MODELING THE ENVIRONMENT IN DEEP REINFORCEMENT LEARNING: THE CASE OF ENERGY HARVESTING BASE STATIONS

Nicola Piovesan, Marco Miozzo, Paolo Dini

CTTC/CERCA, Av. Carl Friedrich Gauss, 7, 08860, Castelldefels, Barcelona, Spain

ABSTRACT
In this paper, we focus on the design of energy self-sustainable mobile networks by enabling intelligent energy management that allows the base stations to mostly operate off-grid by using renewable energy. We propose a centralized control algorithm based on Deep Reinforcement Learning. The single agent is able to learn how to efficiently balance the energy inflow and spending among base stations observing the environment and interacting with it. In particular, we provide a study on the performance achieved by this approach when considering different representations of the environment. Numerical results demonstrate that using a good level of abstraction in the choice of the representation variables may enable a proper mapping of the environment into actions to take, so as to maximize the numerical reward.

Index Terms— reinforcement learning, deep learning, energy sustainability, mobile networks

1. INTRODUCTION
One of the principal goals of artificial intelligence is to produce autonomous agents that learn their optimal behaviors by interacting with the environment and improve them over time through trials and errors [1]. In recent years, Deep Learning (DL) has enabled the application of Reinforcement Learning (RL) to problems that were previously untractable due to complexity issues, i.e., the high dimensions of the state/action space. In fact, DL allows enhancing classic RL methods by taking advantage of the function approximation and representation learning characteristics of deep Artificial Neural Networks (ANNs).

Recently, Deep Reinforcement Learning (DRL) has been used to design agents able to play a range of video-games by only observing the video pixels [2] and a hybrid DRL system has defeated the world champion in Go [3]. DRL has also been applied to robotics, where robots learn directly from observation of the real world from a camera [4, 5].

A key issue in RL is to define a proper model of the environment sensed by the agents, so that they may accurately capture the system dynamics and learn how to optimally interact with it. In this paper, we investigate on different representations of the environmental state and we provide an analysis of the effects of the state representations on the RL training phase, its policies and on the resulting system performance. In particular, we apply DRL to a mobile cellular network scenario. We consider a two-tier architecture with hybrid power suppliers: Macro Base Stations (MBSs) reside in the first tier to provide baseline coverage and capacity and are powered by the electrical grid, whereas Small Base Stations (SBSs) operate in the second tier to provide capacity enhancement and are supplied by solar panels and batteries. Moreover, the SBSs are interconnected through a micro-grid and can share their exceeding energy with the MBS to further reduce its grid energy consumption. A centralized agent based on DRL is in charge of coordinating the switch ON-OFF of the SBSs to optimize the network performance in terms of energy consumption and traffic served.

The paper is organized as follows. In Section 2, we introduce the system model. In Section 3, we describe the adopted centralized controller based on DRL. In Section 4, we provide numerical results about the training performance, the policies learned by the agent and the achieved network performance. Finally, we draw our conclusions in Section 5.

2. SYSTEM MODEL
The considered Radio Access Network (RAN) consists of clusters of one MBS and \( N \) non-overlapping SBSs. In particular, we consider a Long-Term Evolution (LTE) RAN with a transmission bandwidth \( BW \) divided into \( R \) Resource Blocks (RBs) of 1 ms each [6]. Each SBS has \( N_{UE} \) associated User Equipments (UEs). If the SBS \( i \) is OFF at time \( t \) then its UEs are managed by the MBS. However, the MBS may have reached its capacity limit (i.e., it does not have any available RB to allocate to the UEs) and may drop part of the handed over UEs. This event is defined as system outage.

We define \( \alpha^{(t)} = [\alpha_1^{(t)}, \alpha_2^{(t)}, \ldots, \alpha_N^{(t)}] \) as the vector de-
scribing the operative states of the SBSs, where
\[ a_i(t) = \begin{cases} 0 & \text{if the } i\text{-th SBS is OFF} \\ 1 & \text{if the } i\text{-th SBS is ON} \end{cases} \] (1)

The energy harvesting and traffic demand processes are responsible of the variation in time of the system. We define \( H(t) = [H_1(t), H_2(t), \ldots, H_N(t)] \) as the vector of the energy harvested by the SBSs at time \( t \) whereas the levels of the batteries at time \( t \) is defined as \( B(t) = [B_1(t), B_2(t), \ldots, B_N(t)] \).

