019. [Online]. Available: http://arxiv.org/abs/1907.05707

[37] T. N. Mundhenk, B. Y. Chen, and G. Friedland, "Efficient saliency maps for explainable AI," *CoRR*, vol. abs/1911.11293, 2019. [Online]. Available: http://arxiv.org/abs/1911.11293

[38] H. Asplund, M. Johansson, M. Lundevall, and N. Jaldén, "A set of propagation models for site-specific predictions," in *12th European Conference on Antennas and Propagation (EuCAP 2018)*, 2018, pp. 1–5.

[39] T. Chu, J. Wang, L. Codecà, and Z. Li, "Multi-agent deep reinforcement learning for large-scale traffic signal control," *IEEE Transactions on Intelligent Transportation Systems*, vol. 21, no. 3, pp. 1086–1095, 2020.

[40] A. Saeed, O. Aliu, and M. Imran, "Controlling self healing cellular networks using fuzzy logic," 04 2012, pp. 3080–3084.

[41] J. R. Kok and N. Vlassis, *Using the Max-plus Algorithm for Multiagent Decision Making in Coordination Graphs*. Berlin, Heidelberg: Springer-Verlag, 2006, p. 1–12.

[42] RayTeam, "Rllib: Scalable reinforcement learning," https://docs.ray.io/en/latest/rllib/index.html, 2021, accessed: May 17, 2023.

[43] A. Shojaeifard, K. A. Hamdi, E. Alsusa, D. K. C. So, and J. Tang, "Exact sinr statistics in the presence of heterogeneous interferers," *IEEE Transactions on Information Theory*, vol. 61, no. 12, pp. 6759–6773, 2015.

[44] TeltonikaNetworks, "Mobile signal strength recommendations - teltonika networks wiki," https://wiki.teltonika-networks.com/view/Mobile_Signal_Strength_Recommendations, 2021, accessed: May 17, 2023.