Machine Learning for Edge-Cloud Systems (**ML4ECS**)

In conjunction with **HiPEAC 2025**

Reinforcement Learning Training Strategies for 5G Networks Latency Optimization

Massimiliano Rossi
NTT DATA Italia

Andrea Pazienza, PhD
NTT DATA Italia

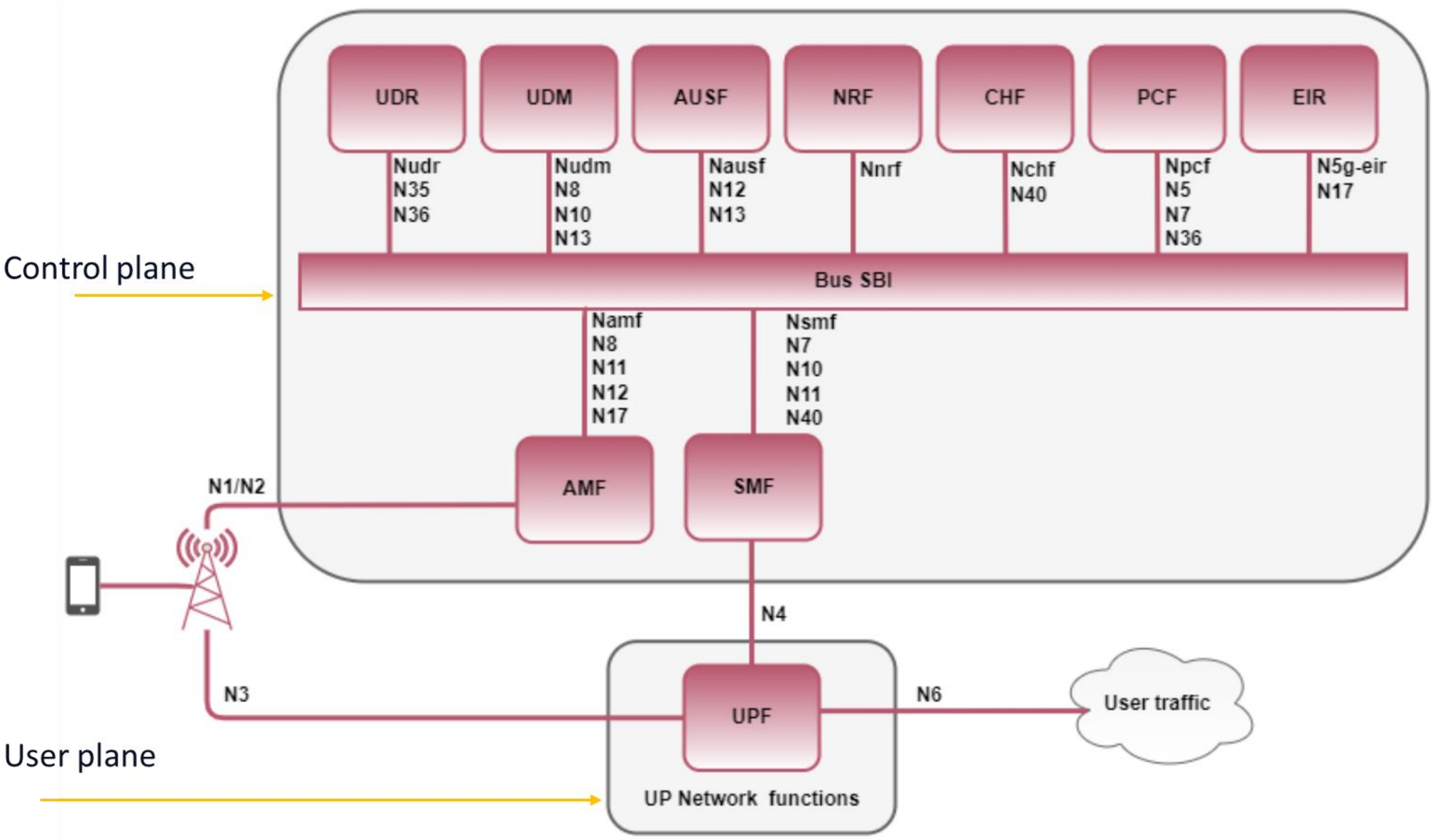
22 January 2025
Barcelona, Spain

The research leading to these results has received funding from the European Community's Horizon Europe Programme under the MLSysOps Project, grant agreement 101092912.