Moreover, we define \( L(t) = [L_1(t), L_2(t), \ldots, L_N(t)] \) with \( L_i(t) \in [0,1] \) as the vector of the traffic loads experienced by the SBSs in the cluster at time \( t \).

The battery level at time \( t+1 \) is computed as
\[
B(t+1) = \min\left( B(t) + H(t) - P(t) \Delta t, B_{\text{cap}} \right) \tag{2}
\]
where \( P(t) = [P_1(t), P_2(t), \ldots, P_N(t)] \) is the SBSs power consumption at time \( t \) and \( B_{\text{cap}} \) is the batteries capacity. Moreover, the SBSs automatically switch OFF when the battery level is below a threshold \( B_{\text{th}} \). The exceeding energy at time \( t \) is defined as \( X(t) = [X_1(t), X_2(t), \ldots, X_N(t)] \) and is computed as
\[
X(t+1) = \max\left( B(t) + H(t) - P(t) \Delta t - B_{\text{cap}}, 0 \right) \tag{3}
\]

The Base Stations (BSs) energy consumption is modeled by the linear function \( P = P_0 + \beta L \), where \( P_0 \) is the baseline energy consumption and \( \beta \) is a scaling factor. In details, we consider \( P_{0\text{MBS}} = 750 \text{W}, \beta_{\text{MBS}} = 600 \) for the MBS and \( P_{0\text{SBS}} = 105.6 \text{W}, \beta_{\text{SBS}} = 39 \) for the SBSs. This model is supported by real measurements and matches the power profiles of real BSs [7].

### 3. DEEP Q-LEARNING ALGORITHM

We consider a centralized control agent located at the MBS site and based on the Deep Q-Learning (DQL) algorithm. The goal of the agent is to learn how to map a state of the environment into an action to take, so as to maximize a numerical reward. The agent discovers which actions lead to the highest reward by trying them. This is a challenging task since the actions not only affect the immediate reward but also the next state of the environment and thus the following rewards. Specifically, DQL is an off-policy temporal difference control algorithm that is able to learn an approximation of the optimal policy independent of the policy being followed [8]. In this specific case, the agent is in charge of coordinating the switch ON/OFF of the SBSs in the cluster. In particular, the objective of the agent control is to reduce the grid energy consumption while maintaining the drop level below a threshold \( D_{\text{th}} \). Consequently, we define the reward function as
\[
r(x, a) = -E_m(x, a) + 1 [D(x, a) \leq D_{\text{th}}] \tag{4}
\]
where \( D(\cdot) \) is the system traffic drop, \([\cdot]\) is the step function, and \( E_m(\cdot) \) is the normalized grid energy consumption (equivalent to the difference between the MBS energy consumption and the excess energy shared by the SBSs). The normalized grid energy consumption at time \( t \) is computed as
\[
E_m(t) = \max \left( \frac{P_{\text{MBS}}(t) \cdot \Delta t - \sum_{i=1}^{N} X_i(t)}{P_{\text{MBS}}^\text{max} \Delta t}, 0 \right) \tag{5}
\]
where \( P_{\text{MBS}} \) and \( P_{\text{MBS}}^\text{max} \) are respectively the power and the peak power consumed by the MBS.

Therefore, the agent is getting a reward equal to 1 every time it is able to satisfy the constraint on the drop whereas an amount proportional to the consumed grid energy is discounted.

The centralized agent is in charge of maintaining a Q-function \( Q(x(t), a(t)) \) which represents the level of convenience in taking the action \( a(t) \) when the system is in state \( x(t) \) at time \( t \). In particular, the Q-function is estimated by using an ANN approximator. In details, the function \( Q(x(t), a(t)) \) is approximated by \( Q(x(t), a(t)|\theta) \), where \( \theta \) represents the ANN parameters.

The approximated Q-function is computed as
\[
\tilde{Q}(x(t), a(t)|\theta) = r(x(t), a(t)) + \gamma \max_{a'} Q(x^{(t+1)}, a'|\theta) \tag{6}
\]
where \( \gamma \) is the discount factor, that sets the importance of future rewards.

