Joint Load Control and Energy Sharing for Renewable Powered Small Base Stations: a Machine Learning Approach

Nicola Piovesan, David López-Pérez, Marco Miozzo, Paolo Dini

***Abstract*—The deployment of dense networks of small base stations represents one of the most promising solutions for future mobile networks to meet the foreseen increasing traffic demands. However, such an infrastructure consumes a considerable amount of energy, which, in turn, may represent an issue for the environment and the operational expenses of the mobile operators. The use of renewable energy to supply the small base stations has been recently considered as a mean to reduce the energy footprint of the mobile networks. In this paper, we consider a hierarchical structure in which part of the base stations are powered exclusively by solar panels and batteries. Base stations are grouped in clusters and connected in a micro-grid. A central controller enables base station sleep mode and energy sharing among the base stations based on the available energy budget and the traffic demands. We propose three different implementations of the controller through Machine Learning models, namely Imitation Learning, Q-Learning and Deep Q-Learning, capable of learning optimal sleep mode and energy sharing policies. We provide an exhaustive discussion on the achieved performance, complexity and feasibility of the proposed models together with the energy and cost savings attained.**

***Index Terms*—Energy Sustainability, Mobile Networks, Machine Learning, Deep Learning, Deep Reinforcement Learning, Energy Efficiency**

I. Introduction

During the last decades, mobile network operators have witnessed an exponential increase in the traffic demand, mainly due to the high request of services from a huge amount of users. The International Telecommunication Union has estimated that there exist almost as many mobile subscribers as people in the world (almost 6.8 billions) [1]. The trend is of a further increase in both the traffic demand and the number of connected devices over the next years. The traffic load is expected to have an annual growth rate of 53% for mobile network alone [2], and the upcoming industrial era (known also as Industry 4.0 [3]), which will connect different types of devices to the mobile infrastructure including human and machine type communications, will definitely exacerbate such an increasing trend. The described scenario may indeed open new opportunities for individuals and organisations to take advantage of the next generations of mobile technology. However, the massive use of Information and Communication Technology (ICT) is increasing the level of energy consumed by the telecommunication infrastructure and its carbon footprint on the environment. The Digital Power Group estimated that 10% of the worldwide electricity generation is due to the ICT industry [4], and has forecasted that it might reach 51% by 2030, which would imply that the 23% of the carbon footprint by human activity will be due to ICT [5]. Environmental sustainability is thus a key requirement for designing next generation mobile networks.

While reviewing the scientific literature, it appears that the most promising approach to reduce the energy consumption of mobile networks is to enable sleep modes in some network elements during periods of low traffic [6]. However, the adoption of these proposals is still scarce due to operators' concerns in coverage holes and failures of the network equipment due to the frequent ON/OFF switches [7]. Recently, the use of sustainable energy for supplying network elements has attracted the attention of the research community, where the interest is driven by the increased efficiency and the reduced costs of energy harvesters and storage devices, specially when installed to supply Small Base Stations (SBSs) [8]. Such a solution has been demonstrated to be environmentally and economically sustainable in both rural and urban areas [9].

We envision a Radio Access Network (RAN) setup in which a hierarchical cell structure is deployed within the same geographical area with Base Stations (BSs) of different scale factors, transmission power, computational capabilities and coverage areas [10]. This federation of BSs together with the distributed harvesters and storage devices at the SBS sites form a micro-grid, whose operations are managed by an energy management system in charge of controlling the intermittent and erratic energy budget from the Renewable Energy Sources (RESs). We consider load control (i.e. enabling sleep mode in the SBSs) as a method to properly manage energy inflow and spending, based on the traffic demand. Moreover, we propose the possibility of improving the network energy efficiency by sharing the exceeding energy that may be available at some BS sites within the micro-grid. This combination of sleep modes and energy sharing represents an advance with respect to state-of-the-art approaches, as will be also discussed in Section II.

In this paper, we investigate on the control of the sleep modes and energy sharing among the BSs by a centralised agent, which implements Machine Learning (ML) techniques and is capable of learning how to efficiently operate the network. We discuss on the achieved performance, complexity and feasibility of different ML approaches. In particular,

N. Piovesan and D. López-Pérez are with Huawei Technologies, Boulogne Billancourt, Paris, France (emails: nicola.piovesan@huawei.com, david.lopez.perez@huawei.com)

M. Miozzo and P. Dini are with the Centre Tecnològic de Telecomunicacions de Catalunya (CTTC/CERCA), Castelldefels, Barcelona, Spain (emails: npiovesan@cttc.es, mmiozzo@cttc.es, pdini@cttc.es)

we study Supervised and Reinforcement Learning models. The proposed supervised approach, called Imitation Learning, learns the optimal policies from labelled data collected by a supervisor. The creation of the training set is a hard task since it relies on an expert supervision. For this reason, Imitation Learning is generally feasible in scenarios with a limited number of SBSs due to complexity issues. Instead, solutions based on Reinforcement Learning (RL) learn from the interaction with the environment and no specific supervision is required. However, a Markov Decision Process has to be properly defined together with a reward function to fit the optimisation problem that must be solved [11]. In this work, we tailor two RL methods based on tabular and deep models, respectively.

The tabular RL method, i.e. Q-learning, is thought for discrete state/action spaces, and requires quantisation to deal with the continuous states of our scenario. Consequently, the number of quantisation levels have to be properly settled to find a reasonable trade-off between the approximation introduced and the performance in terms of cumulative reward, convergence time and memory footprint.

On the other hand, the Deep RL method, i.e. Deep Q-learning, uses Artificial Neural Networks (ANNs) as function approximation to address the quantisation issue of the tabular method, and may handle a continuous state space. Moreover, the ANN tailored for our scenario requires smaller memory and achieves faster convergence than the tabular method. This translates into higher performance in terms of energy saving and system outage. Moreover, deep RL is able to control dense scenarios, i.e. a high (> 4) number of deployed SBSs, where the other models fail for complexity and memory issues.

In summary, the contributions of the paper are listed below:

- *Renewable energy powered RAN with energy sharing*: We propose a RAN architecture with SBSs powered by distributed energy harvesters and storage devices. Energy sharing between BSs is used to improve the energy efficiency of the network.
- *Joint load control and energy sharing problem statement*: We provide a theoretical formulation of the joint load control and energy sharing optimisation problem. The objective of the problem is to minimise the grid energy consumption and the system outage.
- *Centralised load control plus energy sharing using different ML approaches*: We develop three different implementations of a centralised energy-aware RAN controller based on machine learning: namely Imitation Learning, Q-Learning and Deep Q-Learning. The training phase of those three different implementations is analysed against different setups of the learning parameters.
- *Load control plus energy sharing policies and network performance evaluation*: We analyse the load control plus energy sharing policies learned by the three different implementations of the agent and the achieved performance are compared in terms of grid energy consumption and dropped traffic.
- *Energy savings and cost analysis*: We provide an analysis of savings that a mobile operator may achieve by deploying a RAN powered with renewable energy and implementing our centralised learning controller in terms of monetary cost and energy footprint.

The rest of the paper is organised as follows. In Section II, we discuss the related literature. In Section III, we introduce the reference architecture considered in this paper, whereas in Section IV, we provide the system model, which includes the network, the solar panel, the generated traffic and the power consumption of the BSs. In Section V, we formulate the joint load control and energy sharing optimisation problem. In Section VI, we discuss the proposed learning approaches, i.e. Imitation Learning, Q-Learning and Deep Q-Learning. In Section VII, we introduce the simulation scenario, discuss the setup of the parameters for the learning approaches, analyse the obtained policies and the achieved performance in terms of energy consumption and dropped traffic. Finally, in Section VIII, we draw our conclusions.

II. Related Work

In the last years, cooperative communication methods have been deeply investigated for energy saving purposes, with a special focus on sleep modes and traffic offloading [6]. In [12], the problem of minimising the grid energy consumption in a scenario with hybrid powered BSs has been analysed, where the authors applied a two-stage Dynamic Programming (DP) method with the goal of saving grid energy while satisfying the users required quality of service. In [13], an algorithm for optimal switch ON/OFF scheduling based on the resolution of a ski-rental problem is proposed, while in [14], the coordination of the switch ON/OFF between BSs powered by energy harvesting and grid energy is studied, considering a DP formulation of the problem. In this line, in [15], we formulated the optimal switch ON/OFF DP problem as a graph theory shortest-path problem, and solved it by using a variant of the label correcting algorithm. However, the use of DP implies offline optimisation with a priori knowledge of system dynamics and normally leads to high computational complexity.

Approaches based on ML have been recently proposed as a way to schedule the switch ON/OFF of the BSs without any prior knowledge of the system and to manage the algorithm complexity. In [16], a method based on distributed RL has been studied, where each BS autonomously decide whether to switch ON/OFF according to the harvested energy, the traffic demand and the energy storage. The adoption of a distributed architecture allows to reduce the complexity of the algorithm. However, this approach suffers from a lack of coordination. In [17], this problem has been addressed by considering a layered learning algorithm based on the decomposition of the problem into two layers. The first layer is based on RL and in charge of local control at each SBS, whereas the second layer is based on ANNs and manages the network wide coordination among the SBSs.