Key points

- European funded project with a consortium of 12 enterprises and universities around EU
- Objective is to apply ML algorithm to system operation
- Our focus is to analyze telemetries coming from edge data centers and identify the best edge on which moving 5G user plane
- Leverage on 5G network slice for minimizing latency and for obtaining the best end user experience

5G Core Network Architecture



MLSysOps overview

The use case aims to optimize latency in the 5G user plane by applying Machine Learning to infrastructure metrics and automating system operations

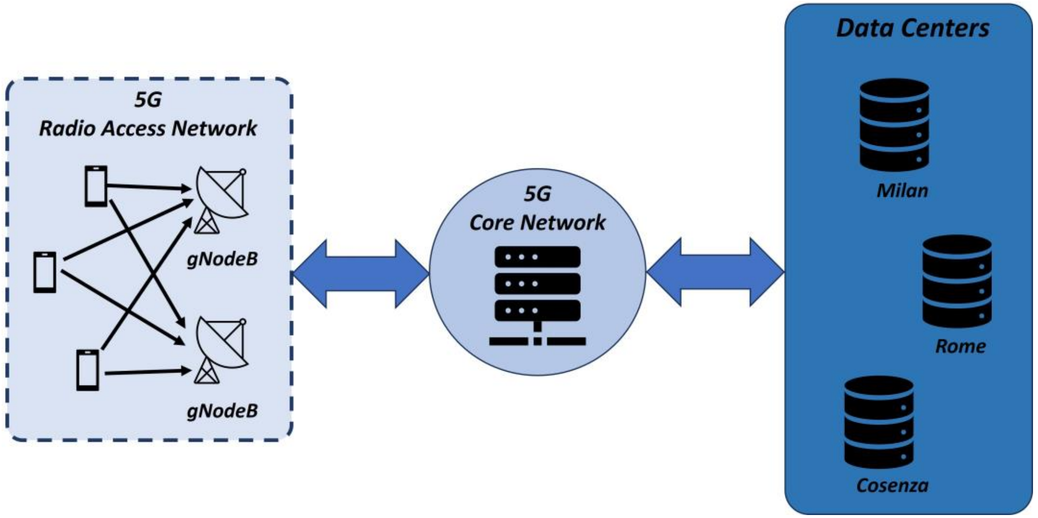
Metrics evaluated

- We kept a platform agnostic approach, so we didn’t use UPF metrics that can vary between UPF vendors
- For this reason an agent runs on a separated VM located in the same environment of UPF and measures:
  - CPU usage [%]
  - Memory usage [%]
  - Disk usage [%]
  - Net in/out absolute [kbps]
  - Net in/out [%]
  - Latency min/max/avg/mdev [ms]
  - Packet loss [%]
- Total bandwidth is statically configured in the Agent
- Agent uses the same network interface of the UPF, so that measurements are reliable

Agents

- Deployed an agent co-located with each UPF
- The agents are programmed in Python and use different software modules for metrics collection.
- Each agent collects metrics at configurable intervals and sends them to a REST API, including agent ID and timestamp.
- The collector ensures data consistency and aggregates metrics from all agents based on ID and timestamps.
- Once all measurements are received for a specific interval, the aggregated data is sent to the ML algorithm for decision-making. A local copy of the data is also saved.

Reinforcement Learning Methodology



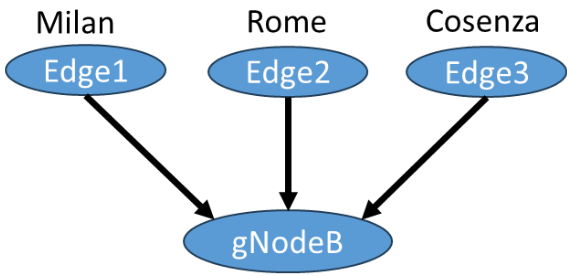
CPU	Memory	Disk	Net in	Net out	Latency avg	Latency mdev	Packet loss
0.6	0.5	0.5	0.7	0.7	1	0.2	0.9

$$\sum_{i=1}^{n} \frac{W(i)}{1+D(i)}$$

- **Edge Nodes**: The key geographical sites for this study are the data centers in Milan, Rome, and Cosenza, representing diverse network conditions.
- **Dataset**: Collected from multiple data centers, simulating varied traffic profiles (e.g., Night, Busy Hour, Daytime) and introducing real-world constraints like bandwidth caps and packet loss.
- **Goal**: To minimize latency while balancing other KPIs such as CPU and memory utilization across edge nodes.
- **RL Agent Design**: The agent's task is to select the optimal data center for minimizing latency, considering various network performance features:
  - Latency (average, mean deviation [ms])
  - CPU, memory, and disk usage [%]
  - Network traffic (net in, net out, packet loss [%])
- **Reward Function**: The agent's reward is calculated based on a weighted sum of key KPIs, prioritizing latency and packet loss.

