IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received April 2, 2020, accepted April 20, 2020, date of publication April 28, 2020, date of current version May 15, 2020.

*Digital Object Identifier 10.1109/ACCESS.2020.2991057*

# Advanced Energy-Efficient Computation Offloading Using Deep Reinforcement Learning in MTC Edge Computing

**ISRAR KHAN[1], XIAOFENG TAO[1], (Senior Member, IEEE), G. M. SHAFIQUR RAHMAN[2], WAHEED UR REHMAN[1,3], AND TABINDA SALAM[1,4]**

[1]National Engineering Laboratory for Mobile Network Technologies, Beijing University of Posts and Telecommunications, Beijing 100876, China
[2]Key Laboratory of Universal Wireless Communications, Ministry of Education, Beijing University of Posts and Telecommunications, Beijing 100876, China
[3]Department of Computer Science, University of Peshawar, Peshawar 25120, Pakistan
[4]Department of Computer Science, Shaheed Benazir Bhutto Women University, Peshawar 25000, Pakistan

Corresponding author: Xiaofeng Tao (taoxf@bupt.edu.cn)

This work was supported in part by the National Natural Science Foundation of China under Grant 61932005 and Grant 61971066, in part by the Beijing Natural Science Foundation under Grant L172033, and in part by the 111 Project of China under Grant B16006.

**ABSTRACT** Mobile edge computing (MEC) supports the internet of things (IoT) by leveraging computation offloading. It minimizes the delay and consequently reduces the energy consumption of the IoT devices. However, the consideration of static communication mode in most of the recent work, despite varying network dynamics and resource diversity, is the main limitation. An energy-efficient computation offloading method using deep reinforcement learning (DRL) is proposed. Both delay-tolerant and non-delay tolerant scenarios are considered using capillary machine type communication (MTC). Depending upon the type of service, an intelligent MTC edge server using DRL decides either process the incoming request at the MTC edge server or sends it to the cloud server. To control communication, we draft a markov decision problem (MDP). This minimizes the long-term power consumption of the system. The formulation of the optimization problem is considered under the constraint of computing power resources and delays. Simulation results delineate the significant performance gain of 12% in computation offloading through the proposed DRL approach. The effectiveness and superiority of the proposed model are compared with other baselines and are demonstrated numerically.

**INDEX TERMS** Machine type communication, mobile edge computing, computation offloading, deep reinforcement learning, energy efficiency.

## I. INTRODUCTION

The future of wireless communication holds various application scenarios in the form of the internet of things (IoT), industry automation, virtual reality (VR) and augmented reality (AR) [1]. Furthermore, it is also envisioned that the number of IoT devices by 2025 will reach a staggering 75 billion [2]. Most of these devices will be actuators, for instance, machine type communication devices (MTCDs) that will dominate over human-type communication devices (HTCDs) such as mobile phones. The machine type communication (MTC) provides the autonomous interaction of sensing, actuation, and processing devices without human supervision [3]–[6]. The traffic characteristics of MTCDs vary from best-effort services such as utility metering, and environmental monitoring systems to ultra-reliable ones such as mission-critical industry, public safety, and health-care [7]. Most of these services are compute-intensive and require computational support to be provided in reality [8].

Cloud computing is considered as one of the most effective solutions to provide compute-intensive support [9]. It is usually connected using a long backhaul that yields intolerable delays and inefficient energy utilization [10]. This significantly affects the applications' performance. In contrast, mobile edge computing (MEC) promises to overcome these challenges by reducing the communication distance. The computation facility in MEC is provided at the access network, thus reducing the energy consumption and

The associate editor coordinating the review of this manuscript and approving it for publication was Huazhu Fu.

transmission delay. However, geo-distributed MEC devices cannot cope with all the computation offloading tasks due to their limited resource capabilities. It means that some have to be processed at the remote cloud server. This entails an efficient solution to classify the traffic based on its delay constraints.

Recently, machine learning (ML) is considered as an effective technique for solving many classification challenges and problems. The authors in [11], [12] have researched the computation offloading problem using machine learning on time-varying computing systems.

The authors in [13] have proposed a reinforced learning (RL) based machine learning algorithm for solving random variation problems in the time domain of the wireless channels. It is done by considering the future reward feedback mechanism from the environment to help achieve long-term goals.

Besides, a binary edge computing-based offloading scheme known as deep reinforcement learning (DRL) is used to choose the user to offload their task to an edge server that serves multiple users [14], [15]. By using the DRL method, MEC determines whether or not the computing task should be offloaded to the MEC server [16]. The authors in [16] and [17] introduced the content of DRL in detail. Both, however, neglect the capability of computation by the mobile device itself. This can reduce the computation delay for some tasks. In [18], the authors consider the cloud and local computing cooperation.

The authors in [19] proposed a DRL method to improve energy efficiency in the internet of vehicles. In the proposed solutions the authors established a fog-cloud offloading system to optimize the power consumption based on delay constraints. The overall system is decomposed into the front-end and back-end. The authors in [20] propose an energy-efficient task offloading strategy based on the channel constraint with deadline awareness. The proposed scheme minimizes the energy consumption of user equipment along with satisfying the deadline conditions of mobile cloud workflows. To handle the massive connection in the presence of heterogeneous edge servers, the authors in [21] investigate the computation offloading management problem. The authors jointly considered the channel states with power consumption in addition to latency and diverse computation resources.

An RL-based algorithm for a multi-user MEC computation offloading system is designed by the authors in [22]. They have replaced the markov decision process (MDP) by proposing a Q-learning based theme. In [23], the DRL is utilized to solve the storage and computation problem of action-value Q. This DRL based algorithm is called deep Q network (DQN). It uses deep neural network (DNN) for the value function estimation of the Q-learning algorithm [24]. In [25], [26] a single edge server based edge-computing system is discussed. Multiple UEs can offload computation to the server. The optimization problem under consideration is the energy consumption and sum cost of delay of all the UEs. However, these works do not consider energy efficiency with delay-tolerant and non-delay tolerant devices, which is more likely the scenario of the future internet.

In this paper, we jointly consider MTC edge and cloud servers for computation offloading to improve energy efficiency with delay constraints. Using DRL, the computational offloading requests are classified as either delay tolerant or non-delay tolerant. Consequently, the delay-tolerant and non-delay tolerant requests are served at the cloud server and an MTC edge server, respectively. The use of edge servers reduces the transmission distance as compared to the cloud server and thus improves the quality of service (QoS) in addition to energy efficiency.

