# Consumer-Centric Internet of Medical Things for Cyborg Applications based on Federated Reinforcement Learning

Prayag Tiwari, Abdullah lakhan, Rutvij H. Jhaveri, Tor-Morten Grønli

***Abstract*—The Internet of Medical Things (IoMT) is the new digital healthcare application paradigm that offers many healthcare services to users. IoMT-based emerging healthcare applications such as cyborgs, the combination of advanced artificial intelligence (AI) robots, and doctors performing surgical operations remotely from hospitals to patients in their homes. For instance, robot-based knee replacement procedures, and thigh medical care real-time performance monitoring systems are cyborg applications. The paper introduces the multi-agent federated reinforcement learning policy (MFRLP) indicated in mobile and fog agents based on the socket remote procedure call (RPC) paradigm. The goal is to design a consumer-centric cyborg-efficient training testing system that executes the overall application mechanism with minimum delays in the IoMT system. The study develops the RPC based on reinforcement learning and federated learning that adopts dynamic changes in the environment for cyborg applications. As a result, MFRLP minimized the training and testing in the mobile and fog environments by 50%, local processing time by 40%, and processing time by 50% compared to existing machine learning (ML) methods for cyborg applications. The code is publicly available at https://github.com/prayagtiwari/CIoMT**

***Index Terms*—Consumer-Centric, IoMT, Federated Learning, Reinforcement Learning, Healthcare.**

## I. Introduction

The Consumer-centric cyborg system is the technological motivation that improves the limitation of the human body via different healthcare technologies [1]. The term ”cyborg” refers to an organism integrated with a synthetic component or technology that relies on feedback and has improved capabilities. Cyborgs are robots with AI linked to the human body and healthcare servers to monitor healthcare. The Cyborg system enables technical efficiency to enhance the disabled human body’s growth with the help of technology and monitor healthcare data via different hospitals [2]. Cyborg agents connect with healthcare sensors, such as thigh servers, knee servers, and foot sensors, to perform different surgical healthcare operations for the patients [3]–[5]. In the IoMT system, cloud computing is vital in offering different healthcare computing and data-intensive services with different pricing models. Many cloud service providers, including Amazon, Google, Azure, and others, offer various services based on different models [6] and have been growing progressively in practice. IoMT is an emerging paradigm in which many healthcare applications have been designed to ease users in the healthcare practice. Consumer-centric cyborg combines different computing, such as embedded computing, software computing, and infrastructure computing, where applications have many different layers and components. For example, embedded systems such as Arduino UNO and raspberry pi are widely deployed boards to support the various healthcare sensors to generate data from the human body and offload data to the server for prediction [7].

(*Corresponding author: Prayag Tiwari, Rutvij H. Jhaveri*)

P. Tiwari is with the School of Information Technology, Halmstad University, Sweden (Email: prayag.tiwari@ieee.org)

A. Lakhan and Tor Morton are with the School of Economics, Innovation and Technology, and Kristiania University College, Oslo, Norway (Email: abdullahrazalakhan@gmail.com, Tor-Morten.Gronli@kristiania.no)

Rutvij H. Jhaveri is with the Department of Computer Science and Engineering, School of Technology, Pandit Deendayal Energy University, India (E-mail: rutvij.jhaveri@sot.pdpu.ac.in)

For instance, in many sports, players have been injured in the game and have left their careers due to injuries. These injuries are very complex, leading players to stop their careers in different sports. As a result, they lose their jobs and careers because their bodies have defaulted in parts, and they cannot play games in different sports. Therefore, the cyborg-based system can remove the disability of humanity in practical life [8].

The study [9] presented the sports healthcare system using psychology and ML, where users can predict and handle any issue during real-time applications. The prediction task is made on cloud computing to analyze the disease in the system. The study [10] suggested wearable biosensor-based healthcare monitoring systems for sports analytics based on different biological sensors. The study [11] suggested the federated learning system with enhanced feature extraction for human activity recognition in different sports. However, training, testing, and computational times are higher and not optimal in terms of time and resource consumption in the network.

The federated learning system is also used with enhanced feature extraction for human activity recognition [12]. The goal is to train and test healthcare sensor data on the different nodes and share it with the global nodes for processing. In the existing ML models, all the data are trained and tested on the centralized ML node leading to a higher delay for the prediction. The federated learning mechanism train and tests data on different nodes, whereas the centralized aggregated model can make predictions easily with the minimum processing delay. As a result, many healthcare applications can be processed simultaneously in the system. However, data security is still a big issue in these models. These works studies [13], [14] federated learning-based healthcare systems in which remote healthcare data are analyzed on different nodes and shared to the aggregated node for processing. The goal is to train and test with the security models and shared data to the aggregated centralized server for processing. Resource constraints (e.g., resource leakage and resource consumption) are the critical issues in these works. However, these methods exploited federated learning and minimized the delay but still suffered from resource leakage and resource consumption.

The resource-efficient reinforcement-enabled healthcare system suggested by these studies [15]–[17]. They tried to minimize resource consumption during cyborg data prediction, and simulation in the dedicated environment for data processing. Many algorithms are suggested, such as SARSA (state, action, reward, state, action), Q-learning, policy, and value function, to handle the prediction tasks in the distributed cyborg-enabled cloud network. All the above studies only optimized the fixed parameters (e.g., processing delay, service delay) for cyborg applications. Still, many parameters (e.g., processing and resource costs) need to be optimized for cyborg applications.

The study focuses on the consumer-centric IoMT system for cyborg applications based on federated reinforcement learning approaches. The goal is to run cyborg-based IoMT applications for different treatments and healthcare services on mobile fog networks. The cyborg offers various robotics-enabled services to patients and doctors to make the treatment in mobile fog networks. We summarize the main contributions of this paper as follows.