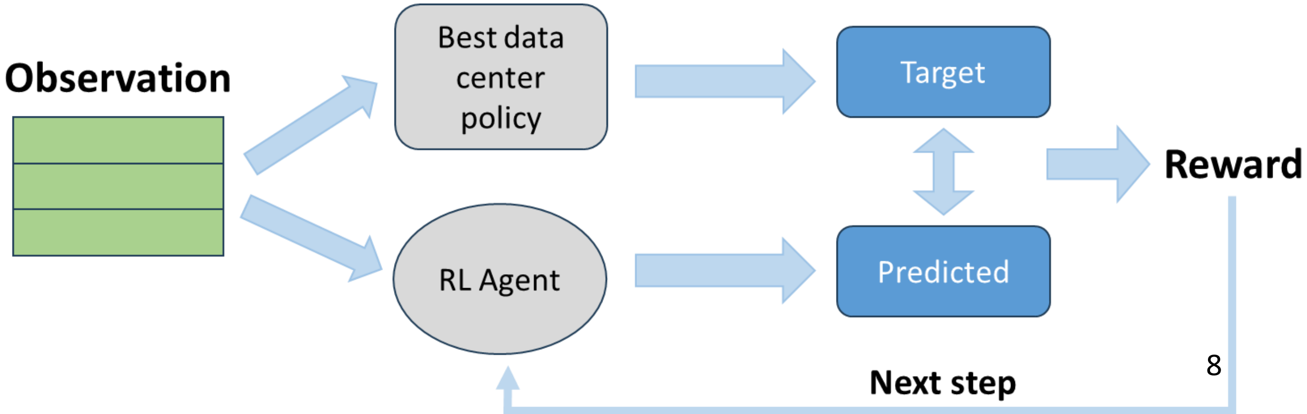
RL Algorithms for 5G Network Optimization

- **RL architectures**:
  1. Deep Q-Network (**DQN**),
  2. Proximal Policy Optimization (**PPO**),
  3. Advantage Actor-Critic (**A2C**).
- **RL Environment**: Developed using Python's **Gymnasium** library.
- **Key Elements**:
  - Environment definition tries to follow the main rules of the 5G network.
  - Each of the three edges shares data with the gNodeB at the same time.
  - Each model was trained using real-time data traffic and fine-tuned through:
    - ✓ Learning rate
    - ✓ Batch size
    - ✓ Discount factor (**γ**)



Observation space
A matrix of 3 rows corresponding to the 3 edges measurements are used as observation state. Each observation contains data from each edge, e.g. *Latency1, Latency2, Latency3, CPU1, CPU2, …, …, PacketLoss1, …*

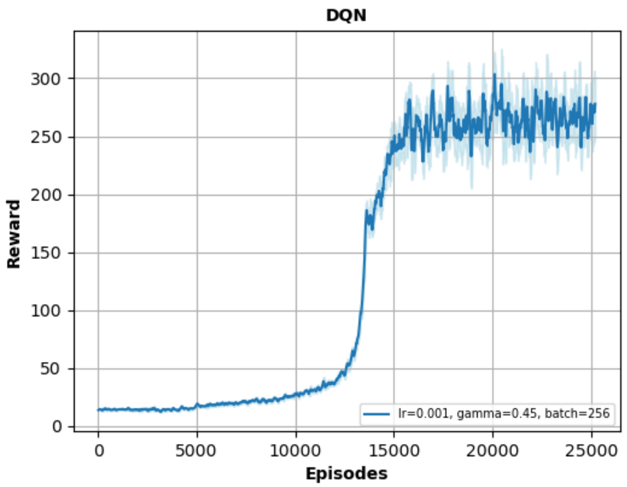
Action space
Selection of an edge node (*Milan*, *Rome*, *Cosenza*).