The main contributions of this paper are:

- A DRL based computation offloading strategy for MTCDs is presented. This scheme improves the energy efficiency of the MTCDs using the edge computing infrastructure with delay constraints. Moreover, it also enables the MTCDs to achieve optimal offloading without knowing the MTC edge and cloud server model.
- The proposed DRL based approach minimizes the energy consumption in the uplink domain of the MTC devices for achieving energy efficiency in the MTC system. It does this by choosing an appropriate tier to perform the task of computation offloading.
- We analyze the power consumption under the different delay constraints and evaluate the energy consumption of the offloading tasks of the delay tolerant and non-delay tolerant MTCDs.
- The impacts of DRL based rewards are illustrated, and the simulation results are compared with Q-learning and fixed resource allocation schemes to analyze performance efficiency. The DRL based proposed scheme achieves 12% more reward than the fixed resource allocation approach.

## II. SYSTEM MODEL

We consider an MTC environment, where a large number of MTCDs are distributed throughout the network. The MTCDs are generating rigorous computing tasks that need to be offloaded based on the allocated maximum power resources and minimum delay constraints. To provide real-time computation and intelligent decision for energy-efficient processing, the MTC edge server, and DRL controller are connected with the evolved node B (eNB). All the MTCDs are connected with the eNB for communication with the MTC edge server and the DRL. The DRL takes an intelligent decision for processing MTCD data either at the edge server or at the cloud server based on delay tolerance. The DRL sends the data of delay-tolerant MTCDs to the cloud server for computation. While the data of non-delay tolerant MTCDs are computed at the MTC edge server. Fig. 1 shows the proposed network mode.

Let all the MTCDs (delay tolerant and non-delay tolerant) be represented as $v = \{1, 2, \ldots, V\}$ and with a single eNB represented as $J$, where $V$ and $J$ are the total number of MTCDs and eNB, respectively. The eNB is

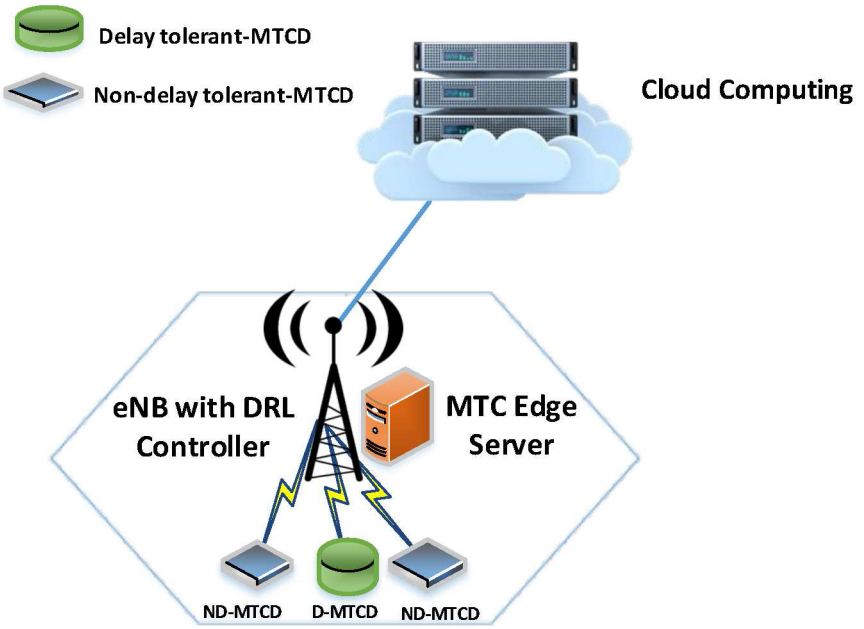


**FIGURE 1.** MTC architecture with Deep Reinforcement Learning (DRL) for achieving energy efficiency.

connected to the MTC edge server and the DRL controller for making intelligent decisions. The eNB is also connected with a cloud server represented as $M$. It has a comparatively larger computation capacity than the MTC edge server.

The problem is approached with the appropriate mode selection to offload the rigorous computing tasks in a given time frame either to the MTC edge server or cloud server. We consider $S_v^{mode} \in \{0, 1\}$ as the indicator function. Where, if Smode $S_v^{mode} = 0$, the DRL controller chooses the MTC edge server for computing the tasks of non-delay tolerant MTCDs. If $S_v^{mode} = 1$, the DRL controller chooses the cloud server for computing the tasks of delay-tolerant MTCDs.

We also assume that the MTCD is equipped with a single antenna and each eNB is equipped with multiple antennas $L_t$. A detailed breakdown of the MTC edge and cloud server communication and the related energy consumption model is provided in the following subsection. It is worth noting that we have considered intra-cell interference in our proposed model.

### A. MEC COMMUNICATION MODEL

Since the MTCDs $v$ do not posses processing capability, all the tasks need to be offloaded to edge computing tier or cloud computing tier. At the edge tier, the tasks will be executed by the MTC edge server. However, as the edge server has limited resource capacity, the edge server cannot tackle both delay-tolerant and non-delay tolerant MTCDs computing within the edge tier. If all the tasks could be tackled at the edge tier then the energy consumption will be very low. The communication link $y_{v,j}$ between the MTCDs $v$ and the eNB $j$ is represented as:

$$y_{v,j} = \boldsymbol{h}_{v,j}^H \boldsymbol{w}_j s_i^v + \sum_{i \neq v, i \in j} \boldsymbol{h}_{v,j}^H \boldsymbol{w}_i s_i^v + n_j, \forall v, j, \tag{1}$$

$h_{v,j}^H$ is the channel state information (CSI) matrix from MTCD $v$ to eNB $j$, $\boldsymbol{w}_j$ represents the transmitted beam-formed towards the MTCD $v$. The received additive Gaussian noise at the MTCD $v$ is denoted by $n_j$ which is distributed as $(0, \sigma_i^2)$. In addition, $s_i^v$ denotes the transmitted signal vector which contains the message to eNB $j$.

