IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Date of publication xxxx 00, 0000, date of current version xxxx 00, 0000.

Digital Object Identifier 10.1109/ACCESS.2017.DOI

A Study on Deep Learning for Latency Constraint Applications in Beyond 5G Wireless Systems

SUREN SRITHARAN[1], HARSHANA WELIGAMPOLA[1], AND HARIS GACANIN[2], (Senior Member, IEEE)

[1]University of Peradeniya, Sri Lanka. (e-mail: {suren.sri, harshana.w}@eng.pdn.ac.lk)
[2]RWTH Aachen University, Germany (e-mail: harisg@ice.rwth-aachen.de)

Corresponding author: Haris Gacanin (e-mail: harisg@ieee.org).

ABSTRACT The fifth generation (5G) of wireless communications has led to many advancements in technologies such as large and distributed antenna arrays, ultra-dense networks, software based networks and network virtualization. However, the need for a higher level of automation to establish hyper-low latency and hyper-high reliability for beyond 5G applications requires extensive research on machine learning with applications in wireless communications. Thereby, learning techniques will take a central stage in the sixth generation of wireless communications to cope up with the stringent application requirements. This paper studies the practical limitations of these learning methods in the context of resource management in non-stationary radio environment. Based on the practical limitations we carefully design and propose supervised, unsupervised, and reinforcement learning models to support rate maximization objective under user mobility. We study the effects of practical systems such as latency and reliability on the rate maximization with deep learning models. For common testing in the non-stationary environment, we present a generic dataset generation method to benchmark across different learning models versus traditional optimal resource management solutions. Our results indicate that learning models have practical challenges related to training limiting their applications. These models need an environment-specific design to reach the accuracy of an optimal algorithm. Such an approach is practically not realistic due to the high resource requirement needed for frequent retraining.

INDEX TERMS Machine learning, low-latency application, learning model ageing, mobile communications, 6G.

I. INTRODUCTION

The fifth generation (5G) of wireless communication has been a key enabler for many services including mobile broadband, mission-critical applications, and massive machine-type communications [1]. The recent development in 5G has led to advancements in techniques such as large and distributed antenna arrays, ultra-dense networks, software-based networks, and network virtualization. However, the continuously evolving need of hyper low latency (i.e. 6G aims at the control plane latency < 1 ms, while user plane latency < 0.1 ms) and higher data rates (e.g. Tbps-level) for emerging applications such as holographic teleportation [2], virtual and augmented reality [3], smart materials based programmable radio environment [4], brain-computer interfacing [5], etc. cannot be fulfilled by the existing 5G technology. Recent developments in artificial intelligence (AI) has shifted the interest towards the development of "autonomous and connected intelligence" [6], [7]. Thereby, learning techniques will take a central stage in the sixth generation (6G) communication systems to design autonomous networks with the stringent application requirements of next generation user-centric communications [8]. However, as often observed [9], the usage of machine learning techniques (i.e. supervised, unsupervised and reinforcement learning) for wireless applications poses many challenges, and better understanding of their practical limitations is of high importance.

There has been growing interest in learning through neural networks for various applications in wireless communica-

tions such as resource allocation, channel estimation, interference management, etc. [10], [11]. Deep neural network (DNN) is a supervised learning technique that requires an offline data-dependent training mechanism with its performance being highly sensitive to the amount and quality of the labeled dataset [12]. Valuable analysis in [11], clearly shows that most of the works evaluate the wireless system-level performance under the assumption that the environment is stationary, while training of a learning model is accomplished – the learner is in a steady-state with the asymptotically converged model. However, these are critical assumptions and their impact needs to be carefully studied.

A misconception of such assumptions renders the practical application of deep learning unclear for wireless services in a non-stationary environment with latency and reliability constraints (e.g. mission-critical applications such as Industry 4.0 with moving robots, ultra-dense ultra-range mobile communication networks, mmWave drone-based networks, to name just a few). As we will show in later sections, a non-stationary radio environment causes the ageing of the model with a high dependency on biased and incomplete datasets. This requires frequent retraining of the model and causes a service disruption (e.g. drop in sum-rate). Thus, the following questions motivate this study: (i) What are the system's performance limitations with practical training? (ii) What is the impact of non-stationarity on learning? (iii) What is the efficiency and efficacy of learning with the change in the complexity of the problem?

This paper presents a comprehensive study of application of deep learning models for wireless applications with stringent latency and reliability requirements, in particular, we focus on resource management in a non-stationary radio environment. We discuss the degree of influence factors such as user mobility, problem complexity, retraining time constraint, etc. have on the design of deep learning models. Unlike previous works [11], we evaluate the wireless system-level performance with respect to the tight latency/reliability learning requirement, and the computational requirements while considering both training and prediction time of learning. Our findings indicate that deep learning models need to be carefully designed to reach the accuracy of an optimal algorithm given the computational complexity constraints, while the latency due to frequent retraining remains a challenge. The contributions of this work are summarized as follows:

- We present a semi-online training methodology to eliminate service disruptions due to ageing of the learning model. We further enhance our approach with a practical design employing reinforcement learning through Deep Q-Network (DQN) model, to circumvent the frequent retraining requirements through online operation. We perform a thorough analysis of the variation of performance under different conditions utilizing state-of-the-art models.
- We study the effect of non-stationarity on the performance of deep learning models and further analyze the variation of performance of these models with respect to other factors such as the problem complexity, non-deterministic user activity, etc.
- The sum-rate maximization with deep learning in multi-carrier systems is prone to power violation. To eliminate the power violation problem, we propose a design of loss function in a non-stationary environment for the training of DNN models to regulate the power violation.
- We investigate the learning bias in the subcarrier allocation vector due to its sparsity[1] leads to poor training of the DNN model [13]. To solve this problem, we propose a method with linear a sequence of learning models referred to as the "pipeline model".
- Our study reveals that the high computational resources required by the existing implementation of deep learning models act as a barrier to employ them for hyper low latency/reliability criteria of future wireless applications (e.g. Industry 4.0). Interactive applications would require sophisticated hardware, and current available computational resources on handheld devices for learning would only suffice for non-interactive applications. Thus, novel training approaches or faster computational resources are required to reach the latency/reliability criteria of Industry 4.0 applications.

TABLE 1: Abbreviations and expansions.

Abbreviation	Expansion
AWGN	Additive white Gaussian noise
BS	Base station
CNN	Convolution neural network
CSI	Channel state information
DNN	Deep Neural Network
DQN	Deep Q-Network
LSTM	Long short-term memory
MDP	Markov decision process
MSE	Mean square error
OFDM	Orthogonal Frequency Division Multiplexing
ReLU	Rectified linear unit
RNN	Recurrent neural network
SINR	Signal-to-interference plus noise ratio
WMMSE	Weighted minimum mean-square error

For convenience, the abbreviations and notations used in this work are listed in Table 1 and Table 2 respectively.

II. RELATED WORKS

Deep learning has been studied by many, to address the challenges in wireless physical-layer [11]. It plays an increasingly important role in the mobile and wireless networking domain. However, to date, the deep learning models have been tailored to specific mobile networking applications as indicated next.

[1]The subcarrier allocation results in an output vector with high number of null values (sparse output).

TABLE 2: Notations and definitions.

Notation	Definition
N, U, B	Number of subcarriers, users, and BSs.
L_t, M_t	Number of multipath and incoming waves at time t.
k	Non-stationarity control parameter.
$h_b^u(\tau, t)$	Channel impulse response between uth user and bth BS at time t.
$g_b^u(n, t)$	Channel gain between uth user and bth BS on nth subcarrier at time t.
$\gamma_b^u(n, t)$	SINR between uth user and bth BS on nth subcarrier at time t.
$p_b^u(n, t)$	Power allocation on nth subcarrier between uth user and bth BS at time t.
$\alpha_b^u(n, t)$	Fraction of nth subcarrier allocated to uth user by time division at time t.
r_t	Sum-rate of the system at time t.
$\hat{R}_t$, $\overline{R}_t$	Relative sum-rate: Eq. (15) and the moving average value of $\hat{R}_t$: Eq. (16) at time t.
W	Bandwidth.
f_D	Doppler frequency.
ρ	Channel decay factor.
σ^2	Additive white gaussian noise.
P_{max}	Maximum power allocation for each BS.
Υ	Loss (cost) function of DNN.
G_t, P_t	Set of $g_b^u(n, t)$ values and $p_b^u(n, t)$ values for each user, BS, and subcarrier at time t.
t_1	Labeling time - Total time taken to label training set using (sub)optimal algorithm.
t_2	Training time - Total time taken to train the model using the training set.
t_3	Testing time - Total time taken to predict the power allocation using the learning model (DNN/DQN).

A. SUPERVISED LEARNING

Most of the deep learning approaches have been based on supervised learning models that utilize labeled datasets to train the model. In the context of resource allocation, supervised learning models have been proposed in works such as [14]–[19]. In [14], Sun et al. have proposed a DNN model generalization which approximates the WMMSE interference management algorithm with high approximation accuracy and higher computational efficiency compared to state-of-the-art interference management algorithms. In [15], the authors have proposed a resource allocation technique for small cells by employing deep learning for dynamic channel selection, carrier aggregation, and fractional spectrum. In [16] Zhou et al. have proposed an efficient DNN for resource allocation in cognitive radio networks aiming at the real-time performance to maximize the energy and spectral efficiency of the network. In [17], Li et al. have proposed a model that utilizes a Hopfield neural network to predict the bit and power allocation in a multi-user OFDM system. In [18], the authors propose a supervised DNN model for subcarrier assignment in an OFDMA/NOMA downlink video transmission system. The proposed model provides near-optimal performance with lower complexity. In [19], the authors propose a framework for "Learning to Optimize for Resource Management" which provides near-optimal performance using fewer data samples. In [20], the authors propose an artificial neural network to compute the optimal user-cell association based on the mobile users' positions in a realistic massive MIMO network. In [21], the authors redesign the classical communication protocol as an autoencoder, and propose a DNN based solution for the physical layer to enhance end-to-end communication in wireless networks. *Moreover, supervised learning techniques have been applied to many other communication problems. However, these works fail to address the performance degradation that occurs in the long run in a changing environment. Especially models which are evaluated on lab-generated datasets disregard the practical consideration related to the dynamicity of the environment and thus do not highlight the importance of retraining.*

B. UNSUPERVISED LEARNING

Unsupervised models, on the other hand, use alternative approaches to train the model to eliminate the dependency on a labeling algorithm. In [22], the authors propose a fast beamforming design method for the sum-rate maximization in a Multiple-Input and Multiple-Output single base station system using unsupervised learning. The proposed convolution model, which is trained offline and provides real-time service, improves computational speed significantly with performance close to the optimal WMMSE algorithm. In [23], the authors modify the training process of the proposed model in two steps with supervised pre-training and unsupervised re-training to optimize the performance. In [24], the authors propose an unsupervised DNN model for optimal resource allocation and interference minimization in multi-channel cognitive radio networks. In [25], the authors propose an unsupervised learning strategy based ensemble model for sum-rate maximization in a fading multi-user interference channel, which outperforms the state-of-the-art methods. In [26], a random edge graph neural network is proposed to parameterize the resource allocation policy which is trained using an unsupervised model-free, primal-dual learning method. *In general, unsupervised techniques converge to a local optimum and thus suffer from performance degradation with time similar to supervised models.*

C. REINFORCEMENT LEARNING

In addition to these approaches, deep reinforcement learning has also been used to solve wireless communications problems [27]–[34]. In the context of sum-rate maximization, DQNs are used commonly in small networks [27], [28]. In [27], the authors propose a distributively implemented, DQN model for multiple BS system that performs efficiently compared to classical techniques. Furthermore, in [28], the

authors propose a DQN based reinforcement learning model for sum-rate maximization in multiple BS system which is first pre-trained and then trained online and it provides better performance in comparison to iterative algorithms such as WMMSE. Apart from these, reinforcement learning has been used in other wireless resource allocation problems such as resource allocation in cognitive radio networks [29], [30], network slicing in radio access networks [31]–[33], and resource allocation for vehicle-to-vehicle communication [34]. Recently, it was a shift from knowledge-discovery toward knowledge-driven (autonomous) communications was demonstrated in [35], where an autonomous wireless system was implemented to address the problem of self-deployment of non-stationary radio nodes. Moreover, in [36], an environment-specific RL agent with Q-learning was devised to solve a self-optimization through joint channel association and location optimization through management system and reasoning for a new optimization strategy. *In general, reinforcement learning techniques have been shown to perform optimally in an efficient manner with low latency, even in dynamic environments that are susceptible to topological changes [34]. Online training enabled through reinforcement learning overcomes the performance degradation in dynamic environments. However, in a highly dynamic environment, the convergence to an optimal solution takes time and is not always guaranteed. Thus, it is important to study this effect and identifying whether the learning model would converge to an acceptable performance level.*

The studies in [14]–[36] have neglected the significant practical limitations related to training requirements of the learning model and the impact of ageing of the learning model due to the evolution of the wireless environment. Hence, we need to understand the requirements for retraining of the model in the presence of an ageing environment and the trade-off between the complexity and the computational efficiency to achieve the target service requirements.

III. SYSTEM MODEL AND PROBLEM FORMULATION

The study considers a multi-user multi-cell network as illustrated in Fig. 1 with U randomly located users. The network consists of B number of base stations (BSs) denoted by the set $\mathcal{B} = \{1, 2, 3, \ldots, B\}$. We address a particular BS by $b \in \mathcal{B}$ with the set of randomly located users $\mathcal{U} = \{1, 2, 3, \ldots, U\}$. The users connected to the bth BS are given by the set $\mathcal{U}_b \subset \mathcal{U}$, while a user $u \in \mathcal{U}$ is connected to only one BS at a given time, i.e. family of sets $\{\mathcal{U}_b\}_{b\in\mathcal{B}}$ is pairwise disjoint. In the OFDM system, a particular subcarrier is denoted by n from a set of subcarriers $\mathcal{N} = \{1, 2, 3, \ldots, N\}$.

As the propagation channel, we assume the time and frequency selective Rayleigh fading channel. The L-path discrete-time channel impulse response between the bth BS and the uth user at the moment t is represented by

$$h_b^u(\tau, t) = \sum_{l=0}^{L_t-1} \overline{h}(l, t)\delta(\tau - \tau_l(t)), \tag{1}$$

where $\overline{h}(l, t)$ denotes the lth path random complex gain. $\delta(\cdot)$ denotes the delta function and $\tau_l(t)$ denotes the time delay of the lth path. L_t indicates the time dependency of the number of paths due to user mobility. $\overline{h}(l, t)$ with M_t incoming waves in the receiver's vicinity is defined as zero-mean independent complex variables given by

$$\overline{h}(l, t) = \sum_{m=0}^{M_t-1} \exp[\theta(l, m)t + \phi(l, m)], \tag{2}$$

where $\theta(l, m) = 2\pi(m - r_1)/M_t$ and $\phi(l, m) = 2\pi r_2$ (r_1 and r_2 are random uniformly distributed numbers between 0 and 1). We also assume that the transmitter and receiver terminals are moving with pedestrian speed generating slow fading conditions. This movement is based on the random waypoint model, where the movement of mobile users, their location, velocity, and acceleration change over time randomly [37]. We assume that the user moves across different locations with a constant velocity however, the position of the user is random. As the users move the channel parameters: 1) the number of multipath (L_t) and 2) the number of incoming waves (M_t) vary depending on the instantaneous position of the user at time t. Thus, the non-stationary (dynamic) nature of the environment is determined by user mobility and defined by the number of paths L_t and the number of incoming waves M_t in Eq. (1). The channel gain $g_b^u(n, t)$ is given as an output of the fast Fourier transform of $h_b^u(\tau, t)$.

Without loss of generality, we made the following assumptions:

(A1) We assume $\tau_0 = 0 < \tau_1 < \ldots < \tau_{L_t-1}$ with the lth path time delay $\tau_l = l\Delta$ where $\Delta = 1$ denotes the time delay separation between adjacent paths.

(A2) We assume that expectation term $E[\sum_{l=0}^{L_t-1} |h_b^u(l, t)^2|] = \frac{1-\rho^{-1}}{1-\rho^{-L_t}} \sum_l \rho^{-l}\delta(\tau - \tau_l)$, where ρ denotes the channel decay factor that ensures the total energy of the channel is normalized to unity.

