remote sensing

MDPI

Article

A Node Selection Strategy in Space-Air-Ground Information Networks: A Double Deep Q-Network Based on the Federated Learning Training Method

Weidong Wang [1], Siqi Li [1], Jihao Zhang [1], Dan Shan [1,2], Guangwei Zhang [1] and Xiang Gao [1,*]

[1] School of Mechatronical Engineering, Beijing Institute of Technology, Beijing 100081, China; 3220185030@bit.edu.cn (W.W.); 3220215037@bit.edu.cn (S.L.); 3120215161@bit.edu.cn (J.Z.); 3120185162@bit.edu.cn (D.S.); 7520220037@bit.edu.cn (G.Z.)

[2] School of Electrical and Control Engineering, Shenyang Jianzhu University, Shenyang 110168, China

* Correspondence: bitxianggao@bit.edu.cn

Abstract: The Space-Air-Ground Information Network (SAGIN) provides extensive coverage, enabling global connectivity across a diverse array of sensors, devices, and objects. These devices generate large amounts of data that require advanced analytics and decision making using artificial intelligence techniques. However, traditional deep learning approaches encounter drawbacks, primarily, the requirement to transmit substantial volumes of raw data to central servers, which raises concerns about user privacy breaches during transmission. Federated learning (FL) has emerged as a viable solution to these challenges, addressing both data volume and privacy issues effectively. Nonetheless, the deployment of FL faces its own set of obstacles, notably the excessive delay and energy consumption caused by the vast number of devices and fluctuating channel conditions. In this paper, by considering the heterogeneity of devices and the instability of the network state, the delay and energy consumption models of each round of federated training are established. Subsequently, we introduce a strategic node selection approach aimed at minimizing training costs. Building upon this, we propose an innovative, empirically driven Double Deep Q Network (DDQN)-based algorithm called low-cost node selection in federated learning (LCNSFL). The LCNSFL algorithm can assist edge servers in selecting the optimal set of devices to participate in federated training before the start of each round, based on the collected system state information. This paper culminates with a simulation-based comparison, showcasing the superior performance of LCNSFL against existing algorithms, thus underscoring its efficacy in practical applications.

Keywords: Space-Air-Ground Information Network; federated learning; delay and energy model; node selection strategy; low-cost node selection in federated learning

Citation: Wang, W.; Li, S.; Zhang, J.; Shan, D.; Zhang, G.; Gao, X. A Node Selection Strategy in Space-Air-Ground Information Networks: A Double Deep Q-Network Based on the Federated Learning Training Method. *Remote Sens.* **2024**, *16*, 651. https://doi.org/10.3390/rs16040651

Academic Editor: Claudio Piciarelli

Received: 20 December 2023
Revised: 28 January 2024
Accepted: 2 February 2024
Published: 9 February 2024

1. Introduction

The rapid advancement of Internet of Things (IoT) technology has catalyzed an unprecedented surge in data generation. Recent projections indicate that the number of active IoT devices may approach 30 billion by 2030 [1]. Despite extensive mobile network deployment, connectivity remains elusive in many remote areas, such as deserts or oceans. This connectivity gap is underscored by research indicating that terrestrial wireless networks cover merely 20% of the global land area and less than 6% of the Earth's surface [2]. This limited reach is attributed to challenging terrain, extended communication distances, and various commercial and engineering complexities. In light of these limitations, there is a consensus among academic and industrial experts on the necessity of Satellite Communication (SatCom) networks as an effective supplement to terrestrial infrastructures. SatCom is deemed crucial for achieving comprehensive global coverage and facilitating seamless connectivity, particularly for critical computing applications like artificial intelligence (AI) [3].

Over the last five decades, the cellular network landscape has undergone a rapid transformation, evolving from the First Generation to the cutting-edge Fifth Generation Mobile Network. Currently, the technological frontier is shifting towards the exploration of the Sixth Generation (6G) of communication technology, which is poised to play a pivotal role in the Smart Information Society envisioned for 2030 [4]. Anticipated to outperform its predecessor, 6G is expected to cater to a plethora of emerging services and applications, marking a significant leap in communication technology [5,6]. The Sixth Generation is expected to revolutionize network coverage and user mobility by optimizing infrastructures in the air, in space, on the ground, and at sea. It aims to seamlessly integrate terrestrial and non-terrestrial networks, thus offering comprehensive and unrestricted coverage [7,8].

Concurrently, the surge in data produced by ubiquitous IoT devices has necessitated the application of AI techniques, such as deep learning, for the development of sophisticated data models. These models are integral to smart IoT applications in healthcare, transportation, and urban management [9]. Traditionally, AI processing has been centralized in cloud servers or data centers [10], a practice that faces significant challenges in the era of IoT data proliferation. Centralized data collection and transmission are often impractical or inefficient due to communication resource constraints and latency issues. Moreover, the sensitivity of the massive data generated raises significant security concerns, as highlighted by global regulatory frameworks like the General Data Protection Regulation in Europe [11].

Federated learning (FL), a pioneering distributed machine learning framework, was introduced by Google in 2016. As a technical paradigm, FL represents a distributed, collaborative form of AI. It enables multiple devices to coordinate with a central server for data training purposes, without necessitating the sharing of the actual dataset [12]. FL is distinguished by several key advantages, including improved communication efficiency and enhanced data privacy. These benefits stem from its ability to transmit only model parameters instead of raw data [13].

In the realm of next-generation wireless communications, particularly those encompassing the Space-Air-Ground Information Network (SAGIN) augmented by AI, FL has garnered significant research interest. This interest is propelled by the expanding integration and prevalence of data-driven applications. However, the financial viability of SatCom, when compared with that of terrestrial mobile networks, remains a concern. To facilitate intelligent adaptive learning for extensive IoT networks and to mitigate the costs associated with high-volume SatCom traffic, recent studies have explored the application of FL within low-Earth-orbit SatCom networks [14].

Previous research has delved into enhancing wireless network efficiency, focusing on Mobile Edge Computing (MEC) and federated learning (FL). In MEC, studies have addressed energy-efficient task offloading challenges in Layer 2 Service Chaining services across multi-Radio Access Technology networks, considering constraints like stringent latency and residual battery energy [15,16]. For FL in wireless networks, a novel model has been proposed to optimize base station resource allocation and user transmission power allocation, jointly considering learning and wireless network metrics. This aims to reduce packet error rates and enhance overall FL performance [17].

To overcome challenges in the Metaverse's channel state and computing resources, a soft actor–critic-based solution has been developed for an efficient FL scheme with dynamic user selection, gradient quantization, and resource allocation [18]. Recognizing the limitations of asynchronous federated learning and semi-asynchronous federated learning methods, a new approach named FedSEA has been introduced as a semi-asynchronous FL framework tailored for extremely heterogeneous devices [19]. Additionally, in vehicular networking scenarios, studies have applied fuzzy logic for client selection, considering parameters such as the number and freshness of local samples, computational capability, and available network throughput [20].

Nevertheless, most of the existing works rely on an unrealistic assumption that edge devices participating in federated training have stable network connections [21,22]. In

practice, the network quality may change due to the movement of devices or the change in environmental factors. Unpredictable network quality increases the difficulty of designing FL algorithms. Meanwhile, mobile industrial devices using battery power [23] in this scenario usually have limited computational and communication resources. This also puts a higher demand on the time and energy consumption of joint training. Some existing solutions employ a Deep Deterministic Policy Gradient (DDPG) to address latency and energy consumption issues when selecting FL participants among IoT device nodes [24,25].

In this paper, we model the delay and energy consumption for each round of federated training. Under the condition of independent and identically distributed data distribution, the Low-Cost Node Selection in Federated Learning (LCNSFL) algorithm, based on the Double Deep Q Network (DDQN), is proposed to minimize the time and energy consumption of each round of federation training. This algorithm aims to minimize the time and energy consumption associated with each training round. LCNSFL assists the edge server in selecting the most efficient device set for participation in federated training, leveraging the system state information collected. Through comparative experimental analysis, we demonstrate that LCNSFL significantly reduces both time and energy consumption in federated training while ensuring high convergence accuracy of the global model.

The rest of the paper is organized as follows: Section 2 models the latency and energy consumption of the federated training process. In Section 3, we design the node selection algorithm for minimizing training consumption in DDQN-based FL. Section 4 evaluates the algorithm, and Section 5 concludes the paper. All acronyms used throughout this survey paper are given in Table 1.

Table 1. List of abbreviations.

