electronics

MDPI

Article

Task-Offloading Strategy of Mobile Edge Computing for WBANs

Yuhong Li * and Wenzhu Zhang

School of Information and Control Engineering, Xi'an University of Architecture and Technology, Xi'an 710055, China; wzzhang@xauat.edu.cn
* Correspondence: liyuhong@xauat.edu.cn

Abstract: In recent years, mobile edge computing has become one of the popular methods to provide computing resources for the body area network, but existing research only considers the problem of minimizing the cost of offloading when solving the optimization problem of task-offloading, ignoring the trust problem of edge computing nodes, and offloading tasks on edge nodes may cause user information disclosure and reduce the quality of user experience. In response to this situation, this study aims to minimize the average user cost and designs a task-offloading strategy based on the D3QN (dueling double deep Q-network) algorithm in conjunction with the blockchain information security storage model. This strategy uses deep reinforcement learning algorithms to obtain the minimum average offloading cost of the system while considering user latency, energy consumption, and data protection conditions. The experimental simulation results show that compared to traditional schemes and other reinforcement learning-based schemes, this scheme can more effectively reduce the average cost of the system, and the average cost is reduced by 31.25% when reaching convergence. In addition, as the complexity of the model increases, this scheme can provide users with better experience quality, with 53.7% of the 1000 users having a very good experience quality.

Keywords: mobile edge computing; wireless body area network; task offloading; reinforcement learning; D3QN; blockchain; quality of experience

Citation: Li, Y.; Zhang, W. Task-Offloading Strategy of Mobile Edge Computing for WBANs. *Electronics* **2024**, *13*, 1422. https://doi.org/10.3390/electronics13081422

Academic Editors: Samiya Khan and Korhan Cengiz

Received: 12 March 2024
Revised: 3 April 2024
Accepted: 8 April 2024
Published: 9 April 2024

1. Introduction

With the rapid development of Internet of Things (IoT) technology and the popularization of 5G networks in recent years, the wireless body area network [1,2] (WBAN) has received attention from various sectors of society, including academia and healthcare, and has become a feasible solution for real-time monitoring of patient health [3]. The WBAN system is generally composed of portable medical devices based on biosensors, and the increasing number of medical devices has generated massive amounts of human physiological health data. However, WBAN devices are unable to provide a satisfactory quality of service (QoS) to users due to the small scale, limited resources, insufficient storage, and computing capabilities. The proposal of mobile edge computing (MEC) makes up for the shortcomings of WBAN devices in terms of energy consumption, resource storage, computing performance, etc. The MEC system offloads massive computing tasks in the WBAN scenario to edge servers, not only alleviating the high latency problem caused by cloud servers being far from terminals but also effectively improving the data processing capabilities of mobile terminals [4,5].

Y. Liao. et al. [6] proposed a new network architecture that combines WBAN and MEC to process wireless big data services. By integrating the access point with a remote radio head, the access point can perform latency-sensitive tasks while tasks with high computing resources can be offloaded to edge servers. REN J Y. et al. [7] focused on the real-time requirements of urgent tasks and designed a task-offloading method assisted by edge computing. Proposed an adaptive algorithm to optimize the delay, which met the low delay requirements of urgent tasks. Y. Liao. et al. [8] considered the limited computing resources of WBAN access points and proposed a task scheduling algorithm to offload

lower-priority tasks to edge servers for execution. In [9], a resource management scheme that minimizes the energy consumption of edge servers without affecting the quality of user experience in the WBAN. A computational task-offloading scheme based on a differential evolution (CTOS-DE) algorithm for WBAN can provide the best strategy for task-offloading in terms of total cost and load balance [10]. These tasks only consider offloading tasks to untrustworthy edge servers. The user information generated by WBAN devices has privacy and needs to be encrypted or protected according to user needs.

Reference [11] proposed a cloud server layer architecture based on blockchain technology for the WBAN to ensure system stability and patient data security. At the same time, a cipher policy attribute-based encryption (CP-ABE) scheme is adopted to ensure the security of patient physiological data. A WBAN architecture model based on blockchain technology in reference [12]. Authentication protocols and blind signature protocols between nodes are designed in a new WBAN model to ensure the security and reliability of blockchain data transmission systems in wireless network environments. In [13], a lightweight terminal device authentication scheme based on blockchain technology combined with a limited number of hash and XOR operations is suitable for WBAN. Xu et al. [14] developed a blockchain security mutual authentication scheme based on WBAN. This scheme has a more effective control effect on energy consumption and has broad application prospects. This work indicated that uploading medical data onto the blockchain for protection is feasible. However, it makes the inherently complex task scheduling problem of edge computing a more complex problem.

Due to the advantages of control decision-making and resource optimization, deep reinforcement learning (DRL) is widely considered to be an ideal tool for solving policy control problems in complex dynamic environments [15]. A joint optimization offloading and migration algorithm based on asynchronous dominance Actor Critical (A3C) was proposed in [16] to solve the migration and offloading problems in mobile edge computing systems in the context of WBANs. Zhang et al. [17] considered a task-offloading and time allocation algorithm called DNN-TOTA, which is based on deep learning and uses deep neural networks and order-preserving quantization (OPQ) methods to generate candidate offloading decision sets to solve the problems of limited energy resources and low computing power of micro sensor nodes in WBAN. In [18], a deep reinforcement learning-based frame aggregation and task-offloading approach (DQN-FATOA) were used to improve the throughput and overall utilization of the WBAN. Inspired by the above research work, this study designed a task-offloading strategy for MEC based on D3QN quality of experience (QoE) in the context of the WBAN to meet users' needs for low-cost and data protection. The specific research is as follows:

(1) A task-offloading and storage model for user QoE is proposed, which takes into account the cost of user task-offloading and data security issues. This model is used to find the optimal edge node for task-offloading to achieve maximum computing resources and reduce the impact of high task execution costs and data leakage to improve the QoE of users;

(2) A D3QN-based system average cost minimization algorithm is proposed to solve the task-offloading problem in the context of WBANs to make task-offloading more effective and utilize computing resources more efficiently;

(3) This study built different user experimental simulation environments and conducted simulation analysis on the algorithm proposed. Compared with solutions such as full local offloading, full edge offloading, random offloading, DQN offloading, and dueling DQN offloading, the proposed solution in this study has a more significant cost reduction for WBAN users and can provide users with a more satisfied QoE.

2. Related Works

Latency and security are the two most important performance considerations for users of WBANs. Although some researchers currently consider both aspects, the efficiency of

task-offloading algorithms is not high. However, if we think independently about these issues, the following studies can provide us with ideas.

2.1. Existing Research Works

For the application of edge computing of WBANs, some scholars began to study it several years ago. To meet the massive data access and personalized service requirements of WBANs constrained by multiple service quality parameters, Yuan et al. [19] proposed a two-stage potential game computation offloading strategy (TPOS) which considered the task priority and user priority of WBANs to optimize resource allocation. The goal of reference [20] is to improve the computing capacity of WBAN users by using a small coordinator mobile edge computing (C-MEC) server. Through this approach, system complexity, computational resources, and energy consumption are significantly transferred from the WBAN users to the C-MEC. Ning et al. [21] constructed an economically efficient IoMT household health monitoring system by dividing it into two sub-networks, namely internal WBAN and external WBAN. It has also proven the effectiveness of their proposed algorithm in terms of overall system average cost and the number of patients benefiting from MEC. In [22], an adaptive resource allocation scheme based on demand prediction solves the problem of adaptively adjusting resource allocation, timely detecting and responding to load changes. A collaborative computing architecture that can handle wireless big data applications was proposed in reference [23]. The access point (AP) of WBANs is integrated with the remote radio head (RRH) to perform high latency and low computational tasks, while the MEC server can handle low latency computationally intensive tasks.

A large number of encryption algorithms have been applied to ensure the security of user information in WBANs. The blockchain encryption scheme proposed in recent years has gradually been applied to encrypt user information in WBANs. Ren et al. [24] designed a task-offloading strategy for a centralized low latency, secure, and reliable decision method (LSRDM-EH) with strong emergency handling capabilities and a dual-layer multi-dimensional comprehensive security strategy based on blockchain. In [25], a task-offloading strategy based on blockchain and trust awareness (BBTAS), through which task-offloading requests are published as blockchain transactions and automatically and securely executed by smart contracts (SCs). A blockchain based on a private WBAN platform to assist wearable IoT devices in healthcare services was proposed in [26]. The platform implements a distributed architecture using WBAN to introduce decentralized configuration and ensure privacy between wearable IoT devices in the network. Son et al. [27] developed a secure authentication protocol for cloud-assisted remote medical information systems using blockchain for access control. This protocol adopted encryption based on ciphertext policy attributes (CP-ABE) to establish access control for health data stored in cloud servers and apply blockchain to ensure data integrity.

2.2. Research Gap

Table 1 summarizes the previously introduced studies. Most of the existing studies utilize the paradigm of MEC for achieving lower latency and higher efficiency of WBANs. Some of these studies use blockchain technology with the edge computing paradigm to ensure privacy security; however, few existing studies consider these factors and deep reinforcement learning (DRL). The novelty of this article comes from the paradigm of blockchain of WBANs and task-offloading with DRL of the proposed work.

Table 1. Features comparison of the optimization strategy of task-offloading in the WBAN.

