Safe Reinforcement Learning for Antenna Tilt Optimisation using Shielding and Multiple Baselines

Saman Feghhi*, Erik Aumayr*, Filippo Vannella†‡, Ezeddin Al Hakim†, Grigorios Iakovidis†‡
*Network Management Research Lab, LM Ericsson, Athlone, Ireland
†Ericsson Research, Stockholm, Sweden
‡KTH Royal Institute of Technology, Stockholm, Sweden

{saman.feghhi, erik.aumayr, filippo.vannella, ezeddin.al.hakim, grigorios.iakovidis}@ericsson.com

***Abstract*—Safe interaction with the environment is one of the most challenging aspects of Reinforcement Learning (RL) when applied to real-world problems. This is particularly important when unsafe actions have a high or irreversible negative impact on the environment. In the context of network management operations, Remote Electrical Tilt (RET) optimisation is a safety-critical application in which exploratory modifications of antenna tilt angles of Base Stations (BSs) can cause significant performance degradation in the network. In this paper, we propose a modular Safe Reinforcement Learning (SRL) architecture which is then used to address the RET optimisation in cellular networks. In this approach, a safety shield continuously benchmarks the performance of RL agents against safe baselines, and determines safe antenna tilt updates to be performed on the network. Our results demonstrate improved performance of the SRL agent over the baseline while ensuring the safety of the performed actions.**

I. Introduction

The scale of modern cellular networks and user service demands make performance optimisation more challenging than ever. Network configuration must be adjusted automatically and in (near) real time to ensure a high level of Quality of Service (QoS) to each User Equipment (UE). The Remote Electrical Tilt (RET) angle of antennas, defined as the change in the vertical orientation of antennas with respect to the horizon by an electrical design, is one of the most important variables to control for Self-Organising Networks (SONs). The goal is to find an optimal tradeoff for some of the network Key Performance Indicators (KPIs), such as *quality*, *coverage* and *capacity* [1]. For example, an increase in antenna downtilt correlates with a stronger signal in a more concentrated area, as well as higher capacity and reduced interference radiation towards other cells in the network. However, excessive downtilting can result in insufficient coverage in a given area, with some UE unable to receive a minimum Reference Signal Received Power (RSRP).

Many existing solutions to downtilt adjustment often use rule-based algorithms to optimise the tilt angle based on the historical network performance. These rules are usually created by domain experts, and thus lack the scalability and adaptability required for modern cellular networks. Developments in Reinforcement Learning (RL) [2] introduce novel data-driven solutions where RET optimisation is performed by an agent that continuously performs tilt angle adjustments and observes the changes in the environment in order to learn the optimal action for a given network state [3]–[6].

However, when interacting with a sensitive environment, such as a cellular network, even smallest unsafe modifications can lead to drastic performance loss and disruption, which makes traditional RL solutions risky and impractical. Therefore, RL has found limited application in mission-critical use cases. These limitations lay grounds for the study of safe RL, where an RL agent is deployed in a real-world scenario with restricted exploration of the environment. The restriction can be achieved by various safety mechanisms that minimise the adverse effects of unsafe actions [7]. Our approach relies on existing baseline solutions that are known to be safe and are compared with the actions proposed by the RL agent. When the agents' actions are deemed *unsafe*, the baseline action is chosen instead.

In this paper, we propose a comprehensive modular architecture for safe RL and demonstrate its viability on the RET optimisation use case. Safety is defined with respect to a minimum performance level compared to one or more safe baselines that must be ensured at any time. This modular approach is applicable to different RL agents that can follow different models using a wide range of hyper-parameters. The architecture consists of a safety shield, safety logic modules, a set of known safe baselines, and one or more RL agents. The safety shield wraps around the target environment with the goal of decoupling the RL agent from the environment, thereby preventing the RL agent from directly and uncontrollably affecting the environment. The shield logic modules provide methods to evaluate action suggestions from RL agents and baselines, and to choose a safe action to be performed on the environment. The baselines are considered safe on a subset of the state-action space, but sub-optimal in terms of performance. For example, an individual baseline may be limited to a particular area of the network, and therefore be safe only for that subset of the environment. In those circumstances, our approach benefits from the ability to compare multiple baselines, from which one or more may be unreliable for a given network segment. The modularity of the proposed solution allows for customisation by modifying the shield logic module based on the use case and available agents and baselines.

