DRL-Based Adaptive Sharding for Blockchain-Based Federated Learning

Yijing Lin, *Graduate Student Member, IEEE*, Zhipeng Gao, *Member, IEEE*, Hongyang Du, *Graduate Student Member, IEEE*, Jiawen Kang, Dusit Niyato, *Fellow, IEEE*, Qian Wang, Jingqing Ruan, and Shaohua Wan

***Abstract*— Blockchain-based Federated Learning (FL) technology enables vehicles to make smart decisions, improving vehicular services and enhancing the driving experience through a secure and privacy-preserving manner in Intelligent Transportation Systems (ITS). Many existing works exploit two-layer blockchain-based FL frameworks consisting of a mainchain and subchains for data interactions among intelligent vehicles, which resolve the limited throughput issue of single blockchain-based vehicular networks. However, the existing two-layer frameworks still suffer from a) strong dependency on predetermined and fixed parameters of vehicular blockchains which limit blockchain throughput and reliability; and b) high communication costs incurred by interactions among intelligent vehicles between the mainchain and subchains. To address the above challenges, we first design an adaptive blockchain-enabled FL framework for ITS based on blockchain sharding to facilitate decentralized vehicular data flows among intelligent vehicles. A streamline-based shard transmission mechanism is proposed to ensure communication efficiency almost without compromising the FL accuracy. We further formulate the proposed framework and propose an adaptive sharding mechanism using Deep Reinforcement Learning to automate the selection of parameters of vehicular shards. Numerical results clearly show that the proposed framework and mechanisms achieve adaptive, communication-efficient, credible, and scalable data interactions among intelligent vehicles.**

***Index Terms*— Blockchain sharding, federated learning, reputation, deep reinforcement learning.**

Manuscript received 17 January 2023; revised 4 May 2023; accepted 11 June 2023. Date of publication 22 June 2023; date of current version 18 October 2023. This work is supported by the National Natural Science Foundation of China (62072049, 62101012, 62102099), BUPT Innovation and Entrepreneur-ship Support Program (2023-YC-A131), the National Research Foundation, Singapore, and Infocomm Media Development Authority under its Future Communications Research & Development Programme, DSO National Laboratories under the AI Singapore Programme (AISG Award No: AISG2-RP-2020-019), Energy Research Test-Bed and Industry Partnership Funding Initiative, Energy Grid (EG) 2.0 programme, DesCartes and the Campus for Research Excellence and Technological Enterprise (CREATE) programme, and MOE Tier 1 (RG87/22). The associate editor coordinating the review of this article and approving it for publication was C. Li. *(Corresponding author: Zhipeng Gao.)*

Yijing Lin and Zhipeng Gao are with the State Key Laboratory of Networking and Switching Technology, Beijing University of Posts and Telecommunications, Beijing 100876, China (e-mail: yjlin@bupt.edu.cn; gaozhipeng@bupt.edu.cn).

Hongyang Du and Dusit Niyato are with the School of Computer Science and Engineering, Nanyang Technological University, Singapore 639798 (e-mail: hongyang001@e.ntu.edu.sg; dniyato@ntu.edu.sg).

Jiawen Kang is with the School of Automation, Guangdong University of Technology, Guangzhou 510006, China (e-mail: kavinkang@gdut.edu.cn).

Qian Wang is with the College of Computer Science, Beijing University of Technology, Beijing 100124, China (e-mail: wangqian2020@bjut.edu.cn).

Jingqing Ruan is with the Institute of Automation, Chinese Academy of Sciences, Beijing 100190, China (e-mail: ruanjingqing2019@ia.ac.cn).

Shaohua Wan is with the Shenzhen Institute for Advanced Study, University of Electronic Science and Technology of China, Shenzhen 518110, China (e-mail: shaohua.wan@uestc.edu.cn).

Color versions of one or more figures in this article are available at https://doi.org/10.1109/TCOMM.2023.3288591.

Digital Object Identifier 10.1109/TCOMM.2023.3288591

I. Introduction

With the development of Intelligent Transportation Systems (ITS), vehicles are connected to generate massive amounts of data. To make better use of this part of data, blockchain and Federated Learning (FL) are widely adopted to help intelligent vehicles to make decisions, improve vehicular services and enhance the driving experience in a secure and privacy-preserving manner.

Blockchain is a peer-to-peer trust network [1], which provides trusted and decentralized data records through chain structures, consensus algorithms, and encryption technologies in ITS. Moreover, vehicular business logic is executed through programmable smart contracts [2]. A decentralized and traceable vehicular application will finally be implemented through blockchain and smart contracts among intelligent vehicles. FL is a new distributed training paradigm, which can provide collaborative training and iterative parameter sharing among multiple intelligent vehicles [3]. Currently, an integration of FL with vehicular blockchains can solve problems such as single point of failure and model security in ITS [4]. Many works [5], [6], [7], [8] exploit two-layer vehicular blockchain-based FL frameworks consisting of a mainchain and subchains for data interactions among intelligent vehicles, which can resolve the limited throughput issue of single blockchain-based vehicular networks. However, the two-layer frameworks with one mainchain and specific subchains for ITS are still facing the following problems in ITS.

- **Q1:** High communication costs are incurred due to frequent interactions among intelligent vehicles between the mainchain and subchains. Data generated from intelligent vehicles of subchains need to frequently interact with the mainchain to synchronize updates, which costs high communication overhead.
- **Q2:** The two-layer frameworks strongly depend on predefined and fixed parameters of vehicular blockchains like specific subchain numbers, data size, and consensus algorithms, which affects the blockchain scalability of an integration FL with vehicular blockchains in ITS.

