AoA-Based Pilot Assignment in Massive MIMO Systems Using Deep Reinforcement Learning

Yasaman Omid, Seyed MohammadReza Hosseini, Seyyed MohammadMahdi Shahabi, Mohammad Shikh-Bahaei, Arumugam Nallanathan

***Abstract*—In this paper, the problem of pilot contamination in a multi-cell massive multiple input multiple output (M-MIMO) system is addressed using deep reinforcement learning (DRL). To this end, a pilot assignment strategy is designed that adapts to the channel variations while maintaining a tolerable pilot contamination effect. Using the angle of arrival (AoA) information of the users, a cost function, portraying the reward, is presented, defining the pilot contamination effects in the system. Numerical results illustrate that the DRL-based scheme is able to track the changes in the environment, learn the near-optimal pilot assignment, and achieve a close performance to that of the optimum pilot assignment performed by exhaustive search, while maintaining a low computational complexity.**

***Index Terms*—Deep Reinforcement Learning, Pilot Assignment, Pilot Contamination, Massive MIMO.**

I. Introduction

THE ever increasing demand for wireless throughput necessitates innovative technologies to be developed. Granted the high throughput achieved by large number of antennas, massive multiple-input multiple-output (M-MIMO) is considered as a solution to these requirements. By employing time division duplexing (TDD) for transmission and reception, pilot-based techniques can be used for channel state information (CSI) acquisition in such systems. However, in case the users are assigned correlated pilot sequences, the CSI acquisition faces inevitable interference, referred to as pilot contamination which remains a challenging concern that can be minimized through pilot decontamination techniques.

In [1] a smart pilot assignment (SPA) method is presented which aims to minimize the interference caused for the users with the worst channel quality. To this end, a cost function which depends on large-scale fading coefficients of the users is minimized through a sequential algorithm. Building on this work, the authors in [2] presented low complexity pilot assignment schemes based on the large scale fading coefficients. Although the performance of the algorithms is good, consideration limited to only large scale fading coefficients, which depends on only the distance of users from the BS, may not be enough. Thus, in [3] a pilot assignment scheme was presented based on both large-scale fading coefficient and the angle of arrival (AoA) information of the users, resulting in a higher performance. The authors in [4] proposed a soft pilot reuse (SPR) method in which they showed that this method can increase spectral efficiency in many practical cases. Nonetheless, such methods require large value of overhead as a result of employing large number of pilot sequences. Recently, learning-based methods have been exploited to address the problem of pilot contamination. In [5], a deep learning (DL)-based pilot design method was presented to reduce pilot contamination for a multi-user M-MIMO system. Through unsupervised learning, a multi-layer fully connected deep neural network (DNN) was devised to solve the Mean Square Error (MSE) minimization problem online. The authors in [6] presented a DL-based scheme for joint optimization of pilot design and channel estimation in a multi-user MIMO system. They used two-layer neural networks (TNNs) for pilot design, and DNNs for channel estimation. In [7], employing adaptive moment estimation (ADAM) algorithm, an end-to-end DNN structure was presented to jointly design the pilot signals and the channel estimator. Since all the proposed DL-based assignment methods follow a blind search trajectory, a huge amount of offline data would be inevitable to cover all possible pilot assignment patterns.

Motivated by the above, in this paper, we propose an AOA-based pilot assignment scheme for multi-cell M-MIMO system using deep reinforcement learning (DRL). At first, a cost function representing the pilot contamination is designed based on the location of the users to determine their channel quality. Then, by defining proper sets of states, sets of actions and reward functions based on the channel characteristics and the resultant maximum cost functions, the agent learns the policy for pilot assignment that can adapt to channel variations while minimizing the cost function. Numerical results show that the proposed method is able to track different channel realizations and select the relevant pilot assignment that results in a close performance to that of the exhaustive search algorithm.

II. System Model and Problem Formulation

In this paper, we consider the uplink transmission of a multi-cell M-MIMO system, which consists of L cells each containing a BS with M antennas serving K single antenna users. By employing Orthogonal Frequency Division Multiplexing (OFDM), each sub-carrier channel

Yasaman Omid and Arumugam Nallanathan are with the School of Electronic Engineering and Computer Science, Queen Mary University of London, U.K. (e-mail: y.omid@qmul.ac.uk; a.nallanathan@qmul.ac.uk).