In the system model for energy efficiency, the received signal to interference plus noise ratio (SINR) is an important parameter. We use it to ensure the QoS and evaluate the channel capacity. The SINR from the MTCD $v$ to the eNB $j$ is represented by the following equation:

$$SINR_{v,j} = \frac{|\boldsymbol{h}_{v,j}^H \boldsymbol{w}_{v,j}|^2}{\sum_{i \neq v} |\boldsymbol{h}_{i,j}^H \boldsymbol{w}_{i,j}|^2 + \sigma_j^2}, \quad \forall v, j, \tag{2}$$

where $\boldsymbol{w}_v$ is the power for beam-forming of the link between MTCD $v$ and the eNB $j$. The transmit power $\boldsymbol{w}_v$ can vary for all the MTCDs. Its variance can result from the channel conditions and the power constraints imposed by the eNB.

The optimal transmission rate minimizes transmission and communication delays. It also forces the system to ensure the QoS. Keeping this under consideration and Shannon's channel capacity, the following equation represents the achievable transmission rate from MTCD $v$ to eNB $j$:

$$R_{v,j} = W \log_2(1 + \gamma_{min}) \tag{3}$$

where the total achievable channel bandwidth of the network is represented by $W$ and $\gamma_{min}$ is the lower bound to ensure the QoS.

We define the target SINR as $\gamma_{min}$, which can be represented as

$$\gamma_{min} = \frac{|\boldsymbol{h}_{v,j}^H \boldsymbol{w}_{v,j}|^2}{\sum_{i \neq v} |\boldsymbol{h}_{i,j}^H \boldsymbol{w}_{i,j}|^2 + \sigma_j^2}, \quad \forall v, j, \tag{4}$$

Equation (4) suggests that the threshold $\gamma_{min}$ must be satisfied with choosing the channel in the access network.

### B. ENERGY CONSUMPTION MODEL

In the considered MEC architecture, the energy consumption is calculated as: (i) Energy consumption for uploading and downloading the computing task and (ii) processing energy consumption in a different tier. Generally, the uploading of energy consumption occurs in two phases. First, the DRL controller decides if the task can be computed at the MTC edge server. If it can be computed at the MTC edge server then the MTCD $v$ uploads the generated tasks to the MTC edge server for processing. If the task is beyond the processing capability of the edge server, then it will be assigned to the cloud server. The task in the latter case consumes higher energy as compared to the former case. We consider the trade-off between delay and power consumption. We also consider the QoS of the network to offload all the tasks within the tolerable delay.

We assume that each MTCD generates $W_v^{task}$, which needs to be offloaded within the allocated time frame either at the edge tier or cloud-tier, based on the task size and offloading urgency. With transmission delay $D_{v,M}$ from MTCD $v$ to cloud-tier $M$ and computing delay $D_{edge}^{computing}$ at the edge node, total delay from MTCD to cloud-tier is considered as [27];

$$\begin{aligned} D_{total} &= D_{v,M} + D_{edge}^{computing}, \\ &= D_{v,j} + D_{j,M} + D_{edge}^{computing}, \\ &= \frac{W_v^{task}}{R_{v,j}} + \frac{W_v^{task}}{R_{j,M}} + \frac{W_v^{task}}{\phi_{j,v}}, \end{aligned} \tag{5}$$

where $D_{v,j}$ and $D_{j,M}$ are the transmission delays from MTCD $v$ to eNB $j$, and eNB $j$ to cloud-tier $M$, respectively. While $R_{*,*}$ represents the achievable transmission rate in the respective hop. Furthermore, $\phi_{j,v}$ represents the maximum available computing resource at the edge.

Similarly, the energy consumption from uploading the data from MTCD $v$ to cloud-tier $M$ including processing at the edge and cloud tiers can be represented as [27];

$$\begin{aligned} E_{total} &= E_{v,j}^{upload} + E_{j,M}^{upload} + E_v^{Edge} + E_v^{Cloud}, \\ &= P_v \frac{W_v^{task}}{R_{v,j}} + P_j \frac{W_v^{task}}{R_{j,M}} + \sum_{v=1}^{V} \omega_j N_{j,v} + \sum_{v=1}^{V} \omega_M N_{M,v} \end{aligned} \tag{6}$$

where $E_{v,j}^{upload}$ and $E_{j,M}^{upload}$ represents energy consumption for uploading data from MTCD $v$ to eNB $j$ and subsequently to the cloud-tier $M$, respectively. While, processing at the edge and cloud tiers are represented as $E_v^{Edge}$ and $E_v^{Cloud}$, respectively. Furthermore, $P_v$ represents power consumption for uplink data transmission from MTCD $v$ to eNB $j$, $N_{j,v}$ represents computing resources of MTC edge Server $j$ to MTCD $v$, $P_j$ represents power consumption from edge tier $j$ to cloud-tier $M$ and $N_{M,v}$ represents computing resources of Cloud $M$ to MTCD $v$.

## III. PROBLEM FORMULATION

To minimize the power consumption we draft a joint optimization problem. We explicitly consider the CSI and transmission delay to achieve energy efficiency in the edge and cloud server-based MTC architecture. The energy optimization problem is stated as

$$\begin{aligned} \underset{\{s_v^{mode}\}}{\text{minimize}} \; & (1 - s_v^{mode}) \left( P_v \frac{W_v^{task}}{R_{v,j}} + \sum_{v=1}^{V} \omega_j N_{j,v} \right) \\ & + s_v^{mode} \left( P_j \frac{W_v^{task}}{R_{j,M}} + \sum_{v=1}^{V} \omega_M N_{M,v} \right) \end{aligned} \tag{7}$$

$$s.t. \; \frac{|\boldsymbol{h}_{v,j}^H \boldsymbol{w}_{v,j}|^2}{\sum_{i \neq v} |\boldsymbol{h}_{i,j}^H \boldsymbol{w}_{i,j}|^2 + \sigma_j^2} \geq \gamma_{min}, \quad \forall v, j \tag{8}$$

$$P_v \frac{W_v^{task}}{R_{v,j}} + P_j \frac{W_v^{task}}{R_{j,M}} \leq P_{total}, \quad \forall v \tag{9}$$

$$\sum_{v=1}^{V} \omega_j N_{j,v} + \sum_{v=1}^{V} \omega_M N_{M,v} \leq P_{max} \tag{10}$$

where constraint (8) represents the QoS demand for each MTCD $v$. The transmit power cannot go beyond the total power $P_{total}$ and is represented by the constraint (9). Likewise, constraint (10) depicts that the computing power is also bounded by the maximum power capacity $P_{max}$. Furthermore, the optimization variable $s_v^{mode}$ has been exploited to make the appropriate decision by selecting a proper mode to offload the data.