II. Related Work

Recently, many works have explored the use of RL methods as a solution for the RET optimisation problem [3]–[6]. In [5], the authors maximise the UE's throughput fairness and energy efficiency of the network using an RL-based approach. [6] addresses a similar problem while expanding the scope to a group of cells, each with its own antenna configuration, resulting in an extremely complex problem. Both [3] and [4] use Fuzzy Reinforcement Learning methods, utilising fuzzy control theory through the use of predefined rules in order to model the inherent uncertainty of the observed states of cellular networks and reduce the state dimensionality. The main limitation with the majority of this work is that the solutions propose potentially unsafe actions that can have an adverse impact on the network.

Further, there has been a large body of safe methods proposed in the context of safe RL [7]. Examples of strategies used include the use of accumulated past knowledge [8], making conservative action choices that prioritise avoiding the worst-case scenario, or requesting guidance from an external agent or a human operator when the current state is considered too risky [9]. In the method proposed in [10], a baseline policy is trained and continuously updated by safe previous actions. Exploratory actions that are generated by adding Gaussian noise to agent's actions, are then benchmarked with baseline policy using a distance-based safety function. The ReQueST algorithm (Reward Query Synthesis via Trajectory) optimisation [11] attempts to build a generative model from hypothetical rewards collected from random exploration of the state space, which is then verified and updated by human feedback.

In [12], the definition of shielding is introduced where temporal logic, a model-based method, is proposed to filter out unsafe actions proposed by the RL agent. In [13], the authors consider an SRL method based on an actor-critic model where a safety threshold ε is progressively adjusted during the training, which only allows actions that are close to that of a safe baseline policy. In [14], the authors introduce a shield logic as a linear combination of suggested actions from the RL agent or the baseline with a parallel pioneer policy to prevent drastic changes in the control policy. Many of these works are tied to a specific RL algorithm, and none of them consider multiple baselines for increased safety in state spaces where an individual baseline might be inadequate. Our approach provides the flexibility to plug in various types of RL agent, baselines, and decision logic.

III. Reinforcement Learning

Reinforcement Learning (RL) [2] is a long-established decision-making framework, in which an agent interacts with an environment by exploring its states and selecting actions to maximise the long-term return based on a reward signal. An RL agent follows a policy π choosing the action to be performed at each state. The agent devises an optimal policy by dynamically optimising a value function that is estimated based on previous interactions of the RL agent. At each time step, the agent can choose to either follow the recommended action from the estimated best action (exploitation), or choose a random action to gather further information about the environment and update its knowledge (exploration).

There exist multiple approaches to find the optimal policy. Q-learning is one of the most popular RL algorithms that finds the optimal policy by estimating the state-action value function (Q-function) $Q^{\pi(s,a)}$, representing the expected cumulative reward when being in state s, executing action a, and following policy π afterwards. When Artificial Neural Networks (ANNs) are used to parameterise the Q-function, the algorithm is known as Deep Q-Network (DQN) [15]. DQN uses optimisation methods such as Stochastic Gradient Descent (SGD) to minimise Mean Squared Error (MSE) between target values and the parameterised Q-function.

Another popular approach to find the optimal policy is the Actor-Critic (AC) algorithm. AC uses ANN to parameterise two networks: the *actor network*, taking as input the state and returning the action to be performed on the environment, and the *critic network*, evaluating the action proposed by the actor by computing the value function. In this study we investigate the performance of DQN and AC as RL agents for our proposed safe RL architecture.