(A3) We assume the Jakes's fading model, where incoming rays constituting each propagation path arrive at a user with uniformly distributed angles [38]; the normalized autocorrelation function[2] is given by $E[h_b^u(\tau_1, t)h_b^u(\tau_2, t + \varsigma)] = J_0(2\pi f_D\varsigma)$ at delay ς when the maximum Doppler shift is f_D.

(A4) We assume that the size of the OFDM symbol is equivalent to the channel coherence time - the channel gains remain constant during an OFDM symbol and vary symbol-by-symbol. Thus, the negative effect of inter-carrier interference is not considered. The Jakes's fading channel model is assumed where the channel varies symbol-by-symbol as correlated fading. The fading in-phase and quadrature components are assumed to be independent and identically distributed (i.i.d.) random variables [39][3], which

[2]The autocorrelation function $E[h_b^u(\tau_1, t)h_b^u(\tau_2, t + \varsigma)]$ measures the statistical correlation between two propagation paths with propagation delays τ_1 and τ_2 as a function of time-difference ς. $J_0(\alpha) = \frac{1}{\pi}\int_0^{\pi} exp(j\alpha cos\theta)d\theta$ is the zero order Bessel function of the first kind.

[3]We note here that the simulations can be modified to introduce the interference, while later being reduced by using a cancellation technique.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2020.3040133, IEEE Access

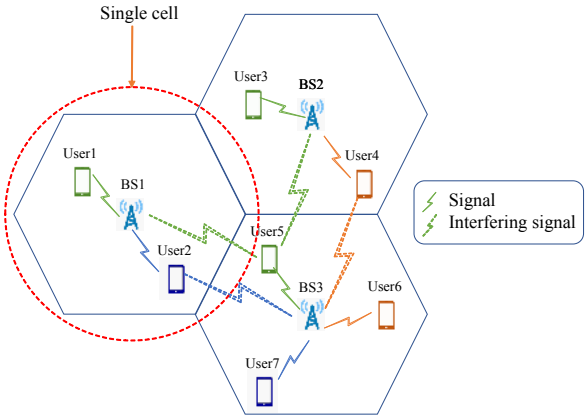


FIGURE 1: Network model.

is the important assumption for evaluation of reinforcement learning algorithm. Furthermore, we assume that the guard interval of each OFDM symbol is sufficiently long so that the inter-symbol interference is not present at any time instant.

Since the objective of the study is to investigate the implications of learning methods, we believe that such assumptions (i.e., an inter-carrier and inter-symbol interference-free OFDM system [39]) may not be a demerit for this work. We follow these assumptions to keep the investigation of our learning study trackable while performing a thorough analysis across different models. These assumptions help us emphasize the objective of this study and explain the findings related to learning more clearly.

Next, we formulate the following rate maximization problems as our case studies that are later used to analyze the effectiveness of learning models in different non-stationary environments.

A. LOW COMPLEXITY PROBLEM

We start with the rate maximization of the single-user single-cell OFDM system (i.e. $U = 1$, $B = 1$) where time index t has been omitted for brevity. The model is described by a single cell with only one user. The sum-rate maximization problem is defined by

$$\begin{aligned} \max_{p(n)} \quad & r = \sum_{n=1}^{N} W \log_2[1 + \gamma(n)] \\ s.t. \quad & 0 \leq \sum_{n=1}^{N} p(n) \leq P_{max} \end{aligned} \tag{3}$$

where r and W denote the transmission rate of the system and the bandwidth respectively. $\gamma(n)$ is the signal-to-interference plus noise ratio (SINR) between the user and its serving BS on the nth subcarrier, $p(n)$ is the power allocated to the nth subcarrier and P_{max} is the maximum power allocation of BS.

The corresponding SINR is given as follows

$$\gamma(n) = \frac{|g(n)|^2 p(n)}{\sigma^2} \tag{4}$$

where $g(n)$ is the channel gain between the user and the BS and σ^2 is the variance of the additive white Gaussian noise (AWGN).

To solve the rate maximization problem, we utilize the water filling algorithm described in [40] as a benchmark for this case study.

B. MODERATE COMPLEXITY PROBLEM

1) Multi-user Single-cell OFDM System

The model can be described by a single cell (B=1) in Fig. 1. The sum-rate maximization problem is given by

$$\begin{aligned} \max_{p(n)} \quad & r = \sum_{u=1}^{U} \sum_{n=1}^{N} W \log_2[1 + \gamma^u(n)] \\ s.t. \quad & 0 \leq \sum_{n=1}^{N} p^u(n) \leq P_{max}, \end{aligned} \tag{5}$$

where the SINR between the uth user and its serving BS, $\gamma^u(n)$, is given by

$$\gamma^u(n) = \frac{|g^u(n)|^2 p^u(n)}{\sigma^2}. \tag{6}$$

Here $g^u(n)$ is the channel gain between the uth user and its serving BS. $p^u(n)$ is the power allocation to the uth user on the nth subcarrier.

The optimization problem consists of two sub-problems namely subcarrier allocation and power allocation. Thus, it is comparatively more complex than the previous case. As the first step of the algorithm, each subcarrier is allocated to a unique user (but a single user can have multiple subcarriers). After subcarrier allocation, power is allocated to each subcarrier. As the benchmark, we utilize a greedy method in which the subcarrier allocation is done through ranking based on SINR [41] and the power allocation is accomplished using the water filling algorithm [40].

2) Multi-user Multi-cell Single-carrier System

The system is modeled with B BSs and U randomly located users as shown in Fig. 1. The sum-rate maximization problem is given by

$$\begin{aligned} \max_{p_b^u(n)} \quad & r = \sum_{b=1}^{B} \sum_{u=1}^{U} W \log_2[1 + \gamma_b^u(n)] \\ s.t. \quad & 0 \leq \sum_{u \in \mathcal{U}_b} p_b^u(n) \leq P_{max}; \forall b \in \mathcal{B} \end{aligned} \tag{7}$$

where n denotes the frequency component of the SC system. $p_b^u(n)$ is the down-link power allocation between the bth BS and the uth user. We consider a particular channel realization $h_b^u(t)$ having $L_t = 1$ (i.e. a single path channel) given by Eq. (1). Now, the SINR is defined by

$$\gamma_b^u(n) = \frac{|g_b^u(n)|^2 p_b^u(n)}{\sum_{b' \in \mathcal{B} \setminus \{b\}} |g_{b'}^u(n)|^2 p_{b'}^{u'}(n) + \overline{\sigma}^2} \tag{8}$$

where $p_b^u(n)$ is the downlink power allocation between bth BS and uth user. $\overline{\sigma}$ denotes the composite Gaussian noise variance of additive noise and residual Gaussian interference

after the frequency domain equalization. $g_{b'}^{u}(n)$ is the channel gain between uth user and BSs excluding bth BS, $p_{b'}^{u'}(n)$ is the down-link power allocation between BS b' and connected user u'. The weighted minimum mean-square error (WMMSE) algorithm [42] is used as a benchmark for this case study. The set difference operator is defined by "\".

C. HIGH COMPLEXITY PROBLEM

The system is modeled with B BSs, each with N subcarriers and U randomly located users as shown in Fig. 1. Due to inter-cell interference, the rate maximization problem is Non-deterministic Polynomial-time hard [43]. The sum-rate maximization problem is given by

$$\begin{aligned} \max_{p_b^u(n)} \quad & r = \sum_{b=1}^{B}\sum_{u=1}^{U}\sum_{n=1}^{N} W \log_2[1+\gamma_b^u(n)] \\ s.t. \quad & 0 \leq \sum_{u\in\mathcal{U}_b} p_b^u(n) \leq P_{max}; \forall b \in \mathcal{B}, \end{aligned} \tag{9}$$

where $\gamma_b^u(n)$ and $p_b^u(n)$ denote the SINR and the allocated power between the uth user and the bth BS on nth subcarrier. The SINR $\gamma_b^u(n)$ is given by

$$\gamma_b^u(n) = \frac{|g_b^u(n)|^2 p_b^u(n)}{\sum_{b'\in\mathcal{B}\setminus\{b\}}\sum_{u'\in\mathcal{U}_{b'}} \alpha_{b'}^{u'}(n) p_{b'}^{u'}(n)|g_{b'}^{u}(n)|^2 + \sigma^2}. \tag{10}$$

We assume that each user can utilize any of the subcarriers, while users in the same cell cannot utilize the same subcarriers at the same time. The load variable $\alpha_b^u(n) \in [0,1]$, is defined as the fraction of subcarrier n allocated to user $u \in \mathcal{U}_b$ by time division. Intuitively, $\alpha_b^u(n)$ can be interpreted as the probability of receiving interference from bth BS on nth subcarrier. To ensure that users in the same cell do not occupy the same subcarrier at the same time, we must have $\sum_{u\in\mathcal{U}_b} \alpha_b^u(n) \leq 1, \forall n \in \mathcal{N}$ and any given time.

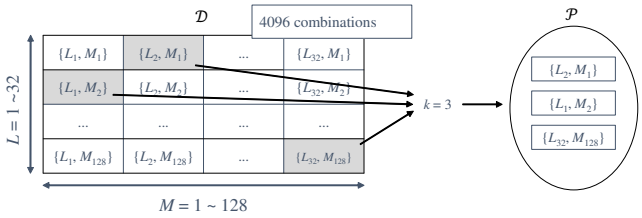
Due to the high data rates in BSs, it is not practical to evaluate instantaneous interference and therefore, we consider an average interference taken over time. According to this assumption, the total power $q_b(n) = \sum_{u\in\mathcal{U}_b} \alpha_b^u(n) p_b^u(n)$, $\forall b \in \mathcal{B}, \forall n \in \mathcal{N}$ with Eq. (10) leads to the SINR given by

$$\gamma_b^u(n) = \frac{p_b^u(n)|g_b^u(n)|^2}{\sum_{b'\in\mathcal{B}\setminus\{b\}} q_{b'}(n)|g_{b'}^u(n)|^2 + \sigma^2}. \tag{11}$$

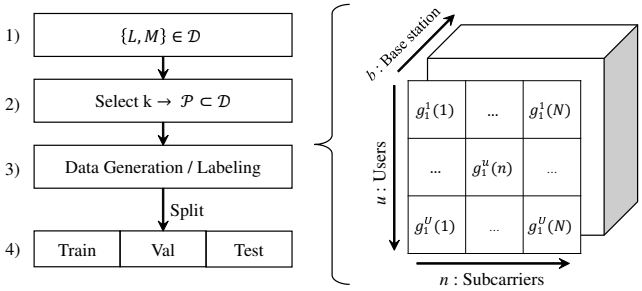
Now, the problem defined by Eq. (9) and Eq. (11) is a non-convex optimization problem. We use the suboptimal linear programming method proposed in [44] as the benchmark. In [44], the problem is converted to a convex optimization problem and solved by sequential least squares programming (SLSQP) optimization.

IV. LEARNING PRELIMINARIES

We first develop a generic framework for dataset generation, and then we briefly describe DNN and DQN models. The proposed methodology has 2 main steps namely: 1) Data generation and labeling and 2) Training and evaluation. The pseudocode for these steps are given in the relevant sections.



(a) (L_t, M_t) pairs.



(b) Framework.

FIGURE 2: Data generation.

A. DATASET GENERATION FRAMEWORK

For common testing in the non-stationary environment, we devise the generic dataset generation method that can be used to benchmark across different learning models.

Fig. 2 illustrates the data generation, labeling, and training process devised in this paper. This is the general process for the data generation, and it can be applied to any type of wireless application. The following steps are taken:

1) We assume that (L_t, M_t) pairs in Eq. (1) may only vary within a full set $\mathcal{D} = \{(L, M) \mid L \in \mathbb{Z}^+_{L_{max}}, M \in \mathbb{Z}^+_{M_{max}}\}$ where $\mathbb{Z}^+_\alpha = \{1, \ldots, \alpha\}$ represents the positive set of integers up to α. (e.g. if L_{max} = 32, and M_{max} = 128, L_t varies between 1 and 32, while M_t varies between 1 and 128, resulting in a full set with $card(\mathcal{D})$=4096, where $card(\cdot)$ denotes the cardinality of the set. See Fig. 2a).
2) For the sake of computational complexity, we assume that the user's movement area is constrained. The degree of the user's movement is given by the non-stationarity control parameter k and it determines the number of stationary points in the user's constrained movement area. A subset $\mathcal{P} \subset \mathcal{D}$ $(card(\mathcal{P}) = k)$ is created by choosing k number of (L_t, M_t) pairs from the full set $\mathcal{D}$ and each (L_t, M_t) pair corresponds to a stationary point in the user's movement area as illustrated in Fig. 2a. We assume that the user's mobility results in an observation of a limited number of L_t and M_t values which always lie within the subset $\mathcal{P}$. Note that the non-stationarity parameter k can also be interpreted as a measure of randomness of a dataset. i.e. a higher k describes the higher user mobility/randomness and thus, a higher number of possible (L_t, M_t) pairs and vice versa.

TABLE 3: System parameters and settings.

Parameters	Setting
k	1 (Stationary), [2, 4096] (Non-stationary)
L_{max}	32
M_{max}	128
W	100 Mbps
f_D	40 Hz
ρ	0.2 dB
σ^2	10^{-3} µW
P_{max}	10 µW
N, B, U	Vary depending on the case study.

3) A dataset of $g_b^u(n)$ values are generated as shown in Fig. 2 by using k selected pairs $(L_t, M_t) \in \mathcal{P}$. Next, the dataset is labeled using a known optimal/suboptimal algorithm[4]. In our sum-rate maximization problem, the labeled dataset describes the output as power allocation for the given input as user mobility determined channel gain.
4) The labeled dataset is split as Training, Validation and Testing sets[5] for evaluation of deep learning models as described next.

For convenience, the channel generation parameters are summarized in Table 3. Unless stated otherwise, these parameters are used for all the studies. The pseudocode for the data generation framework and labeling is given in Pseudocode 1.

Pseudocode 1 Data generation framework and labeling.

```
Initialize network model parameters
        (N, B, U, L_max, M_max, D, etc.)
Choose non-stationarity parameter k and
        generate set P ⊂ D                    ▷ refer Fig. 2a
for t = 1 to No. of samples do
    Choose (L, M) ∈ P                         ▷ refer Fig. 2b
    Generate channel impulse h_b^u(τ, t)      ▷ refer Eq. (1)
    Calculate channel gain g_b^u(n, t) = FFT(h_b^u(τ, t))
    Find (near-)optimal power allocation.
        ▷ refer water filling [40], WMMSE [42], SLSQP [44]
Split dataset as train, validation, and test  ▷ refer Fig. 2b
```

B. LEARNING MODELS

1) Deep Neural Networks

DNNs are supervised learning models that are used to find patterns in a dataset and make predictions upon them. Given a labeled dataset with input and its expected output pair, the DNN model learns the dataset structure generators (i.e. pattern) and predicts on a new set of data based on them. At a time t, the input is the set of channel gain values given by $G_t = \{g_b^u(n) \mid \forall n \in \mathcal{N}, \forall b \in \mathcal{B}, \forall u \in \mathcal{U}_b\}$ and the target output is the power allocation strategy set given by $P_t = \{p_b^u(n) \mid \forall n \in \mathcal{N}, \forall b \in \mathcal{B}, \forall u \in \mathcal{U}_b\}$.

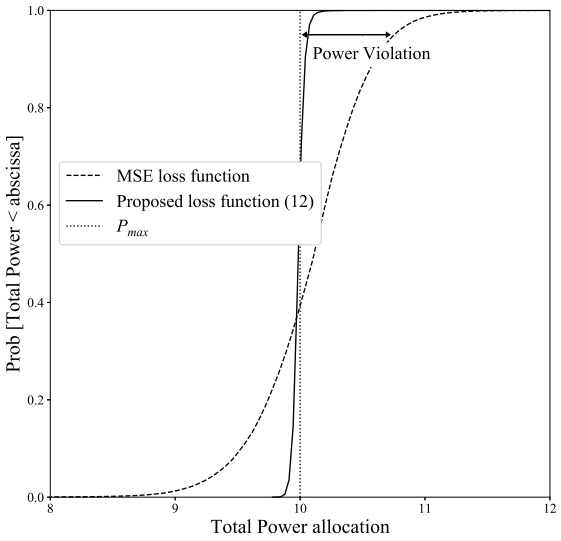


FIGURE 3: Illustration of power violation problem at the output of the DNN model.