## IV. DRL BASED ENERGY EFFICIENCY IN MTC EDGE COMPUTING

The computation offloading problem is formulated to achieve energy efficiency based on MDP. To solve this problem we use a DRL algorithm.

The problem of MDP is designed by using the state and action space, and immediate reward. From hereon the model will be defined with the tuple of $\{\mathcal{S}, \mathcal{A}, \mathcal{P}(s_{t+1}|s_t, a_t), \mathcal{R}(s_t, a_t)\}$ where the scenario of the environment is represented by the set of states symbolized by $\mathcal{S}$, and the set of a possible number of actions is symbolized by $\mathcal{A}$. After performing the action $a_t$ at state $s_t$, the state transitioning probability to the state $s_{t+1}$ is given by $\mathcal{P}(s_{t+1}|s_t, a_t)$ and $R(s_t, a_t)$ characterizes the received reward, when action $a_t$ has taken at state $s_t$. The objective of the MDP

model is to obtain the maximum accumulated rewards $R$ over a long period $T$. In the following subsection, we provide a detailed breakdown of our computation offloading problem for MTCDs.

#### 1) STATE SPACE

The system state is designed for the energy efficiency by offloading the MTCD data. It is compatible with the currently available communication mode, power resources, and CSI. We assume that there are two modes for each MTCD $v$. The first mode is the eNB tier mode $M_v^{eNB}$ and the second is cloud tier mode $M_v^{cloud}$. Thus, the state space can be designed as $s_t = \{M_v^{mode}, H_{v,J}^H, D_{v,j}, D_{computing}^{edge}\}$ where, $H_{v,J}^H$ is the CSI for MTCD $v$ and eNB $j$, and $D_{v,j}$ is generated delay for offloading the tasks to the eNB computing tier.

#### 2) ACTION SPACE

To achieve enhanced performance and to accomplish the tasks the agent can perform a large number of actions. Nevertheless, to perform a large number of actions massive computation capacity is required. This, in turn, reduces the overall system performance. Therefore, it is considered that the agent performs only one action at each time state $t$ based on the state space $s_t$. The action space is represented as $a_t = [s_v^{mode}]$. Based on the available CSI and delay of $D_{v,j}$ the agent selects the optimal communication mode for offloading the delay.

#### 3) REWARD FUNCTION

The purpose of rewarding the previous action is to give feedback to the RL model. Consequently, it is crucial to define an appropriate reward to improve the learning process. An efficient rewarding function influences the process for finding optimal policy [28]. The proposed reward function is closely related to the energy-minimizing problem (7). It can be achieved by efficient computation offloading. The negative-sum cost of the system's energy consumption is regarded as an immediate reward. The immediate reward $r_t$ depends primarily on the precise mode selection. Therefore, the immediate reward can be represented as

$$r_t = -\left((1 - s_v^{mode})\left(P_v \frac{W_v^{task}}{R_{v,j}} + \sum_{v=1}^{V} \omega_j N_{j,v}\right) + s_v^{mode}\left(P_j \frac{W_v^{task}}{R_{j,M}} + \sum_{v=1}^{V} \omega_M N_{M,v}\right)\right) \quad (11)$$

The accumulative reward for the longer period $T$ is considered as minimizing the energy cost of the system. Let $\xi$ be the discount factor between $[0, 1]$ that relates the current mode selection with the effect of future reward. Then, $R_t = \sum_{t=0}^{T} \xi^t r_t$, where $R_t$ is the accumulative future reward. It is worth mentioning that more importance on immediate rewards is placed for a lower value of $\xi$ [29]. The overall rewards are always negative because we attempt to minimize the energy cost of the system. Thus, our main goal is to maximize the reward for minimizing energy.

The state transition probability $\mathcal{P}(s_{t+1}|s_t, a_t)$ is important for attaining long term accumulative reward. However, it is quite difficult to obtain the transition probability in the MDP problem. Hence, the Q-learning has been used to solve the MDP problems. The Q-function is defined as $Q(s_t, a_t)$ which shows the quality of an action $a_t$ at a given state $s_t$. The maximum expected achievable reward is gained by the Q-function that follows the policy $\pi$, which can be stated as [28];

$$Q^{\pi}(s, a) = \mathbb{E}\left[\sum_{t=0}^{T} \xi^t r_t | s_0 = s, a_0 = a, \pi\right], \quad (12)$$

where from the beginning state $s_0$ and action $a_0$ the Q-function is responsible for obtaining the accumulative rewards.

Hereafter, by following the Bellman criterion, the optimal Q-function is estimated as [30];

$$Q^*(s_t, a_t) = \mathbb{E}_{t+1}\left[r_t + \xi \max_{a_{t+1}} Q^*(s_{t+1}, a_{t+1})\right], \quad (13)$$

Usually, the Q-function is attained in an iterative manner by means of the information of $(s, a, r, s', a')$ at the next time state $s_{t+1}$. Henceforth the updated Q-function can be given as [28];

$$Q_{t+1}(s, a) = (1 - \alpha)Q(s_t, a_t) + \alpha\left(r_t + \xi \max_{a' \in A} Q\left(s_{t+1}, a'\right)\right) \quad (14)$$

where $\alpha$ represents the learning rate and $\alpha \in [0, 1]$ represents the weight of the current offloading decision. By using the proper learning rate, the repetition algorithm makes sure that $Q_t(s, a)$ will definitely be converged to $Q^*(s, a)$. The optimal action $\mathcal{A}$ and mode selection $s_v^{mode}$ can be obtained by repetitively updating the Q-value from each state-action pair [28].

The RL problem for a fewer number of state-action scenarios can be efficiently solved by using the Q-learning algorithm. It is because when the state-action pair is large, it becomes challenging to go over each step, having all the samples that are stored in Q-table [28]. This behavior intrinsically limits the traditional RL with fully observed low-dimensional state space [31]. That is why the energy minimization problem is formulated in the dynamic wireless environment. This leads to an extremely large number of state space. To enhance the performance by avoiding the curse of dimensionality and the limitations of Q-learning we use the DQN approach [28].

### A. DRL BASED OFFLOADING

In this subsection, we introduce a DRL based model selection. The DRL acts as an agent. It combines the traditional RL and DNN. It is used for efficient computation offloading to accomplish energy efficiency in the MTC edge server. This approach significantly accelerates the learning process [32].

A DRL has a unique pair in the form of possible action and an output unit. The state portrayal can be regarded as an input source to the neural network (NN) [29]. In a single

**TABLE 1. Experimental parameters.**

| | |
|---|---|
| Number of eNB | 1 |
| Number of MTCDs ($v$) to eNB | 10 |
| Channel bandwidth ($W$) | 20 MHz |
| Power noise ($\sigma^2$) | -101dBm |
| Pathloss model | $128 + 37 * \log_{10}$(distance) |
| Reward decay $\xi$ | 0.9 |
| Rate of learning $\alpha$ | 0.01 |
| Experience replay buffer size $N_D$ | 500 |
| Size of Mini-batch $B$ | 32 |

pass, the DRL can compute the Q-values for all possible actions in a given state. This reduces the complexity of the system [28], [32].

The DRL becomes capable to achieve better performance by using the NN. This happens due to less interaction with the complex environment by using the inherent generalization capability and the replay memory. DRL also approximate $Q(s, a; \theta) to Q^*(s, a)$ that results in performance improvement and avoids the unreasonable Q-learning situation in RL [32].

In the algorithm 1, we analyze the energy-efficient computation offloading. The input state is the delay between MTCD and eNB. The Q-values $Q(s, a, \omega)$ having weight of $\omega$ are their outputs. They detect the precise estimate regarding all possible actions. The controller performs the action $a_t$ based on the given state. After executing the action $a_t$ the system moves to a new state $s_{t+1}$. From the reward functions of energy efficiency the agent simultaneously calculates the reward $r_t$. This reward is based on the selected modes. As an interaction the replay memory $D$ stores this transition $(s_t, a_t, r_t, s_{t+1})$. The DQN is updated after a limited number of iterations through a batch of random sample of size $B$ selected from the replay memory $D$. The following equation gives the minimized loss function by training the DQN towards the target value [33];

$$L(\omega) = \mathbb{E}\left[(r_t + \xi \max_{a' \in A} \hat{Q}(s_{t+1}, a'; \hat{\omega}) - Q(s_{t+1}, a_t; \omega))^2\right] \tag{15}$$

where $(r_t + \xi \max_{a' \in A} \hat{Q}(s_{t+1}, a'; \hat{\omega})$ represents the target network [29]. In every certain period, the agent will set the weights of the DQN to the target DQN.

**Algorithm 1** DRL Based Energy Optimization Algorithm in MEC

**Initialization:**
Initialize $Q(s, a)$ with weights $\omega$ randomly.
Construct $\hat{Q}(s, a)$ with weights $\hat{\omega}$ from $Q(s, a)$ that is randomly initialized previously.
Setup the initial value of $D$ with capacity $N_D$, where $D$ is the replay memory.
Let $B$ be the maximum size of mini-batch.
Let $E_{max}$ be the maximum training episode.

**For** *episode* $= 1$ to $E_{max}$ **do**
Initialize the beginning state $s_0$.
**For** $t = 1$ to $T - 1$: $t =$ decision step **do**
Generate random $x \in (0, 1)$.
**If** $x \leq \varepsilon$:
The agent will perform a random action $a_t$.
**Else**:
The agent will perform the action $a_t$ as $a_t = \arg\max_{a_t \in A} Q(s_t, a_t; \omega)$.
**End If**
Execute action $a_t$ in the emulator.
Observe the reward $r_t$ according to the formulated problem (11). Based on the performed action $a_t$ for energy efficiency, evaluate the reward $r_t$ whether the energy consumption is minimized or not and then observe the upcoming state $s_{t+1}$.
Store the reward $r_t$ which is achieved according to formulated problems together $s_t$, $s_{t+1}$ and $a_t$ as $(s_t, a_t, r_t, s_{t+1})$ in replay memory D.
Randomly sample the mini-batch with the size $B$ of transitions $(s_t, a_t, r_t, s_{t+1})$ from the replay memory $D$.
Perform a gradient descent on $(r_t + \xi \max_{a' \in A} \hat{Q}(s_{t+1}, a'; \hat{\omega}) - Q(s_{t+1}, a_t; \omega))^2$ with respect to $\omega$ to train the Q-network based on selected transition.
Periodically set the value of parameter $\omega$ to $\hat{\omega}$.
**End For**
**End For**

## V. SIMULATIONS RESULTS AND DISCUSSIONS

In this section, we have demonstrated the performance of the simulation. For simulation, we have used python 3.6 and tensor flow 1.14 to build the neural network. To construct a neural network, two fully connected dense layers have been selected. We choose 1 eNB, and 10 MTCDs. Furthermore, for each MTCD, the transmit power is chosen as 10dBm. The network area is selected as 400∗400. The rest of the parameters are illustrated in Table 1.

### A. PERFORMANCE ANALYSIS

In the beginning, we compared the DRL learning parameter, where the transmit power, a single eNB, MTC-edge server and cloud computing server domain are taken into consideration. Furthermore, the delay tolerance and non-delay tolerance offloading schemes are also taken into account. The performance of DRL's convergence stabilizes when the batch size is set to 32. With the minimal batch size, the DRL exhibits rough performance. When we take the larger batch size of 64, its impact is almost similar to that of batch size 32. In both cases, the DRL shows a very smooth performance. This can be seen in Fig. 2.

The impacts of different learning rates are depicted in Fig. 3, where can see that when the learning is very high $\alpha = 0.09$, the agent learns very fast but it is trapped between local optimums because it cannot explore all the states. After a certain number of steps, the agent's cumulative reward starts