IV. A Safety Shield Architecture for Safe Reinforcement Learning

Our approach is an SRL model that uses a modular shield architecture with one or more RL agents and baselines. Figure 1 illustrates an overview of the proposed modular SRL structure. It comprises a set of online RL agents that are benchmarked against safe baselines through a safety shield that chooses the action to be performed on the environment.

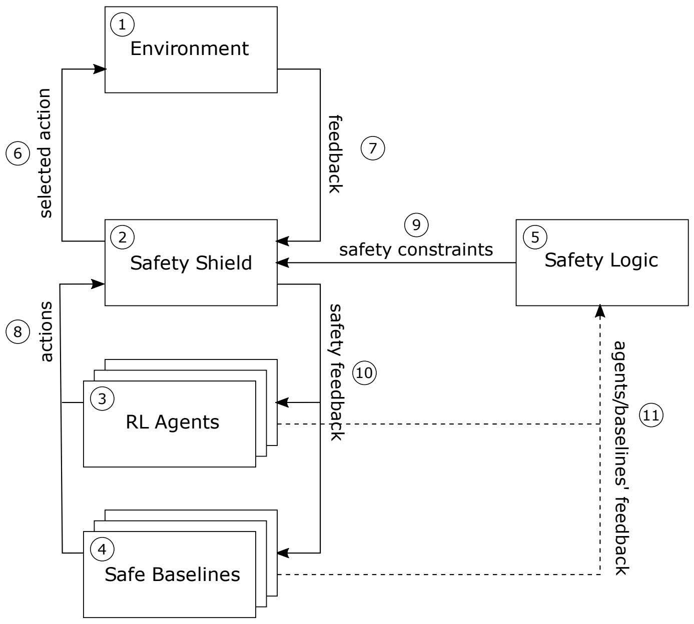


Fig. 1: Safe RL architecture with safety shield, logic and multiple baselines.

The architecture components are described using the numbers in Figure 1:

(1) **Environment**: This component represents the environment that the agent is interacting with, modelled as a standard RL problem [2].
(2) **Safety Shield**: This component is the mediator between the RL agents (3) and the environment (1). Contrary to the traditional RL model, where agents directly interact with the environment and receive a feedback, the safety shield acts as a proxy between agents and the environment to protect it from possibly unsafe actions proposed by the agents. The safety shield collects proposed actions by agents and baselines (8) and chooses a final safe action (6) to be performed on the real environment. The environment feedback (7) includes the actual action performed on the environment and is fed back to agents and baselines through the safety feedback (10).
(3) **RL Agents**: A set of online RL agents indirectly interact with the environment by proposing actions (8) to the safety shield. The agents are continuously trained during the RL interaction to improve their future recommended actions. Using different learning methods enables parallel experimentation with multiple techniques and parameters, to determine the suitable safe action amongst them.
(4) **Safe Baselines**: Baselines also receive feedback about the current state of the environment to suggest future actions. Contrary to the RL agent actions, baseline actions are considered safe on a subset of the state-action. Examples of safe baselines are rule-based algorithms that recommend actions satisfying safety rules designed by domain experts and supervised models trained on previous safe actions. The baselines are considered sub-optimal in terms of maximum performance but safe with regards to the minimum level of performance they provide at any given time.
(5) **Safety Logic**: The logic provides the safety constraints (9) to the safety shield, which are used to select the safe action from agents and baselines. The architecture supports various safety logic implementations, for example (i) using only the proposed actions from agents and baselines, where the safety logic chooses the best action by some safety criteria, or (ii) using agent and baseline internal variables such as convergence as additional information to gauge the safety of the proposed action. This is enabled through the feedback function. The shield logics proposed in this study are of type (i).

V. Antenna Tilt Optimisation using Safe RL

The definition of RET optimisation problem follows previous works for *Coverage-Capacity Optimization* (CCO) [1], [16], where the goal is to maximise network *coverage* and *capacity*, while minimising inter-cell *interference*. We define network measurements for capacity and coverage KPIs for time t and cell c to be $\textsc{Cov}_{t,c}$ and $\textsc{Cap}_{t,c}$ respectively. Also, the effect of negative interference on neighbouring cells is modelled by the quality KPI $\textsc{Qual}_{t,c}$. Note that above metrics are considered risk KPIs, i.e. high values indicate high risk for insufficient coverage, capacity or quality in a given cell.