- The study designs the socket RPC-based mobile and fog agents in which cyborg applications can run with the different consumer-centric sensors at the mobile and fog nodes in the IoMT system.
- The goal is to design a cyborg-efficient training testing system that executes the overall application mechanism with minimum delays in the IoMT system.
- The paper introduces the MFRLP algorithm framework to adapt to the dynamic changes in the environment for cyborg applications.
- The study derived the mathematical model for the considered problem and implemented it in the simulation environment with optimal results.

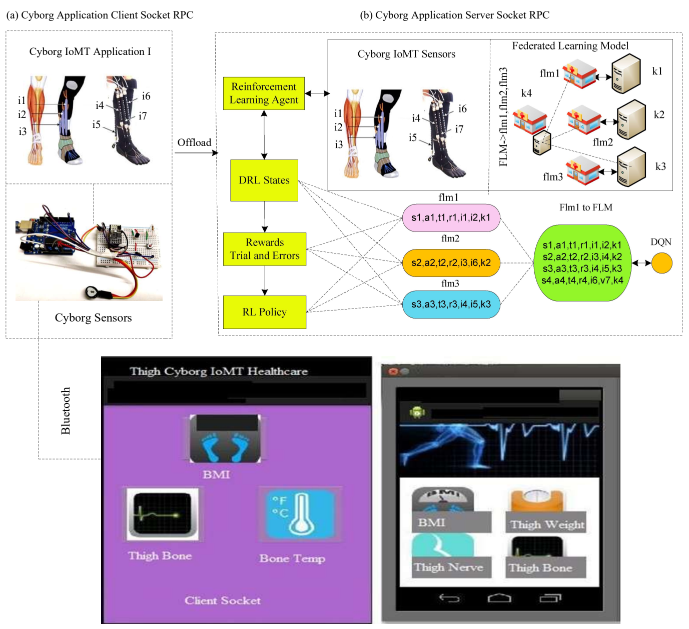

Fig. 1:
Cyborg Enabled IoMT System Based Deep Federated Schemes.

TABLE I: Mathematical Notation

| Notation | Description |
|---|---|
| $M$ | Number of mobile devices |
| $m$ | The particular mobile device |
| $\zeta_m$ | Particular speed of device |
| $\epsilon_m$ | Resource limitation of device |
| $op_m$ | Operating System of mobile |
| $ratio_m$ | Operating System of mobile |
| $K$ | Number of Fog Nodes |
| $\zeta_k$ | Particular speed of fog |
| $\epsilon_k$ | Resource limitation of fog |
| $ratio_k$ | Ratio of the fog node |
| $op_k$ | Operating System of fog |
| $I$ | Number of Healthcare independent tasks |
| $i$ | Particular healthcare task $i$ |
| $i_d$ | Deadline of task $i$ |
| $i_d - ratio_{i_d}$ | Ratio of deadline weight |
| $w_i$ | Workload of task $i$ in bits |
| $T_i^e$ | Required CPU to complete task process |
| $status_i$ | Status of the task |
| $D$ | Collection of datasets |
| $d$ | Particular dataset for the training |
| $FLM$ | Number of federated learning model |
| $flm1$ | Particular federated learning model |
| $L_i^e$ | Location execution of the task |
| $F_i^e$ | Execution time on the fog node for task $i$ |
| $C_i^e$ | Communication time of a task $i$ |
| $Train_i^e$ | Training and testing time of task on different nodes |

## II. Proposed System

In this section, the study devises the IoMT-based cyborg healthcare system using federated reinforcement learning in mobile fog cloud networks, as shown in Fig.1. The system is designed based on client and server sockets and implemented on different healthcare datasets. In the proposed system, (a) denotes the healthcare client, which is designed based on socket programming. All the applications are installed on the client side. The mobile applications are connected to the Android operating system and other thigh and knee monitor data sensors. They are connected to mobile devices via ESP 2286 Wifi Bluetooth module. All the workloads are offloaded to the cyborg IoMT server for processing. (b) The proposed system, as shown in Fig.1 composed of different components. For instance, reinforcement learning agents, DRL states, reward, trial and error, and RL policy based on DQN. All the workloads processing in the different states: $s1,a1,t1,r1,i1,...,S,A,T$. The cyborg healthcare datasets are trained and tested at different fog nodes such as $flm1 \leftarrow k1, flm2 \leftarrow k2, flm3 \leftarrow k3$ and shared to the aggregated node $k4$ based on the proposed federated learning method in the different hospital networks. The study defines the case study that designs the cyborg IoMT system based on the socket and RPC mechanisms. The socket RPC is the critical mechanism that offers different services and an application programming interface (API) to the practitioner to design the IoMT applications with large classes. The client socket is the mobile device where healthcare tasks are installed, where all tasks must be executed and offloaded to the fog nodes for processing in the system.

**Problem Formulation:** The study considers the different constraints, such as action, reward, and delay, as the numeric parameters. Therefore, this is considered integer linear programming as an optimization problem. The notation $I$ denotes the number of healthcare tasks, e.g., $\{i=1,...,I\}$ in the system. Each task has different tuples: $i = \{w_i, i_d, status_i\}$. Whereas $w_i$ is the workload determined in several megabytes (MB), $i_d$ is the deadline of the workload, and $status_i$ is the execution status of the workload in the system.

It considers the $M$ number of heterogeneous mobile nodes, e.g., $\{m=1,...,M\}$. Each mobile node has the following tuples: $m = \{\epsilon_m, \zeta_m, ratio_m\}$. The notation $\epsilon_m$ represents the mobile speed, $\zeta_m$ resources, and mobile remaining $ratio_m$ of the resource. Each node has a particular operating system $op_m$. For instance, each mobile device generally uses the Android X86 runtime environment with different emulators.

