electronics

MDPI

Article

Task Offloading Scheme for Survivability Guarantee Based on Traffic Prediction in 6G Edge Networks

Zhengjie Sun [1], Hui Yang [1,*], Chao Li [1], Qiuyan Yao [1], Ao Yu [2], Jie Zhang [1], Yang Zhao [3], Sheng Liu [3] and Yunbo Li [3]

[1] State Key Laboratory of Information Photonics and Optical Communications, Beijing University of Posts and Telecommunications, Beijing 100876, China; sunzhengjie@bupt.edu.cn (Z.S.); lc96@bupt.edu.cn (C.L.); yqy89716@bupt.edu.cn (Q.Y.); zjls@aliyun.com (J.Z.)
[2] Kuaishou Technology Co., Ltd., Beijing 100085, China; yuaobupt@gmail.com
[3] Department of Fundamental Network Technology, China Mobile Research Institute, Beijing 100053, China; lc96bupt@aliyun.com (Y.Z.); gggls@aliyun.com (S.L.); sss1006@aliyun.com (Y.L.)
* Correspondence: yanghui@bupt.edu.cn

Abstract: With the development of sixth-generation (6G) mobile networks, the rise of emerging intelligent services has led to a huge increase in traffic. As an important technology to support the development of 6G, mobile edge computing (MEC) effectively meets the ultra-low latency requirements of most emerging services. However, due to the limited processing capacity of edge nodes, overload on any edge node will cause service degradation, interruption, and even node failure, weakening the advantages of MEC and reducing the survivability of the whole network. In this paper, we propose a task offloading scheme based on traffic prediction for node-overload protection to ensure the survivability of the 6G edge networks. We transformed the network survivability guarantee problem into a task offloading problem under the constraint of future available resources based on traffic prediction and developed a particle swarm optimization algorithm based on policy gradient (PSO-PG) to jointly optimize offloading decisions, routing, and computing resource allocation. Simulations verify the effectiveness of our proposed scheme and guarantee the survivability of 6G edge networks. Meanwhile, evaluation in multiple scenarios with different node scales has verified the wide applicability of our work.

Keywords: 6G edge networks; mobile edge computing; survivability guarantee; node-overload protection; traffic prediction; task offloading

check for updates

Citation: Sun, Z.; Yang, H.; Li, C.; Yao, Q.; Yu, A.; Zhang, J.; Zhao, Y.; Liu, S.; Li, Y. Task Offloading Scheme for Survivability Guarantee Based on Traffic Prediction in 6G Edge Networks. *Electronics* **2023**, *12*, 4497. https://doi.org/10.3390/electronics12214497

Academic Editor: Djuradj Budimir

Received: 19 September 2023
Revised: 22 October 2023
Accepted: 30 October 2023
Published: 1 November 2023

Copyright: © 2023 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (https://creativecommons.org/licenses/by/4.0/).

1. Introduction

In recent years, with the rapid advancement of 6G technological reformation, the rise of emerging intelligent services such as intelligent traffic control, holographic communication, mission-critical services, and smart homes has put forward higher requirements for ultra-low latency [1–3]. According to IDC predictions, global network traffic will reach 175ZB by 2025 [4,5]. The rapid growth of network traffic and the constantly updated demand for new service scenarios are powerful driving forces for the development of 6G [6]. The emerging intelligent services driven by 6G have increasingly urgent demands for high-reliability network performance and ultra-low latency [7,8]. In order to meet the more stringent requirements of intelligent services, mobile edge computing (MEC) is considered a key technology to solve the difficulties faced by 6G, representing an important component of distributed architecture and edge intelligence in the 6G vision [8].

Supported by MEC, massive intelligent services access computing resources with lower delay [9–11]. Intelligent services in new scenarios have the features of rapid changes in demands and large amounts of data. Constrained by limited resources and computing capacity on edge nodes, it is difficult to meet the ultra-low latency requirements of expanding large-scale services. On the one hand, when the edge node receives a large

number of processing requests in a short time, the traffic bursts will cause node overload, leading to service degradation or interruption, weakening the advantages of MEC, which is intolerable for delay-sensitive services [12]. On the other hand, node failures caused by node overload significantly reduce the available resources, while fault recovery or node replacement is stiff and time-consuming [8,13]. It is urgent to improve the survivability of 6G edge networks.

The traffic surge brought about by the development of emerging services has led to a shortage of computing and network resources, and the lack of control over the use of available resources can easily lead to the overprovisioning of node resources, increasing the possibility of node overload in 6G edge networks. Meanwhile, the development of 6G expands the coverage scale of the network and increases the risk range of node failure. The decrease in survivability will seriously affect the quality of service and user experience. Ensuring the development of 6G networks requires evolving towards intelligence, and the key challenge in guaranteeing survivability is the multi-node collaborative scheduling of multidimensional resources. With the continuous improvement of artificial intelligence (AI) technology, depending on its management capacity for complex systems, it has become mature enough to be deployed in complex networks. AI-assisted algorithms have become a feasible solution to solve the difficulties of future complex network systems [14,15]. In the classic scenarios covered by 6G, ultra-low latency and ultra-high reliability can be realized with the assistance of artificial intelligence. Therefore, designing the pro-active mechanism by utilizing AI techniques that prevent node overload has become a promising solution for ensuring network survivability. However, a large number of studies have focused on carrying out dynamic scheduling based on load balancing without attaching importance to the future available resource utilization. Since traffic prediction can be representative of future resource demand, it should similarly reflect the specifics of the future available resource. Excellent predictive capacity is a crucial means of evaluating future available resources. Directly deploy AI algorithms on large-scale edge nodes to predict future available resources based on existing collected data. Furthermore, offloading blindly may lead to excessive resource competition for the tasks deployed on the same edge node, resulting in high latency overhead. Therefore, integrating future available resources into the task offloading scheme for node overload protection is of great significance for the survivability of the entire 6G edge network.

In this paper, we propose a task offloading scheme based on available resources for survivability guarantee, aiming to ensure the survivability of 6G edge networks. The scheme is composed of two algorithms, including a mapping of future available resources based on traffic prediction and an overload protection scheme based on a particle swarm optimization algorithm based on policy gradient (PSO-PG). Firstly, in order to maximize resource utilization, the predicted traffic scale is applied to map future available resources. Secondly, based on future available resources, a PSO-PG heuristic algorithm was developed to determine task offloading schemes from a global optimization perspective to provide reliable overload protection. Our contributions can be summarized as follows:

- We deploy the accurate traffic prediction model on edge nodes by constructing a mapping between traffic prediction and future available resources to achieve resource visualization of the entire 6G edge network, so as to maximize the advantages of using edge resources while ensuring the survivability of the network.
- In response to the highly dynamic nature of networks, they may face the challenge of failing to adapt fixed algorithmic parameters to mutating network environments. We develop the PSO-PG algorithm for the design of node-overload protection schemes in dynamic networks. The key parameters of the PSO algorithm are adaptive adjustments by policy gradients (PGs) that interact with the actual network environment. The improved algorithm solves the problems of difficulty in manually configuring parameters and the inability to be updated in time according to the actual operating onditions.
- Under the constraints of the future available resources, utilizing the advantages of the PSO-PG algorithm, we propose an innovative survivability guarantee framework

for 6G edge networks. It integrates the prediction of required processing power and the process of task offloading. The scheme effectively realizes the joint optimization of adjustment offloading decisions with routing and computing resources to match the network survivability guarantee, minimizing the increase in service delay due to insufficient or overallocation of resources while guaranteeing the performance of the whole network.

The rest of this paper is organized as follows: Section 2 introduces the related work. Section 3 introduces the system architecture and scheme process. Section 4 introduces task offloading based on traffic prediction schemes for node-overload protection. Section 5 presents the evaluation of the proposed scheme and analyzes the simulation results.

2. Related Work

The service degradation and node failures caused by node overload greatly reduced the survivability of the entire edge network, which means a huge disaster for both users and operators. Node-overload protection to ensure the survivability of the whole network has become a research hotspot.

In the study of [16], the protecting strategy was determined by the single threshold at the current moment of the network state. The services would be offloaded to any other server when exceeding the threshold. In [17], the authors divided the task into multiple subtasks, which would be offloaded to multiple edge nodes for overload protection by considering both network reliability and latency. In [18], the protecting scheme, which is based on offloading all tasks from the overloaded node to the destination node, was decided by arriving at the maximum reduction of energy consumption under the delay constraint. Liu et al. [17] had proposed the offloading scheme for overload protection by prioritizing the latency-sensitive tasks, in which they schedule the latency-sensitive tasks to the cloud server according to the set priority. Sun et al. [19] proposed a delay-aware offloading scheme in which all tasks will be offloaded to the nearest node to minimize transmission latency.

In [20], due to the constraints of the load balancer, task offloading would be developed among edge servers efficiently. In [21], value iterations of the Markov Decision Process (MDP) were used to perform cost-effective task offloading. In [22], the authors transformed the task offloading problem into a Markov process to develop the optimal strategy through the mathematical framework of the set cost model.

In [23], integrating the cache resources and computing resources through deep reinforcement learning, the optimized overload protection scheme was developed through multiple iterations. In [24], the proposed method would allow each user to independently learn the efficient offloading destination to avoid overload, achieving the joint optimization of global power consumption and latency. In [25], the authors adopted an adaptive admission control strategy for overloaded node protection. The strategy utilized the observation state model of reinforcement learning, combined with the actual consideration of offloading costs, to develop an offloading strategy that adapts to a time-varying environment.