The static blockchain scalability and high communication overhead affect the efficiency of the framework with

a mainchain and subchains in ITS, but have seldom been studied in existing works. To address the above problems, we construct an adaptive blockchain-enabled FL framework for efficient data interactions in ITS with blockchain sharding and deep reinforcement learning, which avoids the limitations of vehicular shards with fixed parameters. The framework utilizes a reputation-based shard selection mechanism to evaluate behaviors of intelligent vehicles within shards to maintain data credibility. Moreover, we propose a streamline-based shard transmission mechanism to reduce communication overhead almost without compromising the FL accuracy. Parameters generated from vehicular shards are committed to neighboring shards by the leader within shards which can reduce interactions between shards and vehicular blockchains, and improve the throughput of vehicular blockchains. Finally, we design an adaptive blockchain sharding mechanism based on Deep Reinforcement Learning (DRL) to improve the blockchain dynamic scalability in the blockchain-enabled FL framework for ITS. The main contributions of this paper are summarized as follows.

- We design an adaptive blockchain-enabled FL framework for ITS with parallel shards to facilitate vehicular data flows. A reputation-based shard selection mechanism is proposed to identify and eliminate malicious vehicles.
- To reduce frequent interactions among parallel shards in the framework, we design a streamline-based shard transmission mechanism to ensure high communication efficiency without compromising the FL accuracy (**for Q1**).
- To improve the adaptive scalability of the framework, we formulate the problem and establish a DRL-based sharding mechanism to automate the construction of vehicular shards. Unlike traditional methods that rely on a strong assumption of complete information, our DRL-based sharing mechanism can adaptively learn and obtain an optimal policy in an online fashion (**for Q2**).

The rest of the paper is organized as follows. Related work is discussed in Section II. The adaptive blockchain-enabled FL framework for ITS and a reputation-based shard selection mechanism is illustrated in Section III. A streamline-based shard transmission mechanism is illustrated in Section IV. An adaptive sharding mechanism based on DRL is presented in Section V. The proposed framework is evaluated in Section VI. Conclusions and future works are discussed in Section VII.

II. Related Work

The proposed mechanism is based on emerging technologies such as blockchain sharding, federated learning, reputation mechanisms and deep reinforcement learning. Therefore, we divide the literature review into three parts: (a) Blockchain-enabled FL framework, (b) Blockchain Sharding, and (c) DRL-based Adaptive Sharding.

A. Blockchain-Enabled FL Framework

FL [3] was first proposed to train data remaining distributed over many clients and has been applied to speech recognition, image processing, COVID-19, and other fields. The theoretical studies of malicious attacks, aggregation algorithms, or convergence on FL are also active fields, like Krum [9] and FABA [10]. Traditional federated learning is a primary-secondary architecture, i.e., a centralized primary node aggregates parameters from a large number of distributed clients. Therefore, the primary node becomes the bottleneck of the whole framework, which can lead to a single point of failure.

The decentralized and traceable nature of blockchain makes it an essential complement to federated learning. Many researchers combine blockchain with decentralized aggregation methods to address single points of failure and secure models. Warnat-Herresthal et al. [11] combined edge computing, blockchain, and federated learning to propose a decentralized training model, which allows all members to merge parameters with equal rights without violating privacy. Zhou et al. [12] proposed Byzantine-resilience for distributed learning based on blockchain to protect model safety. Jin et al. [13] proposed a cross-cluster method based on cross-chain and federated learning to improve aggregation efficiency and reduce communication delay. Chai et al. [8] proposed a hierarchical blockchain and federated learning framework for knowledge sharing. Yuan et al. [5], [6] proposed a calculation method based on sharding and federated learning, ChainsFL, which used the Raft algorithm inside the shard [14] and adopted DAG on the mainchain [15]. However, the above methods adopt a static strategy for sharding, which cannot meet the requirement of dynamic environments in ITS.

As the development of the blockchain-based FL framework, researchers are exploring various ways to address the challenges of reliability in dynamic environments. One promising approach is to incorporate reputation-based mechanisms into the framework to eliminate malicious nodes. Feng et al. [7] proposed a two-layered blockchain architecture for federate learning and construct a reputation-based incentive mechanism to resist malicious nodes. Huang et al. [16] utilized multi-weighted subjective logic to improve reputation update in distributed vehicular edge computing. Kang et al. [17] proposed a reputation-based work selection scheme for FL through multiweight subjective logic. The proposed scheme combined direction reputation opinions with recommended reputation opinions. They also proposed a reliable worker selection mechanism for the consortium blockchain-based FL framework [18]. And they proposed a three-weight subjective logic model to precisely manage reputation in blockchain-based vehicular networks [19]. Zhang et al. [20] proposed a proof-of-reputation blockchain to evaluate the quality of services. The proposed mechanism evaluated all other nodes through personal reputations and aggregate personal reputations to calculate the global reputations of all nodes. Gao et al. [21] proposed a node evaluation mechanism based on the state of nodes to divide nodes into shards. The proposed mechanism utilized multiweight subjective logic to select nodes. Moudoud et al. [22] proposed a subjective multi-weight logic based incentive mechanism to improve the reliability of FL based on the blockchain sharding. However, the above methods do not consider neutral situations like network failures incurred in the blockchain-based FL framework.

B. Blockchain Sharding

Since blockchain is achieved by multiple nodes concurrently processing transactions, its performance is limited by the network scale to broadcast and validate messages. Blockchain sharding is a promising approach to improve blockchain performance. Sharding was first proposed to scale up to distributed transactions across hundreds of data centers [23]. Elastico [24] was the first sharding protocol to support public blockchains where the TPS increases linearly corresponding to the number of shards. OmniLedger [25] proposed state sharding and exploited PoW and BFT to ensure security and correctness. RapidChain [26] exploited synchronous BFT to ensure security and implement cross-shard transactions by relaying transactions. Chainspace [27] designed a distributed commit protocol to guarantee consistency and scalability in the decentralized infrastructure. Monoxide [28] exploited asynchronous consensus zones to support blockchain scalability while maintaining decentralization and security. Dang et al. [29] enhanced consensus protocol to improve the throughput of shard-based blockchain. They introduced Intel SGX to achieve the high performance of shard formation protocol. Pyramid [30] exploited a layered blockchain system to reduce overhead and maintain the sharding performance in the cross-shard transactions. Huang et al. [31] proposed a multi-queue model based on the drift-plus-penalty technique and PBFT to realize elastic resource allocations in the sharding blockchain. These previous works focus on the sharding mechanism for traditional blockchain networks, and do not support adaptive sharding mechanisms. In contrast, the adaptive sharding mechanism we propose in this paper aims at the blockchain-enabled FL framework based on DRL in ITS.