It also considers the $K$ number of heterogeneous fog nodes, e.g., $\{k = 1, \dots, K\}$. Each fog node has the following tuples: $k = \{\epsilon_k, \zeta_k, ratio_k\}$. The notation $\epsilon_k$ represents the fog speed, $\epsilon_k$ resources, and fog remaining $ratio_k$ of the resource in the system. Each node has a particular operating system $op_k$ in the research. For instance, each fog node generally used the X86 runtime environment with different emulators in the system for healthcare applications.

$$x_{i,m,k} = \begin{cases} x_{i,k} = 1, & \textit{Offload} \\ x_{i,m} = 2, & \textit{local-execute} \end{cases} \tag{1}$$

Equation (1) determines either workload offload or locally executed on a mobile device in the system. If the $x_{i,m} = 1$ shows that workload will offload to the fog nodes for the processing; otherwise, $x_{i,m} = 2$ means workload scheduled locally on the mobile device.

$$y_{i,k} = \begin{cases} y_{i,k} = 1, & \textit{Assigned} \\ several_{i,k} = 0, & \textit{Waiting} \end{cases} \tag{2}$$

Equations (1) and (2) have processing alignment and determine the workload assigned to the fog node or not for further execution. If the $y_{i,k} = 1$ that means the workload assigned to the fog node for the processing; otherwise, $y_{i,k} = 0$ workload waits till resource availability in the system. The study determines the processing time of all workloads of healthcare tasks on mobile devices in the following way.

$$L_i^e = \sum_{i=1}^{I} \sum_{m=1}^{M} x_{i,m} \frac{w_i}{\zeta_m} \times \epsilon_m - ratio_m. \tag{3}$$

Equation (3) determines the local execution time of the workload from the given resources and shows the remaining resources of the system. The study considers the two ways of communication time between a mobile node and fog nodes determined in the following way.

$$c_{w_i,m,k} = \begin{cases} y_{i,m} = 1, m \leftrightarrow k \\ y_{i,m} = 2, k \leftrightarrow K \end{cases} \tag{4}$$

Equation (4) determines the round-trip delays of communication across the nodes that are determined in equation (5).

$$C_i^e = \sum_{i=1}^{I} \sum_{d=1}^{D} \sum_{k=1}^{K} \sum_{flm=1}^{FLM} \frac{w_i}{bw_m} + \frac{w_i}{bw_k} \times c_{w_i,m,k}. \tag{5}$$

The study determined the training and testing data maps on the different nodes. Therefore, the training and testing models determine in the following way.

$$Training_i^e = \sum_{d=1}^{D} \sum_{k=1}^{K} \sum_{flm=1}^{FLM} flm1 \leftarrow \frac{d}{\zeta_k} \sum_{r=1}^{R} \times \epsilon_k - ratio_k + r. \tag{6}$$

Equation (6) determined the federated learning models $FLM$ on the different nodes and shared them with the aggregate node. Each trained model $flm$ is trained on the particular node $k$. Each node, including states, has a particular reward $r$ based on trained and tested data and given inputs of applications.

$$F_i^e = \sum_{i=1}^{I} \sum_{k=1}^{K} \frac{w_i}{\zeta_k} \times \epsilon_k - ratio_k. \tag{7}$$

Equation (7) determines the fog execution time of the tasks in the system. The total processing time of all tasks is determined in the following ways.

$$Total = L_i^e + c_i^e + Training_i^e + F_i^e, \quad \forall i = 1, ...I. \tag{8}$$

Equation (8) determines the total processing time of the tasks with their rewards accuracy in the network. The study makes the scheduling of the tasks with the following objective method in the system as convex optimization based on numeric integer linear programming constraints.

$$\min Total, \quad i = 1, ..., I. \tag{9}$$

Equation (9) determines the objective function of the study. Subject to

$$\sum_{i=1}^{I} \sum_{k=1}^{K} w_i \leq \epsilon_k. \tag{10}$$

All the requested workloads must be less the available resources in the system as determined in equation (10).

$$\sum_{i=1}^{I} \sum_{k=1}^{K} w_i \leq \leq i_d. \tag{11}$$

All the requested workloads must be executed under their deadlines on all nodes as determined in equation (11).

$$\sum_{i=1}^{I} w_i \leftarrow m, k, \quad \forall, m = 1, ..., k = 1, ..., K. \tag{12}$$

One task can be scheduled to one node at a time as determined in equation (12).

$$\sum_{k,m=1}^{K,M} w_i \leftarrow m, k, \quad \forall, i = 1, ...I. \tag{13}$$

One node can be scheduled one task at a time as determined in equation (13).

## III. Federated Deep Reinforcement Learning Enabled DQN-based Schemes

The study develops federated deep reinforcement learning-enabled policies and deep q-learning neural networks. Initially, the state solves this problem based on the Markov decision process, where the problem is divided into different states. Then creates an adaptive DQN and policy based on a model-free mechanism in which all states are independent and learn from trial/error. It balances exploration and exploitation by trying and seeing what works best. It devised a solution for the problem, which is MFRLP based on DQN. It has $S$ number of states, $A$ number of actions, $R$ number of commutative rewards, transition timestamp $T$ with the trail, and errors $TR, ER$ and determined in the following way.