Abbreviation	Definition
6G	Sixth Generation
AI	Artificial intelligence
Bnq	Best network quality device selection strategy
DDPG	Deep Deterministic Policy Gradient
DDQN	Double Deep Q Network
DRL	Distributed Reinforcement Learning
ES	Edge server
FL	Federated learning
IoT	Internet of Things
LCNSFL	Low-Cost Node Selection in Federated Learning
MDP	Markov Decision Process
MEC	Mobile Edge Computing
RLI	Reinforcement Learning Intelligence
SAGIN	Space-Air-Ground Information Network
SatCom	Satellite Communication
UAV	Unmanned Aerial Vehicle

2. System Model and Problem Definition

In the paper, we investigate the time and energy consumption of FL with various IoT devices and time-varying channel state information. First, the time and energy consumption are defined. Then, the nodes are selected to minimize the consumption in each federated training step. Finally, a DDQN-based algorithm is proposed to solve the formulated problem.

As depicted in Figure 1, the architecture of the Distributed Reinforcement Learning (DRL)-enhanced FL system is presented for implementation in the SAGIN. The SAGIN encompasses satellites, Unmanned Aerial Vehicles (UAVs), and ground base stations. The IoT devices refer to cameras and robotic arms, etc., which establish connections to the edge cloud server. To achieve the federated averaging scheme, these devices first transmit the local model parameters to the edge cloud server, and then the edge cloud server aggregates those parameters. These aggregated algorithms are employed to update the

global federation model. It should be noted that this process in a server can be augmented with a Reinforcement Learning Intelligence (RLI) component. The RLI aids the FL system by strategically selecting a subset of devices to partake in the federation training.

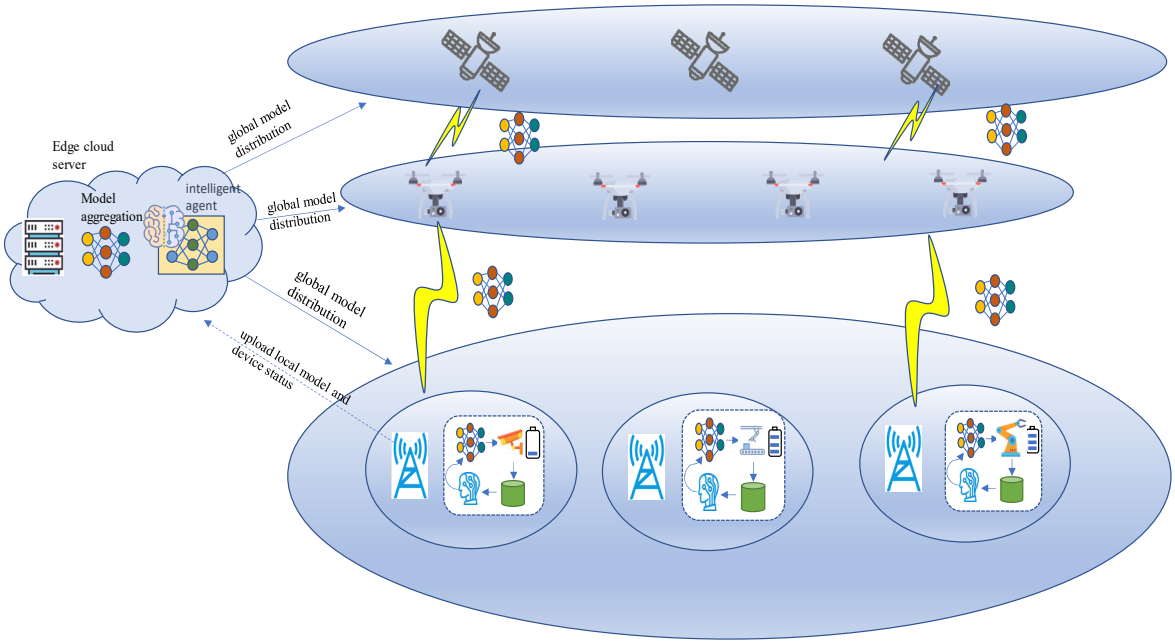

Figure 1. Simple diagram of FL implementation in IoT scenarios through the SAGIN.

Prior to the initiation of each round of federated training, the edge cloud server carefully selects a subset of IoT devices to actively participate in the training process, delivering to them the latest iteration of the global model. The designated IoT devices leverage locally available data for model training and subsequently transmit the refined model parameters back to the edge cloud server. Following this, the edge cloud server undertakes the essential task of parameter aggregation, systematically iterating through the aforementioned steps until the accuracy of the global model reaches the predefined target.

The time and energy requirements for each round of federated training can be categorized into three components: computational time for local model training, duration of data transmission, and idle waiting periods for devices. Similarly, IoT device energy consumption during training includes computational energy, energy required for parameter transmission, and energy consumed during idle waiting.

Due to the superior communication resources available to the edge cloud server, whose downstream bandwidth significantly exceeds that of the upstream bandwidth of industrial IoT devices, the time and energy requirements for devices to download the global model in each training round may be reasonably overlooked.

We can integrate energy consumption sensors on each IoT device node to monitor real-time power consumption. With the data collected from these sensors, we can quantify the energy consumption of each node under various tasks. However, it is not feasible for the server to obtain the local training time and energy consumption of all devices in advance before the start of a federated training round. Therefore, designing a node selection strategy to avoid high-energy-consuming devices and those with poor channel quality from participating in the federated training process will help reduce the time and energy consumption of the federated training process. This is crucial for resource-constrained IoT devices.

This paper focuses on FL algorithm design for node selection without considering the optimization of the DDQN network architecture. Key symbols in this study are listed in Table 2.

Table 2. Main symbolic representations.

Variable	Definition
c_i	The number of cycles required for device i to train a single sample
d_i	The number of samples that device i has
f_i	CPU frequency of device i
τ	Number of local iterations
N_0	Noise power
k	Communication round number
$B_{i,k}$	The bandwidth allocated to device i in the k-th communication round.
$p_{i,k}$	The transmission power of device i
$g_{i,k}$	The channel gain of device i
b_i^k	The data transmission rate of device i in the k-th communication round
w_i^k	The local model of device i
σ	The effective capacitance coefficient related to the computing chipset

The FL system consists of an edge server (ES) and N IoT devices. The devices are denoted by an index i: $i \in S$, $S = \{1, 2, \ldots, N\}$. The dataset of device i is represented as D_i, and its size is denoted as di. (x_i, y_i) represents a data sample from D_i, where x_i denotes the sample data and y_i represents the sample label. During local model training, x_i is used as the input to the model, and y_i is used as the expected output. The cross-entropy between y_i and the model's predicted output y_i' is calculated as the local model's loss function.

Figure 2 illustrates the scheduling process of federated synchronous learning. Prior to the commencement of each FL round, the ES selects n devices to participate in the training process. The set of selected devices is represented as S_{sub}, where $S_{sub} = \{1, 2, \ldots, n\}$. The ES distributes the global model to each selected device, which then performs τ iterations to update the model using its local dataset D_i. After completing local model training, each device uploads its local model's weight parameters to the ES. The ES performs the model aggregation algorithm to obtain the latest global model and then proceeds to the next round of federated training.

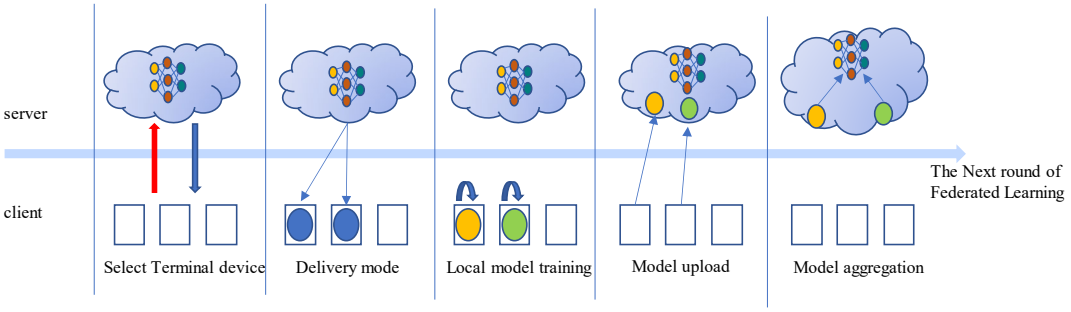

Figure 2. Illustration of federated synchronous learning scheduling process.

