# Joint Blockchain and Federated Learning-based Offloading in Harsh Edge Computing Environments

Guanjin Qu, Huaming Wu*
Center for Applied Mathematics
Tianjin University
Tianjin 300072, China
{guanjinqu,whming}@tju.edu.cn

Naichuan Cui
Sch. of Electrical and Electronic Engineering
Nanyang Technological University
Singapore 639798, Singapore
CUIN0001@e.ntu.edu.sg

## ABSTRACT

With the popularity of edge computing, numerous Internet of Things (IoT) applications have been developed and applied to various fields. However, for the harsh environment with network fluctuations and potential attacks, traditional task offloading decision-making schemes cannot meet the requirements of real-time and security. For this reason, we propose a novel task offloading decision framework to cope with the special requirements of the environment. This framework uses a task offloading decision model based on deep reinforcement learning algorithms, and is located on the user side to reduce the impact of network fluctuations. To improve the efficiency and security of the model in harsh edge computing environments, we adopt federated learning and introduce the blockchain into the process of parameter upload and decentralization of federated learning. In addition, we design a new blockchain consensus algorithm to reduce the waste of computing resources and improve the embedding and propagation speeds of the blockchain. Furthermore, we demonstrate the effect of task offloading of this model by performing offloading decisions on a simulation platform.

## CCS CONCEPTS

• **Computing methodologies** → *Distributed computing methodologies*; *Simulation evaluation*; • **Networks** → Network reliability; • **Security and privacy** → *Network security*.

## KEYWORDS

harsh environments, edge computing, federated learning, blockchain

**ACM Reference Format:**
Guanjin Qu, Huaming Wu and Naichuan Cui. 2021. Joint Blockchain and Federated Learning-based Offloading in Harsh Edge Computing Environments. In *SIGMOD '21: ACM International Conference on Management of Data, June 20–25, 2021, Xi'an, Shaanxi, China.* ACM, New York, NY, USA, 6 pages.

*Corresponding Author.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.
*SIGMOD '21, June 20–25, 2021, Xi'an, Shannxi, China*
© 2021 Association for Computing Machinery.
ACM ISBN 978-1-4503-XXXX-X/18/06…$15.00

## 1 INTRODUCTION

Recently, the popularity of edge computing has changed the traditional information communication structure by connecting devices at the network edge to Internet of Things (IoT) and cloud centers. The computing tasks of IoT devices can be offloaded to the edge or cloud for processing, which can break through the resource limitations of mobile devices, e.g., reduce computing load, improve task processing efficiency, and save energy consumption [13, 15]. However, it is still difficult to popularize edge computing in harsh environments, e.g., suburban and rural areas, satellite-terrestrial networks, and low orbit satellite edge computing [2], which own the following characteristics:

- In a stable environment, there are many edge nodes and abundant communication resources. However, the edge service nodes in the harsh environment are scarce and communication resources are scarce, and there is a possibility that the node may suddenly disconnect.
- The stable environment has little interference, and the harsh environment has untrusted nodes, and even the possibility of malicious attacks.

Therefore, the harsh environment requires more attention to reliability and security. How to make the edge offloading system have a certain ability to resist harsh environments is becoming the key to the development of edge computing. The task offloading decision algorithm is the most critical part of the entire offloading model. Currently, task offloading decision-making schemes are mainly divided into traditional algorithms and intelligent algorithms. Traditional computing offloading techniques often use some heuristic algorithms [6, 8, 12]. Complex algorithms are usually difficult to solve and require a huge amount of computation. Methods based on deep learning, e.g., deep reinforcement learning, can promote offloading decision-making, dynamic resource allocation, and content caching, which are conducive to coping with the growth of communication and computing in emerging IoT applications. Recently, intelligent algorithms have gradually emerged. By introducing neural networks and other methods, offloading decision-making can achieve good results, but there are still many challenges, e.g., slow learning speed and long training time. At present, common intelligent offloading algorithms are generally deployed on edge servers because they require sufficient computing capacity. For this reason, existing research considers the introduction of distributed learning into the task offloading intelligent algorithm [4], but in a harsh environment, how to ensure that the parameters are kept safe during the transfer process is worth studying.