C. DRL-Based Adaptive Sharding

Although blockchain sharding can improve the performance of processing and validating transactions, it is difficult to handle with complex and flexible on-chain environments. Deep Reinforcement Learning [32] is an emerging technology to adapt to various environments. It has been widely used in the edge computing [33]. It integrates deep learning and reinforcement learning to provide the understanding and decision-making abilities to end-to-end learning. Liu et al. [34] introduced DRL to improve blockchain throughput. It built an efficient blockchain which can balance decentralization, security, and scalability by dynamically adjusting the size and block interval of the blockchain. Yun et al. [35] builded a sharding network on the basis of [34]. Both sharding and Directory Committee used the PBFT algorithm and combined it with DRL to build a network that can resist malicious attacks on the blockchain. Lu et al. [36] proposed to implement an algorithm combining DRL and federated learning through smart contracts to achieve node selection to improve the reliability of model parameters. Zhang et al. [37] proposed a state block-based sharding network and dynamically constructed shards based on DDPG [38]. However, those existing works do not break through the resource limitations of blockchain with fixed parameters, which is not able to improve blockchain performance adaptively.

TABLE I
KEY NOTATIONS

Notation	Definition	Notation	Definition
b_{ij}^{te}	Belief	d_{ij}^{te}	Distrust
n_{ij}^{te}	Posteriori uncertainty	e_{ij}^{te}	Priori uncertainty
χ_{ij}^{te}	Weight	M_t	Average size
R_t	Transmission rate	T_v	Verification time
M	Message size	Υ_{ij}^{te}	Reputation
q_{ij}^{te}	Success probability	R_{ij}^{te}	Expected reputation
ϕ_{ij}^{te}	Familiarity value	ψ_{ij}^{te}	Freshness value
T_{round}	Shard formation	T_{intra}	Intra-shard time
$T_{prop}^{s,\delta}$	Message propagation	$T_{val}^{s,\delta}$	Validation delay
T_{inter}	Inter-shard time	Ω	Throughput
$\mathcal{G}_i$	Global round	$\mathcal{S}^t$	State space
$\mathcal{A}^t$	Action space	R^t	Reward

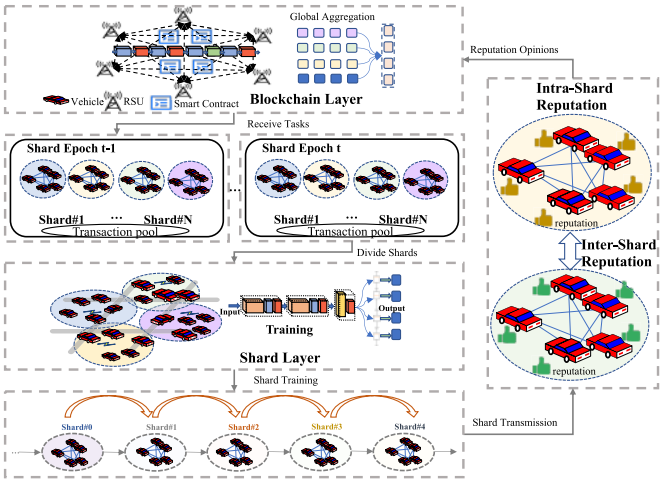


Fig. 1. Architecture of the adaptive blockchain enabled federated learning framework for ITS.

III. TWO LAYER BLOCKCHAIN-BASED FL FRAMEWORK

In this section, we introduce the proposed two-layer blockchain-based FL framework and their respective components to address the challenges of trust among participating vehicles in the federated learning process. We further elaborate on the reputation-based shard selection scheme to maintain trust among the vehicles involved in the framework. It effectively assesses the trustworthiness of the participating vehicles based on their intra-shard and inter-shard reputations, thereby eliminating malicious vehicles from the system. Key notations are summarized in Table I.

A. System Model

The architecture of the adaptive blockchain-enabled FL framework for ITS can be divided into two main layers: the blockchain Layer and the shard Layer, as shown in Fig. 1. These layers work in tandem to provide an efficient and secure environment for the federated learning process to take place among the vehicles. The description of the proposed framework is as follows.

1) Blockchain Layer: The blockchain layer consists of one blockchain, in which Roadside Units (RSUs) can collect transactions and aggregate training parameters to implement federated learning process. Blockchain is utilized to record

hashes of learning results to maintain the security in a weak trust environment. Besides, blockchain receives training logic and issues training tasks from intelligent vehicles with limited computing resources through smart contracts. Other intelligent vehicles can join the blockchain layer to complete the tasks. Since the throughput of single blockchain is limited, the blockchain layer can dynamically configure multiple shards with consensus, data size and number of shards to divide intelligent vehicles into different networks in different shard epochs. Intelligent vehicles from shards can train a highly accurate and generalizable machine learning model to provide vehicular services to users.

2) Shard Layer: Intelligent vehicles in different shards collect environmental data as datasets. They receive model training tasks from smart contracts of blockchain, and then complete training tasks. Each intelligent vehicle acts as a follower in shard and a worker in FL to accomplish tasks collaboratively, including shard training and transmission. A leader vehicle is selected from consensus protocol, aggregates parameters of this shard, and submits intermediate results to neighboring shards. Since blockchain can dynamically configure shards, the formation of shard will be eliminated after a shard epoch. When each shard epoch is completed, model training results of multiple shards are uploaded to blockchain for global aggregation. Vehicles of shards receive optimal parameters to start next shard epoch. The details of the workflow are illustrated in Section IV. Since vehicles within or across shards may perform malicious actions to corrupt task results, it is necessary to construct reputation opinions among participants to eliminate malicious vehicles, as shown in Section III-B. There are two types of reputation opinions including intra-shard and inter-shard opinions to manage trust among unknown vehicles from the same or different shards, as shown in Fig. 1.