Energy sharing methods among BSs have also attracted the interest of the research community as a way to improve the energy efficiency of the network. In [18], a solution for the deployment of a micro-grid is discussed, where a grid entity is introduced to manage the energy flow between the grid

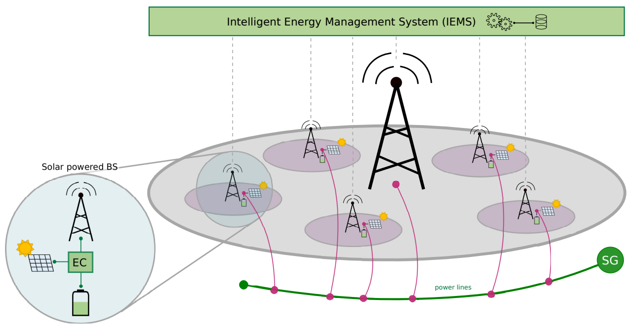


Fig. 1. Diagram illustrating the reference framework, including the radio access network with multiple tiers, the intelligent energy management system and the micro-grid connections. In left-bottom side, a simplified scheme of the solar-powered BS is shown.

operator and a group of BSs. Moreover, in [13], the authors propose an algorithm that jointly optimises the transmitted power of BSs and the transferred energy to maximise the sum-rate throughput for all the users.

In our previous work [19], we merged the concepts of communication cooperation and energy sharing by formulating a joint load control plus energy sharing problem in a two-tier mobile network with energy harvesting capabilities. We applied DP to define the optimal policies and determine the performance bounds that an intelligent control may achieve in the case of knowing the system dynamics a priori. The results show that a combination of switch ON/OFF and energy sharing allows to reduce the energy consumption of the RAN up to almost 50% lower than a grid-connected network deployment, with relevant cost savings for the operator. Following those encouraging results, in this paper, we extend our previous work, and consider three online solutions based on different ML methods for the control of a cluster of BSs powered with renewable energy and with energy sharing capabilities. To the best of our knowledge, this is the first paper focusing on centralised learning methods for the optimisation of BS switch ON/OFF plus energy sharing policies. Moreover, the study reported here may serve as a compendium of supervised and reinforcement learning approaches for network optimisation and their comparison in terms of system performance, implementation complexity, memory footprint and feasibility.

III. Reference Scenario

We consider a multi-tier RAN architecture where Macro Base Stations (MBSs) provide the baseline coverage and capacity, whereas SBSs are deployed to enhance the capacity in the hotspots (e.g. city centres, shopping malls, etc.). The densification of the RAN increases the capacity offered to end users thanks to a higher spectrum reuse and a better spectral efficiency. In the proposed framework, the mobile infrastructure is divided into clusters. Each cluster is composed of a MBS and a set of SBSs, which do no overlap in coverage. The MBS is powered by the electricity grid, assuring a reliable connection. The SBSs are, instead, only powered by the renewable energy they harvest through a photovoltaic panel and store into a battery. The solar power is deemed the most appropriate due to the good efficiency of commercial photovoltaic panels as well as the wide availability of the solar source for typical installations [20].

The BSs in the cluster are connected into a micro-grid and the renewable energy is managed based on a *harvest-store-share* approach. In more details, the power harvested by the solar panel is consumed by the SBS and any excess is stored into the battery. Whenever the battery has reached its full capacity, the excess energy is shared with the MBS that will use it to reduce its demand from the electricity grid. The micro-grid is implemented by deploying power lines. Low resistive losses (i.e. the energy lost in the conductor due to Joule heating) are guaranteed by the short distances between MBS and SBSs. When the SBSs have exceeding energy that is not needed by the MBS, this energy is locally dissipated at the SBSs site and not injected into the micro-grid. In this way, the micro-grid energy balance is maintained.

In the proposed framework, we envision a central control unit named Intelligent Energy Management System (IEMS), located at the MBS site. The IEMS has the knowledge of the overall network condition, enabling a better coordination of the cluster. In particular, the SBSs operational states depend on the dynamics of the energy harvesting and the traffic demand. Therefore, the IEMS is in charge of opportunistically operating the network to achieve efficient utilisation of the harvested energy and prevent SBSs blackout during periods with low renewable energy arrivals and high traffic demand. The Energy Controller (EC) is an entity located at the SBS sites, which is in charge of communicating to the IEMS the necessary local information (e.g. the battery level) and to implement the decisions taken by the central controller. The reference control architecture is shown in Figure 1.

IV. System Model

In this section, we introduce the mathematical models of the network, the solar panel, the traffic demand and the power consumption of the BSs.

A. Network Model

We consider a two-tier RAN composed of clusters of one MBS and N non-overlapping SBSs. Clusters do not interfere among each other. When a SBS is switched OFF, the associated User Equipments (UEs) are handed over the MBS to continue their transmissions. However, the MBS may reach the limit of its capacity and not be able to provide service to all the UEs.

The system evolves in time based on the variation of the traffic load and the energy harvested and stored into the batteries. The traffic load experienced by the SBSs in the cluster at time t is defined as $\mathbf{L}^{(t)} = \left[L_1^{(t)}, L_2^{(t)}, \ldots, L_N^{(t)}\right]$ with $L_i^{(t)} \in [0,1]\,\forall i$. The energy harvested at time t is defined as $\mathbf{H}^{(t)} = \left[H_1^{(t)}, H_2^{(t)}, \ldots, H_N^{(t)}\right]$. The energy stored into the batteries at time t is defined as $\mathbf{B}^{(t)} = \left[B_1^{(t)}, B_2^{(t)}, \ldots, B_N^{(t)}\right]$.

Using the above definitions, the energy stored into the batteries at the beginning of the next time step can be calculated according to:

$$\mathbf{B}^{(t+1)} = \min\left(\mathbf{B}^{(t)} + \mathbf{H}^{(t)} - \mathbf{P}^{(t)}\Delta_t, B_{\text{cap}}\right) \tag{1}$$

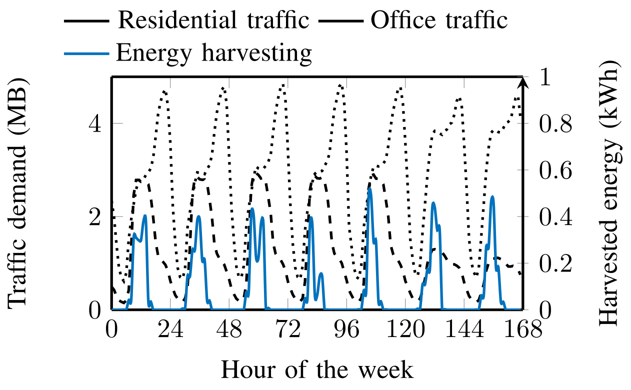


Fig. 2. Example of temporal variation of the traffic (black lines) and energy harvesting process (blue line) in a week of December

where $\mathbf{P}^{(t)} = \left[P_1^{(t)}, P_2^{(t)}, \ldots, P_N^{(t)}\right]$ is the power consumed by the SBSs at time t, B_{cap} is the capacity of the batteries, and Δ_t is the duration of the time step. It is important to note that SBSs automatically switch OFF when the battery level is below a threshold B_{th} .

Similarly, the amount of energy that exceeds the battery capacity, and can thus be shared within the network, is defined as $\mathbf{X}^{(t)} = \left[X_1^{(t)}, X_2^{(t)}, \ldots, X_N^{(t)}\right]$, and can be computed according to:

$$\mathbf{X}^{(t+1)} = \max\left(\mathbf{B}^{(t)} + \mathbf{H}^{(t)} - \mathbf{P}^{(t)}\Delta_t - B_{\text{cap}}, 0\right) \tag{2}$$

B. Solar panel model

Real measurements of solar radiation covering 20 years (1991-2010) are provided in the US National Solar Radiation Data Base [21]. The database contains hourly information about solar radiation and other meteorological data. The solar radiation has been converted into harvested solar energy by considering the model introduced in [22].

The solar energy traces have usually a bell shape with a peak at the central hours of the day, whereas the energy harvested during night hours is negligible. Moreover, their magnitudes depend on the location (e.g. longitude and latitude) and the period of the year (i.e. the seasons). An example of solar energy traces for a week of December in Los Angeles is shown in Figure 2.

C. Traffic Model

Following the model introduced in [23], we consider N_{UE} subscribers randomly positioned in the area of each SBS, that can be classified in two types, ordinary and heavy, according to their demand of traffic. In more details, ordinary users demand for $d_o = 112.5$ MB/h, whereas heavy users demand for $d_h = 900$ MB/h. According to [23], the number of active users in the peak hours corresponds to 16% of the subscribers, and the variation of both the number of active users and the traffic per active user is modelled by the function $\vartheta(t)$. Therefore, the average amount of traffic requested to each SBSs at time t is defined as:

$$T_{\text{SBS}}(t) = N_{\text{UE}} \cdot \vartheta(t) \cdot \left[(1 - r_h) \cdot d_o + r_h \cdot d_h\right], \tag{3}$$

where r_h is the ratio of heavy users. The traffic profile function $\vartheta(t)$ has been designed by considering real traffic profiles described in [24], which are derived combining time, location and frequency information of thousands of cellular towers. Traffic variability is added to the profiles following a normal distribution with zero mean and standard deviation derived from the measurements of real mobile traffic traces [25].

The analysis provided in [24] demonstrates that the urban mobile traffic usage can be described by five basic time domain patterns that corresponds to functional regions, i.e. residential, office, transportation, entertainment and comprehensive. In this paper, we focus on residential and office profiles, being them the most traffic intensive in metropolitan areas. An example of their traffic profiles is shown in Figure 2. We can notice that both profiles present a high activity during the day and lower during night. However, the highest amount of traffic is concentrated in the daylight hours (i.e. from 10 am to 6 pm) and in the evening hours (i.e. from 6 pm to 12 pm) for the office profile and the residential profile, respectively.