**State:** It has set of autonomous states, $S = \{s = 1, ..., S\}$, where each state consists of different tuples $s = (s1, a1, t, tr, r, m, k, i)$. The main objective is to minimize the total time of IoMT cyborg application and maximize the commutative rewards as determined in equation (14).

$$\begin{aligned} S = \{s = 1, ..., S\}, \quad \forall i = 1, ..., I, \\ m = 1, ...M, k = 1, ..., K. \end{aligned} \tag{14}$$

**Action:** In this, it considers two types of input: mobile agent action and server agent action (e.g., servers). Therefore, the method has the mobile and server agents and is determined in the following way.

$$A = \sum_{a=1}^{A} a \leftarrow m \leftarrow i \in I, \quad m = 1, ..., M, i = 1, ..., I. \tag{15}$$

Equation (15) determines the mobile agent action on the workload during local processing and offloading. The server agent determines the action in the following way.

$$A = \sum_{a=1}^{A} a \leftarrow k \leftarrow i \in I, \quad k = 1,...,K, i = 1,...,I. \tag{16}$$

Equation (16) determines the server agent action in the system.

**MFRLP Algorithm Framework:** Algorithm 1 is the main framework of the study in which the system will show how to process the input to output via the different states.

**Algorithm 1:** Proposed MFRLP Algorithmic Framework

```
   Input  :{$M,K,I,D,FLM$
   Output  min$Total$;
   :
1  begin
2      foreach ($m=1$ to $M$) do
3          Prepare and Train Scheme;
4          $D \leftarrow FLM$;
5          Call Mobile Agent;
6          $I \leftarrow M$;
7          Server Agent;
8          $D \leftarrow I \leftarrow K$;
9      Optimize $Total$;
```

**Training and Testing based on Federated Learning:** Federated learning is the collaborative algorithm called an aggregated algorithm that trains the local healthcare datasets across the different fog nodes and is shared with the central fog nodes. For instance, cyborg applications offload their requests to the different fog nodes. The main fog node has all trained and tested models and is shared with the local fog nodes for optimal results. In this study, the centralized hospital-enabled fog node is the aggregated model based on the proposed Algorithm 2.

**Algorithm 2:** Federated Learning Enabled Training Scheme

```
   Input  :{$FLM,K,D,S$} ;
1  begin
2      $k$ particular node;
3      $batch$ Batch size;
4      $w \sim W$ Weights of node;
5      foreach ($D,K,FLM,S$) do
6          Determined the weights of nodes;
7          $k \leftarrow W = \{k \leftarrow w \leftarrow \epsilon_k, \zeta_k$;
8          Determined the loss function for aggregate node;
9          $k \leftarrow$
             $W[w,batch,k,flm,d,s] = LF(w,batch,k,flm,d,s)$;
10         Determined
             the federated communication based on equation (5);
11         Trained all datasets on different nodes on the given bench-
             mark based on a deep neural network and equation (6);
12         Update Trained and Tested model;
13         $k \leftarrow W[w,batch,k,flm,d]$
             shared to the $k1,k2,k3 \sim s1,s2,s3$;
14     End Inner;
15 End Main;
```

The federated learning-enabled method has the following steps in work.

- In the start, the algorithm has different inputs: healthcare datasets, weights, batch, nodes, training, and testing model.
- All the models inside federated learning are trained and tested based on a deep neural network on the different nodes and shared to the aggregated fog node $k \leftarrow W$ as defined in steps 1 to 5.
- From 6 to 12 steps determined the training and testing models of federated learning. First, different datasets were trained based on the deep neural network and shared with the aggregated model. Algorithm 2 determines the communication delay between nodes, which is the most crucial part of delay-sensitive healthcare applications.
- After training all models, Algorithm 2 shared the models with the local fog nodes, as shown in steps 12 to 13. The local dataset training is done in different states. Each trained and tested model is updated at different time intervals.

**Adaptive DQN-Enable Mobile Agent Policy:** Many uncertainties occur during the workload processing in the mobile and fog nodes. For instance, mobile resources could be leaked, the workload deadline could be missed, and the training/testing time of nodes could be high. Therefore, the study suggested adaptive scheduling schemes with the improved Q-learning scheme and divided the workload execution into mobile and server agents. The mobile and server agents will start the process only on one condition when the entire workload is to be executed under the given deadline. The proposed algorithm meets the resource requirements of requested workloads and their given quality of service requirements during execution on the different nodes. The memory adaptive replay of the mobile agent stores the local workload mechanism in the memory. For instance, (s1,a1,t1,r1,i1,i2,m1), (s2,a2,t2,r2,i3,i4,m1), (s3,a3,t3,r3,i4,i5,m1), (s4,a4,t4,r4,i6,i7,m1). The server agent will also store the data of the workload applications during execution. For instance, (s1,a1,t1,r1,i1,i2,k1), (s2,a2,t2,r2,i3,i4,k3), (s3,a3,t3,r3,i4,i5,k2), (s4,a4,t4,r4,i6,i7,k4). The main goal is to process future applications with the minimum errors and execute them with the highest commutative reward. The ultimate goal of Algorithm 3 is to schedule all workloads with the optimal objective function and the DQN optimal policy $\lambda^*$ from the input $i = 1 to I$ to output $I \leftarrow \lambda^* = Q(S,A,T,ER,RR)$. In this paper, the multi-agent deep q-networks (DQN) enabled method consists of main and network models. The Algorithm 3 has the following parameters.

- Main-Network: This network holds the execution model of all systems and has held the list of states in the network as defined in steps 1 to 5.
- Network weight exploiting Bellman Equation.
- Choose an optimal action based on Epsilon-Greedy Exploration Strategy.

Algorithm 3 has the following main steps in detail.