This paper mainly focuses on task offloading decision-making against harsh edge computing environments. This kind of environment has communication fluctuation and security problems, especially client poisoning, which is the most common defect in distributed systems [10]. The main contributions of this paper can be summarized as follows:

- To effectively take offloading decisions in harsh environments, we propose a novel task offloading framework based on Deep Reinforcement Learning (DRL), which is placed on the user side to resist network fluctuations caused by communication.
- To improve the efficiency and security of the model in harsh environments, the blockchain format is adopted to transmit parameter information in federated learning. As far as we know, this is the first to combine federated learning with blockchain is integrated into the field of edge computing.
- We propose a new blockchain consensus algorithm to improve the waste of computing power and slow iteration in the traditional consensus algorithm. The improved blockchain consensus algorithm can improve the efficiency and safety of the entire model.

## 2 SYSTEM MODEL AND OFFLOADING ALGORITHM

This section provides a detailed task offloading framework in harsh environments and the specific algorithms used in the framework.

### 2.1 System Model

Fig. 1 shows a schematic diagram of edge computing in a harsh environment, where the user side can communicate through an edge server (low-orbit satellite). As the communication between the user and the edge server may be interrupted, how to overcome the impact caused by communication is the focus of edge computing in harsh environments.

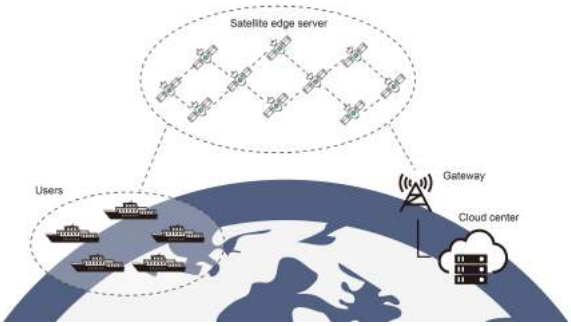



**Figure 1: A typical harsh environment: maritime satellite communications**

As depicted in Fig. 2, we design a brand-new distributed task offloading framework, which can meet the efficient and stable operation of task offloading decision-making algorithms, and has the ability to withstand harsh environments and to cope with the complex topology of multiple edge servers and multiple cloud servers. The overall process of the proposed framework is as follows: After the user generates a task, the task information is collected by the local device for offloading decision-making. At the same time, because the offloading decision algorithm uses a neural network, in order to improve the training speed of the neural network on the user side, we introduce federated learning to update the parameters. The federated learning improves the training speed of the neural network in the task offloading decision-making algorithm by collecting the parameters of the neural network in different client terminals and aggregate them to the client terminal to update the parameters. Fig. 2 shows a typical federated learning algorithm flow. In addition, in order to ensure that the parameters will not be tampered with during the parameter update process, we adopt a blockchain-based data structure and create a new consensus algorithm to ensure the safety and efficiency of the parameter transfer process.

### 2.2 Offloading Algorithm

This model is mainly divided into a task offloading decision algorithm based on deep reinforcement learning, and a federated learning and blockchain mechanism based on an improved task offloading decision algorithm. The following two parts of the algorithm will be introduced in detail.

*2.2.1 Task Offloading Decision Algorithm.* We use a deep reinforcement learning algorithm as the task offloading decision model. In order to improve the model's ability to resist network fluctuations, we place the model's offloading decision-making algorithm on the user side, which can prevent the unloading decision server from being unable to connect to the offloading decision server due to sudden network disconnection. The specific parameter settings are as follows:

- *State S*: The state in this model includes task information, communication information with other nodes, computing capability information of different devices, and pre-unloading action information. When the user equipment generates a task to be uninstalled, it will perceive other nodes communicating with it, and obtain the communication and calculation information of the node. After that, it will be standardized as a set of state input task offloading models, and after the actions are executed, the information of the tasks in the original state and the previous offloading action information will be updated
- *Action a*: Actions include local execution, offloading to a certain edge server, and offloading to a certain cloud server. It should be noted that since the nodes sensed each time might be different, the same action value in different states may represent different servers. This is also to resist the impact of communication fluctuations in the harsh environment.
- *Reward R*: Reward is a negative linear correlation of the total cost of the task. The total cost is the weighted sum of the delay and energy consumption of the task. The former includes calculation delay and transmission delay, while the latter focuses on the energy consumption of the user side.