The resource allocation scheme uses the methodology defined in [26]. The modulation and coding scheme is assigned to each UE as a function of its Signal to Interference plus Noise Ratio (SINR), which is given by:

$$\text{SINR} = \frac{|g_0|^2 P_{t,0}}{\sum_{i=1}^{N_I} |g_i|^2 P_{t,i} + \sigma_0^2}, \tag{4}$$

where $P_{t,0}$ and g_0 are the transmission power and the channel gain for the useful transmission, respectively, N_I is the number of interferes, and $|g_i|^2$ and $P_{t,i}$ represent the channel gain and the transmission power of the i-th interferer. Finally, σ_0^2 is the power of the thermal noise.

D. Power Consumption Model

The power consumption of a BS at time t is approximated by the linear function:

$$P^{(t)} = P_0 + \beta L^{(t)}, \tag{5}$$

where P_0 is a static component representing the baseline power consumption of the BS at zero load, while β is the scale factor. These two parameters depend on the particular BS type. In this paper, we consider $P_0^{\text{MBS}} = 750$W, $\beta^{\text{MBS}} = 600$W for the MBS and $P_0^{\text{SBS}} = 105.6$W, $\beta^{\text{SBS}} = 39$W for the SBS [23].

V. Optimisation Problem

The evolution of the system under study, and described in the above sections, may be defined by a Markov Decision Process as: $\boldsymbol{x}^{(t+1)} = f\left(\boldsymbol{x}^{(t)}, \boldsymbol{a}^{(t)}, \boldsymbol{w}^{(t)}\right)$, where $\boldsymbol{x}^{(t)}$ is the state of the system, $\boldsymbol{a}^{(t)}$ is the control and $\boldsymbol{w}^{(t)}$ is the random disturbance at time t (i.e. the randomness of the energy arrival and traffic processes).

The state of the system at time t is defined as $\boldsymbol{x}^{(t)} = \left[\boldsymbol{B}^{(t)}, h^{(t)}\right]$, where $h^{(t)} \in [0, 23]$ is the hour of the day at time t, while the system control is defined as $\boldsymbol{a}^{(t)} = \left[a_1^{(t)}, a_2^{(t)}, \ldots, a_N^{(t)}\right]$, where

$$a_i^{(t)} = \begin{cases} 0 & \text{if the } i\text{-th SBS is switched OFF} \\ 1 & \text{if the } i\text{-th SBS is switched ON.} \end{cases} \tag{6}$$

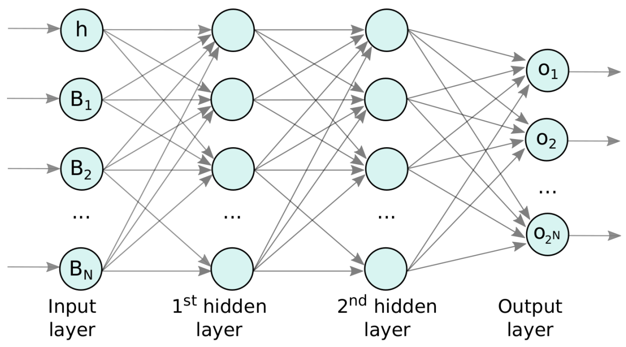


Fig. 3. Example of a multiclass classifier based on ANNs.

The optimisation goal is to minimise the grid energy consumption, while keeping the percentage of dropped traffic experienced in the cluster below a threshold, D_{th}. This optimisation problem can be formulated as:

$$\begin{aligned} \text{P1:} \min_{\{\boldsymbol{a}^{(t)}\}_{t=1,\dots,K}} & \sum_{t=1}^{K} \mathrm{E_m}(\boldsymbol{x}^{(t)}, \boldsymbol{a}^{(t)}, \boldsymbol{w}^{(t)}) \\ \text{subject to } & D^{(t)} \leq D_{\text{th}} \end{aligned} \tag{7}$$

where K is the time horizon or the number of times the control is applied, $D^{(t)}$ is the normalized traffic drop at time t and $\mathrm{E_m}$ is the normalised grid energy consumption drained by the MBS, given the operative modes of the SBSs at time t, whether ON or OFF. The normalised grid energy consumption drained by the MBS at time t is computed as:

$$\begin{aligned} \mathrm{E_m}(\boldsymbol{x}^{(t)}, \boldsymbol{a}^{(t)}, \boldsymbol{w}^{(t)}) = & \\ \max \Bigg(& \frac{P_{\text{MBS}}(\boldsymbol{x}^{(t)}, \boldsymbol{a}^{(t)}, \boldsymbol{w}^{(t)}) \cdot \Delta_t}{P_{\text{MBS}}^{\max} \Delta_t} \\ & - \frac{\sum_{i=1}^{N} X_i(\boldsymbol{x}^{(t)}, \boldsymbol{a}^{(t)}, \boldsymbol{w}^{(t)})}{P_{\text{MBS}}^{\max} \Delta_t}, 0 \Bigg). \end{aligned} \tag{8}$$

We proposed an offline solution of (7) in [15], where the sequential decision making problem has been represented through a graph, and the energy minimisation has been formulated as a shortest-path problem. A variant of the Label Correcting Algorithm [27] has being used to compute the minimum path, which defines the optimal switch ON/OFF sequence of the SBSs, given a priori knowledge of both the energy arrivals and the traffic demand. In this paper, we adopt such offline solution to compute performance bounds, and compare them with the policies learned through the proposed ML approaches.

VI. Machine Learning Models

In this section, we discuss the different implementations of the centralised agent at the IEMS, namely Imitation Learning and the two variants of RL, Q-Learning and Deep Q-Learning.

A. Imitation Learning

Imitation Learning (IL) is a supervised approach that consists on learning from a set of labelled data provided by an external supervisor [28]. Each example in the training set is a description of the system state with a specification of the correct action the agent shall take in that situation. In this case, the objective of the agent is to extrapolate (i.e. generalise) its response, so that it acts correctly in situations not included in the training set. Specifically, the agent is trained on the policies adopted by the offline algorithm in [15]. More in details, a multi-class classifier based on ANNs is used to learn a mapping from the state of the system $\boldsymbol{x}^{(t)}$ to the action $\boldsymbol{a}^{(t)}$. An example of the general ANN architecture for a multiclass classifier is shown in Figure 3. We consider the multilayer perceptron as basic architecture for the ANN, consisting on multiple fully connected layers of neurons [29]. The output of a neuron is computed using the following equation:

$$output = f_a\left(\sum_i (z_i \cdot w_i + b_i)\right), \tag{9}$$

where z is the input of the neuron, w is the weight of the connection to the neuron, b is the bias and $f_a(\cdot)$ is the activation function. The Rectified Linear Unit (ReLU) activation function (i.e. $f_a(z) = \max(0, z)$) is used for the hidden layers, whereas the SoftMax activation function is used for the output layer.

The model is optimised using the categorical cross entropy loss function, defined as:

$$\mathcal{L}(y - \hat{y}_i) = -\sum_i y_i \log \hat{y}_i \tag{10}$$

where y is a grounded truth vector (i.e. the actions taken by the offline algorithm), and $\hat{y}$ is a vector of predictions. The detailed steps of the training algorithm are specified in Algorithm 1.

Algorithm 1 Training of the ANN classifier

```
Input: training dataset $\mathcal{D}$ containing $n_d$ state/action pairs $<\boldsymbol{x}, \boldsymbol{a}>$
Input: number of training epochs $n_e$
Output: ANN classifier $c(\cdot)$
1: Randomly initialise the weights of the ANN $c$
2: for epochs $= 1, ..., n_e$ do
3:     for $n = 1, ..., n_d$ do
4:         Set $\boldsymbol{x}(n)$ as the input of the ANN classifier $c$ at iteration $n$ and $\boldsymbol{a}(n)$ as its associated target output.
5:         Train the ANN classifier $c$ on the input $\boldsymbol{x}(n)$ by running the backpropagation algorithm with the goal of minimising the categorical cross entropy defined in (10).
6:     end for
7: end for
8: return $c$
```

We consider a dataset containing observations collected in one year. A portion of 80% of the dataset is used for training the classifier, whereas the remaining 20% is used to validate the learned model.

The trained ANN is then implemented in the IEMS and manages the operative modes of the SBSs by the procedure described in Algorithm 2.

Algorithm 2 Imitation Learning agent

```
Input: classifier $c$
1: for $t = 1, \dots, T$ do
2:     $\boldsymbol{x}^{(t)} \leftarrow$ current state of the system
3:     $\boldsymbol{a} \leftarrow c\left(\boldsymbol{x}^{(t)}\right)$
4:     take action $\boldsymbol{a}^{(t)}$
5: end for
```

B. Reinforcement Learning

The objective of RL is to learn how to map the experienced situation (i.e. the state of the system) into the best action to take at every decision cycle t. By trying different actions, the agent learns the optimal behaviour of the system that maximizes a cumulative reward. This phase is called exploration and is aimed at training the algorithm for the stable phase, called exploitation, in which the agent will use the learned policies. A good trade off between exploration and exploitation has to be maintained to assure that the system is continuously exploring new state/action pairs and updating its policies with actions that return higher rewards.

The centralised controller implements an agent in charge of maintaining a policy and a Q-function $Q(\boldsymbol{x}^{(t)}, \boldsymbol{a}^{(t)})$, representing the level of convenience in taking the action $\boldsymbol{a}^{(t)}$ at time step t, given that the system is in state $\boldsymbol{x}^{(t)}$. As a result of the execution of an action at time t, the environment returns a reward $r^{(t)}$, which is used to update the Q-values, $Q(\boldsymbol{x}^{(t)}, \boldsymbol{a}^{(t)})$. In our proposals, the reward given to the agent after taking action $\boldsymbol{a}^{(t)}$, being in state $\boldsymbol{x}^{(t)}$ at time t, is defined as:

$$r^{(t)} = -\mathrm{E}_{\mathrm{m}}(\boldsymbol{x}^{(t)}, \boldsymbol{a}^{(t)}) + 1(D^{(t)} \leq D_{\mathrm{th}}), \tag{11}$$

where $1(\cdot)$ is the step function. The rationale behind this equation is to mimic the optimisation problem in (7): the agent is getting a reward equal to 1 each time it is able to maintain the drop level below the threshold D_{th}, and a discount proportional to the consumed grid energy is applied. Moreover, in our implementations, an ϵ-greedy exploration policy is adopted: the learned action is taken with probability $1-\epsilon$ (i.e. exploitation), whereas a random action is taken with probability ϵ (i.e. exploration).