B. Reputation-Based Shard Selection Scheme

In the proposed two-layer blockchain-based framework, since unreliable vehicles may falsify data to obstacle the aggregation process within or across shards, subjective logic is a possible solution to manage the degree of trust among unknown vehicles through opinion metric, which can eliminate malicious nodes via reputation opinions. The Traditional Subjective Logic (TSL) [39] utilizes a vector of belief, disbelief, uncertainty and relative atomicity functions to represent opinion metric, which can be expressed as $b = \frac{r}{r+s+2}$, $d = \frac{s}{r+s+2}$, and $u = \frac{2}{r+s+2}$. r and s represent the positive and negative observations of events. Besides, authors in [40] proposed a Multi-Weight Subjective Logic (MWSL) to consider some factors involved in shards. However, those methods do not consider neutral situations like network failures, which cause inaccurate expected opinions.

Therefore, we present a reputation-based shard selection scheme based on Multi-Weight Multi-Value Subjective Logic (MWMVSL) to eliminate malicious vehicles. The shard selection scheme includes the inter and intra shard reputations corresponding to vehicles and shards. Intra reputations refer to evaluating vehicles within the same shard. Inter reputations are utilized to judge vehicles across shards.

1) Intra-Shard Reputation of Vehicles: We divide a time duration of shards into multiple timeslots $t \in \mathcal{T} = \{t_1, \ldots, t_e, \ldots, t_E\}$ to express the survival time of shards. The reputation opinion of a leader vehicle i on a follower vehicle j in a shard s and a timeslot t_e can be represented by $\Upsilon_{ij}^{t_e} = \{b_{ij}^{t_e}, d_{ij}^{t_e}, n_{ij}^{t_e}, e_{ij}^{t_e}, q_{ij}^{t_e}, a\}$, where $b_{ij}^{t_e} \in [0,1], d_{ij}^{t_e} \in [0,1], n_{ij}^{t_e} \in [0,1]$ and $e_{ij}^{t_e} \in [0,1]$ stand for belief, distrust, posteriori uncertainty and priori uncertainty (without evidences), respectively [41]. $q_{ij}^{t_e} \in [0,1]$ represents the success probability of processing transactions which determine the priori uncertainty $e_{ij}^{t_e}$ of the reputation opinion $\Upsilon_{ij}^{t_e}$. The base rate $a \in [0,1]$ is the coefficient utilized to compute the probability expectation value of a reputation opinion [40]. Besides, the relationship between variables is $b_{ij}^{t_e} + d_{ij}^{t_e} + n_{ij}^{t_e} + e_{ij}^{t_e} = 1$. Referring to the secure sharding using subjective logic model [22], we can obtain $b_{ij}^{t_e} = \frac{\alpha_i^{t_e}(1-e_{ij}^{t_e})}{\alpha_i^{t_e}+\beta_i^{t_e}+\gamma_i^{t_e}}$, $d_{ij}^{t_e} = \frac{\beta_i^{t_e}(1-e_{ij}^{t_e})}{\alpha_i^{t_e}+\beta_i^{t_e}+\gamma_i^{t_e}}$, $n_{ij}^{t_e} = \frac{\gamma_i^{t_e}(1-e_{ij}^{t_e})}{\alpha_i^{t_e}+\beta_i^{t_e}+\gamma_i^{t_e}}$, and $e_{ij}^{t_e} = 1 - q_{ij}^{t_e}$, where $\alpha_i^{t_e}, \beta_i^{t_e}$, and γ^{t_e} are the number of positive, negative and neutral interactions during a time slot t_e. There are three types of interactions, including honest, dishonest and neutral interactions among vehicles. Honest interactions process transactions to increase intra-shard reputations of vehicles. Dishonest interactions like malicious vehicles deliberately disobey the predefined protocol. Neutral interactions refer to vehicles which accidentally occur network failures leading to negative effects. Therefore, according to [41], the expected reputation opinion of a leader vehicle i on follower vehicle j in a shard s and a timeslot t_e is

$$R_{ij}^{t_e} = b_{ij}^{t_e} + a n_{ij}^{t_e} + \frac{1}{2} e_{ij}^{t_e}. \tag{1}$$

To control the intra-shard reputation opinions of vehicles more flexibly and dynamically, we consider the factors of familiarity and freshness of interactions to calculate the reputation opinions of vehicles.

a) Familiarity: Followers of each shard feed back positive, negative and neutral responses to a leader in a time epoch t_e. When a follower's response is the same as that of the majority of vehicles, the leader increases its reputation; otherwise, it can decrease reputations of vehicles. The leader considers a follower's response as a neutral interaction if a follower crashes down and feeds back nothing. Therefore, the familiarity value of a leader i in a shard s for a follower j during a time epoch t_e, $\phi_{ij}^{t_e}$, is equal to the sum of the belief, distrust and posteriori uncertainty values as

$$\phi_{ij}^{t_e} = b_{ij}^{t_e} + d_{ij}^{t_e} + n_{ij}^{t_e} = 1 - e_{ij}^{t_e}. \tag{2}$$

b) Freshness: Since each shard lasts for a time epoch t_e, consensus processes will be executed for many times. Moreover, it is hard to predict behaviours of each unknown vehicles. The follower may return positive, negative and neutral results to a leader in different consensus rounds. Therefore, recent interactions between a leader i and a follower j are more important than past events, which can reflect the reputation opinion with a larger weight. The freshness value of a leader i in a shard s during a time epoch t_e for a follower j, $\psi_{ij}^{t_e}$, is denoted as $\psi_{ij}^{t_e} = \theta^{t_E - t_e}$, where θ is predefined parameters to adjust the freshness [16].