In this study, we utilize a fully connected neural network with multiple hidden layers to predict the power allocation. The DNN takes G_t as the input and P_t as the output. Thus, in general, the input layer has a size $card(G_t)$ and the output layer is of size $card(P_t)$. The number and size of hidden layers vary depending on the case study. Each layer uses ReLU (Rectified linear unit) as the activation function except for the output layer which uses Leaky ReLU with a low slope (i.e. gradient = 0.01) to avoid the dying ReLU problem [13].[6]

a) Training

The training set is used to derive the weights of the model through backpropagation. Traditionally, the model is trained such that the loss (cost) function Υ, defined by the mean square error (MSE)[7] between the expected (near-)optimal output $p_b^u(n)$ and the DNN predictions $\hat{p}_b^u(n)$, is minimized. However, the independent allocation of power to each subcarrier might lead to total power being excess of the total power allocation budget P_{max} in Eq. (9), i.e. a power violation problem. This occurs since the DNN model is not aware of the power constraint in Eq. (9).

To solve the power violation problem of DNN we propose

[4]Depending on the specific case study the dimensions of the data sample and the specific labeling algorithm may change. The default dataset size is hundred thousand (100K) samples.

[5]Unless stated otherwise, the labeled dataset is split as follows: the first 20K data samples are the training set, the next 20K samples are the validation set, and the last 60K are the testing set.

[6]Unless stated otherwise, all DNNs in this study have the same architecture

[7]MSE is generally used as the loss function for a regression problem in a form $\sum_{b\in\mathcal{B}}\sum_{u\in\mathcal{U}_b}\sum_{n\in\mathcal{N}}\left|\hat{p}_b^u(n) - p_b^u(n)\right|^2$ [13].

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2020.3040133, IEEE Access

a loss function[8] which regulates the maximum power limit constraint as follows

$$\Upsilon = \sum_{b\in\mathcal{B}} \left(\sum_{u\in\mathcal{U}_b} \sum_{n\in\mathcal{N}} |\hat{p}_b^u(n) - p_b^u(n)|^2 + \beta \left| \left(\sum_{u\in\mathcal{U}_b} \sum_{n\in\mathcal{N}} \hat{p}_b^u(n) \right) - P_{max} \right|^2 \right) \tag{12}$$

where β is the weighing factor of the power violation term. Fig. 3 shows the effect of the modified loss function and how the standard MSE loss function [13] results in high power allocation and exceeds the maximum BS power P_{max} which violates the maximum power constraint in Eq. (9).

The loss Υ is minimized by varying the model weights, and the minimum of the loss function is searched by the gradient descent algorithm [13]. The update of weights in each layer independently results in redundant calculation and thus, the backpropagation (chain rule) algorithm is used to avoid redundant calculations. The backpropagation is done through the mini-batch gradient descent method[9]. This results in a higher speed of training due to parallel computation and avoids problems such as the over/underestimation of the error [45]. Furthermore, the Adam optimization technique is used to accelerate the training process [46].

The training process is made up of multiple training iterations[10]. After the completion of each training iteration, the updated model is checked for any improvements through the validation stage as described next.

b) Validation and Convergence

After the update of the weights, the improvements in the model are validated in this stage. A model is considered to have improved if the error Eq. (12) on the validation set has reduced after the update.

Finally, the training and validation steps are repeated until convergence for multiple epochs[11]. The DNN model is said to have converged when the minimum of the loss function is reached. In general, the back-propagation algorithm cannot be shown to converge, and there are no well-defined criteria for stopping its operation. Furthermore, a low number of epochs will result in underfitting, while a large number would result in overfitting [47]. Therefore, we make a reasonable assumption regarding the convergence point and terminate the training process by defining the following threshold parameters [48]:

[8]Unless stated otherwise, all DNN models use the proposed loss function Eq. (12) with $\beta = 0.1$.

[9]The training set is broken down into multiple batches in this method. A Batch refers to a set of data samples. Unless stated otherwise, the batch size of 32 samples is used.

[10]A training iteration is when a single batch has been passed forward and backward through the DNN once

[11]An epoch is when the entire training set has been passed forward and backward through a DNN once. In other words, when the training iterations are done on all the batches in the training set, it is known as an epoch.

TABLE 4: DNN parameters.

DNN Parameters	Setting
Batch size	32
Total epochs E_{tot}	100 epochs
Early stopping: error ϖ	10^{-6}
Early stopping: epochs $\hat{E}$	50 epochs.
Input layer size	$card(G_t)$
Output layer size	$card(P_t)$
Loss function Υ	Eq. (12) with $\beta = 0.1$

(i) First the change in error $|\Upsilon_e - \Upsilon_{e-1}| \leq \varpi$ is measured where Υ_e denotes the error at the eth epoch and ϖ denotes the minimum threshold for the change in error.

(ii) The error change is not monotonic, thus an increase in error doesn't indicate that the loss is minimal. Therefore, the change in error measured in (i) is monitored for $\hat{E}$ number of epochs and if it is still below ϖ, then we assume convergence.

(iii) Finally, if the aforementioned criterion is not met then the model is stopped after E_{tot} epochs to avoid overfitting.

The termination of the training process is therefore determined by either of the three bullets defined above[12].

c) Testing

Once the DNN model is trained, the model is evaluated based on the prediction made on the test data. Since this data is unseen during the training and validation steps, the evaluation of the model on this set can be generalized to all other datasets.

For convenience, the parameters of the DNN are summarized in Table 4. Unless stated otherwise, the DNN has the architecture and parameters specified. The pseudocode for the DNN training and evaluation methodology is given in Pseudocode 2.

Pseudocode 2 DNN training and evaluation.

```
Create the DNN model and initialize parameters
        (Loss, Epochs, etc.)
for e = 1 to E_tot do
    Predict user/power allocation          ▷ refer Section IV-B1
    Calculate Loss.                        ▷ refer Eq. (12)
    Update model weights through gradient descent.
                                           ▷ refer Section IV-B1 a)
    Validate updated weights using the validation set.
                                           ▷ refer Section IV-B1 b)
    Check for early stopping.              ▷ refer Section IV-B1 b)
    Transfer weights at regular intervals for semi-online
      training.                            ▷ refer Section V-A3
Evaluate DNN performance using the testing set.
                                           ▷ refer Section IV-B1 c)
```

[12]Unless stated otherwise, E_{tot} = 100 epochs, $\varpi = 10^{-6}$, and $\hat{E}$ = 50 epochs.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2020.3040133, IEEE Access

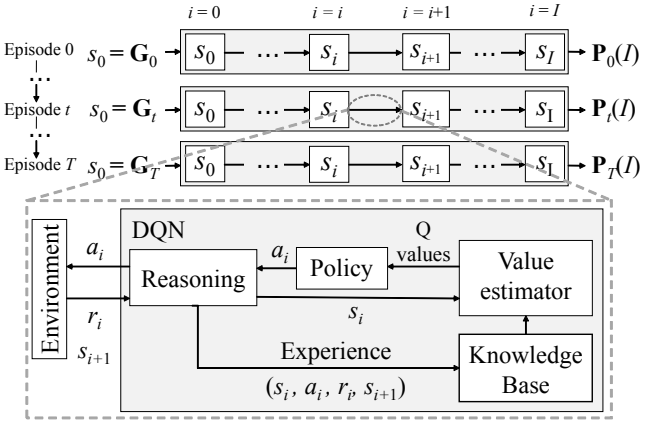


FIGURE 4: DQN agent model.

2) Deep Q-learning

Deep Q-learning is a reinforcement learning model [49], [50] that combines deep learning and reinforcement learning to form powerful models that can be used to solve wireless problems in a non-stationary environment [51]. The DQN observes the environment through a set of environment-specific observable parameters ($\mathcal{S}$ observation space); interacts with an environment by a set of actions ($\mathcal{A}$ action space) that can be executed and tries to increase the reward through a specific action from the action space. Our objective is to predict the optimal power allocation strategy given a set of observed channel gains. As opposed to DNNs, DQNs do not require a labeled dataset, but DQNs can be used only for Markov Decision Processes (MDP). Therefore, the given problem must be converted to an MDP in which for each state $s_i \in \mathcal{S}$ and action $a_i \in \mathcal{A}$, the resulting state s_{i+1} would only depend on the prior state s_i and the corresponding action a_i. Note that, DQN cannot be directly applied to the sum-rate maximization problem formulation since in time domain, the future state G_{t+1} is not dependent on the current state G_t and the corresponding action P_t. Therefore, we reformulate the problem by focusing on time t and consider the process of estimating P_t using G_t to be an MDP as explained in the following section.

a) DQN preliminaries

We propose an MDP that allows the DQN to adjust the elements of P_t iteratively at each time step t depending on the channel gain G_t. The process of estimating the set P_t using G_t for a given time t is defined as an episode in our study. Each episode corresponds to a time $t > [t+1, t+2, \ldots)$ and each episode consists of several steps (denoted by i) as illustrated in Fig. 4. At the beginning of an episode, the initial state s_0 is observed from the environment. The state space contains the channel state information (CSI) for each user and subcarrier in a given network (The exact details about the state space is defined in the relevant section where the DQN model is introduced). Depending on the state s_i, an action a_i is taken which results in new power allocation and as a result a new state s_{i+1} is observed along with the corresponding reward r_i. This is given by an arrow circled using dotted lines in Fig. 4. The underlying processes of the DQN are also shown in Fig. 4 and they will be explained later.

The action a_i is chosen based on the Q-values which have a direct correlation to the reward r_i resulting from each action a_i. The value estimator: DNN, estimates the Q-values for all possible actions for a given state s_i. Then, a suitable action a_i is selected using the policy considering the Q-values obtained from the value estimator. Then, the action a_i is executed on the environment and the corresponding reward r_i for action a_i in state s_i is obtained along with the resulting state s_{i+1} of the environment. The total sum-rate of the network (given in Section III) is used as the reward since our objective is to maximize the sum-rate. Finally, the tuple (s_i, a_i, r_i, s_{i+1}), which is defined as the experience, is stored in the Knowledge Base.

The state space s_i consists of the channel gain information G_t and the power allocation $P_t(i)$. At each step i, depending on the action a_i, the power allocation $P_t(i)$ is updated, resulting in a new state s_{i+1}. Thus, it is evident that the future state s_{i+1} can be derived using the current state s_i and action a_i making this an MDP. To estimate the power allocation, we propose the following MDP formulation depending on the network model:

- Single carrier system: In a single carrier system, power can be allocated without violating the power constraint as described in Section IV-B1. Thus, at each step i, the DQN chooses a discrete level from the action space a_i based on which the power is allocated to a user.
- Multicarrier system (OFDM): In a multi-user OFDM system, power cannot be allocated to each user independently as in the previous case, as the power violation problem may occur (see discussion in Section IV-B1). However, by dividing the power budget P_{max} into discrete increments of I levels, and allocating $\delta = P_{max}/I$ amount of power to each subcarrier at each step i, we can ensure that the total power allocation does not exceed P_{max} (resulting in the case of no power violation).

In a new episode, DQNs usually reset the environment to a predefined starting state. But in our study, in a new episode, the channel gain $\mathbf{G}_{t+1}$ is correlated to the previous episode's channel gain $\mathbf{G}_t$. Thus, there are two approaches to initialize the power allocation at the start of an episode.

- We assume that a particular time t is independent of previous time steps $t \in (0, t-1)$. i.e. an episode is independent of previous episodes. Therefore, all elements of $P_{t,0}$ are initialized to zero at the beginning of an episode.
- We assume that the power allocated at time t is related to time $t+1$. Thus, we allocate $P_t(I)$ to $P_{t+1}(0)$ at the beginning of an episode.

In this study, we consider the first initialization method. An episode ends at $i = I$ when the entire set $P_t(i = I)$ is estimated or if $P_t(i)$ leads to a power violation.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2020.3040133, IEEE Access

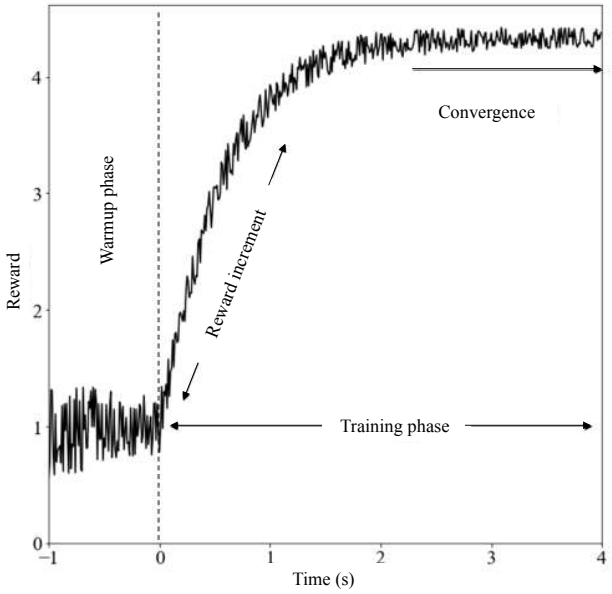


FIGURE 5: Different phases in DQN training process.

b) DQN operation

The operation of the DQN can be described through the four phases shown in Fig. 5.

Warm-up phase

In the beginning, the DQN agent has no experience about its environment, and the knowledge base is empty (i.e. observation, taken action, and the corresponding reward). Initially, the weights of the Value estimator are initialized randomly in the range [0,1]. To start the operation, we obtain the experience by executing actions on the environment based on the Q values from the initialized value estimator without training the DQN. Then these experiences are stored in the knowledge base. This phase is known as the warm-up phase. We select the warm-up phase to be 100 episodes so that the knowledge base is adequately filled.

Exploration/Exploitation Mechanism [50]

The decision-making function (or the Policy) is in a dilemma between two actions: 1) selecting the action that worked so far - exploiting the knowledge, or 2) choosing a random action without considering Q-values to gain more reward - exploration for higher rewards. By only exploiting DQN cannot reach the optimal solution, while if we often explore the convergence is slow. The trade-off between the exploration and exploitation is designed by ϵ-greedy strategy in the policy. In this study, the DQN starts the training phase with ϵ=0.8 at the beginning and it is linearly reduced to ϵ=0.01 within 1000 episodes and then maintained at 0.01 (1000 episodes is equivalent to 1 second since we sample the channel gain information for each 1ms). We chose these specific values through trial and error so that the DQN converges to a solution quicker.

TABLE 5: DQN parameters.