In this paper, we consider the Q-Learning (QL) and Deep Q-Learning (DQL) algorithms, as instances of RL. QL 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 [11]. DQL is a variant of QL, which adopts an ANN as a function approximation to estimate the Q function [30]. Next, we detail the specific implementation of the two algorithms.

1) Q-Learning: In the standard QL algorithm, the Q-function is stored in a tabular form. The table describing the Q-function is therefore named Q-table. The reward $r^{(t)}$ is used to update the Q-value $Q(\boldsymbol{x}^{(t)}, \boldsymbol{a}^{(t)})$ according to the following rule:

$$\begin{aligned} Q(\boldsymbol{x}^{(t)}, \boldsymbol{a}^{(t)}) \leftarrow & Q(\boldsymbol{x}^{(t)}, \boldsymbol{a}^{(t)}) + \\ & \alpha[r^{(t)} + \gamma \max_{\boldsymbol{a}'} Q(\boldsymbol{x}^{(t+1)}, \boldsymbol{a}') - Q(\boldsymbol{x}^{(t)}, \boldsymbol{a}^{(t)})], \end{aligned} \tag{12}$$

Algorithm 3 Q-Learning algorithm

```
1: Initialize $Q(\boldsymbol{x}, \boldsymbol{a}) \, \forall \boldsymbol{x} \in \boldsymbol{X}, \boldsymbol{a} \in \boldsymbol{A}$ arbitrarily
2: for episodes = $1, ..., M$ do
3:     Initialize $\boldsymbol{x}^{(1)}$
4:     for $t = 1, \dots, T$ do
5:         Select action $\boldsymbol{a}^{(t)} = \max_{\boldsymbol{a}'} Q(\boldsymbol{x}^{(t)}, \boldsymbol{a}')$ with probability $1-\epsilon$ otherwise take a random action with probability $\epsilon$.
6:         Execute action $\boldsymbol{a}^{(t)}$ and observe the reward $r^{(t)}$ and the next state $\boldsymbol{x}^{(t+1)}$
7:         Update the Q-value by using (12)
8:     end for
9: end for
```

where α is the learning rate and γ is the discount factor. The learning rate indicates to what extent newly acquired information overrides old information. Setting $\alpha = 0$ makes the agent learns nothing, whereas $\alpha = 1$ makes the agent consider only the most recent information. The discount factor sets the importance of future rewards. A value $\gamma = 0$ makes the agent short-sighted, only considering current rewards, whereas a value of γ close to 1 makes it strive for a long-term high reward.

The state variables need to be quantised due to the discrete nature of the Q-table. Therefore, the state of the system at time t is defined by $\hat{\boldsymbol{x}}^{(t)} = \left[\hat{\boldsymbol{B}}^{(t)}, h^{(t)}\right]$, where $\hat{\boldsymbol{B}}^{(t)} = \left[\hat{B}_1^{(t)}, \dots, \hat{B}_N^{(t)}\right]$ is a vector representing the quantised values of the battery levels of the SBSs in the cluster and h is a variable representing the hours of the day. The detailed steps of the QL algorithm are listed in Algorithm 3.

We highlight here that this algorithm has two main drawbacks:

- The continuous input space needs to be quantised due to the discrete nature of the Q-table, leading to quantisation errors;
- The dimension of the Q-table exponentially increases with the number of SBSs in the cluster. Therefore, the storage of the Q-table may not be feasible for clusters with a high number of SBSs.

These two drawbacks may have an important effect on the convergence time and the accumulated reward at convergence, as well as on the memory required and the computational complexity of the solution, specially in dense scenarios. A method based on deep RL is, thus, introduced to mitigate those drawbacks, and reduce the implementation complexity of the learning algorithm.

2) Deep Q-Learning: In the DQL approach, the Q-function is estimated by using a ANN approximator [30]. In details, the Q-function is approximated by the function $Q\left(\boldsymbol{x}^{(t)}, \boldsymbol{a}^{(t)} | \theta\right)$, where θ represents the ANN parameters. The state of the system is the input of the ANN, whereas the output layer corresponds to the predicted Q-values of the individual action for the input state. More in details, the number of neurons in the input layer is equal to $N+1$, i.e. the hour and the values of battery level of all the SBSs, whereas the number of neurons in the output layer is equal to 2^N, i.e. all possible

Algorithm 4 Deep Q-Learning algorithm

```
1:  Initialize the replay memory $\mathcal{R}$ to capacity $L$
2:  Initialize the ANN with random weights
3:  for episodes $= 1, ..., M$ do
4:      Initialize state $\boldsymbol{x}^{(1)}$
5:      for $t = 1, \ldots, T$ do
6:          Select action $\boldsymbol{a}^{(t)} = \max_{\boldsymbol{a}'} Q(\boldsymbol{x}^{(t)}, \boldsymbol{a}'; \theta)$ with probability $1 - \epsilon$ otherwise take a random action with probability $\epsilon$.
7:          Execute action $\boldsymbol{a}^{(t)}$ and observe the reward $r^{(t)}$ and the next state $\boldsymbol{x}^{(t+1)}$
8:          Store the experience $(\boldsymbol{x}^{(t)}, \boldsymbol{a}^{(t)}, r^{(t)}, \boldsymbol{x}^{(t+1)})$ in $\mathcal{R}$
9:          Sample a random batch of $l$ experiences $(\boldsymbol{x}^{(j)}, \boldsymbol{a}^{(j)}, r^{(j)}, \boldsymbol{x}^{(j+1)})$ from $\mathcal{R}$
10:         Set $y^{(j)} = r^{(j)}$ for terminal state $\boldsymbol{x}^{(j+1)}$ otherwise $y^{(j)} = r^{(j)} + \gamma \max_{\boldsymbol{a}'} Q(\boldsymbol{x}^{(j+1)}, \boldsymbol{a}'; \theta)$
11:         Perform gradient descend step on $\big(y^{(j)} - Q(\boldsymbol{x}^{(j)}, \boldsymbol{a}^{(j)}; \theta)\big)$
12:     end for
13: end for
```

combinations of ON/OFF operative modes of the SBSs in the cluster.

The agent experience at each time step t, $e_t = (\boldsymbol{x}^{(t)}, \boldsymbol{a}^{(t)}, r^{(t)}, \boldsymbol{x}^{(t+1)})$, is stored into a replay memory $\mathcal{R} = \{e_1, \ldots, e_L\}$ with dimension L. A batch of l experiences is randomly sampled from $\mathcal{R}$ and used to perform the update of the Q-function by training the ANN. This phase is defined as *experience replay*.

During the training of the ANN, stochastic gradient decent is used to minimise a sequence of loss functions that changes at every training iteration i,

$$L_i\left(\theta_i\right) = \mathbb{E}_{\boldsymbol{x},\boldsymbol{a}\sim\rho(\cdot)}\left[\left(y_i - Q\left(\boldsymbol{x}^{(t)}, \boldsymbol{a}^{(t)}; \theta_i\right)\right)\right]^2 \tag{13}$$

where $y_i = \mathbb{E}_{\boldsymbol{x}'\sim\varepsilon}\left[r + \gamma \max_{a'} Q\left(\boldsymbol{x}', \boldsymbol{a}'; \theta_{i-1}\right) | \boldsymbol{x}^{(t)}, \boldsymbol{a}^{(t)}\right]$ is the target at iteration i and $\rho\left(\boldsymbol{x}, \boldsymbol{a}\right)$ is the probability distribution over states and actions named *behaviour distribution*. The ANN parameters from the previous iteration, θ_{i-1}, are kept fixed when optimising the loss function $L_i\left(\theta_i\right)$. The parameters are updated at every iteration and the expectations are replaced by single samples from the behaviour distribution ρ and the state distribution ε. We highlight that this particular solution allows to simultaneously update the Q-values for all the possible actions in a given state with only one forward pass in the ANN. After performing the update of the ANN parameters, an action is taken according to the ϵ-greedy policy. The detailed steps of the DQL algorithm are listed in Algorithm 4.

This approach presents several advantages with respect to the standard QL. DQL allows to easily deal with the continuous input space and it take advantage of the generalisation capacity of ANNs to estimate the Q-function. In fact, all the ANN parameters θ are updated when training for a single state-action pair. On the contrary, QL estimates the Q-function independently for each state-action pair due to its tabular form. Moreover, the adoption of experience replay and the randomisation of the replay buffer allows reducing the variance of the ANN parameters θ during the training of the ANN by breaking the correlations between consecutive samples of the stored experience of the agent.

TABLE I
SIMULATION PARAMETERS

	Parameter	Value
Scenario	Solar panel size (m^2)	4.48 (16×16)
	Solar panel efficiency (%)	21
	Battery capacity (kWh)	2
	MBS transmission power (dBm)	43
	SBS transmission power (dBm)	38
	Bandwidth (MHz)	20
	Channel model	Okumura-Hata [31]
	Heavy users traffic (MB/h)	900
	Ordinary users traffic (MB/h)	112.5
	Heavy users percentage (%)	50
	Traffic profile	Residential, Office
SBS controller	Battery threshold B_{th} (%)	20
	High drop level D_{th} (%)	10
	Δ_t (hour)	1

VII. Numerical Results and Discussion

In this section, we discuss the numerical results achieved after our extensive simulation campaign. In particular, we present the performance of the training phase of the three ML models, analyse the learned policies and the energy shared in the micro-grid. After, we provide an analysis of the network performance in terms of the grid energy consumption and the dropped traffic. Finally, we discuss the energy and monetary cost savings achieved by the proposed RAN architecture.

A. Simulation Scenario

The scenario considered in this analysis consists of a single cluster of 1 MBS placed in the middle of a 1 km^2 area and N SBSs randomly deployed. In order to avoid overlap, the distance between SBSs is at least of 100 m, and SBSs have a maximum transmission power of 38 dBm. The UEs are distributed in a radius of 50 m from each SBSs to mimic a hot-spot scenario [32]. Each SBS is supplied by a solar panel of 4.48 m^2 area and a lithium ion battery of 2 kWh capacity. The solar panel consists of an array of 16×16 solar cells of Panasonic N235B solar modules that have a single cell efficiency of about 21%. These dimensions allow to fully recharge the batteries in a typical winter day, and are consistent with the results obtained in [19]. The solar energy arrivals are generated according to the city of Los Angeles, following the approach described in Section IV-B. The traffic demand is modelled as described in Section IV-C, considering a heavy user ratio of 50%. Further details about the simulation parameters are given in Table I.

Finally, the training of all the learning algorithms is performed by considering a cluster of 3 SBSs. The simulations have been run on a machine with an Intel®Core$^{\mathrm{TM}}$ i5-6300U CPU @ 2.40GHz and 8 GB of RAM.

B. Imitation Learning Training

A dataset containing the offline policies computed over one year has been split into a training dataset and a validation

TABLE II
TRAINING AND VALIDATION PERFORMANCE FOR DIFFERENT ANN ARCHITECTURES

Number of neurons			Training			Validation		
1st layer	2nd layer	3rd layer	Loss	Accuracy	MA F-score	Loss	Accuracy	MA F-score
8	-	-	0,17	0,93	0,93	0,19	0,93	0,93
32	-	-	0,15	0,94	0,94	0,16	0,93	0,93
64	-	-	0,14	0,94	0,94	0,16	0,93	0,93
512	-	-	0,11	0,95	0,95	0,15	0,94	0,94
32	32	-	0,12	0,95	0,95	0,14	0,95	0,95
64	64	-	0,10	0,95	0,95	0,15	0,94	0,94
32	32	32	0,11	0,95	0,95	0,16	0,94	0,94

dataset, as described in Section VI-A. The training dataset has been used to train ANN classifiers with different architectures using the Adam version of gradient descent [33], with the goal of minimising the categorical cross entropy, introduced in (10).

The loss, the accuracy and the macro average (MA) F-score measured on the training and validation set, are reported in Table II. The accuracy is defined as the number of correct predictions divided by the total number of predictions whereas the MA F-score is computed as the average of the categories F-scores, which are:

$$\mathrm{F1} = 2 \cdot \frac{p \cdot r}{p + r}, \tag{14}$$

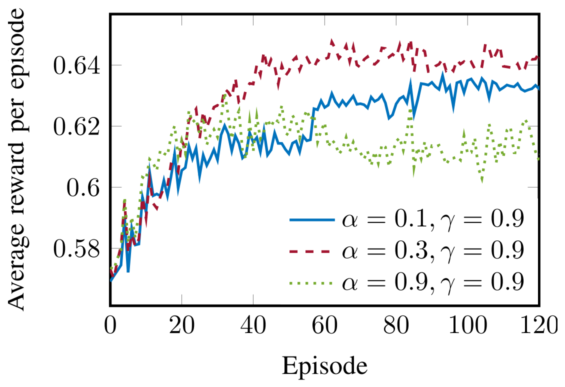
where p is the number of correct positive results divided by the number of all positive results (i.e. precision), whereas r is the number of correct positive results divided by the number of all the samples that should have been identified as positive (i.e. recall).

The ANN architecture with two hidden layers of 32 neurons is selected, based on the maximum accuracy, MA F-score, training velocity (=300 epochs). Adding more neurons/layers leads to an increment of the computational complexity, without any positive impact to the performance.

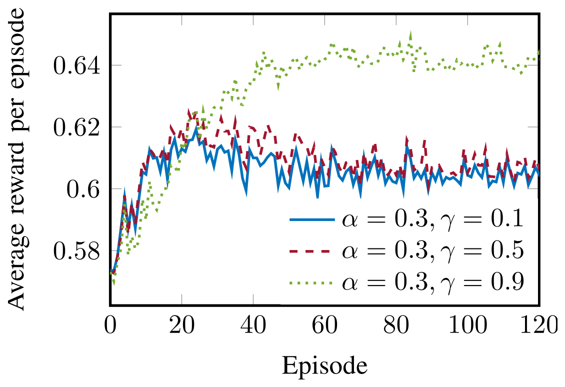
C. Q-Learning Training

An analysis on the influence of the learning parameters to the training performance is presented in this section for the QL algorithm. The QL algorithm exploration rate is set to $\epsilon = 0.9$ at the beginning of the training, and discounted by 10% at each episode, until reaching the minimum value of $\epsilon_{\min} = 0.05$. In this way, the exploration is prioritised during the first phase of the training, enabling a faster convergence. Moreover, the battery levels $\mathbf{B}$ are quantised by considering a uniform quantiser with 3 levels, whereas the hour of the day h is a variable quantised with 24 levels. This choice represents a good trade-off between achieved performance and computational complexity, according to the performed simulations.

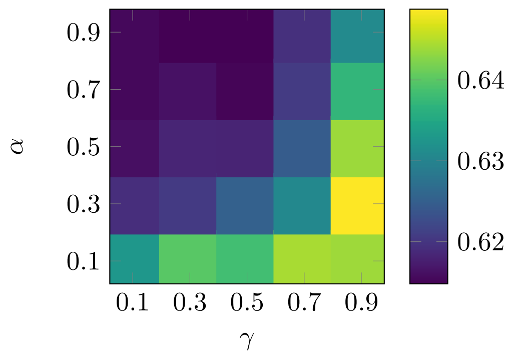
Figure 4 reports the performance obtained for different setups of the training parameters. The average reward per episode achieved by the QL agent for different values of the learning rate are shown in Figure 4(a). The highest reward is achieved for $\alpha = 0.3$. Using a smaller learning rate (e.g. $\alpha = 0.1$) makes the agent to converge slower to the maximum average reward, which is additionally slightly smaller than $\alpha = 0.3$. Instead, higher oscillations are experienced for a



(a)

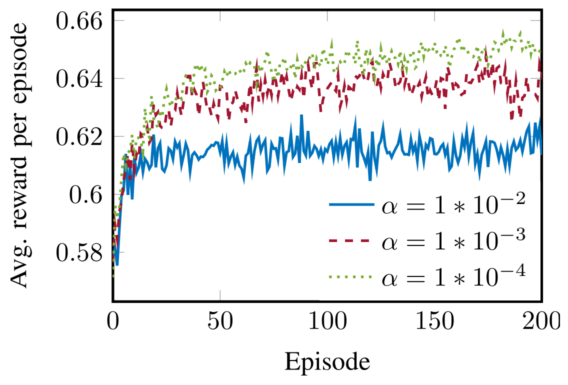


(b)

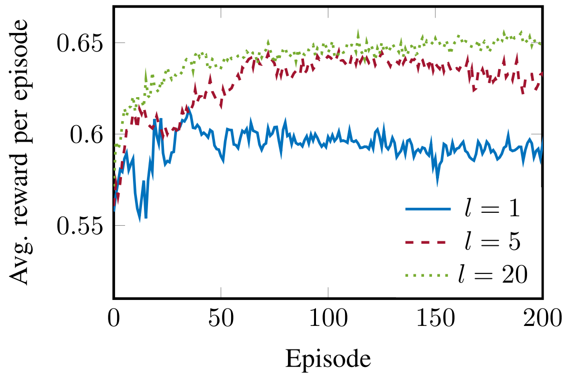


(c)

Fig. 4. Average reward per episode of the QL algorithm when adopting different values of learning rate (a) and discount factor (b). Values of average reward per episode achieved at convergence are indicated in (c).



(a)



(b)

Fig. 5. Average reward per episode of the DQL algorithm when adopting different values of learning rate (a) and batch size (b). The dimension of the batch is $l = 20$ in (a) whereas the learning rate is $\alpha = 1 \cdot 10^{-4}$ in (b). The discount factor is $\gamma = 0.9$.

bigger learning rate (e.g. $\alpha = 0.9$), leading to a low average reward per episode.

Figure 4(b) shows the average reward per episode for three different values of the discount factor. The highest average reward is achieved for $\gamma = 0.9$, which implies that the system needs to be optimised over a long horizon. In fact, a big value of γ makes the agent looking for a long-term high reward by giving more importance to future rewards.

Finally, Figure 4(c) shows the average reward per episode achieved at convergence for different values of the learning rate α and discount factor γ. The highest average reward is achieved for $\alpha = 0.3$ and $\gamma = 0.9$.

D. Deep Q-Learning Training

An analysis of the influence of the learning parameters to the training performance is presented in this section for the DQL algorithm. In particular, Figure 5 reports the performance achieved for different values of the learning rate and the batch size.

Figure 5(a) shows the average reward per episode achieved by the DQL agent for three different values of the learning rate. The discount factor and the batch size are fixed to $\gamma = 0.9$ and $l = 20$, respectively. The value at convergence and the number of episodes needed to converge are influenced by the learning rate. In particular, big values of α make rapidly achieve a stable average reward per episode, whereas smaller learning rates lead to a slower convergence, but a higher average reward per episode in the long run.