Seyyed MohammadReza Hosseini is with the Department of Electrical Engineering, K. N. Toosi University of Technology, Tehran, Iran (e-mail: s.m.hosseini@ee.kntu.ac.ir).

Seyyed MohammadMahdi Shahabi and Mohammad Shikh-Bahaei are with the Department of Engineering, King's College London, U.K. (e-mail: mahdi.shahabi@kcl.ac.uk; m.sbahaei@kcl.ac.uk).

is characterized through a flat-fading model. The location of the k-th user of the l-th cell is denoted by $\mathbf{z}_{kl}$ and the location of the jth BS is represented by $\mathbf{z}_j$. The channel coefficient between this pair of user and BS is demonstrated by $\mathbf{g}_{jkl} = \sqrt{\frac{D_{jkl}}{P}} \sum_{p=1}^{P} \mathbf{a}(\omega_{jkl}^{(p)}) \alpha_{jkl}^{(p)}$, where $D_{jkl} = c\|\mathbf{z}_{kl} - \mathbf{z}_j\|_2^{-\eta}$ is the large scale fading coefficient, η is the path loss coefficient and c is a constant depending on cell-edge SNR. Moreover, $\mathbf{a}(\omega_{jkl}^{(p)}) \in \mathbb{C}^M$ represents the antenna steering vector corresponding to $\omega_{jkl}^{(p)} \in [0, 2\pi)$, in which p is the path index from the set of P possible paths, and $\alpha_{jkl}^{(p)}$ is the stochastic phase of the p-th path. The constant c is calculated for the cell radius R as $c[\mathrm{dB}] = \gamma_{\mathrm{SNR}} + 10\eta \log_{10}(R) + 10 \log_{10}(\sigma^2)$, where $\gamma_{\mathrm{SNR}}(dB)$ denotes the cell edge SNR and σ^2 is the noise variance in the receiver. Using the uniform linear arrays (ULA) the antenna steering vectors for each antenna are modeled as $[\mathbf{a}(\omega_{jkl}^{(p)})]_m = \exp(-j2\pi m d \cos(\omega_{jkl})/\lambda)$, where d and λ represent the distance between the antennas and the signal wavelength respectively. Moreover, $[\mathbf{f}]_j$ denotes the j-th element of the vector $\mathbf{f}$. Assuming that the number of paths, P, tends to infinity, and the AoAs are independent and identically distributed (i.i.d) random variables, using the law of large numbers it can be concluded that the channel vector $\mathbf{g}_{jkl}$ is with a zero mean Gaussian distribution with the following covariance matrix $\mathbf{R}_{jkl} = \mathbb{E}\left[\mathbf{g}_{jkl}\mathbf{g}_{jkl}^H\right] = D_{jkl} \int_0^{2\pi} p(\omega_{jkl}) \mathbf{a}(\omega_{jkl}) \mathbf{a}^H(\omega_{jkl}) \, \mathrm{d}\, \omega_{jkl}$, in which $p(\omega_{jkl})$ represents the probability density function (PDF) of the variable $\omega_{jkl} \in [\omega_{jkl}^{\min}, \omega_{jkl}^{\max}]$. In this uniform distribution, the values of $\omega_{jkl}^{\min}$ and $\omega_{jkl}^{\max}$ are calculated by $\omega_{jkl}^{\min} = \omega_{jkl}^{\mu} - \omega_{jkl}^{\delta}$ and $\omega_{jkl}^{\max} = \omega_{jkl}^{\mu} + \omega_{jkl}^{\delta}$, where $\omega_{jkl}^{\mu} = \arctan\left(\frac{[\mathbf{z}_{kl}]_2 - [\mathbf{z}_j]_2}{[\mathbf{z}_{kl}]_1 - [\mathbf{z}_j]_1}\right)$ and $\omega_{jkl}^{\delta} = \arcsin\left(\frac{r_{kl}}{[\mathbf{z}_{kl}]_1 - [\mathbf{z}_j]_1}\right)$ [3]. Here, r_{kl} represents the scatter radius around the k-th user in the l-th cell. Also, $[\mathbf{f}]_n$ denotes the nth element of the vector $\mathbf{f}$. Now, we assume that a target user with an AoA within the region $I_{jkj} = [\omega_{jkj}^{\min}, \omega_{jkj}^{\max}]$ is located in the system. The goal is to assign pilots to the interfering users in the neighbouring cells in a way that the interference on the channel estimation of the target user would be minimized. The AoA of these users with respect to the target BS is within $I_{jS(l,k^{[j]})l} = [\omega_{jS(l,k^{[j]})l}^{\min}, \omega_{jS(l,k^{[j]})l}^{\max}]$. The notation $S(l, k^{[j]})$ stands for a user in the lth cell that has the same pilot as the kth user in the jth cell. In the Massive MIMO systems it is shown that the channel estimation process for the target user would be without any error if I_{jkj} and $I_{jS(l,k^{[j]})l}$ are non-overlapping [8]. Furthermore, it can be shown that the users with the steering vectors $\mathbf{a}(\omega_{jS(l,k^{[j]})l})/\sqrt{M}$, cause only a negligible channel estimation error for the target user, in case the following value is small [3]