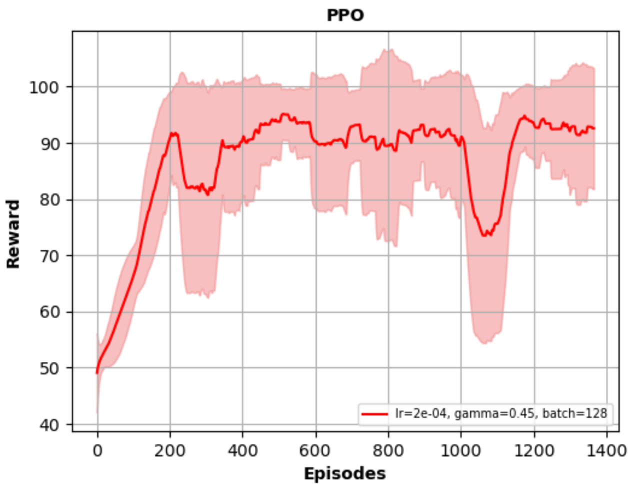


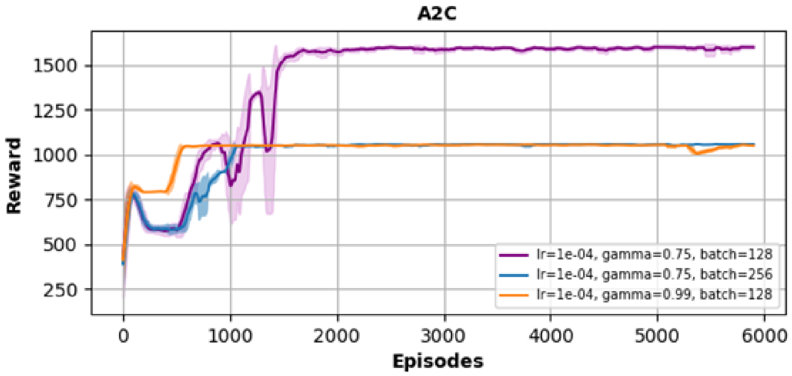
Results and Evaluation

- **DQN** emerged as the most effective algorithm, achieving the highest performance with a maximum reward of **338** and stable convergence.
  - Best configuration: **Learning rate** = **0.001**, **γ** = **0.45**, **Batch size** = **256**.

- **PPO** and **A2C** models demonstrated slower or unstable convergence, indicating they are less suited to the task compared to DQN.







New RL Training Strategies to avoid Overfitting

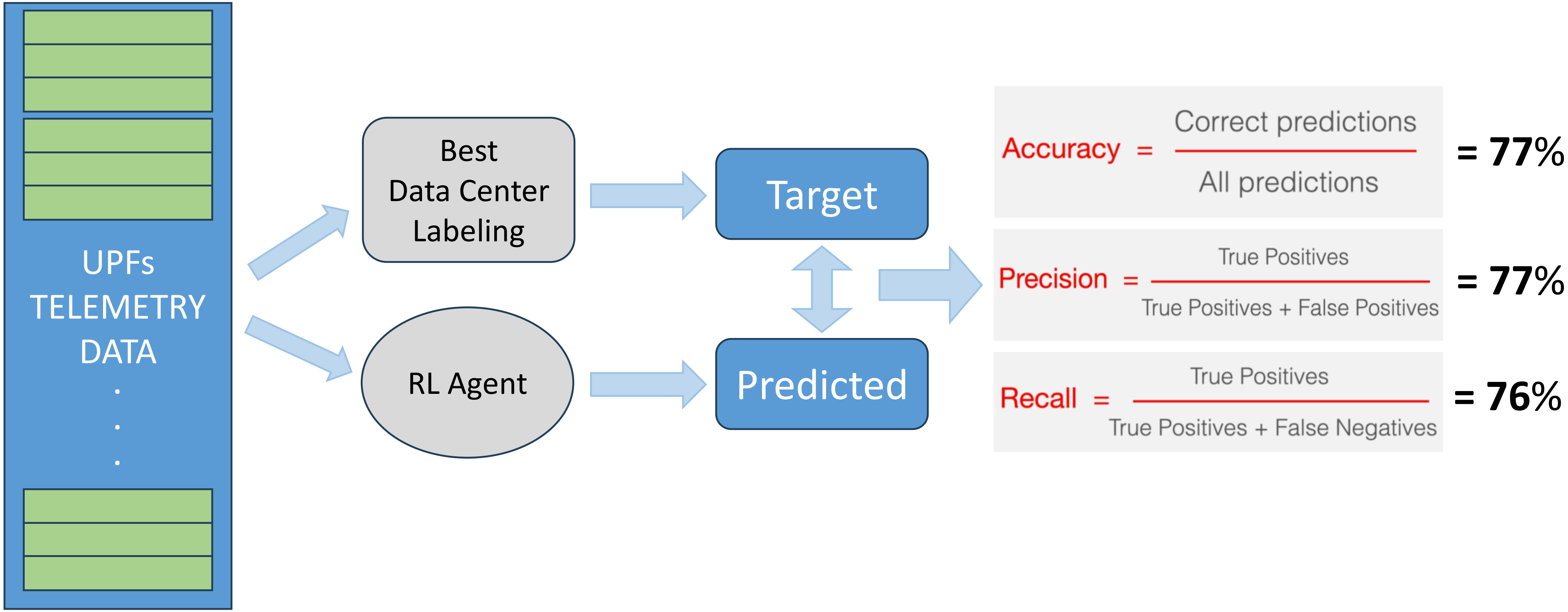
1. **Optimization of UPF Selection Criteria**

