Deep Reinforcement Learning for cell on/off energy saving on Wireless Networks

Joan S. Pujol-Roig[1], Shangbin Wu[1], Yue Wang[1], Minsuk Choi[2], and Intaik Park[2]

[1]Samsung Electronics R&D Institute UK, Surrey, TW18 4QE, UK
[2]Samsung Research, Seoul R&D Campus, Umyeon dong, Seoul, Republic of Korea

GLOBECOM 2021 - 2021 IEEE Global Communications Conference | 978-1-7281-8104-2/21/$31.00 ©2021 IEEE | DOI: 10.1109/GLOBECOM46510.2021.9685279

***Abstract*—Increased network traffic demands have led to extremely dense network deployments. This translates to significant growth in energy consumption at the radio access networks, resulting in high network operation costs (OPEX). In this work, we apply deep reinforcement learning to reduce the energy consumption at the base station in dense wireless networks, by allowing cells that overlap in geographical areas to be put in standby mode according to the changing network conditions. We start by formulating the problem of the cell on/off energy saving in dense wireless networks as a Markov decision process. Then, a deep reinforcement learning (DRL) solution is proposed. This DRL solution takes into account different key performance indicators (KPIs) of both the network and user equipment and aims to reduce the energy consumed by the network without significantly impacting the overall KPIs. The performance of the proposed solution is evaluated using a practical network simulator.**

***Index Terms*—Reinforcement learning, Energy Saving, Cell on/off, Deep Neural Networks.**

I. Introduction

Mobile network operators are facing significant challenges in providing fast and seamless services due to the increase in network traffic. One way to cope with traffic growth is denser network deployments, which reduce the physical distances between the base stations, increasing the number of overlapping cells over the same area. As a result, geographical areas that once were covered by one macro base station are now being covered by twenty or more small base stations (SBS) [1]. Nonetheless, the increase in the number of SBS entails a surge in energy consumption which in turn increases the network operational expenditures (OPEX), as 39% of the network OPEX can come from power consumption costs [2]. It also brings environmental concerns, as information and communication technology (ICT) already account for 2.5% of the world's CO_2 emissions, a figure that is expected to rise [1]. To revert this tendency, new energy-saving solutions, that make future networks more eco-friendly with reduced OPEX, are desired.

Current network operations are tailored to cope with peak traffic periods load, which accounts for approximately 28% of the network uptime [3], resulting in network resources being underutilized the rest of the time, wasting energy unnecessarily. Besides, modern radio access networks (RAN) solutions allow SBS to be placed on standby mode or even to be completely switched off. By designing a network that is traffic-aware, the cellular traffic dynamics can be exploited to increase the cells energy efficiency, so that SBSs can be switched on/off depending on the traffic demands. This work aims to provide a traffic-aware solution to dynamically manage cell operations and decide when SBSs are to be turned on/off based on the changing network conditions.

Extensive research has emerged in the literature addressing energy saving in cellular wireless networks. For example, a cell coverage adjustment was proposed in [4] and [5] to reduce the network's cell size, and thus the power consumption, based on the serving UEs channel conditions relative to SBSs. The algorithm presented in [6] dynamically adjusts the energy consumption of a base station according to the maximum cell load of that base station and the load of the neighboring base stations. In [7] a network configuration optimization is proposed to save energy by leveraging traffic prediction.

Another approach is to deactivate co-located cells on a sectorial deployment. To this end, authors in [8] dynamically migrate users across different cells and optimize base station operations by changing the on/off states of base station cells to reduce energy consumption using a greedily algorithm. A two-tier network power saving is discussed in [9], where power consumption is minimized by computing the minimum separation distance between two cell types. A channel-aware energy-saving solution for cognitive wireless sensor network (WSN) was modeled as a hybrid game in [10], where each WSN node decides a transmission strategy according to the change of its channel state. In [11], authors formulate the on/off switching decision of the network cells as a linear integer programming problem, which is solved by relaxing the constraints and employing a series of Lagrangian dual methods. In [12], authors propose an algorithm that relies on a heuristic to determine a subset of SBS that can be placed in low-consumption mode, showing a 53% energy savings in dense areas, and 23% in sparse areas. The work in [13] also considers taking on/off actions on a subset of SBSs. Through Gamma approximation, the authors analyze the network load and based on the gathered statistics, a centralized and decentralized on/off scheduling strategies are proposed.