The state representations of the environment are the input of the ANN, whereas there is a separate output neuron for every action. In this way, the output of the ANN corresponds to the predicted Q-values of all the possible actions. This architecture allows computing all the Q-values with a single forward pass through the ANN.

The replay memory \( \mathcal{R} = \{e_1, \ldots, e_{s_e}\} \) is adopted to store the previous experiences, which are defined as:
\[
e_t = (x(t), a(t), r(t), x(t+1)) \tag{7}
\]
A batch of \( k \) experiences is randomly sampled from \( \mathcal{R} \) at every time step to perform the training of the ANN. In particular, the ANN is trained with the goal of minimizing the loss
\[
L = \frac{1}{T} \sum_{t=0}^{T} \left[ \tilde{Q}(x(t), a(t)|\theta) - Q(x(t), a(t)|\theta) \right]^2 \tag{8}
\]
After training the ANN, an action is taken according to a \( \epsilon \)-greedy policy. In details, a random action (exploration) is taken with probability \( \epsilon \) whereas the action which maximize
Algorithm 1 Deep Q-Learning algorithm
1: Initialize the replay memory $R$ to capacity $s_r$
2: Initialize the neural network with random weights
3: for episodes = 1, ..., $M$ do
4: Initialize state $s_1$
5: for $t=1,...,T$ do
6: Select action $a^{(t)} = \max_{a'} Q(x^{(t)}, a')$ with probability $1 - \epsilon$. This exploration policy allows to continuously discover actions that maximize the agent rewards. The detailed steps of the DQL algorithm are reported in Algorithm 1.
7: Execute action $a^{(t)}$ and observe the reward $r^{(t)}$ and the next state $x^{(t+1)}$.
8: Store the experience $(x^{(t)}, a^{(t)}, r^{(t)}, x^{(t+1)})$ in $R$
9: Sample a random batch of experiences $(x^{(t)}, a^{(t)}, r^{(t)}, x^{(t+1)})$ from $R$
10: Set $y^{(t)} = r^{(t)}$ for terminal state $x^{(t+1)}$ otherwise $y^{(t)} = r^{(t)} + \gamma \max_{a'} Q(x^{(t+1)}, a')$
11: Perform gradient descend step on $y^{(t)} - Q(x^{(t)}, a^{(t)})$
12: end for
13: end for

the Q-function (exploitation) is selected with probability $(1 - \epsilon)$. This exploration policy allows to continuously discover actions that maximize the agent rewards. The detailed steps of the DQL algorithm are reported in Algorithm 1.

In this work, we considered the following representations of the environment:

1. $B$: $x = [B]$
2. $HBL$: $x = [H, B, L]$
3. $Bh$: $x = [B, h]$
4. $HBLh$: $x = [H, B, L, h]$

where $h \in [0, 23]$ represents the hour of the day in which the measurements are collected. The rationale behind these choices is to model the environment with an incremental number of variables that may (or may not) lead to a better representation of the dynamic processes characterizing the system.

### 4. NUMERICAL RESULTS

In this section, we provide numerical results on the performance achieved when considering the environment representations presented in Section 3.

The scenario considered in this analysis consists of a cluster of 1 MBS placed in the center of a 1 km$^2$ area and 5 SBSs randomly placed. Moreover, 100 UEs are deployed in a radius of 50 m from each SBS to mimic a hotspot scenario. The SBSs are powered by a solar panel of 4.48 m$^2$ area and a lithium ion battery of 2 kWh capacity. These dimensions allow a fully recharge of the battery in a typical day of winter. Realistic energy harvesting traces are obtained considering the city of Los Angeles. User traffic is based on the classification proposed in [7] and aggregated downlink traffic has been generated based on the profiles defined in [9]. In particular, we consider the traffic profile of a residential area, which is characterized by a peak of the demand in the night, when solar energy is not available. Additional simulation parameters are described in Table 1.