Authors	Edge Computing	Blockchain	KPIs	Applications
Liao et al. [6,8]	√	×	QoS, task offloading successful ratio	General framework
REN et al. [7]	√	×	latency	real-time algorithm
Bishoyi et al. [9]	√	×	energy consumption	management scheme
Zhang et al. [10,17]	√	×	total cost, load balance	CTOS-DE, DNN-TOTA
Deng et al. [11]	×	√	stability, security	CP-ABE
Xiao et al. [12]	×	√	security, reliability	architecture model
Wang et al. [13]	×	√	security	safety protocol
Xu et al. [14]	×	√	security, energy consumption	security scheme
Yuan et al. [16,18]	√	×	throughput, overall utilization	A3C, DQN-FATOA
Yuan et al. [19]	√	×	resource allocation	TPOS
Liao et al. [20,23]	√	×	computing capacity, latency	system modeling
Ning et al. [21]	√	×	system average cost	system modeling
Ren et al. [24,25]	×	√	latency, security, reliability	LSRDM-EH, BBTAS
Baucas et al. [26]	×	√	security	private platform
Son et al. [27]	×	√	data integrity	CP-ABE
Proposed work	√	√	system average cost, security, QoE	D3QN

3. Network Architecture and System Model

3.1. Network Architecture

Here, we proposed a multi-user edge computing network architecture of WBANs, as shown in Figure 1. The proposed network architecture includes a WBAN and a hospital consortium chain network connected by edge servers. Each hospital is controlled by the nearest edge server that performs data tasks for offloading WBAN devices and stores them on the edge blockchain. The system includes a set of edge servers $\boldsymbol{M} = \{1, 2, \ldots, M\}$, each of which is controlled M by the hospital HP_m. We assume that each user $\boldsymbol{J} = \{1, 2, \ldots, J\}$ is surrounded by a set of WBAN devices $\boldsymbol{N} = \{1, 2, \ldots, N\}$. In the proposed scheme, we consider all the tasks that a WBAN device needs to perform as a set. And there are several subtasks in the task set, which are further divided into continuous tasks and independent tasks. We will first preprocess the task to determine whether the local device's resources can be executed and choose whether to offload.

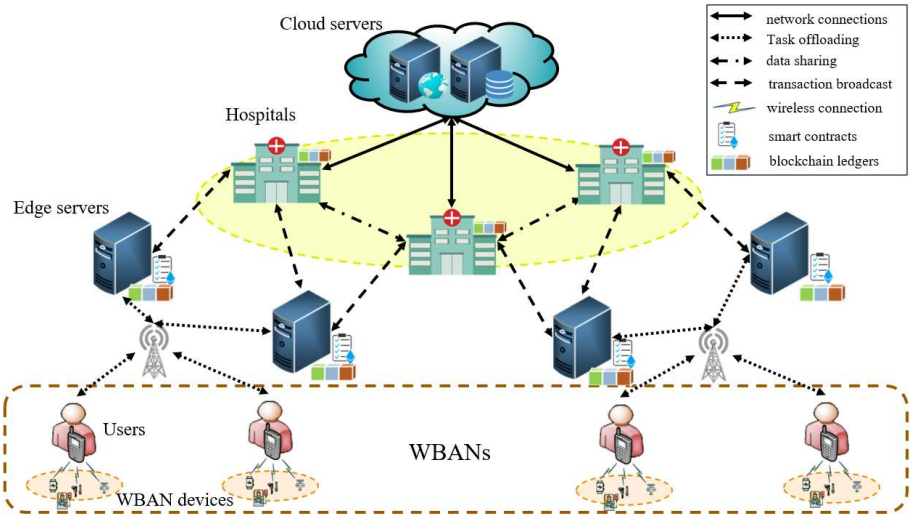


Figure 1. Network architecture.

3.2. Task Model

This section introduces the task model of our environment, with a focus on the hospital's diversion mode. According to QoE requirements (i.e., latency, energy consumption, and data security), each user's data tasks can be executed on edge servers through task-offloading or on local WBAN devices.

In addition, each WBAN device n runs an application that includes multiple different computing tasks k. These tasks n_k can be executed in these local WBAN devices and can

also be wirelessly transmitted to edge servers for execution through the network layer with blockchain records uploaded as transactions. In order not to lose generality, we represent $n \in \{1, 2, \ldots \boldsymbol{N}\}$ and $k \in \{1, 2, \ldots, \boldsymbol{K}\}$ respectively represent the set of WBAN devices and the set of intensive computing tasks. Available servers are represented by $\{0, 1\}$, where 0 refers to the WBAN devices themselves, while 1 refers to the edge servers. We calculate the offloading decision variable as x_{n_k}, where $\left(x_{n_k} = 0\right)$ represents the computation task k of the WBAN device n to be processed locally, and $\left(x_{n_k} = 1\right)$ represents the task n_k to be transmitted and processed on the edge servers. In addition, each computing task must be executed only once (i.e., on a local WBAN device or remote edge server).

In this study, we assume that each task n_k can be represented as

$$Y_{n_k} = \left\{B_{n_k}, C_{n_k}, D_{n_k}, R_j, T_{n_k}\right\}, \tag{1}$$

Among them, B_{n_k} is whether the task n_k is uploaded to the blockchain, C_{n_k} is the number of CPU cycles required to complete the task n_k, D_{n_k} is the input data size of the task n_k (in bits), R_j is the health data record of the user j, and T_{n_k} is the maximum allowable delay required to complete the task n_k (in seconds).

3.3. Communication Model

Each WBAN device uploads tasks to edge servers through wireless media, and the task-offloading time depends on its uplink transmission rate. According to the Shannon formula, the data rate of the uplink channel can be expressed as:

$$r_n = W \log_2\left(1 + \frac{p_n H_n}{\sigma^2}\right), \tag{2}$$

In the formula, W is the uplink channel bandwidth; p_n is the transmission power of the WBAN device n; H_n is the channel gain between the WBAN device n and the edge server m; and σ^2 shows the Gaussian white noise power.

3.4. Storage Model

Deploy a hospital within a region and link edge servers with WBAN users. The blockchain is controlled by edge servers. When a WBAN user performs a data request to an edge server, he creates a shared transaction and submits it to the blockchain so that the edge server can process the request and return the data. If the edge server is searching for data locally, the server will immediately send it to the user. Otherwise, other edge servers will be required to find the address of the requested data and send them to the user.

Users can choose to store data on WBAN devices, edge servers, or upload blockchain. In the blockchain storage model, the raw health data associated with the analysis results from the data offloading scheme is stored in the blockchain system on the edge server. Here, we analyzed a representative network with hospitals HP_m, edge servers $m(m \in M)$, WBAN devices $n(n \in N)$, and user $UID_j(j \in \mathrm{J})$. The data are stored on the blockchain through four steps through edge servers, as shown in Figure 2.

(1) The raw health data of the user UID_j collected from the WBAN device n is calculated and offloaded to the edge server m.

For simplicity, we assume that each user j has a health data record. Then, the edge server aggregates the raw data and calculation results and adds them to the data records identified by the user ID_j as

$$R_j = (rawdata \parallel computedResult), \tag{3}$$

(2) Edge server m encrypts this data into

$$C_j^{enc} \leftarrow \left(EncR_j, PKM_m\right), \tag{4}$$

Among them, PKM_m is the private key of MEC_m. For data security, these data will be uploaded to the blockchain storage node located at the top of the edge server.

(3) Blockchain storage nodes will use hash functions H_b to generate their encrypted hash values for encrypted data

$$h_j = H_b\left(C_j^{enc}, timestamp\right), \tag{5}$$

Here, we keep the hash values in the smart contracts on the edge server for quick data lookup rather than relying on classic distributed hash tables.

(4) The data records in the blockchain nodes on the edge server m are synchronized with other blockchain nodes in the hospital consortium chain through P2P networks for global data sharing. Edge servers m are also added $(h_j, UID_j, PKM_m, timestamp)$ as transactions and broadcasted to the hospital consortium chain network

$$MEC_m \rightarrow *: (h_j, UID_j, PKM_m, timestamp), \tag{6}$$

Other edge servers will receive transactions and extract offloading information h_j, UID_j, which will then be stored on their blockchain through smart contracts.

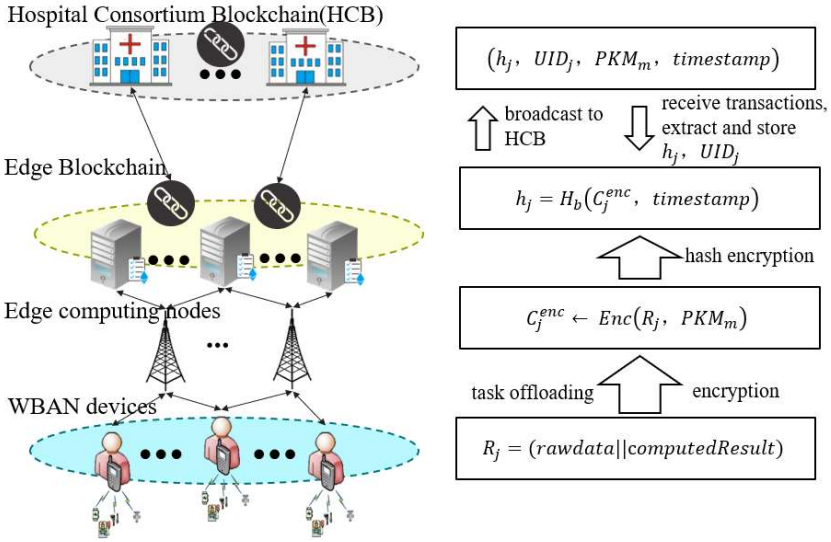


Figure 2. Data blockchain storage process.

3.5. Calculation and Offloading Model

In the existing literature, there is a lack of joint consideration of important QoE indicators in health data offloading, including processing time, energy consumption, memory usage, and data security issues. Due to the limitations of the current plan, we have considered these factors and formulated the issue of offloading health data. Two computational models were considered: local execution and edge execution.

(1) Local execution: $x_{n_k} = 0$ is local execution; we assume that WBAN devices may have different computing capabilities. In addition, the computing tasks of WBAN devices will be processed and stored locally. Meanwhile, the overall energy and time consumed can be calculated as follows:

$$E_{n_k}^{local} = \xi_n c_{n_k}, \tag{7}$$

$$T_{n_k}^{local} = \frac{c_{n_k}}{f_n^{local}}, \tag{8}$$

In the formula, ξ_n and f_n^{local}, respectively, represent the CPU cycle energy consumption and computing power (i.e., the number of CPU cycles per second) of the WBAN device n.