- The system takes the input from the mobile agent, e.g., $i,...,I,m = 1,...,M\}$. The application will execute on mobile and fog nodes in different states, as shown in Algorithm 3. The DQN is adaptive, where performances of resources and assigned workloads are executed with continuous performance in terms of a given deadline.
- All the workloads were initially scheduled at the mobile agent based on available resources before offloading to the fog nodes. The exploitative action added more reward to the memory

**Algorithm 3:** Adaptive DQN-Mobile and Server Agents Scheme

```
Input :$\{i,...,I,m=1,...,M\}$;
Output $\min L_i^e$;
:
1  begin
2      Initialize Q-Table[];
3      Discount-Factor[];
4      Initiate $s,a,t,r,tr \in TR, er \in ER$;
5      Initialize Replay $Z$;
6      Initialize main and target network;
7      $Total \leftarrow Q(S,A,T)$;
8      foreach $(i \in I \& S,A,T)$ do
9          Make the initial assignment based on equation (1);
10         Initialize the mobile agent state;
11         Discount-Factor[$s,a,t,m,i$]=$s,a,t,m,i,...,S,A,T$;
12         if $(Total_i \leq i_d)$ then
13             Initiate
                 the states, action, transition, errors, and rewards;
14             Determined
                 the local execution time based on equation (3);
15             Call
                 Algorithm 2 to determined trained federated models;
16             Random select probability policy $\lambda^*$;
17             $\lambda^* \leftarrow Q(S,A,T,TR,ER,I,M,K) \leftarrow Total$;
18             Determined
                 the nodes execution time based on equation (7);
19             Differentiate Discount-Factor[$s,a,t,m,i$];
20             Optimize
                 the objective function based on the loss function;
21             Q-Table[$Total$] $= Total \leftarrow \lambda \leftarrow er,tr$;
22             Determines the reward
                 and errors based on equation (15) and equation (16);
23             Call Global Exploration and Exploitation
                 Scheme to choose the best actions in different states;
24             Call Algorithm 4;
25             Replay Objective[Mobile,
                 Server]=$\lambda$; New Searching $\lambda'$ if $(\lambda \geq \lambda')$ then
26                 if $(Total \leftarrow i \sim I \leq i_d)$ then
27                     Determined
                         objective $total$ based on equation 9;
28                     Update Q-Table[$Total$];
29                     $\lambda' = Q(S,A,T) \leftarrow Total$;
30                     Call Algorithm
                         4 to search optimal objective function;
31                 End Searching;
32             End Solution;
33         End States;
```

replay when the current total delay was less than the deadline at the mobile level, as shown in Fig.1. Then, the action in the particular state will be taken based on the given weight and offloaded to the respective fog node for execution. All the tasks are offloaded in the stochastic form based on a Poisson process. The mobile agent mechanism starts from steps 2 to 13.

- Federated Learning Scheme: Algorithm 3 calls Algorithm 2 to find the aggregated model for the scheduling and execution in the server agent. Algorithm 3 determines the positive rewards and errors where resources and deadlines are widely missed in work. All the mechanism starts from steps 14 to 17.
- The study improved the q-learning method, which looks at stochastic rewards from when a workload is submitted until it is finished. It is necessary to adapt to the changes and improve q-table values until and unless meet the deadline and improve the q-learning as defined in steps 18-26.
- The adaptive and flexible states are designed in this current study, and previous and current data in each state are analyzed using a discount factor. The discount factor differentiates between the current state and previous states rewards in terms of minimum processing time and the deadline for all workloads, as defined in steps 11 and 19.
- Algorithm has two replay memories: mobile agent decision and server agent decision that are saved in different states, actions, transition, greedy scheduling mechanism, errors, and rewards. Both the memory replay helps to the agents what to do with the new workloads based on current performance in work.
- Objective Function: Algorithm 3 improves the objective function of the cyborg IoMT workloads in the mobile and fog nodes based on searching Algorithm 4. The searching algorithm executes all workloads until getting their results after completion on nodes.

**Explorative and Exploitive Searching Trade-Off Scheme:** The searching mechanism is important in maintaining the performance of distributed cyborg applications in geographically distributed fog nodes for execution. Searching is the way to control the exploration and exploitation of states and choose optimal actions for the objective function. There are different stages in the search mechanism, such as selection, expansion, execution, and backpropagation. Algorithm 4 traverses the entire resources and searches

**Algorithm 4:** Explorative and Exploitive Searching Trade-Off Scheme

```
Input :$\{ERR,RR,S\}$ ;
Output $\lambda^*$;
:
1  begin
2      foreach $(i \in I,S,A,RR,TR,M,K)$ do
3          Selection;
4          If($x_{i,m}$)==0
               $\lambda = Q(S,A,T) \leftarrow s1,a1,t1,r,er,i1 \leftarrow m1 \sim M$;
5          Expansion;
6          $\lambda' = Q(S,A,T) \leftarrow s2,a2,t2,r,er,i1 \leftarrow m1 \sim M$;
7          Execution;
8          $\lambda' = Q(S,A,T) \leftarrow s3,a3,t3,r,er,i1 \leftarrow m1 \sim M$;
9          backpropagation;
10         $\lambda{**} = Q(S,A,T) \leftarrow s4,a4,t4,r,er,i1 \leftarrow m1 \sim M$;
11         else
12             $\lambda = Q(S,A,T) \leftarrow s1,a1,t1,r,er,i1 \leftarrow k1 \sim K$;
13             Expansion;
14             $\lambda' = Q(S,A,T) \leftarrow s2,a2,t2,r,er,i1 \leftarrow k1 \sim K$;
15             Execution;
16             $\lambda' = Q(S,A,T) \leftarrow s3,a3,t3,r,er,i1 \leftarrow k1 \sim K$;
17             backpropagation;
18             $\lambda{**} = Q(S,A,T) \leftarrow s4,a4,t4,r,er,i1 \leftarrow k1 \sim K$;
```

the optimal resources to optimize the objective functions of all healthcare workloads and maximum overall commutative rewards. The selection step will allocate all the workload to the optimal resources. In the expansion, all the scheduled will be replaced from the existing execution process to the new process. In step 3, all the workloads are evaluated based on their deadlines. Finally, in step 4, Algorithm 4 searches the most optimal nodes to improve the overall