DQN Parameters	Setting
Number of episodes	100K
Warm-up episodes	100
Steps per episode (Episode size) I	50
Epsilon ϵ	$0.8 \rightarrow 0.01$
Discount factor Γ	0.99
Knowledge base: memory size	5000
Value estimator: Learning rate μ	0.01
Value estimator: batch size	256
Value estimator: architecture	Section specific

Training

The Q-value for the state s_i and action a_i is given by

$$Q(s_i, a_i; \theta) = E[r_i + \Gamma Q(s_{i+1}, a_{i+1})] \tag{13}$$

where θ denotes the parameters/weights of the value estimator implemented by DNN in Fig. 4. Γ (=0.99) is the discount factor that adds the effect of valuing rewards received earlier higher than the rewards received later. During training, a set of experiences is retrieved from the knowledge base. We train the value estimator to predict Q values such that the DQN maximizes the final reward by updating θ as follows

$$\theta_{i+1} := \theta_i + \mu \left[r_i + \Gamma \max_{a'} Q(s_{i+1}, a'; \theta_i) - Q(s_i, a_i; \theta_i) \right] \times \nabla Q(s_i, a_i; \theta_i) \tag{14}$$

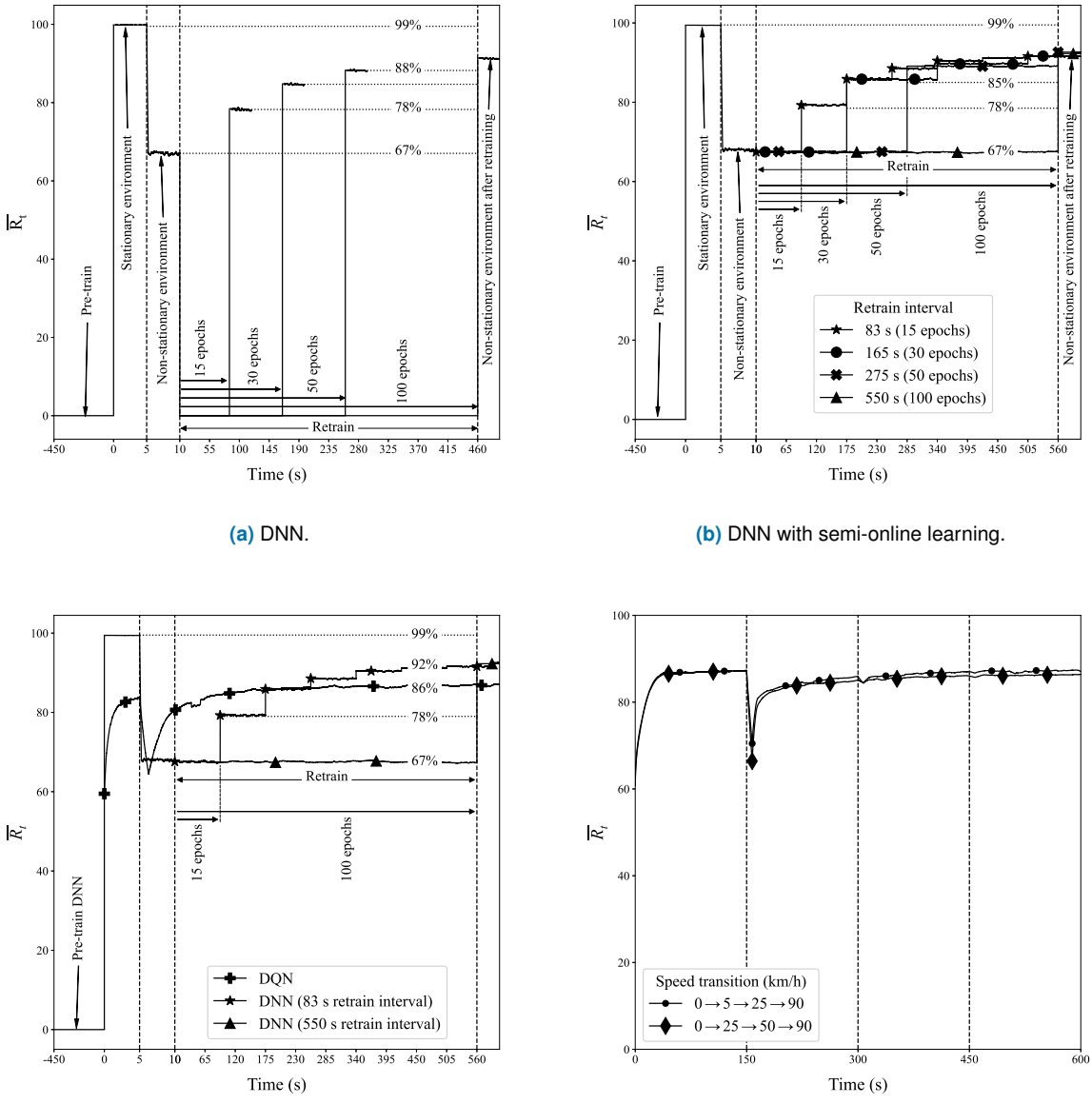
where μ is the learning rate (hyperparameter for optimization algorithm that determines the size at each iteration while moving toward a minimum of a loss function). This is known as experience replay.

Convergence and Prediction

When the training phase begins, the DQN starts to train the value estimator to predict Q values that are more likely to give higher rewards. In general, during the training phase, the reward increases as shown in Fig. 5. This reward increment period can vary with the complexity of the problem, the DNN model of the value estimator, hyperparameters used to train the DQN, and the computing power. A DQN is said to be converged when the reward given by the DQN for a specific environment plateaued in a maximum reward as illustrated in Fig. 5. We can assume that the action taken by the DQN is near-optimal after the convergence of the DQN.

Since DQN is an online model, prediction can be done while training. When predicting, ϵ-greedy policy uses $\epsilon = 0$. The reward increases with the experience gained while exploring/exploiting the environment. For convenience, the parameters of the DQN are summarized in Table 5. Unless stated otherwise, we use the specified parameters. The pseudocode for training and evaluation of the DQN is given in Pseudocode 3. Note that after convergence the training can be stopped and the model can be evaluated by selecting the best action ($\epsilon = 0$).

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2020.3040133, IEEE Access



(a) DNN.

(b) DNN with semi-online learning.

(c) DQN.

(d) DQN under fast-fading conditions.

FIGURE 6: Variation of performance of learning models in a mobile radio (i.e. non-stationary) environment.

V. CASE STUDIES AND DISCUSSIONS

A. LEARNING MODEL AGEING IN A WIRELESS ENVIRONMENT

First, we focus our attention on ageing of the learning model with respect to non-stationary radio environment. For this, we consider the sum-rate maximization problem as described in Section III-A for a single-user single-cell OFDM system with $N = 32$ subcarriers. We present the results averaged for 20 simulations in Fig. 6.

In this case study, we compare and analyze the performance in both stationary and non-stationary environments. During the initial stage, we consider the environment to be stationary and the model is trained under this condition. The channel between a stationary user and a BS is defined by Eq. (1) with $L_t = 6$, $M_t = 20$ incoming waves in the user's vicinity, and an exponential power delay profile with decay factor $\rho = 0.2$ dB. A dataset is generated as described in Section IV-A. The water filling [40] algorithm, which derives the optimal power loading, was used to label and slice the dataset as shown in Fig. 2.

1) Stationary environment

We utilize a DNN with 5 hidden layers each containing 300 nodes as described in Section IV-B1. The DNN is initially

Pseudocode 3 DQN training and evaluation.

```
Create the DNN used to estimate Q-values and
    initialize model parameters (Loss, Epochs, etc.)
for each episode t do
    Initialize the starting state s_0
    for i = 0 to I − 1 do                          ▷ refer Fig. 4
        Calculate Q-values by feeding s_i to DQN.
        Choose the action a_i based on the epsilon
          greedy policy.                  ▷ refer Section IV-B2 b)
        Calculate the reward r_i and derive the
          new state s_{i+1}.              ▷ refer Section IV-B2 a)
        Store the experience (s_i, a_i, r_i, s_{i+1}) in the
          knowledge base.                 ▷ refer Section IV-B2 a)
    Train the DQN through experience replay
        at regular intervals.                      ▷ refer Eq. (14)
```

trained in the stationary environment for 450 seconds using 20K training and 5K validation samples. We consider the time at which the training ends to be the $t = 0$s point. Then, the system performance is evaluated by measuring the relative sum-rate defined as

$$\hat{R}_t = \frac{\hat{r}_t}{r_t} \times 100\%, \tag{15}$$

where $\hat{r}_t$ is the sum-rate obtained by the output of the DNN model and r_t is the expected sum-rate given by the water filling algorithm [40]. Since the relative sum-rate per sample might have a high dynamic range we define the moving average of the relative sum-rate as follows:

$$\overline{R}_t = \frac{1}{w} \sum_{i=t-w}^{t} \hat{R}_i. \tag{16}$$

where w is the time window size[13]. From Fig. 6a we observe that the $\overline{R}_t$ value is $\approx 99.5\%$ in the stationary environment indicating that the DNN output is close to the optimal sum-rate.

2) Non-stationary environment

The DNN trained in the stationary wireless environment provides near-optimal performance for $t \in (0, 5)$ as shown in Fig. 6a. Next, we let the environment evolve after at $t = 5$ to observe the performance of the pre-trained model in a non-stationary environment. To observe the effect of non-stationarity, we simulate the user movement at a vehicular speed of 20 km/h at $t = 5$s as shown in Fig. 6a. The environment generators such as the distance, multipath, and scattered waves vary as a result. Therefore, a new dataset is created with $k = 100$, $L_{max} = 32$ and $M_{max} = 128$ as described in Section IV-A. In comparison to the optimal solution, a significant drop in the DNN prediction performance is observed due to the non-stationarity after the 5^{th} second. This is because the pre-trained DNN model does not capture the various environment generators defined by $k = 100$ due to user mobility.

Thus, in order to capture this new variation, the DNN model was next retrained after the 10^{th} second with data generated in the new environment. The training process was redone multiple times to analyze the effect of training duration on the performance of the DNN. The training duration (number of epochs) was increased as shown in Fig. 6a and we observed that the relative sum-rate increases from 78% to 92% as the number of epochs is increased from 15 to 100. The increase is significant from 15 to 50 epochs, but it saturates at about 100 epochs. This confirms that the loss function reduces and saturates after some time which is shown by the increase and saturation of the relative performance.

The time axis in Fig. 6a has been scaled to clearly visualize the performance and thus the training time is much longer compared to the prediction time. So, we note that the performance increases with the training time (number of epochs) at the expense of efficiency. This trade-off is studied in detail in later studies.

Another observation is that the relative sum-rate drops to zero during the retraining period. This is because the new training set is labeled by a traditional signal processing algorithm and the DNN is retrained during this period. This is referred to as the offline training period when the DNN model cannot make any new predictions. In a highly non-stationary system, this retraining is vital to maintain the wireless performance at the target level, and retraining must be done repeatedly as the environment evolves and the wireless performance degrades below a threshold. However, frequent offline training triggered by a non-stationary environment effectively reduces the overall system performance and efficiency of the DNN. This makes the usage of DNNs a challenging task in general for ultra-low latency applications such as augmented reality which require the system to be at continuous operation at an acceptable reliability level at all times.

3) Semi-online training through dual DNN in a non-stationary environment

It can be seen from Fig. 6a that the relative sum-rate $\overline{R}_t$ falls to zero during the offline retraining period which makes it unsuitable for real-time applications. To overcome this issue, we exploit the fact that the current model is already trained well enough for the stationary system. We introduce a semi-online training methodology, whereby the learning model copes with the drop in performance through continuous retraining. However, the model cannot make predictions while being trained. Thus, two DNN models are used in parallel, one for training and another for prediction. The prediction model does continuous predictions, while the training model continues the cycle of reading the input, labeling, training and validation, and periodic update of the prediction model weights. While training, the low performance of the DNN model is maintained: we maintain the effective performance at 67% using this semi-online learning mechanism as shown

[13]Unless stated otherwise, a window size of $w = 5000$ is used.

in Fig. 6b. Consequently, by employing this approach we guarantee the continuous operation of the wireless system without service dropping to zero while maintaining the prediction performance at the expense of slightly higher prediction time in non-stationary environments. However, the channel changes can occur rapidly and not gradually as we assume in this section. This would trigger frequent retraining and such a model requires a sophisticated learning system for parallel operation. Therefore, retraining the proposed model in a highly mobile (i.e. non-stationary) environment would not be practical due to the limited availability of resources.

4) Online training through DQN in a non-stationary environment

The high computational requirements needed for continuous retraining makes the proposed semi-online training methodology impractical. However, reinforcement learning models such as DQNs provide efficient means for online training which would be highly useful in this scenario. We repeat the experiments using a DQN model. The problem is converted to an MDP first using the first MDP formulation specified in Section IV-B2:

- *State space* - The state space s_i is formed by the concatenation of the current channel gain values of each subcarrier $\mathbf{G}_t$, the previous power allocation of each subcarrier $\mathbf{P}_t(i-1)$, and the corresponding rate of each subcarrier $\mathbf{R}_t(i-1)$.
- *Action Space* - The action space has a size $A = N$ and it corresponds to the power allocation to each subcarrier. At each time step i, the power of the chosen subcarrier is incremented by $\delta = P_{max}/I$. Note that after I steps the total power allocation would be P_{max} and thus power violation doesn't occur.
- *Reward* - We use the sum-rate which is given by Eq. (3) as the reward for the DQN.
- *Value estimator* - We use a DNN with 2 hidden layers each of size 256 and ReLu activation for each layer. Furthermore, we use the parameters specified in Table 5.

As shown in Fig. 6c, we begin the training at $t = 0$ and the model continues its online training and prediction cycle without any interruptions. Initially, we observe that the relative performance of DQN starts at a low value and increments steadily. This occurs since reinforcement learning models such as DQNs are not pre-trained unlike supervised learning models such as DNNs. Then at $t = 5$s, the environment starts changing, and as a result, the performance of DQN drops. However, soon after the drop, the DQN model recovers its performance level unlike the DNN or the proposed semi-online training methodology. This steady increment of DQNs performance is attributed to its efficient online training methodology and thus DQNs can perform retraining more efficiently in a non-stationary environment as opposed to DNNs. Thus, reinforcement learning methodologies are much more suitable for real-time time-constrained applications under non-stationary wireless environment.

Finally, after some time, we could observe that the performance of the DNN models reaches and exceeds the performance of the saturated DQN performance. The DNNs perform better in this scenario since they are trained using optimal power allocation values, unlike DQNs which use exploration and exploitation strategy for training. This gives DNN a competitive advantage if the labeling algorithm provides the optimal solution. In later sections, we study whether DNNs have the same advantage in complex problems where the labeling algorithm is suboptimal.

5) Variation of DQN performance under fast-fading condition

In the previous section, we observed that the DQN handles non-stationarity comparatively well. Even though the reinforcement learning model experiences a sudden drop in performance, after some time the model recovers and converges to the initial performance level. Note that, the time taken for convergence is slightly higher in a non-stationary environment ($\sim$100 s) when compared to the stationary environment ($\sim$20 s), suggesting that as the non-stationarity increases, the convergence time increases too. Thus, if the channel conditions vary rapidly, the reinforcement learning model may not be able to learn the channel parameters in time. As a result, the reinforcement learning model may not converge and would have comparatively lower performance. To observe this effect, these conditions were simulated by changing the velocity of the user (and thereby the Doppler frequency). Fig. 6d shows the variation of performance of DQN as the user movement varies in a stepwise manner, from stationary position to pedestrian velocity, and finally to high vehicular speed. First, when the speed is increased in steps as 0 km/h $\rightarrow$ 5 km/h $\rightarrow$ 25 km/h $\rightarrow$ 90 km/h, we observe that at each transition the DQN model's performance drops and then recovers. This drop occurs due to the change in the system and the recovery suggests that the model was able to learn these changes. Next, as the first transition in velocity is increased from 0 km/h $\rightarrow$ 25 km/h, the drop in performance is slightly higher. However, with time, the drop during the transitions (from medium to high speed) becomes insignificant in both these cases. This is because the first case study has relatively low complexity and thus the model can learn the system changes easily and converge faster. Therefore, the DQN performs sufficiently well under fast-fading conditions in this case study. In later sections, we discuss whether these observations hold for higher complexity problems under fast fading conditions and we study the effect of increasing complexity on the variation of performance of DQN.

B. LOW COMPLEXITY PROBLEM

We consider an OFDM system with N = 128 subcarriers. For the propagation channel, we have chosen the normalized Doppler frequency $f_D \times T \times N = 5 \times 10^{-5}$ where f_D = 40 Hz is the Doppler frequency with a user data rate of 1/T = 100 Mbps. Such Doppler frequency corresponds to moving

TABLE 6: Variation of $\overline{R}_t$ with the number of subcarriers N.

N	t_2 (min)	t_3(ns)/sample	$\overline{R}_t$
32	5.8	27	99%
64	6.1	29	95%
128	7.7	36	90%
256	9.1	38	79%
512	12.6	52	39%

terminal speed of 22 km/h for 2 GHz carrier frequency.[14] The computer simulations were done using $\sigma^2 = 10^{-3}$ µW, and P_{max} = 10 µW.

A dataset is generated with a non-stationarity factor of $k = 10$, then labeled using the water-filling algorithm [40] and finally split as shown in Fig. 2. We consider the DNN with five hidden layers each with 300 nodes as specified in Section IV-B1.

1) The impact of problem complexity

First, the variation of the performance of the model is studied as a function of the optimization problem complexity. This is done by altering the number of subcarriers N. Consequently, the model needs to learn more complex features from the dataset. The performance is measured through the relative sum-rate Eq. (16) with a window size w equal to the total number of samples. In addition, the variation of the total training time t_2 and the prediction time per sample t_3/sample (see Fig. 7) with the number of subcarriers N are also given in Table 6.

We observe that the performance of the DNN decreases as the complexity of the problem increases proportionally to N. This is because the DNN model needs to be retrained with more data and higher complexity models to cope with the increase of problem complexity. Furthermore, the efficiency, which is measured in terms of the training time t_2, and prediction time t_3/sample also worsens as the complexity increases. This is attributed to the change in the input/output layer size of the DNN, which results in a higher number of weights. Despite this increase, the prediction time is well within the margin of the millisecond level latency requirement for this low-complex case study.

Therefore, we conclude that the performance of the DNN degrades and its efficiency worsens as the complexity of the problem increases. The performance can be improved to an acceptable level by altering training parameters such as the number of training samples and the number of training epochs, or alternatively, by modifying the DNN to learn more complex data by increasing the number of hidden layers and the number of nodes. However, this can lead to poor efficiency (i.e. longer training or prediction time, respectively). Beyond 5G applications such as VR require high reliability (performance) and low-latency(efficiency). Thus, we need a proper trade-off between the system performance and learning efficiency, and this relationship is discussed next.