Energy savaging has also been extensively discussed in standards. In 4G long-term evolution (LTE) and 5G new radio (NR), various energy-saving solutions have been proposed mostly in the frame format configurations. For example, multimedia broadcast multicast service single frequency network (MBSFN) subframes have less common reference signals

978-1-7281-8104-2/21/$31.00 ©2021 IEEE

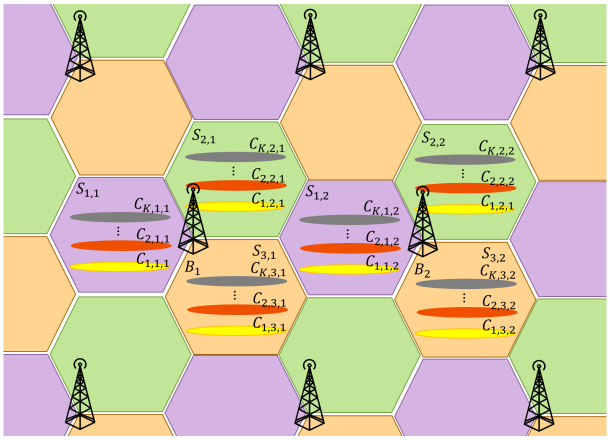


Fig. 1: Network deployment considered, where S denotes the ENB sector, and C the sector's cell.

(CRSs) than normal subframes and, if a cell's load is low, more MBSFN subframes can be configured to reduce power consumption [14]. Besides frame format configurations, sleeping intervals are also supported to reduce power consumption.

More recently, reinforcement learning (RL) was employed for energy saving in wireless networks. In [15], a deep reinforcement learning (DRL) solution for the management and orchestration (MANO) of resources used by virtual network functions is presented. The solution accounts for, among other KPIs, the energy consumption at the time to decide the resource allocated to each virtualised network function (VNF). Authors in [16] leverage Q-learning to optimize sleeping intervals so that energy consumption is reduced while latency constraints are met. In [17], authors propose a Q-Learning algorithm that selects a BS sleep mode depending on the geographical location and moving velocity of UEs in order to maximizes the trade-off between energy savings and delay. Deep convolution neural network (DCNN) assisted RL was also proposed in [18] to perform network power optimization. The location information, traffic map, throughput map, and loads of cells are used to construct a two-dimensional matrix as input to the DCNN, which outputs a binary decision for each cell to be placed in on/off mode accordingly.

In this work, we consider a 3-sector multiple-eNB deployment, with each sector having several operational cell bands. Similar to the aforementioned works, this work aims to minimize the overall network energy consumption while minimizing the impact on the UEs. To reduce energy consumption and increasing the efficiency of the network resources, each cell can be put in standby or active mode using a set of thresholds, such that underused resources can be switched off and turned back on when needed. To do so, this paper leverages DRL, in particular, we adapt the solution presented in [19] to obtain continuous values for load threshold selection, which are used to control the operational modes of the SBS of the eNBs. Numerical results of the proposed solution are presented using a network simulator. The results show that the DRL solution proposed in this work can achieve an average power saving of up to 27% in the simulated environment.

Compared to existing works in the literature, the main advantages of the DRL based energy saving solution presented in this paper are the following:

- While most of the previous aforementioned works propose static solutions, designed for specific traffic patterns, and reactive, as the criteria to trigger a cell management decision is made beforehand [8]–[13], this paper presents a solution that is proactive and dynamic, i.e., an agent learns from the traffic itself and understands the network context to trigger different responses accordingly.
- In contrast to [16], [17], we select a continuous value for a set of thresholds, resulting in a more versatile solution than selecting a sleeping time from a small set of predefined discrete-time intervals.
- We rely on less parameters than [18], i.e., we do not need to gather information such as UE location information, throughput map, etc. to enable power saving. This leads to less monitoring cost in a practical network.
- A realistic network simulator, with static, moving UEs, and traffic profiles obtained from practical network, has been used to develop and validate the DRL solution, hence proving a robust DRL ready to be deployed in practical settings. Although it is not feasible to compare results with existing solutions due to different network settings and data sets, we provide a comparison of the proposed DRL with existing rule-based methods in Section IV, where considerable power saving is observed.

II. System Model

In this work, we consider a wireless network with B eNBs, denoted by $\mathcal{B} = \{\mathrm{B}_1, \mathrm{B}_2, \ldots, \mathrm{B}_B\}$. Each of the eNBs implements a 3-sector deployment, we denote by $\mathrm{S}_{i,j}$ sector $i, \forall i \in [3]$ [1] of eNB B_j. Similarly each sector can be further broken down into cells denoted by $\mathcal{C} = \{\mathrm{C}_{1,i,j}, \mathrm{C}_{2,i,j}, \ldots, \mathrm{C}_{K,i,j}\}$, where $\mathrm{C}_{k,i,j}$ represents cell k of sector S_i from eNB B_j, and there are a total of K cells types. The network considered in this work is depicted in Figure 1. Each cell type C_k have different frequency bands at which they operate and bandwidth sizes, which are denoted by f_k and BW_k, respectively. As the notation indicates, both the frequency bands and bandwidths are homogeneous for the whole network, meaning that cell $\mathrm{C}_{k,i,j}$ will have the same f_k and BW_k $\forall i \in [3]$ and $\forall j \in [B]$.

Each cell type is able to provide a total of $M_k,\ k \in [K]$ physical resources blocks (PRBs) to the users of the cell. Let $U^t_{k,i,j}$ denote the number of users scheduled in $\mathrm{C}_{k,i,j}$ at time t. We denote by $N^t_{k,i,j}$ the number of PRBs used at time t, in cell $\mathrm{C}_{k,i,j}$ and can be computed as:

$$N^t_{k,i,j} = \sum_{u=0}^{U^t_{k,i,j}} n^t_{k,i,j,u} \tag{1}$$

[1] For positive integer K, $[K]$ denotes the set $\{1, 2, \ldots, K\}$

where $n^t_{k,i,j,u}$ is the number of assigned PRBs of the u^{th} scheduled user in $\mathrm{C}_{k,i,j}$. Following the cell load at time t is defined as follows:

$$l^t_{k,i,j} = \frac{N^t_{k,i,j}}{M_k} \tag{2}$$

Furthermore, we consider a radio access network with the 3GPP remote unit (RU)-distributed unit (DU) functional split; so that each cell is powered by an underlying RU-DU association. In particular, we consider that each cell is powered by a unique RU and a single DU. The power consumption of a cell $C^t_{k,i,j}$ given load $l^t_{k,i,j}$ can be expressed as follows:

$$P^t_{k,i,j} = \left(P^0_k + P^r_k l^t_{k,i,j}\right) m_{k,i,j} + P^{\emptyset}_k \left(1 - m_{k,i,j}\right), \tag{3}$$

where P^0_k is the offset of power consumption for cell type $k \in [K]$ and P^r_k is the variable power type, which scales linearly with the current load of the cell. The variable $m_{k,i,j} \in \{0,1\}$ is a binary variable and it expresses the current mode of the cell, being 0 the low-power mode and 1 active mode. Finally, $P^{\emptyset}_k$ denotes the power being consumed by the cell in low-power mode.

The signal to interference plus noise ratio (SINR) $\mathrm{SINR}^t_{k,i,j,u}$ of the uth scheduled user in $\mathrm{C}_{k,i,j}$ at time t is presented as

$$\mathrm{SINR}^t_{k,i,j,u} = \frac{P_{\mathrm{TX}} g^t_{k,i,j,u} \frac{n^t_{k,i,j,u}}{M_k}}{n^t_{k,i,j,u} N_0 + P_{\mathrm{TX}} \sum\limits_{\substack{p\in[3]\\ p\neq i}} \sum\limits_{\substack{q\in[B]\\ q\neq j}} g^t_{k,p,q,u} \frac{n^t_{k,i,j,u}}{M_k} l^t_{k,p,q}} \tag{4}$$

where P_{TX} is the transmission power of a cell and $g^t_{k,p,q,u}$ is the channel fading between $\mathrm{C}_{k,p,q}$ and the uth scheduled user in $\mathrm{C}_{k,i,j}$. The first term of the denominator corresponds to additive white Gaussian noise, while the second accounts for the inter-cell interference, assuming worst-case scenario, i.e., that all active PRBs collide. The throughput $D_{k,i,j}$ of $\mathrm{C}_{k,i,j}$ can then be expressed as

$$D_{k,i,j} = \sum_{u}^{U^t_{k,i,j}} \min\left(n_{k,i,j,u} \log_2(1 + \mathrm{SINR}^t_{k,i,j,u}), \mathrm{V}^t_{k,i,j,u}\right) \tag{5}$$

where $\mathrm{V}^t_{k,i,j,u}$ is the buffered data size of the uth user in $\mathrm{C}_{k,i,j}$ at time t. As a result, it can be observed that users with non-full buffer traffic have been considered in cell throughput calculation.

Similarly we define the user u throughput of cell $C^t_{k,j,i}$ as:

$$d_{k,i,j,u} = \min\left(n_{k,i,j,u} \log_2(1 + \mathrm{SINR}^t_{k,i,j,u}), \mathrm{V}^t_{k,i,j,u}\right). \tag{6}$$

Finally, the UEs handover procedure is based on reference signal received power (RSRP), and the number of handovers (H^t_j) is expressed as:

$$H^t_j = \sum_{k=0}^{K} \sum_{i=0}^{3} \frac{|U^t_{k,i,j} - U^{t-1}_{k,i,j}|}{2}. \tag{7}$$

We assume an artificial intelligent (AI) module built on top of each eNB, such that the agent is responsible for controlling the energy saving features of all the K cells in each of the three sectors of that eNB. Furthermore, to avoid constant on/off switching, we limit the agent action to be taken every T seconds. Meaning the agent can only change the thresholds that control the cell modes every T seconds. Furthermore, to reduce the constant sampling of network resources, which is time and resource (energy) consuming, we limit the metric gathering solution to take snapshots of the network state every T/L seconds, where L is a design parameter.