$$\frac{\mathbf{a}^H(\omega_{jS(l,k^{[j]})l}) \mathbf{R}_{jkj} \mathbf{a}(\omega_{jS(l,k^{[j]})l})}{M} = \frac{1}{M} \int J^2(\omega_{jkj}, \omega_{jS(l,k^{[j]})l}) p(\omega_{jkj}) \, \mathrm{d}\, \omega_{jkj}. \quad (1)$$

In (1), the value of $J(\omega_{jkj}, \phi)$ is calculated as $J(\omega_{jkj}, \phi) = \sqrt{D_{jkj}} \left| \sum_{m=1}^{M} \exp(2\pi j (m-1) \frac{d}{\lambda} \times (\cos(\phi) - cos(\omega_{jkj})) \right|$. Based on this, an alternative method for minimizing (1) is presented in [3], where for pilot assignment a cost function is used which represents the interference of a user in the lth cell that has the same pilot sequence as the kth user in the jth cell. This cost function is represented by

$$G_{k^{[j]}, S(l,k^{[j]})} = G_{k^{[j]}}^{aprx}(\omega_{jkl}^{\min}) + G_{k^{[j]}}^{aprx}(\omega_{jkl}^{\max}), \quad (2)$$

in which

$$G_{kj}^{aprx}(\phi) = \sqrt{D_{jkj}} \times \begin{cases} 1, & \cos(\phi) \le \cos\left(\pi - \omega_{jkj}^{\min}\right), \\ 1 - \zeta_1(\phi), & -\cos\left(\omega_{jkj}^{\min}\right) \le \cos(\phi) \le \cos(\psi_{jkj}^{\max}), \\ \zeta_2(\phi), & \cos\left(\psi_{jkj}^{\min}\right) \le \cos(\phi) \le \cos\left(\omega_{jkj}^{\min}\right), \\ 1, & \cos(\phi) \ge \cos(\omega_{jkj}^{\min}), \\ 0, & \text{elsewhere}. \end{cases} \quad (3)$$

In (3), the values for $\zeta_1(\phi)$ and $\zeta_2(\phi)$ are calculated by $\zeta_1(\phi) = \frac{\cos(\phi) + \cos(\omega_{jkj}^{\min})}{\cos(\psi_{jkj}^{\max}) - cos(\omega_{jkj}^{\min})}$ and $\zeta_2(\phi) = \frac{\cos(\phi) - \cos(\psi_{jkj}^{\min})}{\cos(\omega_{jkj}^{\min}) - \cos(\psi_{jkj}^{\min})}$. Moreover, $\psi_{jkj}^{\min}$ and $\psi_{jkj}^{\max}$ are determined via calculating the zeros of the functions $J(\omega_{jkj}, \phi)$ and $J(\pi - \omega_{jkj}, \phi)$ as [3]

$$\psi_{jkj}^{\min} = \max\left(\min_r\left(\{\phi_{r,\omega_{jkj}^{\min}}\}_r\right), \min_r\left(\{\phi_{r,\pi-\omega_{jkj}^{\min}}\}_r\right)\right), \quad (4)$$

$$\psi_{jkj}^{\max} = \min\left(\max_r\left(\{\phi_{r,\omega_{jkj}^{\min}}\}_r\right), \max_r\left(\{\phi_{r,\pi-\omega_{jkj}^{\min}}\}_r\right)\right), \quad (5)$$

where $\{\phi_{r,\omega_{jkj}^{\min}}\}_r$ and $\{\phi_{r,\omega_{jkj}^{\min}}\}_r$ denote the zeros of $J(\omega_{jkj}, \phi)$ and $J(\pi - \omega_{jkj}, \phi)$ respectively. It is noteworthy to mention that as the number of BS antennas tends to infinity, equation (2) tends to $\max_{\omega_{jkj}} J(\omega_{jkj}, \phi)$. In the following, by using a DRL-based strategy, the cost function in (2) is applied to find a near-optimal pilot assignment strategy.