Assuming that device i requires c_i CPU cycles to train a single sample and operates at a frequency of f_i, the computation time required for device i to execute local model training in the k-th round of federated training is given by

$$t_{i,k}^{cal} = \frac{\tau \cdot c_i \cdot d_i}{f_i} \tag{1}$$

After completing the local model updates, device i uploads the trained model's weight parameters w_i^k to the ES. The transmission time of the model is calculated using the following formula:

$$t_{i,k}^{com} = \frac{\left| w_i^k \right|}{b_i^k} \tag{2}$$

where $|w_i^k|$ represents the size of the model weights, b_i^k represents the data transmission rate of device i in the k-th round of federated training, and b_i^k is calculated using the following equation:

$$b_i^k = B_{i,k} log_2(1 + \frac{p_i^k \cdot \mathrm{A}}{N_0 \cdot (d_i^k)^\eta}). \tag{3}$$

The transmission signal of the UAV system in the SAGIN is affected by various fading effects in the ground-to-space channel, including large-scale path fading, shadow fading, and small-scale fading. To address this, a generalized CE2R channel model is introduced. In this model, A is a constant and is the path loss index. It is a general form of the path loss model, and parameter adjustments are required on a case-by-case basis [26].

According to the above equation, the data transmission rate of device i is influenced by the allocated bandwidth to device i (b_i^k), the transmission power of device i (p_i^k), the channel gain (g_i^k), and the noise power (N_0). Changes in environmental conditions can affect the data transmission rate and introduce uncertainty in transmission delays. The training time of device i in the k-th round of federated training is the sum of the computation time for local model updates and the model transmission time. Therefore, the training time of device i can be obtained using the following equation:

$$t_{i,k}^{total} = t_{i,k}^{cal} + t_{i,k}^{com}. \tag{4}$$

In the federated synchronous model, the next round of federated training begins only when the ES receives the local model parameters from all devices and creates a new global model. Therefore, the learning time of the k-th round of federated training is determined by the slowest device, and it can be represented by the following formula:

$$T_k = \underset{i \in S_{sub}}{max} \left\{ t_{i,k}^{total} \right\}. \tag{5}$$

According to the energy model proposed in reference [27], the energy consumed by device i during local model training in the k-th round of federated training can be calculated using the following equation:

$$E_i^{k,c} = \sigma \cdot \tau \cdot c_i \cdot d_i \cdot f_i^2 \tag{6}$$

where σ represents the effective capacitance coefficient, which is related to the properties of the chip itself.

During model transmission, device i performs the model transmission with a power of p_i^k. Therefore, the energy consumption of device i for transmission is calculated as

$$E_i^{k,t} = t_{i,k}^{com} p_i^k. \tag{7}$$

The device that completes local model training and model transmission first needs to wait for other devices that have not finished. Fast devices may experience idle waiting time, and the energy consumed during this idle waiting period is referred to as idle energy consumption. Therefore, the idle energy consumption of device i is equal to the product of idle waiting time and the energy consumption per unit time in the idle state. The calculation formula is as follows:

$$E_i^{k,idle} = (T_k - t_{i,k}^{total}) \cdot E_i^{ps} \tag{8}$$

where E_i^{ps} represents the energy consumption per unit time of device i during the idle waiting period. Therefore, the total energy consumed by device i during the k-th round of federated training can be calculated as follows:

$$E_i^k = E_i^{k,c} + E_i^{k,t} + E_{i,}^{k,idle}. \tag{9}$$

Therefore, the total energy consumed during the k-th round of federated training can be calculated as follows:

$$E^K = \sum_{i=1}^{n} E_i^k. \tag{10}$$

Equations (11)–(17) describe the device selection problem under the independent and identically distributed data distribution. It can be formulated as selecting a group of devices in each round of federated training to minimize the training time consumption and energy consumption cost. To reduce the idle waiting time of fast devices and minimize unnecessary idle energy consumption, the selected devices must have similar training time costs. This problem can be formulated as the minimization of T_k. The formulation of the optimization problem is as follows:

$$min(T_k + \lambda \cdot \sum_{i=1}^{N} a_i^k \cdot E_i^k) \tag{11}$$

$$f_{\min} \leq f_i \leq f_{\max} \tag{12}$$

$$0 \leq \sum_{i=1}^{N} a_i^k \cdot b_i^k \leq B \tag{13}$$

$$1 \leq \sum_{i=1}^{N} a_i^k \leq N \tag{14}$$

$$E_i^k < E_{max} \tag{15}$$

$$1 \leq i \leq N \tag{16}$$

$$1 \leq k \leq K. \tag{17}$$

Due to different requirements of different FL tasks for the latency and energy consumption in each training round, a trade-off between time cost and energy cost can be achieved by using the hyperparameter λ. In Equation (11), λ represents the preference for the optimization objective. If λ takes a relatively large value, it indicates that the optimization model focuses more on reducing energy consumption during training. If λ takes a relatively small value, the optimization model pays more attention to reducing training time cost. a_i^k represents the decision variable, where $a_i^k = 1$ indicates that device i is selected to participate in federated training in the k-th communication round, and $a_i^k = 0$ indicates that device i does not participate in the federated training process. Equation (12) represents the constraint on device operating frequency. Devices with excessively high or low frequencies will not be selected, ensuring that the computational latency and computational energy consumption are not too large. Equation (13) represents the constraint that the total bandwidth of the selected devices should not exceed the server's bandwidth. Equation (14) represents the constraint that at least one device should be selected to participate in federated training in a communication round, and the maximum number of selected devices does not exceed the total number of devices. Equation (15) is the constraint on the energy consumption of device i, which should not exceed a specified upper limit.

3. Design of Node Selection Algorithm for Minimum Training Cost in FL Based on DDQN

Due to the nonlinearity of the constraints and the unpredictable changes in the network states of each device, solving the optimization problem described in Equation (11) is extremely challenging. In order to find the optimal set of devices for each round of federated training and achieve a dynamic trade-off between time cost and energy cost, the DDQN algorithm is considered for solving the node selection problem. This node selection scheme is named LCNSFL.

To solve the node selection problem in FL using the DRL approach, it is first necessary to abstract the problem as a Markov Decision Process (MDP). An MDP consists of a system state $S(t)$, an action space $A(t)$, a policy π, a reward function r, and neighboring states $S(t+1)$. The detailed parameter description is as follows.

(a) System state: This chapter considers a practical scenario with dynamic network bandwidth. However, it is assumed that the network state remains relatively stable within a short time slot and does not undergo drastic changes within a few tens of seconds. The state space $S(t)$ for DRL is defined as the combination of the device's data transmission rate $\beta(t)$, operating frequency $\zeta(t)$, signal transmission power $T_p(t)$, and the number of samples owned by the device $I(t)$. Thus, at time slot t, the system state can be represented by the following equation:

$$S(t) = \{\beta(t), \zeta(t), I(t), T_p(t)\}. \tag{18}$$

It is worth noting that due to the heterogeneity of devices and the instability of network conditions, the data transmission rate and operating frequency of each device may vary at different time slots t. Therefore, before each round of federated training, it is necessary to sample the data transmission rate, operating frequency, and other information of each device multiple times and take the average as the current state variables. On the other hand, the number of samples owned by each device and the signal transmission power are relatively stable, so the sampled values at time slot t can be used as the system's state variables.

(b) The action space, denoted as $A(t)$, is a vector consisting of discrete variables (0 or 1). $a_i^t \in A(t)$ represents the selection status of device i at time slot t. $a_i^t = 1$ indicates that device i is selected to participate in federated training at time slot t, while $a_i^t = 0$ indicates that device i does not participate in federated training at time slot t.

(c) The policy π represents the mapping from the state space $S(t)$ to the action space $A(t)$, i.e., $A(t) = \pi(S(t))$. The goal of DRL is to learn an optimal policy π that maximizes the expected reward based on the current state.