In order to improve the model's ability to resist network fluctuations, we place the model's offloading decision algorithm on the user side, which can prevent the situation where the offloading

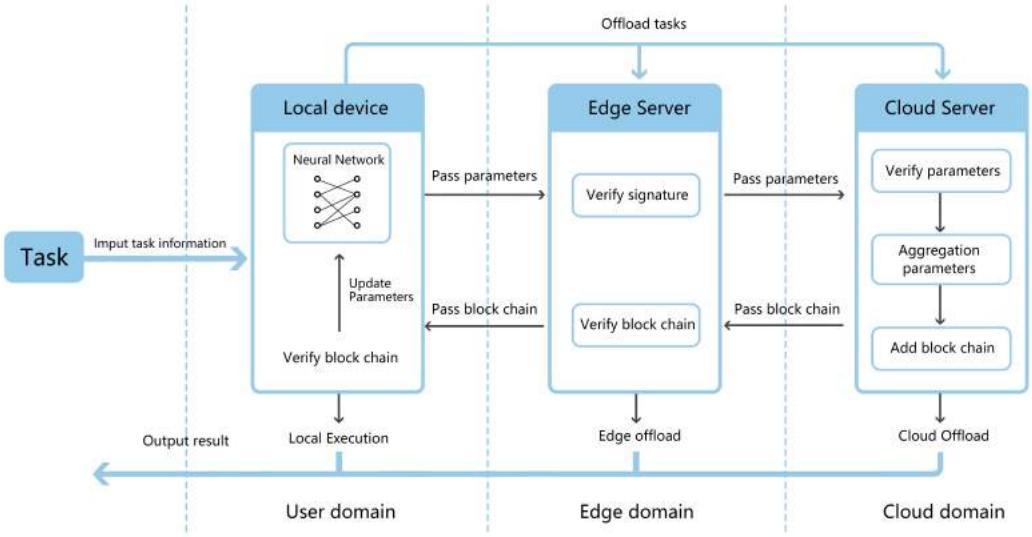



**Figure 2: The proposed task offloading-decision framework against harsh environments**

decision server cannot be connected due to sudden network disconnection. Here, we can directly use current mature DRL algorithms, e.g., Deep Q-Learning (DQN) [9] and Deep Deterministic Policy Gradient (DDPG) [3].

*2.2.2 Task Offloading Decision Algorithm.* Since deep reinforcement learning requires the use of neural networks, which leads to training requires a lot of calculations and puts a lot of pressure on the user side, we introduce a distributed training method named federated learning [5]. However, it will inevitably encounter two challenges in a harsh environment as follows:

- How to prevent the updated parameters from being tampered with during the decentralization process?
- How to identify the junk parameters uploaded by untrusted users maliciously?

To this end, we have improved the blockchain and introduced it into federated learning, so that the parameters of users in federated learning are uploaded in the form of transactions, and the aggregated parameters of federated learning are decentralized in the format of blockchain to ensure that the parameters are transferred. It will not be modified during the process. The federal learning algorithm combined with blockchain can effectively prevent security attacks in harsh environments, such as uploading wrong parameters to control the global model by poisoning clients [11].

The traditional blockchain uses a Proof-of-Work algorithm for consensus authentication [7]. However, in the edge computing environment with resource-constrained, the use of the Proof-of-Work algorithm will result in a lot of waste of computing power, and there is still the possibility of brute force cracking attacks [14]. For this reason, we adopt a new consensus mechanism to adapt to parameter transmission in harsh environments. We introduce the average reward value in deep reinforcement learning into the consensus algorithm. Since the average reward value can reflect the offloading effect of the model, the effect of the parameters can be verified by verifying the average reward value.

Fig. 3 shows the structure and dissemination process of the improved blockchain. The following will introduce the role of blockchain in federated learning according to the blockchain process:

- **Information chain upload:** After a certain number of training sessions are completed on the user side, the neural network parameters will be transmitted in the form of an information chain. The format of the information chain is shown in the figure, which is similar to the format of the transaction chain in the traditional blockchain. The information chain contains public key hashes, timestamps and digital signatures to facilitate identity verification. In addition, it also needs to include the number of network iterations, parameter information, and average reward value. After the information chain is generated, it will be uploaded to the edge server. In order to reduce communication consumption, it is only necessary to upload it once to the edge server with the highest communication intensity.
- **Information chain transmission:** The edge server will verify the signature after receiving the information chain. If the verification succeeds, it will upload it to the cloud server. If it fails, the information chain will be discarded. When network fluctuations occur and the cloud server cannot be connected, it can be passed to other edge servers. Because federated learning is not very robust. Even if some parameters are not uploaded to the cloud server, the final parameter aggregation will not be affected.
- **Block embedding:** The cloud server collects parameter information and verifies the signature within a period of time. Then, the cloud server will calculate whether the average reward value of each parameter matches the average reward value on the information chain (a certain deviation is allowed) by simulating the offloading task. The cloud server that first completes all parameter calculation tasks will gain leadership and be responsible for embedding the blockchain.