[14]Unless stated otherwise, we use the same parameters for data generation of all other system models.

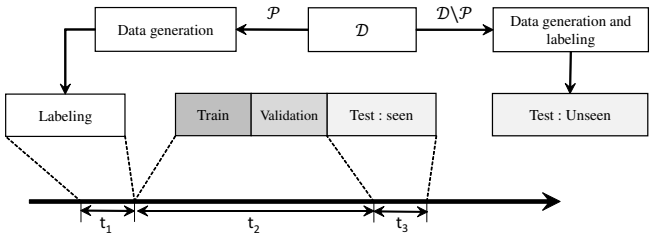


FIGURE 7: A dataset with seen and unseen $(L_t,\ M_t)$ pairs and measuring computational efficiency.

The wireless system's performance is measured in terms of the relative sum-rate given by Eq. (15) and the learning efficiency is measured in terms of the training time and the prediction time of the DNN. The time evaluation has three time steps namely, t_1 (labeling), t_2 (training and validation), and t_3 (prediction on the test dataset) as shown in Fig. 7. The main objective of learning methods must be to reduce the prediction time t_3 to match the ultra-low-latency requirement. However, in a highly dynamic wireless environment, retraining becomes vital due to model ageing. Thus the training and labeling (in case of supervised techniques) must also be considered when evaluating the efficiency of learning models. We conduct each of these simulations by keeping the structure of DNN and the dataset fixed while changing one parameter at a time.

a) Impact of the Number of Training Samples

As the first experiment, the training parameter was changed by varying the number of training samples. The variation of the performance and efficiency as a function of the training size is shown in Table 7. We observe that the relative sum-rate $\overline{R}_t$ increases with training size up to a certain point before it reaches saturation. This confirms the need for a significant amount of training data for good prediction even in relatively low complexity cases.

We also observe that the training time t_2 is extremely high compared to t_1 and t_3. This may pose limitations to learning for applications with tight latency and reliability requirements. For example, interactive virtual reality applications require a latency of about 20 ms to avoid distortion and motion sickness, and the currently available online virtual reality applications have a latency of around 100 ms which can only be used for non-interactive applications such as streaming [3]. However, the training time t_2 is in the order of minutes which indicates that online retraining becomes a challenging task and cannot be a solution to the ageing problem in real-time applications as discussed before.

We note here, that the experiments were conducted on a 16 Core i7-6700 central processing unit, which runs at 4.00 GHz and can perform nearly 40 GigaFlops [52]. State-of-the-art processors such as the Ascend 910 delivers 256 TeraFlops which is nearly 5000x times faster. Therefore,

TABLE 7: Variation of performance with the number of training samples with $k = 10$.

Training set size	t_1 (s)	t_2 (s)	t_3 (s)	$\overline{R}_t$
1K	1.25	372	4.07	87.9%
5K	1.47	534	3.31	88.4%
10K	1.83	744	3.15	90.1%
20K	2.62	1002	2.64	90.3%
50K	4.16	2424	1.58	90.6%

TABLE 8: Variation of efficiency with the number of training samples.

Training set size	t_1 (ns)/sample	t_3 (ns)/sample
1K	59	51
5K	58	44
10K	60	45
20K	62	44
50K	59	52

even though learning models in our studies are constrained by time for real-time applications, powerful AI chips coupled with fast communication technologies supported by 5G cloud-based architectures could be used to implement deep learning solutions. The practicality of utilizing such sophisticated hardware and robustness of these solutions must be studied through experimentation and application. However, as mentioned before, the processing time must be less than 2 ms for a round trip time of less than 20 ms for online applications such as virtual reality [53]. Therefore, even though the studied learning solutions could be used for non-interactive applications, it remains questionable as to whether frequent retraining can be done within a short period where the radio environment is evolving – even with the support of computationally powerful devices supported by cloud architecture.

Finally, we consider the prediction efficiency of the learning model compared to the water filling algorithm. The times t_1 and t_3 are the total times taken by the water filling algorithm and the DNN model respectively to calculate the power allocation strategy. Since the total elapsed time may not give a good indication of the performance, the time taken per sample by water filling method (t_1/sample) and the DNN model (t_3/sample) were measured as shown in Table 8. It can be observed that the DNN prediction time per sample is a few nanoseconds faster compared to the labeling time. However, we note that this observation is specific only to this simpler case study. In later, more complex case studies, we show that learning models can be highly efficient while promising near equal performance when compared to classical algorithms.

b) Impact of the Number of Hidden Layers

The complexity of the DNN was changed by varying the number of hidden layers. The efficiency and accuracy measures are indicated in Table 9. We note here that since the time t_1 remains unchanged (i.e. the labeling set is not altered), it is omitted from the table. One can observe that both the training time t_2 and the prediction time t_3 decrease as the number of layers decreases. However, the accuracy of the model decreases as well. Through these observations, we depict the inability of the DNN model to learn complex functions through the use of simple models. As the number of hidden layers increases both t_2 and t_3 increase together with the performance. However, after a certain point, t_2 and t_3 keep on increasing while the increment in the performance becomes negligible. This shows that the DNN may not be able to perfectly model a problem beyond a certain limit.

TABLE 9: The impact of the number of hidden layers.

Number of hidden layers	t_2 (s)	t_3 (s)	$\overline{R}_t$
1	564	1.51	68.2%
3	810	2.28	88.4%
5	1002	2.64	90.3%
7	1194	3.67	91.0%
9	1578	4.89	90.8%

TABLE 10: Variation of $\overline{R}_t$ on seen and unseen data with the channel non-stationarity factor k.

k	$\overline{R}_t$ of seen data	$\overline{R}_t$ of unseen data
1	99.5%	67.2%
5	93.6%	89.8%
10	92.2%	89.9%
50	92.1%	90.8%
100	91.6%	91.5%

Through these observations, we conclude that a good balance between efficiency and accuracy could be achieved by adjusting the model parameters. Globally best parameters do not usually exist and thus the model and training parameters must be optimized on a case-by-case basis through ablation studies and experimentation ensuring that it satisfies the latency and performance requirements.

2) Variation of performance with non-stationarity and the ageing effect

Next, the relative sum-rate was measured while changing the environment generators of the training dataset; (L_t, M_t) pairs. Note that in the previous section, the performance of the DNN model was evaluated based on test data which exhibits a low variation when compared to the training data. However, in reality, as the channel model varies, the characteristics of the input data would deviate from that of the training data and the performance of the DNN model would change as a result. To observe this variation, a new dataset of 100K samples was generated as shown in Fig. 7. In this case, the (L_t, M_t) pair values of the new dataset show a high variation compared to the training data. We codify these test sets as seen and unseen data.

From Table 10, we make the following observations:

- Relative sum-rate on seen data is high when the known (L_t, M_t) pairs, k is low. Due to the low non-stationarity

factor k, the data distribution is narrow and consequently, the DNN model manages to learn the behavior of the data distribution.
- As k increases, the relative sum-rate on the seen dataset reduces. While increasing the non-stationarity factor k the distribution of the data becomes broader and naturally, the DNN model requires a large set of samples and more parameters to learn the behavior of the dataset. However, the number of samples and the DNN architecture is kept constant, thus the overall performance reduces.
- The unseen dataset shows an opposite trend, as the relative sum-rate increases with k. This is because the DNN model generalizes the data distribution with more diverse samples. Therefore, we can see an increase in relative sum-rate with unseen data with larger k.
- The relative sum-rate of DNN output on both seen and unseen datasets saturates and becomes equal after a certain k value (around 1% out of 4096). This is because the DNN model reaches its capacity to learn from the given data due to the limitation of the parameters in the DNN model.

The observations listed above can be explained through "overfitting". In Table 10, the parameter k refers to the degree of non-stationarity of the channel data, i.e. variation/randomness in the dataset. Thus, the "seen data" shows a similar pattern to the training set whereas "unseen data" shows a high variation in comparison. For a lower k, the model easily learns the patterns in the training data due to its low variation and "overfits" on the training data. This can be observed through the high performance on seen data (which shows a similar pattern as the training set) but a poor performance on the unseen data. As k increases the non-stationarity variation in the training data increases, and as a result, overfitting is minimized during training. This can be observed with the decrease in performance on the "seen data" with an increase in k. However, since the trained model is not overfitted, the performance of the model on the "unseen data" increases due to "generalization".

The degradation of performance in supervised learning models due to model ageing as discussed in Section V-A also occurs due to the overfitting. In that case, the supervised model (DNN) is initially trained based on training data which has a particular pattern. Initially, this pattern exists in the test dataset as well and thus the trained DNN model performs well on this test set. However, with time as the features of the test data change due to the user movement (i.e., non-stationarity of the channel), the pattern which was initially observed in the training set is not repeated anymore. As a result, the performance of the DNN degrades on this new test set with high variation. Thus, the performance degradation of DNN with time occurs due to the initial overfitting and the high variation which occurs with time.

Now let us combined these ideas and analyze the ageing effect on models trained under different k values with time. The model with high overfitting ($k = 1$) performs well initially (99%), but its performance degrades drastically over time (68%) whereas the model with the least overfitting ($k = 100$) has a comparatively lower (91%) but stable performance. Therefore, the degradation can be reduced if overfitting is minimized. However, even though this degradation is small and could be overcome through generalization in this case study, we highlight the following issues related to generalization:

- Generalization usually results in lower performance than if the model was trained for a specific environment. Due to the low complexity of this case study, the difference in performance (99% vs 91%) is low. However, as the complexity of increases, this difference becomes significant. This is discussed in a later section where the performance of the model degrades significantly with the complexity. (See Fig. 13)
- In these studies, we exert a boundary/limit, within which the channel parameters ($L = 1 \sim 32$, $M = 1 \sim 128$) may vary. However, this assumption may not be extended to real-life cases where the system may vary a lot. Thus, in reality, generalization would require data with high variance obtained under different conditions and this is a challenge.

Due to these reasons, we observe a significant degradation of performance in a non-stationary environment as the complexity of the problem increases.

C. MODERATE COMPLEXITY PROBLEM

In this section, we study 1) Multi-user single-cell OFDM and 2) Multi-user multi-cell single-carrier communication scenarios.

1) Multi-user single-cell OFDM

We assume a single-cell OFDM system with N = 32 subcarriers and U = 4 users as specified in Section III-B1. We assume that there is no interference if the BS does not allocate the same subcarrier to multiple users. A dataset is generated with the non-stationarity factor $k = 10$, then labeled using the greedy optimization strategy in [41] and split as shown in Fig. 2.

As explained in Section III-B1, each subcarrier is allocated to a unique user and thus, all the other subcarriers have zero power allocation at that time. This leads to a highly biased and sparse set P_t. Due to this sparsity, a single DNN model cannot be employed. In such a scenario the DNN "adapts" to predict the zeros in the output labels. Thus, the converged model predicts values close to zero which is not optimal. If we represent sparse data in an n-dimensional vector space, they are not clustered closely together. This makes the learning process of the DNN inefficient. To address this problem, we propose a design with two separate sequential DNN models for each sub-problem (subcarrier and power allocation) as illustrated in Fig. 8. DNN model A predicts the subcarrier allocation algorithm, while the DNN model B predicts the power allocation algorithm (water filling).

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2020.3040133, IEEE Access

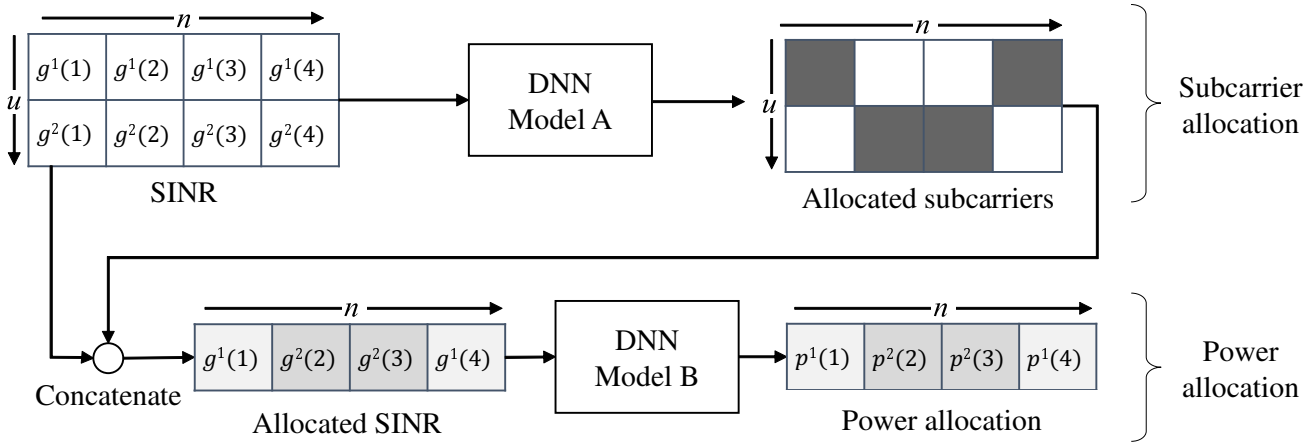


FIGURE 8: DNN model for single-cell OFDM case.

- The DNN Model A predicts subcarrier allocation and has a different architecture. The DNN has one hidden layer with 200 nodes. The input layer has $(N \times U)$ nodes with each node representing the normalized channel gain $g_b^u(n)$. The output layer has $(N \times U)$ nodes with each node representing the binary value for the subcarrier allocation. A different approach could be to use lower granularity of CSI such as received signal strength indicator that would impact the performance while the methodology would not change. Each layer uses ReLU as the activation function except for the last layer which uses a Sigmoid function. The problem is a multi-class multi-label problem where the users are assumed to be the classes and each user can be labeled with multiple subcarriers. Therefore, binary cross-entropy loss function was chosen for the DNN [13]. The output refers to the subcarrier allocation: A true value would mean that a subcarrier is allocated to that user and if the value is false, then it is not allocated.
- Model B predicts the power allocation. Since the labeling algorithm (water filling) for this subproblem is the same as in Section V-B, we use the DNN model with the same architecture.

The combined DNN was evaluated by calculating the average value of the relative sum-rate given by Eq. (15) where $\hat{r}_t$ is the sum-rate obtained from the DNN output and r_t denotes the optimal sum-rate obtained through the greedy optimization approach described in [41].

We observed the variation of the relative sum-rate while changing different parameters. Initially, the non-stationarity was changed to see its effect in a different system setup. Next, we change the problem complexity with a specific focus on studying the effect of user activity. Finally, we modify the model architecture and study the effect of incorporating memory models to handle the ageing problem.