Considering the familiarity and freshness factors in the intra-shard reputation, the overall weight of the reputation opinion of a leader i in a shard s for a follower j during a time epoch t_e can be denoted as $\chi_{ij}^{t_e} = \mathcal{F}_1\phi_{ij}^{t_e} + \mathcal{F}_2\psi_{ij}^{t_e}$, where $\mathcal{F}_1$ and $\mathcal{F}_2$ are predefined weighting factors to adjust the values of familiarity and freshness, which satisfy $\mathcal{F}_1 + \mathcal{F}_2 = 1$. Therefore, the final intra-shard reputation opinion of a leader in a shard s during a time window for a follower j can be denoted as $b_{ij} = \frac{\sum_{t_1}^{t_E}\chi_{ij}^t b_{ij}^t}{\sum_{t_1}^{t_E}\chi_{ij}^t}$, $d_{ij} = \frac{\sum_{t_1}^{t_E}\chi_{ij}^t d_{ij}^t}{\sum_{t_1}^{t_E}\chi_{ij}^t}$, $n_{ij} = \frac{\sum_{t_1}^{t_E}\chi_{ij}^t n_{ij}^t}{\sum_{t_1}^{t_E}\chi_{ij}^t}$, and $e_{ij} = \frac{\sum_{t_1}^{t_E}\chi_{ij}^t e_{ij}^t}{\sum_{t_1}^{t_E}\chi_{ij}^t}$. The final expected reputation value of a vehicle can be denoted as

$$R_{ij} = \frac{\sum_{t_1}^{t_E}\chi_{ij}^t R_{ij}^t}{\sum_{t_1}^{t_E}\chi_{ij}^t}. \tag{3}$$

2) Inter-Shard Reputation: The intra-shard reputation cannot be computed in the inter-shard reputation because vehicles in different shards have different intra-shard reputations. Therefore, based on [41], we construct the discount and parallel reputation opinions to introduce inter-shard reputation based on the intra-shard reputation opinions.

a) Discount reputation: The intra-shard reputations are updated according to interactions between a leader and followers in the same shard. It is inconvenient for other leaders outside a shard to construct reputation opinions on followers on the shard. When the shard begins to exchange a leader, other leaders also need some time to form a reputation opinion about the new leader. Therefore, we introduce discount reputation opinions so that other leaders can form quickly reputation opinions on followers from other shards. Leaders are responsible to transmit information across shards. We assume that a and b are leaders from different shards, and c is a follower of b from the same shard. The reputation opinions between $a \rightarrow b$ and $b \rightarrow c$ can be expressed as $\Upsilon_{ab}^{disc} = \{b_{ab}^{disc}, d_{ab}^{disc}, n_{ab}^{disc}, e_{ab}^{disc}\}$ and $\Upsilon_{bc}^{disc} = \{b_{bc}^{disc}, d_{bc}^{disc}, n_{bc}^{disc}, e_{bc}^{disc}\}$. The discount reputation opinion $a \rightarrow c$ can be expressed as $\Upsilon_{ac}^{disc} = \{b_{ac}^{disc}, d_{ac}^{disc}, n_{ac}^{disc}, e_{ac}^{disc}\}$, where the belief, distrust, posteriori uncertainty and priori uncertainty values can be denoted as

$$\begin{aligned} b_{ac}^{disc} &= b_{ab}^{disc} b_{bc}^{disc}, \quad d_{ac}^{disc} = b_{ab}^{disc} d_{bc}^{disc} \\ n_{ac}^{disc} &= 1 - b_{ac}^{disc} - d_{ac}^{disc} - e_{bc}^{disc}, \quad e_{ac}^{disc} = e_{bc}^{disc}. \end{aligned} \tag{4}$$

b) Parallel reputation: When there are different types of tasks in the proposed framework, shards in parallel do not have frequent interactions, which cannot construct valid reputation opinions to evaluate vehicles from other shards. Therefore, we introduce parallel reputation opinions to construct trust between parallel shards. We assume a and b are leaders of parallel shards from different tasks, and c takes part in the same task with a now while used to participate in the same task within b in the past. The reputation opinions on $a \rightarrow c$ and $b \rightarrow c$ can be denoted as $\Upsilon_{ac}^{para} = \{b_{ac}^{para}, d_{ac}^{para}, n_{ac}^{para}, e_{ac}^{para}\}$ and $\Upsilon_{bc}^{para} = \{b_{bc}^{para}, d_{bc}^{para}, n_{bc}^{para}, e_{bc}^{para}\}$. Then the parallel reputation opinion $a \rightarrow b$ can be denoted as $\Upsilon_{ab}^{para} = \{b_{ab}^{para}, d_{ab}^{para}, n_{ab}^{para}, e_{ab}^{para}\}$, where the believe, distrust, posteriori uncertainty and priori uncertainty values can be denoted as

$$\begin{aligned} b_{ab}^{para} &= \frac{e_{bc}^{para} b_{ac}^{para} + e_{ac}^{para} b_{bc}^{para}}{e_{ac}^{para} + e_{bc}^{para} - e_{ac}^{para} e_{bc}^{para}} \\ d_{ab}^{para} &= \frac{e_{bc}^{para} d_{ac}^{para} + e_{ac}^{para} d_{bc}^{para}}{e_{ac}^{para} + e_{bc}^{para} - e_{ac}^{para} e_{bc}^{para}} \\ n_{ab}^{para} &= \frac{e_{bc}^{para} n_{ac}^{para} + e_{ac}^{para} n_{bc}^{para}}{e_{ac}^{para} + e_{bc}^{para} - e_{ac}^{para} e_{bc}^{para}} \\ e_{ab}^{para} &= \frac{e_{ac}^{para} e_{bc}^{para}}{e_{ac}^{para} + e_{bc}^{para} - e_{ac}^{para} e_{b\rightarrow c}^{para}}. \end{aligned} \tag{5}$$

IV. A Streamline-Based Shard Transmission Mechanism

To improve communication efficiency almost without compromising accuracy, in this section, we illustrate a streamline-based shard transmission mechanism. The workflow of the mechanism is divided into five stages according to the execution order of computing tasks, as shown in Fig. 1. It includes registration, task release, shard training, shard transmission, and global aggregation.

A. Registration

Vehicles with FL training requirements and resources need to register on the blockchain to obtain legal identities. The identity includes account address, public key, private key, certificate for authentication, and transaction verification.