TABLE I: Parameters for the simulator and the RL training.

Simulator parameter	Value
Number of BSs	7
Number of cells	21
Number of UEs	2000
Frequency	2 GHz
Traffic volume	20 Mbps
Antenna height	32 m
Minimum downtilt angle	1°
Maximum downtilt angle	16°
DQN Hyper-parameter	**Value**
Discount factor	0.0
Learning rate	0.001
Batch size	50
Test episode length	20
Number of evaluation episodes	25
AC Hyper-parameter	**Value**
Discount factor	0.0
Learning rate	0.03
Test episode length	20
Number of evaluation episodes	25

We now formulate the RET optimisation problem as a use case for the safe RL architecture introduced in the previous section, with the following components:

- *Environment*: consists of a simulated mobile network environment in an urban area. The parameters of this environment are detailed in Table I. Further, to accommodate for terminal stages of RL agents, the cell downtilt values are reset to random values at the end of each episode.
- *State space:* $\mathcal{S} \subseteq [0,1]^4$: comprises normalised values of *downtilt* θ, *coverage* $\textsc{Cov}$, *capacity* $\textsc{Cap}$ and *quality* $\textsc{Qual}$ for time t and cell c., i.e. $s_{t,c} = [\theta_{t,c}, \textsc{Cov}_{t,c}, \textsc{Cap}_{t,c}, \textsc{Qual}_{t,c}]$.
- *Action space:* $\mathcal{A} = \{-1, 0, 1\}$: consists of discrete changes of the current downtilt.
- *Reward:* $r_{t,c} = -\log(1 + \textsc{Cov}_{t,c}^2 + \textsc{Cap}_{t,c}^2 + \textsc{Qual}_{t,c}^2)$: consists of the squared log-sum of coverage, capacity, and quality KPIs.
- *Safe baselines*: we considered two baselines for the experiments conducted for this study:
  - *Rule-based baseline:* this baseline is based on a legacy RET algorithm currently in place for antenna tilt optimisation that suggests antenna tilt actions based on $\textsc{Cov}_{t,c}$, $\textsc{Cap}_{t,c}$ and $\textsc{Qual}_{t,c}$.
  - *Model-based baseline:* this baseline is generated from a DQN-based offline learner trained on historical data of performed antenna tilts.
- *RL Agents:* In the experiments conducted for this study, we use an RL agent using the DQN or the AC algorithm. The training parameters of these agents are given in Table I. The same policy is independently applied to all cells to suggest a specific action for each cell.

In this setting, we study the performance of two safety logics, namely State Predictor Shield logic and K-Shield logic that are described below.

A. State Predictor Shield Logic

The State Predictor shield logic uses a pre-trained supervised learning model to assess the value of the actions proposed by the different agents and baselines.

We train a multi-target regressor on historical data from the environment, where the input for the model is defined as a given environment state plus a tilt change action $(\textsc{Cov}_{t,c}, \textsc{Cap}_{t,c}, \textsc{Qual}_{t,c}, \theta_{t,c})$, and the output is defined as the environment state after applying the action $(\textsc{Cov}_{t+1,c}, \textsc{Cap}_{t+1,c}, \textsc{Qual}_{t+1,c})$. The regressor in our experiments is a fully connected multi-layer perceptron.We also experimented with extended state-space, including additional KPIs, but that did not noticeably impact the prediction accuracy of the model.

With a sufficiently large sample size, a model can be trained that is reasonably accurate in predicting the likely change in environment state for each of the proposed actions. In our experiments, we train the supervised State Predictor model on data that we synthesised from the simulation environment by randomising environment parameters to create a varied set of data points with different antenna tilt angles and states. Interestingly, a model trained on data that comes from a real-world network performed similarly well, which underlines the viability of using predicted network KPIs for the purpose of ensuring safety of the RL agent.