   The implemented constraints for UPF selection are:
   - **CPU Usage**: must remain below 90% threshold
   - **Packet Loss**: new UPF must ensure a minimum 20% reduction compared to previous UPF

2. **Reward System Refinement**

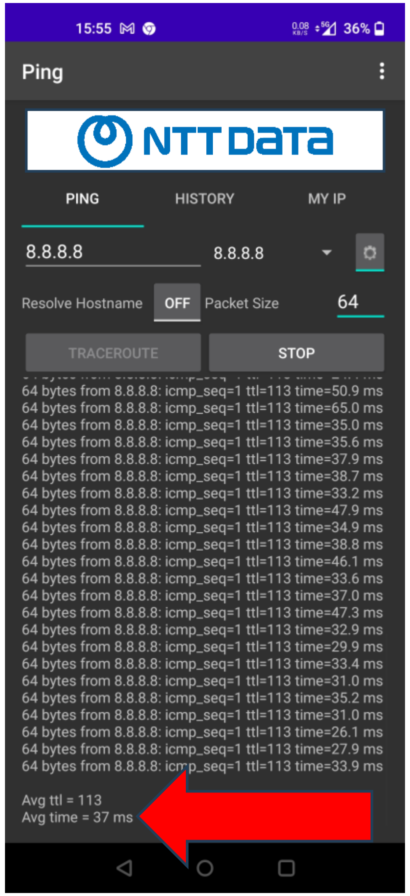
   The agent's reward system has been enhanced considering:
   - **Performance Metrics**
     - **Latency**:
       - Positive reward if below dataset mean
       - Penalty if above
     - **Packet Loss**:
       - Positive reward if below dataset mean
       - Penalty if above
   - **Selection Accuracy**
     - Significant bonus if selected UPF matches optimal target index

Classification Metrics

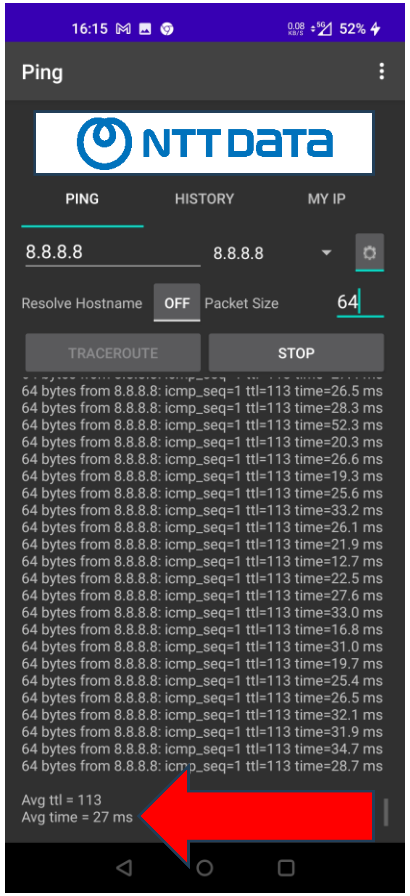


Conclusions

- Deep Q-Network (DQN) proves highly effective in reducing latency with **stable reward convergence**, leveraging new RL training strategies like UPF selection constraints and a **refined reward system for robust performance** across varied traffic conditions.
- The system's success is validated through real-time telemetry and hand-labeled data. Here are the results on user experience before and after the ML algorithm decision.
- ML estimated in few seconds that UPF 1 would have the best set of infrastructure values in order to optimize latency and packet loss.
- Human decision would have been not feasible in real world because of many parameters to be evaluated in changing traffic condition.
- Future efforts will focus on optimizing latency in Edge-Cloud Continuum interactions and adapting to diverse data center scenarios.



BEFORE



AFTER

NTT DATA

Demo

- Device connected to 5G SA environment associated to a defined network slice
- The network slice is bound to a specific UPF that, in this case, has bad network conditions
- Agents collects metrics and feed the ML algorithm which makes a decision about UPF change
- UPF change is done by leveraging on 5G network slice
- End user latency is improved

Demo link: https://youtu.be/EujiS2twBvI