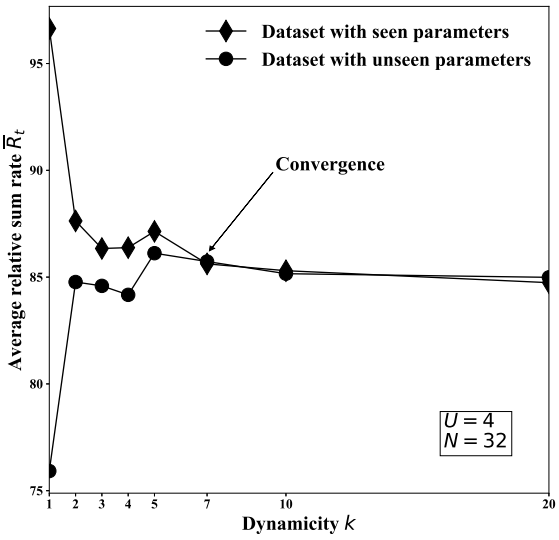


FIGURE 9: Variation of average relative sum-rate with non-stationarity on seen and unseen datasets.

a) Variation of performance with non-stationarity

The variation of the average relative sum-rate $\overline{R}_t$ with the non-stationarity k is shown in Fig. 9. Two examples of datasets with seen and unseen $(L_t,\ M_t)$ pair values were generated as shown in Fig. 7. The average relative sum-rate $\overline{R}_t$ is low for the unseen data with lower values of non-stationarity factor k. This is because the DNN is trained with a dataset with samples from a near non-stationary environment and thus is overfit and biased. Thus, when the model predicts the power allocation for a data sample which is from an unseen environment with high movement, the performance is poor. However, as the non-stationarity factor k of the training set increases, the DNN model generalizes the behavior of the mobile radio (i.e. non-stationary) environment and thus, the

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2020.3040133, IEEE Access

average relative sum-rate $\overline{R}_t$ of seen and unseen datasets converge. 5G and beyond targets a highly dynamic wireless communication (e.g. users in vehicles, industrial IoT devices, etc). Therefore, it is critical to do comprehensive study on how algorithms behave in dynamic conditions as mentioned above.

b) Impact of the non-deterministic user activity

Next, we analyze the effect of varying problem complexity. We observed that increasing the complexity by changing parameters such as N and U results in a lower performance similar to Section V-B. However, we note that in reality, parameters such as B and N remain fixed during the operation of a system, but the number of users U could change depending on user movements. When the total number of users U changes, the size of the input vector changes as well. In a fully connected DNN, the layer sizes cannot be changed dynamically. Thus, if U changes, a new DNN would have to be trained. In such a case multiple trained DNNs would be required for continuous operation. If we arrange the channel gain information in a 2D matrix, with users in one dimension and subcarriers in the other, we can assume that each element in the matrix is spatially co-related. This spatial correlation can be modeled through convolution kernels. In a later section, we propose a Convolution Neural Network (CNN) to estimate the power allocation instead of a DNN to tackle the problem of non-deterministic user activity (See Section V-D).

c) Recurrent Neural Networks with LSTM cells to study limitations associated with non-stationarity

The ageing problem of supervised learning models in a non-stationary environment occurs due to the dynamic nature of the observed data. This time-variant property can be modeled using memory elements. To study the effect of such networks the DNN was modified by incorporating Long short-term memory (LSTM) cells. LSTM cells are a type of recurrent neural networks (RNN) which use "memory cells" to maintain the state information for a longer period. LSTM cells use complex architecture, which overcomes issues such as vanishing and exploding gradient which are seen in RNNs with simple feedback loops [54].

The DNN model B (which predicts the power allocation) was modified by adding an LSTM layer at the input layer. Moreover, RNN models require historic data in addition to current data. Thus, the dataset was modified such that each data sample included the channel gain values of the previous 10 time-steps. Our results indicated that after the update of the model with the LSTM cells a similar trend is observed as illustrated by Fig. 6a. This occurs since the training data still only contains the stationary channel values and the RNN model is biased towards this training data which results in poor performance in a non-stationary environment. We concluded that the ageing problem occurs due to the training process. The dependency of supervised learning models on the training data leads to the ageing effect, and this cannot be rectified by modifying the Neural Network architecture.

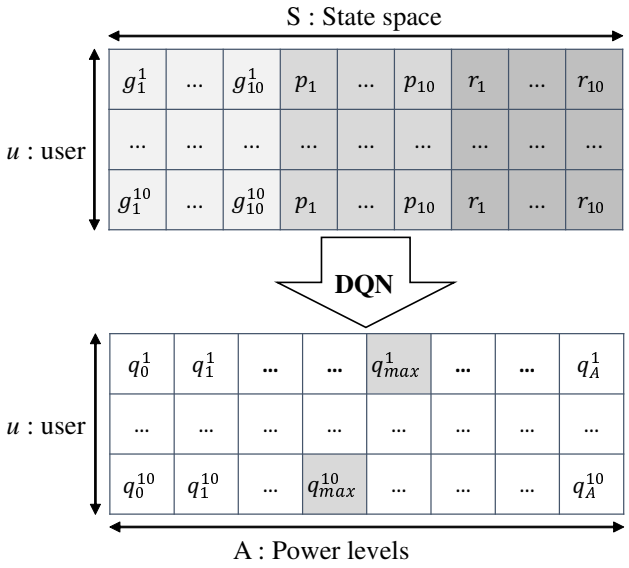


FIGURE 10: State and Action Space of proposed DQN implementation for Multi-cell Single-carrier system.

2) Multi-user multi-cell single-carrier

We consider a single carrier system with B BSs and U users that are randomly located as specified in Section III-B2. In reality, the bth BS allocates time slots to each user $u \in \mathcal{U}_b$ in a weighted manner in order to maintain fairness. For simplicity, we assume that the bth BS always connects to a particular user u and decides the power allocation to minimize interference. Note that due to this assumption, $B = U$ in this case study. However, in the high complexity case, we remove this assumption and consider a more realistic system.

As discussed in Section V-A, the ageing problem of the DNN makes it unsuitable under non-stationary channel conditions. However, when trained till saturation, the DNN was able to perform better in comparison as shown in Fig. 6c. In this section, we extend the study further to find if this observation holds. Thus, we compare the performance of both the DNN and DQN models under different conditions. For the DQN model, we employ a similar methodology as in [27], [28] with minor changes to the model architecture and the training parameters. While these works focus on the optimal converged performance, our study focuses on the variation of performance during training under different problem complexities and analyses the reason for any variation of performance during the training phase.

A dataset is generated by Eq. (1) with $L_t = 1$. In this dataset, a single sample represents $\mathbf{G}_t$; the set of channel gain values $\{g_1^1, \ldots, g_B^U\}$ between all the users and BSs for a given time t. The data are labeled using the WMMSE algorithm [42] and split as shown in Fig. 2.

a) DNN model approach

We utilize a DNN with 5 hidden layers, each containing 300 nodes as described in Section IV-B1. The DNN takes the g_b^u values as the input and output the corresponding power allocation for each user p_b^u. In a single carrier system, the power violation is not an issue since the BS has only one carrier and there is no possibility of allocating power higher than the maximum power allocation. Therefore, we omit the regularization part in Eq. (12) and thus. the loss function is the MSE.

b) DQN model approach

The DQN implementation requires the problem to be converted to an MDP as specified in Section IV-B2. Since this is a single carrier system, the second MDP formulation proposed in Section IV-B2 is used. In the proposed DQN design, at time t, the channel gain values are represented by $\mathbf{G}_t = \{g_1^1, \ldots, g_B^U\}$ and the power allocation decisions at each step i are represented by $\mathbf{P}_t(i) = \{p_1, \ldots, p_B\}$ for all the BSs as illustrated in Fig. 4. Furthermore, $\mathbf{R}_t(i)$ represents all the corresponding sum-rates $\{r_1, \ldots, r_B\}$ for all the BSs at step i. Using these terms, for this case study we define the state space, action space and reward as follows:

- *State space* - The state space for a U = 10 system has the values concatenated/stacked together is shown in Fig. 10. It is composed of the current channel gain values of each user and interfering users $\mathbf{G}_t$, previous power allocation for each user $\mathbf{P}_t(i-1)$, and previous rate of each user $\mathbf{R}_t(i-1)$ stacked together. We can use this state space to predict an action that will estimate the power levels of all BSs. But estimating the power levels of all the users simultaneously will require an action space of size A^U. This is not practical for DQNs since the action space increases exponentially. Therefore, in a single step i, we observe the state of a single user (i.e. a row in state space as shown in Fig. 10). Let's call this the observable state. Using this observable state, we predict the BS power level for that particular user using the DQN. Similarly, we predict the power level for all the users in step i and after that, we proceed to step $i+1$. We can reduce the size of state space to $U \times 3$ and action space to A through this methodology.
- *Action Space* - The action space of a DQN is finite. However, since the power allocations in our case can have infinite values, the total power was divided into A+1 discrete power levels as $\mathcal{A} = \{(j \times P_{max})/A; 0 \leq j \leq A\}$. Fig. 10 shows the action space for U = 10 users.
- *Reward* - We use the sum-rate which is given by Eq. (7) as the reward for the DQN. Note that this also corresponds to the summation of individual rewards $R_t(i)$.

Finally, the value estimator of the DQN as illustrated in Fig. 4 is a fully connected neural network. It has 3 hidden layers each with 128 nodes. The input layer has (U x 3) nodes. The output layer has A nodes. Each layer uses ReLU as the activation function except for the last layer which uses the linear function. The DQN uses the MSE as the loss and the model is trained using the epsilon greedy policy with a vanishing epsilon value from 0.8 to 0.01 as specified in Table 5.

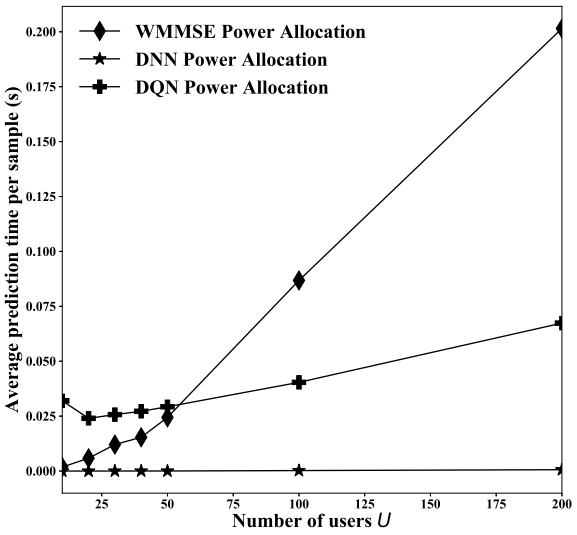


FIGURE 11: Variation of prediction time with number of users.

Our discussions are centered around the two metrics: the relative performance and the prediction efficiency of learning models. The relative performance is measured using the relative sum-rate given by Eq. (15), where $\hat{r}_t$ denotes the sum-rate calculated using the DQN/DNN models, while r_t denotes the sum-rate given by the WMMSE algorithm. The efficiency is defined by the time taken to predict the power allocation of a unit sample. The variation of the relative performance and the efficiency with the number of users is shown in Fig. 13 and Fig. 11 respectively.

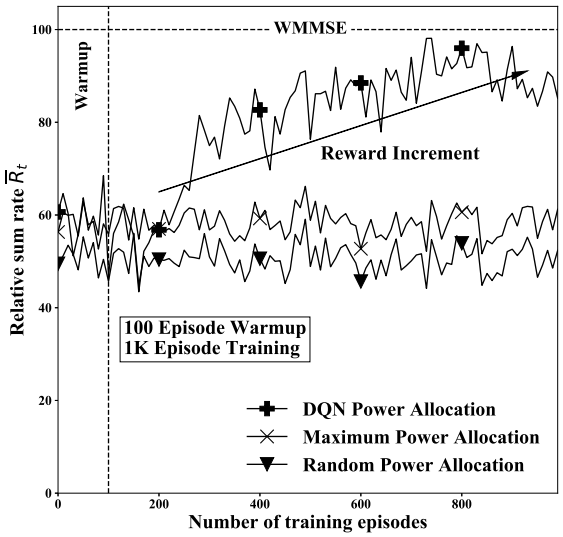
c) Efficiency of learning models

As shown in Fig. 11, the prediction time increases as the complexity of the problem increases. The increment in the prediction time of the DNN is negligible whereas the increment in the DQN model is slightly higher, and the increment of WMMSE time is exponential. This exponential increase in the prediction time of WMMSE algorithms can be attributed to its iterative optimization technique to find the solution. Therefore, we conclude that WMMSE is very inefficient compared to other models.

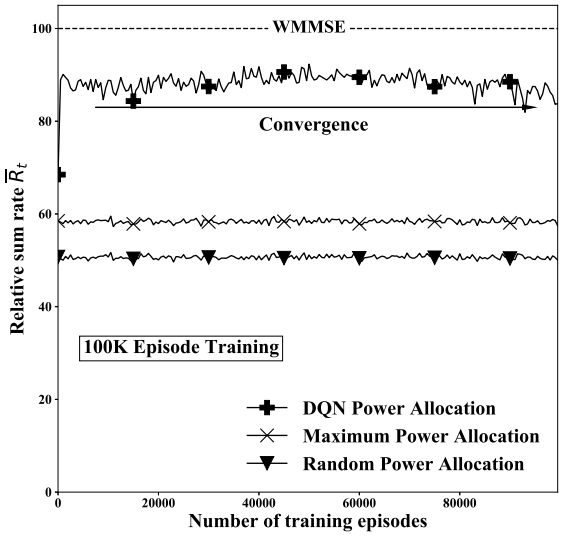
Note

The DNN takes comparatively less time to predict the power allocation after training the model since the prediction time depends only on the number of parameters in the DNN model. However, since DNNs require a labeled dataset to be trained, the training time of DNNs would be increased due to

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2020.3040133, IEEE Access



(a) First 1K episodes.



(b) 100K episodes.

FIGURE 12: Variation of Relative sum-rate of DQN throughout the training.

the inefficiency of the WMMSE algorithm (since the dataset would have to be labeled first by WMMSE). Therefore, DQNs act as suitable alternatives since they do not require to be pre-trained as DNNs, and the increment in the prediction time with problem complexity is low compared to WMMSE algorithm. Due to these two reasons, DQNs are preferred for low-latency application. However, the prediction time per sample for this case study ($\sim 25\,ms$) is much higher than the millisecond latency required by 6G applications. Therefore,

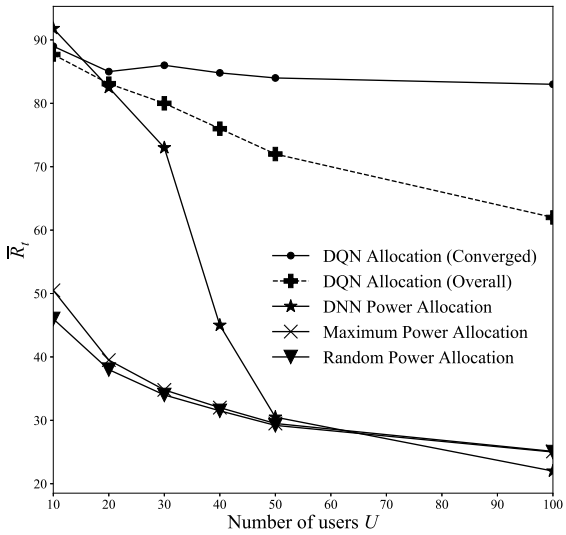


FIGURE 13: Variation of relative sum-rate $\overline{R}_t$ with number of users U.

real life application would require computationally powerful devices and optimized models to achieve the latency requirements.

d) Relative sum-rate of DQN during training

First, we study the variation of the performance of the DQN throughout the training process and compare it with the optimal performance given by the WMMSE algorithm. Furthermore, the reward (sum-rate) from random power allocation and maximum power allocation strategies are also considered for comparative study[15]. The variation of the reward throughout the training process is shown in Fig. 12a and Fig. 12b. "Warm-up" phase is defined by the first 100 episodes followed by the training phase. Note that during the warm-up phase, the relative performance of the DQN is nearly equal to the random power allocation method. However, after the saturation (convergence) phase the relative performance of the DQN is much higher than the random and maximum power allocation ratios. This confirms that the DQN has been adequately trained.

We note here that the DQN is initially untrained and would perform poorly compared to the DNN and WMMSE algorithms. Therefore, when the overall average performance of the DQN is compared with other models, this would also include the warm-up phase and the reward increment phase which have lower performance compared to the saturation value. So, in order to compare the optimal performance, we consider two performance metrics for the DQN: The overall

[15]In random power allocation strategy, power is allocated to each user randomly. In the maximum power allocation strategy, the maximum power of each BS (P_{max}) is assigned

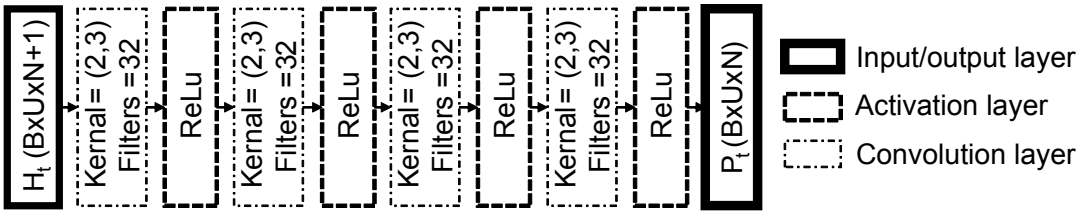


FIGURE 14: Proposed DNN model to predict power allocation of multi-cell OFDM case.

DQN performance and the converged DQN performance. The overall DQN performance was measured by considering the time-averaged relative sum-rate till convergence. The converged DQN performance was measured using the time-averaged relative sum-rate on a new dataset with $epsilon$=0 after training the DQN till convergence.

e) Relative sum-rate of learning models

Fig. 13 shows the overall and converged relative performance of the DQN together with DNN, random power, and maximum power allocation techniques. From the figure, we observe that, as the number of users increases, the performance of both these models degrade. This is attributed to the increase in problem complexity with the number of users. However, the degradation of the performance of DNN is significantly higher. This occurs since the DNN is a supervised learning model, and as such requires more training data and higher complexity model. However, the DNN model has the same complexity and thus may not be able to model a more complex problem. A complex DNN model would be required as the problem complexity increases. However, as discussed in the previous section, this would make the DNN inefficient and would not be a suitable solution.

As opposed to DNNs, DQNs show low degradation of performance as the problem complexity (number of users) increases. Further, we observe a performance gap between the overall performance and converged performance of the DQN. This occurs due to the initial low performance of DQN during the training phase as shown in Fig. 12a. We note that the overall performance degrades more in comparison to the converged performance as the problem complexity increases. This occurs since the DQN model requires more time to converge (higher number of training episodes) as the problem complexity increases resulting in a longer training phase and thereby a lower overall performance. This would be a major concern for high complexity problems since convergence takes time and cannot be guaranteed in a highly mobile environment as shown in Fig. 6c.

These results indicate that the performance of the DNN, in terms of both the time consumption and the relative sum-rate, degrades as the complexity of the problem increases. On the other hand, even though the WMMSE algorithm provides optimal performance, it has poor efficiency at higher complexity cases. Thus, we conclude that the proposed DQN model would be more suitable for this case study since it provides a good trade-off between performance and efficiency when compared to the iterative optimal algorithms.

We note here that the application of this model is limited to smaller systems, and extending it to larger systems, considering information from all BSs requires a large observation to the DQN. In addition, DQN predicts a single action from a predefined discrete set of actions leading to unmanageable action space. Thus, extending this model to real life beyond 5G applications which consist of larger networks with multiple subcarriers requires further studies. In what follows, we propose a DNN based unsupervised learning method for resource allocation in large realistic network scenario.

D. HIGH COMPLEXITY PROBLEM

We assume a multi-user multi-cell OFDM system with a varying number of subcarriers N having B BSs and U randomly located users. In this scenario we first consider a non-stationary system. The data generation methodology from the single-cell OFDM case study is extended to multi-cell scenario. That is, we calculate the channel gain for each subcarrier between a given BS and all users in the wireless network (including users in other BSs). Then we have a 2D matrix of channel gain information $\mathbf{G}_t$ of size $U \times N$ and calculate the channel gain for all the BSs, stack them to form a 3D matrix $\mathbf{G}_t^3$ of size $B \times U \times N$. Each user is assumed to be connected to the closest BS. The user's connectivity is represented in an adjacency matrix $\mathbf{A}_t$ of size $B \times U$, where element $\mathbf{A}_{t,b,u}$ = 1 if user u is connected to the BS b and 0 otherwise. The dataset consists of 1000 samples and each sample consists of channel gain information ($\mathbf{G}_t^3$) and BS-user connectivity information ($\mathbf{A}_t$) for a given time t.

For each user in the system we need to predict two variables: (i) power allocation, and (ii) subcarrier allocation. In this case study, we define a single DNN model to predict power allocation $\mathbf{P}_t$ and assume subcarriers are allocated equally to all users. Current mobile systems use a distributed system where each cell's decisions are taken independently of their neighbor's CSI. However, in 5G New radio, the Central Unit and Distributed Unit functional split architecture allows for coordination for performance features, load management, real-time performance optimization.

We propose a DNN model that inputs $\mathbf{G}_t^3$ and $\mathbf{A}_t$ from the input layer as illustrated in Fig. 14. We concatenate $\mathbf{A}_t$ along the dimension of the subcarriers in $\mathbf{G}_t^3$ to construct the 3D input matrix. Thus, the size of the input matrix is $B \times U \times (N+1)$. The proposed DNN model consists of four convolution layers with kernel size $(2, 3)$ and 32 filters. Each

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2020.3040133, IEEE Access

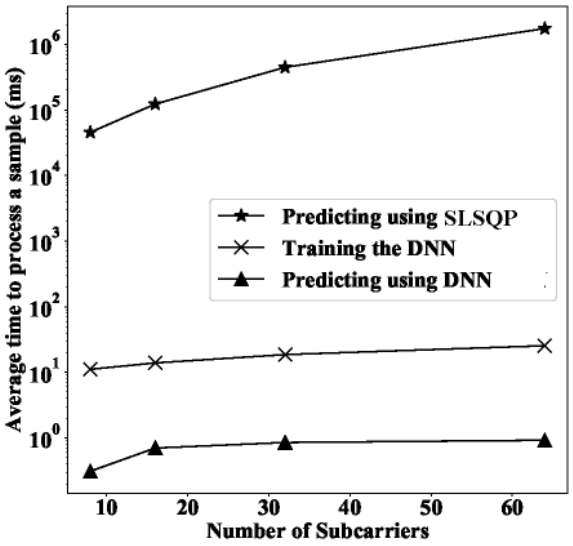


FIGURE 15: Variation of labeling, training, and prediction time with the number of subcarriers.

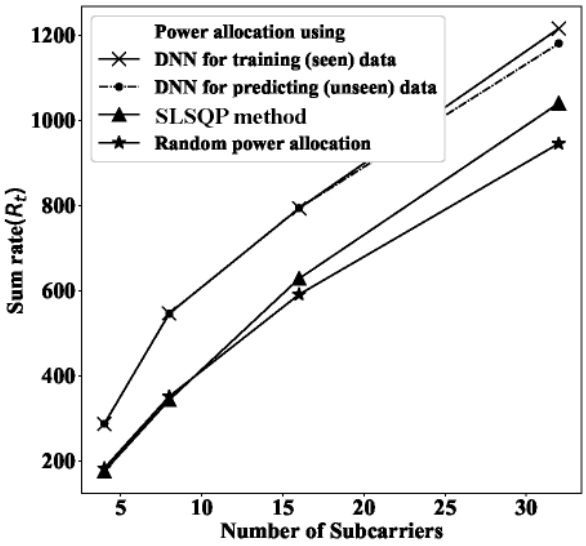


FIGURE 16: Variation of average sum-rate with the number of subcarriers.

layer is passed through a ReLu activation function to make the input-output correlation non-linear. The model predicts power allocation $\mathbf{P}_t$ from the output layer as shown in Fig. 14. We propose an unsupervised DNN model with a loss function that does not require labeled data to train. The loss function consists of the sum-rate and the power regulation constraint introduced in Eq. (12). The loss function is given as,

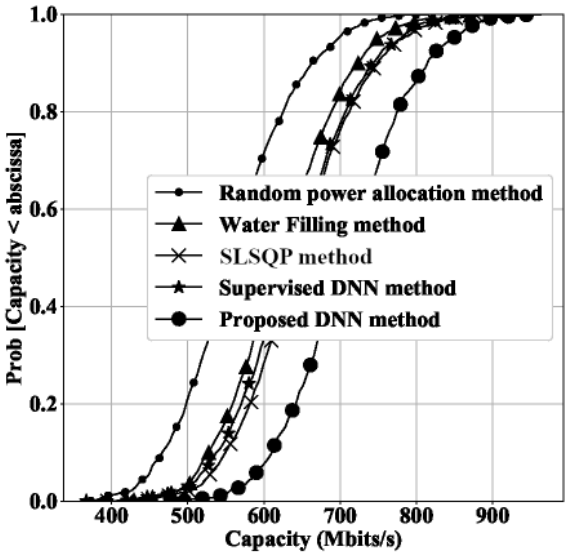
$$\Upsilon = \sum_{b\in\mathcal{B}} \left(- \sum_{u\in\mathcal{U}_b} \sum_{n\in\mathcal{N}} log_2[1+\gamma_b^u(n)] + \beta \left| \left(\sum_{u\in\mathcal{U}_b} \sum_{n\in\mathcal{N}} \hat{p}_b^u(n) \right) - P_{max} \right|^2 \right) \tag{17}$$

where $\gamma_b^u(n)$ is given by Eq. (11). First, we measure the variation of sum-rate and prediction time with the number of subcarriers through different techniques. Then, we analyze the distribution of power, sum-rate, and user-rate. The proposed model is compared with the suboptimal SLSQP solution [44] and the random power allocation strategy. We also consider the greedy water filling algorithm from Section V-C1 by applying it to each BS independently. In addition, a supervised model with similar architecture as the proposed unsupervised model, but with the loss function proposed in Eq. (12) is also considered.

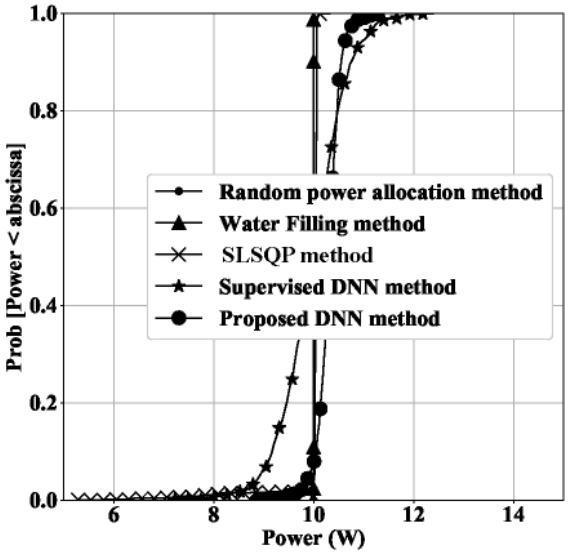
1) Variation of efficiency of the proposed DNN model

We consider a wireless system with $B = 2$ and $U = 4$, while the number of subcarriers N varies from 16 to 64. A relatively low number of users and BSs were chosen for this comparative study since the suboptimal algorithm chosen as a benchmark takes a long time to calculate the power allocation for large datasets. In Fig. 15 consider the case where there exist $N = 16$ subcarriers. DNN takes around 0.7 ms to predict the power allocation per sample in the dataset, while it takes around two minutes per sample with SLSQP solution [44]. The dataset consists of 1000 samples. Thus, for the bench-marking process itself takes around $2\,\text{min} \times 1000 \approx 33.3$ hours. This is because [44] takes multiple iterations to converge to a computationally exhaustive solution. To train the DNN, it takes around $14\,\text{ms} \times 1000 = 14\,\text{s}$ per epoch and a total training time of $14\,\text{s} \times 100 = 1400\,\text{s}$. 6G applications aim at sub-msec latency. The proposed unsupervised method predicts the power allocation several orders less than the suboptimal solution and it is close to the expected latency requirement of 6G applications. In addition, since the unsupervised learning model is independent of the labeling algorithm, the retraining time required in non-stationary environment is significantly lower than that of supervised models. However retraining is still in the order of minutes and thus the usage of DNNs are detrimental in highly dynamic wireless environment which require constant retraining.

When the number of subcarriers increases, the time taken for the suboptimal SLSQP algorithm [44] increases exponentially as illustrated in Fig. 15. This is because the SLSQP algorithm in [44] takes more time to compute each iteration because of the large vector representing the state. This implies that when the number of BSs and users increases, the time taken for [44] increases exponentially as well. For example, in this wireless system, it takes around 1 minute to predict power allocation using a sample from the dataset for a given time t. For the system with $N = 64$ subcarriers



(a) Maximum sum-rate.



(b) Total power allocation.

FIGURE 17: Cumulative probability distribution of sum-rate and total power allocation for each BS.

having $B = 2$ BSs and $U = 16$ randomly located users, it takes around 1 hour to predict a sample. Thus, in order to predict the complete dataset, it will take 1000 hours or roughly 41 days. As a result, using a linear programming methods [44] for large wireless systems with multiple BSs and/or crowded areas is not practical because of the long time for computation.

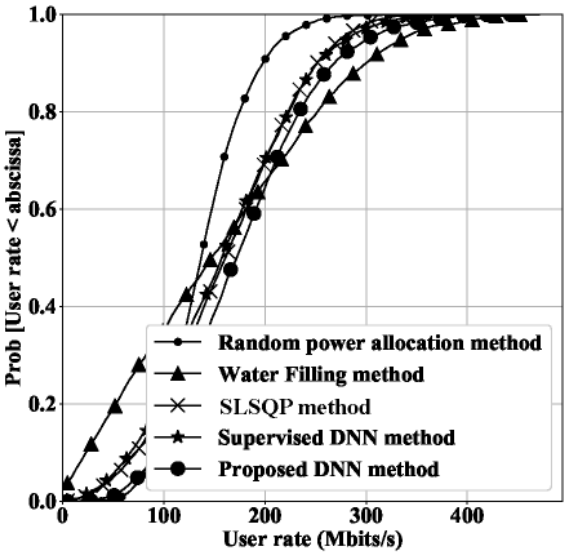


FIGURE 18: Cumulative probability distribution of sum-rate of each user in a BS.

2) Variation of average sum-rate with the number of subcarriers

Fig. 16 illustrates the variation of average sum-rate in a multi-user multi-cell OFDM system with $B = 2$ and $U = 4$, while the number of subcarriers N varies from 4 to 32. We also compare the average sum-rate of the proposed model on the seen data and unseen data (See Fig. 7). When the number of subcarriers increases the average sum-rate increases in all methods since we keep the bandwidth of a subcarrier constant. Therefore, larger the number of subcarriers, higher bandwidth to transmit data. But it is clear that the proposed method gives a higher sum-rate regardless of the number of subcarriers. Furthermore, when considering unseen data, we observe a drop in performance in the proposed method as the number of subcarriers increases (N=32). This occurs since the model's ability to generalize decreases as the model as the problem complexity increases. As a result, the performance on unseen data decreases with problem complexity.

Next, we compare the cumulative sum of the probability distribution of sum-rate and total power from different algorithms as shown in Fig. 17. The proposed model improves the sum-rate in comparison to other methods. This is because the proposed model uses an unsupervised learning technique that considers all CSI from all BSs and the inter-cell interference through the loss function. Furthermore, the proposed loss function penalizes high power allocation when power violation occurs, and as a result, the proposed model violates the total power allocation constraint less frequently in comparison as given in Fig. 17b.

Fig. 18 illustrates how the sum-rate of each user is distributed with different power allocation algorithms. We ob-

serve that the water filling algorithm shows a high variation of user rate, while the proposed algorithm has a lower variation and higher average user rate in comparison. We conclude that the proposed method gives a higher rate for all the users, which results in a higher sum-rate of the entire network.

Note

Reinforcement learning through the proposed DQN model becomes a challenge in this high complexity case study due to the exponential increase of the state and action spaces. An increase in the problem complexity leads to an unmanageable action space of the DQN. However, instead of using DQNs, policy gradient optimization reinforcement learning approaches that can estimate multiple actions in a single step can be used. In addition to overcoming the limitations related to finite action space, this can also reduce the number of steps required to estimate a solution (or simply complete an episode). Furthermore, dimensionality reduction techniques such as auto-encoder can be used to manage the increased complexity. Thereby, the reinforcement learning approaches would have a comparatively smaller observation space as opposed to the current model which uses the whole CSI space as an observation. In addition, the proposed unsupervised learning model uses a loss function that does not rely on a labeled dataset. This can be used as the reward function in reinforcement learning techniques to train the model optimally. However, these type of reinforcement learning methods are hard to train and pose many challenges. Thus, their performance and limitations are interesting future works.

VI. CONCLUSION

This paper presented a comprehensive comparison study on practical design limitations for resource management of learning in a non-stationary radio environment. We study the sum-rate maximization problem in multiple network scenarios with different complexities.

Through our case studies, we observe specific limitations of learning models such as power violation, sparse output, performance degradation with non-deterministic user activity, etc. We analyzed and proposed solutions to these issues and such limitations make the applicability of learning methods for wireless physical layer problems challenging tasks. However, two prominent issues that exist commonly for many wireless applications are the ageing effect and the high computational complexity. Through our studies, we showed that the ageing effect results in poor performance in highly complex and highly mobile radio environments. Furthermore, the low latency requirements for beyond 5G applications combined with the high computational requirement to train learning models act as a limitation for the application of learning models for interactive applications. To this end, we highlighted the importance of a generalized unbiased dataset for efficient training of learning models and show that a trade-off between the computational efficiency and prediction accuracy can be balanced with the proper choice and design of the learning architecture. While we show that reinforcement learning techniques provide a good balance between performance and efficiency while handling the ageing problem effectively, we also highlight that contemporary learning methods are limited to the applications for non-interactive services.