(2) Edge execution: $x_{n_k} = 1$ is edge execution, the computing tasks of WBAN devices will be entirely processed on edge servers. In addition, users can choose whether to

upload blockchain to protect data, $b_{n_k} = 0$ indicating that the task n_k is not uploaded to the blockchain and stored on the edge server while $b_{n_k} = 1$ indicating that the task n_k is uploaded to the blockchain. Based on the transmission and computation time of edge execution, as well as the time for uploading the blockchain, the overall time consumption can be calculated as follows:

$$T_{n_k}^{edge} = \frac{d_{n_k}}{r_n} + \frac{c_{n_k}}{f_n^e} + b_{n_k} T_{n_k}^{block}, \tag{9}$$

Among them, f_n^e represents the computing power allocated to the edge server of the WBAN device n; $T_{n_k}^{block}$ is the total time for the task n_k to upload the blockchain, and is represented as

$$T_{n_k}^{block} = T_{n_k}^{g} + T_{n_k}^{d}, \tag{10}$$

Among them, $T_{n_k}^{g}$ is the average time required for the blockchain system to generate new blocks; $T_{n_k}^{d}$ is the time consumption of data transmission. In this study, we have chosen a practical Byzantine fault-tolerant (PBFT) as the consensus mechanism in the proposed blockchain system according to reference [28]. And $T_{n_k}^{d}$ can be calculated as

$$\begin{aligned} T_{n_k}^{d} = \frac{1}{P}[&\min\left\{\frac{MS_b}{r_{n_m,n'_m}}, T_{\lim}\right\} + \min\left\{\underset{n_i \neq n_m, n'_m}{max} \frac{MS_b}{r_{n_m,n_i}}, T_{\lim}\right\} \\ &+\min\left\{\underset{n_i \neq n_j \neq n_m}{max} \frac{MS_b}{r_{n_i,n_j}}, T_{\lim}\right\} + \min\left\{\underset{n_i \neq n_j}{max} \frac{MS_b}{r_{n_j,n_i}}, T_{\lim}\right\} + \min\left\{\underset{n_i \neq n_m}{max} \frac{MS_b}{r_{n_m,n_i}}, T_{\lim}\right\}] \end{aligned}, \tag{11}$$

Among them P is the overall size of the block; S_b represents the number of bytes contained in each block; T_{lim} represents the average time it takes for the block producer to create a new block; and r_{n_i,n_j} represents the data transmission rate of the link between each pair of WBAN devices.

Intelligent WBAN devices also expect to receive the final result of transactions within a limited time for delay-tolerant inter-device communication of WBANs to meet their latency requirements. Therefore, we assume that a block should be published and validated within multiple consecutive block intervals, and this constraint should satisfy the following:

$$T_{n_k}^{block} \leq \rho \cdot T_g, \tag{12}$$

The ρ number of block intervals should be satisfied $\rho > 1$.

In this study, the computing resources of edge servers were shared among all WBAN devices during offloading.

In addition, we note that the computing power F of edge servers allocated to WBAN devices during offloading should be limited by the following constraints:

$$\sum_{n=1}^{N} \sum_{k=1}^{K} x_{n_k} f_n^e \leq F, \tag{13}$$

In addition, based on communication time, the overall energy consumption of edge execution can be calculated as follows:

$$E_{n_k}^{edge} = p_n \frac{d_{nk}}{r_n} + b_{n_k} E_{n_k}^{block}, \tag{14}$$

We define $E_{n_k}^{block}$ as the energy consumption of uploading task n_k to the blockchain, which is measured through mobile measurement tools.

Subsequently, the total energy and time consumed in executing the task n_k can be calculated as follows:

$$\mathrm{T}_{n_k}^{total} = (1 - x_{n_k}) T_{n_k}^{local} + x_{n_k} T_{n_k}^{edge}, \tag{15}$$

$$\mathrm{E}_{n_k}^{total} = (1 - x_{n_k}) E_{n_k}^{local} + x_{n_k} E_{n_k}^{edge}, \tag{16}$$

Finally, the main goal is to minimize the time and energy costs of each WBAN device to the greatest extent possible in the proposed framework. Therefore, the system average cost of task execution for total WBAN devices can be modeled as a linear combination of latency and energy consumption, which can be expressed as:

$$Z_{n_k}^{total} = \gamma \mathrm{T}_{n_k}^{total} + (1-\gamma)\mathrm{E}_{n_k}^{total}, \tag{17}$$

In the formula, $\gamma \in [0,1]$ represents the latency and energy consumption parameter weights of the task n_k, which can be adjusted according to the user's QoE.

3.6. QoE Model

In a WBAN, the QoE of users is jointly affected by service latency and energy consumption. While Z_{qoe}^{total} in Equation (18) is equal to the system average cost $Z_{n_k}^{total}$ in Equation (17). According to the QoE mapping function [29] shown in Figure 3 and Equation (18), the satisfaction level of users can be obtained.

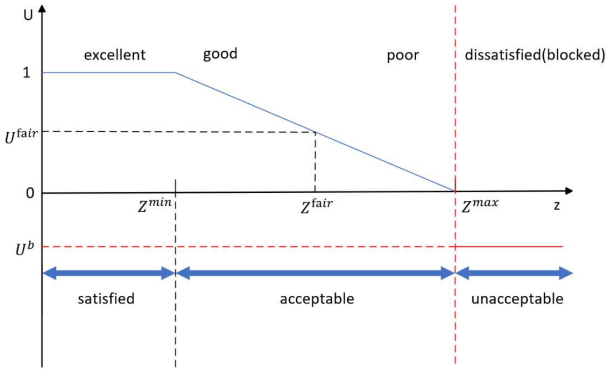


Figure 3. QoE mapping function.

Then, the system average cost is mapped to one of the practical levels based on the QoE function with predefined thresholds, named $Z_{qoe}^{\min}$ and $Z_{qoe}^{\max}$. The mapping function, as shown in the figure, is defined as

$$U_i = \begin{cases} 1, \text{if } z_i \le Z_{qoe}^{\min} \\ \frac{Z_{qoe}^{\max} - z_i}{Z_{qoe}^{\max} - Z_{qoe}^{\min}}, \text{if } Z_{qoe}^{\min} < z_i \le Z_{qoe}^{\max}, \\ U^b, \text{otherwise} \end{cases} \tag{18}$$

Note that $Z_{qoe}^{\min}$ and $Z_{qoe}^{\max}$ are determined by measuring the average opinion score. We also define an optional point $Z_{qoe}^{\text{fair}} \in (Z_{qoe}^{\min}, Z_{qoe}^{\max})$ from which users feel a significant decrease in service experience.

3.7. Problem Statement

This section presents a multi-user model with multi-task computation offloading and resource allocation in a blockchain-based WBAN system. Here, minimizing the time and energy consumption to complete tasks is the main objective of the model. The summary of our constraint optimization problem is as follows:

$$\begin{gathered}\min_{x} \sum_{n=1}^{N} \sum_{k=1}^{K} Z_{n_k} \\ s.t. \\ \mathrm{E}_{n_k}^{total} - E_{n_k}^{local} \leq 0, C1 \\ \mathrm{T}_{n_k}^{total} - T_{n_k}^{local} \leq 0, C2 \\ \mathrm{T}_{n_k}^{total} \leq T_{n_k}, C3 \\ \sum_{n=1}^{N} \sum_{k=1}^{K} x_{n_k} f_n^e \leq F, C4 \\ \sum_{n=1}^{N} \sum_{k=1}^{K} \left(1 - x_{n_k}\right) r_n + x_{n_k} r_n \leq R, C5 \\ \sum_{n=1}^{N} \sum_{k=1}^{K} T_{n_k}^{block} \leq \rho \cdot T_g, C6 \\ x_{n_k} \in \{0,1\}, C7 \\ b_{n_k} \in \{0,1\}, C8 \end{gathered} \tag{19}$$

The objective function aims to reduce the overall energy and time cost of WBAN devices by computing offloading. In addition, the $C1$ constraint and the $C2$ constraint, respectively, limit the consumption limits of energy and time. $C3$ represents the maximum allowable delay required to T_{n_k} complete the task n_k. Similarly, the limitations on CPU capacity and uplink data rate of edge servers are defined using constraints $C4$ and $C5$, while R represents the overall uplink data rate. Constraint $C6$ indicates that a block should be published and validated within multiple consecutive block intervals. Finally, constraints $C7$ and $C8$ ensure the binarization of the offloading decision variables and uploading blockchain variables.

The solution to the problem can be found by obtaining the optimal value of the task-offloading decision. Due to x being a binary variable, the feasible set for this problem is non-convex. Therefore, this is a non-convex integer problem and is classified as an NP-hard problem. Moreover, it is difficult to solve problems using mathematical analysis and formula derivation, especially when the number of WBAN devices is large. As the number of WBAN devices increases, the scale of the problem grows exponentially. Therefore, DRL is effectively utilized as a new field to solve such problems and obtain near-optimal solutions rather than using traditional optimization methods.

3.8. Task-Offloading and Storage Model for QoE

For ease of description, the following concepts are defined: task set of WBAN devices $y \in \{1, 2, \ldots, Y\}$; WBAN devices set $n \in \{1, 2, \ldots, N\}$; and edge servers set $m \in \{1, 2, \ldots, M\}$.

The execution process of Algorithm 1: When the task y is executed, check if there are any edge servers currently idle. If a server is idle, further determine whether to upload the blockchain. If block = True, schedule the task y to the edge server m for execution and upload to the blockchain; otherwise, schedule the task y to the edge server m for execution and storage. If there are no idle servers, the task will be executed on the local WBAN devices. The number of tasks, WBAN devices, edge servers, and QoE settings are set by ourselves. At the same time, the task attributes, WBAN device attributes, and edge server attributes in the model environment are randomly generated by the program within a certain range. After all tasks are completed and executed, use Equations (17) and (18) to output the user's satisfaction value.