(d) The reward function r is aligned with the optimization objective, which is to minimize the weighted sum of time cost and energy cost. Therefore, the reward function r can be expressed as follows:

$$\begin{aligned} r &= -\left(T_k + \lambda \sum_{i=1}^{N} a_i^k E_i^k\right) \\ &= -\left(\lambda \sum_{i=1}^{N} a_i^k \left(E_i^{k,c} + E_i^{k,t} + E_i^{k,idle}\right) + \max_{i \in S_{sub}} \left\{t_{i,k}^{cal} + t_{i,k}^{com}\right\}\right) \\ &= -\left(\lambda \sum_{i=1}^{N} a_i^k \left(\sigma \cdot \tau \cdot c_i \cdot d_i \cdot f_i^2 + \frac{w_i^k}{b_i^k} p_{i,k} + \left(t_{i,k}^{cal} + t_{i,k}^{com} - T_k\right) E_i^{ps}\right)\right. \\ &\left. + \max_{i \in S_{sub}} \left\{\frac{\tau \cdot c_i \cdot d_i}{f_i} + \frac{w_i^k}{b_i^k}\right\}\right). \end{aligned} \tag{19}$$

(e) The adjacent state $S(t+1)$ is determined based on the current state $S(t)$ and the policy π. The specific expression is as follows:

$$S(t+1) = S(t) + \pi(S(t)). \tag{20}$$

It is important to note that the reward defined in Equation (19) is an immediate reward, i.e., the instantaneous reward feedback, denoted as r_t, received from the environment after the agent takes action $a(t)$ based on policy π at time t. The goal of DRL is to maximize the sum of immediate rewards and discounted future rewards. Based on the Bellman equation and temporal difference algorithm, to obtain the optimal policy using the value function approach in DRL, it is necessary to estimate the action-value function for future time steps and discount the action-value function for future time steps using a discount factor, γ. The temporal difference target is defined as $r_t + \gamma \cdot Q(s_{t+1}, a_{t+1})$, where $Q(s_t, a_t)$ is the estimated

action-value function at time t. Since r_t is the true reward obtained at time t, it is considered more reliable than the estimate $Q(s_t, a_t)$. Therefore, the goal is to approximate the action-value function estimated at time t to $r_t + \gamma \cdot Q(s_{t+1}, a_{t+1})$. The temporal difference error, $r_t + \gamma \cdot Q(s_{t+1}, a_{t+1}) - Q(s_t, a_t)$, is optimized using loss functions such as mean square error to improve the accuracy of the neural network's estimation of the state value.

The FL node selection problem is addressed by employing the DDQN reinforcement learning model on edge cloud servers, achieving a dynamic trade-off between energy consumption and training time. The DDQN algorithm, which is capable of processing continuous states and generating discrete actions through neural networks, is particularly well suited for addressing the node selection challenge in FL within IoT scenarios. In comparison with alternative algorithms such as DDPG, Trust Region Policy Optimization, and others, the DDQN strikes a favorable balance concerning algorithmic simplicity, sample complexity, and parameter tuning flexibility and mitigates the issue of Q-value overestimation observed in the standard Deep Q Network algorithm.

As shown in Figure 3, edge cloud servers are usually deployed at edge locations close to production or sensing devices, such as factories or production lines, and are connected to IoT devices through base stations. Such a deployment can optimize the efficiency of data processing and decision making, reduce data transmission delays, and improve the real time and responsiveness of the system.

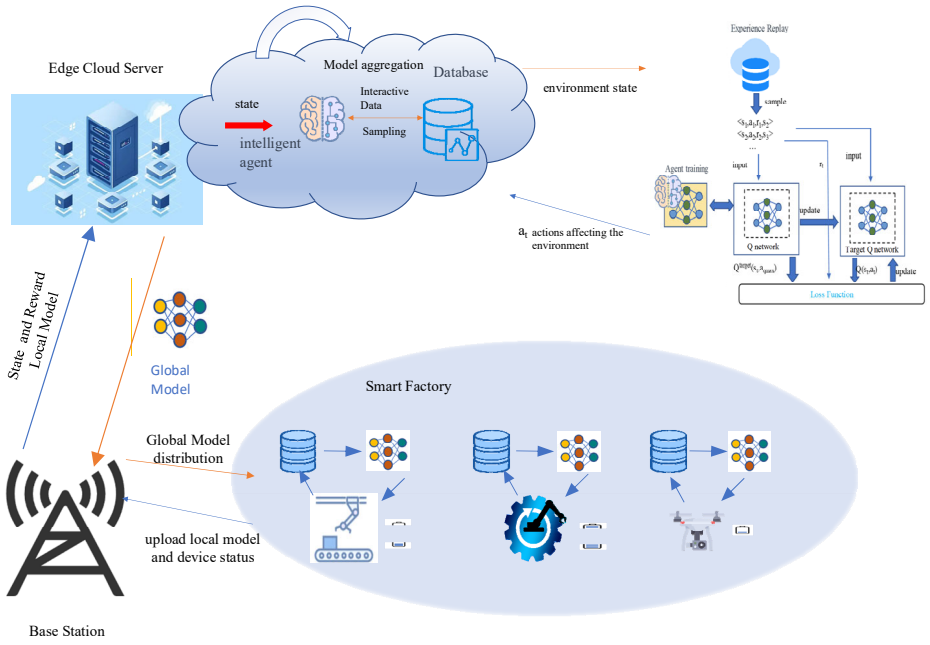

Figure 3. Edge cloud server deployment and overview of the DDQN algorithm.

The DDQN algorithm consists of a Q network and a target Q network. For convenience's sake, the target Q network is denoted as "target Q". In order to compress the action space, during the training phase, the index corresponding to the maximum Q value output by the Q network at time t is selected as the device number for local model updates. This approach reduces the action space from 2^N to N, where N is the total number of IoT devices. The Q network outputs the action a_t based on the current state s_t, which interacts with the FL environment. Subsequently, the state of the FL environment transitions from s_t to s_{t+1}, and a scalar is returned to the agent as an immediate reward, r_t. A boolean variable "*done*" is defined as the termination flag for federated training. When the communication rounds of federated training reach the upper limit, "*done*" is set to true, and the training process is terminated. The tuple $< s_t, a_t, r_t, s_{t+1}, done >$ is stored in the experience replay buffer as a record of the interaction between the agent and the environment. When the interaction data in the experience replay buffer reach a certain quantity, the Q network trains on the data from the buffer, updating the node selection strategy.

The Q network is updated using a temporal difference algorithm. The state s_t and action a_t are inputted into the Q network, yielding the action value at time t, denoted as $Q(s_t, a_t)$. Then, the next state s_{t+1} is inputted into the Q network to obtain the q values for different actions, and the action corresponding to the maximum q value, denoted as a_{maxq}, is selected. Next, s_{t+1} is inputted into the target Q network to find the q value corresponding to the action a_{maxq}, denoted as $Q^{target}(s_{t+1}, a_{maxq})$. Finally, $Q(s_t, a_t)$ is used as the predicted value of the network, and $r_t + \gamma Q^{target}(s_{t+1}, a_{maxq})$ is used as the actual value of the network. Mean square error is used as the loss function to perform backpropagation on $r_t + \gamma Q^{target}(s_{t+1}, a_{maxq}) - Q(s_t, a_t)$. To ensure the stability of the training process, in practice, only the parameters of the Q network are updated, while the weight parameters of the target Q network are fixed. The detailed training flow of the LCNSFL node selection algorithm is as follows (Algoritm 1):

Algorithm 1. The training process of the FL node selection algorithm based on LCNSFL.

Input: Q network, target Q network Q_{tar}, target network update frequency f_{tar}, greedy factor e, greedy factor decay factor β, minimum sample size of the experience pool *mBatch*, maximum communication rounds of FL T, total number of devices N.
Output: The trained Q network Q and the trained target Q network Q_{tar}.
1: Initialize the local models of the devices, initialize the Q network as Q, initialize the target Q network as Q_{tar}, and set the step counter as $step = 0$.
2: **for** $t = 1$ to T **do**
3: Collect the state *s(t)*
4: *done = False*
5: Generate a random number *rd*
6: **if** *rd > epsilon* **then**
7: *a(t) = random(0,N)*
8: **else**
9: $a(t) = \underset{a(t) \in A}{\operatorname{argmax}} Q(s(t), a(t))$
10: **end if**
11: The edge server selects a device based on the action *a(t)*, performs local model training on the selected device, and updates the global model.
12: Compute the instantaneous reward *r(t)* based on Formula (19) and update the state from *s(t)* to *s(t+1)*.
13: **if** $t = T$ **then**
14: *done = False*
15: **end if**
16: Store the tuple information <*s(t), a(t), r(t), s(t+1), done*> in the experience pool.
17: **if** *the number of samples in the experience pool > mBatch* **then**
18: Randomly sample mBatch number of samples from the experience pool
19: Use the Q network to estimate the q value *Q(s(t), a(t))* at time *t*.
20: Use the Q network to estimate the q value *Q(s(t+1), a(t+1))* at time *t+1* and obtain the action a_{maxq} corresponding to the maximum q value.
21: Use the target Q network to estimate the action value $Q_{tar}(s(t+1), a_{maxq})$ at time $t + 1$.
22: Optimize the Q network based on the stochastic gradient descent (SGD) method using Formula (23).
23: Update the greedy factor $e = e \cdot e^{-\beta}$.
24: **if** $step \% f_{tar} = 0$ **then**
25: Update the parameters of the target Q network (Q_{tar}) using the parameters of the Q network.
26: **end if**
27: *step = step + 1*
28: **end if**
29: **end for**
30: **return** the trained Q network Q and the trained target Q network Q_{tar}.