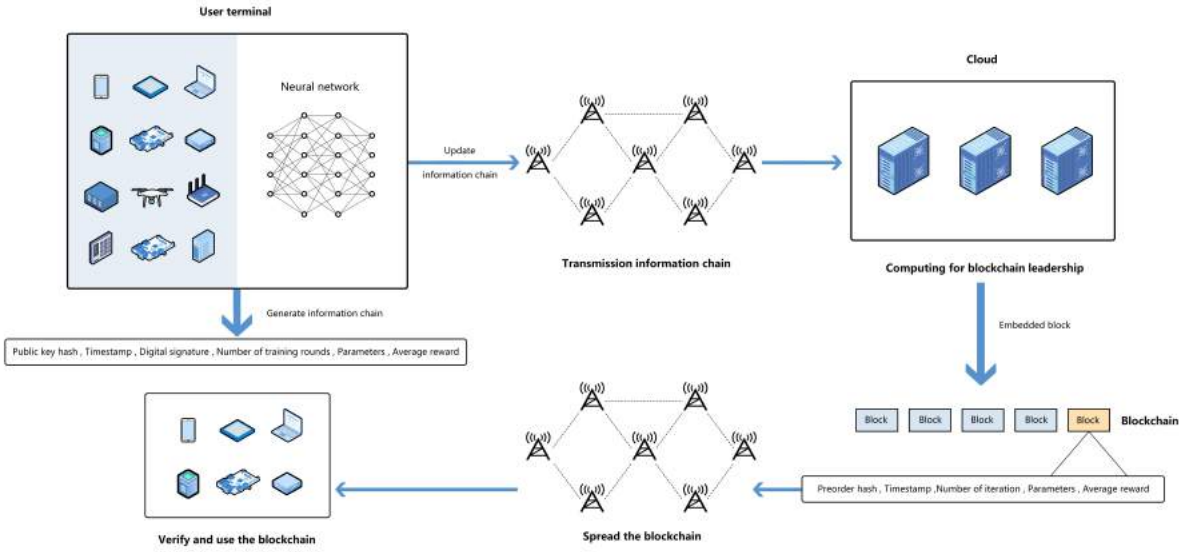



**Figure 3: The blockchain structure and communication process**

The leader server will aggregate the parameters and embed the updated parameters into the blockchain, and then broadcast the entire network.

- **Blockchain spread:** When the cloud server receives the blockchain, it will verify whether the aggregated parameters meet the marked average reward value. If it does, it will stop its own parameter calculation process and continue to broadcast to other nodes. When the edge server receives the blockchain, it will also verify the parameters, and continue to broadcast to other nodes after the verification is passed. When the user terminal receives and passes the verification, the parameters of the neural network are updated to the parameters on the latest block.

During the entire verification process, the cloud server will eliminate the parameters that fail to pass the signature verification, low unit profit value, and the number of iterations to ensure that the collected parameters are true and effective, thereby preventing untrusted users from maliciously uploading junk parameters. In order to reduce the communication volume of the blockchain in the harsh environment, the original blockchain structure is improved as shown in Fig. 3. It still has the characteristics of unforgeability. We require that the number of iterations for each aggregation is greater than that of the original block. The transmission of the blockchain can ensure the training process of each federation learning be synchronized. Using calculation average reward value instead of traditional PoW verification can prevent malicious tampering of parameters by illegal users: when someone tries to construct a false parameter and incorporate it into the blockchain for broadcasting, the lower-end server will perform the parameter after receiving the blockchain. We can verify that if the average reward generated by the simulated uninstallation does not match the block, the false parameters will be detected. If they match, even if the parameters are costly, the fraud will not have a negative impact on the model, thus in the perspective of game theory Prevented parameter tampering.

The introduction of blockchain into federated learning will inevitably increase the extra cost of the system in terms of delay and energy consumption mainly caused by the mining of blockchain and consensus algorithm. For cloud servers with high computing capabilities, whether it is a mining algorithm or a consensus algorithm, we only test the neural network instead of training, so the time cost is very low and it is not sensitive to energy consumption. For the local device, because we only need to perform the average reward calculation once, it only takes 0.25 seconds to verify the authenticity of the blockchain using a 2.9 GHz CPU. Therefore, the time delay and energy consumption caused by using blockchain are acceptable, because to ensure the reliability of the model in harsh environments, we must increase the complexity of the model.