III. DRL-Based Pilot Assignment Strategy

In this section, at first, the basics of DRL is explained, then the algorithm is applied to the pilot assignment problem.

A. DRL Algorithm

Any RL system contains an agent which interacts with the environment in a series of discrete time steps, a set of states and a set of actions. By taking an action in a time step, the agent transitions from one state to the next. Also, each action results in a reward for the agent. The reward function is introduced as $R : \mathbb{S} \times \mathbb{A} \to \mathbb{R}$, with $\mathbb{S}$, $\mathbb{A}$ and $\mathbb{R}$ representing the discrete set of states, the set of actions and the set of rewards, respectively. Accordingly, the episode return is defined as $R_t = \sum_{i=1}^{T} \mu^{i-t} R(a_i, s_i)$. Here, $\mu \in [0, 1]$ stands for the discount factor of the future rewards, $a_i \in \mathbb{A}$ represents the action in the ith time step and $s_i \in \mathbb{S}$ is the state of the ith time step. By taking the action a_i the state of the agent transitions to s_{i+1} and the agent gains the reward r_{i+1}. Utilizing Q-learning [9], an efficient policy is adopted to estimate the state-action value function (or in other words Q-function) that maximizes the future reward. The value function is represented

as the expected return over all episodes, when starting from a state s and performing the action a by following the policy $\pi : \mathbb{S} \rightarrow \mathbb{A}$ as $Q^{\pi}(s,a) = \mathbb{E}[R_t|s_t = s, a_t = a, \pi]$. Using Bellman equation, [9], the optimal value function is given by $Q^{*}(s,a) = \max_{\pi} Q^{\pi}(s,a)$ and the optimal policy follows $\pi^{*}(s) = \arg\max_{a \in \mathbb{A}} Q^{*}(s,a)$. Additionally, in case of a large number of states, deep RL (DRL) is employed as a potential solution to systems suffering from infeasible generalization for unobserved states. To the best of our knowledge, DRL accounts for using neural networks to approximate the Q-function. The outgoing construction referred to as Q-neural network (QNN) leads to the following approximation for the action-value function $q(s,a,\boldsymbol{\kappa}) \approx Q^{*}(s,a)$, where $\boldsymbol{\kappa}$ refers to some parameters that define the Q-value. In a specific time step t, where the state is s_t and the QNN weights are $\boldsymbol{\kappa}$, the DRL agent takes an action with regards to $a_t = \arg_a \max q(s,a,\boldsymbol{\kappa})$ where $q(s,a,\boldsymbol{\kappa})$ is the output of QNN for every possible action a. Then, the agent receives the reward r_{t+1} and transitions to the state s_{t+1}. Thus, the experience set at the time step t would be $(s_t, a_t, r_{t+1}, s_{t+1})$ which is used in training QNN. The Q-value $q(s,a,\boldsymbol{\kappa})$ is then updated towards the target value

$$x^{trg}_{r_{t+1},s_{t+1}} = r_{t+1} + \mu \max_{a} q(s_{t+1}, a, \boldsymbol{\kappa}). \tag{6}$$

The update rule in DRL is to find the value of $\boldsymbol{\kappa}$ in QNN through a training phase, in which the square loss of $q(s,a,\boldsymbol{\kappa})$ is minimized. The square loss of $q(s,a,\boldsymbol{\kappa})$ is defined as

$$w(s_t, a_t, r_{t+1}, s_{t+1}) = (x^{trg}_{r_{t+1},s_{t+1}} - q(s_t, a_t, \boldsymbol{\kappa}))^2, \tag{7}$$