The DDQN algorithm separates action selection and value estimation. Specifically, it selects the action corresponding to the maximum q value at state s_{t+1} from the Q network, denoted as a^*. The value of action a^* at state s_{t+1} is estimated using the target Q network. The calculation of the TD target in the DDQN is expressed as follows:

$$Y^{target} = r + \gamma Q^{target}\left(s_{t+1}, \underset{a' \in A}{\arg max}\, Q(s_{t+1}, a'; \theta); \theta^{-}\right). \tag{21}$$

During the training process, the parameters θ^{-} of the target Q network are frozen, meaning that the target Q network is not updated. The parameters of the Q network, denoted as θ, are copied to θ^{-} every f_{req} iteration. The loss function of the Q network is defined as follows:

$$\iota(\theta) = \left(Y^{target} - Q(s, a; \theta)\right)^2. \tag{22}$$

The update process of minimizing the loss function $\iota(\theta)$ through gradient descent is as follows for θ:

$$\theta_{t+1} = \theta_t + \alpha\left(Y^{target} - Q(s_t, a_t; \theta_t)\right)\nabla_{\theta_t} Q(s_t, a_t; \theta_t) \tag{23}$$

The symbol α represents the step size or learning rate for the update.

4. Performance Evaluation

The training process of FL assisted by the LCNSFL node selection algorithm during the testing phase is as shown in Figure 4.

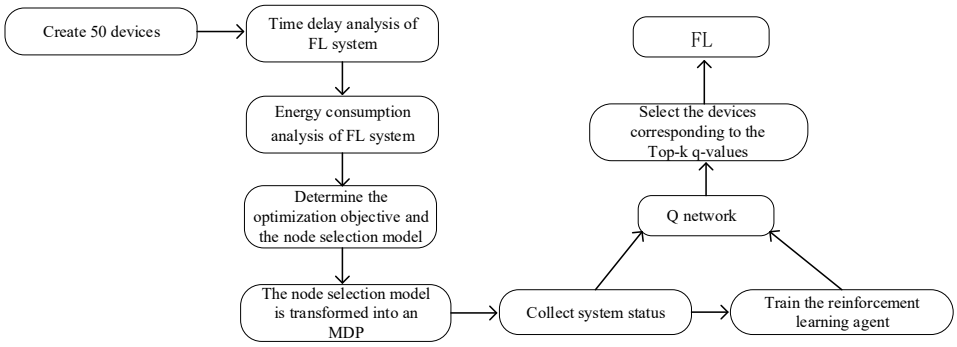

Figure 4. Training process of FL assisted by LCNSFL algorithm.

In the testing phase, the node selection strategy differs from that in the training phase. During the training phase, only the devices with the highest q values are selected to participate in the FL training process. However, in the testing and application phases, the top n devices are chosen based on the descending order of their q values to participate in local training and global model updates.

4.1. Experimental Environment Configuration

4.1.1. Dataset and Data Allocation Method

In order to show the application of the classification task, the MNIST dataset is selected as the local dataset source, and a two-layer MLP model is used as the training model for FL. The samples of each category were randomly sampled from the training set, and 50 samples of each category are randomly sampled. A total of 500 samples are selected to form a local dataset, and the constructed dataset is assigned to each IoT device.

4.1.2. Simulation Parameter Settings

To reflect the differentiated computing and communication capabilities of IoT devices, different working frequencies and channel bandwidths are assigned to them. In the simulation of the communication scenario of a smart factory in the suburbs of the city, the path loss index η is generally between 2 and 4, and the constant A is generally between

50 and 150. The device attribute information in this section's experiment is referenced from [28], and the specific settings can be found in Table 3.

Table 3. Simulation parameter setting.

Parameter Type	Parametric Description	Setting
Local model and model parameters	Number of local iterations	3
	Local dataset size	500
	Number of fully connected layers	3
	Activation function	Relu
	Learning rate	0.01
	Optimizer	SGD
DDQN parameters	Training steps	500
	Reward discount factor	0.9
	Q Network learning rate	0.001
	Greedy factor	1
	Greedy factor decay factor	0.01
	Minimum sample size (mBatch)	64
	Q Network linear layer count	2
	Q Network hidden layer dimension	128
	Target Q Network update frequency (freq)	20
	Weighting factor (λ)	0.6
Device attributes	Number of devices	50
	Path Loss Index (η)	3
	A	100
	Model size	10 MB
	Node bandwidth	6~8 Mbps
	Slow node bandwidth	0.1~0.3 Mbps
	Server bandwidth	100 Mbps
	Transmission power	0.2~0.4 w
	Working frequency	500 MHz~900 MHz
	Energy consumption per bit for training	0.02~0.04 J
	Cycle required for training a unit of bit data	6000~7000

For the convenience of experimental comparison, 10 devices are selected to participate in the federated training process in each round.

4.1.3. Comparative Algorithms

Random selection strategy: in each round of federated training, random selection is used to choose n devices to participate in federated training, and the federated averaging algorithm is used to aggregate the global model.

The best network quality device selection strategy (Bnq): Wang et al., pointed out that communication latency, which is influenced by factors such as network uncertainty and bandwidth limitations, has been proven to be a bottleneck affecting the performance of FL [29]. Therefore, selecting devices with good channel quality to participate in federated training can reduce time costs and improve the training efficiency of FL. Based on this, in each round of federated training, the n devices with the best channel quality, i.e., the highest data transmission rate, are selected from all devices to participate in federated training. The model aggregation is performed using the federated averaging algorithm.

4.2. DDQN Model Training and Effect Analysis

Figure 5 shows the plot of the change in the reward value obtained by the LCNSFL algorithm in the training phase with the number of iteration rounds. It can be seen from the figure that in the initial stage of training, the DRL agent learns a poor device selection policy due to the lack of interaction information with the environment and thus obtains a small reward value. With the increase in iteration rounds, the DRL agent constantly updates its node selection strategy, making the optimization objective expressed in Equation (11)

smaller and smaller, so that the reward value obtained by the DRL agent becomes higher and higher, and converges after 300 iterations. A higher reward value obtained by the DRL agent indicates a better device selection policy learned by the DRL agent. Here, MA is the Moving Average (MA), and the orange dotted line is the result of smoothing the return curve.

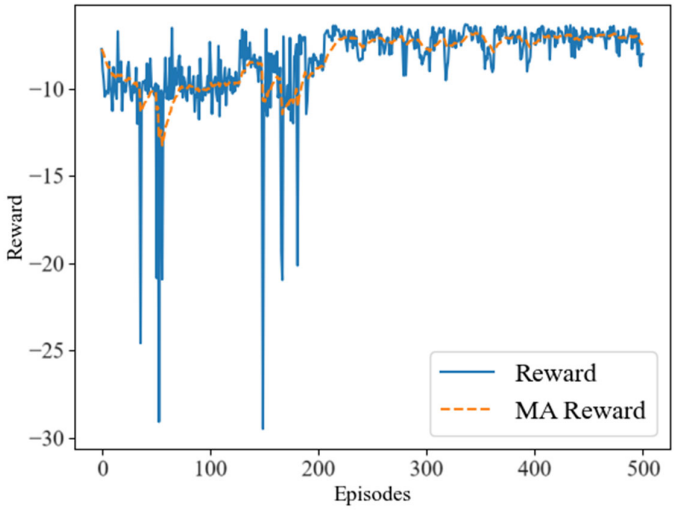

Figure 5. Training return curve of the LCNSFL algorithm.

Figure 6 shows the change plot of the loss function of LCNSFL in the training phase with the number of training rounds, where the blue curve represents the loss function of each training round and the orange dotted line is the result after smoothing the loss curve. Corresponding to Figure 5, in the early stage of training, due to the lack of interaction information between the DRL agent and the FL environment, the loss function represented by Equations (23) is large. With the increase in the number of interactions between the Q network and the environment, the Q network and target Q network update their own parameters by using the interaction data and Equation (24) so that the training loss decreases rapidly. After 200 iterations, the training loss reaches a low level and tends to be stable. This indicates that the node selection scheme based on the DDQN proposed in this chapter can converge, and the DRL agent can better complete the device selection task in the FL environment.