## 3 EXPERIMENTAL VERIFICATION AND ANALYSIS

In this section, we will implement the proposed offloading framework on a simulation platform called Fedchain[1], and evaluate the effectiveness and reliability of the model.

### 3.1 Experimental Setup

The equipment and tools used in the experiment are summarized in Table 1, and the entire structure of the experimental platform is shown in Fig. 4.

The edge-cloud collaborative computing environment usually involves IoT devices, edge nodes and cloud nodes. Here, we set this environment to have one cloud server, one edge server, one communication center and several IoT devices. We run the simulation platform on devices in the edge-cloud collaborative computing environment to implement communication and other functions. Among them, the simulation experiment is performed based on the HTTP protocol, and Flask is used as the HTTP service framework.

[1]More information about the proposed Fedchain will be given in our other papers.

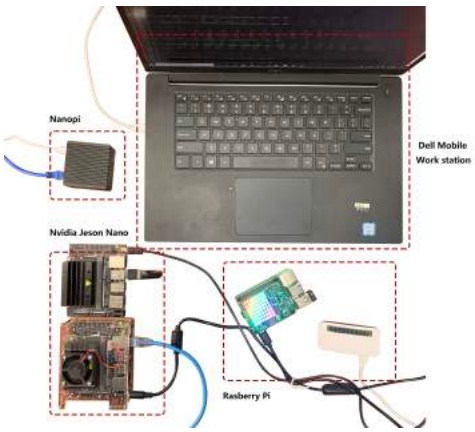



**Figure 4: Equipment prototype with Fedchain**

We simulate harsh environments on Fedchain. For the security risks in a harsh environment, we use certain devices as poisoning clients to upload wrong neural network parameters, in order to destroy the global model. For the simulation of communication fluctuations, every time we make an offloading decision, the computing power and bandwidth of the offloading server perceived by our device change in real time and satisfy the Poisson distribution. After the offloading decision is generated, the model will evaluate the effect according to the perceived environmental information. So although our experimental equipment is connected to the LAN through a router, it seems to be a stable communication environment with high bandwidth. However, through the simulation platform and parameter settings in the model, we can simulate the possible communication fluctuations when tasks are offloaded.

**Table 1: Hardware tools for experimental verification.**

| Tools | Description |
|---|---|
| Router | Responsible for building the network |
| Dell Workstation | Simulate cloud server |
| NVIDIA Jetson nano | Simulate edge server |
| Raspberry pi | Simulate IoT devices |
| Nanopi | Communication center |

## 3.2 Experimental Analysis

From Fig. 5, the loss function curve of the device after using federated learning is not as stable as the control device without federated learning, but it is still within the convergence range. This is because our reinforcement learning uses a frozen network to reduce the correlation between states, this will reduce the stability of the loss curve during the federated learning process. Since the definition of the loss function is different from that of the conventional deep learning network, the loss function curve of the neural network cannot represent the convergence of the DRL algorithm [1], so we use the average reward of reinforcement learning to represent the effect of the model.

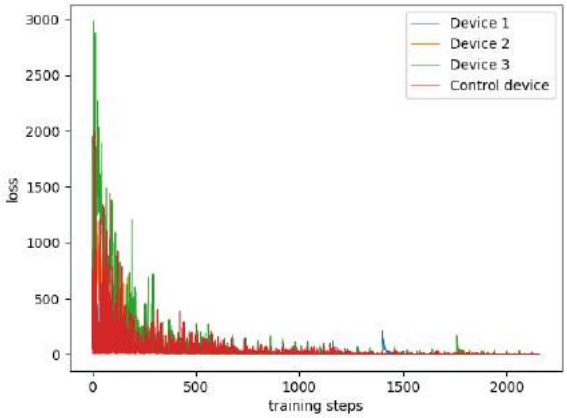
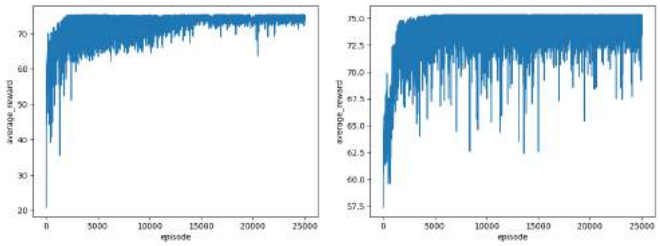