The State Predictor shield logic uses this pre-trained model to compare the actions proposed by agents and baselines by querying the model with each action in addition to the current environment state. From the predicted next state, the shield logic can calculate various metrics to score the proposed actions, such as a reward or a safety constraint. In our experiments, the safety logic compares proposed actions by using the state KPI directly. It will choose the action that achieves the best predicted state KPI and forwards it to the shield instance for execution on the environment.

B. K-Shield Logic

The logic in this scenario is inspired by the approach proposed in [14] and is generalised to accommodate multiple safe baselines. We assume there exists an RL agent that is trained next to a set of safe baselines. The logic chooses the action based on a weighted combination of suggested actions from the agent and baselines.

Formally, the control policy π_C is a linear combination of the RL policy π_L and B safe baseline policies π_s^i as follows:

$$\pi_{\mathcal{C}} = p \sum_{i=1}^{B} \omega^i \pi_s^i + (1-p)\pi_L \tag{1}$$

The parameter $k \in [0,1]$ determines the probability of choosing (one of) the safe baselines over the RL policy. Formally, when selecting between the safe baselines or the RL policy, we sample $p \sim \mathcal{B}(k)$, i.e. a Bernoulli random variable with parameter k. The probability k is initially close to 1 to ensure that the control policy is dominated by safe baselines when the RL policy has not accumulated enough experience to be able to execute safe actions. Subsequently, its value is reduced by a diminishing factor $d \in (0,1)$ every time the reward during previous episodes is improved compared to the past w episodes. Each episode is indexed as $e = 1, \ldots, E$. The update is executed each w episodes as:

$$k_{e+1} = \begin{cases} \max\{0, k_e - d\} & \hat{R}_{[e-w,e]} \geq \hat{R}_{[e-2w,e-w-1]} \\ k_e & \text{otherwise} \end{cases} \tag{2}$$

where $\hat{R}_{[i,j]}$ denotes the average reward calculated from episode i to episode j.

The baselines' probabilities $b_i \in [0,1] : \sum_{i=1}^{B} b_i = 1$ are hyper-parameters controlling the importance of each of the baselines. If $p = 1$, we select one of the baselines $(\omega^i = 1)$ with probability b_i, for each of the baselines. The baselines probabilities can be either:

- preset values based on the performance of safe baselines, and defined by domain experts or use-case stakeholders (e.g., network operators in RET optimisation), or
- they can be state-action dependent, allowing higher weights for policies that outperform others in certain sub-spaces of the problem (e.g., if a safe baseline has higher performance on a sub-region of a cellular network).

In the experiments conducted for this study, we investigate scenarios where an RL agent competes with a single baseline and with multiple baselines with predefined weights. The results of experimentation with different weights are detailed in the next section.

VI. Evaluation Results

The results discussed in this section present the outcomes of our shielded safe RL experiments using the modular shield architecture described in Section IV. We ran each experiment with 6 random seeds, and each plot shows the average of all random seeds (continuous and dashed lines), as well as the minimum and maximum values that were reached over all seeds (shaded areas). We applied a running average window to smooth out the curves for better visual comprehension. Finally, we focus on the early phase of RL training, in which exploration is prevalent and therefore most unsafe behaviour of an unrestricted RL agent is to be expected. The long-term performance advantages of RL over traditional approaches have already been shown in the literature, e.g. [3]–[5].

The following elements will be shown in the plots:

1) An unrestricted RL agent, with unrestricted access to the environment,
2) One or more baselines, which are considered sub-optimal but safe,
3) A safe RL agent that is governed by the safety shield using a shield logic.

The plots focus on reward (higher is better), coverage and quality risk KPIs (lower is better) The capacity KPI is omitted as it was not impacted by antenna tilts across all experiments.

A. Comparison of unrestricted RL agent with baseline and safe RL agent

In this section we compare the performance of unrestricted RL agents with baselines and with the safe RL agents using the two shield logics described in sections V-A and V-B.