B. Task Release

Vehicles write training requirements through an interface provided by smart contract into a queue. The requirement includes models, test sets, shard configuration, local training epochs τ, and global communication rounds $\mathcal{G}$. The signed transaction is broadcast to intelligent vehicles of blockchain to set up shards to execute training tasks. Intelligent vehicles are divided into shards according to their account addresses on the blockchain. The signed transaction is the genesis transaction of shards. The genesis transaction writes training tasks into shards so that vehicles of shards know and execute tasks.

C. Shard Training

Intelligent vehicles $\{F_{11}, F_{12}, \cdots, F_{sv}, \cdots, F_{\mathcal{S}\mathcal{V}}\}$ in shards receive training tasks from the genesis transaction, and perform off-chain training tasks according to training tasks, where $\mathcal{S}$ is the number of shards and $\mathcal{V}$ is the number of vehicles in a shard. F_{sv} represents a follower vehicle v in a shard s. An vehicle F_{sv} needs to use the input-output pairs (x, y) of its own local environmental dataset D_{sv} to train model parameters w for local training epochs τ, where x is a vector of features and y is a vector of labels. The loss function for each sample k is defined as $l(x_k, y_k|w)$. The goal of F_{sv} is to calculate optimal model parameters $\min_{w\in\mathbb{R}^d} f_{sv}(w)$ where $f_{sv}(w) \leftarrow \frac{1}{\tau}\sum_{j=1}^{\tau} l(x_k, y_k|w)$, where τ is the local training epoch of FL. By using the gradient descent algorithm on D_{sv}, w of $f_{sv}(w)$ for each sample k can be updated layer by layer $w_{sv} \leftarrow w_{sv} - \eta\nabla l(x_k, y_k|w_k)$, where η is the learning rate of F_{sv}. F_{sv} will execute local epochs

τ to obtain better w_{sv}. After F_{sv} finishes local epochs, F_{sv} signs a transaction to share optimal w_{sv} with the leader vehicle L_s in the same shard s. L_s exploits the federated averaging algorithm [3] to aggregate optimal parameters in a shard. The optimization objective $f_s(w)$ and the update of model parameters w_s of the leader vehicle L_s in a shard are

$$\min_{w\in\mathbb{R}^d} f_s(w) \leftarrow \frac{1}{\mathcal{V}}\sum_{v=1}^{\mathcal{V}}\frac{|D_{sv}|}{\sum_{j=1}^{\mathcal{V}} D_{sj}} f_{sv}(w)$$
$$w_s \leftarrow \sum_{v=1}^{\mathcal{V}}\frac{|D_{sv}|w_{sv}}{\sum_{j=1}^{\mathcal{V}} D_{sv}}. \tag{6}$$

$\mathcal{V}$ is the number of vehicles in a shard and D_{sv} is the dataset of the follower vehicle F_{sv} in the shard s.

D. Shard Transmission

The leader vehicle L_s does not directly upload optimized models w_s to blockchain like ChainsFL [5] and TwoLayer [7], which may cause huge communication overhead and is limited by the throughput of blockchain due to frequent interactions between the mainchain and subchains. Therefore, inspired by Chained Hotstuff [42], we propose a streamline-based interaction mechanism. Each shard includes a leader vehicle and multiple follower vehicles elected by consensus algorithms.

The leader vehicle L_s represents the leader vehicle of a shard s. L_{s-1} represents the leader vehicle of the previous shard $s-1$. L_s receives w_{s-1} from the previous leader L_{s-1}, and verifies whether the current round $\mathcal{G}_i$ is consistent. If so, L_s broadcasts w_{s-1} with its signature to followers in a shard as the shard aggregation parameters in the current round. w_{s-1} will be the initial training parameters for the next communication round. Otherwise, L_s broadcasts the last valid w_{s-1} to followers $\{F_{11}, F_{12}, \cdots, F_{sv}, \cdots, F_{s\mathcal{V}}\}$ in this shard.

The follower vehicle F_{sv} represents the follower vehicle v in a shard s. F_{sv} obtains the latest w_{s-1} from L_s and extracts the parameters of the last communication rounds $\mathcal{G}_i$ from w_{s-1}. The follower vehicle F_{sv} needs to use the shard aggregation parameter w_{s-1} of the previous shard L_{s-1} to start a new communication round. Since the data structure of all shards is the same, the number of aggregations is reduced by streamlining to improve communication efficiency. $F_{(s+1)v}$ received optimal parameters w_s as $w_{(s+1)v}$ from L_s in $\mathcal{G}_i$. $F_{(s+2)v}$ received optimal parameters w_{s+1} as $w_{(s+2)v}$ from L_{s+1} in $\mathcal{G}_i$. The streamline is contacted with the above shards to train latest model parameters together for better use of data.

E. Global Aggregation

When the current communication round $\mathcal{G}_i$ is identical to the specific round $\mathcal{G}$, leader vehicles $\{L_0, \cdots, L_s, \cdots, L_{\mathcal{S}}\}$ in all shards will upload optimal parameters $\{w_0, \cdots, w_s, \cdots, w_{\mathcal{S}}\}$ to blockchain for the global aggregation while $\mathcal{S}$ represents the number of shards. Each leader vehicle in a shard writes hashes of the latest parameters into smart contract, signs the contract transaction, and share parameters with RSUs in the blockchain. RSUs validate messages of $\{L_0, \cdots, L_s, \cdots, L_{\mathcal{S}}\}$ and performs global aggregation. After completing the training task, leaders that uploads final results deletes it in the queue and starts the next training taks.

V. Adaptive Sharding Construction Mechanism Based on DRL

According to the above description, we can know that the bottleneck of the proposed mechanism is shards, which strongly depends on predetermined and fixed construction of shards. The fixed shards will limit the throughput of the blockchain-enabled FL framework. In this section, we first present the problem formulation to analyse critical factors in the construction of shards. Moreover, we design a DRL-based model to improve the performance of the proposed framework.