Algorithm 1: Task-offloading and storage model for QoE

Input: number of tasks; number of WBAN devices; number of edge servers; QoE setting value; output: user QoE satisfaction value;
Initialize the system model environment;
for each task y in the WBAN device W:
  if there are available servers m in the edge server collection M then:
    if block = True then:
    the task y is scheduled to edge servers m for execution and stored in the blockchain;
    else:
    the task y is scheduled to edge servers m for execution and storage;
    end if
  else:
  the task y is executed and stored on a local WBAN device n;
  end if
Until the tasks Y are completed.

4. Task-Offloading Strategy Based on D3QN

In this section, due to the exponential increase in computational complexity and dimensionality of the decision-making process with the number of actions and states, we introduce and apply D3QN to address the formulaic problem discussed in this study. Based on the advantages of D3QN, the best decision can be made in inter-device communication supporting blockchain with delay-tolerant data to maximize system rewards.

4.1. DQN Algorithm and Variants

(1) Q-learning

Q-learning is a well-known reinforcement learning algorithm based on temporal difference [30], which uses the Q-function $Q\ (s,a)$ as the value estimation function instead of the value function. Therefore, the reward $Q'(s,a)$ value when the state s and the action a can be calculated as follows:

$$Q'(s,a) = E^*\left[\sum_{s\in S}\left(r\left(s,a,s'\right)\right) + \xi max Q'\left(s',a'\right)\right], \tag{20}$$

To solve the Q-learning problem, the Q-value for each step needs to be updated as follows:

$$Q(s,a) \leftarrow Q(s,a) + \delta\left[r + \xi \underset{a\in\mathcal{A}}{max} Q(s',a') - Q(s,a)\right], \tag{21}$$

Among them δ is the learning rate, which satisfies $\delta \in (0,1)$. And ξ represents the discount factor that will affect the current reward value in the future, and it satisfies $\xi \in (0,1)$.

(2) DQN

Based on the advantages of the Q-learning algorithm, the DQN algorithm was proposed and improved. Define the mean square deviation between the target Q value and the current Q value as the loss function $\mathcal{L}(\theta)$, expressed as

$$\mathcal{L}(\theta) = E\left[\left(r + \xi \underset{a'}{\max} Q(s',a';\theta^-) - Q(s,a;\theta)\right)^2\right], \tag{22}$$

Among them θ are the parameters of the neural network, while θ^- are the parameters for maintaining stable Q values and stable training process of the target DQN. According to Equation (22), DQN updates its network parameters by minimizing $\mathcal{L}(\theta)$.

(3) Double DQN

The double DQN algorithm has a structure similar to DQN, but it contains two Q networks. The Q value is updated as follows:

$$Q = r + \gamma Q_2\left(s', \arg\underset{a'}{max} Q_1(s',a,\theta_1), \theta_2\right), \tag{23}$$

Among them r are the instant rewards; the Q_1 is evaluation network; the Q_2 is target network; θ_1 and θ_2 are the evaluation network parameters and target network parameters; and γ is the attenuation factor. Defined a as an action; s' and a' is the next action and the next state.

(4) Dueling DQN

Dueling DQN is an improved algorithm of DQN, which can estimate q values with small variances and use greedy strategies to ensure sufficient exploration of action space. Dueling DQN divides the output layer into two parts: the value function network and the advantage function network. The linear combination of these two parts forms the final output of the Q network, represented as follows:

$$Q(s,a,\omega,\alpha,\beta) = V(s,\omega,\alpha) + \left(A(s,a,\omega,\beta) - \frac{1}{|\mathcal{A}|} \sum_{a' \in \mathcal{A}} A(s,a',\omega,\beta) \right), \tag{24}$$

In Equation (24) s is defined as a state; ω represents the parameters of the convolutional layer; α and β are two fully connected laminar parameters; $|\mathcal{A}|$ represents the number of selectable operations; $V(s,\omega,\alpha)$ is a state value function and $A(s,a,\omega,\beta)$ is an action advantage function.

4.2. D3QN Algorithm Framework and Neural Network

Considering the shortcomings of the above two algorithms, the D3QN algorithm was formed by combining the Double DQN and Dueling DQN, as shown in the framework in Figure 4. The D3QN algorithm inherits the double network structure of the Double DQN algorithm, including the evaluation network Q_1 and the target network Q_2. The evaluation network Q_1 uses the state information input from the experience buffer to select the next action $a^{next} = \arg\max_{a'} Q_1(s',a,\theta_1)$ for the target network Q_2 and output $Q_1(s,a,\theta_1)$ to the loss function $\mathcal{L}(\theta)$. The loss function $\mathcal{L}(\theta)$ updates the network parameters θ based on Equation (22). According to Equation (23), we evaluate a^{next} through the target network Q_2 to select the action $a\prime$ that maximizes the Q value. Then, the evaluation network Q_1 and the target network Q_2 combined the network structure of the Dueling DQN algorithm, while adding a state value function $V(s)$ and an action advantage function $A(s,a)$ between the last hidden layer and the output layer. And $V(s)$ only related to the state s and $A(s,a)$ is influenced by the state s and action a. The optimal action-value function $Q(s,a)$ is used to represent the linear combination of these two parts, and its specific calculation method is shown in Equation (24). This enables the D3QN algorithm to obtain the value of each action through the value of the state value function and the value of the action advantage function to produce more accurate results.

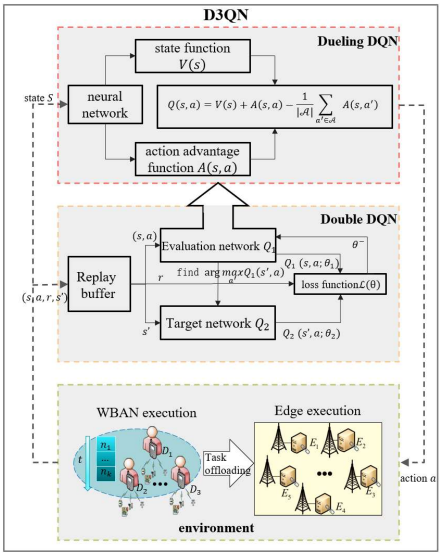


Figure 4. D3QN algorithm framework.

The neural networks of the evaluation network and the target network are shown in Figure 5. The evaluation network consists of one input layer and two hidden layers. The number of neurons in the input layer is the same as the dimension of state information. The input layer and hidden layer use the ReLU function as the activation function. The target network consists of two hidden layers and one output layer. The hidden layer uses the ReLU function as the activation function. And we added the state value function and action advantage function between the last hidden layer and the output layer.

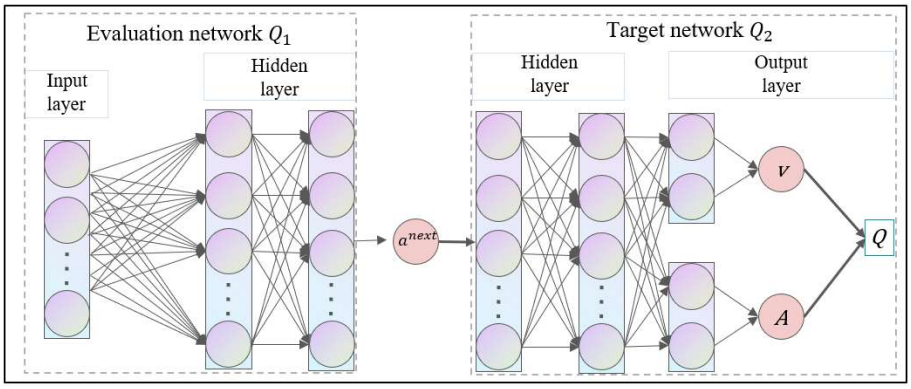


Figure 5. Neural network graph for D3QN.

4.3. MDP Mode of Task-Offloading Strategy Based on D3QN

To minimize the average cost of the system, the basic elements of reinforcement learning are set based on the model considered in this article. Here, we model the proposed problem as a standard Markov decision process (MDP) and solve this MDP problem using a task-offloading strategy based on D3QN. Namely, the state space, action space, and rewards are sequentially set, as shown below:

(1) Action space $a\ (t)$: Combination space, including decision-making for data cache selection and selection of computing nodes. In time slot t, to maximize system rewards, the intelligent agent will make decisions in the proposed network architecture. Formally, let A be the action vector, and composite actions $a\ (t) \in A$ can be represented as

$$a\ (t) = \left\{\ a_{ca}(t), a_{comp}(t)\right\}, \tag{25}$$

Among them, $a_{ca}(t)$ represents the selection of data storage and $a_{comp}(t)$ represents the selection of computing servers. For $a\ _{ca}(t) = \{\ 0, 1, 2\}$, Where 0 represents the decision of data storage in the WBAN device; 1 represents the decision of data caching to the edge computing server; 2 represents the decision of data storage in the blockchain. For $a_{comp}(t) = \{\ 0, m\}$, 0 indicates that the computing task will be executed on the local WBAN device. At the same time, m indicates that the computing task will be executed on the idle edge computing server M that is nearest to the WBAN device n.