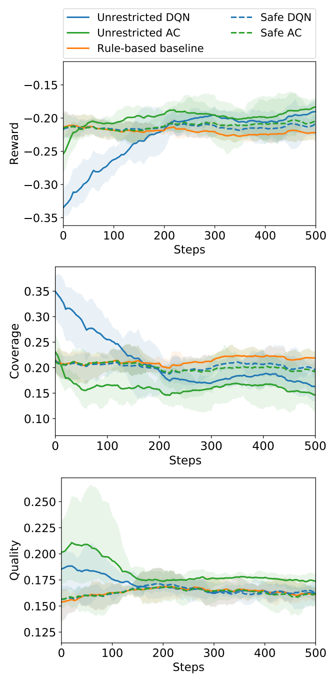
The State Predictor logic uses a pre-trained machine learning model to estimate the impact of the proposed tilt change action on the network as described in Section V-A.

Figure 2a (top) shows the immediate reward that is returned by the environment. We observe that the unrestricted DQN agent (solid blue) starts off with a sub-optimal reward, which is significantly worse than the performance of the safe baseline. When the safety shield is applied, the safe DQN agent (dashed blue) closely mimics the baseline behaviour and slightly outperforms the baseline early on. This is the desired effect that the safety shield with the State Predictor logic was designed to accomplish.

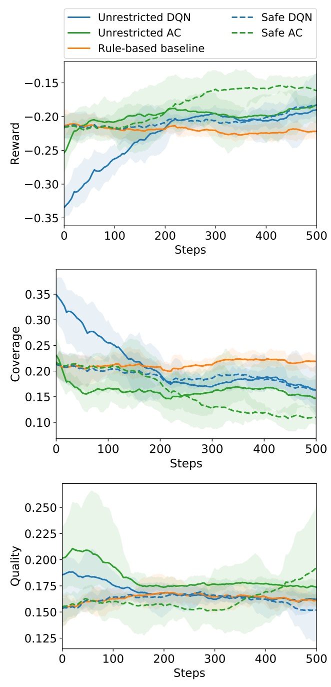
Figure 2a (middle and bottom) focuses on coverage and quality, which are two important performance indicator. In our experiments, coverage and quality are risk-based KPIs, which means that lower values indicate better network conditions. In both cases, it is apparent that the unrestricted DQN agent starts off at a much higher risk level before it learns the environment and its performance is improved, although the effect is more pronounced on the coverage KPI. The safe baseline (orange) is at a safe level from the start with little fluctuation, where the safe level of performance is set by network operator experts. The State Predictor shield logic causes the DQN agent to follow the baseline performance more closely from the start, thus creating a safe agent interaction with the environment. In the case of coverage, the safe agent slightly outperforms the baseline on average early on.

Before comparing the performance of the **K**-Shield logic, we first tune the diminishing factor d and the average reward window size w as described in Section V-B. We compared the performance of the safe DQN agent using K-Shield logic for for $d \in \{0.1, 0.2\}$ and $w \in \{2, 5\}$. While the results show very small difference, a smaller d indicates a more gradual transfer to RL agent's policy, allowing to rely on safe baselines for a longer period of the training. Also smaller w indicates more frequent comparison between average rewards over previous episodes, which allows quicker adaptation to current rewards, which in turn improves the performance of the agent governed by K-Shield logic.

Following this observation, $d = 0.1$ and $w = 2$ are selected as optimal values for K-Shield logic parameters that will be used for the K-Shield logic experiments. Figure 2b (top) shows the reward that is returned by the environment over time. As it can be observed, the safe DQN agent with K-Shield logic (dashed blue) follows the safe baseline policy during the period in which the performance of unrestricted DQN (blue) is inferior to that of the the safe baseline. It then gradually switches towards unrestricted DQN agent as the model is further trained to achieve greater rewards than the safe baseline.



(a) State Predictor Logic.