TABLE III
AVERAGE REWARD PER EPISODE FOR DIFFERENT ANN ARCHITECTURES

Number of neurons			Avg. reward per episode
1st layer	2nd layer	3rd layer	
10	-	-	0.63
20	-	-	0.64
20	20	-	0.64
50	-	-	0.65
50	50	-	0.66
50	50	50	0.65
100	-	-	0.65
100	100	-	0.65

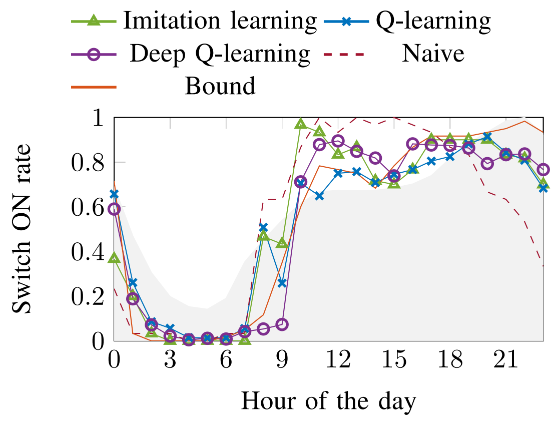
Figure 5(b) shows how the training performance is affected by the batch size l. The average reward per episode shows high oscillations and a slow convergence if experience replay is not used (i.e. $l = 1$). Instead, enabling it allows to reduce the magnitude of the oscillations and the number of epochs needed to converge. We highlight here that this does not come for free in terms of execution time. In fact, the running time per episode increases linearly with the dimension of the batch size l, since the number of forward and backward passes through the ANN scales linearly with l.

Finally, the average reward per episode achieved varying the numbers of neurons per layer and the number of layers is reported in Table III. The average reward per episode increases with the dimension of the ANN till a performance limit. Increasing the dimension of the ANN only leads to longer training times without any positive influence on the average reward per episode. The best average reward per episode is achieved with an ANN architecture composed of two layers, both consisting of 50 neurons. Moreover, the best results have been obtained using the linear activation function (i.e. $f_a(x) = x$) for the output layer and the ReLU activation function (i.e. $f_a(x) = \max(0, x)$) for the input and hidden layers. We experimentally noticed that the property of ReLU of turning any negative argument into zero helps the convergence speed and the average reward per episode at convergence.

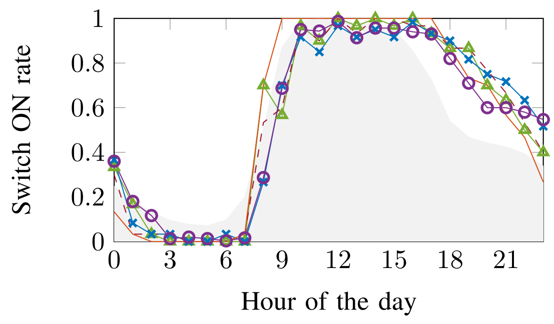
E. ON-OFF Policies

In this section, we analyse the daily average SBSs switch ON rate experienced by the three different implementations of the agent in a cluster of 3 SBSs. In particular, the switch ON rates experienced in December and July are reported in Figure 6 and Figure 7, respectively, for a residential and an office area. Those months represent the worst and the best case in terms of harvested energy, respectively. The shape of the traffic demand profiles is indicated by the shaded areas. The three agent implementations are compared with a naive approach that operates by turning OFF the SBSs when their battery levels go below the threshold B_{th} and switching them ON when the level is above it. The performance bound using the offline optimisation is also reported in the figures as a reference. We refer to this policy as *Bound* policy.

In general, the policy works so that the SBSs save energy during nights, when the traffic is low, and provide service

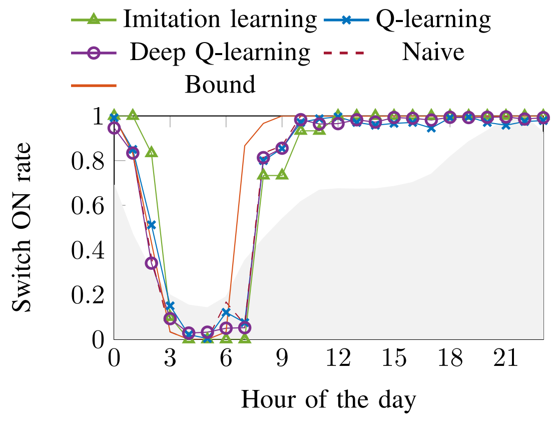


(a) Residential area

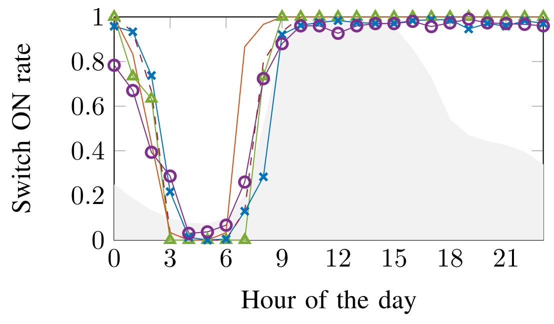


(b) Office area

Fig. 6. Average switch ON rate of the SBSs when adopting different policies in a residential (a) and office (b) area in the months of December. The shaded area represents the shape of the traffic demand.



(a) Residential area



(b) Office area

Fig. 7. Average switch ON rate of the SBSs when adopting different policies in a residential (a) and office (b) area in the months of July. The shaded area represents the shape of the traffic demand.

during the high traffic demand hours. The length of the night switch OFF period depends on the energy availability and, thus, on the season of the year. In particular, longer switch OFF periods are generally observed in December for both the traffic areas.

Figure 6(a) shows the switch ON rate for a residential area in the month of December.

The bound policy is characterised by a high switch ON rate of 0.94 in average during the traffic peak, i.e. from 6 pm to midnight, whereas an average switch ON rate of 0.75 is experienced from 10 am to 3 pm. This means that the bound policy is partially switching OFF the SBSs during those hours to save energy needed to provide service during the peak hours.

The naive policy is switching ON the SBSs as soon as sufficient energy is harvested in the morning. Therefore, high switch ON rate is measured from 10 am to 3 pm, i.e. in correspondence with the peak of the harvesting process. Then, as soon as the energy reserves fall below the threshold B_{th}, the switch ON rate decreases and reaches low values during the peak of the traffic demand, leading to high traffic drop.

The QL and DQL policies have an average switch ON rate of 0.7 and 0.8, respectively, between 10 am and 3 pm. Those values highlight that the adopted policies are following the behaviour of the bound policy: they partially switch OFF the SBSs during the daytime and maintain the switch ON rate to an average of 0.8 during the peak of the traffic demand. The same behaviour is reproduced by the IL algorithm. We highlight here that the behaviour of the QL, DQL and IL policies follows the behaviour of the bound policy with some differences due to the stochastic nature of the energy and traffic processes.

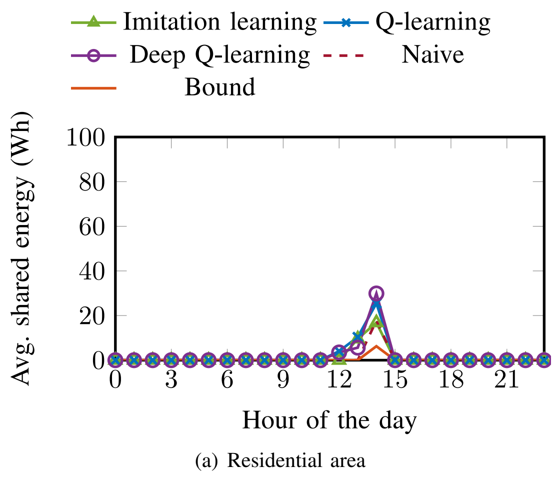
Figure 6(b) shows the switch ON rate for an office area in the month of December. The office traffic profile is easier to manage, since the peak of the traffic demand coincides with the period of more intensive energy arrivals (i.e. around midday). In all the studied cases, the SBSs are switched ON after the night sleep starting from 7 am, and a high switch ON rate is experienced from 10 am to 5 pm for all the policies (i.e. during the peak of the traffic demand).

Finally, Figure 7(a) and Figure 7(b) show the switch ON rate in the month of July for the residential and the office areas, respectively. In both the cases, bound policies show an increase of the switch ON rate after 6 am, whereas in the case of the naive and learning agents, it starts raising from 7 am. No other particular difference from the studied algorithms is noticed due to the high availability of harvested energy in summer, which allows the agent to keep the SBSs ON until midnight in both the traffic areas regardless the specific implementation.

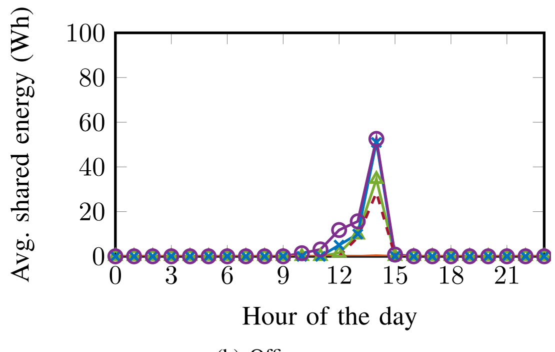
F. Shared Energy Assessment

In this section, we analyse the behaviour of the different implementations of the agent in terms of the daily energy shared between SBSs and MBS. The average amount of energy shared with the MBS and used for its operations is reported in Figure 8 and Figure 9 for the month of December and July, respectively. Results are provided for both a residential and an office area.