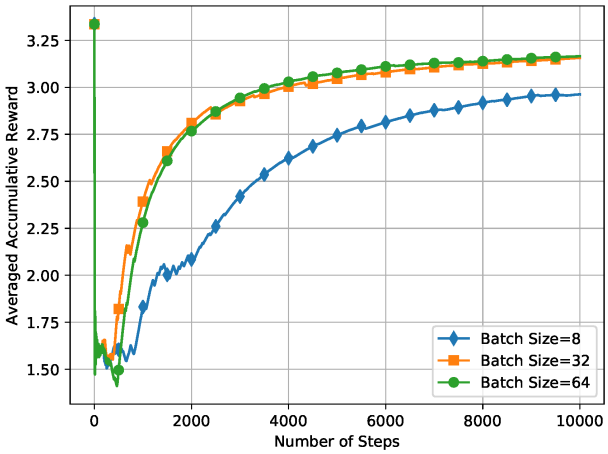


**FIGURE 2.** Evaluation of DRL performance with various batch size.

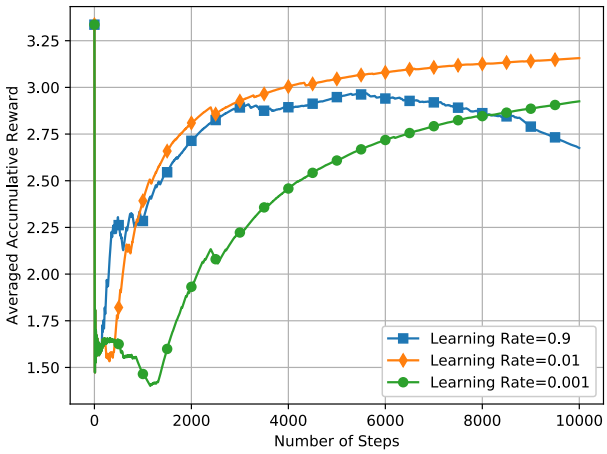


**FIGURE 3.** Evaluation of DRL performance with various learning rate.

to decrease. Consequently, the output did not achieve optimality. In contrast, when the learning is low, around 0.001, the system learns very well but slowly, which generates a long delay. However, with the learning rate of 0.01, the agent achieves very stable performance.

Fig. 4 shows the QoS requirements' effect on our proposed system. It can be seen that when the QoS is higher around 400 Kbps, the system consumes more power levels around 230mW for ensuring the best performance of the system. However, with the lower QoS demand, power consumption is linearly decreased.

Fig. 5 demonstrates the performance tackling the delay tolerance and non-delay tolerance service request. We consider the same number of requests. However, it has been observed that when the maximum possible number of requests are served at the eNB, the power consumption remains minimum. Despite the same number of requests, the non-delay tolerance service consumes less power about 128mW because of its proximity to the eNB, while the delay tolerance services consume the power of 160mW. It is because of long-distance and traversing on different mid nodes. It can also be observed

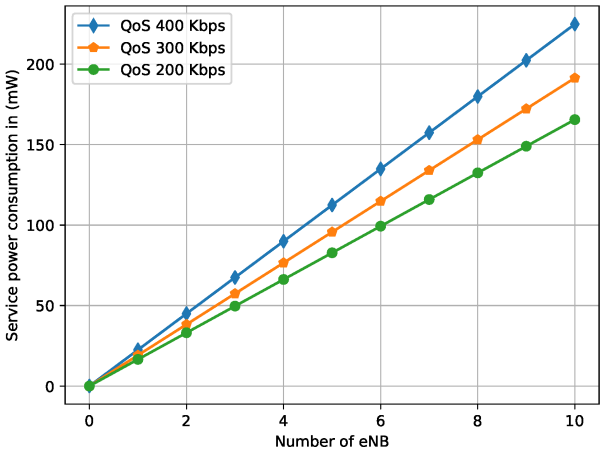


**FIGURE 4.** Comparison of power consumption under different QoS.

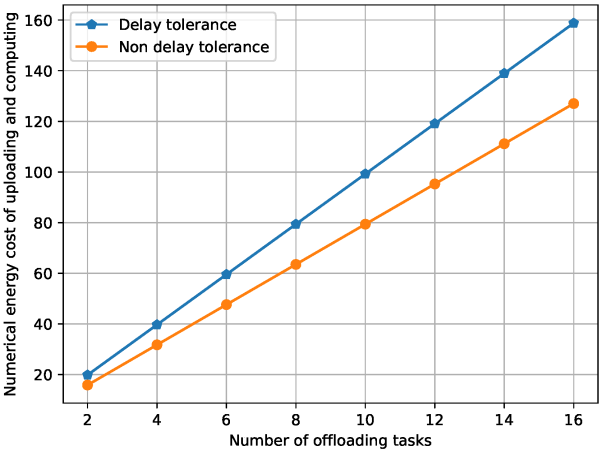


**FIGURE 5.** Energy consumption of delay and non-delay tolerance offloading.

that the delay tolerance mode consumes around 20% higher energy than the non-delay tolerance mode.

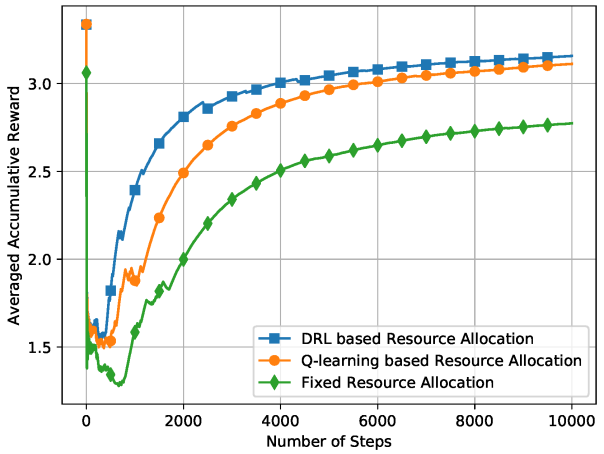


**FIGURE 6.** Benchmark comparison.

To compare the performance of DRL, we have considered the Q-learning and fixed resource allocation algorithm. The overall performance comparison is showed in Fig.6.

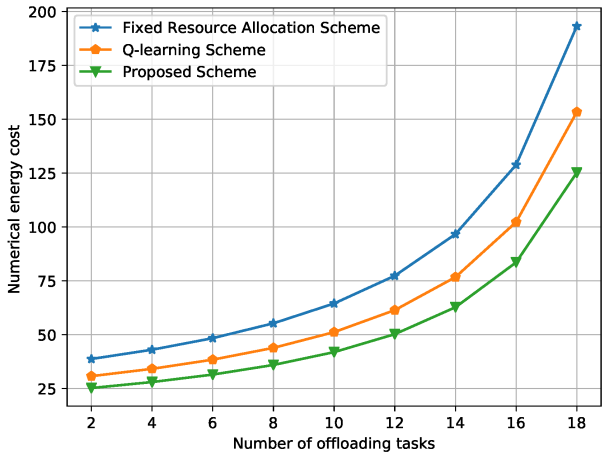


**FIGURE 7.** The energy consumption of the offloading task with different schemes.

It can be observed that the DRL achieves higher performance as compared with other benchmarks. In the DRL approach, the agent exploits the optimal resources and shows the best performance in achieving cumulative rewards, which leads to the highest energy efficiency. The proposed scheme achieves 12% more reward than the fixed resource allocation approach. While the Q-learning method gains 3% lower reward as compared with the proposed scheme.