(b) K-Shield Logic.

Fig. 2: Comparison of reward, coverage and quality for DQN and AC using unrestricted agents, baseline only and safe agents governed by the shield logic.

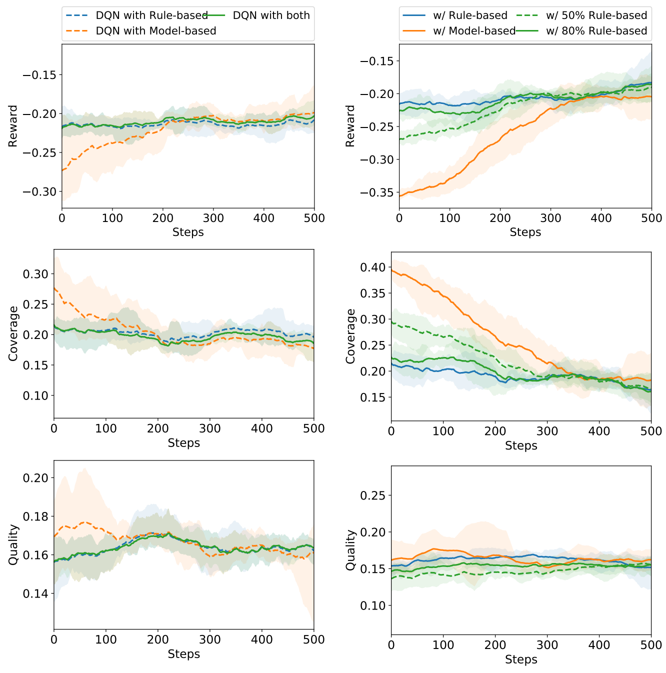
Further, similar to the State Predictor logic, for coverage and quality KPIs (Figure 2b middle and bottom), the K-Shield logic prevents the DQN agent from performing high-risk actions by following the safe baseline policy at the start of the training phase, where the DQN agent performs mostly exploratory actions, and subsequently switching to the DQN actions when it has gathered enough experience to execute a safe action. This behaviour is more visible for the coverage KPI.

We also compare the unrestricted and safe AC agent to the unrestricted and safe DQN agent for both the State Predictor logic (Figure 2a, green) and the K-Shield logic (Figure 2b, green). We observe that the trend for reward, coverage and quality KPIs when using a DQN agent is similar to when the safety logic is used to restrict actions of the AC agent, although the range of values for the two agent types differs.

B. Comparison of multiple baselines

In this section, we compare different baselines in a scenario where the shield logic has access to multiple baselines and can choose between them according to its strategy. The two baselines are Rule-based and Model-based, as described in Section V. To keep the plots readable, these experiments focus the DQN agent only.

Figure 3a shows a comparison of the received rewards and the achieved risk KPIs when training a DQN learner on a shielded environment where the State Predictor logic chooses between different baselines. In the network envi-



(a) State Predictor Logic

(b) K-Shield Logic

Fig. 3: Comparison of reward, coverage and quality KPIs for shield-logic governed DQN using rule-based, model-based and both baselines together.

ronment considered for this study, the model-based baseline (dashed orange) performs worse than the alternative baseline (dashed blue). However, when providing the State Predictor shield logic with both baselines, it is able to select the best proposed action based on the predicted state (green).

Figure 3b compares the reward, coverage and quality KPIs of a K-Shield logic restricted DQN agent using both Rule-based and Model-based baselines for different importance weights. It can be seen that in this scenario, where the Rule-based baseline is outperforming the Model-based baseline, higher weights on the Rule-based baseline result in faster convergence to better reward (top) and coverage (middle). This is less apparent for quality (bottom) as the difference between the performance for different scenarios is very small and within the margin of error.