Furthermore, the values of $\boldsymbol{\kappa}$, are updated by a semi-definite gradient scheme used for minimizing (7) as

$$\boldsymbol{\kappa} \leftarrow \boldsymbol{\kappa} + \rho[x^{trg}_{r_{t+1},s_{t+1}} - q(s_t, a_t, \boldsymbol{\kappa})] \bigtriangledown q(s_t, a_t, \boldsymbol{\kappa}), \tag{8}$$

in which ρ is the learning rate step size. As the QNN weights are updated, the target value changes. In DQN, the quasi-static target network approach is used in which the target Q-network $q(.)$ in (6) is replaced by $q(s_t, a_t, \hat{\boldsymbol{\kappa}})$, and the parameter $\hat{\boldsymbol{\kappa}}$ is updated after every T time steps by $\hat{\boldsymbol{\kappa}} = \boldsymbol{\kappa}$. Moreover, the experience replay method [10] can be used for stability of the system. In this method, instead of training QNN at the end of each time step with only one experience, multiple jointed experiences can be utilized for batch training. In other words, a replay memory with a fixed capacity is considered in which the set $v = (s, a, r, s^{'})$ is saved in specific time steps. A mini-batch referred to as B, with M_{batch} random experiences is selected for a training course, and the loss-function is calculated based on them. By using the experience replay method and the quasi-static target network approach, the variable $\boldsymbol{\kappa}$ is updated by

$$\boldsymbol{\kappa} \leftarrow \boldsymbol{\kappa} + \frac{\rho}{M_{\text{batch}}} \sum_{b \in B} [x^{trg}_{r,s^{'}} - q(s, a, \boldsymbol{\kappa})] \bigtriangledown q(s, a, \boldsymbol{\kappa}), \tag{9}$$

where $x^{trg}_{r,s^{'}} = r + \mu \max_{a^{'}} q(s^{'}, a^{'}, \hat{\boldsymbol{\kappa}})$. In the following the DRL algorithm is employed for the purpose of pilot assignment.

B. Application of DRL Algorithm in Pilot Assignment

In this section, using the cost function in (2), a pilot assignment method is proposed. In this case, the cost function for the ith user in the jth cell is presented as $G_{k^{[j]}} = \sum_{\substack{l=1 \\ l \neq j}}^{L} G_{k^{[j]},S(l,k^{[j]})}$. Considering this cost function, the optimization problem for pilot assignment becomes $\min_{\boldsymbol{\pi}} \quad [\max_{k,j} \quad G_{k^{[j]}}]$. Note that, this optimization problem is based on the AoA distribution of the users, because of the definition for $G_{k^{[j]}}$. To design a DRL platform for solving this problem, first the state set, the action set and the reward set need to be defined. Hence, the state set in the time step n is defined by the following information:

- The binary pilot assignment pattern for all users.
- The pilot index and the cell index of the users with the maximum cost function in each cell, in the time step $n-1$.
- The pilot index of the user that action takes place on, in the time step $n-1$.
- The cell index of the user that action takes place on, in the time step $n-1$.
- The value of the maximum cost function in each cell in the system, in the time step n.

We assume that the pilot index and the cell index of the user with the maximum cost function are denoted by k'' and l'', respectively. Also, k and l stand for the pilot index and the cell index of the selected user by the agent, to exchange its pilot with another user in the lth cell, with the same pilot as the target user. In this case, the state set can be represented by $\mathbb{S} = \left\{ \tilde{u}_j(i), G_{[j]}, k, l, k'', l'' \;\middle|\; j = 1, \ldots, L \;,\; i = 1, \ldots, K \right\}$, where $G_{[j]}$ represents the value of the maximum cost function in the jth cell. The notation $\tilde{u}_j(i)$ stands for the binary index of a user in j-th cell which uses the i-th pilot.

In order to introduce the action set, the user with the maximum cost function is considered. For the sake of simplicity we refer to this user as the target user, and its cell as the target cell. Then, in the neighbouring cell, a random user is selected and it's pilot is switched with the pilot of the user that is assigned with the same pilot sequence as the target user. Note that, in case the selected user already has the same pilot as the target user, no action would be taken. To be more specific, when the DRL structure is moving from the nth time step to the $n+1$th time step, the action set for the target user is $\mathbb{A}_{kl} = \{a_{k',l'} | k' = 1, \ldots, K \;,\; l' = 1, \ldots, L\}$, in which $a_{k',l'}$ is defined as

$$a_{k',l'} = \begin{cases} \text{No action is taken,} & k' = k \\ \begin{cases} u^{(n+1)}_{l'}(k') = u^{(n)}_{l'}(k) \\ u^{(n+1)}_{l'}(k) = u^{(n)}_{l'}(k') \end{cases} & k' \neq k \end{cases} \tag{10}$$

where $u^{(n)}_{l}(k)$ denotes the user of the l-th cell that utilizes the k-th pilot sequence at the time step n.