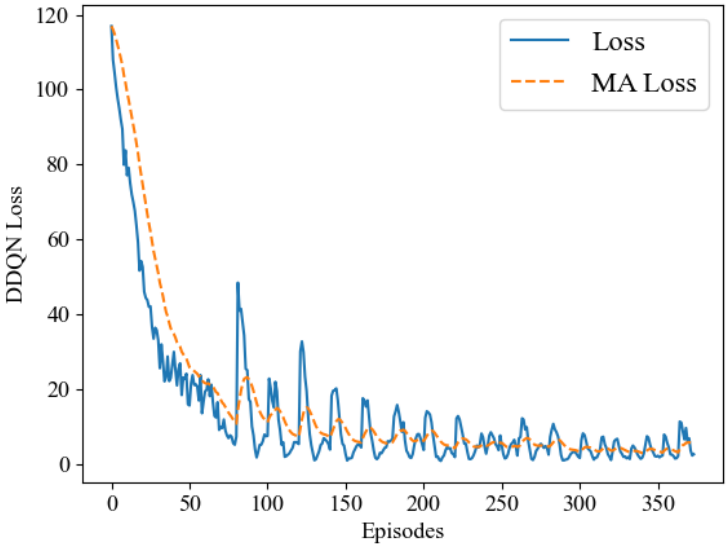

Figure 6. Loss function change curve during the training phase.

The variation curves of training time cost and energy cost for the LCNSFL algorithm-assisted training phase are displayed in Figures 7 and 8. The horizontal axis represents the communication rounds of FL, and the vertical axis represents the training cost of FL. In the early exploration stage, the intelligent agent adopts a more random strategy for device selection to obtain richer interaction data. As a result, the training time cost and energy cost of FL show significant fluctuations. As the number of training iterations increases, the DRL agent learns more optimal device selection strategies, resulting in a decrease in the optimization objective represented by Equation (11). Therefore, in the later stages of training, the time cost and energy cost of each round of federated training tend to stabilize and decrease.

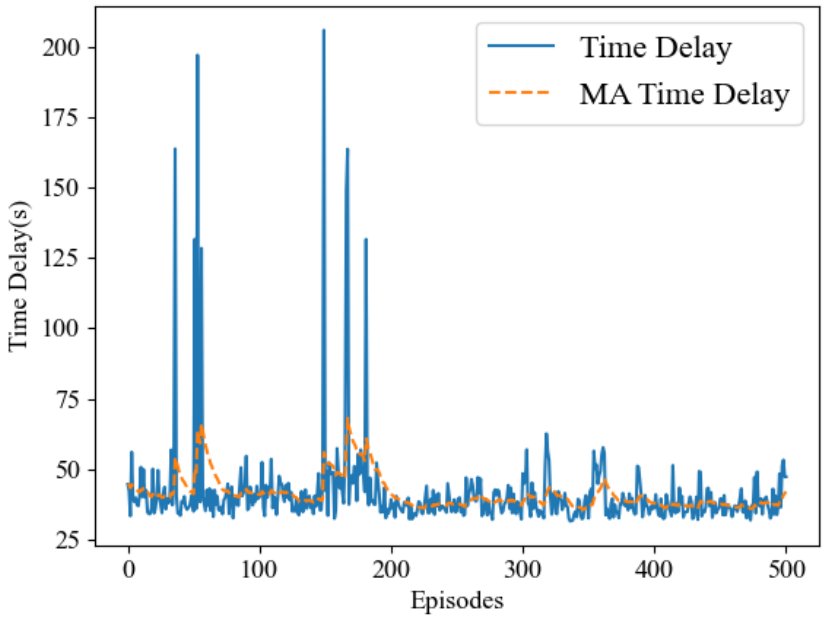

Figure 7. The curve depicting the variation in time cost during the training phase.

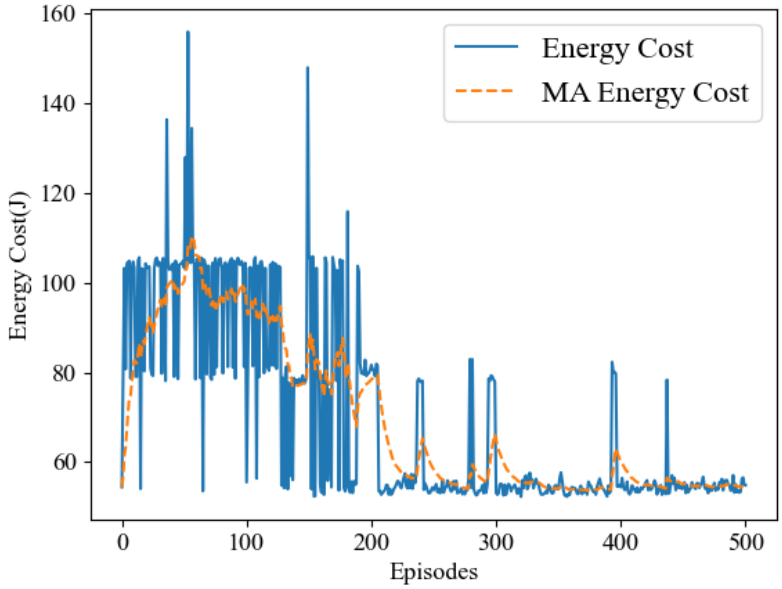

Figure 8. The curve illustrating the variation in energy cost during the training phase.

4.3. Comparison of Training Costs for Different Equipment Selection Strategies

To validate the performance of the LCNSFL node selection algorithm, this experimental section will consider the time cost, energy cost, weighted training cost, and global model

accuracy required for each round of federated training as measures of the node selection algorithm's performance. To demonstrate the algorithm's robustness, five devices with poor channel quality will be randomly selected.

To demonstrate the effectiveness of the LCNSFL algorithm, in Figure 9, we firstly present the time cost of FL in each round of federated training under the assistance of three node selection strategies, LCNSFL, random selection, and Bnq selection, in a dynamic network scenario. From the graph, it can be observed that FL assisted by the random selection algorithm tends to select devices with weaker computing capabilities or poorer channel quality. Bnq selection, guided by the algorithm, selects the best n (n = 10) devices in terms of channel quality for each round of training, thus being less affected by changes in channel conditions. However, devices with good channel quality do not necessarily possess strong computing capabilities. The Bnq selection algorithm may select devices with good channel quality but weak computing capabilities, resulting in higher time costs due to more time being consumed for local model training. The LCNSFL algorithm considers the computing resources and channel bandwidth of devices comprehensively, achieving minimization of both computation and transmission time. It avoids selecting devices with poor channel quality or weak computing capabilities. Therefore, FL assisted by the LCNSFL algorithm exhibits lower time costs compared with the above two algorithms. The time cost fluctuations between different training rounds are smaller, allowing it to better adapt to the FL environment and exhibit robustness.

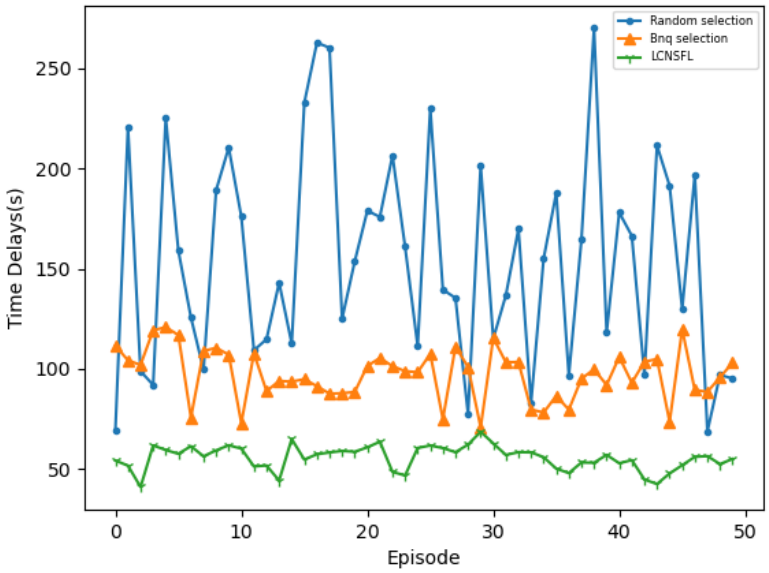

Figure 9. Comparison of time cost.

The energy cost of FL in each round of federated training under the assistance of three node selection strategies is compared in Figure 10. It is evident from the graph that FL assisted by LCNSFL has lower energy costs. The fluctuations in energy costs between different training rounds are also minimized. This is attributed to the reinforcement learning agent's ability to intelligently select the optimal subset of devices based on the collected system states, achieving a dynamic trade-off between energy cost and time cost.

Finally, in Figure 11, we compare the weighted cost of federation learning assisted by three node selection strategies in each round of federation training, where the weighted cost of one round of federation training consists of 50% time cost and 50% energy cost.

The LCNSFL algorithm takes both time and energy costs into account when performing node selection and minimizes the weighted sum of time and energy costs for each round of federation training in the DRL approach. It is obvious that the weighted cost consumed by the LCNSFL algorithm in each communication round is significantly smaller than that consumed in the other two node selection strategies, indicating that the LCNSFL

algorithm can optimize the training cost of FL in dynamic network scenarios by node selection in IoT scenarios.