| Parameter                        | Value          |
|----------------------------------|----------------|
| Solar panel size (m$^2$)         | 4.48 (16×16)   |
| Solar panel efficiency (%)       | 21             |
| Battery capacity (kWh)           | 2              |
| MBS transmission power (dBm)     | 43             |
| SBS transmission powe (dBm)      | 38             |
| Bandwidth (MHz)                  | 20             |
| Channel model                    | Okumura-Hata [10] |
| Heavy users traffic (MB/h)       | 900            |
| Ordinary users traffic (MB/h)    | 112.5          |
| Heavy users percentage (%)       | 50             |
| Traffic profile                  | Residential    |
| Battery threshold $B_{th}$ (%)   | 20             |
| High drop level $D_{th}$ (%)     | 10             |

#### 4.1. Training performance

In this section, we analyze the influence of the environment state representation on the algorithm training. In particular, the agent parameters have been set to the values that provide the best average reward per episode according to the results of the performed simulation campaign.

We consider the multilayer perceptron as basic architecture for the ANN, consisting of multiple fully connected layers of neurons [11]. Specifically, we adopt a ANN architecture with two hidden layers and 50 neurons per layer. The linear activation function is used for the output layer, whereas the ReLU activation function is used for the input and hidden layers. Finally, the learning rate is $\alpha = 10^{-4}$, the discount factor is $\gamma = 0.9$ and the size of the experience replay batch is $k = 20$. The exploration rate is set to $\epsilon = 0.9$ at the beginning of the training and discounted by 10% at every training episode, until reaching the minimum value of $\epsilon_{\text{min}} = 0.05$.

Figure 1 shows the average reward per episode for the different representations. The highest is achieved by the Bh and HBLh representations: both of them reach 0.546 on average after 160 episodes. The EBL representation reaches 0.542 on average after 120 episodes, whereas the B representation reaches 0.540 on average after 30 episodes. Therefore, we can note that: i) increasing the number of state variable does not necessarily improve the average reward per episode, and ii) including the hour of the day in the environmental model increases the average reward, at the expenses of a longer training phase.
4.2. Switch ON-OFF policy

In this section, we analyze the policy selected by the agent when considering the different representations. We adopt the switch ON rate as a metric to describe the behaviors of the SBSs. Moreover, we focus on the month of December since it represents the most challenging month for learning a policy due to the scarce availability of the harvested energy.

The daily average switch ON rate of the SBSs is reported in Figure 2. The shape of the residential traffic profile is also depicted. A common behavior can be observed regardless the adopted environmental model. The SBSs switch OFF during night hours, when the traffic is low, to save the stored energy. Then, the SBSs are gradually switched ON, starting from 7 AM to provide service during high traffic demand hours. However, the B and HBL policies show a higher average switch ON rate between 1 AM and 7 AM than the Bh and HBLh policies. Moreover, the B policy is maintaining the switch ON rate to an average of 0.9 between midday and 6 PM. The HBL, Bh and HBLh policies are maintaining the switch ON rate to an average of 0.80, 0.79 and 0.78 in the same period, respectively. In particular, a local minimum is experienced at 6 PM. Finally, the policies show different average switch ON rates also during the peak of the traffic (i.e. between 7 PM and midnight). In details, the B and HBL policies maintain an average switch ON rate of 0.65 and 0.67, respectively. On the other hand, the Bh and HBLh reach higher values, respectively 0.74 and 0.76.

4.3. Energy consumption and network performance

The effects of the learned policies on the network performance are discussed in this section. In particular, the grid energy consumption and the traffic drop experienced in the month of December are reported in Table 2, considering the different representations of the environment.

In this section, we analyze the policy selected by the agent when considering the different representations. We adopt the switch ON rate as a metric to describe the behaviors of the SBSs. Moreover, we focus on the month of December since it represents the most challenging month for learning a policy due to the scarce availability of the harvested energy.

The daily average switch ON rate of the SBSs is reported in Figure 2. The shape of the residential traffic profile is also depicted. A common behavior can be observed regardless the adopted environmental model. The SBSs switch OFF during night hours, when the traffic is low, to save the stored energy. Then, the SBSs are gradually switched ON, starting from 7 AM to provide service during high traffic demand hours. However, the B and HBL policies show a higher average switch ON rate between 1 AM and 7 AM than the Bh and HBLh policies. Moreover, the B policy is maintaining the switch ON rate to an average of 0.9 between midday and 6 PM. The HBL, Bh and HBLh policies are maintaining the switch ON rate to an average of 0.80, 0.79 and 0.78 in the same period, respectively. In particular, a local minimum is experienced at 6 PM. Finally, the policies show different average switch ON rates also during the peak of the traffic (i.e. between 7 PM and midnight). In details, the B and HBL policies maintain an average switch ON rate of 0.65 and 0.67, respectively. On the other hand, the Bh and HBLh reach higher values, respectively 0.74 and 0.76.

4.3. Energy consumption and network performance

The effects of the learned policies on the network performance are discussed in this section. In particular, the grid energy consumption and the traffic drop experienced in the month of December are reported in Table 2, considering the different representations of the environment.

The worst performance in terms of traffic drop is achieved by the B representation. The HBL representations leads to 6% less drop than the B representation, at the price of a small increase in the grid energy consumption. The best performance is achieved by the HBLh representation, which leads to the smallest values of grid energy and traffic drop. The Bh representation returns very similar values of grid energy and traffic drop. This suggests that a simple model including the battery level and the hour of the day is sufficiently good to capture the dynamics of the environment and return high performance.

5. CONCLUSIONS

In this paper, we analyzed the performance achieved by a centralized controller based on DQL, when considering different representations of the environment. Numerical results show that the adopted representations affect both the number of training episodes needed to converge and the asymptotic values of convergence. Moreover, the state representations may produce different operative policies, since they change the way the agent senses the environment. Finally, the performance of the system under study show that including the hour of the day in the state representation is fundamental to efficiently reduce both the energy consumption and the traffic drop.
6. REFERENCES

[1] Kai Arulkumaran, Marc Peter Deisenroth, Miles Brundage, and Anil Anthony Bharath, “Deep reinforcement learning: A brief survey,” *IEEE Signal Processing Magazine*, vol. 34, no. 6, pp. 26–38, 2017.

[2] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin Riedmiller, “Playing atari with deep reinforcement learning,” *arXiv preprint arXiv:1312.5602*, 2013.

[3] David Silver, Aja Huang, Chris J Maddison, Arthur Guez, Laurent Sifre, George Van Den Driessche, Julian Schrittwieser, Ioannis Antonoglou, Veda Panneershelvam, Marc Lanctot, et al., “Mastering the game of go with deep neural networks and tree search,” *nature*, vol. 529, no. 7587, pp. 484, 2016.

[4] Sergey Levine, Chelsea Finn, Trevor Darrell, and Pieter Abbeel, “End-to-end training of deep visuomotor policies,” *The Journal of Machine Learning Research*, vol. 17, no. 1, pp. 1334–1373, 2016.

[5] Sergey Levine, Peter Pastor, Alex Krizhevsky, Julian Ibarz, and Deirdre Quillen, “Learning hand-eye coordination for robotic grasping with deep learning and large-scale data collection,” *The International Journal of Robotics Research*, vol. 37, no. 4-5, pp. 421–436, 2018.

[6] Stefania Sesia, Matthew Baker, and Issam Toufik, *LTE-the UMTS long term evolution: from theory to practice*, John Wiley & Sons, 2011.

[7] G. Auer, O. Blume, V. Giannini, I. Godor, MA. Imran, Y. Jading, E. Katranaras, M. Olsson, D. Sabella, P. Skillermark, et al., “EARTH Deliverable D2.3: Energy efficiency analysis of the reference systems, areas of improvements and target breakdown,” January 2013.

[8] Richard S Sutton and Andrew G Barto, *Reinforcement learning: An introduction*, MIT press, 2018.

[9] Fengli Xu, Yong Li, Huadong Wang, Pengyu Zhang, and Depeng Jin, “Understanding Mobile Traffic Patterns of Large Scale Cellular Towers in Urban Environment,” *IEEE/ACM Trans. Netw.*, vol. 25, no. 2, pp. 1147–1161, April 2017.

[10] M. Hata, “Empirical formula for propagation loss in land mobile radio services,” *IEEE Transactions on Vehicular Technology*, vol. 29, no. 3, pp. 317–325, August 1980.

[11] Simon Haykin, *Neural networks: a comprehensive foundation*, Prentice Hall PTR, 1994.

9000