# ML 5G-PHY-Reiforcement Learning: Scheduling and Resource Allocation

Khalid Albagami
Ericsson AB
k.albagami@hotmail.com

Abdullah Alajlan
Ericsson AB
abdullah.alajlan@ericsson.com

Bayan Aloquibi
Ericsson AB
Bayan.aloquibi@ericsson.com

Raghad Alturki
Ericsson AB
Raghad.alturki@ericsson.com

## 1. Introduction

With the proliferation of more smartphones and high data rate applications, numerous new applications have been emerging worldwide to satisfy the large demand for mobile data services. Massive multiple-input-multiple-output (MIMO) technique is a key for 5G networks due to its promising high data rate and system throughput. Yet, massive MIMO has major challenges with regards to high computational complexity and great difficulties in the exploitation of the multiple antenna systems. Selecting the best beam to communicate to a user under certain radio conditions, or what so called beamforming, is a critical process and one of the challenges in MIMO deployment. This problem intrigued researchers to employ the artificial intelligence (AI) in MIMO system for 5G network, and particularly the problem of beam selection.

The reinforcement learning (RL) is a machine learning method based on rewarding desired behaviors and/or punishing undesired ones. It is a learning paradigm that can perceive and interpret the environment, take actions, and learn through trial and error. This kind of interaction with the surrounding environment makes RL apt to be considered for MIMO implementation in 5G cellular networks. Successful application of RL on 5G networks problems requires a large data to train and evaluate the RL agent. Therefore, the use of virtual worlds has been studied by researchers to generate datasets by creating realistic representations of deployment sites together with physics and sensors simulations [1]. This enables the RL agent training for tasks such as user scheduling and beam selection.

The beam selection problem in this challenge is modeled as a game that must be solved with RL, and is based on a simulation methodology named CAVIAR, proposed in [2]. The CAVIAR simulation integrates three subsystems: the communication system, the AI and ML models, and finally the virtual reality (VR) or augmented reality (AR) components. In this challenge, the problem is based on simulating a communication system immersed in a virtual world created with AirSim and Unreal Engine as shown in Figure 1 [3-paper]. The goal is to schedule and allocate resources to Unmanned Aerial Vehicles (UAVs), cars and pedestrians, composing a scenario with aerial and terrestrial users equipment (UEs). The base station (BS) executes the RL agent and periodically takes actions based on the information captured from the environment which includes channel estimates, buffer status, and positions from a Global Navigation Satellite System, such as GPS. Then, the RL agent is rewarded based on the quality of service provided to the users.

*Figure 1: Picture taken from the simulated CAVIAR environment*

## 2. Dataset

In this section, we will demonstrate the different data types collected from running the simulation environment for multiple episodes of time in [4].

### 2.1. Pre-calculated environment data

The environment contains many obstacles (buildings, trees etc.) and 37 UEs (user equipment) divided into three categories: UAVs (unmanned aerial vehicle), cars and pedestrians.
A sample of the pre-calculated data collected in [4] to be used to train the agent is shown in Figure 2, Figure 3 and Figure 4. The data focus on pointing out the location and whereabouts of the different UEs while moving through the environment. Also, it provides details on the UE orientation, velocity and acceleration in x, y and z coordinates.

| timestamp | obj | pos_x | pos_y | pos_z | orien_x | orien_y | orien_z | orien_w |
|---|---|---|---|---|---|---|---|---|
| 1.62681E+18 | uav1 | 14.00067585 | -28 | 6.395352745 | -4.93E-05 | 8.26E-12 | 1 | 3.01E-07 |

*Figure 2: Position and orientation pre-calculated sample data*

| linear_acc_x | linear_acc_y | linear_acc_z | linear_vel_x | linear_vel_y | linear_vel_z | angular_acc_x | angular_acc_y | angular_acc_z |
|---|---|---|---|---|---|---|---|---|
| 0.000928302 | -4.35E-10 | 0.34125137 | -0.001040355 | 8.31E-10 | 0.02116608 | -2.08E-12 | -0.001808743 | 7.46E-17 |

*Figure 3: Continued Linear and angular acceleration/velocity pre-calculated sample data*

| angular_vel_x | angular_vel_y | angular_vel_z |
|---|---|---|
| -5.63E-12 | 0.000140757 | 1.57E-08 |

*Figure 4: Continued angular velocity pre-calculated sample data*

The pre-calculated data is collected for several episodes each episode spans for 3 minutes of time and the sampling rate of data is 10 ms.

## 2.2. Live environment data

The second data type is frequently collected each 10 ms while the agent is training and trying to solve the environment. The live environment data provides information about transmitted, buffered and dropped packets. It also contains information about the channel magnitude and bitrate calculated for each transmission interval 10 ms.

## 2.3. Output

In this environment, three UEs out of 37 are considered at each transmission interval. The RL agent then schedules one UE to the selected beam.

The agent goal is to provide the best optimal beam and selected UE that will minimize the dropped packets and maximize the transmitted packets.

# 3. Models and Model training

This section presents the different models selected to train the agent to beat the environment and their training cumulative rewards (Return) G results.

After exploring the available state-of-the-art reinforcement learning (RL) models, three techniques are considered: advantage actor critic (A2C), deep Q-networks (DQN) and Q-learning.

Then, we tried to see if all the above models will be suitable and applicable to be used with the environment states' complexity level. Since Q-learning is the least agile model when it comes to supporting continuous many input state variables in the observation space. We excluded and focused on training and testing the other three models in the list.

After training the agent for 100 episodes DQN and A2C models results are quite similar. Since there is almost no difference in the results of the two models, we decided to focus on training and testing the agent on one model (A2C).

We then started to look for the various ways that we can improve the baseline agent performance shown in Figure 5. We decided that we should tackle the observation space to enhance the cumulative rewards (Return) G.

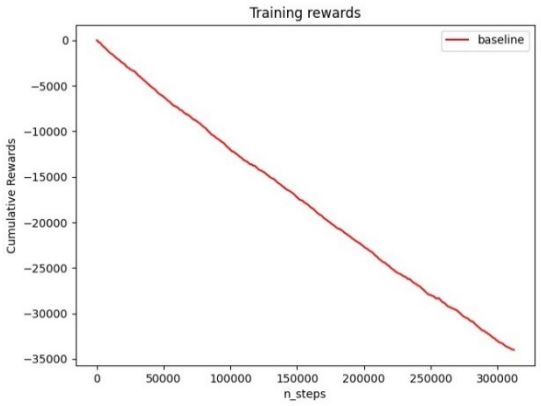


*Figure 5:Baseline Training cumulative rewards vs number of steps results for 50 episodes*

Then we created multiple scenarios for the recommended chosen environment observation space and trained the agent on these scenarios as shown in Table 1.

***Table 1**: Scenarios to improve cumulative rewards G*

| Scenario | Observation Space | Number of states |
|---|---|---|
| A | Channel Magnitude, target UE, Transmitted Packet | 3 |
| B | Channel Magnitude, target UE (x,y,z) position | 4 |
| C | Transmitted, buffered, dropped packets. | 3 |
| D | (x,y,z) position for target UE and other UEs, target UE buffer and other UEs buffer | 12 |
| E | Target UE, target UE buffer, other UEs buffer. | 4 |
| Baseline | Target UE (x,y,z) position, transmitted, buffered, dropped packets, and bits rate | 7 |

The graphs in Figure 6 show the training cumulative rewards (Return) G with respect to the number of timesteps the agent took to train.

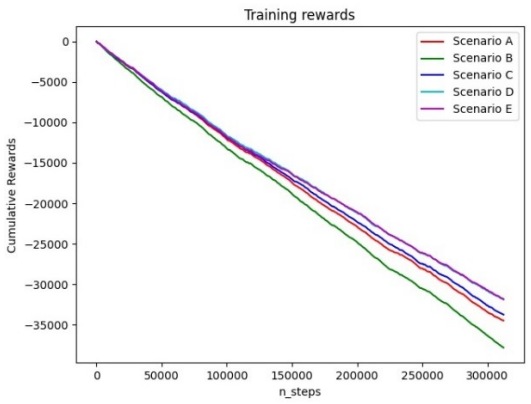


*Figure 6: Scenario A,B,C,D and E Training cumulative rewards vs number of steps results for 50 episodes*

From Figure 6, scenarios D and E have a less decaying slope for G than the other scenarios. But scenario E is achieving the same results of scenario D with less number of states. So, the final selected model is scenario E.

## 4. Results:

This section shows a brief overview of the changes we made and the results we had. We adjusted the Hyper Parameters – Mainly Learning rate LR & Gamma – to see the effect and try to tune the agent as much as possible, we noticed that setting gamma at 0.99 shows the best results. On the other hand, when we set a higher value for LR – near to zero – we obtain more robust results with less exploration, and the agent uses only one or two actions no matter how the state varies, we tried several values until reaching what seems to be a good value for LR at 0.001.

After enhancing the performance slightly, we considered other ways to enhance learning of the agent. As reinforcement learning depends highly on the state, action and reward, and since we are bound to the given actions, we enhanced the model by changing the states.

In Table 2, the final selected hyperparameters used during training and testing.

***Table 2:*** *A2C Model Hyperparameters*

| | |
|---|---|
| **learning_rate** | 1e-3 |
| **n_steps** | 2 |
| **n_env** | 100 |
| **n_batchs** | 200 |
| **gamma** | 0.99 |
| **policy** | MlpPolicy |

Once we tried all the possible options that could improve the performance, we collected the best results and compared them with the baseline as shown in Table 3.

***Table 3****: Agent performace comparision for selected model and baseline trained on Eq1*

| | **Baseline** | **Scenario E** |
|---|---|---|
| Training Accumulative Rewards ($G_T$) for 350 eps | -218376 | -214158 |
| Validation Accumulative Rewards ($G_V$) for 150 eps | -142800 | -143600 |
| Test Accumulative Rewards ($G_{Test}$) for 200 eps | -175100 | -184500 |

## 5. References

[1] E. Egea-Lopez, F. Losilla, J. Pascual-Garcia, and J. M. Molina-Garcia-Pardo. Vehicular networks simulation with realistic physics. IEEE Access, 7:44021–44036, 2019.

[2] A. Oliveira, F. Bastos, I. Trindade, W. Frazão, A. Nascimento, D. Gomes, F. Müller, and A. Klautau.Simulation of Machine Learning-Based 6G Systems in Virtual Worlds.Submitted to ITU Journal onFuture and Evolving Technologies, 2021

[3] ML 5G-PHY-Reiforcement Learning: Scheduling and Resource Allocation (The challenge report).

[4] Aldebaro Klautau, Ailton Oliveira, Isabela Trindade, Wesin Alves. Generating MIMO Channels For 6G Virtual Worlds Using Ray-tracing Simulations. IEEE Statistical Signal Processing Workshop, 2021