The next step is to define the reward set. To do so, two thresholds are considered for the cost function as g_1 and g_2 ($g_2 > g_1$). In this case, by moving from the nth to the $n+1$th time step, the instantaneous reward is defined as $r = r^{(1)} + r^{(2)} + r^{(3)}$, where

$$r^{(1)} = \begin{cases} +1, & G^{(n+1)}_{[l'']} < g_1 \\ 0, & g_1 < G^{(n+1)}_{[l'']} < g_2 \\ -1, & G^{(n+1)}_{[l'']} > g_2, \end{cases} \tag{11}$$

$$r^{(2)} = \begin{cases} -1, & \text{Action is taken} \\ 0, & \text{Action is not taken,} \end{cases} \tag{12}$$

$$r^{(3)} = \begin{cases} +2, & G_{[l'']}^{(n+1)} < g_1,\ G_{[l'']}^{(n)} > g_2 \\ +1, & G_{[l'']}^{(n+1)} < g_1,\ g_1 < G_{[l'']}^{(n)} < g_2 \\ +1, & g_1 < G_{[l'']}^{(n+1)} < g_2,\ G_{[l'']}^{(n)} > g_2 \\ -1, & g_1 < G_{[l'']}^{(n+1)} < g_2,\ G_{[l'']}^{(n)} < g_1 \\ -1, & G_{[l'']}^{(n+1)} > g_2,\ g_1 < G_{[l'']}^{(n)} < g_2 \\ -2, & G_{[l'']}^{(n+1)} > g_2,\ G_{[l'']}^{(n)} < g_1 \\ 0, & \text{Otherwise} \end{cases} \tag{13}$$

Note that, $G_{[l'']}^{(n)}$ and $G_{[l'']}^{(n+1)}$ represent $G_{[l'']}$ before and after taking an action in the nth time step, respectively. Furthermore, $r^{(1)}$ in (11) represents the certain reward achieved in the $n+1$th time step, and $r^{(2)}$ in (12) denotes the cost of overhead on the system caused by taking an action in the nth time step. Obviously, taking any action, regardless of the certain reward it might or might not achieve, results in a negative reward due to the overhead it causes for the system. Finally, $r^{(3)}$ in (13), stands for the relative reward during the transition from the nth time step into the $n+1$th time step. Based on this definition for the reward set, it can be claimed that taking any action could result in a negative instantaneous total reward, unless the certain reward and the relative reward are adequately positive. This definition for the reward set, results in taking more targeted actions.

IV. Numerical Results

In this section, the DRL-based algorithm is simulated and analyzed. To evaluate the proposed method, the minimum rate among all users is used as a comparison benchmark, which is calculated by $\Gamma_{kj} = \log(1+\gamma_{kj,\infty})$ for the kth user in the jth cell. The SINR value in an asymptotic regime where $M \to \infty$ is denoted by $\gamma_{kj,\infty}$ which according to the random matrix theory is calculated by $\gamma_{kj,\infty} = \frac{|D_{jkj}|^2}{\sum_{l\neq j}^{L}|D_{jS(l,k^{[j]})l}|^2}$ [11]. Furthermore, the proposed schemes are compared to the exhaustive search method and the random pilot assignment, which respectively give the upper and the lower bounds of performance for the pilot assignment problem. The performance of the DRL-based scheme is also compared to that of the SPR method in [4], which necessitates using more orthogonal pilot sequences than the number of users in each cell, resulting in high system overhead. It is assumed that the system is composed of $L = 7$ cells each containing a BS with $M = 100$ antennas serving $K = 4$ single-antenna users. Also, the path loss coefficient is set to $\eta = 2.5$, the cell-edge SNR is set to $\gamma_{\text{SNR}} = 20$ dB, the noise variance in the receiver is $\sigma^2 = 0.001$, and the cell radius is $R = 1000$ m. Other parameters of the channel model are $P = 50$, $\lambda = 0.1$ m and $d = \lambda/2$. The QNN structure utilized for the deep reinforcement learning is a deep residual network (ResNet) [12] with six hidden layers. This QNN structure is depicted in Fig. 1. In this structure, each hidden layer contains 128 neurons. For the sake of simplicity, we refer to the realization of QNN by ResNet as ResNet. Furthermore, each