In [26], the authors proposed a joint optimization method for computing offloading and content caching based on traffic prediction to predict the traffic of each edge server. By deploying a deep spatiotemporal residual network, followed by the use of genetic algorithms to implement the selection of the destinations with the lowest execution latency for determining the destinations for task offloading. In [27], the authors proposed a collaborative computing offloading approach using deep reinforcement learning to minimize latency by combining the Mixed-Integer Nonlinear Programming (MINLP) optimization problem. In [28], the authors proposed a fuzzy neural network (FNN) and game theory task offloading scheme, applying FNN to predict future traffic and adopting game theory to determine the optimal task offloading strategy for the user so as to satisfy the delayed requests for services.

In [10], the authors proposed a multi-task cooperative computing mechanism for overload protection in 6G networks, which focuses on node selection and path optimization to

improve the efficiency of computational task execution and quality of service by optimizing the delay-aware multi-task cooperative computing objective. In [29], the authors proposed a novel profile-based data-driven VNF performance-resource analysis (PDPA) framework that analyzes the complex relationship between network performance KPIs, resource allocation, and utilization to achieve assurance for next-generation networks. In [30], the authors proposed deep reinforcement learning based on migration techniques to explore network slicing, guaranteeing the normal operation of the network under extreme conditions, avoiding unstable system performance, and violating network service level agreements (SLAs). In [31], the authors improved the probability of smart device connectivity by calculating the probabilistic sensitivity of the optimal paths between smart devices and would also optimize the communication delay and data preprocessing time by analyzing the different sensitivities to improve the stability of smart device access to the network using machine learning regression algorithms.

These studies focused on latency, regardless of the impact of policy execution on the performance of the entire network. In addition, these studies tend to fix the algorithm parameters for task offloading. The algorithm parameters cannot be adjusted to follow the dynamic network state, which is inflexible. Our work would compensate for these shortcomings by jointly applying traffic prediction and a task offloading scheme. By adjusting the algorithm parameters in a timely manner following the dynamic network state, the scheme has achieved an optimal trade-off between guaranteed service latency and the performance of the network.

3. System Architecture

In this section, we first describe the architecture of the 6G edge network and develop an optimal task offloading scheme for node-overload protection with guaranteed quality of service and survivability of the 6G edge networks. The symbols appearing in this section are explained in Table 1.

Table 1. Symbol.

Symbol	Meaning
M_L	The local MEC server
P_i	The taskset
E_m	The candidate destination node set
Tr_i	The transmission delay
Tc_i	The processing delay
T_i	The total delay
R	The transmission rate
δ_m	The limit ratio of the maximum computing capacity
C_k	The computing capacity of the MEC server
$\lambda_{i,m}$	The task i is offloaded to the migrated server m
$T_{i,max}$	The delay threshold of task i
d_i	The data size of task i
$L^i(a_o,a_d)$	The task i uses the link (a_o,a_d)
Di,m	The number of frequency slots allocated for transmitting task i
$fsi(ao,ad)$	The frequency slot fs in the optical path
Vi,m	The computing capacity allocated to task i by the server m
c_i	The computing capacity required to process task i
DFS	The capacity of the frequency slot of the optical path

3.1. Description of Network Architecture

The MEC servers with computing capability and storage capability have been placed at the edge nodes in 6G edge networks, providing users with a low-delay processing guarantee [32]. The 6G edge network architecture is shown in Figure 1, which is composed of a cloud layer, an edge layer, and a device layer. The edge layer is juxtaposed by the MEC servers and the open-flow switches. Under the unified coordination of the

controller, the MEC servers frequently interact through the ultra-large-capacity optical fiber interconnection [33]. It is worth noting that the computing capacity of the MEC server is limited.

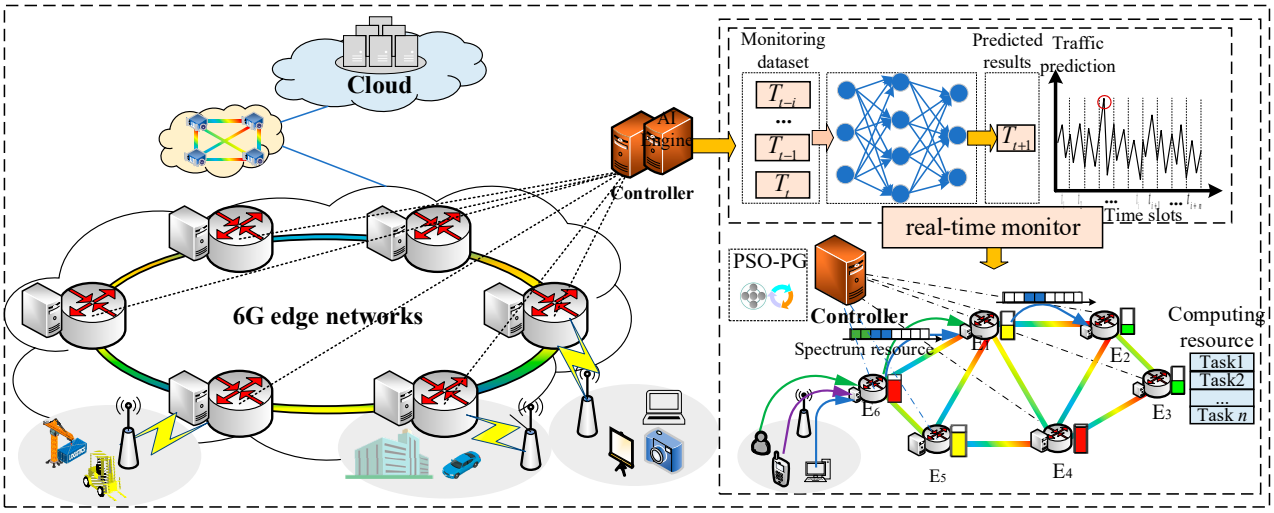


Figure 1. The edge collaboration based on traffic prediction for node-overload protection in 6G edge networks.

In the architecture of 6G edge networks, each edge node interacts with the network environment in real time to realize the perception of the network state. Simultaneously, routine monitoring is continuously implemented, and reporting information is sent to the cloud periodically. The cloud also sends the state information of the relevant node to other edge nodes. With a certain degree of autonomy, the edge node can process tasks directly from the device layer in a timely manner.

In this paper, we focus on the constraints of average delay and limited computing resources in the case of single-node overload. The whole process includes node-overload prediction based on traffic prediction and task offloading for overload protection, as shown in Figure 1. The purpose of the traffic prediction is to predict the scale and timing of burst traffic. The node-overload protection scheme will be developed through task offloading based on edge collaboration before node-overload.

3.2. The Establishment of an Offloading Model

In order to obtain the network state at a certain time in a dynamically changing network, the network state in a short time is recognized as the quasi-static network state. The task offloading based on edge collaboration can solve the node-overload problem to ensure the survivability of the 6G edge networks, supported by the global view of the entire system resources provided by the controller.

In this paper, we mainly describe the protection scenario of single-node overload. Assuming that the 6G edge networks are composed of m edge nodes, the predicted overload node is labeled as M_L, and the task set generated on it is identified as $\{P_i | i = 1, 2, \ldots, N\}$. The set of candidate destination nodes is defined as $\{E_m | m = 1, 2, \ldots, M-1\}$, which would accept the task offloading requests. It is worth noting that each task is required to be processed within its maximum tolerance latency. The computing resource will be provided by the destination edge node.

The controller continuously monitors the state of the whole edge network in real time, including the available resources and computing capacity. The computing capacity is determined by the processing capacity of the CPU on the MEC server. The computing capacity is set to C_k, which represents the CPU cycles completed per second. In order to satisfy the QoS, it is necessary to set the limit ratio of the maximum computing capacity,

which is determined by the predicted traffic scale in the next timeslot. The limit ratio is indicated as δ_i,which presents a negative correlation with the predicted traffic.

Latency is an important evaluation metric that cannot be ignored during the whole process of node-overload protection. In this process, the latency consists of the transmission delay from the terminal to the local server, the transmission delay from the local server to the destination MEC server, and the processing delay on the destination MEC server.

It is worth noting that the terminal connects to the local MEC server through the base station (BS), which consists of two components: the wireless delay from the terminal to the BS and the forward transmission delay from the BS to the local MEC server [30]. For each task, the delay of these two components is unavoidable, which would proceed indiscriminately, independent of the subsequent offloading decisions. Therefore, this part of the delay will no longer be considered in this section.

The edge collaboration relies on EONs, which are supported by flexible grid technology. The spectrum resources of each link are divided into frequency slots (*FSs*), where the capacity of each *FS* is set to B = 6.25 Gbps. In this paper, the modulation format is BPSK, and the transmission rate is R = 6.25 Gbps. The transmission delay of task i is described in Formula (1).

$$Tr_i = \sum_{m \in E_m} \frac{\lambda_{i,m} d_i}{RD_{i,m}} \tag{1}$$

where $\lambda_{i,m}$ represents whether task i is offloaded to the destination server m. d_i represents data size of task i. R indicates the transmission rate of each spectrum slot. $D_{i,m}$ is the number of frequency slots allocated for transmitting task i to the server m.

The MEC servers are equipped with high-speed CPU processors, whose processing delay is defined by Equation (2).

$$Tc_i = \sum_{m \in E_m} \frac{\lambda_{i,m} c_i}{V_{i,m}} \tag{2}$$

where c_i represents the computing capacity required to process task i, implying the number of CPU cycles required. $V_{i,m}$ represents the computing capacity allocated to task i by the destination server m.

Therefore, the total delay of the whole process of task offloading is exhibited as follows:

$$T_i = \sum_{m \in E_m} \left(\frac{\lambda_{i,m} d_i}{RD_{i,m}} + \frac{\lambda_{i,m} c_i}{V_{i,m}}\right) \tag{3}$$

Objective:

$$\min \frac{1}{N} \sum_{i \in P_i} T_i \tag{4}$$

Constraint:

$\frac{c_i}{V_{i,m}} \leq T_{i,\max}$ ensures that the processing delay of the task is within its maximum tolerable delay.

$\sum_{i \in P_i} \lambda_{i,m} V_{i,m} \leq \delta_m C_m$ indicates that the amount of tasks allocated to the edge server should be within the computing capacity of the server, and δ_m indicates the limit ratio of the maximum computing capacity of the server m.

When $\sum_{m \in E_m} \lambda_{i,m} = 1$, each task is allowed to be allocated to only one server for processing.

$Lp^{i,d}(a^o, a^d) D_{i,d} = fs^i(a^o, a^d)$ represents that the frequency slot fs in the optical path (a^o, a^d) occupied from the local MEC to the destination MEC.

When $\sum_{i=1}^{n} fs^i(a^o, a^d) \leq D_{FS}$, it is constrained that the number of frequency slots occupied by all tasks should be within the capacity of the frequency slot of the optical path.

When $(fs^i(a^o, a^d)_{,f} + fs^i(a^o, a^d)_{,f+1}) \geq \sum_{f' \in [f+2, FS]} fs^i(a^o, a^d)_{,f'}$, it is worth noting that the allocation of frequency slots follows the principle of proximity.

4. Task Offloading Scheme Based on Traffic Prediction for Node-Overload Protection

In order to improve the survivability of 6G edge networks and guarantee services with low latency processing, we propose a task offloading scheme based on traffic prediction for node-overload protection, as described below in detail.

4.1. Traffic Prediction Based on LSTM

In the architecture of the 6G edge network, the expanding scale of devices has driven the generation of a large number of new services. MEC has become the primary way to process services generated by devices. The whole process should be supported by low delays so as to improve the quality of the experience. Since the predicted traffic can reflect the future available resources, we have collected the historical traffic recorded by the edge nodes, which can be used to predict the future traffic scale by deploying the prediction model on the edge nodes. The main purpose of traffic prediction is to design a pro-active mechanism that prevents node overload by measuring the scale of traffic bursts. With accurate prediction, we will implement the task offloading scheme in a timely manner before node overload.

Due to the fact that traffic flow is a typical time series, we apply the Long-Short Term Memory (LSTM) algorithm to the traffic prediction model. LSTM is an improved recursive neural network algorithm that overcomes the problem of inaccurate learning of past information due to gradient vanishing in RNN [34]. By incorporating memory function and dynamically adjusting the correlation weight coefficients between sequences, LSTM achieves high-precision prediction of long and short time series. Monitoring and collecting the historic traffic generates a periodic time-series list by edge nodes as input to the LSTM. The entire process is shown in Figure 2.

The forget gate would determine which information in the cell state could be transferred from the previous moment to the current moment. The previous information would be reserved when the output was equal to 1. Otherwise, the previous information would be dropped, as shown in Formula (5).

$$f_t = \alpha(W_f[h_{t-1}, x_t] + b_f) \tag{5}$$

In the above formula, f_t represents the forget gate state. x_t is the input time-series.α represents the activation function. h_{t-1} is output value of the last hidden layer. W_f is the weight value of forget gate and input gate. b_f is the bias value.

The input gate can be used to determine which information could be saved into the cell state, conducted in two sections. First, the input gate is shown in Formula (6). Similar to the f_t, the matrix value in the i_t represents the discard or retention of the information in $\overline{C}$. Conduct the candidate cell state $\overline{C}$ for saving x_t and h_{t-1}. To determine the useful memories by dot product with i_t, the retained information is added as the new memory to the cell state, as shown in Formula (7). The update cell state is shown in Equation (8), which is jointly updated by the useless information forgotten in f_t and the latest useful information in i_t.

$$i_t = \alpha(W_i[h_{t-1}, x_t] + b_i) \tag{6}$$

$$\overline{C} = \tanh(W_C[h_{t-1}, x_t] + b_c) \tag{7}$$

$$C_t = \overline{C} \times i_t + C_{t-1} \times f_t \tag{8}$$

Finally, we obtain the output value, which will be determined by the state of the cells. First, we identify the sigmoid layer to determine which part of the cell state will be

output. Next, we process the cell state through tanh (to obtain a value between −1 and 1) and multiply it with the output of the sigmoid gate. Finally, output the corresponding prediction results.

$$o_t = \alpha(W_o[h_{t-1}, x_t] + b_o) \tag{9}$$

$$h_t = o_t \times \tanh(C_t) \tag{10}$$

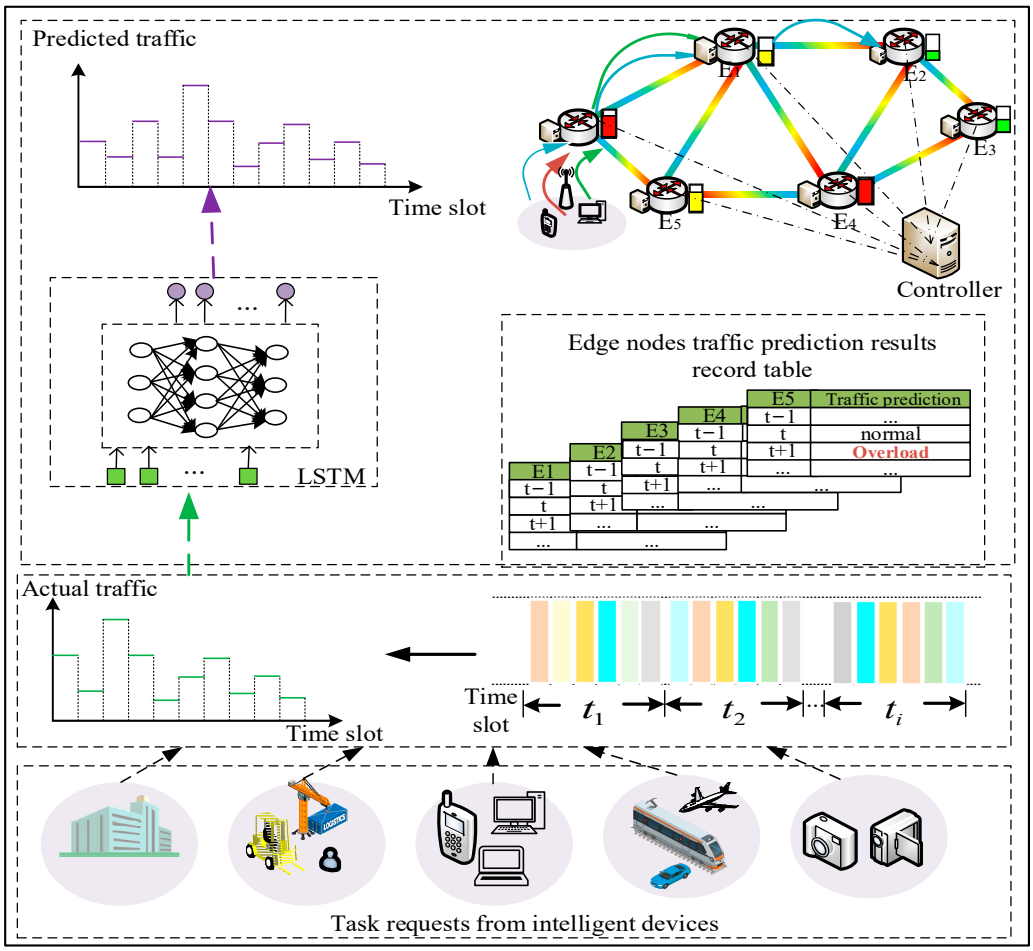


Figure 2. The process of traffic prediction.

Through the action of the fully connected hidden layers, we obtain the output time series, which represents the future traffic scale on edge nodes. Since traffic prediction can be representative of future resource demand, it should similarly reflect the specifics of the future available resource. The predicted traffic scale is mapped to future available resources, represented by the limit ratio of the maximum computing capacity provided by the server, as a basis for accepting or rejecting offloading requests in the future. The node-overload protection scheme based on traffic prediction is described in detail in the next section.

LSTM is a prediction solution based on an asymmetric cost function, owing to the fact that the cost penalty set by the cost function is not necessarily constant but is related to the loss of service amount due to insufficient supply [35]. The accuracy of traffic prediction is critical to the outcome of resource allocation, and incorrect prediction may lead to over- or under-provisioning of resources, thus increasing the cost of operating the network. Oversupply occurred when the predicted traffic was higher than the actual traffic, which would lead to the allocation of more resources than required. Unnecessary resource allocation leads to increased costs. Conversely, under-provisioning leads to

QoS degradation by allocating fewer resources than required, inevitably resulting in loss of service.

4.2. PSO-PG-Based Task Offloading Scheme under Multi-Edge Collaboration

In our previous work, we applied LSTM to predict the future traffic scale. To avoid node overload, our work would focus on task offloading based on edge collaboration for node-overload protection. However, with the expansion of network scale, it is difficult for the traditional MINLP algorithms to obtain optimal solutions within an acceptable timeframe. Therefore, in the heuristic framework of the PSO algorithm, we apply the PSO-PG to develop the node-overload protection scheme based on edge collaboration for ensuring the survivability of the network, which balances the trade-off between average latency and resource usage [36,37].

The PSO-PG algorithm, which utilizes a policy gradient to guide the optimization direction of PSO, interacts with the intelligent agent, having achieved parameter tuning for the PSO algorithm through timely feedback [38,39]. The interaction process of PSO-PG is shown in Figure 3. Task offloading is based on edge collaboration for node-overload protection, whose key idea is to thoroughly scan all possible combinations of available edge servers and network resources and finally develop the optimal offloading scheme. Specifically, the offloading scheme is divided into making offloading decisions and selecting the destination MEC server.

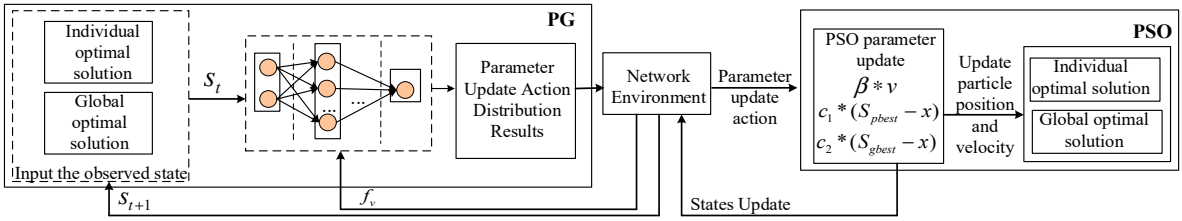


Figure 3. The interaction process of PSO-PG algorithm.

First encoding the particles, the *ith* particle (the feasible scheme for task *i*) is defined as:

$$x_i = \{p_{i,d}, p_{i,s}, p_{i,p}, p_{i,c}, p_{i,o} | i \in Q\} \tag{11}$$

where $p_{i,d}$ is the destination MEC server d selected for offloading the task i, $p_{i,s}$ is the number of FSs allocated to the task i, $p_{i,p}$ is the routing path allocated to the task i, $p_{i,c}$ is the computing resource allocated for task i, $p_{i,o}$ is the allocation order of the task i.

The particle swarm coding with the *kth* iteration ${X_i}^k$, which represents the detail candidate offloading scheme.

$$ {X_i}^k = \left\{ {x_1}^k, {x_2}^k, \ldots, {x_Q}^k \right\} \tag{12}$$

For the selection of destination MEC servers, we prioritize selecting MEC servers with lighter workloads in the prediction results, which means there are more available resources. In order to minimize the average delay, all remaining computing resources are allocated to offloading tasks while ensuring sufficient computing resources for tasks supported on the candidate servers. In routing resource allocation, path selection follows the shortest K-path principle, and spectrum allocation follows the first-fit principle.

As the heuristic optimization algorithm, PSO-PG explores the optimal solution through particle iteration. Typically, a particle swarm is composed of N particles $P[V_i, X_i]$, which are associated with velocity and position in the dimension R of the search space.

$$V_i = \left\{ V_i^1, V_i^2, \ldots, V_i^R \middle| i = 1, 2, \ldots, N \right\} \tag{13}$$

$$X_i = \left\{ X_i^1, X_i^2, \ldots, X_i^R \middle| i = 1, 2, \ldots, N \right\} \tag{14}$$

The optimization objective of PSO is to find the global optimal solution by iteration in the feasible optimal solution space. In each merit-seeking process, the particle velocity and position update are shown in Equations (15) and (16).

$$V_i^{t+1} = \omega \times V_i^t + c_1 \times r_1 \times (S_{pbest,i}^{\ t} - X_i^t) + c_2 \times r_2 \times (S_{gbest,i}^{\ t} - X_i^t) \tag{15}$$

$$X_i^{t+1} = X_i^t + V_i^{t+1} \tag{16}$$

ω is the inertia weight, c_1 is the cognitive acceleration coefficient, c_2 is the social acceleration coefficient, and r_1 and r_2 are uniformly distributed (0,1) random numbers.

The velocity V_i^{t+1} describes the final optimization direction based on the accumulated velocity of particles in the past and the current particle optimization direction, which are obtained by calculating the difference between the optimal and current positions of particles. During the optimization process, the positions of the particles are continuously updated, which update stops if the change in the current search direction is small enough that the optimization process is close to optimal.

In the traditional PSO algorithm, due to the fixed parameters, it cannot make some adjustments based on actual operation, which can lead to premature convergence of the algorithm. In order to overcome the challenge, we ultimately adopt the improved algorithm PSO-PG to achieve dynamic adjustment of key parameters.

Relying on the intelligent agent, the PG can interact with the network environment in real time, and the environment will input the current state S_t into the intelligent agent. The output action A_t will be applied to the PSO, and the action results will feedback the corresponding reward value R_t with the latest state S_{t+1} to the intelligent agent. The agent will determine the parameters for tuning PSO based on the feedback reward values, helping the particle determine the optimization direction, as shown in Algorithm 1.

Algorithm 1: The algorithmic process of PSO-PG.

1. Initialize the PG training parameters, learning rate ∂ and the loss function L_θ.
2. Initialize the size of particle swarm N, the maximum number of iterations $It_{\max}$, inertia weight β and learning factors c_1 and c_2.
3. Obtain the initialization fitness value according to the initial parameters.
4. **For** $It \leq It_{\max}$ **do**
5. Input the individual optimal solution S_{pbest} and the global optimal solution S_{gbest} calculated by PSO into the PG.
6. Output action probability distribution A_t through PG iterative operation.
7. The action selection function will select action a according to A_t and input a to the PSO.
8. PSO receives the action a according to the update rule for parameters to update ω, c_1, and c_2.
9. Update particle swarm velocity and position according to Formulas (20) and (21).
10. Update S_{pbest} and S_{gbest}, and calculate the new *Fitness* value according to Formula (24).
11. Return the new reward value R_t by PG.
12. Store the obtained the state s, action$a$, and reward R_t in intelligent agent.
13. Input the updated S_{pbest} and S_{gbest} into PG.
14. **End For**

The input state S_t is the relative distance between the particle and the global best position ${S_{gbest}}^t_{,i}$ and the relative performance between the particle and the global best fitness.

$$S = \begin{cases} d = \sum_{i=1}^{N} x_{gbest,i} - x_i \\ f = \sum_{i=1}^{N} F_{gbest,i} - F_i \end{cases} \tag{17}$$

The intelligent agent performs the computation of the output action probabilities by means of the iterative neural network hidden layer activation function, which will select the action A_t input with the highest output probability for the network environment. The output action probability distribution is a seven-dimensional action space.

$$A_t = [A_0,\ A_1,\ A_2,\ \ldots,\ A_6] \tag{18}$$

$A_0 + A_1 + A_2 + \ldots + A_6 = 1$, where A_0, A_1 represents the increase and decrease of ω, A_2, A_3 represents the increase and decrease of c_1, A_4, A_5 represents the increase and decrease of c_2, and A_6 represents no change of parameters.

When the calculated *Fitness* is compared with the result of the previous iteration, the reward value R_t will give positive feedback when the *Fitness* is improved. Conversely, R_t will give negative feedback.

Adaptive updating of the PSO-PG and the improved formulae for particle velocity and position updates are as follows:

$$V_i^p(t+1) = \omega_t \times V_i^p(t) + c_{1,t} \times r_1 \times ({S_{pbest}}^t_{,i} - X_i^p(t)) + c_{2,t} \times r_2 \times ({S_{gbest}}^t_{,i} - X_i^p(t)) \tag{19}$$

$$X_i^p(t+1) = X_i^p(t) + V_i^p(t+1) \tag{20}$$

The resource allocation of the offloading scheme is evaluated by the *Fitness* at the time slot t, which is used to guide the particle towards the optimal search. The *Fitness* value is a trade-off between latency, resource utilization and blocking rate.

Average delay: the average response delay of all tasks as described in Section 3, $aveT = \frac{1}{N} \sum_{i \in P_i} T_{ti}$.

Blocking probability: the proportion of the number of blocked tasks to the total number of tasks. $Bp = \frac{N_b}{N}$, where N_b represents the number of blocked tasks and N represents the number of all tasks.

Resource utilization: the percentage of the number of occupied frequency slots at time slot t. When $\mathrm{Re} = \frac{Fs_m}{Fs}$, Fs_m represents the number of occupied frequency slots, and Fs indicates the number of all frequency slots.

Thus, the *Fitness* function is defined as

$$Fitness = \alpha \times Bp + \beta \times aveT + \varepsilon \times \frac{1}{\mathrm{Re}} \tag{21}$$

α, β, and ε are the weight coefficient.

From the above equation, it can be perceived that the *Fitness* is a combination of three components. In order to perform the optimized offloading scheme, it is necessary to optimize towards minimizing *Fitness* value.

In PSO-PG, the particles would be updated and evaluated by Equations (19) and (20), as shown in the Algorithm 2, in which it updates ${S_{pbest}}^t_{,i}$ and ${S_{gbest}}^t_{,i}$ by contrasting *Fitness*, and finally obtains the global optimal solution ${S_{gbest}}^t_{,i}$.

Algorithm 2: Task offloading for node-overload protection based on PSO-PG.

1. Obtain the predicted node overload.
2. Establish an ascending sort as a set of candidate offloading nodes;
3. Initialize particle position $X_i{}^k = \left\{ x_1{}^k, x_2{}^k, \ldots, x_Q{}^k \right\}$ and particle velocity vector $v_i{}^k = \left\{ v_1{}^k, v_2{}^k, \ldots, v_Q{}^k \right\}$;.
4. **While** the algorithm has not converged **do**
5. **For** each particle i **do**
6. Decode each particle swarm through (12) to obtain the offloading scheme.
7. Select the initial destination MEC server and allocation resource.
8. Calculate the *Fitness* (F_i) by Formula (21).
9. $S_{gbest} = \min(S_{pbest.i})$.
10. if $Fitness(F_i) < Fitness(S_{pbest,i})$ then
11. $S_{pbest,i} = F_i$
12. end if
13. if $Fitness(S_{pbest,i}) < Fitness(S_{gbest})$ then
14. $S_{gbest} = S_{pbest,i}$
15. end if
16. **End For**
17. Choose the smallest *Fitness* value as the optimal scheme and update particle through the Formulas (19) and (20).
18. **End While**
19. **Output** the optical scheme S_{gbest}.

5. Evaluation

The detailed description of the scheme mentioned above is focused on two parts: traffic prediction and task offloading. AI-integration algorithms need to carry out a large amount of data for training and learning, and the lack of data will affect the effect of training. Moreover, a huge amount of high-performance arithmetic power is needed to support the AI algorithm training process [40–42]. Therefore, in this program, we deploy the algorithm on a high-performance GPU virtual machine for training. The simulation environment in this paper is run on a multi-core server with 12 2.5 GHz Intel Core i5-7200U CPU cores, 2 NVIDIA Titan XP GPU cores, and 32 GB of RAM for accelerated training of neural networks in a virtual machine. In this section, similarly, we divide the simulation into two parts for evaluation.

5.1. Simulation Setup and Results Analysis for Traffic Prediction Based on LSTM

We have trained the LSTM algorithm based on the historical traffic dataset on edge servers. The dataset used for simulation was traffic data collected every five minutes in July 2019 from a university data center in Beijing, China. The trained model is deployed on the edge nodes for traffic prediction. We collected 10,000 pieces of data from the historical dataset, including timestamp, arrival time, processing time, source IP and port, destination IP and port, packet size, device logs, etc. Eighty percent of the dataset is used for training and 20% for testing. The parameter settings for LSTM are depicted as follows: input_size = 5, batch_size = 32, learning rate = 0.001, epoch is set to 100, and the maximum hidden layer is set to 2, hidden units = 256.

Support Vector Machines (SVM) have the advantage of high prediction accuracy in small datasets due to their strict mathematical-logical nature. Deep-learning Neural Networks (DNN) have a strong nonlinear fitting capability and a wide range of applications in prediction using the ability to find optimized solutions at high speed. Recurrent Neural Networks (RNN) have the ability to capture dependencies in time series due to the presence of recurrent links in the network, empowering algorithms with powerful memory. We make a comprehensive comparison between the LSTM prediction algorithm proposed in this paper and the algorithms mentioned above, which are all prominent in the field of prediction.

Firstly, we compared the prediction accuracy of LSTM with SVM, DNN, and RNN, as shown in Figure 4. With the increasing number of epochs, the prediction accuracy of each algorithm presents an upward trend. It can be obviously observed that LSTM performs exceptionally well in prediction accuracy among numerous algorithms. In Figure 5a, the blue trajectory represents the actual traffic, while the green trajectory represents the predicted traffic predicted by LSTM, which makes it evident that the predicted traffic trend is almost identical to the actual traffic trend. Figure 5b shows the trend of traffic trajectories over a period of time, which intuitively reveals the comparison between the predicted trajectories of different algorithms and the actual trajectories. These results indicate that our selected algorithm has excellent performance in traffic prediction.

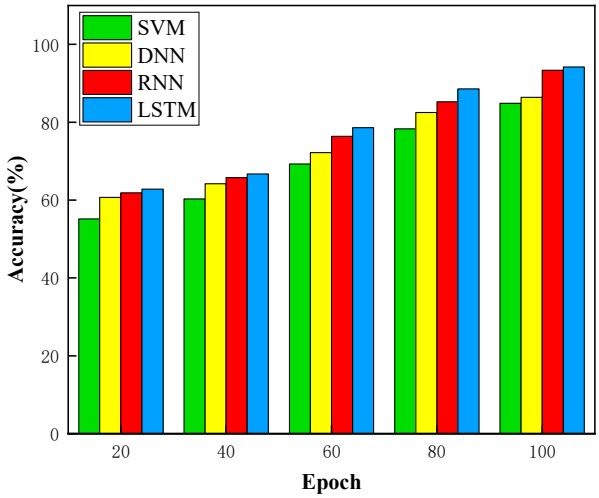


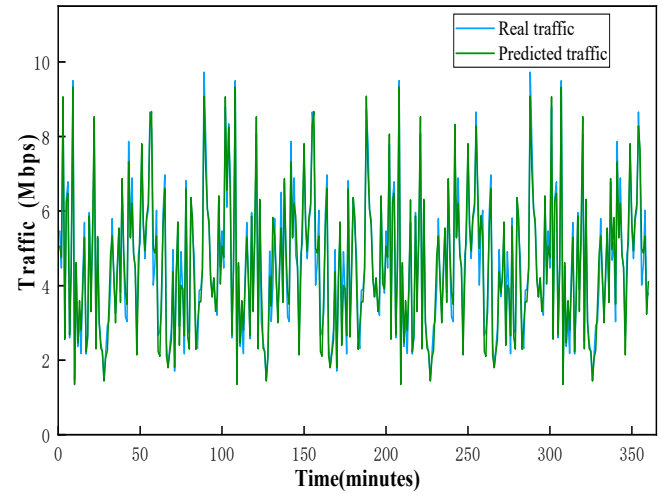
Figure 4. The comparison of prediction accuracy.

In order to better measure the accuracy of prediction results, we evaluated the performance of these algorithms in terms of prediction accuracy (MAE, MRE, and RMSE). In order to evaluate objectivity, we utilized the same dataset for training and testing during the prediction period.

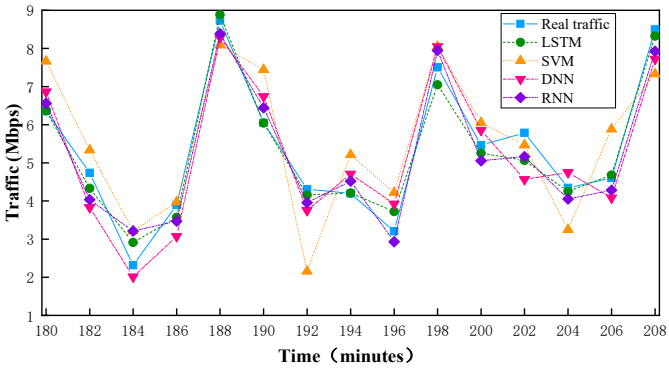
In Table 2, it can be learned that the prediction accuracy of LSTM exceeds 94%, with the lower MAE, MRE, and RMSE performing better than the other three algorithms. It should be noted that RNN has higher prediction accuracy compared with SVM and DNN. Due to being able to only process the dependency between short-term sequences, it is unsuitable for long-term prediction. On this basis, LSTM can learn long-term dependency between sequences, which improves prediction accuracy.

Table 2. Detail performance comparison of prediction algorithms.

Algorithm	Accuracy (%)	MAE	MRE (%)	RMSE
LSTM	94.2	0.21	3.24	0.27
SVM	84.9	0.46	6.83	0.59
DNN	86.4	0.35	4.35	0.42
RNN	93.4	0.26	3.96	0.31



(**a**)



(**b**)

Figure 5. The comparison between real traffic and predicted traffic. (**a**) Comparison between real traffic and predicted traffic by LSTM; (**b**) Comparison between real traffic and predicted traffic using different algorithms.

Additionally, we analyze the computational complexity of the proposed LSTM, which would compare with the other benchmark algorithms, as shown in Table 3.

Table 3. Order of magnitude comparison of computational complexity of algorithms.

Algorithm	Computational Complexity
LSTM	O[4(nm + n^2 + n)], n: hidden size, m: input size
SVM	O[Nsv^3], Nsv: number of support vectors
DNN	O[8nd^2], n: input size, d:vector dimension
RNN	O[nd^2], n: input size, d:vector dimension

By comparing the computational complexity of the algorithms by order of magnitude, SVM has an order of magnitude greater computational complexity than the other three algorithms. And the computational complexity of LSTM, DNN, and RNN is in the same order of magnitude. LSTM can overcome the problem of gradient explosion and has high prediction accuracy, resulting in better overall performance.

From the above results, SVM has a lower prediction accuracy when the data fluctuates greatly. DNN can lead to inaccurate predictions due to the loss of some of the feature values during the training process. RNN is prone to gradient explosions due to the chain rule processing. LSTM is an improved recursive neural network algorithm that overcomes the problem of inaccurate learning of past information due to gradient vanishing in RNN.