We have selected the eNB and radio resources without considering the optimality for a fixed resource allocation algorithm. As a result, it shows lower performance in achieving cumulative rewards. The Q-learning based algorithm shows a higher performance than that of the fixed resource allocation approach. However, it yields lower efficiency in achieving rewards as compared with the DRL approach. It is because of traversing past each state of Q-learning. The Q-learning approach requires every previous state to be revisited. This catalyzes the unexpected power consumption and accumulates fewer rewards.

In Fig. 7. The total energy consumption of three different schemes is compared. The proposed scheme outperforms both the Fixed Resource Allocation and Q-learning Schemes and consumes the lowest energy, which is approximately 125mW. The Fixed Resource Allocation Scheme consumes the highest amount of energy, which is around 190mW. Whereas the Q-learning consumes comparatively lower energy that is approximately 155mW.

## VI. CONCLUSION

In this paper, we minimize the energy consumption of the machine type communication devices (MTCDs) using the deep reinforcement learning (DRL) approach. MTC edge computing architecture utilizes the DRL technique to improve the offloading process of MTCDs. Our problem is formulated under power constraints and QoS requirements. We compared the DRL, Q-learning, and fixed resource-based allocation machine learning techniques. Among these techniques, the simulation results prove that the DRL based approach outperforms in achieving cumulative rewards and is most efficient in minimizing the energy consumption of the MTCDs.

## REFERENCES

[1] D. Wang, D. Chen, B. Song, N. Guizani, X. Yu, and X. Du, "From IoT to 5G I-IoT: The next generation IoT-based intelligent algorithms and 5G technologies," *IEEE Commun. Mag.*, vol. 56, no. 10, pp. 114–120, Oct. 2018.
[2] A. Ali, W. Hamouda, and M. Uysal, "Next generation M2M cellular networks: Challenges and practical considerations," *IEEE Commun. Mag.*, vol. 53, no. 9, pp. 18–24, Sep. 2015.
[3] T. Salam, W. U. Rehman, and X. Tao, "Data aggregation in massive machine type communication: Challenges and solutions," *IEEE Access*, vol. 7, pp. 41921–41946, 2019.
[4] Y. Mehmood, N. Haider, M. Imran, A. Timm-Giel, and M. Guizani, "M2M communications in 5G: State-of-the-Art architecture, recent advances, and research challenges," *IEEE Commun. Mag.*, vol. 55, no. 9, pp. 194–201, Sep. 2017.
[5] Z. Dawy, W. Saad, A. Ghosh, J. G. Andrews, and E. Yaacoub, "Toward massive machine type cellular communications," *IEEE Wireless Commun.*, vol. 24, no. 1, pp. 120–128, Feb. 2017.
[6] A. Kiani and N. Ansari, "Edge computing aware NOMA for 5G networks," *IEEE Internet Things J.*, vol. 5, no. 2, pp. 1299–1306, Apr. 2018.
[7] W. U. Rehman, T. Salam, A. Almogren, K. Haseeb, I. Ud Din, and S. H. Bouk, "Improved resource allocation in 5G MTC networks," *IEEE Access*, vol. 8, pp. 49187–49197, 2020.
[8] S. Li, N. Zhang, S. Lin, L. Kong, A. Katangur, M. K. Khan, M. Ni, and G. Zhu, "Joint admission control and resource allocation in edge computing for Internet of Things," *IEEE Netw.*, vol. 32, no. 1, pp. 72–79, Jan. 2018.
[9] M. De Donno, K. Tange, and N. Dragoni, "Foundations and evolution of modern computing paradigms: Cloud, IoT, edge, and fog," *IEEE Access*, vol. 7, pp. 150936–150948, 2019.
[10] N. Li, C. Cao, and C. Wang, "Dynamic resource allocation and access class barring scheme for delay-sensitive devices in machine to machine (M2M) communications," *Sensors*, vol. 17, no. 6, p. 1407, 2017.
[11] Q.-V. Pham, L. B. Le, S.-H. Chung, and W.-J. Hwang, "Mobile edge computing with wireless backhaul: Joint task offloading and resource allocation," *IEEE Access*, vol. 7, pp. 16444–16459, 2019.
[12] H. Cao and J. Cai, "Distributed multiuser computation offloading for cloudlet-based mobile cloud computing: A game-theoretic machine learning approach," *IEEE Trans. Veh. Technol.*, vol. 67, no. 1, pp. 752–764, Jan. 2018.
[13] Z. Xu, Y. Wang, J. Tang, J. Wang, and M. C. Gursoy, "A deep reinforcement learning based framework for power-efficient resource allocation in cloud RANs," in *Proc. IEEE Int. Conf. Commun. (ICC)*, May 2017, pp. 1–6.
[14] M. Min, L. Xiao, Y. Chen, P. Cheng, D. Wu, and W. Zhuang, "Learning-based computation offloading for IoT devices with energy harvesting," *IEEE Trans. Veh. Technol.*, vol. 68, no. 2, pp. 1930–1941, Feb. 2019.
[15] B. Li, Z. Fei, J. Shen, X. Jiang, and X. Zhong, "Dynamic offloading for energy harvesting mobile edge computing: Architecture, case studies, and future directions," *IEEE Access*, vol. 7, pp. 79877–79886, 2019.
[16] Y. He, C. Liang, Z. Zhang, F. R. Yu, N. Zhao, H. Yin, and Y. Zhang, "Resource allocation in software-defined and information-centric vehicular networks with mobile edge computing," in *Proc. IEEE 86th Veh. Technol. Conf. (VTC-Fall)*, Sep. 2017, pp. 1–5.
[17] Y. He, N. Zhao, and H. Yin, "Integrated networking, caching, and computing for connected vehicles: A deep reinforcement learning approach," *IEEE Trans. Veh. Technol.*, vol. 67, no. 1, pp. 44–55, Jan. 2018.
[18] C.-C. Lin, D.-J. Deng, and C.-C. Yao, "Resource allocation in vehicular cloud computing systems with heterogeneous vehicles and roadside units," *IEEE Internet Things J.*, vol. 5, no. 5, pp. 3692–3700, Oct. 2018.
[19] X. Wang, X. Wei, and L. Wang, "A deep learning based energy-efficient computational offloading method in Internet of Vehicles," *China Commun.*, vol. 16, no. 3, pp. 81–91, Mar. 2019.
[20] Y. Wang, L. Wu, X. Yuan, X. Liu, and X. Li, "An energy-efficient and deadline-aware task offloading strategy based on channel constraint for mobile cloud workflows," *IEEE Access*, vol. 7, pp. 69858–69872, 2019.
[21] S. Li, Y. Tao, X. Qin, L. Liu, Z. Zhang, and P. Zhang, "Energy-aware mobile edge computation offloading for IoT over heterogenous networks," *IEEE Access*, vol. 7, pp. 13092–13105, 2019.