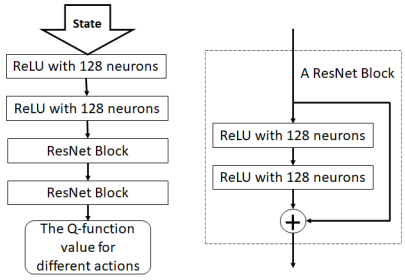

Figure 1: The structure of a QNN realised by ResNet

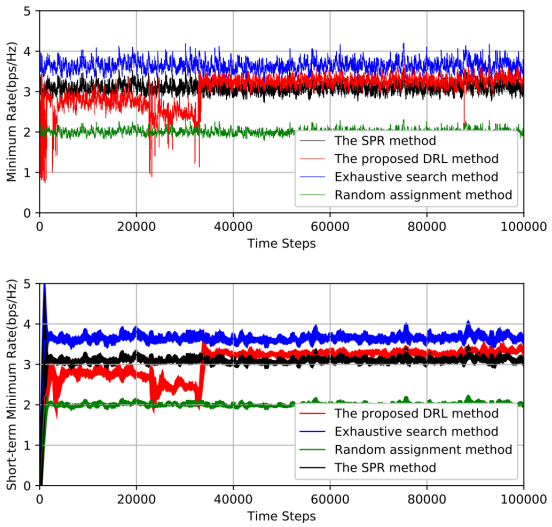

Figure 2: Minimum achievable rate versus time steps; a comparison among the proposed DRL method, the SPR method, the exhaustive search method and the random pilot assignment.

item in this ResNet structure is referred to as a ResNet block. The ReLU functions [13] in this structure, are considered as the activation functions for the neurons. The first two hidden layers of this ResNet are completely connected to each other and they are followed by two ResNet blocks. Each ResNet block contains two consecutive hidden layers plus a shortcut from the input to the output of the ResNet block. Whenever the $\boldsymbol{\kappa}$ coefficient in the QNN is updated, a mini-batch with 200 experimental samples are randomly selected from the 500 previous experiences in the experience replay reservoir, in order to calculate the loss function. Note that the experience replay reservoir is updated in a first in first out (FIFO) manner, and whenever the experience memory is full, the older experiences are removed for the new experiences to be restored. Also, in order to update $\boldsymbol{\kappa}$ by the mini-batch gradient descent method, the RMSprop [14] algorithm is used. An exponential Decay ϵ-greedy Algorithm is applied in the system, so that the DRL structure would not get stuck in a sub-optimal decision policy before learning adequate experiences. The value of ϵ at first is set to 0.5, but it is decreased gradually with every time step by the rate of 0.9975, until it reaches a threshold of 0.0001. Note that, having a positive value for the ϵ at all times results in adaptability of the decision policy to

Figure 3: The achievable reward versus time steps. The First figure demonstrates the agent's rewards throughout time, the second figure shows the short-term and the long-term rewards, and the third figure illustrates the negative reward ratio.

the future changes. Also, the discount factor μ is set to 0.9.

In Fig. 2, the performance of the DRL algorithm is depicted in terms of the minimum achievable rate versus the AoA time steps, and it is compared to the exhaustive search method, the random pilot assignment and the SPR scheme. Each time step defines the time period in which the AoA intervals and the large scale fading coefficients remain unchanged. To calculate the short-term average minimum rate in each time step, the average of the minimum rates over the last 50 time steps is calculated. As the time steps increase, the performance of the proposed method tends to the performance of the exhaustive search algorithm. Ergo, it can be concluded that by the passage of time, the proposed algorithm gains the ability to track the changes in the channels and learn the effective pilot assignment policy. Note that in the higher time steps, the gap between the performances of the proposed DRL method and the exhaustive search algorithm is small. Also, after gaining enough experiences, the proposed scheme outperforms the SPR method while the system overhead of the SPR method is much greater than that of the DRL-based scheme. To be specific, assuming that in the SPR method, the ratio of marginal users to central users is 1/3, in a cluster of 7 cells, the number of required orthogonal pilot sequences of the SPR method is 250% of the DRL-based scheme. Fig. 3 describes the performance of the DRL method in terms of the achievable reward. The first figure demonstrates the agent's rewards throughout time. It can be seen that as the time passes, the agent learn the right policy to achieve mostly positive rewards. In the second figure, to calculate the short-term achievable reward in each time step, the average of the rewards among the last 50 time steps is calculated. Furthermore, to calculate the long-term reward in each time step, the reward in all of the time steps, from the beginning until the current step are considered. As the time step increases, the positive rewards supersede the negative ones by the agent. Moreover, the short-term reward and the long-term reward are both fully positive in the higher time steps. The third figure, shows the ratio of the negative rewards, and it can be seen that as the agent gains more experiences, the ratio of the negative rewards decreases. Hence, it can be concluded that this algorithm finally reaches an effective pilot assignment.