User traffic is modeled using a file transfer protocol (FTP) traffic model proposed in 3GPPP TR.36814, where each user receives a fixed size file with exponentially distributed reading time, where the mean of the reading time is time variant such that the traffic pattern is aligned with the one in [18].

III. Problem Formulation

Although the problem could be formulated as a multi-agent reinforcement learning (MARL) problem, to reduce complexity, in this work, each agent in MARL is treated independently as in standard single-agent RL problem, while the other agents are seen as part of an evolving and dynamic environment. To avoid the problem that these local environments are non-stationary and non-Markovian as stated in [20], we use the techniques proposed in [21], where we adjust the frequency of the learning process, i.e., one agent is trained for several episodes while the other agents behaviour is frozen during that time-frame. To this end, we formulate the problem addressed in this work as an single-agent MDP.

At each decision point T, the agent placed at the eNB observes a state $s^{(t)} \in \mathcal{S}$, where $\mathcal{S}$ is the state space, and selects and action $a^{(t)} \in \mathcal{A}\left(s^{(t)}\right)$, where $\mathcal{A}\left(s^{(t)}\right)$ is the set of all possible actions in state $s^{(t)}$. Set $A = \cup_{s^{(t)}\in\mathcal{S}} \mathcal{A}\left(s^{(t)}\right)$ is referred as the action space. Action $a^{(t)}$ in state $s^{(t)}$ incurs a certain reward $R\left(s^{(t)}, a^{(t)}\right)$, where $R : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ denotes the reward function, and the agent transitions to a new state $s^{(t+1)} \in \mathcal{S}$ with probability $p\left(s^{(t+1)} \mid s^{(t)}, a^{(t)}\right) \in \mathcal{P}$, where $\mathcal{P} : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to [0,1]$ is a probability kernel. At each interaction, the agent maps the observed state $s^{(t)}$ to a probability distribution over the action set $\mathcal{A}\left(s^{(t)}\right)$. This Markov decision process (MDP) is thus characterized by the 4-tuple $\langle s, a, r, p\left(s' \mid s, a\right)\rangle$. The *policy* of the agent, denoted by π, specifies the probability of selecting action $a^{(t)} = a$ in state $s^{(t)} = s$, and is given by $\pi\left(a \mid s\right)$.

Following [22], the *state-value function* $V_\pi\left(s\right)$is defined as the expected discounted reward the agent would accumulate starting at state s following policy π:

$$V_\pi\left(s\right) \doteq \mathbb{E}_\pi\left[\sum_{t=1}^{\infty} \gamma^{(t-1)} R\left(s^{(t)}, \pi\left(s^{(t)}\right)\right) \mid s^{(1)} = s\right],$$

where $0 \leq \gamma \leq 1$ is the discount factor that determines how far into the future the agent "looks", i.e., $\gamma = 0$ corresponds to a "myopic" agent, that focus only on its immediate reward, while $\gamma = 1$ represents an agent concerned with the reward over the whole time horizon. The action-value function, also referred as Q-function, is defined as:

$$Q_{\pi}\left(s,a\right) \doteq \mathbb{E}_{\pi}\left[\sum_{k=0}^{\infty}\gamma^{k}R\left(s^{(t+k)},\pi\left(s^{(t+k)}\right)\right) \mid s,a\right].$$

We define the optimal value function, $V^{*}(s)$, as the maximum expected total discounted reward obtained starting in state s and following the optimal policy:

$$V^{*}(s) = \max_{\pi}\mathbb{E}_{\pi}\left[V_{\pi}\left(s\right)\right]. \tag{8}$$

The goal is to find a policy $\pi*$ whose value function is the same as the optimal value function $V_{\pi*}\left(s\right) = V^{*}$.

Next, we define the state, action spaces and reward function for the problem addressed in this work.

1) State Space: At each decision epoch T, the agent is given a snapshot of distinct metrics so that it can understand the current network situation. The state-space of the cell managing problem that this work address is then the set of all possible values these network metrics take. The metrics gathered by the agent are the following:

- Hour of the day and day of the week, given that traffic demand has a strong time-dependent correlation.
- Cell mode of the K cells, captures the current modes of the different cells of the eNB sector, so that the agent knows which cells are on and which cells are off.
- Sector one-hot encoding: This parameter encodes the 3 sectors of the eNB, so that the agent knows to which sector the actions produced will be applied to. The fact that the agent produces an action per sector, limits the total number of actions to $2K$.
- K Cells' load. This parameter registers the average load ($l_{k,i,j}^{t}$) of all the cells in the 3 sectors of the eNB, plus the average load over the past decision times $-T, -2T, -3T, -4T$ in these sectors. This information is useful to understand the evolution of the network load.