By incorporating memory function and dynamically adjusting the correlation weight coefficients between sequences, LSTM achieves high-precision prediction of long and short time series. In addition, in the literature [43], the authors proposed an advanced algorithm that combines LSTM with other algorithms, which also brings high-precision prediction results and becomes a key direction for our future research.

5.2. Simulation Setup and Results Analysis for Task Offloading Based on PSO-PG

In this section, we develop a task offloading scheme based on traffic prediction (TPO-TP) to avoid node overload and ensure the survivability of the network. In this section, we adopted the PSO-PG algorithm for task offloading.

In order to evaluate the performance of the proposed scheme for different scales of edge nodes, we set the scale of edge nodes in the simulated network topology to be in the range of 5–10. The fiber distance between edge nodes varies randomly between 20 and 100 Km. The modulation format is BPSK, and the link bandwidth is set to 6.25 Gbps per fiber with 100 FSs. For all MEC servers, the computing capacity is 3×10^{10} cycles/s. The initial workload per MEC server is randomly distributed in $[2,\ 5] \times 10^8$ cycles/s. The data size in task request is random in $[1, 2] \times 10^9$ bits. Simultaneously, the computing resource demand for a task request is randomly distributed at $[4, 30] \times 10^7$ cycles.

In addition, the simulation of the PSO-PG is implemented in Python 3.6 in Tensorflow. The particle size is 30, and the maximum number of iterations is 150. Initialized learning factor $c_1 = c_2 = 1.2$, initial value of inertia weights $\omega = 0.9$, the maximum training number *Trainmax* = 100, learning rate $\delta = 0.05$, and discount factor $\gamma = 0.95$. The PSO-PG is to be evolved in the direction of minimizing *Fitness* value. As shown in Equation (21), the *Fitness* goal of this paper is to achieve the minimum trade-off overhead by balancing the three indicators of average delay, resource utilization, and blocking probability. Therefore, we set α to 200, β to 5, and ε to 15.

To verify the efficiency of the proposed scheme, we compared the following schemes in Table 4, conducting a comprehensive evaluation of the following aspects.

Table 4. Description of the comparison schemes.

Schemes	Description of Scheme
PFS	Task offloading according to the principle of path priority without predicting before.
RFS	Task offloading according to the principle of resource priority without predicting before.
SVM-TPO	Task offloading based on SVM prediction results.
DNN-TPO	Task offloading based on DNN prediction results.
RNN-TPO	Task offloading based on RNN prediction results.

- Average delay

Average delay is a key metric for evaluating performance. We comprehensively evaluate the performance of the proposed scheme in terms of both edge node scale and task request scale. As shown in Figure 6a, we fixed the number of edge nodes to eight and clearly learned the changes in average delay under different task request scales. It can be intuitively observed that as the scale of task requests increases, the average delay of each scheme presents an upward trend. Due to the limited resources of edge nodes, more task requests mean a reduction in the available resources for each task. The average delay of PFS is largely greater than that of other schemes under large-scale task requests, owing to the fact that it only considers path distance and ignores the limited processing capacity of edge nodes. The limited processing capacity increases the waiting delay. RFS regards the priority of providing idle resources as primary for task offloading; thus, sufficient resources will generate a relatively lower average delay. Since path distance is not taken into account during offloading, it is intolerable for some latency-sensitive services. SVM-TPO, DNN-TPO, and RNN-TPO all consider future traffic scales, resulting in a certain reduction in

average delay compared with unpredictable offloading schemes. TPO-TP achieved the minimum average delay among all the schemes compared, since the proposed scheme will achieve optimized task offloading based on future traffic scales through accurate prediction. In dynamic network environments, the proposed scheme is more stable due to the fact that it considers the network resource utility in highly dynamic networks to ensure that both task offloading and network performance constraints are met.

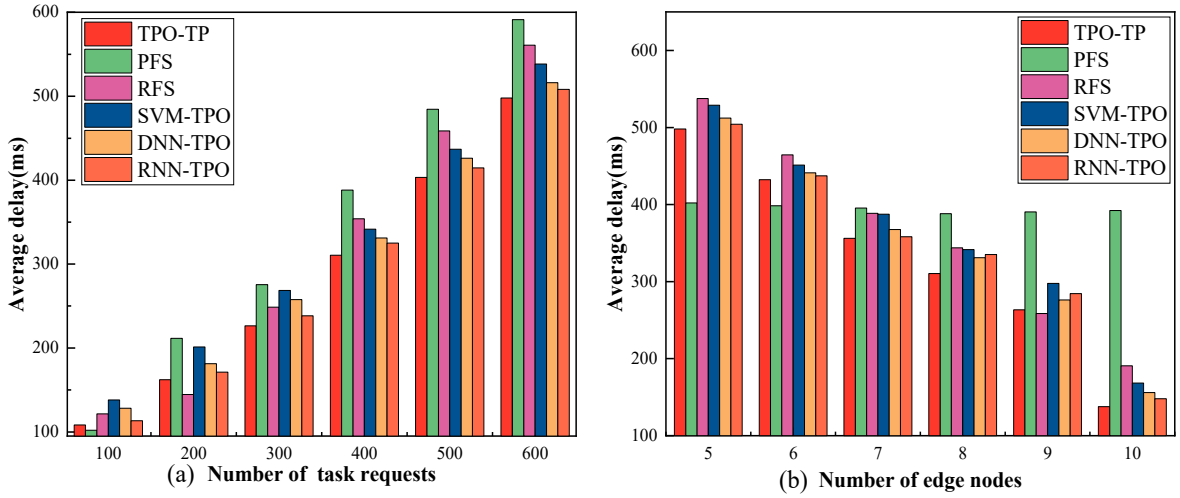


Figure 6. (**a**) Comparison of average delay under different task request scales; (**b**) Comparison of average delay under different edge nodes scales.

We further evaluated the changes in average delay under different edge node scales presented in Figure 6b, setting the number of task requests to 400. It can be clearly observed that for PFS only related to path length, the increase in available resources has no reduction in average delay. In the other schemes, there appears to be a downward trend with the expansion of edge nodes, resulting in an average delay due to the increase in available resources. In addition, to avoid failing to offload caused by the bust traffic increase sharply on the edge node, the schemes with a prediction mechanism are slightly better than RFS. The performance of TPO-TP is superior to other schemes, which can be attributed to the fact that the PSO-PG algorithm demonstrates the ability to adjust algorithm parameters according to network state in dynamic networks. It overcomes the challenge that, due to the changing network state, the competition for resources between services makes some of the service demands unfulfilled in the case of limited resources.

- Blocking probability

The blocking probability is a key indicator reflecting QoS, with excessive delay and insufficient resources being important reasons for blocking. Similarly, the performance of the proposed scheme is evaluated in terms of task request scale and edge node scale. From Figure 7a, it can be clearly seen that the blocking probability of all schemes increases as the scale of task requests increases, owing to the increased failure probability of offloading due to resource competition. Due to the intense service competition with limited computing resources, PFS has a high blocking probability. Due to ensuring sufficient computing resources, the blocking probability of other schemes remains low. Compared with RFS and PFS, these flexible offloading schemes based on global planning greatly reduce the blocking probability. With high-precision traffic prediction, TPO-TP allows tasks to be offloaded to the optimal MEC server for processing. It would never allocate excessive resources to ensure latency constraints, thereby increasing the throughput of the network and reducing the blocking probability. Compared with the flexible offloading scheme based on global planning, TPO-TP ultimately achieved a 39.6% reduction in blocking probability under the larger task request scale, which proves the effectiveness of our proposed scheme.

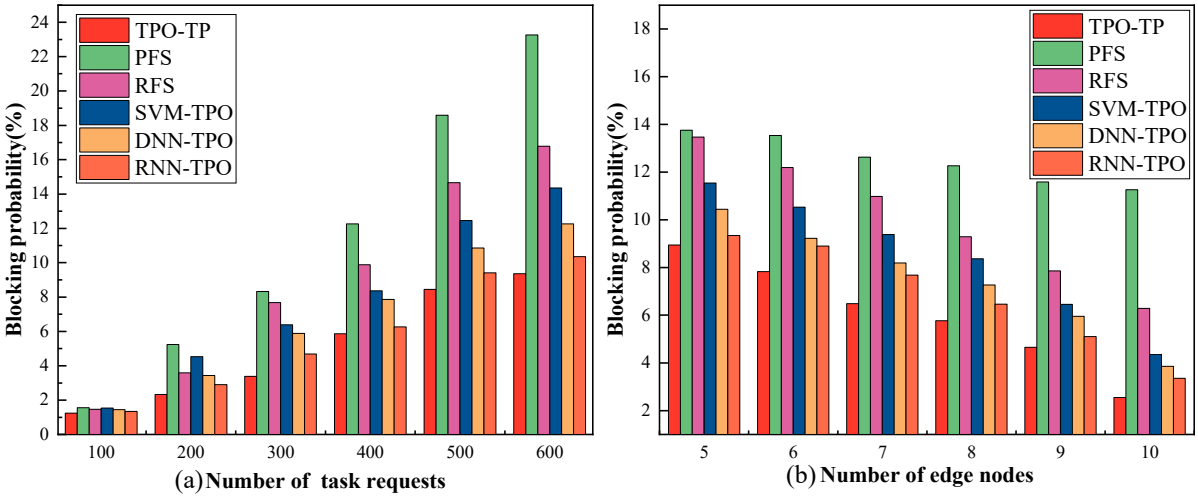


Figure 7. (**a**) Comparison of blocking probability under different task request scales; (**b**) Comparison of blocking probability under different edge nodes scales.