Finally, we observed that the drop in performance under rapidly changing, fast-fading conditions was insignificant in the low complexity case study. On the other hand, increasing the problem complexity to observe the issues that arise under fast-fading conditions is not viable through DQNs as they pose many limitations and challenges when applied to high complexity problems. Thus, the effect of fast-fading condition must be studied using a different reinforcement learning model which could handle high complexity. In order to achieve optimal performance in a rapidly varying environment, faster convergence is required and this can be achieved by changing the problem formulation or different components and hyperparameters of the learning model such as the learning policy, the actor-critic algorithms, and the policy optimization algorithms [55]. The effect of fast-fading nature on different performance-related issues such as optimal performance, convergence rate, etc. based on these modifications will be addressed as a separate comprehensive study in the future.

In addition, the low-latency and high reliability requirements of interactive 6G applications raise challenges for the application of deep learning techniques. Thus, further studies must be conducted to improve the efficiency of reinforcement learning techniques and ensure robustness in highly mobile radio environments. Modification such as dimensionality reduction (autoencoders), training optimization [56], and policy gradient optimization [57] could be used to handle the shortcoming observed in reinforcement learning and this raises interesting questions regarding the training challenges and the applications for real-world problems which should be addressed in the future. The last point is energy efficiency of the future wireless with learning. It is not clear what will be the limitations of signal processing and computational resources at the mobile terminals and how this will limit learning capability.

REFERENCES

[1] F. Tariq, M. R. A. Khandaker, K. Wong, M. A. Imran, M. Bennis and M. Debbah, "A Speculative Study on 6G," IEEE Wireless Communications, Vol. 27, No. 4, pp. 118-125, Aug. 2020.

[2] Z. Zhang, Y. Xiao, Z. Ma, M. Xiao, Z. Ding, X. Lei, G. K. Karagiannidis and P. Fan, "6G Wireless Networks: Vision, Requirements, Architecture, and Key Technologies," IEEE Vehicular Technology Magazine, Vol. 14, No. 3, pp. 28-41, Sept. 2019.

[3] M. S. Elbamby, C. Perfecto, M. Bennis and K. Doppler, "Toward low-latency and ultra-reliable virtual reality," IEEE Network, Vol. 32, No. 2, pp.78-84, 2018.

[4] H. Gacanin and M. Di Renzo, "Wireless 2.0: Towards an Intelligent Radio Environment Empowered by Reconfigurable Meta-Surfaces and Artificial Intelligence," IEEE Vehicular Technology Magazine, Special Section, 2020.

[5] W. Saad, M. Bennis and M. Chen, "A Vision of 6G Wireless Systems: Applications, Trends, Technologies, and Open Research Problems," IEEE Network, Vol. 34, No. 3, pp. 134-142, May/June 2020.

[6] H. Gacanin, "Autonomous Wireless Systems with Artificial Intelligence: A Knowledge Management Perspective," IEEE Vehicular Technology Magazine, Special issue on 6G: What is Next?, Vol. 14, No. 3, pp. 51-59, Sept. 2019.

[7] I. F. Akyildiz, A. Kak and S. Nie, "6G and Beyond: The Future of Wireless Communications Systems," IEEE Access, Vol. 8, pp. 133995-134030, 2020.

[8] H. Gacanin and M. Wagner, "Artificial Intelligence Paradigm for Customer Experience Management in Next-Generation Networks: Challenges and Perspectives," IEEE Network Magazine, Vol. 33, No. 2, March/April 2019.

[9] N. Kato, B. Mao, F. Tang, Y. Kawamoto and J. Liu, "Ten Challenges in Advancing Machine Learning Technologies toward 6G," IEEE Wireless Communications, Vol. 27, No. 3, pp. 96-103, June 2020.

[10] G. Gui, M. Liu, F. Tang, N. Kato and F. Adachi, "6G: Opening New Horizons for Integration of Comfort, Security and Intelligence," IEEE Wireless Communications, 2020.

[11] C. Zhang, P. Patras and H. Haddadi, "Deep learning in mobile and wireless networking: A survey," IEEE Communications Surveys & Tutorials, Vol. 23, No. 3, pp. 2224 - 2287, March 2019.

[12] Y. LeCun, Y. Bengio and G. Hinton, "Deep learning," Nature, Vol. 521, No. 7553, pp. 436-444, May 2015.

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

[14] H. Sun, X. Chen, Q. Shi, M. Hong, X. Fu and N. D. Sidiropoulos, "Learning to Optimize: Training Deep Neural Networks for Interference Management," IEEE Transactions on Signal Processing, Vol. 66, No. 20, pp. 5438-5453, Oct. 2018.

[15] U. Challita, L. Dong and W. Saad, "Proactive Resource Management for LTE in Unlicensed Spectrum: A Deep Learning Perspective," IEEE Transactions on Wireless Communications, Vol. 17, No. 7, pp. 4674-4689, July 2018.

[16] F. Zhou, X. Zhang, R. Q. Hu, A. Papathanassiou and W. Meng, "Resource Allocation Based on Deep Neural Networks for Cognitive Radio Networks," 2018 IEEE/CIC International Conference on Communications in China (ICCC), pp 40-45, Beijing, China, Aug. 2018.

[17] S. F. Li, M. Y. Jiang, A. M. Dong and D. F. Yuan, "Adaptive Subcarrier, Bit and Power Allocation Based on Hopfield Neural Network for Multiuser OFDM," Applied Mechanics and Materials, Vol. 325-326, pp. 1706-1711, June 2013.

[18] S. Tseng, Y. Chen, C. Tsai and W. Tsai, "Deep-Learning-Aided Cross-Layer Resource Allocation of OFDMA/NOMA Video Communication Systems," IEEE Access, Vol. 7, pp. 157730-157740, 2019.

[19] Y. Shen, Y. Shi, J. Zhang and K. Letaief, "LORM: Learning to Optimize for Resource Management in Wireless Networks with Few Training Samples," IEEE Transactions on Wireless Communications, Vol. 19, No. 1, pp. 665-679, Jan. 2020.

[20] A. Zappone, L. Sanguinetti and M. Debbah, "User Association and Load Balancing for Massive MIMO through Deep Learning," 52nd Asilomar Conference on Signals, Systems, and Computers, pp. 1262-1266, CA, USA, 2018.

[21] T. O'Shea and J. Hoydis, "An Introduction to Deep Learning for the Physical Layer," in IEEE Transactions on Cognitive Communications and Networking, Vol. 3, No. 4, pp. 563-575, Dec. 2017.

[22] H. Huang, W. Xia, J. Xiong, J. Yang, G. Zheng and X. Zhu, "Unsupervised Learning-Based Fast Beamforming Design for Downlink MIMO," IEEE Access, Vol. 7, pp. 7599-7605, 2019.

[23] H. Huang, Y. Peng, J. Yang, W. Xia and G. Gui, "Fast Beamforming Design via Deep Learning," IEEE Transactions on Vehicular Technology, Vol. 69, No. 1, pp. 1065-1069, Jan. 2020.

[24] W. Lee, "Resource Allocation for Multi-Channel Underlay Cognitive Radio Network Based on Deep Neural Network," IEEE Communications Letters, Vol. 22, No. 9, pp. 1942-1945, Sept. 2018.

[25] F. Liang, C. Shen, W. Yu and F. Wu, "Power Control for Interference Management via Ensembling Deep Neural Networks," IEEE/CIC International Conference on Communications in China (ICCC), pp. 237-242, 2019.

[26] M. Eisen and A. Ribeiro, "Optimal Wireless Resource Allocation With Random Edge Graph Neural Networks," IEEE Transactions on Signal Processing, Vol. 68, pp. 2977-2991, 2020.

[27] Y. S. Nasir and D. Guo "Multi-Agent Deep Reinforcement Learning for Dynamic Power Allocation in Wireless Networks," IEEE Journal on Selected Areas in Communications, Vol. 37, No. 10, pp. 2239 - 2250, 2019.

[28] F. Meng, P. Chen and L. Wu, "Power Allocation in Multi-User Cellular Networks with Deep Q Learning Approach," 2019 IEEE International Conference on Communications (ICC), pp. 1-6, Shanghai, China, 2019.

[29] F. Shah-Mohammadi and A. Kwasinski, "Deep Reinforcement Learning Approach to QoE-Driven Resource Allocation for Spectrum Underlay in Cognitive Radio Networks," 2018 IEEE International Conference on Communications Workshops (ICC Workshops), pp. 1-6, Kansas City, USA, 2018.

[30] A. Kwasinski, W. Wang and F. S. Mohammadi, "Reinforcement Learning for Resource Allocation in Cognitive Radio Networks," Machine Learning for Future Wireless Communications, pp.27-44, 2020.

[31] G. Sun, Z. T. Gebrekidan, G. O. Boateng, D. Ayepah-Mensah and W. Jiang, "Dynamic Reservation and Deep Reinforcement Learning Based Autonomous Resource Slicing for Virtualized Radio Access Networks," IEEE Access, Vol. 7, pp. 45758-45772, 2019.

[32] G. Sun, K. Xiong, G. O. Boateng, D. Ayepah-Mensah, G. Liu and W. Jiang, "Autonomous Resource Provisioning and Resource Customization for Mixed Traffics in Virtualized Radio Access Network," IEEE Systems Journal, Vol. 13, No. 3, pp. 2454-2465, Sept. 2019.

[33] R. Li, Z. Zhao, Q. Sun, C. I C. Yang, X. Chen, M. Zhao, H. Zhang, "Deep Reinforcement Learning for Resource Management in Network Slicing," IEEE Access, Vol. 6, pp. 74429-74441, 2018.

[34] H. Ye, G. Y. Li and B. F. Juang, "Deep Reinforcement Learning Based Resource Allocation for V2V Communications," IEEE Transactions on Vehicular Technology, Vol. 68, No. 4, pp. 3163-3173, April 2019.

[35] H. Gacanin, E. Perenda and R. Atawia, "Self-Deployment of Non-stationary Wireless System by Knowledge Management with Artificial Intelligence," IEEE Transactions on Cognitive Communications and Networking, Vol. 5, No. 4, pp. 1004-1018, Dec. 2019.

[36] H. Gacanin, E. Perenda, S. Karunarathne and R. Atawia, "Self-optimization of Wireless Systems with Knowledge Management: An Artificial Intelligence Approach," IEEE Transactions on Vehicular Technology, Vol. 68, No. 10, pp. 9682-9697, Oct. 2019.

[37] E. Hyytiä, H. Koskinen, P. Lassila, A. Penttinen, J. Roszik and J. Virtamo, "Random waypoint model in wireless networks," Networks and algorithms: Complexity in physics and computer science, Helsinki, Vol. 590, 2005.

[38] J. G. Proakis, *Digital Communications*, 3rd edition. McGraw-Hill, 1995.

[39] H. Gacanin, M. Salmela and F. Adachi, "Performance Analysis of Analog Network Coding with Imperfect Channel Estimation in a Frequency-Selective Fading Channel," IEEE Transactions on Wireless Communications, Vol. 11, No. 2, pp. 742-750, Feb. 2012.

[40] Q. Qi, A. Minturn and Y. Yang, "An efficient water-filling algorithm for power allocation in OFDM-based cognitive radio systems," 2012 International Conference on Systems and Informatics (ICSAI2012), pp. 2069-2073, Yantai, China, May 2012.

[41] C. Y. Wong, R. S. Cheng, K. B. Lataief and R. D. Murch, (1999). "Multiuser OFDM with adaptive subcarrier, bit, and power allocation," IEEE Journal on selected areas in communications, Vol. 17, No. 10, pp. 1747-1758, Oct. 1999.

[42] Q. Shi, M. Razaviyayn, Z. Q. Luo, Zhi-Quan and C. He, "An Iteratively Weighted MMSE Approach to Distributed Sum-Utility Maximization for a MIMO Interfering Broadcast Channel," IEEE Transactions on Signal Processing, Vol. 59, No. 9, pp. 4331 - 4340, Sept. 2011.

[43] Z.Q. Luo and W. Yu, "An introduction to convex optimization for communications and signal processing," IEEE Journal on Selected Areas in Communications, Vol.24, No.8, pp.1426-1438, Aug. 2006.

[44] Z. Yang, C. Pan, H. Xu, J. Shi and M. Chen, "Power Control and Resource Allocation for Multi-Cell OFDM Networks With Load Coupling," IEEE Access, Vol. 6, pp. 15969-15979, 2018.

[45] S. Ruder, "An overview of gradient descent optimization algorithms," 2016. arXiv preprint arXiv:1609.04747.

[46] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," 2014. arXiv preprint arXiv:1412.6980.

[47] R. Caruana, S. Lawrence and C. L. Giles, "Overfitting in neural nets: Backpropagation, conjugate gradient, and early stopping," Advances in neural information processing systems, pp. 402-408, Vancouver, Canada, 2001.

[48] Y. Bengio, "Practical recommendations for gradient-based training of deep architectures," Neural networks: Tricks of the trade. Lecture Notes in Computer Science, Vol. 7700, pp. 437-478. Springer, Berlin, Heidelberg. 2012.

[49] S. Haykin, *Neural Networks and Learning Machines.*, Pearson Education India, 3rd Edition, 2010.

[50] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*, A Bradford Book, 2nd Edition, 2018.

[51] K. Arulkumaran, M. P. Deisenroth, M. Brundage and A. A. Bharath, "Deep Reinforcement Learning: A Brief Survey," IEEE Signal Processing Magazine, Vol. 34, No. 6, pp. 26-38, Nov. 2017.
[52] CPU Performance, https://setiathome.berkeley.edu/cpu_list.php, Feb, 2020 (accessed on Feb. 27, 2020).
[53] M. Bennis, M. Debbah and H. V. Poor, "Ultrareliable and low-latency wireless communication: Tail, risk, and scale," Proceedings of the IEEE, Vol. 106, No. 10, pp.1834-1853, Sept. 2018.
[54] A. Sherstinsky, "Fundamentals of recurrent neural network (rnn) and long short-term memory (lstm) network," Physica D: Nonlinear Phenomena, Vol. 404, pp.132306, March 2020.
[55] L. Liu and U. Mitra, "On Sampled Reinforcement Learning in Wireless Networks: Exploitation of Policy Structures," IEEE Transactions on Communications, Vol. 68, No. 5, pp. 2823-2837, May 2020.
[56] Z. Wang, T. Schaul, M. Hessel, H. Van Hasselt, M. Lanctot and N. De Freitas, "Dueling network architectures for deep reinforcement learning," International conference on machine learning, pp. 1995-2003, New York, USA, 2016.
[57] T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver and D. Wierstra, "Continuous control with deep reinforcement learning," 2015. arXiv preprint arXiv:1509.02971

SUREN SRITHARAN is an undergraduate student in the Computer Engineering program at the University of Peradeniya, Sri Lanka and will be graduating in 2020 with a BS in Computer Engineering. He has a strong interest in the field of Machine Learning, Algorithmic programming and Optimization with applications specifically related to Machine Vision and Wireless Communication.

HARSHANA WELIGAMPOLA is pursuing B.Sc. degree on Computer Engineering in University of Peradeniya, Sri Lanka. His research interests include Artificial Intelligence, Neural Networks, Optimization and Image Processing.

HARIS GACANIN [IEEE SM'12, IEICE SM'12] received his Dipl.-Ing. Degree in Electrical engineering from the University of Sarajevo in 2000. In 2005 and 2008, respectively, he received MSc and Ph.D. from Tohoku University in Japan. He was with Tohoku University from 2008 until 2010 first as Japan Society for Promotion of Science postdoctoral fellow and later, as Assistant Professor. In 2010, he joined Alcatel-Lucent (now Nokia) where he led the research department at Nokia Bell Labs. Currently, he is a full (chair) professor at RWTH Aachen University in Germany. His professional interests are related to broad area of digital signal processing and artificial intelligence with applications in communication systems. He has 200+ scientific publications (journals, conferences and patent applications) and invited/tutorial talks. He is an Associate Editor of IEEE Communications Magazine and previously serviced at IEICE Transactions on Communications and IET Communications. He is IEEE VTS Distinguished Lecturer and he acted as a general chair and technical program committee member of various international conferences. He is a recipient of several Nokia's awards for innovations, IEICE Communication System Study Group (2015) Award, the 2013 Alcatel-Lucent Award of Excellence, the 2012 KDDI Foundation Research Award, the 2009 KDDI Foundation Research Grant Award, the 2008 Japan Society for Promotion of Science (JSPS) Postdoctoral Fellowships for Foreign Researchers, the 2005 Active Research Award in Radio Communications, 2005 Vehicular Technology Conference (VTC 2005-Fall) Student Paper Award from IEEE VTS Japan Chapter and the 2004 Institute of IEICE Society Young Researcher Award.