2) Action State Space: Depending on the variation of loads experienced by the different cells of the eNB the agent can modify a pair of thresholds that are used to change the different cell modes. These thresholds are:

- Deactivation Threshold ($\mathrm{th}_{k,i,j}^{d}$): This threshold sets the value at which the average load over the last decision period T must be below for a cell to be turned off:

$$m_{k,i,j} = \begin{cases} 0 \text{ if } \sum_{t=0}^{L}\frac{l_{k,i,j}^{t}}{L} < \mathrm{th}_{k,i,j}^{d} \\ 1 \text{ otherwise} \end{cases} \tag{9}$$

- Activation Threshold ($\mathrm{th}_{k,i,j}^{a}$): The activation threshold allows the agent to wake-up inter-sector coexisting cells, that are in sleeping mode.

$$m_{k',i,j} = \begin{cases} 1 \text{ if } \sum_{t=0}^{L}\frac{l_{k,i,j}^{t}}{L} > \mathrm{th}_{k,i,j}^{a} \\ 0 \text{ otherwise} \end{cases}, \forall k' \neq k \tag{10}$$

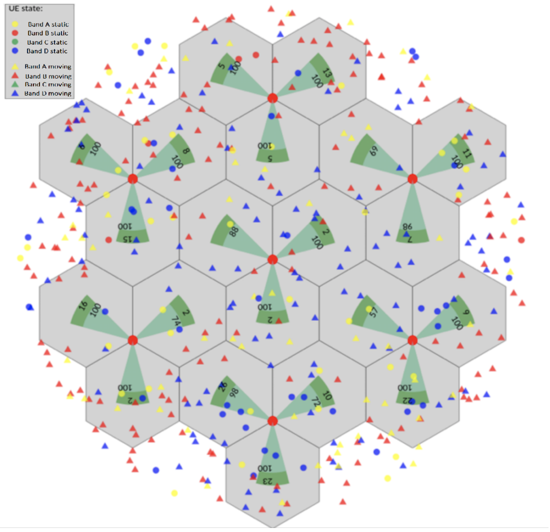


Fig. 2: Experimental setup.

3) Reward function: The reward function drives the agent behavior, and the goal of this work is to minimize the energy consumption of the network deployment while satisfying certain UEs KPIs. To this end, we can define the reward function of the eNB B_j at time t as:

$$R_{j}^{t} = \begin{cases} \sum_{k=0}^{K}\sum_{i=0}^{3}\left(e^{-\beta P_{k,i,j}^{t}} + \alpha\frac{D_{k,i,j}^{t}}{U_{k,i,j}^{t}} - \psi H_{j}^{t}\right) \\ \text{if } d_{k,i,j,u} \geq d_{\min} \forall u \in U_{k,i,j}^{t} \\ -10 \text{ otherwise} \end{cases} \tag{11}$$

where β, α and ψ are design parameters that depend on the implementation, and calibrate how important it is each of the KPIs. While $d_{\min}$ is the minimum throughput that must be guaranteed to the each active user.

A. Reinforcement Learning For Cell Management

Given the high dimensionality of the state space and the need for continuous action selection in our model (threshold values), the proposed solution leverages an actor-critic architecture and deep neural nets (DNNs) as policy and action-value function approximators. The actor-critic method combines the benefits of both value-based methods and policy optimization. While the critic estimates the action-value function $Q_{\phi}(s,a)$, the actor derives a policy $\pi_{\theta}(s)$ using the action-value estimates provided by the critic to update the policy. In this section we present our DRL solution for cell management. For ease of notation in the rest of the section we use $s^{(t)} = s$, $s^{(t+1)} = s'$, $a^{(t)} = a$, $R\left(s^{(t)}, a^{(t)}\right) = r$. Following the results of [19], we implement two critics so that we obtain two distinct estimates of the action-values; thus, two different DNNs, parameterized by $\boldsymbol{\phi}_1$ and $\boldsymbol{\phi}_2$, are used. The goal of the two critic is to reduce overestimation and better policies are obtained by limiting the critics' updates to the minimum between the two estimates.

Another DNN, parameterized by $\boldsymbol{\theta}$, is used for the policy parameterization of the actor. The goal of the actor network is to select the continuous action a based on the current state s. A deterministic policy is used for action selection, which we denote by μ, i.e., parameters $\boldsymbol{\theta}$ map state and action s to parameters $\mu_\theta(s) = a$. Finally, three more DNNs are employed, corresponding to the target networks, and are parametrized by $\boldsymbol{\phi}_1^-, \boldsymbol{\phi}_2^-, \boldsymbol{\theta}^-$, respectively, whose function will be explained later.

Based on network state s and the action a, the critics estimate the value function $Q_{\phi_i}\left(s, a\right)$, $i = 1, 2$. As is typical in value-based solution, we use off-policy temporal difference of 0, i.e., TD(0), using the minimum of both estimates to update all critic network, which equals to the minimization of the following loss function, for $i = 1, 2$:

$$L_{Q_{\phi_i}}\left(s, a\right) = \frac{1}{2}\left(r + \gamma \min_{i=1,2}\left\{Q_{\phi_i^-}\left(s', \mu_{\theta^-}\left(s'\right),\right)\right\}\right. \qquad (12)$$
$$\left. - Q_{\phi_i}\left(s, a\right)\right)^2 .$$

This loss function is minimized through gradient descent.