As shown in Figure 7b, the blocking probability in all schemes, except for PFS, presents a downward trend due to the increase in available resources. PFS exhibits a higher blocking probability owing to adopting the principle of path distance priority, where the increase in available resources will not reduce the blocking probability but rather increase the computing complexity as the expansion network scales. All other schemes significantly reduce the blocking probability as the expansion of edge nodes scales. However, these schemes with traffic prediction benefited from a comprehensive consideration of the future available resources from a global perspective, achieving lower blocking probabilities. The proposed TPO-TP has the lowest blocking probability among all schemes, which represents that the proposed scheme can be ensured to satisfy the delay constraints of the service in the dynamic network state.

- Resource utilization

We have further evaluated the proposed scheme from the perspective of resource utilization. Resource utilization is a critical evaluation criterion under the constraints of limited edge resources, reflecting the percentage of used resources in total resources. Figure 8a indicates the trend of resource utilization under different task request scales. The simulation results indicate that after accurate traffic prediction, resource utilization has increased by about 12.3%, highly compliant with the survival guarantee of fast offloading in a highly dynamic network environment. The scheme ensures meeting service requirements while maintaining high network resource utilization, which similarly proves the rationality of the proposed scheme. In Figure 8b, TPO-TP performs better than other schemes in the more complex network topologies, ensuring full utilization of established links in the network. These results further reveal the superiority of the proposed scheme and verify the effectiveness and applicability of the proposed prediction-assisted offloading scheme.

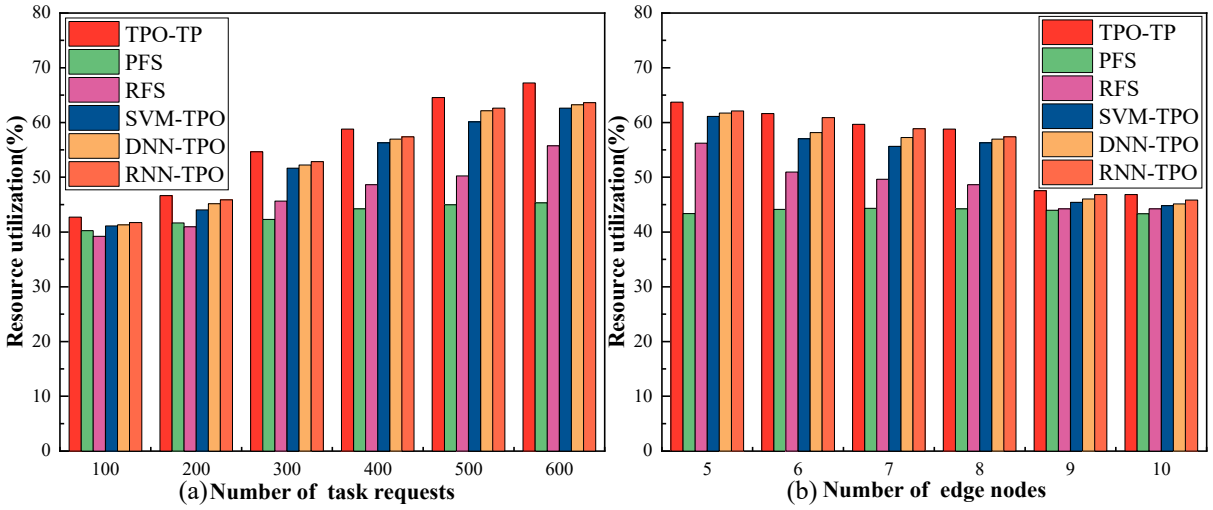


Figure 8. (**a**) Comparison of resource utilization under different task request scales, (**b**) Comparison of resource utilization under different edge nodes scales.

6. Conclusions and Future Work

In this paper, in order to ensure the low-delay requirements of intelligent services and the survivability of 6G edge networks, we propose a task offloading scheme based on traffic prediction for node-overload protection. In this scheme, we achieve future resource visualization of the entire 6G edge network by deploying an accurate traffic prediction model on the edge nodes. By constructing a mapping between traffic prediction and future available resources, our work has achieved a network survivability guarantee while maximizing the benefits of edge resources. Furthermore, we develop a task offloading scheme based on the PSO-PG algorithm under dynamic networks that realizes timely updating of the algorithm parameters according to the real-time network state. In the offloading process, the scheme effectively realizes the joint optimization of tuning offloading decisions with routing and computing resources to match network survivability guarantees. Simulation results indicate that the proposed scheme has reached the minimum trade-off overhead in terms of average delay, blocking probability, and resource utilization, validating the effectiveness of our work.

A network survivability guarantee would be a promising direction to support 6G development. Future research work will focus on using deep reinforcement learning to solve task offloading problems, which will contribute to solving the problem of algorithm dimension explosion. The surge in algorithm dimensions will no longer be an obstacle to updating mathematical models, further improving the details of resource allocation for survivability assurance. Ultimately, our goal is to develop an intelligent survivability guarantee scheme under the constraints of low latency. Therefore, the key of future work is to develop a task offloading strategy based on deep reinforcement learning under precise traffic prediction, achieving fine-grained allocation of multidimensional resources, and overcoming the shortcomings of traditional algorithms in poor performance with fixed allocation caused by dimension explosion.

Author Contributions: Conceptualization, Z.S. and H.Y.; methodology, Z.S., H.Y. and C.L.; software, Z.S. and Q.Y.; validation, Z.S., A.Y., J.Z. and Y.Z.; formal analysis, Z.S., S.L. and Y.L.; investigation, Z.S. and H.Y.; writing—original draft preparation, Z.S.; writing—review and editing, Z.S. and H.Y. All authors have read and agreed to the published version of the manuscript.

Funding: This work has been supported in part by the NSFC project (62122015, 62201088, 62271075) and is supported by the joint project of the China Mobile Research Institute and the Fund of SKL of IPOC (BUPT) (IPOC2021ZT04) and supported by "the Fundamental Research Funds for the Central Universities".

Data Availability Statement: Not applicable.