(2) State space $s\ (t)$: In each time slot, agents dynamically monitor the network environment and collect system status. Set $\mathcal{S}$ to be the state space, and the system states for each time slot t satisfy $s\ (t) \in \mathcal{S}$, which can be expressed as

$$s(t) = \{e(t), d(t), o(t)\}, \tag{26}$$

Among them, $e(t)$ represents the edge server status. For $e(t) = \left\{e_{comp}(t), e_{ca}(t)\right\}$, $e_{comp}(t)$ and $e_{ca}(t)$, respectively, represent the computing and storage status of the edge server selected for the task. For $e_{ca}(t)$, it can be written as $e_{ca}(t) \in \{0, 1\}$, where $e_{ca}(t) = 0$ indicates that the server storage module is idle and data can be cached in it, and vice versa $e_{ca}(t) = 1$. Similarly, for $e_{comp}(t)$, can be written as $e_{comp}(t) \in \{0, 1\}$, where $e_{comp}(t) = 0$ indicates that the computing module of this server is idle and can perform computing tasks during this time slot, otherwise $e_{comp}(t) = 1$. In this study, we assume that the caching and computing state of

edge servers can be modeled as a Poisson distribution. It $d(t)$ is the input data volume y for a time slot t task and $o(t)$ is the maximum allowable delay required to complete the task in the time slot t. Meanwhile, we record $s\prime(t)$ as a new state after the action $a\prime(t)$.

(3) System reward $r(t)$: After each time slot t, the system will receive instant rewards $r(t)$ based on different actions $a(t)$. Generally speaking, the reward function should be positively correlated with the objective function, and the optimization objective of the problem is to obtain the maximum system reward. In Equation (17) of the second section, the goal is to minimize the weighted sum of delay and energy consumption, while the goal of D3QN is to maximize system rewards. Therefore, the reward function should be negatively correlated with the weighted sum of delay and energy consumption. We will define a unified system reward as follows:

$$RE(\pi) = -\sum_{t \in \mathbf{T}} Z_{n_k}(t), \tag{27}$$

Among them, π represents the optimization strategy, which is the set of operations for each time slot; T represents the set of all decision time slots; and Π represents the set of all possible strategies $\pi \in \Pi$.

Therefore, the optimization problem is explained as finding the optimal strategy $\pi\prime$ to maximize long-term cumulative system rewards $RE(\pi')$, which can be expressed as

$$\pi' = \underset{\pi \in \Pi}{\operatorname{argmax}} RE(\pi), \tag{28}$$

Based on the above settings, the algorithm flow for minimizing the average system cost based on D3QN proposed in this article is shown in Algorithm 2. The specific explanation is as follows. Step (2)–step (11) is the training process. In step (3), reset the system environment model. In step (5), the intelligent agent observes the state from the environment, including the edge server state, task size, and maximum allowable delay. Input s the D3QN network and output a the offloading storage strategy for user tasks. From step (6) to step (8), the intelligent agent updates the task calculation results based on the corresponding actions, respectively calculates the system average cost Z and reward r according to Equations (17) and (27), and updates the next state s', which is (s, a, r, s') stored in the experience buffer. Update s to s' and repeat the above steps until the training ends. Step (9) trains the deep neural network by randomly sampling a small batch of experience samples from the experience buffer and updates the network parameters according to Equations (22)–(24).

Algorithm 2: Minimum System Average Cost Algorithm Based on D3QN

1: Initialize the system model environment, initialize the experience buffer D, θ_1 initialize the evaluation network Q_1 with random parameters, θ_2 initialize the target network with random parameters Q_2, $\theta_2 = \theta_1$;
2: Loop training times $Episode = 1, 2, \ldots, M$:
3: reset the system model environment;
4: **for** each user $j = 1, 2, \ldots, J$ **do**:
5: select an action a based on strategy π by state s;
6: the intelligent agent executes action a in state s, enters the next state s' and updates the Q value, calculates the cost of the current task Z according to Equation (17), and obtains reward r according to Equation (27);
7: (s, a, r, s') is stored in experience buffer;
8: update status $s = s'$;
9: randomly sample batch experience samples from the experience buffer, calculate the loss function $\mathcal{L}(\theta)$ according to Equation (22), and update the parameter Q_1 of the evaluation network θ_1 and the parameter Q_2 target network θ_2;
10: Until $j = J$;
11: Until $Episode = M$;
12: Output system average cost.

5. Experimental Simulation and Analysis

In this section, simulation results demonstrate the performance improvement of our proposed strategy in MEC-supported WBAN communication. Significant advantages of the D3QN algorithm can be observed in terms of system average cost under different training parameters, computational offloading conditions, and constraints.

5.1. Simulation Environment

This article uses computer simulation to verify the performance of the proposed algorithm. The simulation software environment is Python 3.9 and the Pytorch 1.9 framework. The simulation hardware platform is a personal computer with an Intel (R) Core (TM) i5-7300HQ 2.50 GHz processor, NVIDIA GeForce GTX 1050 graphics card, and 16 GB of memory.

In the application scenario, users are evenly distributed within a circular area with a radius of 300 m. The base station equipped with MEC servers is located at the center of the circle, and users within the circle can use the network for offloading. The WBAN devices are evenly distributed within a circle with a usable radius of 40 m. According to the parameter settings in reference [31], Table 2 provides the parameter values for the simulation environment.

Table 2. Environment parameters.

Parameter	Symbolic Representation	Set Value
Energy consumption coefficient	γ	0.5
Channel bandwidth (MHz)	W	40
Transmission power of WBAN devices (mW)	p	10
Channel gain between WBAN devices and MEC servers (dB)	H_n	16
Additive Gaussian white noise variance	σ^2	-100
CPU cycle energy consumption of WBAN devices (Megacycles)	ξ	1200
Computing power of WBAN devices (GHz)	f^l	1.5~2
Computing power of edge servers (GHz)	f^e	5~7
The average time required for a blockchain system to generate new blocks (s)	Ts	1
Energy consumption of task upload blockchain (J)	E^{block}	5×10^{-5}
Task Size (MB)	d	1.5~2

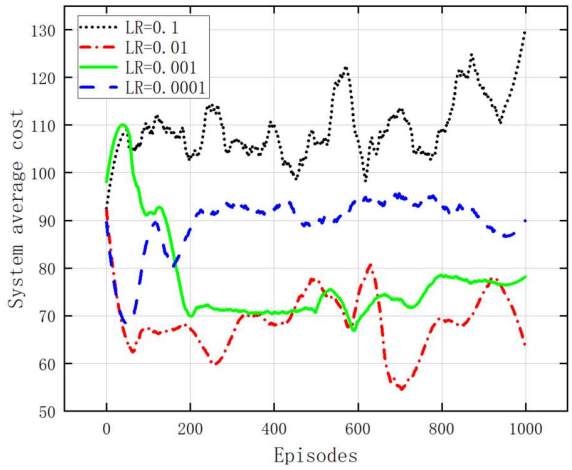
5.2. Discussion of Results

5.2.1. Convergence Performance

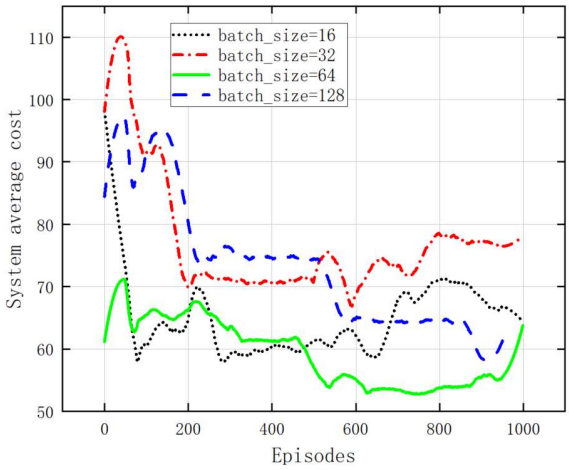
This section demonstrates the convergence performance of our work, where we applied different values to each hyperparameter and selected appropriate values for our simulation. Here, we choose the system average cost for the evaluation metric according to Equation (17).

We conducted an experimental analysis of the different values of learning rate, batch processing, and experience memory, as shown in Figure 6. In the regular case of three WBAN devices and seven edge servers, three sets of data are taken for comparison. Figure 6a shows the system average cost in a D3QN communication network supporting blockchain proposed at different learning rates. The learning rate in DRL refers to the size of the network parameters updated by the gradient of the loss function. In other words, a higher learning rate means a larger range of parameter updates. As shown in Figure 6a, the system average cost maintains a lower and more stable range at a lower learning rate, as it can find the exact position of the optimal value set. It can also be seen that the higher the learning rate, the higher the cost, and the greater the fluctuation. Therefore, in the simulation of this article, we chose a learning rate of 0.01. Figure 6b shows that when the size of batches is small, the system average cost is high and does not converge. When the size of batches is 64, the system average cost is low and converges. Therefore, we chose a batch size of 64. Figure 6c shows the impact of memory size on the system's average cost. When the memory size is 1000, the system average cost is higher and does not converge.

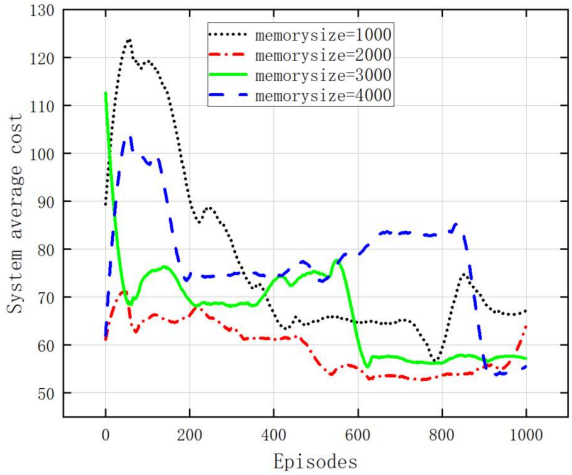
While the memory size is 3000, the system average cost is lower but does not converge. So, we chose the memory size of 2000.



(**a**) Training convergence performance under different learning rates.



(**b**) Training convergence performance under different batch sizes.



(**c**) Training convergence performance under different experience buffer sizes.

Figure 6. Training convergence performance under different parameters.

Therefore, a set of data with a learning rate of 0.01, batch size of 64, and memory size of 2000 will be used for comparison in Table 3.

Table 3. Convergence behavior of the algorithm under various hyperparameter settings.