TABLE II: Experiment Configuration and Resource Specifications

| Configuration Values | Elements |
|---|---|
| Programming | Java, Python, XML |
| Sensors | Arduino knee arthroplasty |
| Sensor | Osteoarthritis Thigh Arduino |
| Agents | Mobile Socket Client, Server Socket Agent |
| $S$ | 100 |
| $A$ | 100 |
| $R$ | $1 \sim 5$ |
| $ER$ | $0.1 \sim 0.8$ |
| $TR$ | $1 \sim 5$ |
| Replay | 20,000 (MB) |
| $\lambda$ | $50 \sim 300$ |
| Round | $50 \sim 500$ |
| $i=1$ | Cyborg Thigh Text 1000 (MB) |
| $i=2,3,4$ | Cyborg Thigh images 2000 (MB) |
| $i=5,6$ | Cyborg Thigh Video 10,000 (.AVI) |
| $i=7$ | Cyborg Thigh Text 20,000 (MB) |
| $m=1$ | Android X86 16 GB,4 RAM |
| $m=2$ | Android X86 32 GB,8 RAM |
| $m=3$ | Android X86 48 GB,16 RAM |
| $m=4$ | Android X86 60 GB,24 RAM |
| $k=1$ | Laptop Cisco X86 5000 GB, 16 RAM |
| $k=2$ | Laptop Cisco X86 15000 GB, 32 RAM |
| $k=3$ | Laptop Cisco X86 25000 GB, 64 RAM |
| $k=4$ | Laptop Cisco X86 50000 GB, 96 RAM |
| Consumer-Centric Sensors | blood glucose monitoring, sensor fusion |

TABLE III: Training and Testing details on Cyborg Dataset

| d1 | d2 | d3 |
|---|---|---|
| Gryscope Thigh | Robotic-Assisted Knee Surgery | Navio System |
| 500(MB) | 1024 (MB) | 2048 (MB) |
| $flm1$ | flm2 | flm3 |
| 1000 (ms) | 2000 (ms) | 5000 (ms) |

rewards and minimize the overall delay of the cyborg applications. The proposed methodology, e.g., MFRLP, consists of different training and testing, scheduling, and searching mechanisms with $Olog(n \times n)$ complexity. The notation $olog(n \times n)$ is the training and testing of reinforcement learning in different nodes, searching and scheduling workloads in different states.

## IV. Performance Evaluation

In this section, the study initially uses socket-based mobile and fog agent systems based on the Java programming language. Then, for the cyborg application purpose, the study identifies the subject's gait angle of knee flexion using a sensor fusion inertial measurement unit and the accelerometer and gyroscope on mobile and fog nodes. Finally, all the simulation parameters are explained and defined in Table II. The cyborg datasets are available[1] as shown in Table III. The table consists of these parameters such as d1, d2, d3, Gryscope Thigh, Robotic-Assisted Knee Surgery, Navio System, 500(MB), 1024 (MB), 2048 (MB), flm1, flm2, flm3, 1000 (ms), 2000 (ms), and 5000 (ms). At the same time, dataset size and training/testing time are mentioned in Table III during simulation.

[1]https://github.com/liezliez/GaitAngle-using-SensorFusion-IMU/blob/main/Sudut_Gait_2021-04-10_13-02-28.xls

**Case Study of Cyborg Application Client Socket RPC and ANOVA statistical Implementation:** The study designed the proposed system based on socket RPC Java API. The system installed the different Cyborg IoMT applications via sensors and connected to the mobile devices via Bluetooth technology, as shown in Fig.1. The study implemented ANOVA relative percentage deviation for experimental result analysis for the simulation. These Navio Surgical System and Mako Robotic-Assisted Knee Surgery datasets are implemented in the simulation to train and test the models in the federated learning-enabled system. The work is efficient and needs attention to implement the cyborg-enabled system during knee surgery replacement with robots instead of doctors.

**Comparison of Reinforcement Learning and Federated Learning Schemes:** The applications start from the mobile agent and execute different applications in different states with the determined local processing time. The study defined the set of time transitions as $T$ on the x-axis, and the y-axis shows the processing time based on the initial state with the mobile agent, as shown in Fig.2. It is observed from Fig.2, that the local mobile agent only initiates applications when and has a lower local processing time before offloading for processing. The mobile agent's processing time

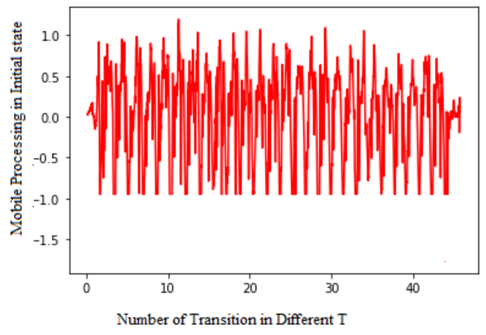

Fig. 2: Initial State Mobile Agent Performance.

increases while offloading all workloads to the servers for execution, as shown in Fig.3. The processing time reached near 3.0; it is higher for mobile agents compared to Fig.2. The mobile agent can only

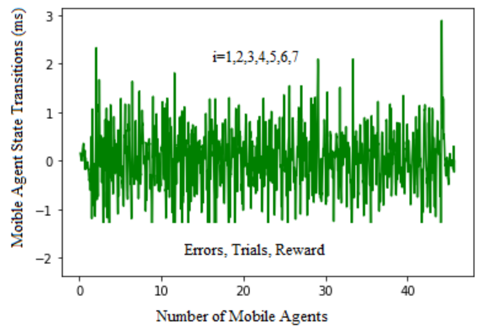

Fig. 3: Mobile Agent Enabled Offloading Based on Reinforcement Learning

initiate applications and offload them to the server for processing. However, server processing time consists of offloading time, training and testing time before execution further. Fig.4(a) shows the training and testing with the existing federated learning and reinforcement learning schemes [13]–[17] has a higher training time as compared to Fig.4(b). The main reason is that the existing ML model only centralized all datasets on a single node, such as $k4$, across different computing nodes, e.g., $k1$,$k2$,$k3$. The study devised the MFRLP algorithm framework in which all the datasets are trained across fog nodes and shared with the aggregated node without exchanging them during processing, as shown in Fig.4(b). The figures show that the overall training and offloading processing time was minimized with the federated reinforcement learning policy in different states compared to existing ML methods for cyborg applications.

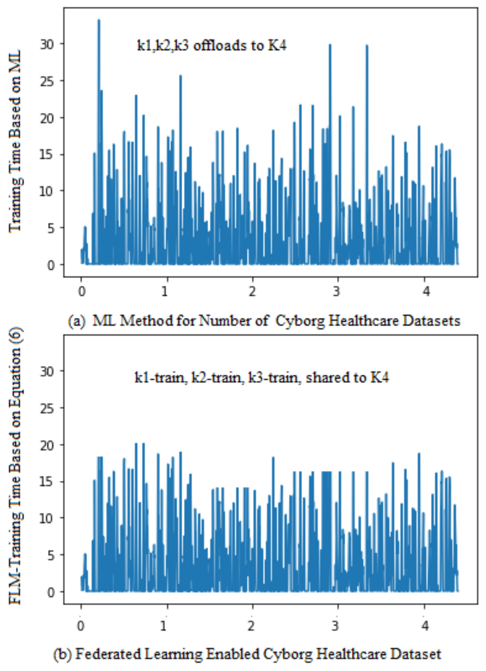

Fig. 4: Training Time of Cyborg Dataset on Heterogeneous Computing Nodes.

**Comparison of ML Algorithms:** The study implemented different reinforcement and federated learning algorithms for the performance evaluation of the cyborg healthcare workloads: the deep q-network reinforcement method, SARSA, and MFRLP. All algorithms are dynamic and learn from the environment based on trial and error, which improves the application's overall reward. Initially, all the workloads installed on the mobile devices are based on agents such as $m1$ and $m2$. Fig.5(a, b) shows the processing time of the particular mobile devices for all applications with and without transitions in the system. It is observed that the processing time with the transition is larger than the initial processing time in different states. The main reason is that during the transition of the workload from one state to another, it incurred extra time and depended upon the system's mobile processing speed and resources. The question is how the MFRLP outperformed all existing ML methods. The algorithm MFRLP divides the processing time into mobile agents and server agents. All the mobile agents exploited the replay memory to learn from trial and error, improve the overall execution based on current history, and predict the future implementation of all workloads. However, the existing DQN and SARSA only have one agent that takes action based on current data in different states. However, data processing in different states incurred more processing delays because there is a lot of complex processing in distributed applications. Therefore, it is a good choice that the starting mechanism and execution should be divided into two parts for the system's performance. Fig.5 motivates us that the knee robotics at the client

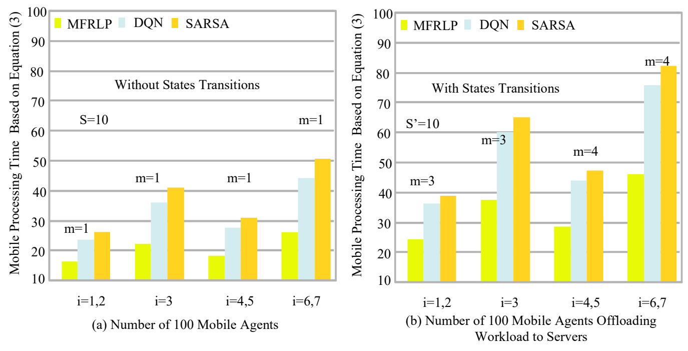

Fig. 5: Performances of Mobile Agents for Cyborg Applications.

side can execute the applications with minimum delays in the system. It has been observed that the centralized training and testing, and

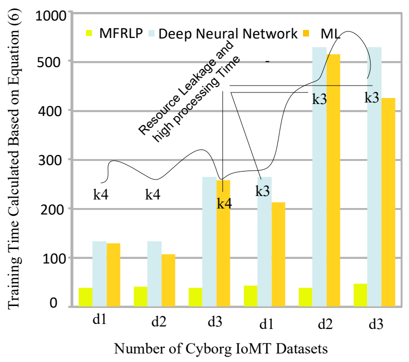

Fig. 6: Performances of Fog Nodes for Cyborg Applications.

execution of the existing ML methods lead to resource leakage and higher delays when they are executed workloads, as shown in Fig.6. Therefore, the MFRLP policy is more optimal and can improve the overall processing mechanism of the cyborg applications and minimize the overall delay of the cyborg applications, as shown in Fig.7.

## V. Conclusion

This work presents the consumer-centric IoMT system to run the cyborg applications based on a federated reinforcement learning framework in a real-world healthcare setting. The overall study provides the presented mathematical model, problem description, formulation, algorithm design, and simulation to show the entire system for cyborg applications. Based on the socket RPC paradigm, it introduces the MFRLP algorithm framework for mobile and fog agents. The goal is to create a cyborg-like training testing system that runs the entire application procedure with minimal system latency. Essential technologies such as reinforcement learning and federated learning are adapted to capture dynamic changes in the environment for cyborg applications. Compared to previous ML approaches for cyborg applications, MFRLP reduced training and testing time by 50%, local processing time by 40%, and 50%.

## References

[1] S. P. Mohanty and F. Pescador, "Introduction consumer technologies for smart healthcare," *IEEE Transactions on Consumer Electronics*, vol. 67, no. 1, 2021.

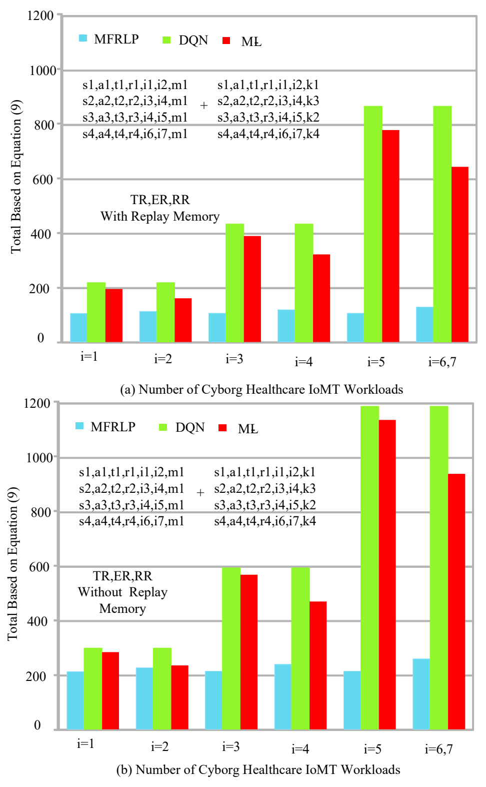

Fig. 7:
Total Performance of All Application with Different Methods.

[2] A. Fox, "The (possible) future of cyborg healthcare: Depictions of disability in cyberpunk 2077," *Science as Culture*, vol. 30, no. 4, pp. 591–597, 2021.

[3] J. Luo, Y. Li, M. He, Z. Wang, C. Li, D. Liu, J. An, W. Xie, Y. He, W. Xiao *et al.*, "Rehabilitation of total knee arthroplasty by integrating conjoint isometric myodynamia and real-time rotation sensing system," *Advanced Science*, vol. 9, no. 8, p. 2105219, 2022.

[4] V. P. Yanambaka, S. P. Mohanty, E. Kougianos, and D. Puthal, "Pmsec: Physical unclonable function-based robust and lightweight authentication in the internet of medical things," *IEEE Transactions on Consumer Electronics*, vol. 65, no. 3, pp. 388–397, 2019.

[5] A. M. Joshi, P. Jain, S. P. Mohanty, and N. Agrawal, "iglu 2.0: A new wearable for accurate non-invasive continuous serum glucose measurement in iomt framework," *IEEE Transactions on Consumer Electronics*, vol. 66, no. 4, pp. 327–335, 2020.

[6] G. Orive, N. Taebnia, and A. Dolatshahi-Pirouz, "A new era for cyborg science is emerging: the promise of cyborganic beings," *Advanced Healthcare Materials*, vol. 9, no. 1, p. 1901023, 2020.

[7] S. Bharati, P. Podder, M. R. H. Mondal, and P. K. Paul, "Applications and challenges of cloud integrated iomt," *Cognitive Internet of Medical Things for Smart Healthcare: Services and Applications*, pp. 67–85, 2021.

[8] M. A. Oliva, J. P. Borondo *et al.*, "Cyborg as a surgeon: A theoretical framework for cyborg acceptance in healthcare service," in *Societal Challenges in the Smart Society*. Universidad de La Rioja, 2020, pp. 57–70.

[9] S. Naik and E. Sudarshan, "Smart healthcare monitoring system using raspberry pi on iot platform," *ARPN Journal of Engineering and Applied Sciences*, vol. 14, no. 4, pp. 872–876, 2019.

[10] S. Ye, S. Feng, L. Huang, and S. Bian, "Recent progress in wearable biosensors: From healthcare monitoring to sports analytics," *Biosensors*, vol. 10, no. 12, p. 205, 2020.

[11] Z. Xiao, X. Xu, H. Xing, F. Song, X. Wang, and B. Zhao, "A federated learning system with enhanced feature extraction for human activity recognition," *Knowledge-Based Systems*, vol. 229, p. 107338, 2021.

[12] L. Sun and J. Wu, "A scalable and transferable federated learning system for classifying healthcare sensor data," *IEEE Journal of Biomedical and Health Informatics*, 2022.

[13] W. Y. B. Lim, S. Garg, Z. Xiong, D. Niyato, C. Leung, C. Miao, and M. Guizani, "Dynamic contract design for federated learning in smart healthcare applications," *IEEE Internet of Things Journal*, vol. 8, no. 23, pp. 16 853–16 862, 2020.

[14] J. Li, Y. Meng, L. Ma, S. Du, H. Zhu, Q. Pei, and X. Shen, "A federated learning based privacy-preserving smart healthcare system," *IEEE Transactions on Industrial Informatics*, vol. 18, no. 3, 2021.

[15] S. Tang and J. Wiens, "Model selection for offline reinforcement learning: Practical considerations for healthcare settings," in *Machine Learning for Healthcare Conference*. PMLR, 2021, pp. 2–35.

[16] M. Baucum, A. Khojandi, and R. Vasudevan, "Improving deep reinforcement learning with transitional variational autoencoders: A healthcare application," *IEEE Journal of Biomedical and Health Informatics*, vol. 25, no. 6, pp. 2273–2280, 2020.

[17] J. I. Guerrero, G. Miró-Amarante, and A. Martín, "Decision support system in health care building design based on case-based reasoning and reinforcement learning," *Expert Systems with Applications*, vol. 187, p. 116037, 2022.