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

References

1. Petrov, V.; Lema, M.A.; Gapeyenko, M.; Antonakoglou, K.; Moltchanov, D.; Sardis, F.; Samuylov, A.; Andreev, S.; Koucheryavy, Y.; Dohler, M. Achieving End-to-End Reliability of Mission-Critical Traffic in Softwarized 5G Networks. *IEEE J. Sel. Areas Commun.* **2018**, *36*, 485–501. [CrossRef]
2. Spantideas, S.; Giannopoulos, A.; Cambeiro, M.A.; Trullols-Cruces, O.; Atxutegi, E.; Trakadas, P. Intelligent Mission Critical Services over Beyond 5G Networks: Control Loop and Proactive Overload Detection. In Proceedings of the 2023 International Conference on Smart Applications, Communications and Networking (SmartNets), Istanbul, Turkiye, 25–27 July 2023; pp. 1–6. [CrossRef]
3. Skarin, P.; Tärneberg, W.; Årzen, K.-E.; Kihl, M. Towards Mission-Critical Control at the Edge and Over 5G. In Proceedings of the 2018 IEEE International Conference on Edge Computing (EDGE), San Francisco, CA, USA, 2–7 July 2018; pp. 50–57. [CrossRef]
4. Sun, Z.; Yang, H.; Li, C.; Yao, Q.; Wang, D.; Zhang, J.; Vasilakos, A.V. Cloud-Edge Collaboration in Industrial Internet of Things: A Joint Offloading Scheme Based on Resource Prediction. *IEEE Internet Things J.* **2022**, *9*, 17014–17025. [CrossRef]
5. Wu, D.; Han, X.; Yang, Z.; Wang, R. Exploiting Transfer Learning for Emotion Recognition Under Cloud-Edge-Client Collaborations. *IEEE J. Sel. Areas Commun.* **2021**, *39*, 479–490. [CrossRef]
6. Lin, Z.; Lin, M.; Champagne, B.; Zhu, W.-P.; Al-Dhahir, N. Secrecy-Energy Efficient Hybrid Beamforming for Satellite-Terrestrial Integrated Networks. *IEEE Trans. Commun.* **2021**, *69*, 6345–6360. [CrossRef]
7. Yang, H.; Yao, Q.; Bao, B.; Yu, A.; Zhang, J.; Vasilakos, A.V. Multi-associated parameters aggregation-based routing and resources allocation in multi-core elastic optical networks. *IEEE/ACM Trans. Netw.* **2022**, *30*, 2145–2157. [CrossRef]
8. Yao, Q.; Yang, H.; Li, C.; Bao, B.; Zhang, J.; Cheriet, M. Federated Transfer Learning Framework for Heterogeneous Edge IoT Networks. *China Commun.* [CrossRef]
9. Lin, Z.; Niu, H.; An, K.; Hu, Y.; Li, D.; Wang, J.; Al-Dhahir, N. Pain Without Gain: Destructive Beamforming From a Malicious RIS Perspective in IoT Networks. *IEEE Internet Things J.* **2023**. [CrossRef]
10. Yu, T.; Yang, H.; Yao, Q.; Yu, A.; Zhao, Y.; Liu, S.; Li, Y.; Zhang, J.; Cheriet, M. Multi visual GRU based survivable computing power scheduling in metro optical networks. *IEEE Trans. Netw. Serv. Manag.* **2023**. [CrossRef]
11. Wu, D.; Bao, R.; Li, Z.; Wang, H.; Zhang, H.; Wang, R. Edge-Cloud Collaboration Enabled Video Service Enhancement: A Hybrid Human-Artificial Intelligence Scheme. *IEEE Trans. Multimed.* **2021**, *23*, 2208–2221. [CrossRef]
12. An, K.; Lin, M.; Ouyang, J.; Zhu, W.-P. Secure Transmission in Cognitive Satellite Terrestrial Networks. *IEEE J. Sel. Areas Commun.* **2016**, *34*, 3025–3037. [CrossRef]
13. Ma, R.; Yang, W.; Shi, H.; Lu, X.; Liu, J. Covert communication with a spectrum sharing relay in the finite blocklength regime. *China Commun.* **2023**, *20*, 195–211. [CrossRef]
14. Guo, C.; He, W.; Li, G.Y. Optimal Fairness-Aware Resource Supply and Demand Management for Mobile Edge Computing. *IEEE Wirel. Commun. Lett.* **2021**, *10*, 678–682. [CrossRef]
15. Gunawardena, J. Learning Outside the Brain: Integrating Cognitive Science and Systems Biology. *Proc. IEEE* **2022**, *110*, 590–612. [CrossRef]
16. Shi, C.; Ding, L.; Wang, F.; Salous, S.; Zhou, J. Joint Target Assignment and Resource Optimization Framework for Multitarget Tracking in Phased Array Radar Network. *IEEE Syst. J.* **2021**, *15*, 4379–4390. [CrossRef]
17. Liu, J.; Zhang, Q. Offloading schemes in mobile edge computing for ultra-reliable low latency communications. *IEEE Access* **2018**, *6*, 12825–12837. [CrossRef]
18. Chen, L.; Xu, J.; Zhou, S. Computation peer offloading in mobile edge computing with energy budgets. In Proceedings of the GLOBECOM 2017—2017 IEEE Global Communications Conference, Singapore, 4–8 December 2017; pp. 1–6.
19. Sun, X.; Ansari, N. Latency aware workload offloading in the cloudlet network. *IEEE Commun. Lett.* **2017**, *21*, 1481–1484. [CrossRef]
20. Li, J.; Luo, G.; Cheng, N.; Yuan, Q.; Wu, Z.; Gao, S.; Liu, Z. An end-to-end load balancer based on deep learning for vehicular network traffic control. *IEEE Int. Things J.* **2019**, *6*, 953–966. [CrossRef]
21. Taleb, T.; Ksentini, A.; Frangoudis, P. Follow-me cloud: When cloud services follow mobile users. *IEEE Trans. Mobile Comput.* **2019**, *7*, 369–382. [CrossRef]
22. Wang, S.; Urgaonkar, R.; Zafer, M.; He, T.; Chan, K.; Leung, K.K. Dynamic Service Migration in Mobile Edge Computing Based on Markov Decision Process. *IEEE/ACM Trans. Netw.* **2019**, *27*, 1272–1288. [CrossRef]
23. He, Y.; Zhao, N.; Yin, H. Integrated networking, caching, and computing for connected vehicles: A deep reinforcement learning approach. *IEEE Trans. Veh. Technol.* **2018**, *67*, 44–55. [CrossRef]
24. Chen, Z.; Wang, X. Decentralized computation offloading for multi-user mobile edge computing: A deep reinforcement learning approach. *arXiv* **2018**, arXiv:1812.07394. [CrossRef]
25. Jitani, A.; Mahajan, A.; Zhu, Z.; Abou-Zeid, H.; Fapi, E.T.; Purmehdi, H. Structure-Aware Reinforcement Learning for Node-Overload Protection in Mobile Edge Computing. *IEEE Trans. Cogn. Commun. Netw.* **2022**, *8*, 1881–1897. [CrossRef]

26. Fang, Z.; Xu, X.; Dai, F.; Qi, L.; Zhang, X.; Dou, W. Computation Offloading and Content Caching with Traffic Flow Prediction for Internet of Vehicles in Edge Computing. In Proceedings of the 2020 IEEE International Conference on Web Services (ICWS), Beijing, China, 19–23 October 2020; pp. 380–388. [CrossRef]
27. Tian, H.; Xu, X.; Qi, L.; Zhang, X.; Dou, W.; Yu, S.; Ni, Q. CoPace: Edge Computation Offloading and Caching for Self-Driving With Deep Reinforcement Learning. *IEEE Trans. Veh. Technol.* **2021**, *70*, 13281–13293. [CrossRef]
28. Xu, X.; Jiang, Q.; Zhang, P.; Cao, X.; Khosravi, M.R.; Alex, L.T.; Qi, L.; Dou, W. Game Theory for Distributed IoV Task Offloading With Fuzzy Neural Network in Edge Computing. *IEEE Trans. Fuzzy Syst.* **2022**, *30*, 4593–4604. [CrossRef]
29. Ferdosian, N.; Moazzeni, S.; Jaisudthi, P.; Ren, Y.; Agrawal, H.; Simeonidou, D.; Nejabat, R. Autonomous Intelligent VNF Profiling for Future Intelligent Network Orchestration. *IEEE Trans. Mach. Learn. Commun. Netw.* **2023**, *1*, 138–152. [CrossRef]
30. Nagib, A.M.; Abou-zeid, H.; Hassanein, H.S. Toward Safe and Accelerated Deep Reinforcement Learning for Next-Generation Wireless Networks. *IEEE Netw.* **2023**, *37*, 182–189. [CrossRef]
31. Prasanna Kumar, G.; Shankaraiah, N. An Efficient IoT-based Ubiquitous Networking Service for Smart Cities Using Machine Learning Based Regression Algorithm. *Int. J. Inf. Technol. Comput. Sci. (IJITCS)* **2023**, *15*, 15–25. [CrossRef]
32. Yang, H.; Zhao, X.; Yao, Q.; Yu, A.; Zhang, J.; Ji, Y. Accurate fault location using deep neural evolution network in cloud data center interconnection. *IEEE Trans. Cloud Comput.* **2022**, *10*, 1402–1412. [CrossRef]
33. Li, C.; Yang, H.; Sun, Z.; Yao, Q.; Bao, B.; Zhang, J.; Vasilakos, A.V. Federated hierarchical trust-based interaction scheme for cross-domain industrial IoT. *IEEE Internet Things J.* **2023**, *10*, 447–457. [CrossRef]
34. Yang, Z.; Yao, Y.; Gao, H.; Wang, J.; Mi, N.; Sheng, B. New YARN Non-Exclusive Resource Management Scheme through Opportunistic Idle Resource Assignment. *IEEE Trans. Cloud Comput.* **2021**, *9*, 696–709. [CrossRef]
35. Eramo, V.; Lavacca, F.G.; Catena, T.; Perez Salazar, J.P. Application of a Long Short Term Memory neural predictor with asymmetric loss function for the resource allocation in NFV network architectures. *Comput. Netw.* **2021**, *193*, 108104–108116. [CrossRef]
36. Feriani, A.; Hossain, E. Single and Multi-Agent Deep Reinforcement Learning for AI-Enabled Wireless Networks: A Tutorial. *IEEE Commun. Surv. Tutor.* **2021**, *23*, 1226–1252. [CrossRef]
37. Giannopoulos, A.; Spantideas, S.; Tsinos, C.; Trakadas, P. Power Control in 5G Heterogeneous Cells Considering User Demands Using Deep Reinforcement Learning. In Proceedings of the IFIP International Conference on Artificial Intelligence Applications and Innovations, Crete, Greece, 25–27 June 2021; Springer: Cham, Switzerland, 2021; Volume 628. [CrossRef]
38. Liu, C.H.; Chen, Z.; Tang, J.; Xu, J.; Piao, C. Energy-efficient UAV control for effective and fair communication coverage: A deep reinforcement learning approach. *IEEE J. Sel. Areas Commun.* **2018**, *36*, 2059–2070. [CrossRef]
39. Mur, D.C.; Gavras, A.; Ghoraishi, M.; Hrasnica, H.; Kaloxylos, A. (Eds.) *AI and ML–Enablers for beyond 5G Networks*; White Paper; 5G PPP Technology Board: Kista, Sweden, 2021.
40. Yang, H.; Yao, Q.; Yu, A.; Lee, Y.; Zhang, J. Resource assignment based on dynamic fuzzy clustering in elastic optical networks with multi-core fibers. *IEEE Trans. Commun.* **2019**, *67*, 3457–3469. [CrossRef]
41. Yu, A.; Yang, H.; Feng, C.; Li, Y.; Zhao, Y.; Cheriet, M.; Vasilakos, A.V. Socially-aware traffic scheduling for edge-assisted metaverse by deep reinforcement learning. *IEEE Netw.* **2023**. [CrossRef]
42. Li, C.; Yang, H.; Sun, Z.; Yao, Q.; Zhang, J.; Yu, A.; Vasilakos, A.V.; Liu, S.; Li, Y. High-Precision Cluster Federated Learning for Smart Home: An Edge-Cloud Collaboration Approach. *IEEE Access* **2023**. [CrossRef]
43. Eramo, V.; Catena, T. Application of an Innovative Convolutional/LSTM Neural Network for Computing Resource Allocation in NFV Network Architectures. *IEEE Trans. Netw. Manag.* **2022**, *19*, 2929–2943.

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.