Hyperparameter	Range	Convergent Cost	Convergent Episodes	Chosen
Learning rate	0.1	112.23	50	0.01
	0.01	68.54	64	
	0.001	75.12	200	
	0.0001	88.47	211	
Batch size	16	66.67	78	64
	32	71.32	201	
	64	55.45	556	
	128	60.69	923	
Memory size	1000	65.43	415	2000
	2000	54.32	514	
	3000	57.59	612	
	4000	55.93	923	

5.2.2. System Performance

In this section, six different task execution methods were applied to evaluate the performance of the system, which are:

- **Local execution:** All computing tasks will be executed and stored locally on the WBAN devices, and thus, local does not care about edge server computing capacity and resources;
- **Edge execution (MEC):** All computing tasks will be offloaded for executing and storing on the edge servers, and thus, MEC does not care about WBAN devices' computing capacity and resources;
- **Random offloading execution:** All computing tasks will be randomly selected to be executed and stored locally or offloaded to edge servers for execution and storage;
- **Dueling DQN offloading model execution:** All computing tasks will be selected by the dueling DQN offloading model to obtain the lowest cost offloading strategy;
- **A3C offloading model execution** [16]: All computing tasks will be selected by the A3C offloading model to obtain the lowest cost offloading strategy;
- **D3QN offloading model execution:** All computing tasks will be selected by the D3QN offloading model to obtain the lowest cost offloading strategy.

Figure 7 compares the relationship between the average system cost and training frequency of six different offloading model systems with a fixed WBAN of three devices and seven edge servers with a cost weight γ of 0.5. The system average cost of local execution remains stable at a high level and consistently higher than the other five offloading models. The system average cost for random offloading remains stable at around 90. The system average cost for "MEC" remains stable at around 80. The average user costs of "A3C offloading execution" and "Dueling DQN offloading execution" fluctuate within the range (60, 120), but the variance of the latter is smaller than the former. For the "D3QN offloading execution", the agent is in the exploration stage before 50 training sessions, and the system average cost for users is relatively high; in 50–600 training sessions, the agent quickly learns the optimal offloading scheme, and the system average cost decreases rapidly as the network parameters are updated; after 600 training sessions, convergence was achieved, and the system average cost was significantly lower than the other five offloading models. Compared to the "Dueling DQN" model, the system average cost has been reduced by 31.25%. Figures 8 and 9 show the relationship between the total latency and energy consumption of six different offloading model systems and the number of training episodes. Figure 8 clearly shows that the D3QN offloading model performs better than the other five offloading models in system average latency. As shown in Figure 9, the D3QN offloading model execution results in a more stable and lower system average energy consumption compared to the other five offloading models.

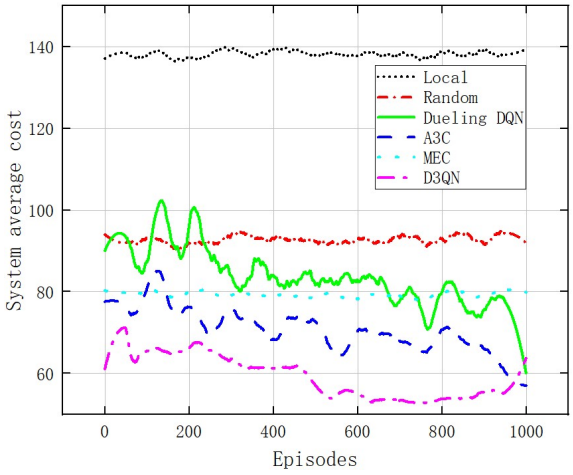


Figure 7. Comparison of average user costs under different algorithms.

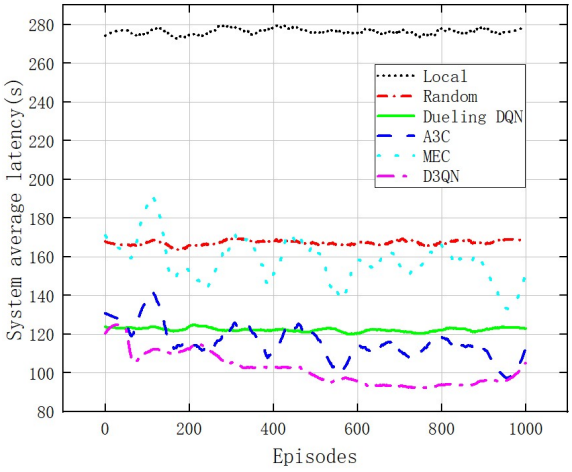


Figure 8. Comparison of average user latency under different algorithms.

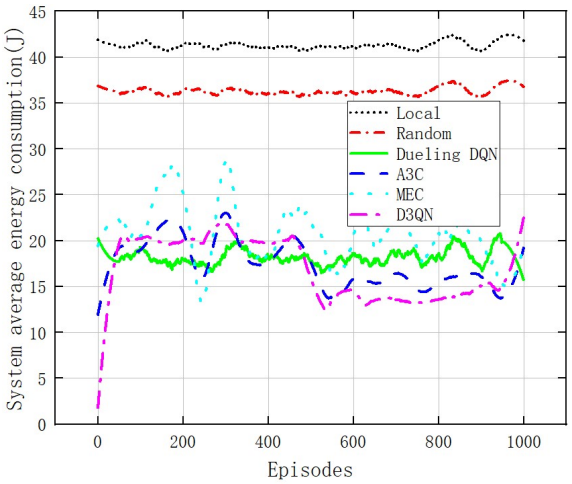


Figure 9. Comparison of average user energy consumption under different algorithms.

Figure 10 shows the comparison of the system average cost for different offloading models under different task sizes with a fixed number of three WBAN devices and seven edge servers with a cost weight γ of 0.5. The values of tasks in the experiments are, respectively, set to ranges 1.5–2 MB, 3.5–4 MB, 5.5–6 MB, 7.5–8 MB, 9.5–10 MB, and 11.5–12 MB. In Figure 10, they are represented by 2 MB, 4 MB, 6 MB, 8 MB, 10 MB, and 12 MB, respectively. The system average cost of the model proposed in this article and the comparison

model increases with the increase of task data size. The system average cost of the "D3QN offloading model execution" is the lowest, while the system average cost of the "local execution" model is the highest. When the number of users is fixed, and the task size is 2 MB, the difference in system average cost between the D3QN offloading model and other models is small. However, as the size of task data increases, the advantage gradually expands. As the task data size is 4 MB, the system average cost performance of the A3C and dueling DQN offloading model is similar to that of the D3QN offloading model. Due to the intelligent agent's ability to adaptively adjust its offloading strategy, the performance advantage of the A3C and dueling DQN offloading model is reflected when the task data size increases to 12 MB. However, it is still worse compared to the D3QN offloading model. The above results further indicate that D3QN is more effective in reducing the system average cost when the task data size is large.

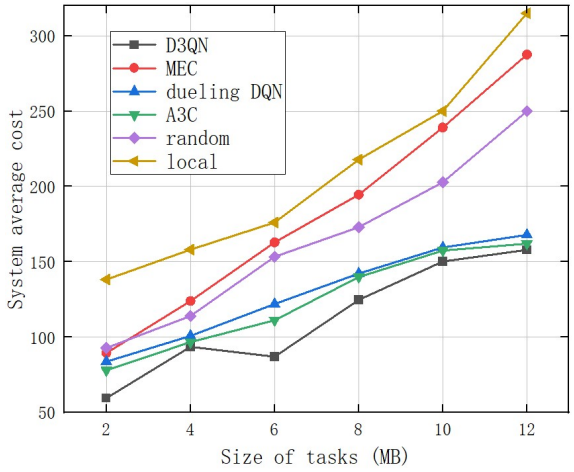


Figure 10. Comparison of system average costs under different task sizes.

Figure 11 compares the blockchain scheme with local and edge schemes to demonstrate the advantages of the blockchain scheme. Based on the results in Figure 11, the blockchain solution has the lowest system average cost. For example, executing a 2 MB file through the blockchain scheme only takes 5.1 s, while in the edge scheme and local scheme, it takes about 6.3 s and 7.5 s, respectively. This enables the use of blockchain solutions to save 19–32% of costs. In addition, compared with local and edge solutions, the proposed blockchain solution, respectively, saves 33% and 19% of costs in computing 12 MB files. We also found that the costs of all three solutions increased with the increase in task size, but the blockchain solution still achieved lower costs than the local and edge solutions, which proves the efficiency of the proposed blockchain solution.

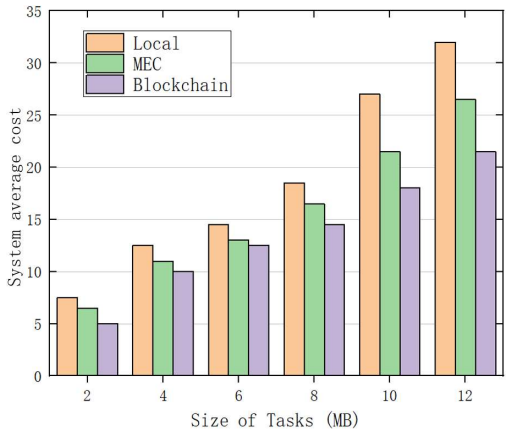


Figure 11. Comparison of the system average costs of local, edge, and blockchain under different task sizes.

The relationship between the system average cost and the number of WBAN devices is shown in Figure 12. From the graph, it can be seen that the proposed model has an overall cost advantage over other models and can achieve near-optimal execution of the model. Specifically, in Figure 12, this gap is rapidly increasing as the number of WBAN devices increases. In addition, our proposed model based on D3QN is closer to the optimal solution than the execution based on DRL. When the number of users exceeds nine, this advantage will be maintained. This is due to the insufficient computing power of edge servers to address WBAN devices simultaneously.

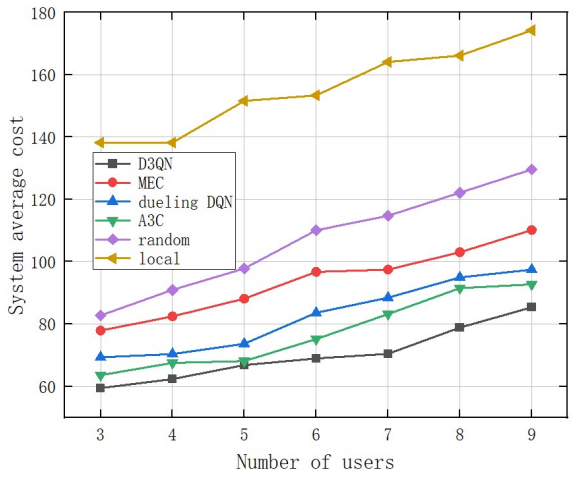


Figure 12. Comparison of system average costs under the number of devices in different WBANs.

Figure 13 reveals a comparison of the average cost of the system for different CPU cycles. We set the number of CPU cycles 10^3 required for the task in the figure to be randomly selected between 10^3 and 2×10^3, and the same applies to other cycles. As the CPU cycles increase, the system average cost of the proposed strategy and comparison strategy significantly increases. Meanwhile, the advantages of the proposed scheme are prominent because D3QN can make optimal decisions based on training, and suitable computing servers can be selected according to different network environments.

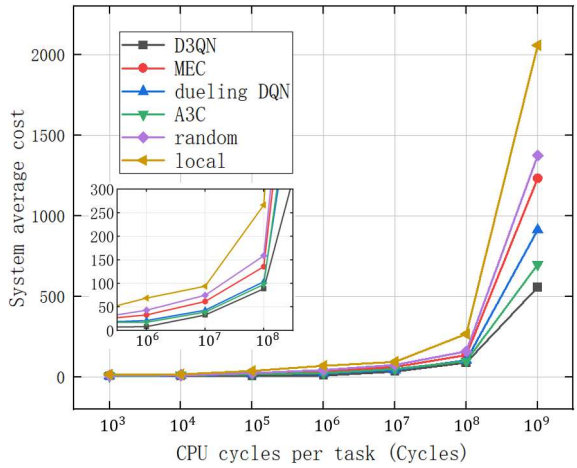


Figure 13. Comparison of the system average cost under the number of CPU cycles required for different tasks.

As shown in Figure 14, the system average cost gradually decreases with the number of edge servers increases, while "local execution" is not affected. It is worth noting that when the number of edge servers exceeds six, the rate of reduction in system average costs slows down significantly. This is because task scheduling is limited by the concurrent number of workflow tasks and the coverage range of wireless signals. The partial order

relationship between tasks results in the concurrent number of workflows being lower than the number of computing resources, leading to some computing resources being idle. Therefore, full consideration should be given to the type of application task and actual environmental conditions when deploying edge servers.

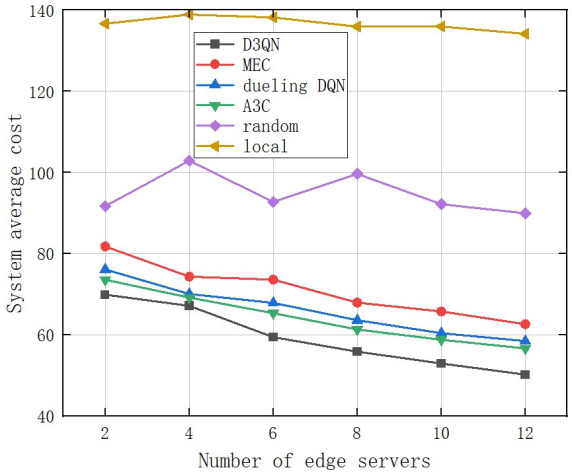


Figure 14. Comparison of system average costs under different edge server numbers.

We can clearly see the variations in the performance metrics of the proposed model in Table 4.

Table 4. Variations with the performance metrics of the proposed model.

Parameter	Range	Average Cost	Variation
Task size (MB)	2, 4, 6, 8, 10, 12	(61.52, 162.23)	Positive
System configuration	local, edge, blockchain	(7.01, 32.12) (6.34, 26.93) (5.12, 22.34)	Negative
Number of devices	3, 4, 5, 6, 7, 8, 9	(59.34, 87.47)	Positive
CPU cycles utilized (Cycles)	$10^3, 10^4, 10^5, 10^6, 10^7, 10^8, 10^9$	(0.01, 588.48)	Positive
Number of edge servers employed	2, 4, 6, 8, 10, 12	(70.49, 51.67)	Negative

5.2.3. QoE Performance

We can map the delay of each user to the utility score according to Equation (18). We determined the minimum tolerance cost for users $Z_{qoe}^{\min} = 60$ and the maximum tolerance cost for users $Z_{qoe}^{\max} = 100$ by measuring the average opinion score. We also defined a tolerance midpoint $Z_{qoe}^{\text{fair}} = 80$ from which users feel a significant decrease in service experience.

In Figure 15, we tested the quality of experience of 1000 users. Among them, 53.7% of users in the D3QN model have excellent QoE; 45.1% of users have good QoE; and 0.12% of users have poor QoE. It can be seen that the user QoE of the D3QN model is significantly higher than the other five models. In addition, the proportion of blocked users obtained by the D3QN model is 0%, while the benchmark is 10%. This is because, in the D3QN model, the user's task is offloaded to the lowest latency computing node. Meanwhile, the "D3QN model" allows agents to offload tasks within T_{n_k} to any computing nodes within the range based on QoE to provide the highest quality.

As an important component of the solution, the impact of blockchain systems on the performance of the solution cannot be ignored. In Figure 16, we discussed the comparison of user QoE with different block size restrictions in the D3QN model. The figure shows that users can gain more and better QoE as the block size increases. However, due to the constraints of the blockchain system, it cannot increase infinitely. Obviously, through the training of the D3QN model, more user data in this scheme have the opportunity to select and upload blockchain systems, ensuring data security.

Figure 17 compares the QoE satisfaction of users with different acceptable delays in the D3QN model. From the graph, it can be seen that as the acceptable delay increases, the proportion of satisfied users significantly increases, indicating a significant increase in the quality of user experience. The reason is that as acceptable latency increases, there is more time to decide and choose the appropriate computing server and blockchain cache. Due to the addition of training through D3QN, the D3QN model can still maintain a lower system average cost under all acceptable delay constraints.

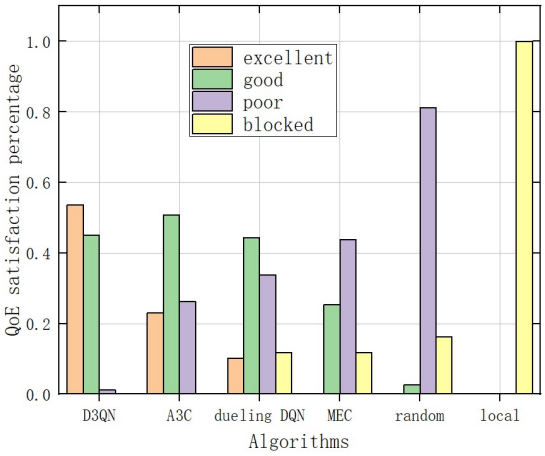


Figure 15. Comparison of user QoE satisfaction under different algorithms.

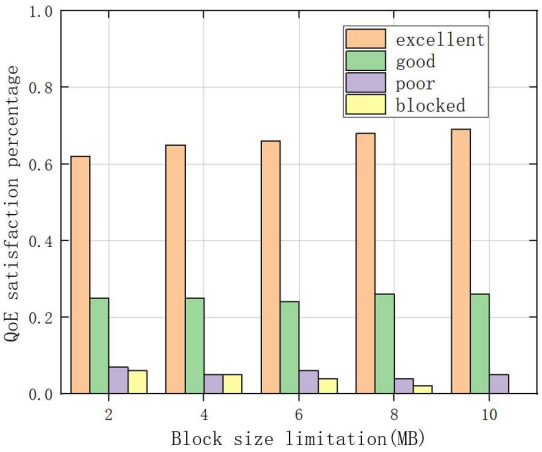


Figure 16. Comparison of user QoE satisfaction under different block size restrictions.

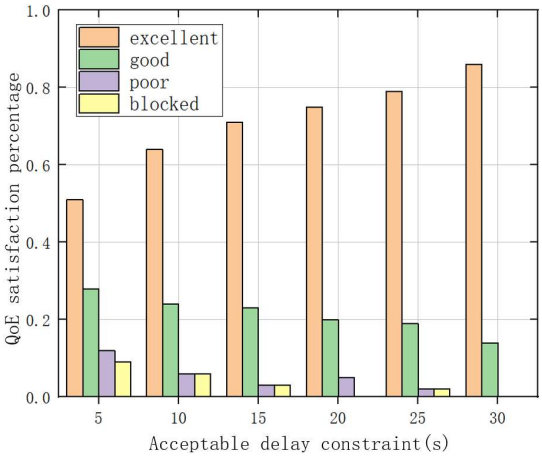


Figure 17. Comparison of user QoE satisfaction under different acceptable delays.

6. Conclusions

This study proposed a new approach to address the QoE of users in task execution of WBANs by jointly considering resource allocation between computing servers and blockchain systems to reduce unnecessary latency and energy consumption to improve system performance. In the proposed model, local WBAN devices or edge servers can be selected to execute complex computing tasks, and blockchain systems can be chosen to ensure the security of users' data. Due to differences in the selection and decision-making of network resources, such as offloading or blockchain, we used the D3QN algorithm to solve joint decision optimization problems. It is possible to make the best decisions about computing offloading and blockchain storage at the lowest system average cost after training, including lower data transmission latency, lower energy consumption, and better data security to improve the QoE of users. The simulation results showed that compared with existing solutions, using the proposed algorithm can significantly reduce users' costs and maintain a high-quality user experience. The next step will be to study the scheduling strategies for task-offloading and storage for WBAN users in dynamic environments.