The critics' estimations of the action-value function are gathered by the actor and used to update the policy. In continuous action space, we use the critics' network's gradient that indicates the direction the global Q-value estimate increases, to update the policy parameters. To obtain the gradients, we perform back-propagation through one of the critics networks (for example critic 1). The action network θ is updated as

$$\theta^{t+1} = \theta^t + \alpha \mathbb{E}_{s \sim \rho_\theta}\left[\nabla_\theta \mu_\theta\left(s\right) \nabla_a Q_{\phi_1}\left(s, a\right)\big|_{a=\mu_\theta(s)}\right], \qquad (13)$$

where $s \sim \rho_\theta$ refers to the trajectory sample using network i.

Target networks are used to stabilize the updates and reduce variance. The target networks are updated as

$$\begin{cases} \phi_i^- = \tau \phi_i^t + (1 - \tau)\phi_i^- \\ \theta^- = \tau \theta_i^t + (1 - \tau)\theta^- \end{cases}, \qquad (14)$$

where $\tau \leq 1$ is an hyper-parameter that regulates the update frequency.

The memory buffer stores the interactions of the agents with the environment, to be more precise, we store one-step trajectories, i.e., s, a, r, s'. Once is filled with enough samples, mini-batches of $\mathcal{B}$ samples are obtained by sampling uniformly accross the memory buffer. These batches are then used to compute the losses and update the actor and both critics.

Ensuring the exploration of all possible continuous actions is not possible, and thus, we use the approach proposed in [19], where a zero-mean Gaussian noise is constantly added to the actor (see Eqn. (15)), based in the assumption that similar action should entail similar rewards. After the addition of noise the parameter values are clipped to the allowed range $[a_{\min}, a_{\max}]$:

$$\mu'\left(s\right) = \text{clip}\left(\mu_\theta\left(s\right) + w, a_{\min}, a_{\max}\right), \qquad (15)$$
$$w \sim \text{clip}\left(\mathcal{N}\left(0, \sigma^2\right), -c, c\right),$$

where σ and c are hyperparameters.

The DNN architectures for the action, and critic networks are the same. For all the networks, the inputs are processed by three fully connected layers consisting of 128, 64 and 32 units, respectively. Each fully connected layer is followed by a rectified linear unit (ReLU) activation function. The weights of the fully connected layers are initialized using Xavier initialization with a standard deviation of 10^{-1} [23].

The input of the actor action network is the network state, and the last layer comprises an hyperbolic tangent activation function squeezing the outputs to the range $[0, 1]$. There are $2K$ outputs, corresponding to the two thresholds for each of the K cells of the sector. Finally the critic network gathers the state and the continuous actions, and a single output value is obtained, the estimate of $Q(s, a)$. We use ADAM optimizer for both the actor and the critic, with a learning rate of l_r.

TABLE I: Simulation settings

Simulator Parameters	**Values**
Env. Type	Hex 7
Packet size	1 MB
Running time	1 week
K	4 Cells
Carrier Bands (f_k)	A, B, C, and D
M_k	200, 100, 400 and 200
P_k^0	117.3 W, 136.83 W, 62.8 W and 62.8 W,
$P_k^\emptyset$	33.2 W, 50.1 W, 37.3 W and 37.3 W,
P_k^r	137 W, 146 W, 119.8 W and 119.8 W
d_{min}	1.5Mbps
max UE speed	7 m/s
Scheduler	Proportional fair scheduler
T	60 min
L	12

TABLE II: PAT parameters.

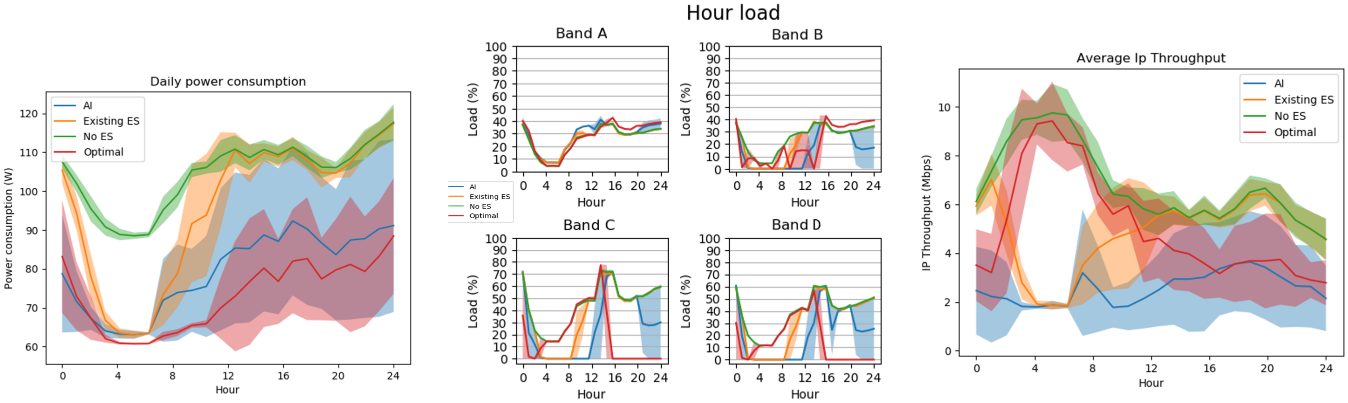
Discount factor γ = 0.99	Target update $\tau = 5 \cdot 10^{-3}$
Learning rate l_r = 10^{-3}	Policy noise σ = 0.2
$a_{\min} = 0$	$a_{\max} = 1$
$c = 0.1$	$\lvert\mathcal{B}\rvert = 128$

IV. Simulations and Numerical Results

The results presented in this work are obtained using a 5G network time-based simulator, which emulates individual multicell eNBs with a resolution of transmission time interval (TTI) of 1 ms. A static RSRP map is used for the UEs power emulation, and UEs are simulated as points undergoing either a random motion with a constant velocity drawn from a uniform distribution or as static UEs. The UEs are uniformly distributed geographically at initialization in a Hex 7 deployment ($B = 7$). The system parameters that are used to conduct the experiments reported in this section are listed in Table I, table II gathers the different hyper-parameters used to obtained the DRL solution, and Figure 2 depicts the network considered in the simulated environment, where colors correspond to different cells bands, while shapes, triangles and circles, correspond to moving and static UEs respectively.

To assess the quality of the proposed algorithm, we compare the proposed DRL solution with three other algorithms.

- No energy saving (No ES): This solution keeps all cell active during the whole duration of the simulation.
- Fixed threshold (Existing ES): This solution sets a fixed value for both the deactivation and activation thresholds.



(a) Average hour power consumption.

(b) Hourly band load.

(c) Average throughput by hour.

Fig. 3: KPIs comparison, the shaded regions demonstrate one standard deviation of the average evaluation.

TABLE III: KPIs Summary (Average per Hour).

Algorithm	Power (W)	Throughput (Mbps)	Handovers
No ES	467.08	7.56	75
Existing ES	423.359	5.43	82
AI	341.96	3.02	115
Optimal	325.12	3.56	233

The most commonly used values for these are $\mathrm{th}_{\mathrm{deac}} = 0.2$ and $\mathrm{th}_{\mathrm{act}} = 0.8$, and these are the values that are used in our experiments.

- Optimal: This algorithm knows the user demands and channel conditions beforehand and selects the resources such that the reward of Eqn. (11) is maximized at each step. Furthermore, this solution is allowed to reallocate users to different cells if that leads to higher rewards (handover is not RSRP based). This algorithm serves as an upper-bound of the maximum power-saving.

Although we have analyzed the feasibility of the AI solution in 6 distinct network settings, due to the lack of space, we provide the results for just a single network configuration. All the reported numerical results are per eNB.

In this network setting, we consider 55 UEs per ENB, where 15 of them are static and the rest dynamic. For this network configuration, 7 days of experiments have been run using 7 different network simulator seeds and the average results are presented in Table III. Figure 3(a) shows that the AI-assisted energy-saving feature presented in this work results in more power-saving comparing to the fixed threshold solution. In particular, AI saves on average 26.98% compared to "no ES" while the threshold-based method only reports 9.5% energy saving, thus, a 17.48% gap to the AI. On the other hand, the distance to the optimal solution is just 4% , as the optimal solution achieves a 30.5% power saving. To understand where this power saving comes from, we need to look at Figure 3(b), and see how AI increases the time bands C, D, and B are off compared to the fixed thresholds solutions during off-peak traffic periods. On the other hand, the optimal solution keeps those bands on, during off-peak traffic periods (night), while turning them off right after the day's peak traffic periods and redirecting the users to band A and B. This is possible given that the optimal solution is aware of the future channel conditions of the users in these bands. Thus, can improve the resource efficiency by reallocating the users to better cells, contrasting with the AI, where handover is based on an RSRP snapshot at a single point in time.

The trade-off for the energy-saving achieved by the DRL solution is the reduction in throughput. Fig 3(c) shows how the average throughput of the AI solution is reduced, nevertheless, as this happens during off-peak traffic periods the penalty paid on the reward function is low, as during these times there are fewer UEs connected. Thus, the AI solution decides to reduce the average throughput during low traffic periods as the energy-saving outweighs the throughput reduction in the reward function. It can be seen that the optimal solution also incurs a reduction in average throughput, matching the presented DRL values during peak-traffic periods, showcasing again, that energy saving comes from a cost of squeezing users into bands and thus, reducing the total number of PRBs allocated to the overall network users. Finally, aNs expected, the number of handovers is highly correlated with the number of bands switching from on to off. Based on the 3(b) we can see that the "No ES" solution executes fewer band mode changes compared to AI. Similarly, AI incurs in fewer cell mode modifications as opposed to the optimal.

V. Conclusions

In this work, we addressed the problem of reducing the energy consumption of dense wireless networks deployments using DRL, where co-located cells can be put into standby (or lower power consumption) mode dynamically, according to the change in real-time traffic conditions. The proposed DRL solution takes into account different KPIs of both the network and the UEs, and aims to reduce the energy consumed by the base stations reducing the impact on the UEs. Numerical results are presented using a practical network simulator, showing how the proposed DRL solution can save up to

26.98% of energy on a dense deployment with a small impact in UEs' KPIs. The results presented in the paper are for a concrete set of network configuration, however, it is noted that the learning algorithm is network agnostic, and thus, can be widely applicable to distinct types of networks configuration, traffic patterns, and KPIs conditions.

REFERENCES

[1] M. Inamdar and H. Kumaraswamy, "Energy efficient 5G networks: Techniques and challenges," in *IEEE ICOSEC*, pp. 1317–1322, 2020.

[2] I. Humar and et.al, "Rethinking energy efficiency models of cellular networks with embodied energy," *IEEE network*, pp. 40–49, 2011.

[3] G. Fettweis and E. Zimmermann, "ICT energy consumption-trends and challenges," in *IEEE SWPMC*, vol. 2, p. 6, 2008.

[4] S. Bhaumik and et al., "Breathe to stay cool: adjusting cell sizes to reduce energy consumption," in *ACM SIGCOMM*, pp. 41–46, 2010.

[5] R. Balasubramaniam and et al., "Cell zooming for power efficient base station operation," in *IWCMC*, pp. 556–560, 2013.

[6] E. Oh and et al., "Dynamic base station switching-on/off strategies for green cellular networks," *Trans. Wireless Coms.*, pp. 2126–2136, 2013.

[7] A. Mosavi and A. Bahmani, "Energy consumption prediction using machine learning: A review." March 2019.

[8] K. Son and et al., "Base station operation and user association mechanisms for energy-delay tradeoffs in green cellular networks," *IEEE JSAC*, pp. 1525–1536, 2011.

[9] S. Cho and W. Choi, "Energy-efficient repulsive cell activation for heterogeneous cellular networks," *IEEE JSAC*, pp. 870–882, 2013.

[10] E. Romero and et al., "A game theory based strategy for reducing energy consumption in cognitive WSN," *IJDSN*, p. 965495, 2014.

[11] M. Feng and et al, "Boost: Base station on-off switching strategy for energy efficient massive mimo hetnets," in *INFOCOM*, pp. 1–9, 2016.

[12] C. Peng and et al., "Greenbsn: Enabling energy-proportional cellular base station networks," *IEEE TMC*, pp. 2537–2551, 2014.

[13] H. Celebi and et. al, "Load-based on/off scheduling for energy-efficient delay-tolerant 5G networks," *IEEE TGCN*, pp. 955–970, 2019.

[14] 3GPP T.R. 36.927, *Potential solutions for energy saving for E-UTRAN*. V16.0.0, July, 2020.

[15] J. S. P. Roig and et al., "Management and orchestration of virtual network functions via deep reinforcement learning," *IEEE JSAC*, pp. 304–317, 2019.

[16] F. E. Salem and et al., "Reinforcement learning approach for advanced sleep modes management in 5G networks," in *IEE VTC*, pp. 1–5, 2018.

[17] A. El-Amine and et al., "Location-aware sleep strategy for energy-delay tradeoffs in 5G with RL," in *IEEE PIMRC*, pp. 1–6, 2019.

[18] S. Wu and et al., "Deep convolutional neural network assisted reinforcement learning based mobile network power saving," *IEEE Access*, pp. 93671–93681, 2020.

[19] S. Fujimoto and et al., "Addressing function approximation error in actor-critic methods," *preprint arXiv:1802.09477*, 2018.

[20] G. J. Laurent, L. Matignon, L. Fort-Piat, *et al.*, "The world of independent learners is not markovian," *International Journal of Knowledge-based and Intelligent Engineering Systems*, vol. 15, no. 1, pp. 55–64, 2011.

[21] M. Kaisers and K. Tuyls, "Frequency adjusted multi-agent q-learning," in *Proceedings of the 9th International Conference on Autonomous Agents and Multiagent Systems: volume 1-Volume 1*, pp. 309–316, 2010.

[22] R. Sutton, , and A. G. Barto, *Introduction to reinforcement learning*, vol. 135. MIT press Cambridge, 1998.

[23] X. Glorot and Y. Bengio, "Understanding the difficulty of training deep feedforward neural networks," in *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, pp. 249–256, 2010.