A. Model Analysis

As described in Section IV, the workflow consists of five stages, shard configuration, off-chain computation, shard consensus, aggregation, and propagation. Off-chain computation and sharding consensus belong within shards, while aggregation and propagation between shards. Therefore, we divide the processing time in the shard formation round into intra-shard time T_{intra}, inter-shard time T_{inter}, and configuration time T_{reco}. The DRL training process is only executed before shards are formed, the FL training process begins only after shards are formed. Then, if the shards are deformed, the FL training process cannot be executed. Therefore, the processing time can be expressed as

$$T_{round} = T_{intra} + T_{inter} + T_{reco}. \tag{7}$$

1) Intra-Shard Time T_{intra}*:* Assuming that there are $\mathcal{V}$ vehicles in each shard, $\mathcal{S}$ number of shards, the consensus algorithm of the shard is δ, and $epoch$ is the unit of measure for shard formation which is different from the epoch of FL. A follower vehicle F_{sv} needs to share parameters with L_s in a shard s. Therefore, T_{intra} consists of three stages, off-chain training time T^s_{comp}, message propagation $T^{s,\delta}_{prop}$ and validation delay $T^{s,\delta}_{val}$ stage. Therefore, the intra-shard time can be expressed as

$$T_{intra} = T^{s,\delta}_{prop} + T^{s,\delta}_{val} + \max epoch \times T^s_{comp}. \tag{8}$$

We take the PBFT [43] algorithm as an example. In the pre-prepare stage, the leader broadcasts the block to $\mathcal{V}-1$ vehicles. During the prepare phase, $\mathcal{V}-1$ vehicles broadcast message to $\mathcal{V}$ vehicles. During the commit phase, $\mathcal{V}$ vehicles broadcast message to $\mathcal{V}$ vehicles. Therefore, the message propagation and validation delay can be expressed as

$$T^{s,\delta}_{prop} = 2\mathcal{V}(\mathcal{V}-1)\frac{M}{R_t} \text{ and } T^{s,\delta}_{val} = 3T_v, \tag{9}$$

where R_t, T_v and M are the message transmission rate, message verification time and message size at each stage.

2) Inter-Shard Time T_{inter}*:* T_{inter} is the time when leader vehicles of shards send messages to other leaders of the next shard. Therefore, T_{inter} is proportional to the message size M while T_{inter} is inversely proportional to the message transmission rate at each stage R_t. It can be expressed as $T_{inter} = \frac{M}{R_t}$.

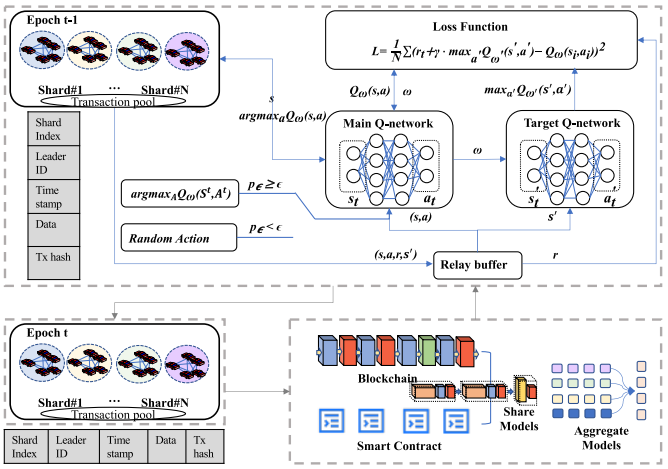


Fig. 2. Interaction of the DQN agent and network of the proposed mechanism.

3) Performance: Assuming that the average data size is M_t with the total of $\mathcal{S}$ shards, the throughput of the entire system is $\Omega = \frac{M\mathcal{S}}{M_t T_{round}}$, where $\mathcal{S} \leq \mathcal{S}_{\max}$ and $M \leq M_{\max}$. $\mathcal{S}_{\max}$ is the maximum number of shards and $M_{\max}$ is the maximum data size. At the same time, T_{inter} should be limited within a duration of T_{intra}.

B. Model Design

1) DQN-Based Dynamic Sharding Mechanism: To consider the dynamic state of the environment, we use an adaptive sharding mechanism based on Deep Q-learning Network (DQN) [32] to solve the above problem, as shown in Fig. 2. Algorithm 1 shows the workflow of the proposed mechanism.

a) State space: The state space at the discrete time epoch is as a union of the state of the total number of vehicles $\mathcal{N}$ in all shards, data transmission rate R_t, the training time T_{comp}, and the specific global communication round $\mathcal{G}_i$. Therefore, the state space can be expressed as $\mathcal{S}^t = [\mathcal{N}, R_t, T_{comp}, \mathcal{G}_i]^t$.

b) Action space: Several actions should be done to adapt to the dynamic environment. They include the number of shards $\mathcal{S}$, data size M, leader of shards $\boldsymbol{l}$, and consensus algorithms of shards δ. Therefore, the action space can be expressed as $\mathcal{A}^t = [\mathcal{S}, M, \boldsymbol{l}, \delta]^t$, where $\boldsymbol{l} = l_k, l_k \in \{0, 1\}$. Vehicles are selected as leader of shards when $l_k = 1$ and $l_k = 0$ otherwise. $\delta = \{0, 1, 2, \cdots\}$ represents different consensus algorithms in the shards like RAFT [14], HOTSTUFF [42], PBFT [43], and VRF [44].

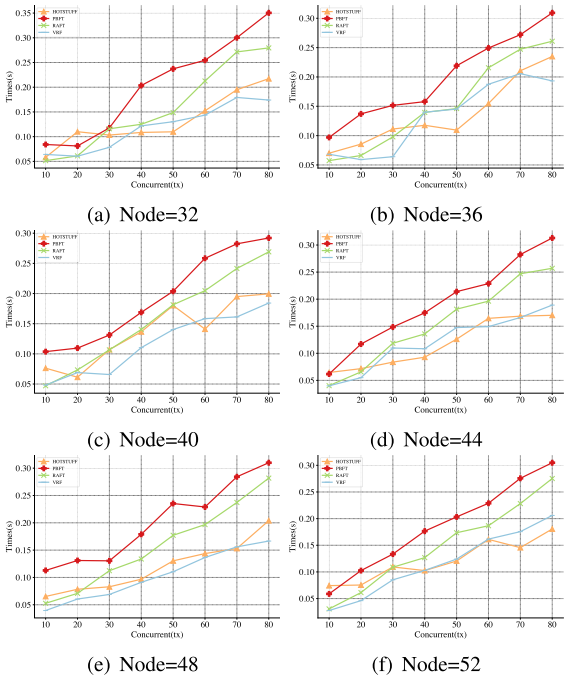
c) Reward function: Shards take action to obtain the next state, and the reward function is obtained according to feedback. The reward function is used to maximize the blockchain throughput of the mechanism, which can be expressed as