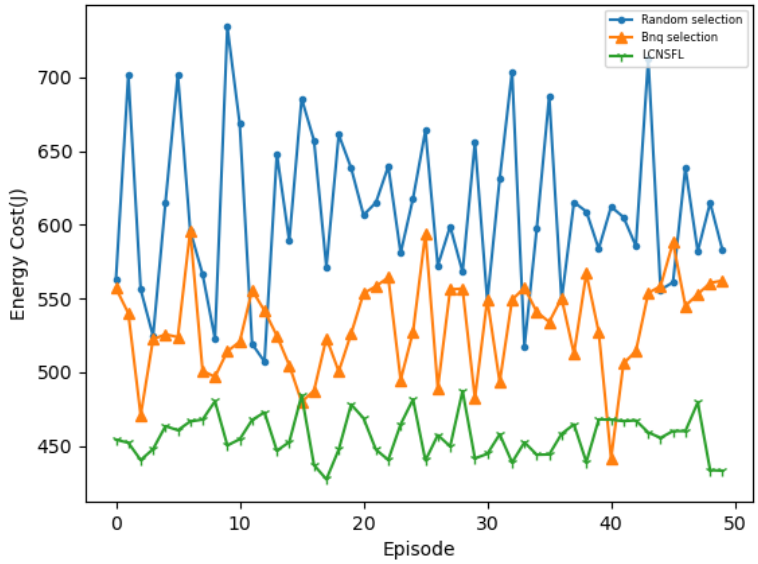

Figure 10. Comparison of energy cost.

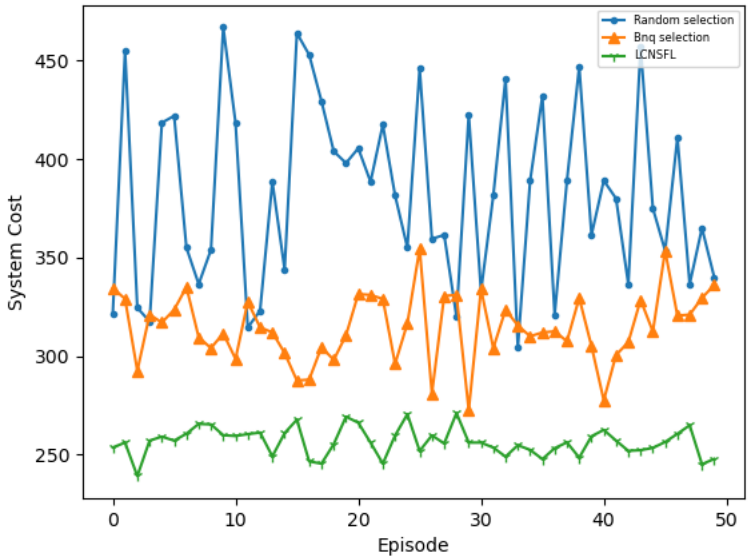

Figure 11. Comparison of weighted training.

Table 4 provides a statistical summary of the time cost, energy cost, and weighted cost incurred during the first 50 rounds of federated training under the assistance of the three node selection strategies.

Table 4. Cost comparison for the first 50 rounds of federated training.

Algorithm Name	Time Cost (s)	Energy Cost (J)	Weighted Cost
LCNSFL	2821.1	22,816.5	12,818.0
Random Selection	7769.0	30,449.0	19,109.1
Bnq Selection	4879.0	26,587.8	15,733.4

In terms of time cost, the LCNSFL algorithm reduces this by 63.7% compared with the random selection strategy and 42.2% compared with the Bnq selection strategy. In terms of energy cost, the LCNSFL algorithm reduces this by 25.1% compared with the

random selection strategy and 14.2% compared with the Bnq selection strategy. In terms of weighted cost, the LCNSFL algorithm reduces this 32.9% compared with the random selection strategy and 18.5% compared with the Bnq selection strategy. This proves that the overall performance of the LCNSFL algorithm is better than that of the other two traditional algorithms.

In Figure 12, the accuracy performances of the global models trained under the assistance of the three node selection algorithms on the test dataset are given. From the graph, it can be observed that all three node selection strategies converge quickly and achieve relatively high accuracy for the global model. The randomness in device selection allows the federated model to learn more diverse data features, which contributes to the convergence and improvement in the global model's accuracy. Due to the inherent randomness of environmental changes, both the LCNSFL and Bnq selection algorithms exhibit a certain level of randomness in device selection based on the sampled information. As a result, all three algorithms assist in achieving high convergence accuracy for the global federated model.

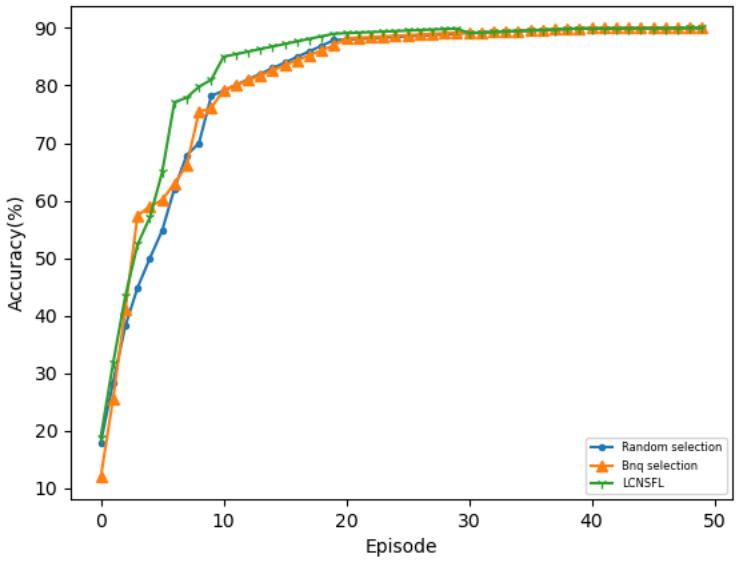

Figure 12. Comparison of accuracy.

From the experimental results, it can be observed that the LCNSFL algorithm, without compromising the accuracy of the global model, effectively reduces the time cost and energy cost of each round of federated training in dynamic network scenarios through precise node selection. This algorithm exhibits good robustness and resilience.

5. Conclusions

The application of FL in SAGIN facilitates collaborative modeling among devices while ensuring the protection of endpoint device data privacy, providing a data-driven solution. However, challenges such as heterogeneous computational resources, fluctuating data transmission rates due to environmental factors, and limited energy resources significantly impact device selection during the federated training process.

In this study, we established models for time delay and energy consumption in the federated training process. Subsequently, we introduced the LCNSFL algorithm based on DDQN to minimize the time and energy costs in each round of federated training. The LCNSFL algorithm adaptively selects the optimal device subset by considering the collected device status information, achieving a dynamic trade-off between time cost and energy cost in federated training.

In our simulation experiments, we observed that the LCNSFL algorithm gradually increases rewards during the training phase and tends to converge after 400 rounds of iterations. The training loss also decreases gradually, reaching a relatively low level and

stabilizing after 400 iterations, indicating effective convergence. Once the reward values converge, the DRL agent performs well in device selection tasks in the federated learning environment, and the time and energy costs of each round of federated training also tend to stabilize.

To further confirm the superiority of the LCNSFL algorithm, we compared it with traditional node selection strategies, namely, the random selection and Bnq selection approaches. The results indicate that the random selection algorithm tends to select devices with weaker computing capabilities, poorer channel quality, or higher energy consumption in federated learning. While Bnq selection performs better than random selection in these aspects, the LCNSFL algorithm outperforms both strategies overall. Therefore, the LCNSFL algorithm, without compromising the global model's accuracy, effectively reduces the time and energy costs per round of federated training in dynamic network scenarios through precise node selection, demonstrating robustness.

Overall, the LCNSFL algorithm offers a promising solution for addressing challenges in federated learning within the SAGIN framework, showcasing its effectiveness and robustness in dynamic network environments.

Author Contributions: Conceptualization, W.W. and G.Z.; methodology, W.W. and X.G.; software, S.L. and J.Z.; validation, G.Z., S.L., and J.Z.; formal analysis, W.W. and D.S.; investigation, W.W. and X.G.; resources, W.W.; data curation, D.S. and G.Z.; writing—original draft preparation, W.W. and X.G.; writing—review and editing, S.L.; visualization, S.L. and J.Z.; supervision, D.S. and X.G.; project administration, X.G. and G.Z.; funding acquisition, X.G. and D.S. 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 Youth Fund (Grant No. 62203048) and the National Natural Science Foundation of China (Grant No. 62073039 and No. 62003225).