Author Contributions: Conceptualization, Y.L.; data curation, Y.L.; formal analysis, Y.L. and W.Z.; methodology, Y.L.; validation, Y.L.; writing—original draft, Y.L.; writing—review and editing, Y.L. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the National Natural Science Foundation of China (72071153), the Natural Science Basic Research Program of Shaanxi (S2023-JC-YB-0513), and the Key Research and Development Program of Shaanxi (2021GY-066).

Data Availability Statement: Data are contained within the article.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Teshome, A.K.; Kibret, B.; Lai, D.T.H. A Review of Implant Communication Technology in WBAN: Progress and Challenges. *IEEE Rev. Biomed. Eng.* **2019**, *12*, 88–89. [CrossRef] [PubMed]
2. Hammood, D.; Alkhayyat, A. An overview of the Survey/Review Studies in Wireless WBAN. In Proceedings of the 2020 3rd International Conference on Engineering Technology and Its Applications (IICETA), Najaf, Iraq, 6–7 September 2020; pp. 18–23.
3. Wang, W.; Qin, T.; Wang, Y. Encryption free data transmission and hand over in two tier WBANs. *Comput. Methods Programs Biomed.* **2020**, *192*, 105411. [CrossRef] [PubMed]
4. Rahman, M.A.; Hossain, M.S.; Loukas, G.; Hassanain, E.; Rahman, S.S.; Alhamid, M.F.; Guizani, M. Blockchain based mobile edge computing framework for secure thermal applications. *IEEE Access* **2018**, *6*, 72469–72478. [CrossRef]
5. Verma, P.; Sood, S.K. Fog assisted IoT enabled patient health monitoring in smart homes. *IEEE Internet Things J.* **2018**, *5*, 1789–1796. [CrossRef]
6. Liao, Y.; Qiao, X.; Shou, L.; Zhai, X.; Ai, Q.; Yu, Q.; Liu, Q. Caching Aided Task-offloading Scheme for Wireless WBANs with MEC. In Proceedings of the 2019 NASA/ESA Conference on Adaptive Hardware and Systems (AHS), Colchester, UK, 22–24 July 2019; pp. 49–54.
7. Ren, J.Y.; Zhu, C.H.; Liao, D.S.; Wan, H.B.; Qin, T.F. Fog Computing Assisted Task-offloading Method of Wireless WBANs. *Comput. Eng. Appl.* **2021**, *57*, 108–115.
8. Liao, Y.; Shou, L.; Yu, Q.; Zhai, X.; Ai, Q.; Liu, Q. A Novel Task-offloading Framework to Support Wireless WBANs with MEC. In Proceedings of the 2019 IEEE Smart World, Ubiquitous Intelligence & Computing, Advanced & Trusted Computing, Scalable Computing & Communications, Cloud & Big Data Computing, Internet of People and Smart City Innovation (Smart World/SCALCOM/UIC/ATC/CBDCom/IP/SCI), Leicester, UK, 19–23 August 2019; pp. 1632–1637.
9. Bishoyi, P.K.; Misra, S. Towards Energy And Cost Efficient Sustainable MEC Assisted Healthcare Systems. *IEEE Trans. Sustain. Comput.* **2022**, *7*, 958–969. [CrossRef]
10. Zhang, R.; Zhou, C. A Calculation Task-offloading Scheme based on Mobile Cloud and Edge Computing for WBANs. In Proceedings of the ICC 2022 IEEE International Conference on Communications, Seoul, Republic of Korea, 16–20 May 2022.
11. Deng, H.; Meng, X.; Guo, J.; Xi, E.; Zhao, H. A Framework of Blockchain Based Security for WBANs. In Proceedings of the 2020 3rd International Conference on Smart Blockchain (Smartblock), Zhengzhou, China, 23–25 October 2020; pp. 75–80.
12. Xiao, L.; Han, D.; Meng, X.; Liang, W.; Li, K.-C. A Secure Framework for Data Sharing in Private Blockchain Based WBANs. *IEEE Access* **2020**, *8*, 153956–153968. [CrossRef]
13. Wang, Y.; Cheng, L.; Xu, J.; Zhang, S. A Terminal Device Authentication Scheme Based on Blockchain Technology in WBAN. In Proceedings of the 2023 IEEE 10th International Conference on Cyber Security and Cloud Computing (CSCloud)/2023 IEEE 9th International Conference on Edge Computing and Scalable Cloud (Edgecom), Xiangtan, China, 1–3 July 2023; pp. 411–416.

14. Xu, J.; Meng, X.; Liang, W.; Zhou, H.; Li, K.-C. A secure mutual authentication scheme of blockchain-based in WBANs. *China Commun.* **2020**, *17*, 34–49. [CrossRef]
15. Li, Y. Deep reinforcement learning: An overview. *arXiv* **2017**, arXiv:1701.07274. [CrossRef]
16. Yuan, X.; Zhu, Y.; Zhao, Z.; Zheng, Y.; Pan, J.; Liu, D. An A3C based Joint Optimization Offloading and Migration Algorithm for SD WBANs. In Proceedings of the 2020 IEEE Globecom Workshops (GC Wkshps), Taipei, Taiwan, 7–11 December 2020; pp. 1–6.
17. Zhang, R.; Li, H.; Qiao, Y.; Li, M.; Xia, X. Deep Learning based Task-offloading and Time Allocation for Edge Computing WBANs. In Proceedings of the GLOBECOM 2022—2022 IEEE Global Communications Conference, Rio de Janeiro, Brazil, 4–8 December 2022; pp. 2206–2211.
18. Yuan, X.; Zhang, Z.; Feng, C.; Cui, Y.; Garg, S.; Kaddoum, G.; Yu, K. A DQN Based Frame Aggregation and Task-offloading Approach for Edge enabled IoMT. *IEEE Trans. Netw. Sci. Eng.* **2023**, *10*, 1339–1351. [CrossRef]
19. Yuan, X.; Tian, H.; Wang, H.; Su, H.; Liu, J.; Taherkordi, A. Edge enabled WBANs for Efficient QoS Provision Healthcare Monitoring: A Two Stage Potential Game Based Computing Offloading Strategy. *IEEE Access* **2020**, *8*, 92718–92730. [CrossRef]
20. Liao, Y.; Han, Y.; Yu, Q.; Ai, Q.; Liu, Q.; Leeson, M.S. Wireless WBAN Mobility Aware Task-offloading Scheme. *IEEE Access* **2018**, *6*, 61366–61376. [CrossRef]
21. Ning, Z.; Dong, P.; Wang, X.; Hu, X.; Guo, L.; Hu, B.; Guo, Y.; Qiu, T.; Kwok, R.Y.K. Mobile Edge Computing enabled 5G Health Monitoring for Internet of Medical Things: A Decentralized Game Theoretical Approach. *IEEE J. Sel. Areas Commun.* **2021**, *39*, 463–478. [CrossRef]
22. Zhang, L.; Yuan, X.; Luo, J. An Adaptive Resource Allocation Approach Based on User Demand Forecasting for E-Healthcare Systems. In Proceedings of the 2022 IEEE International Conference on Communications Workshops (ICC Workshops), Seoul, Republic of Korea, 16–20 May 2022; pp. 349–354.
23. Liao, Y.; Yu, Q.; Zhao, X.; Ai, Q. Wireless Big Data Meets WBANs: An Attempt for Collaborative Task Process Assisted with MEC. In Proceedings of the 2019 IEEE International Conference on Communications Workshops (ICC Workshops), Shanghai, China, 22–24 May 2019; pp. 1–5.
24. Ren, J.; Li, J.; Liu, H.; Qin, T. Task-offloading strategy with emergency handling and blockchain security in SDN empowered and fog assisted healthcare IoT. *Tsinghua Sci. Technol.* **2022**, *27*, 760–776. [CrossRef]
25. Ren, J.; Qin, T. Decentralized Blockchain Based and Trust Aware Task-offloading Strategy for Healthcare IoT. *IEEE Internet Things J.* **2023**, *11*, 829–847. [CrossRef]
26. Baucas, M.J.; Spachos, P.; Gregori, S. Private Blockchain Based Wireless WBAN Platform for Wearable Internet of Thing Devices in Healthcare. In Proceedings of the ICC 2023-IEEE International Conference on Communications, Rome, Italy, 28 May–1 June 2023; pp. 6181–6186.
27. Son, S.; Lee, J.; Kim, M.; Yu, S.; Das, A.K.; Park, Y. Design of Secure Authentication Protocol for Cloud Associated Telecommunications Medical Information System Using Blockchain. *IEEE Access* **2020**, *8*, 192177–192191. [CrossRef]
28. Liu, M.; Yu, F.R.; Teng, Y.; Leung, V.C.M.; Song, M. Performance optimization for blockchain-enabled industrial Internet of Things (IIoT) systems: A deep reinforcement learning approach. *IEEE Trans. Ind. Inform.* **2019**, *15*, 3559–3570. [CrossRef]
29. Pham, X.-Q.; Nguyen, T.-D.; Nguyen, V.; Huh, E. Joint servant caching and task-offloading in multi access edge computing: A qoe based utility optimization approach. *IEEE Commun. Lett.* **2021**, *25*, 965–969. [CrossRef]
30. Qin, B.X.; Zhang, Y.X.; Wu, S.R.; Cao, W.; Li, Z. Intelligent Optimization of Coal Terminal Offloading Process Scheduling Based on Improved D3QN Algorithm. *J. Syst. Simul.* **2024**, *36*, 770–781.
31. Chen, N.; Zhang, S.; Qian, Z.; Wu, J.; Lu, S. When Learning Joins Edge: Real Time Proportional Computing Offloading via Deep Reinforcement Learning. In Proceedings of the 2019 IEEE 25th International Conference on Parallel and Distributed Systems (ICPADS), Tianjin, China, 4–6 December 2019; pp. 414–421.