**Figure 5: The convergence performance under different devices**




(a) Three devices with federated learning (b) A control device without federated learning

**Figure 6: The average reward value of the task offloading model under different devices**

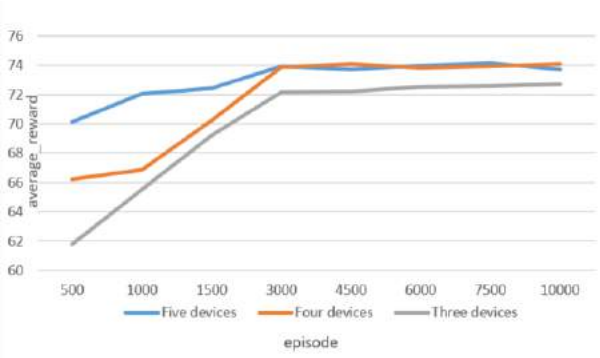



**Figure 7: The effect of using different numbers of devices for federated learning**

From Fig. 6, as the number of training rounds arises, the average reward values of these four devices increase, and gradually approach the highest value, especially the ones with federated learning. This reflects that with the training of the model, its decision-making approaches the optimal solution, especially the one with federated learning. In addition, it can be clearly seen that the equipment using federated learning approaches the optimal solution

faster than the unused equipment. Thus, the use of federated learning does improve the training speed of the model and the optimal solution can be obtained faster.

Fig. 7 demonstrates that when the number of devices participating in federated learning is too small, it will affect the decision-making effect of the model. Besides, as the number of participating machines increases, the convergence speed of the model will also become faster.

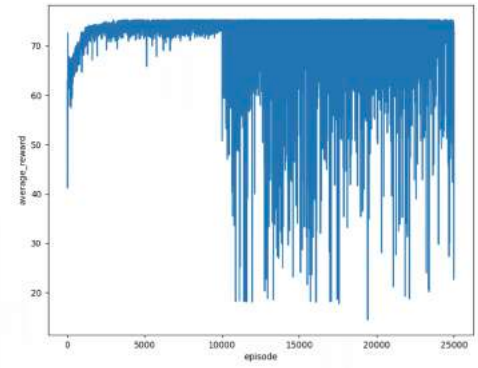
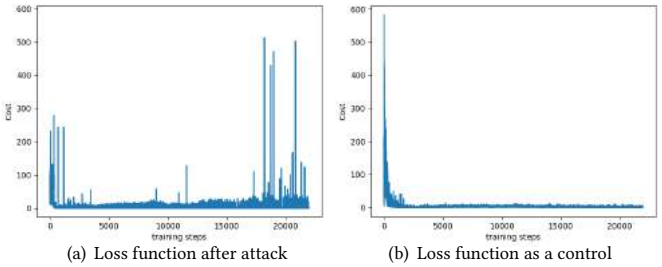



**Figure 8: The effect of using different numbers of devices for federated learning**

To verify the effect of blockchain, we let four devices perform federated learning, and one of them will upload wrong parameters. We start with the consensus algorithm of the blockchain and close it in the $1000^{th}$ round of training. It can be seen from Fig. 8 that the blockchain model can effectively eliminate the wrong parameters at the beginning, and closing it will cause the model to become worse. Thus, the federated learning algorithm introduced into the blockchain has higher security and efficiency, and can prevent potential attacks from illegal users in harsh environments.




(a) Loss function after attack (b) Loss function as a control

**Figure 9: Convergence performance in the attack and normal states**

Fig. 9 shows the convergence of the neural network loss function after being attacked. It can be seen that compared with the loss function under normal conditions, the loss function of the neural network cannot converge well after being attacked, and it is prone to sudden fluctuations. This can illustrate the importance of ensuring that the federated learning aggregation parameters are correct for systems with potential attacks.

## 4 CONCLUSION

Due to communication fluctuations and potential attacks in harsh environments, traditional task offloading models may be difficult to handle these challenges. In this paper, we propose a task offloading decision framework that combines federated learning and blockchain. The decision-making algorithm is placed on the user side to ensure the normal execution of task decisions in harsh environments, and then through the use of federated learning and blockchain structure, the training speed of the model is improved and the risk of potential illegal users is eliminated. It provides a more stable and feasible offloading solution for the popularization of edge computing in a wide range of fields in the future.