Data Availability Statement: Data are contained within the article.

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

References

1. Internet of Things (IoT)—Statistics & Facts. Available online: https://www.statista.com/topics/2637/internet-of-things/#topicOverview (accessed on 28 April 2023).
2. Chen, S.Z.; Liang, Y.C.; Sun, S.H.; Kang, S.L.; Cheng, W.C.; Peng, M.G. Vision, requirements, and technology trend of 6 g: How to tackle the challenges of system coverage, capacity, user data rate, and movement speed. *IEEE Wirel. Commun.* **2020**, *27*, 218–228. [CrossRef]
3. Kodheli, O.; Lagunas, E.; Maturo, N.; Sharma, S.K.; Shankar, B.; Montoya, J.F.M.; Duncan, J.C.M.; Spano, D.; Chatzinotas, S.; Kisseleff, S.; et al. Satellite Communications in the New Space Era: A Survey and Future Challenges. *IEEE Commun. Surv. Tut.* **2021**, *23*, 70–109. [CrossRef]
4. Jiang, W.; Han, B.; Habibi, M.A.; Schotten, H.D. The road towards 6G: A comprehensive survey. *IEEE Open J. Commun. Soc.* **2021**, *2*, 334–366. [CrossRef]
5. Zhang, Z.Q.; Xiao, Y.; Ma, Z.; Xiao, M.; Fan, P.Z. 6G wireless networks vision, requirements, architecture, and key technologies. *IEEE Veh. Technol. Mag.* **2019**, *14*, 28–41. [CrossRef]
6. Li, H.F.; Chen, C.; Shan, H.G.; Li, P.; Chang, Y.C.; Song, H.B. Deep Deterministic Policy Gradient-Based Algorithm for Computation Offloading in IoV. *IEEE Trans. Intell. Transp. Syst.* **2023**, 1–12. [CrossRef]
7. Nayak, S.; Patgiri, R. 6G: Envisioning the Key Issues and Challenges. *arXiv* **2020**, arXiv:2004.04024.
8. Chen, C.; Wang, C.Y.; Li, C.; Xiao, M.; Pei, Q.Q. A V2V Emergent Message Dissemination Scheme for 6G-Oriented Vehicular Networks. *Chin. J. Electron.* **2023**, *32*, 1179–1191. [CrossRef]
9. Mohammadi, M.; Al-Fuqaha, A. Enabling cognitive smart cities using big data and machine learning: Approaches and challenges. *IEEE Commun. Mag.* **2018**, *56*, 94–101. [CrossRef]
10. Sun, Y.H.; Peng, M.G.; Zhou, Y.C.; Huang, Y.Z.; Mao, S.W. Application of machine learning in wireless networks: Key techniques and open issues. *IEEE Commun. Surv. Tutor.* **2019**, *21*, 3072–3108. [CrossRef]
11. Voigt, P.; Bussche, A.V.D. *The EU General Data Protection Regulation (GDPR): A Practical Guide*, 1st ed.; Springer: Berlin, Germany, 2017; pp. 315–320.
12. Konen, J.; Mcmahan, H.B.; Yu, F.X.; Richtárik, P.; Bacon, D. Federated learning: Strategies for improving communication efficiency. *arXiv* **2017**, arXiv:1610.05492.

13. Nguyen, D.C.; Ding, M.; Pham, Q.V.; Pathirana, P.N.; Le, L.B.; Seneviratne, A.; Li, J.; Niyato, D.; Poor, H.V. Federated Learning Meets Blockchain in Edge Computing: Opportunities and Challenges. *IEEE Internet Things J.* **2021**, *8*, 12806–12825. [CrossRef]
14. Chen, H.; Xiao, M.; Pang, Z. Satellite-based computing networks with federated learning. *IEEE Wirel. Commun.* **2022**, *29*, 78–84. [CrossRef]
15. Qin, M.; Cheng, N.; Jing, Z.; Yang, T.; Xu, W.; Yang, Q.; Rao, R.R. Service-oriented energy-latency tradeoff for IoT task partial offloading in MEC-enhanced multi-RAT networks. *IEEE Internet Things J.* **2020**, *8*, 1896–1907. [CrossRef]
16. Jing, Z.; Yang, Q.; Qin, M.; Kwak, K.S. Momentum-Based Online Cost Minimization for Task Offloading in NOMA-Aided MEC Networks. In Proceedings of the 2020 IEEE 92nd Vehicular Technology Conference (VTC2020-Fall), Victoria, BC, Canada, 18 November–16 December 2020; pp. 1–6. [CrossRef]
17. Chen, M.; Yang, Z.; Saad, W.; Yin, C.; Poor, H.V.; Cui, S. A Joint Learning and Communications Framework for Federated Learning Over Wireless Networks. *IEEE Trans. Wirel. Commun.* **2021**, *20*, 269–283. [CrossRef]
18. Hou, X.; Wang, J.; Jiang, C.; Meng, Z.; Chen, J.; Ren, Y. Efficient Federated Learning for Metaverse via Dynamic User Selection, Gradient Quantization and Resource Allocation. *IEEE J. Sel. Areas Commun.* **2023**. [CrossRef]
19. Sun, J.; Li, A.; Duan, L.; Alam, S.; Deng, X.; Guo, X.; Wang, H.; Gorlatova, M.; Zhang, M.; Li, H.; et al. FedSEA: A Semi-Asynchronous Federated Learning Framework for Extremely Heterogeneous Devices. In Proceedings of the 20th ACM Conference on Embedded Networked Sensor Systems, Boston, MA, USA, 6–9 November 2022; pp. 106–119.
20. Cha, N.; Du, Z.; Wu, C.; Yoshinaga, T.; Zhong, L.; Ma, J.; Liu, F.; Ji, Y. Fuzzy Logic Based Client Selection for Federated Learning in Vehicular Networks. *IEEE Open J. Comput. Soc.* **2022**, *3*, 39–50. [CrossRef]
21. Wang, S.Q.; Tuor, T.; Salonidis, T.; Leung, K.K.; Makaya, C.; He, T.; Chan, K. When Edge Meets Learning: Adaptive Control for Resource-Constrained Distributed Machine Learning. In Proceedings of the IEEE Infocom 2018—IEEE Conference on Computer Communications, Honolulu, HI, USA, 16–19 April 2018.
22. Tran, N.H.; Bao, W.; Zomaya, A.; Nguyen, M.N.H.; Hong, C.S. Federated learning over wireless networks: Optimization model design and analysis. In Proceedings of the IEEE Infocom 2019—IEEE Conference on Computer Communications, Paris, France, 29 April–2 May 2019.
23. Nguyen, D.C.; Ding, M.; Pathirana, P.N.; Seneviratne, A.; Li, J.; Vincent, P.H. Federated learning for internet of things: A comprehensive survey. *IEEE Commun. Surv. Tutor.* **2021**, *23*, 1622–1658. [CrossRef]
24. Yang, W.; Xiang, W.; Yang, Y.; Cheng, P. Optimizing Federated Learning with Deep Reinforcement Learning for Digital Twin Empowered Industrial IoT. *IEEE Trans. Ind. Inform.* **2023**, *19*, 1884–1893. [CrossRef]
25. Zheng, J.; Li, K.; Mhaisen, N.; Ni, W.; Tovar, E.; Guizani, M. Exploring Deep-Reinforcement-Learning-Assisted Federated Learning for Online Resource Allocation in Privacy-Preserving EdgeIoT. *IEEE Internet Things J.* **2022**, *9*, 21099–21110. [CrossRef]
26. Sun, R.; Matolak, D.W.; Rayess, W. Air-ground channel characterization for unmanned aircraft systems—Part IV: Airframe shadowing. *IEEE Trans. Veh. Technol.* **2017**, *66*, 7643–7652. [CrossRef]
27. Yang, Z.; Chen, M.; Saad, W.; Hong, C.S.; Shikh-Bahaei, M. Energy efficient federated learning over wireless communication networks. *IEEE Trans. Wirel. Commun.* **2020**, *20*, 1935–1949. [CrossRef]
28. Huo, Y.H.; Song, C.X.; Zhang, J.; Tan, C. DRL-based Federated Learning Node Selection Algorithm for Mobile Edge Networks. In Proceedings of the 2022 IEEE 14th International Conference on Advanced Infocomm Technology (ICAIT), Chongqing, China, 8–11 July 2022.
29. Wang, S.Q.; Tuor, T.; Salonidis, T.; Leung, K.K.; Makaya, C.; He, T.; Chan, K. Adaptive federated learning in resource constrained edge computing systems. *IEEE J. Sel. Areas Commun.* **2019**, *37*, 1205–1221. [CrossRef]