VII. Conclusions

In this paper, we propose a modular architecture for safe RL that enables safe interaction with the environment by benchmarking recommendations of RL agents to multiple safe baselines. The architecture allows experimentation with multiple RL agents, using different models and hyper-parameters, and leverages multiple baselines that may be safe in subsets of the state-action space to create a policy that is safe across the whole state-action space and outperforms individual unrestricted RL agents and baselines. As a concrete use case for our architecture, we study RET optimisation, which is evaluated using a simulated environment that is configured to recreate a real-world urban network.

The safety and performance of the approach using multiple shield logics is compared with an existing rule-based SON algorithm and an offline model that is trained on historical data. The results show that, compared to safe baselines, unrestricted RL agents explore unsafe regions of the state-action space, specially at the start of the RL learning process. The safety shield allows RL agents to start learning in safe states (performing similar to safe baselines) with the help of the shield logics, and eventually outperform safe baselines in many cases, especially for the received reward and the coverage KPI.

A limitation to this approach is the availability of safe baseline(s). Without that, the safety shield would not be able to decide whether a proposed action is safe enough for the environment.

References

[1] V. Buenestado *et al.*, "Self-tuning of remote electrical tilts based on call traces for coverage and capacity optimization in LTE," *IEEE Transactions on Vehicular Technology*, 2017.

[2] L. P. Kaelbling, M. L. Littman, and A. W. Moore, "Reinforcement learning: A survey," *Journal of artificial intelligence research*, 1996.

[3] R. Razavi, S. Klein, and H. Claussen, "A fuzzy reinforcement learning approach for self-optimization of coverage in lte networks," *Bell Labs Technical Journal*, vol. 15, no. 3, pp. 153–175, 2010.

[4] S. Fan, H. Tian, and C. Sengul, "Self-optimization of coverage and capacity based on a fuzzy neural network with cooperative reinforcement learning," *EURASIP Journal on Wireless Communications and Networking*, vol. 2014, no. 1, p. 57, 2014.

[5] W. Guo *et al.*, "Spectral-and energy-efficient antenna tilting in a hetnet using reinforcement learning," in *2013 IEEE Wireless Communications and Networking Conference (WCNC)*. IEEE, 2013, pp. 767–772.

[6] E. Balevi and J. G. Andrews, "Online antenna tuning in heterogeneous cellular networks with deep reinforcement learning," *IEEE Transactions on Cognitive Communications and Networking*, vol. 5, no. 4, pp. 1113–1124, 2019.

[7] J. Garcıa and F. Fernández, "A comprehensive survey on safe reinforcement learning," *Journal of Machine Learning Research*, vol. 16, no. 1, pp. 1437–1480, 2015.

[8] M. Ghavamzadeh, M. Petrik, and Y. Chow, "Safe policy improvement by minimizing robust baseline regret," in *Advances in Neural Information Processing Systems*, 2016, pp. 2298–2306.

[9] L. Torrey and M. E. Taylor, "Help an agent out: Student/teacher learning in sequential decision tasks," in *Proceedings of the Adaptive and Learning Agents workshop (at AAMAS-12)*, 2012.

[10] F. J. G. Polo and F. F. Rebollo, "Safe reinforcement learning in high-risk tasks through policy improvement," in *2011 IEEE Symposium on Adaptive Dynamic Programming and Reinforcement Learning (ADPRL)*. IEEE, 2011, pp. 76–83.

[11] S. Reddy *et al.*, "Learning human objectives by evaluating hypothetical behavior," *arXiv preprint arXiv:1912.05652*, 2019.

[12] M. Alshiekh *et al.*, "Safe reinforcement learning via shielding," in *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

[13] V. Raghunath, "Antenna tilt optimization using Reinforcement Learning with doubly-safe exploration," MSc Thesis, KTH Royal Institute of Technology, 2020.

[14] Y. Zhang, D. Balkcom, and H. Li, "Towards physically safe reinforcement learning under supervision," *arXiv preprint arXiv:1901.06576*, 2019.

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

[16] F. Vannella, J. Jeong, and A. Proutiere, "Off-policy Learning for Remote Electrical Tilt Optimization," *IEEE 92nd Vehicular Technology Conference (VTC Fall)*, 2020.