In general, the energy is shared around noon during a period of time whose duration depends on the season of the year.

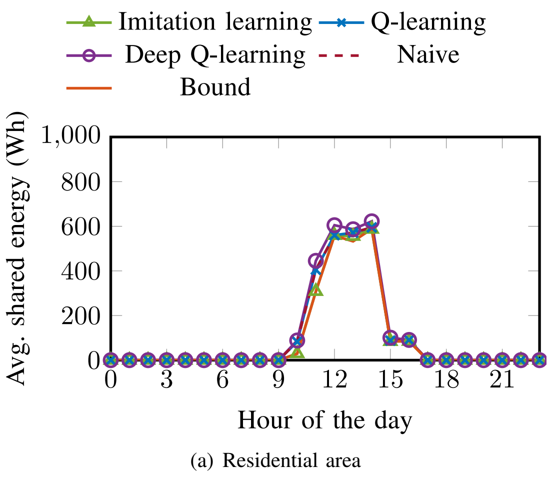


(a) Residential area

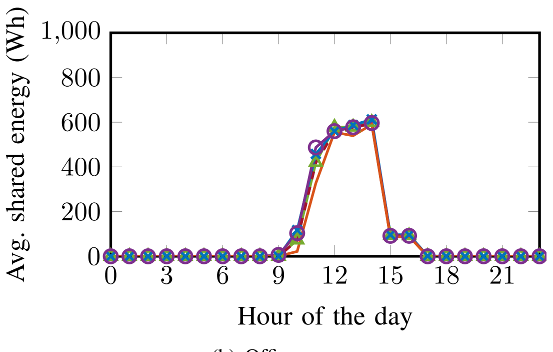


(b) Office area

Fig. 8. Average amount of shared energy used by the MBS in December when adopting different policies in a residential (a) and office (b) area.



(a) Residential area



(b) Office area

Fig. 9. Average amount of shared energy used by the MBS in July when adopting different policies in a residential (a) and office (b) area.

The energy in December is shared between 11 am and 3 pm, whereas in July is shared from 10 am to 4 pm, which is longer since during summer the SBSs harvest more solar energy.

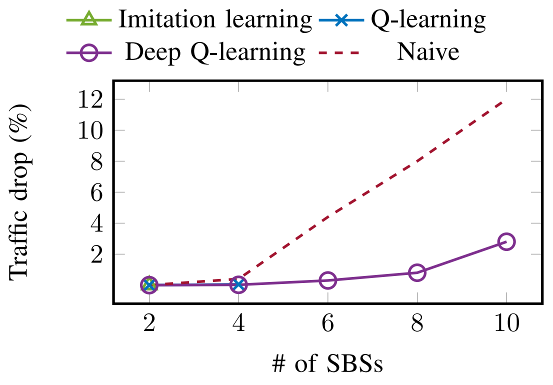
The bound policy shares less energy with respect to the other policies in all the considered scenarios. This is motivated by the fact the bound policy maximises the amount of traffic which is offloaded to the SBSs and, thus, the local use of the harvested energy. This behaviour is confirmed from the high switch ON rates in Figure 6 and Figure 7, and is made possible by the prior knowledge of the temporal variation of the energy and traffic processes.

The naive policy shares less energy than the three implementations based on ML. In fact, the naive policy maintains the SBSs active as long as their energy is available. This policy reduces the probability of filling the batteries and, thus, of sharing the excess energy.

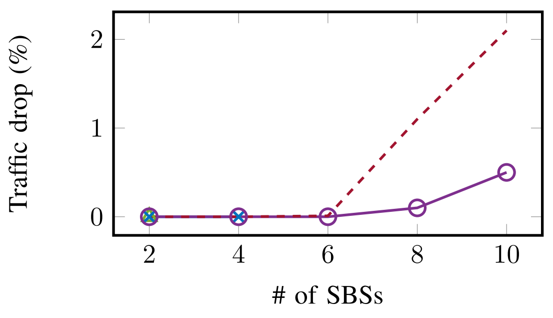
The IL, QL and DQL policies show similar behaviours. In particular, the DQL agent is that leading to higher energy sharing with an average of 39 Wh and 89 Wh in a residential and office area, respectively, for a day of December. Higher values are measured in the month of July, where DQL is sharing an average of 2.54 kWh and 2.51 kWh per day in the residential and office area, respectively.

G. Network Performance

In this section, we compare the three learning algorithms in terms of grid energy consumption and traffic drop. The

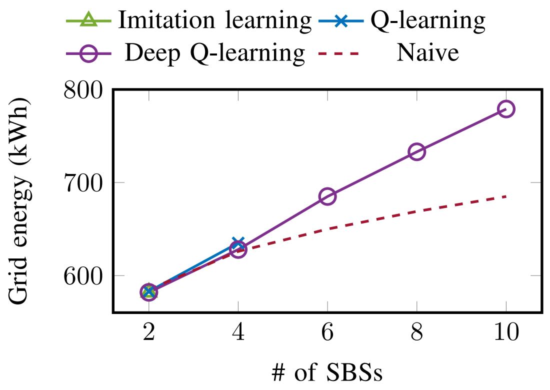


(a) Residential area

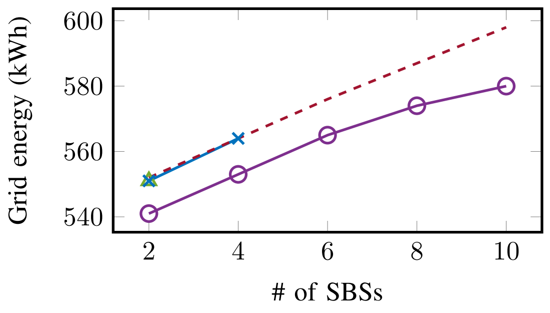


(b) Office area

Fig. 10. Percentage of dropped traffic experienced in the month of December, when adopting the naive policy and the policy obtained by the proposed learning approaches.



(a) Residential area



(b) Office area

Fig. 11. Grid energy consumption experienced in the month of December, when adopting the naive policy and the policy obtained by the proposed learning approaches.

results have been obtained by setting each algorithm with the best configuration of the training parameters, as discussed in Section VII.B-D. In particular, we focus on the performance achieved in the month of December (i.e. the worst in terms of harvested energy).

Figure 10 and Figure 11 show the traffic drop and grid energy consumption performance achieved when considering different cluster sizes in a residential and office area, respectively. In particular, we consider a scenario with 100 UEs deployed in the area of each SBSs, such that the traffic demand is linearly increasing with the number of deployed SBSs in the cluster.

The IL approach cannot be adopted in scenarios with more than 3 SBSs due to the high computational complexity and the amount of memory required by the bound algorithm and the resulting unavailability of training data. Similarly, the QL approach cannot be adopted in scenarios with more than 4 SBSs, due to the high amount of memory required to store the Q-table, which increases exponentially with the number of SBSs (i.e. $|\hat{B}_i^{(t)}|^N \times |h^{(t)}| \times 2^N$). On the contrary, DQL limits this problem thanks to the introduction of the ANN function approximation, in which the number of weights that needs to be estimated is lower than the number of Q -table elements estimated by QL. This makes the memory footprint of the DQL controller much lower, and allows its operations for a network with a higher number of SBSs in the cluster.

Figure 10 shows the traffic drop experienced in a residential and an office area. The ML agents reach the same performance in the scenario with 2 SBSs, whereas DQL reduces the drop by 60% with respect to QL in the scenario with 4 SBSs in a residential area. This is mainly due to the possibility of using continuous state variables as input of the ANN and avoid the quantisation errors of QL.

IL, QL and DQL algorithms lead to lower traffic drop than the naive approach for all the considered cluster dimensions. In the case of DQL, this aspect is more evident for deployments with more than 4 SBSs in the residential area and 6 SBSs in the office area. In those cases, the traffic drop experienced by the naive policy linearly increases with the number of SBSs. We highlight here that DQL reaches significant savings up to 90% with respect to the naive algorithm for both traffic profiles.

The amount of grid energy consumed by the MBS in the month of December is reported in Figure 11, for the residential and the office traffic profiles. DQL saves up to 3% and 1% of the grid energy than IL, QL for all the considered cluster dimensions in office and residential area, respectively. This translates into a high energy efficiency, since DQL is able to serve more traffic. We highlight that the naive policy consumes less grid energy when considering a cluster with more than 4 SBSs in a residential area. This is motivated by the high amount of traffic that is not served by this policy.

H. Energy Savings and Cost Analysis

In this section, we provide an analysis of the grid energy consumed in one year and monetary costs experienced after five and ten years for different cluster dimensions that an operator may achieve though the implementation of the proposed DQL algorithm. They are reported in table IV considering two different traffic areas, namely residential and office. Note that in this analysis a trained version of the DQL agent is implemented in a new environment, characterised by instances of the energy arrival and traffic demand processes that are different from those used during the training phase. In such a way, we generalise the behaviour of the agent and get performance as close as a real implementation in an operative network.

The performance achieved by DQL are compared with a network scenario in which all the BSs are supplied by the power grid, also referred to as *grid-connected*. We consider a cost of 1.17 \$/W for the solar panel (which also includes the installation cost) and 131 \$/kWh for the battery. Moreover, the grid energy has a cost of 0.21 \$/kWh.

The harvesting/storage hardware jointly with the centralised network control based on DQL allows to reduce the grid energy consumption in both the traffic areas. The savings increase as the network deployment is more dense (i.e. more SBSs in the cluster): from 31% to 58% higher than the grid-connected case in the residential area and from 34% to 68% in the office area. Operational Expenditures are decreased accordingly. Costs after 10 years are significantly reduced up to 35% and 44% in the residential and the office area, respectively.

VIII. Conclusions

In this paper, we have proposed a RAN setup in which a hierarchical cell structure is deployed within the same geo-

TABLE IV
ENERGY AND COST ANALYSIS

Traffic	# of SBS	Grid Energy consumption (kWh)			Cost at 5 years ($)			Cost at 10 years ($)		
		Grid connected	DQL	Saving	Grid connected	DQL	Saving	Grid connected	DQL	Saving
Residential	2	8,312	5,753	-31%	8,728	7,735	-11%	17,455	13,775	-21%
	4	10,580	5,932	-44%	11,109	9,617	-13%	22,218	15,845	-29%
	6	12,848	6,292	-51%	13,490	11,689	-13%	26,981	18,295	-32%
	8	15,116	6,560	-57%	15,872	13,664	-14%	31,744	20,552	-35%
	10	17,384	7,236	-58%	18,253	16,068	-12%	36,506	23,666	-35%
Office	2	8,301	5,500	-34%	8,716	7,469	-14%	17,432	13,244	-24%
	4	10,554	5,512	-48%	11,081	9,617	-17%	22,163	14,963	-32%
	6	12,806	5,536	-57%	13,447	10,895	-19%	26,893	16,708	-38%
	8	15,059	5,504	-63%	15,812	12,555	-21%	31,624	18,334	-42%
	10	17,312	5,576	-68%	18,178	14,325	-21%	36,355	20,180	-44%

graphical area with BSs of different scale factors, transmission power, computational capabilities and coverage areas. This federation of BSs together with the distributed harvesters and storage devices at SBSs sites form a micro-grid, whose operations are managed by an energy management system in charge of controlling the intermittent and erratic energy budget from the RESs. We have focused on the design of online algorithms capable of jointly control sleep mode and energy sharing within the micro-grid. Three different implementations of ML models are proposed for the central agent, namely Imitation Learning, Q-Learning and Deep Q-Learning. We have discussed on the achieved performance, complexity and feasibility of those different ML approaches. The DQL algorithm tailored for our scenario presents higher performance in terms of energy saving and system outage, and is able to control highly dense scenarios, where the other approaches fail due to complexity and memory issues. Finally, the energy and cost analysis reported in this work provides an insight on the greater savings that can be achieved by an operator implementing our Deep Q-Learning algorithm in the central controller of the RAN setup with energy harvesting capabilities.

ACKNOWLEDGMENTS

This work has received funding from the European Union's Horizon 2020 research and innovation programme under the Marie Sklodowska-Curie grant agreement No 675891 (SCAVENGE) and by Spanish MINECO grant TEC2017-88373-R (5G-REFINE).

REFERENCES

[1] International Telecommunication Union, "ICT Facts & Figures 2016," [Online]. Available: http://www.itu.int/en/ITU-D/Statistics/Documents/facts/ICTFactsFigures2016.pdf, June 2016 (accessed May, 2017).

[2] Cisco, "Cisco Visual Networking Index: Global Mobile Data Traffic Forecast Update, 2016-2021," [Online]. Available: http://www.cisco.com/c/en/us/solutions/collateral/service-provider/visual-networking-index-vni/mobile-white-paper-c11-520862.html, February 2017 (accessed May, 2017).

[3] M. Wollschlaeger, T. Sauter, and J. Jasperneite, "The future of industrial communication: Automation networks in the era of the internet of things and industry 4.0," *IEEE industrial electronics magazine*, vol. 11, no. 1, pp. 17–27, 2017.

[4] M. P. Mills, "The cloud begins with coal: Big data, big networks, big infrastructure, and big power," *Digital Power Group*, August 2013.

[5] A. S. Andrae and T. Edler, "On global electricity usage of communication technology: trends to 2030," *MDPI Challenges*, vol. 6, no. 1, pp. 117–157, April 2015.

[6] F. Han, S. Zhao, L. Zhang, and J. Wu, "Survey of Strategies for Switching Off Base Stations in Heterogeneous Networks for Greener 5G Systems," *IEEE Access*, vol. 4, pp. 4959–4973, August 2016.

[7] J. Wu, Y. Zhang, M. Zukerman, and E. K.-N. Yung, "Energy-efficient base-stations sleep-mode techniques in green cellular networks: A survey," *IEEE Communications Surveys & Tutorials*, vol. 17, no. 2, pp. 803–826, February 2015.

[8] H. A. H. Hassan, L. Nuaymi, and A. Pelov, "Renewable energy in cellular networks: A survey," in *2013 IEEE Online Conference on Green Communications (OnlineGreenComm)*, October 2013, pp. 1–7.

[9] D. Zordan, M. Miozzo, P. Dini, and M. Rossi, "When telecommunications networks meet energy grids: Cellular networks with energy harvesting and trading capabilities," *IEEE Communications Magazine*, vol. 53, no. 6, pp. 117–123, June 2015.

[10] D. López-Pérez, M. Ding, H. Claussen, and A. H. Jafari, "Towards 1 Gbps/UE in cellular systems: Understanding ultra-dense small cell deployments," *IEEE Communications Surveys & Tutorials*, vol. 17, no. 4, pp. 2078–2101, June 2015.

[11] R. S. Sutton and A. G. Barto, *Reinforcement learning: An introduction*. MIT press, 2018.

[12] J. Gong, J. S. Thompson, S. Zhou, and Z. Niu, "Base station sleeping and resource allocation in renewable energy powered cellular networks," *IEEE Transactions on Communications*, vol. 62, no. 11, pp. 3801–3813, 2014.

[13] G. Lee, W. Saad, M. Bennis, A. Mehbodniya, and F. Adachi, "Online Ski Rental for ON/OFF Scheduling of Energy Harvesting Base Stations," *IEEE Transactions on Wireless Communications*, vol. 16, no. 5, pp. 2976–2990, May 2017.

[14] S. Zhou, J. Gong, and Z. Niu, "Sleep control for base stations powered by heterogeneous energy sources," in *2013 International Conference on ICT Convergence (ICTC)*. IEEE, 2013, pp. 666–670.

[15] N. Piovesan and P. Dini, "Optimal direct load control of renewable powered small cells: A shortest path approach," *Internet Technology Letters*, pp. e7–n/a, 2017, e7.

[16] M. Miozzo, L. Giupponi, M. Rossi, and P. Dini, "Distributed Q-learning for energy harvesting Heterogeneous Networks," in *IEEE International Conference on Communication Workshop (ICCW)*, London, UK, June 2015, pp. 2006–2011.

[17] M. Miozzo, N. Piovesan, and P. Dini, "Coordinated load control of renewable powered small base stations through layered learning," *IEEE Transactions on Green Communications and Networking*, pp. 1–1, 2019.

[18] J. Xu, L. Duan, and R. Zhang, "Cost-aware green cellular networks with energy and communication cooperation," *IEEE Communications Magazine*, vol. 53, no. 5, pp. 257–263, May 2015.

[19] N. Piovesan, D. A. Temesgene, M. Miozzo, and P. Dini, "Joint load control and energy sharing for autonomous operation of 5g mobile networks in micro-grids," *IEEE Access*, vol. 7, pp. 31 140–31 150, 2019.

[20] B. Lindemark and G. Oberg, "Solar power for radio base station (rbs) sites applications including system dimensioning, cell planning and operation," in *Twenty-Third International Telecommunications Energy Conference INTELEC 2001*, October 2001, pp. 587–590.

[21] US National Renewable Energy Laboratory (NREL). National Solar Radiation Data Base, 1991-2010 Update. [Online]. Available: https://rredc.nrel.gov/

[22] N. Piovesan and P. Dini, "Unsupervised learning of representations from solar energy data," in *2018 IEEE 29th Annual International Symposium on Personal, Indoor and Mobile Radio Communications (PIMRC)*, Sep. 2018, pp. 1–6.

[23] G. Auer, O. Blume, V. Giannini, I. Godor, M. 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," Project Deliverable D2.3, www.ict-earth.eu, January 2013.

[24] F. Xu, Y. Li, H. Wang, P. Zhang, and D. 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.

[25] H. D. Trinh, N. Bui, J. Widmer, L. Giupponi, and P. Dini, "Analysis and modeling of mobile traffic using real traces," in *2017 IEEE 28th Annual International Symposium on Personal, Indoor, and Mobile Radio Communications (PIMRC)*. IEEE, 2017, pp. 1–6.

[26] M. Mezzavilla, M. Miozzo, M. Rossi, N. Baldo, and M. Zorzi, "A lightweight and accurate link abstraction model for the simulation of lte networks in ns-3," in *Proceedings of the 15th ACM International Conference on Modeling, Analysis and Simulation of Wireless and Mobile Systems*, ser. MSWiM '12. New York, NY, USA: ACM, 2012, pp. 55–60. [Online]. Available: http://doi.acm.org/10.1145/2387238.2387250

[27] D. P. Bertsekas, *Dynamic programming and optimal control*. Athena scientific Belmont, MA, 1995, vol. 1, no. 2.

[28] T. Osa, J. Pajarinen, G. Neumann, J. A. Bagnell, P. Abbeel, J. Peters *et al.*, "An algorithmic perspective on imitation learning," *Foundations and Trends® in Robotics*, vol. 7, no. 1-2, pp. 1–179, 2018.

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

[30] V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, and M. Riedmiller, "Playing atari with deep reinforcement learning," *arXiv preprint arXiv:1312.5602*, 2013.

[31] 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.

[32] E. U. T. R. Access, "Further advancements for e-utra physical layer aspects, 3gpp ts 36.814," *V9. 0.0, Mar*, 2010.

[33] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," 2014.