V. Conclusion

In this paper, the problem of pilot assignment in multi-cell M-MIMO systems is tackled by the DRL scheme. By using both the distance and the AoA information of the users, a cost function is defined representing the pilot contamination effects in the system, and an optimization problem is formed to minimize this cost function. The DRL algorithm is applied to this problem, to find an effective pilot assignment strategy. Numerical results show that the performance of the DRL-based scheme is better than some methods in the literature while it maintains a lower system overhead.

References

[1] X. Zhu, Z. Wang, L. Dai, and C. Qian, "Smart Pilot Assignment for Massive MIMO," *IEEE Communications Letters*, vol. 19, no. 9, pp. 1644–1647, sep 2015.

[2] S. M. Shahabi, M. Ardebilipour, and Y. Omid, "Low-complexity fairness-driven pilot decontamination schemes for multi-cell massive MIMO systems," *Physical Communication*, vol. 36, p. 100803, oct 2019.

[3] L. S. Muppirisetty, T. Charalambous, J. Karout, G. Fodor, and H. Wymeersch, "Location-Aided Pilot Contamination Avoidance for Massive MIMO Systems," *IEEE Transactions on Wireless Communications*, vol. 17, no. 4, pp. 2662–2674, apr 2018.

[4] X. Zhu, Z. Wang, C. Qian, L. Dai, J. Chen, S. Chen, and L. Hanzo, "Soft Pilot Reuse and Multicell Block Diagonalization Precoding for Massive MIMO Systems," *IEEE Transactions on Vehicular Technology*, vol. 65, no. 5, pp. 3285–3298, may 2016.

[5] J. Xu, P. Zhu, J. Li, and X. You, "Deep Learning-Based Pilot Design for Multi-User Distributed Massive MIMO Systems," *IEEE Wireless Communications Letters*, vol. 8, no. 4, pp. 1016–1019, Aug 2019.

[6] C.-J. Chun, J.-M. Kang, and I.-M. Kim, "Deep Learning-Based Joint Pilot Design and Channel Estimation for Multiuser MIMO Channels," *IEEE Communications Letters*, vol. 23, no. 11, pp. 1999–2003, nov 2019.

[7] X. Ma and Z. Gao, "Data-Driven Deep Learning to Design Pilot and Channel Estimator for Massive MIMO," *IEEE Transactions on Vehicular Technology*, vol. 69, no. 5, pp. 5677–5682, may 2020.

[8] H. Yin, D. Gesbert, M. Filippou, and Y. Liu, "A Coordinated Approach to Channel Estimation in Large-Scale Multiple-Antenna Systems," *IEEE Journal on Selected Areas in Communications*, vol. 31, no. 2, pp. 264–273, feb 2013.

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

[10] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski, S. Petersen, C. Beattie, A. Sadik, I. Antonoglou, H. King, D. Kumaran, D. Wierstra, S. Legg, and D. Hassabis, " Human-level control through deep reinforcement learning ," *Nature*, vol. 518, no. 7540, pp. 529–533, Feb 2015.

[11] S. Ma, E. L. Xu, A. Salimi, and S. Cui, "A Novel Pilot Assignment Scheme in Massive MIMO Networks," *IEEE Wireless Communications Letters*, vol. 7, no. 2, pp. 262–265, apr 2018.

[12] K. He, X. Zhang, S. Ren, and J. Sun, "Deep Residual Learning for Image Recognition," in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, jun 2016, pp. 770–778.

[13] I. Goodfellow, Y. Bengio, and A. Courville, "Deep learning," *MIT press*, vol. 2016.

[14] T. Tieleman and G. Hinton, "Lecture 6.5-rmsprop: Divide the gradient by a running average of its recent magnitude," *COURSERA: Neural networks for machine learning*, vol. 4, no. 2, pp. 26–31, 2012.