## ACKNOWLEDGMENTS

This work is supported by the National Natural Science Foundation of China (61801325, 62071327), Tianjin Research Innovation Project for Postgraduate Students and CCF-Tencent Open Research Fund.

## REFERENCES

[1] Leemon Baird. 1995. Residual algorithms: Reinforcement learning with function approximation. In *Machine Learning Proceedings 1995*. Elsevier, 30–37.
[2] Y. Bi, G. Han, S. Xu, X. Wang, C. Lin, Z. Yu, and P. Sun. 2019. Software Defined Space-Terrestrial Integrated Networks: Architecture, Challenges, and Solutions. *IEEE Network* 33, 1 (2019), 22–28.
[3] Noe Casas. 2017. Deep deterministic policy gradient for urban traffic light control. *arXiv preprint arXiv:1703.09035* (2017).
[4] Liang Huang, Xu Feng, Anqi Feng, Yupin Huang, and Li Ping Qian. 2018. Distributed Deep Learning-based Offloading for Mobile Edge Computing Networks. *Mobile Networks and Applications* (2018), 1–8.
[5] Yuzheng Li, Chuan Chen, Nan Liu, Huawei Huang, Zibin Zheng, and Qiang Yan. 2021. A Blockchain-Based Decentralized Federated Learning Framework with Committee Consensus. *IEEE Network* 35, 1 (Jan. 2021), 234–241.
[6] Yun Li, Shichao Xia, Bin Cao, Qilie Liu, et al. 2019. Lyapunov optimization based trade-off policy for mobile cloud offloading in heterogeneous wireless networks. *IEEE Transactions on Cloud Computing* (2019).
[7] Weiwei Liu, Huaming Wu, Tianhui Meng, Rui Wang, Yang Wang, and Cheng-Zhong Xu. 2021. AucSwap: A Vickrey auction modeled decentralized cross-blockchain asset transfer protocol. *Journal of Systems Architecture* 117 (Aug. 2021), 102102.
[8] Yuyi Mao, Changsheng You, Jun Zhang, Kaibin Huang, and Khaled B Letaief. 2017. A survey on mobile edge computing: The communication perspective. *IEEE Communications Surveys & Tutorials* 19, 4 (2017), 2322–2358.
[9] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin Riedmiller. 2013. Playing atari with deep reinforcement learning. *arXiv preprint arXiv:1312.5602* (2013).
[10] Viraaji Mothukuri, Reza M Parizi, Seyedamin Pouriyeh, Yan Huang, Ali Dehghantanha, and Gautam Srivastava. 2021. A survey on security and privacy of federated learning. *Future Generation Computer Systems* 115 (2021), 619–640.
[11] Luis Muñoz-González, Battista Biggio, Ambra Demontis, Andrea Paudice, Vasin Wongrassamee, Emil C Lupu, and Fabio Roli. 2017. Towards poisoning of deep learning algorithms with back-gradient optimization. In *Proceedings of the 10th ACM Workshop on Artificial Intelligence and Security*. 27–38.
[12] Huaming Wu, Yi Sun, and Katinka Wolter. 2020. Energy-Efficient Decision Making for Mobile Cloud Offloading. *IEEE Transactions on Cloud Computing* 8, 2 (April 2020), 570–584.
[13] Huaming Wu and Katinka Wolter. 2016. Analysis of the Energy-Performance Tradeoff for Delayed Mobile Offloading. In *Proceedings of the 9th EAI International Conference on Performance Evaluation Methodologies and Tools* (Berlin, Germany) *(VALUETOOLS'15)*. ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering), Brussels, BEL, 250–258. https://doi.org/10.4108/eai.14-12-2015.2262654
[14] Huaming Wu, Katinka Wolter, Pengfei Jiao, Yingjun Deng, Yubin Zhao, and Minxian Xu. 2021. EEDTO: An Energy-Efficient Dynamic Task Offloading Algorithm for Blockchain-Enabled IoT-Edge-Cloud Orchestrated Computing. *IEEE Internet of Things Journal* 8, 4 (Feb. 2021), 2163–2176.
[15] Dianlei Xu, Tingting Li, Yong Li, Xiang Su, Sasu Tarkoma, and Pan Hui. 2020. A Survey on Edge Intelligence. *ArXiv* abs/2003.12172 (2020).