[22] Y. Cui, Y. Liang, and R. Wang, "Resource allocation algorithm with multi-platform intelligent offloading in D2D-enabled vehicular networks," *IEEE Access*, vol. 7, pp. 21246–21253, 2019.
[23] M. Eisen and A. Ribeiro, "Large scale wireless power allocation with graph neural networks," in *Proc. IEEE 20th Int. Workshop Signal Process. Adv. Wireless Commun. (SPAWC)*, Jul. 2019, pp. 1–5.
[24] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski, S. Petersen, C. Beattie, A. Sadik, I. Antonoglou, H. King, D. Kumaran, D. Wierstra, S. Legg, and D. Hassabis, "Human-level control through deep reinforcement learning," *Nature*, vol. 518, no. 7540, pp. 529–533, Feb. 2015.
[25] J. Li, H. Gao, T. Lv, and Y. Lu, "Deep reinforcement learning based computation offloading and resource allocation for MEC," in *Proc. IEEE Wireless Commun. Netw. Conf. (WCNC)*, Apr. 2018, pp. 1–6.
[26] Z. Xiong, Y. Zhang, D. Niyato, R. Deng, P. Wang, and L.-C. Wang, "Deep reinforcement learning for mobile 5G and beyond: Fundamentals, applications, and challenges," *IEEE Veh. Technol. Mag.*, vol. 14, no. 2, pp. 44–52, Jun. 2019.
[27] Z. Zhao, S. Bu, T. Zhao, Z. Yin, M. Peng, Z. Ding, and T. Q. S. Quek, "On the design of computation offloading in fog radio access networks," *IEEE Trans. Veh. Technol.*, vol. 68, no. 7, pp. 7136–7149, Jul. 2019.
[28] Y. Sun, M. Peng, and S. Mao, "Deep reinforcement learning-based mode selection and resource management for green fog radio access networks," *IEEE Internet Things J.*, vol. 6, no. 2, pp. 1960–1971, Apr. 2019.
[29] G. M. S. Rahman, M. Peng, S. Yan, and T. Dang, "Learning based joint cache and power allocation in fog radio access networks," *IEEE Trans. Veh. Technol.*, vol. 69, no. 4, pp. 4401–4411, Apr. 2020.
[30] M. Cheng, J. Li, and S. Nazarian, "DRL-cloud: Deep reinforcement learning-based resource provisioning and task scheduling for cloud service providers," in *Proc. 23rd Asia South Pacific Design Autom. Conf. (ASP-DAC)*, Jan. 2018, pp. 129–134.
[31] H. Zhu, Y. Cao, X. Wei, W. Wang, T. Jiang, and S. Jin, "Caching transient data for Internet of Things: A deep reinforcement learning approach," *IEEE Internet Things J.*, vol. 6, no. 2, pp. 2074–2083, Apr. 2019.
[32] T. Yang, Y. Hu, M. C. Gursoy, A. Schmeink, and R. Mathar, "Deep reinforcement learning based resource allocation in low latency edge computing networks," in *Proc. 15th Int. Symp. Wireless Commun. Syst. (ISWCS)*, Aug. 2018, pp. 1–5.
[33] M. Li, L. Yang, F. R. Yu, W. Wu, Z. Wang, and Y. Zhang, "Joint optimization of networking and computing resources for green M2M communications based on DRL," in *Proc. IEEE Global Commun. Conf. (GLOBECOM)*, Dec. 2019, pp. 1–6.

**ISRAR KHAN** received the M.Sc. degree in electronics and communication engineering from the Beijing University of Posts and Telecommunications, China, in 2016, where he is currently pursuing the Ph.D. degree in the major of information and communications engineering, under the supervision of Prof. X. Tao. His research interests are massive machine type communication, edge computing, energy optimization, data offloading, the Internet of Things (IoT), and 5G networks.

**XIAOFENG TAO** (Senior Member, IEEE) received the B.S. degree in electrical engineering from Xi'an Jiaotong University, Xi'an, China, in 1993, and the M.S. and Ph.D. degrees in telecommunication engineering from the Beijing University of Posts and Telecommunications (BUPT), Beijing, China, in 1999 and 2002, respectively. He is currently a Professor with BUPT and a fellow of the Institution of Engineering and Technology. He has authored or coauthored over 200 articles and three books in wireless communication areas. He focuses on 5G/B5G research. He is also the Chair of the IEEE ComSoc Beijing Chapter.

**G. M. SHAFIQUR RAHMAN** is currently pursuing the Ph.D. degree with the Key Laboratory of Universal Wireless Communications, Ministry of Education, Beijing University of Posts and Telecommunications, Beijing, China. His research interests include heterogeneous networks, cloud computing-based radio access networks, device-to-device communications, and the applications of stochastic geometry in wireless communications.

**WAHEED UR REHMAN** received the M.Sc. degree from the University of Westminster, London, U.K., in 2007, and the Ph.D. degree from the Beijing University of Posts and Telecommunications, China, in 2015.

He is currently working as an Assistant Professor with the Department of Computer Science, University of Peshawar. His current research interests include challenges concerning machine type communication networks, mmwave communication, the Internet of Things (IoT), and 5G networks. He also serves as a referee for many reputed international journals and conferences.

**TABINDA SALAM** received the M.Sc. degree (Hons.) in computer science and the M.S. degree from the Institute of Business and Management Sciences (IBMS), Peshawar, Pakistan, in 2004 and 2012, respectively, and the Ph.D. degree in communication and information engineering from the Beijing University of Posts and Telecommunications, Beijing, China, under the supervision of Prof. X. Tao, in 2018. She has been a Faculty Member with the Department of Computer Science, Shaheed Benazir Bhutto Women University, Peshawar, since 2007. Her research interests include caching and data offloading through device-to-device communication, social aware device-to-device communication, massive access management of machine type communication, and cognitive radios.