$$\mathcal{P}1 : R^t(S^t, A^t) = \frac{M\mathcal{S}}{T_{round} M_t},$$
$$\mathcal{C}1 : T_{inter} \leq \gamma T_{intra}, \quad \mathcal{C}2 : \mathcal{S} \leq \mathcal{S}_{max},$$
$$\mathcal{C}3 : M \leq M_{max},$$

if constraints $\mathcal{C}1 - C3$ are satisfied. If constraints are not satisfied, it will be 0.

Algorithm 1 DRL-Based Sharding Mechanism

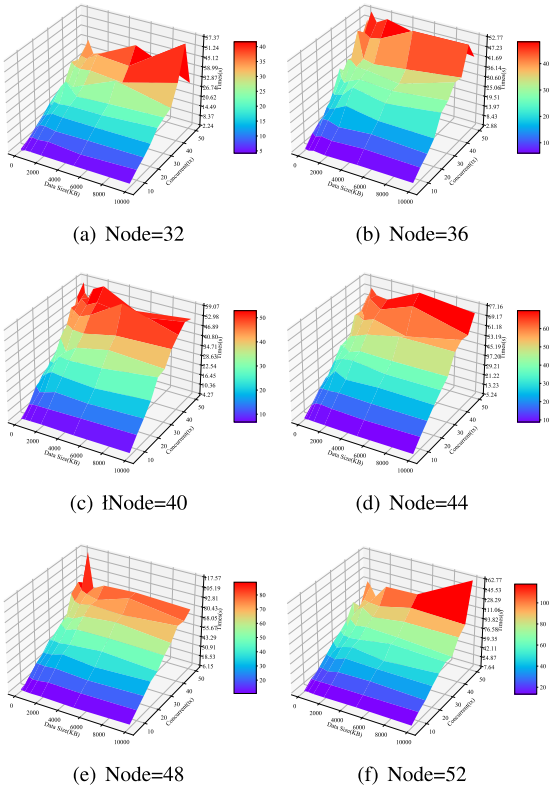
Initialize $Q_\omega(s, a)$ with random weights ω
Initialize $Q_{\omega'}$ with weights $\omega' \leftarrow \omega$
Initialize replay buffer $\mathcal{B}$
for episode $e = 1 \rightarrow E$ **do**
  Initialize environment and receive initial observation state s_1
  for epoch $t = 1 \rightarrow T$ **do**
    if $p_\epsilon < \epsilon$ **then**
      select random action a_t
    else
      select action $a_t = \arg\max_A Q_\omega(S^t, A^t)$
    end if
    execute a_t, observe reward r_t, and proceed to next state s_{t+1}
    store transition (s_t, a_t, r_t, s_{t+1}) in reply buffer $\mathcal{B}$
    randomly sample minibatch of transitions (s_i, a_i, r_i, s_{i+1}) from $\mathcal{B}$
    Set $y_i = \begin{cases} r_i & \text{terminal} \\ r_i + \gamma \max_{a'} Q_{\omega'}(s_{i+1}, a') & \text{otherwise} \end{cases}$
    minimize $L = \frac{1}{N}\sum_i (y_i - Q_w(s_i, a_i))^2$, and update $Q_\omega(s, a)$
    update $Q_{\omega'}$ after the step interval
  end for
end for



(a) Node=32 (b) Node=36
(c) Node=40 (d) Node=44
(e) Node=48 (f) Node=52

Fig. 3. Processing time in the shard consensus.

Since vehicles are decentralized in shards, each vehicle of a shard is required to maintain a copy of the adaptive sharding strategy model. The dynamic sharding construction is initiated by the leader in each shard to construct and broadcast transactions to followed vehicles to reconstruct new shards. The algorithm complexity analysis is illustrated as follows.



(a) Node=32 (b) Node=36

(c) Node=40 (d) Node=44

(e) Node=48 (f) Node=52

Fig. 4. Processing time in the blockchain consensus when δ = DPoS.

VI. Experimental Evaluations

This section evaluates the performance of the proposed mechanisms. First, we analyze the complexity and security of the proposed mechanisms. Second, we evaluate the performance of the adaptive blockchain-empowered FL framework with reputation-based shard selection scheme. Third, we test the performance of the streamline-based shard transmission mechanism. Finally, we evaluate the performance of the DRL-based sharding construction mechanism.

A. Algorithm Analysis

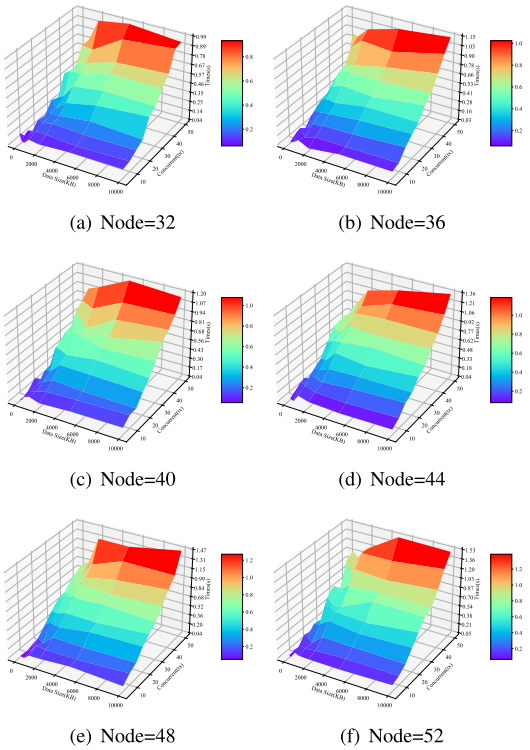
1) