Self-Optimization of Cellular Networks Using Deep Reinforcement Learning with Hybrid Action Space

Mariam Aboelwafa*, Ghada Alsuhli*, Karim Banawan†, and Karim G. Seddik*
*The American University in Cairo, †Alexandria University

***Abstract*—Wireless networks have been going through tremendous proliferation recently. As a result, a continuous configuration and management are necessary to sustain a balanced performance while facing such continued growth and endless changes. A self-managed network is required to replace manual management, which is costly, time-consuming, and error-prone. In this paper, we propose a machine-learning-based cellular network management system. The proposed system aims to enhance the network stability and adaptability to temporal changes (e.g., load imbalances across cells). The presented approach is a deep reinforcement learning scheme that enables a network manager to learn a policy that maximizes the network average sum throughput while trying to minimize the consumed energy and the number of blocked users. In addition to controlling the transmitted power and the cell individual offset, MIMO can be switched ON and OFF to control the consumed energy without affecting the quality of service. This results in a *hybrid* action space, i.e., our action vector has some binary actions as well as continuous actions. We present a novel algorithm to deal with this hybrid action space. Our results reveal that our proposed algorithm is flexible, efficient, and reliable. We report significant performance gains compared to some baselines (without self-management) and previously proposed algorithms.**

***Index Terms*—Reinforcement learning, mobile networks, MIMO, Machine Learning**

I. Introduction

Cellular networks have been facing a lot of rapid changes lately [1]. For instance, the number of users drastically fluctuates over time according to human activities. The coverage requirement varies as well due to fast urban changes. Moreover, energy conservation has become a universal need in the past decade [2], [3]. Cellular networks need to be adaptive enough, i.e., self-organized, to cope with the rapid ongoing changes and meet the new requirements. A crucial criterion for the network's success is to ensure that these rapid network changes do not cause disturbance in the network. In other words, a network *equilibrium* is a must-meet requirement. This gives network immunity against the surrounding environment changes and allows it to self-heal from unexpected problems, such as the non-uniformly distributed loads between the cells.

Self-optimization of the cellular network is usually achieved by controlling the cell individual offset (CIO) values between the cells and/or the transmission power of the base stations. Both approaches can be used to control the cell boundaries. Most of the existing works focus on balancing the load of the cellular network by adjusting the CIOs between the cells, e.g., [4]–[6]. The CIO affects the handover decision of the cell-edge users. This may enhance the throughput of the users in congested cells by offloading the edge users to less-congested cells. Furthermore, the transmission power tuning of the base stations has been found effective in balancing the load of the network in [7]–[9]. Increasing the power level of a base station may be justified by seeking an enhanced channel quality of the non-edge users. Nevertheless, a strict adjustment of the network-wide transmission power is necessary to ensure that increasing the power level does not increase the interference faced by edge users. Thus, joint optimization of the CIO and transmission power has proved its superiority in enhancing the performance of the network and the Quality of Experience (QoE) of the users in [10], [11], compared to separate optimization of each parameter. Another important optimization aspect of cellular networks is minimizing energy consumption. MIMO is one of the most energy-consuming features in cellular networks. Turning MIMO off can save considerable amounts of energy. Nevertheless, consistent monitoring of the users' QoE is necessary to ensure that QoE is not negatively affected by turning off MIMO. In [12], a neural network-based algorithm is proposed to learn the behavior of SISO networks and emulate their performance to decide whether or they are sufficient to reach a satisfactory user's QoE.

In this paper, the main target is to reach a *stable state* of the cellular network after any sudden variation in the surrounding environment. We measure this stable state by meeting five main targets (which can be, in many situations, conflicting): maximizing the sum throughput of the network, achieving load balancing among cells, minimizing the number of blocked users, satisfying QoE for users, and minimizing the energy consumption. We reach this stable state by jointly optimizing three parameters/features. The first parameter is the relative CIO between base stations (eNodeBs or eNBs in 3GPP standards), which affects load balancing and user blocking. Second, we control the eNB transmission power. This parameter has a direct effect on all of the requirements mentioned above. The third feature to be controlled is enabling the MIMO capability of the eNBs. This control affects sum throughput, QoE, and energy consumption. To the best of our knowledge, this is the first work to simultaneously address these three network management controls. Due to the huge dimensionality of the aforementioned problem, it is challenging to devise a good analytical model to capture system's dynamics and/or tackling the problem using classical optimization approaches. This motivates the adoption of Machine Learning (ML) techniques in this work as we will show next.

To control the mentioned parameters, we use a Reinforcement Learning (RL) based approach [13]. The RL has been effectively used to optimize the performance of the cellular network in many previous works, for example, see [5], [6], [14], [15]. In RL, an agent aims at maximizing the long-term average reward. Thus, RL depends mainly on trial and error, in particular, at early exploration phases. This is challenging to apply in a live cellular network since it directly impacts the QoE of the end-user. To tackle this issue, we use a modified version of the simulated cellular network[1] in [11] using NS3 simulator, which is a powerful tool to simulate a network with the capability of extracting network key performance indicators (KPIs). One other major issue with applying RL for the aforementioned self-optimization of cellular networks is the fact that we are dealing with a mix of continuous and discrete control parameters. This is called a *hybrid action space*. This learning problem is challenging on its own aside from cellular network management. To that end, We propose a new RL architecture that allows the agent to take its decision in two successive stages. This is lacking in the RL literature to the best of our knowledge.

The main contribution of this paper can be summarized as:

- Simultaneous control over relative CIOs, eNBs' transmission power, and MIMO scheme using an RL-based framework. The framework automates network management and trains the cellular network to reach a stable state to maximize the sum throughput. To achieve this, joint optimization of the three parameters is formulated as a Markov Decision Process (MDP).
- Present a novel layered approach to handle hybrid action spaces. We give a detailed algorithm to enable the agent to take decisions extracted from discrete and continuous action spaces and apply them to the environment.

II. Technical Background: Reinforcement Learning

Reinforcement learning (RL) is a process in which an agent learns to make decisions (apply actions), observes the impact of its decisions on the surrounding environment, and adjusts its strategy, based on its observation, to achieve a certain goal in the long run (maximize a certain specified reward) [16]. RL can be modeled as a Markov decision process (MDP) in which, at each time step t, the agent receives some representation of the environment (a state, $S(t)$, that belongs to a state space, $\boldsymbol{S}$). Depending on this state, the agent selects an action, ($A(t)$ from a predetermined set of actions, $\boldsymbol{A}$). Next, the agent receives the consequence of its action. That is: a numerical reward $r(t+1)$ and a new state $S(t+1)$ [13].

We can say that the objective of the decision-maker (agent) is to reach the sequence of actions (policy) that maximizes the expected reward function eventually [10]. This *objective* can be characterized as:

$$\max_{\pi} \lim_{L\to\infty} \sum_{t=0}^{L} E[\lambda^t r(t)], \tag{1}$$

[1]Note that, if the simulated environment is similar to to the actual cellular network, the learned agent can be employed in the live cellular network after convergence with slight negative effects on the cellular network.

where $r(t)$ is the reward function, λ is the discount factor that determines the significance of the reward's future expected values, and π is the policy to be learned.

The nature of the action space can be discrete, continuous, or hybrid. In the discrete action space, the actions are selected from a finite set of actions. The actions in continuous action space are selected from a bounded interval. Whereas, in the hybrid action space, some actions are taken from a finite set while the others are taken from a bounded interval.

To solve the aforementioned MDP using RL, there are several variants of RL techniques [13]. There is the basic Q-learning technique that works with a discrete set of actions. There is also the Deep Q-Network (DQN) technique in which a deep neural network can be used to approximate the Q-values. The Double-DQN (DDQN) technique is an extension of DQN in which two different neural networks are implemented for action selection and action evaluation [17]. More techniques exist in literature as well. For a continuous set of actions, various RL techniques can be used. One way is to use Soft-Actor-Critic methods which are explained in detail in [18]. Another way is to use Policy Gradient methods (and their extensions like TD3) which can be understood from [19]. For hybrid action spaces, there are some parameterized approaches in the literature to handle this mixed type of action spaces [20]–[22]. In this work, we present a novel approach to handle hybrid action spaces in this paper.

III. Problem Description and Proposed Approach

A. System Model

We consider an LTE cellular network that consists of N eNBs and U User Equipment (UEs).

1) eNodeBs: Each eNB sends its transmission in the Down-Link (DL) with a power level $P_n \in [P_{\min}, P_{\max}]$ dBm. At time $t = 0, 1, 2, 3, \cdots$, each UE measures the Signal-to-Interference-plus-Noise-Ratio (SINR) of near eNBs and attaches to the cell that results in the highest one.

An eNB can be over-utilized or under-utilized. This is determined according to the value of the eNB utilization ρ_n:

$$\rho_n = \frac{\sum_{i=1}^{U_n} K_{i,n}}{B_n / B_{\text{PRB}}}, \tag{2}$$

where, U_n is the number of UEs served by the nth eNB, $K_{i,n}$ is the number of Physical Resource Blocks (PRBs) that serves the ith user in the nth eNB, B_n is the bandwidth of the nth eNB and B_{PRB} is the bandwidth of one PRB (=180 KHz in LTE). Note that ρ_n is the ratio of the total number of required PRBs of the nth eNB (to serve the attached users) to the maximum number of PRBs that it can offer. Therefore, $\rho_n < 1$ means that the eNB is underutilized while $\rho_n > 1$ means that the eNB is overutilized. Underutilization allows the eNB to serve all its attached users with satisfactory rates while this does not happen in the case of overutilization.

Every eNB can have the MIMO feature turned on or off (depending on the network manager's decision). Turning the MIMO feature on has a considerable effect on the rate received at the receiving end, thus, a better QoE can be achieved at the

UE. However, MIMO is one of the most energy-consuming features in the eNB. When the MIMO feature is turned on, a *scheduler* decides whether to use the multiple antennas to apply Spatial Multiplexing (SMux) or Transmit Diversity (TxD) transmission modes, depending on the channel quality of the UE. Nodes with high channel quality are assigned the SMux transmission mode to achieve higher data rates. On the other hand, nodes with lower channel quality (far nodes) are assigned the TxD transmission mode to improve their received SINR and combat the effect of channel fading.

2) UEs: The uth UE has random motion. It regularly searches for a better cell (according to the higher SINR) and attaches to the better cell if found. Moreover, The channel quality indicator (CQI) ϕ_u of the uth UE is reported to the associated cell periodically. The CQI is a discrete measure that represents the quality of the channel. According to the standard, $\phi_u \in \{0, 1, \cdots, 15\}$. When $\phi_u = 0$, this means that the uth UE is out of coverage (blocked). A higher CQI value corresponds to higher channel quality [23], [24].

When a certain UE is attached to cell i, it might require handover to another neighboring cell j if [23]:

$$\text{RSRP}_j + \theta_{j-i} > \text{Hys} + \text{RSRP}_i + \theta_{i-j}, \tag{3}$$

where RSRP_i and RSRP_j are the measured Reference Signal Received Power from eNBs i and j, respectively. θ_{i-j} is the CIO value of eNB i with respect to eNB j and θ_{j-i} is the CIO value of eNB j with respect to eNB i. Hys is a hysteresis value to minimize repeated handover requests that might occur due to minor signal quality fluctuations.

B. Reinforcement Learning Framework

The mapping of our joint optimization problem to the RL algorithm can be explained in brief as follows: The **agent** is a central network manager. The **environment** is the cellular network under consideration. The **state** is selected to be a subset of the network KPIs that are readily available to the network operator in practice. These KPIs are:

- Resource Block Utilization (RBU) $(B(t))$: The fraction of used PRB blocks that serve the users of each cell. It is an N-length vector. It is a representation of how congested each cell is.
- Total DL throughput of each cell $(R(t))$: It is a representation of the eNB performance. It can be expressed as $R_n(t) = \sum_{u_n=1}^{U_n} R_{u_n}(t)$, where $R_{u_n}(t)$ is the measured throughput of user u_n in the nth eNB.
- Number of active users in each cell $(C(t))$: This measures the number of users that are not idle in a certain time step.
- Modulation and Coding Scheme (MCS) Matrix $(M(t))$: It is a matrix that gives the fraction of users with a certain MCS. It is considered to be a representation of the quality of the channels.

The state is the concatenation of the above vectors (after reshaping $M(t)$):

$$S(t) = [B(t)^T \quad R(t)^T \quad C(t)^T \quad \text{vec}(M(t))^T]. \tag{4}$$

where $\text{vec}(\cdot)$ represents matrix vectorization process.

The **action** contains the features that the agent has control over. These features are:

- Relative CIO values between every two neighboring cells. $\theta_{ij} = -\theta_{ji} = \theta_{i-j} - \theta_{j-i}$. This action belongs to a continuous actions space $[-\theta_{\max}, \theta_{\max}]$.
- Transmission Power of each eNB P_n. It belongs to a continuous action space. The agent chooses a value from the set $[P_{n_0} - P_{\max}, P_{n_0} + P_{\max}]$ for some constant value P_{n_0}.
- Turning MIMO feature on/off for nth eNB m_n. The whole MIMO action vector is $[m_1 \quad m_2 \quad \cdots \quad m_N]^T$. This action is selected from a discrete set of size 2^N binary vectors, since each eNB has a decision of $m_n = 0$ (MIMO off) or $m_n = 1$ (MIMO on).

The proposed **reward** function in this work is[2]:

$$r(t) = \sum_{n=1}^{N} R(t) - \eta \bar{R}(t) \sum_{u=1}^{U} \mathbf{1}(\phi_u = 0) - \mu \sum_{n=1}^{N} m_n, \tag{5}$$

where η and μ are hyper-parameters that are selected to meet the operator's requirements, and $\bar{R}(t)$ is the average user throughput at time instant t. This reward function consists of a linear combination of three terms. The first term is the total network throughput (to be maximized). The second term is the sum throughput of the out-of-coverage (blocked) users scaled by a hyper-parameter η. This term is to be minimized (a penalty); that is why it is being subtracted. This penalty is scaled by η to control how significant it is to the agent. The last term is also a penalty. It is the number of eNBs that have the MIMO feature turned on. It is also scaled by another hyper-parameter μ. It should be noticed, that the choice of the hyper-parameters is determined based on the interests of the operator depending on the main objective of the service provider, the channel conditions and the network settings.

Thus, the agent's objective is to reach the policy (sequence of action) that tries to maximize the total network throughput, minimize the number of out-of-coverage users, and minimize the consumed energy due to turning MIMO on. Note that choosing the reward function to be the total throughput with no penalties may not reflect the operator's objective. Without the blocked user penalty, the agent may choose to keep only the users with high rates and alter CIOs or reduce power levels to force edge users to handover to a poorer performance cell. That's why putting a penalty on the number of out-of-coverage users is important. The other penalty aims at limiting the consumed energy due to turning MIMO on. Without this energy penalty, the agent may opt to always turn MIMO on with no regard to the energy consumption.

C. Proposed Algorithm

The main issue in formulating the proposed cellular network optimization as an MDP is having a hybrid action space. To solve this issue, we present a scheme that adopts two existing RL algorithms in a layered fashion. The first one is the Double

[2]It should be noted that our proposed framework can readily be extended for any other reward function.

Deep Q-Network (DDQN) [17] which is used for discrete action spaces (MIMO on/off in our case). The second one is Twin Delayed Deep Deterministic Policy Gradient (TD3) [19] which is used for continuous action space (transmission power and CIOs in our case). The presented scheme is simple, yet efficient and one of its main advantages is that it requires no modification in the core of the used techniques (DDQN and TD3).

The agent takes the decision in two stages.

- **First Stage:** The agent observes the state and takes the action of MIMO on/off based on the DDQN technique [17]. The action is taken from the discrete set $\{0, 1\}$ (for each eNB). Note that the chosen action is not applied to the environment until the end of the second stage.
- **Second Stage:** The first stage action is augmented with the state being observed and then it decides on the CIO and the variation in power level actions based on the TD3 technique. They are selected from the continuous intervals $[-\theta_{\max}, \theta_{\max}]$ and $[-P_{\max}, P_{\max}]$ respectively.

After the two stages, the augmented action is given by:

$$\begin{aligned} A(t) =& [(\theta_{ij} : i \neq j,\ i, j \in \{1, \cdots, N\}, \\ & (P_n : n \in \{1, 2, \cdots, N\}), \\ & (m_n : n \in \{1, 2, \cdots, N\})] \end{aligned} \tag{6}$$

$A(t)$ is then applied to the environment. Note that as the agent explores the whole action space, the effect of the different combinations of the first stage action (MIMO on/off) and second stage action (Relative CIOs and Power Levels) is learned.

An overview of the proposed scheme can be seen in Fig. 1 and it is described in more detail in Algorithm 1.

Algorithm 1 Proposed RL framework

```
1: Determine Reward Function.
2: Reset all values.
3: repeat
4:     procedure STAGE ONE
5:         Observe State ($S(t)$).
6:         Select MIMO feature decision (DDQN) ($A_M(t)$).
7:         Create a new augmented state ($S_{aug}(t) = [S(t), A_M(t)]$).
8:     end procedure
9:     procedure STAGE TWO
10:        Observe state ($S_{aug}(t)$).
11:        Select relative CIO and power level actions (TD3) ($[A_C(t), A_P(t)]$)
12:        Apply augmented action to the network $A_{aug} = [A_C(t), A_P(t), A_M(t)]$.
13:    end procedure
14:    Calculate Reward.
15:    Calculate next state.
16: until Reward Function Converges
```

It is worth mentioning that turning on the MIMO feature for a certain eNB does not mean that SMux can be applied for all users attached to this eNB. There is a *scheduler* applied for each eNB that decides which users to apply SMux and which to use TxD depending on the channel quality of each user. NS3 has SISO as the default running scheme and it leaves absolute liberty to the user to turn MIMO modes (SMux or TxD) on or off (i.e. The scheduler does not have a role in selecting the suitable MIMO mode). The main concern is that turning SMux for users with low CQI will only make things worse. TxD is more suitable here to enhance the channel of less fortunate users. To solve this problem, we created a simple scheduler that applies SISO for all users when the agent chooses $m_n = 0$. On the other hand, when the agent chooses $m_n = 1$, the CQI of each attached user is tested and the SMux is selected for users that have $\phi_u \geq 7$, and TxD is selected otherwise. The CQI threshold is determined such that TxD is applied for users that use QPSK and SMux for users that have higher MCS.

TABLE I: Simulation Parameters

Parameter	Notation	Value
Number of eNBs	N	3
Inter-eNB distance	d_{ij}	500 m
Bandwidth	B_n	20 MHz
Basic eNB transmission power	P_{n_0}	30 dBm
Penalty on blocked users	η	$\in [0, 2]$
Penalty on applying MIMO	μ	$\in [0, 10]$
Training steps	Steps	8,000
Steps per episode		255
Step time		0.2 seconds

IV. Performance Evaluation

A. Simulation Setup:

In our simulations, we consider a cellular network consisting of 3 eNBs, placed on the vertices of an equilateral triangle, and 42 UEs. Each cell has 10 users centralized around the eNB. On the edges between every two nodes, there are 4 edge users. The inter-eNB distance is set to be 500 meters. UEs have random mobility patterns in boxes around their starting points with a constant velocity v_u. UEs are assumed to be active all the time.

The environment, which is represented by the described cellular network, is simulated using NS3 simulator (LENA module) [25]. The agent is simulated using Python. The interface between the agent and the environment is implemented using NS3gym interface [26]. This interface is responsible for applying agent actions to the environment and feeding back the reward and new environment state to the agent. After applying the agent action, the environment is simulated using the control parameters sent by the agent action. The reward is calculated using (5). Simulation parameters are as summarized in Table I.

Next, two scenarios are evaluated. The first one is a cellular network consisting of 3 identical cells. Each contains 10 users close to the eNB (center users U_c). Each edge between two cells contains 4 users (edge users U_e). The second scenario is a cellular network consisting of 3 unidentical cells. First cell has $U_c = 18$ (congested), second cell has $U_c = 9$ (medium utilization) and third cell has $U_c = 3$ (underutilized). For both scenarios, the target is to find the optimum policy to control

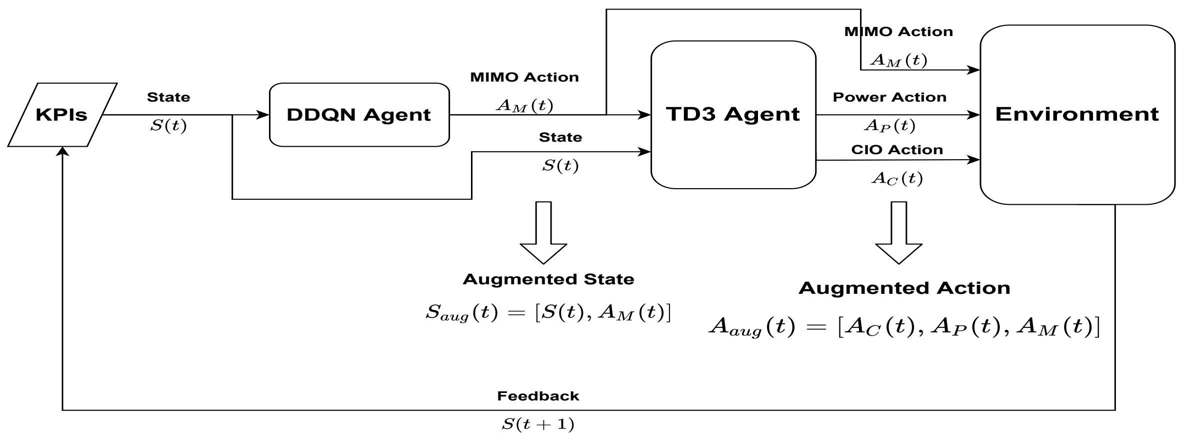

Fig. 1: An Overview of the Decision Making Process.

relative CIOs, eNBs transmission powers, and MIMO feature state such that the reward function is maximized.

B. Results of First Scenario: 3 Identical Cells

In Fig. 2, the network sum throughput (in Mbps) is plotted against the number of episodes of the training phase. We compare 2 different settings: RL agent and Baseline (BL). The RL agent is the proposed approach in this paper. It is evaluated for different values of the hyper-parameter μ which scales the MIMO energy penalty. It is also evaluated with MIMO always on and with MIMO always off. BL is the setting in which the agent has no control over any network feature. It is evaluated with MIMO always on and with MIMO always off as well. We observe that the sum throughput increases during the training phase until it converges. It is noticed that as μ increases, the agent's desire to turn MIMO on decreases (since it negatively affects the reward function). Therefore, the sum throughput for smaller μ values is higher. However, reducing μ means increasing the consumed energy as a result of turning the MIMO feature on.

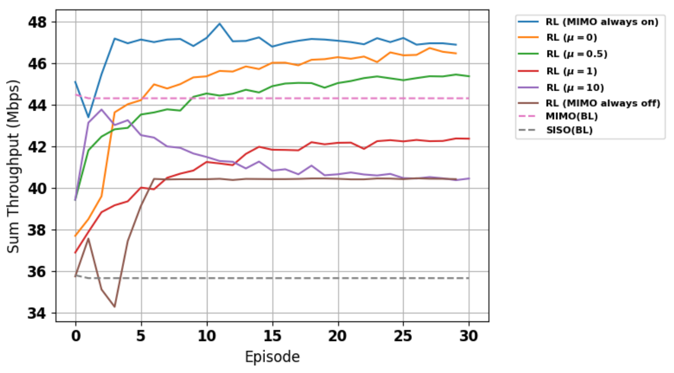

Fig. 2: Effect of MIMO Energy Penalty μ on Sum Throughput

We observe also that for all values of μ, the sum throughput is higher than the BL setting. In addition, the RL agent that has control over the MIMO feature outperforms the RL agent that has MIMO turned off by default (SISO). It also approaches a close performance to the RL agent that has MIMO turned on by default (but with less energy as we can see later). Furthermore, we can see in Fig. 2 that turning on MIMO with no other control on any network feature (BL) gives a higher sum throughput than turning MIMO off (with and without control over other features). However, our proposed scheme with $\mu = 0, 0.5$ outperforms BL with MIMO on but it consumes less MIMO energy.

It is worth mentioning that at $\mu = 10$, the sum throughput curve increases at first and decreases afterward. This is because the first portion of the learning phase is mainly for the exploration of the action space. This means that the agent is trying to apply many random actions to examine their effect on the environment. Turning MIMO on in this case ($\mu = 10$) will probably increase the sum throughput but it will also increase the penalty (which is scaled by 10). Therefore, the reward will decrease, and eventually, the agent will learn that turning MIMO off is more beneficial (in this case) to the reward (penalized sum throughput). Since we plot only part of the reward (sum throughput), the behavior shown in the curve is logical.

Fig. 3 illustrates the percentage of time the MIMO is turned on for different values of μ after convergence. Please note that these percentages are considered a measure of the percentage of consumed energy compared to turning MIMO on as a default setting.The presented values of μ include two extreme cases ($\mu = 0$) and ($\mu = 10$). The first one is having no energy penalty. The sum throughput approaches the MIMO curve and MIMO is turned on 97% of time. The second one approaches the SISO curve and MIMO is turned on 0.9% of time.

From another perspective, we report the minimum throughput and the number of unsatisfied users to make sure that the proposed scheme works in their favor as well and not only from a collective point of view. We see in Fig. 4 and Fig. 5 that our system outperforms BL scheme regarding the less fortunate users. It is worth mentioning that the unsatisfied users in Fig. 5 are determined as the users with individual throughput ≤ 1 Mbps. This threshold is determined based on the nature of the network, the scenario, the Bandwidth used, and the number of users. It might vary depending on the criterion determined by the operator. This worst-case might vary by a very considerable amount which causes these fluctuations.

To study the effect of the blocked users penalty, we varied

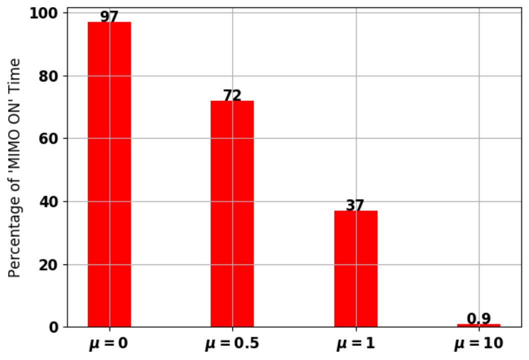

Fig. 3: Effect of MIMO Energy Penalty μ on Ratio of Time MIMO is Turned on

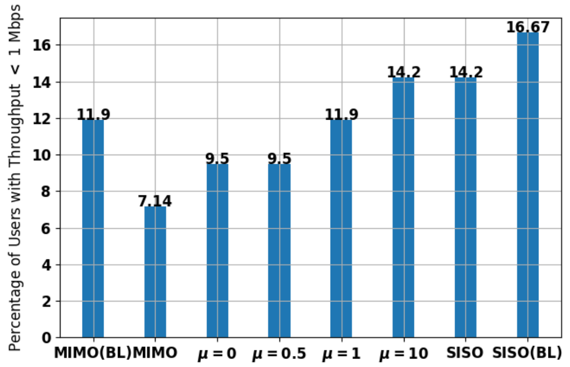

Fig. 5: Effect of MIMO Energy Penalty μ on Percentage of Unsatisfied Users

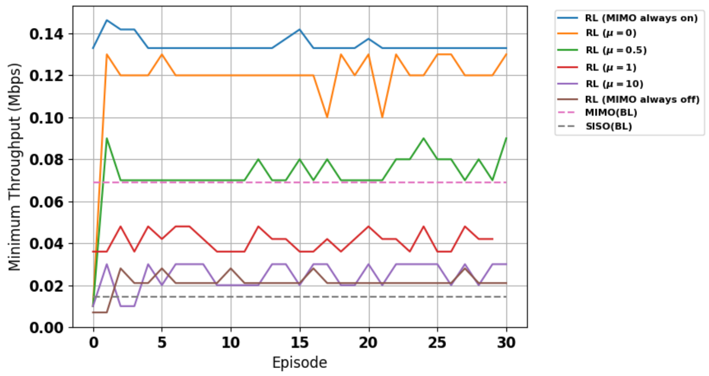

Fig. 4: Effect of MIMO Energy Penalty μ on Minimum Throughput

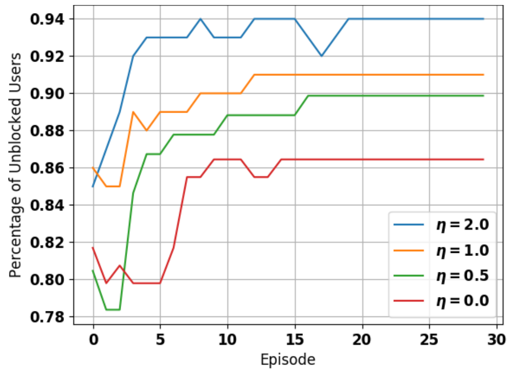

Fig. 6: Effect of Blocked Users Penalty (η) on Percentage of Unblocked Users

the hyper-parameter that scales it (η) and plotted the percentage of unblocked users in the network and the sum throughput in Fig. 6 and Fig. 7. We notice here that there is a trade-off between the sum throughput and the percentage of unblocked users. As (η) increases, the percentage of unblocked users increases but the sum throughput decreases. This is because trying to achieve a higher sum throughput might lead to block users with low CQIs to other cells (because they will consume resources and will not contribute much in the sum throughput) and give more resources to users with high CQIs (which will benefit the sum throughput of the network).

C. *Results of Second Scenario: 3 Unidentical Cells*

Here, we check the performance of the proposed algorithm in presence of the second scenario mentioned above, which consists of 3 unidentical cells. We consider this second scenario as a change in the number of users in the network due to an event for instance. The proposed algorithm performs similarly to the first scenario. Several results and investigations that were reported for the first scenario are not presented for the second scenario due to the limited space available in this paper.

In Fig. 8, the sum throughput is plotted vs the training episodes. As expected, the sum throughput increases during the training phase until it converges. We can see the effect of the MIMO energy penalty μ. In this scenario as well, the larger the penalty, the less the sum throughput (due to less desire to turn MIMO on). Moreover, we notice here also the anomaly of the curve at ($\mu = 10$) which increases at first and decreases afterward. This replicates the case of the first scenario and it is expected due to the special nature of this extreme case as explained earlier.

The results of the second scenario prove that the proposed scheme delivers the network to a stable state with a maximized sum throughput after changes that occur in the network. The proposed scheme has the advantage of reaching a *learned* policy to control the network. Learning means that the stable state can be regained again after probable occurrences.

V. Conclusion

In this paper, we present an RL framework to gain control over three main features in cellular networks, which are relative CIOs, transmitted power levels, and MIMO on/off switching. This control allows the network to reach a stable, balanced state from the perspective of load balancing, the number of covered users, QoE, and energy consumption. The advantage of this work is that it enables the network to regain this stable state and achieve a balance between different, sometimes conflicting, targets, without operator intervention, after any change in the environment. Moreover, this work creates a layered approach that enables our framework to

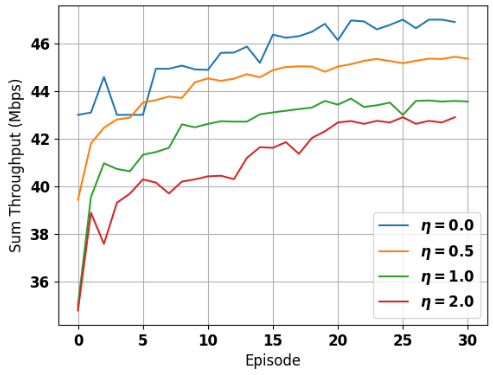

Fig. 7: Effect of Blocked Users Penalty (η) on Sum Throughput

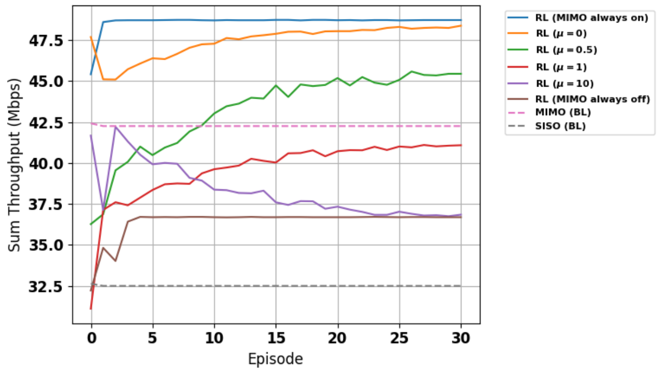

Fig. 8: Effect of MIMO Energy Penalty μ on Sum Throughput (Unidentical Cells)

handle hybrid action space in a relatively easy, yet efficient, way. The results show that the proposed scheme achieves higher sum throughput than the baseline system (where there is no RL control) while taking into consideration other important performance metrics like energy consumption and the number of blocked users. Moreover, our proposed framework can be readily extended to other network optimization targets.

References

[1] Sumita Mishra and Nidhi Mathur, "Load balancing optimization in lte/lte-a cellular networks: a review," *arXiv preprint arXiv:1412.7273*, 2014.

[2] Stefano Buzzi, I Chih-Lin, Thierry E Klein, H Vincent Poor, Chenyang Yang, and Alessio Zappone, "A survey of energy-efficient techniques for 5g networks and challenges ahead," *IEEE Journal on Selected Areas in Communications*, vol. 34, no. 4, pp. 697–709, 2016.

[3] Junaid Ahmed Khan, Hassaan Khaliq Qureshi, and Adnan Iqbal, "Energy management in wireless sensor networks: A survey," *Computers & Electrical Engineering*, vol. 41, pp. 159–176, 2015.

[4] Kareem Attiah, Karim Banawan, Ayman Gaber, Ayman Elezabi, Karim Seddik, Yasser Gadallah, and Kareem Abdullah, "Load balancing in cellular networks: A reinforcement learning approach," in *Consumer Communications & Networking Conference (CCNC)*. IEEE, 2020, pp. 1–6.

[5] G Alsuhli, Karim Banawan, Karim Attiah, Ayman Elezabi, Karim Seddik, Ayman Gaber, M Zaki, and Y Gadallah, "Mobility load management in cellular networks: A deep reinforcement learning approach," *Accepted for publication in IEEE Transactions on Mobile Computing*, 2021.

[6] Y. Xu, W. Xu, Z. Wang, J. Lin, and S. Cui, "Load balancing for ultradense networks: A deep reinforcement learning-based approach," *IEEE Internet of Things Journal*, vol. 6, no. 6, pp. 9399–9412, 2019.

[7] Sameh Musleh, Mahamod Ismail, and Rosdiadee Nordin, "Load balancing models based on reinforcement learning for self-optimized macro-femto lte-advanced heterogeneous network," *Journal of Telecommunication, Electronic and Computer Engineering (JTEC)*, vol. 9, no. 1, pp. 47–54, 2017.

[8] Heng Zhang, Xue-song Qiu, Luo-ming Meng, and Xi-dong Zhang, "Achieving distributed load balancing in self-organizing lte radio access network with autonomic network management," in *2010 IEEE Globecom Workshops*. IEEE, 2010, pp. 454–459.

[9] Anwesha Mukherjee, Debashis De, and Priti Deb, "Power consumption model of sector breathing based congestion control in mobile network," *Digital Communications and Networks*, vol. 4, no. 3, pp. 217–233, 2018.

[10] Ghada Alsuhli, Karim Banawan, Karim Seddik, and Ayman Elezabi, "Optimized power and cell individual offset for cellular load balancing via reinforcement learning," in *Wireless Communications and Networking Conference (WCNC)*. IEEE, 2021, pp. 1–7.

[11] Ghada Alsuhli, Hassan A Ismail, Kareem Alansary, Mahmoud Rumman, Mostafa Mohamed, and Karim G Seddik, "Deep reinforcement learning-based cio and energy control for lte mobility load balancing," in *Consumer Communications & Networking Conference (CCNC)*. IEEE, 2021, pp. 1–6.

[12] Mariam Aboelwafa, Mohamed Zaki, Ayman Gaber, Karim Seddik, Yasser Gadallah, and Ayman Elezabi, "Machine learning-based mimo enabling techniques for energy optimization in cellular networks," in *Consumer Communications & Networking Conference (CCNC)*. IEEE, 2020, pp. 1–6.

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

[14] Faris B Mismar, Jinseok Choi, and Brian L Evans, "A framework for automated cellular network tuning with reinforcement learning," *IEEE Transactions on Communications*, vol. 67, no. 10, pp. 7152–7167, 2019.

[15] Fan Meng, Peng Chen, Lenan Wu, and Julian Cheng, "Power allocation in multi-user cellular networks: Deep reinforcement learning approaches," *IEEE Transactions on Wireless Communications*, vol. 19, no. 10, pp. 6255–6267, 2020.

[16] Nguyen Cong Luong, Dinh Thai Hoang, Shimin Gong, Dusit Niyato, Ping Wang, Ying-Chang Liang, and Dong In Kim, "Applications of deep reinforcement learning in communications and networking: A survey," *IEEE Communications Surveys & Tutorials*, vol. 21, no. 4, pp. 3133–3174, 2019.

[17] Qi Zhang, Tao Du, and Changzheng Tian, "A sim2real method based on ddqn for training a self-driving scale car," *Mathematical Foundations of Computing*, vol. 2, no. 4, pp. 315, 2019.

[18] Tuomas Haarnoja, Aurick Zhou, Kristian Hartikainen, George Tucker, Sehoon Ha, Jie Tan, Vikash Kumar, Henry Zhu, Abhishek Gupta, Pieter Abbeel, et al., "Soft actor-critic algorithms and applications," *arXiv preprint arXiv:1812.05905*, 2018.

[19] Richard S Sutton, David A McAllester, Satinder P Singh, Yishay Mansour, et al., "Policy gradient methods for reinforcement learning with function approximation.," in *NIPs*. Citeseer, 1999, vol. 99, pp. 1057–1063.

[20] Olivier Delalleau, Maxim Peter, Eloi Alonso, and Adrien Logut, "Discrete and continuous action representation for practical rl in video games," *arXiv preprint arXiv:1912.11077*, 2019.

[21] Michael Neunert, Abbas Abdolmaleki, Markus Wulfmeier, Thomas Lampe, Tobias Springenberg, Roland Hafner, Francesco Romano, Jonas Buchli, Nicolas Heess, and Martin Riedmiller, "Continuous-discrete reinforcement learning for hybrid control in robotics," in *Conference on Robot Learning*. PMLR, 2020, pp. 735–751.

[22] Haotian Fu, Hongyao Tang, Jianye Hao, Zihan Lei, Yingfeng Chen, and Changjie Fan, "Deep multi-agent reinforcement learning with discrete-continuous hybrid action spaces," *arXiv preprint arXiv:1903.04959*, 2019.

[23] ETSI LTE, "Evolved universal terrestrial radio access (e-utra); physical layer procedures," *ETSI TS*, pp. 136–213, 2017.

[24] ETSI TSGR, "Lte: Evolved universal terrestrial radio access (e-utra)," *Multiplexing and channel coding (3GPP TS 36.212 version 10.3. 0 Release 10) ETSI TS*, vol. 136, no. 212, pp. V10, 2011.

[25] Nicola Baldo, Marco Miozzo, Manuel Requena-Esteso, and Jaume Nin-Guerrero, "An open source product-oriented lte network simulator based on ns-3," in *Proceedings of the 14th ACM international conference on Modeling, analysis and simulation of wireless and mobile systems*, 2011, pp. 293–298.

[26] Piotr Gawłowicz and Anatolij Zubow, "ns3-gym: Extending openai gym for networking research," *arXiv preprint arXiv:1810.03943*, 2018.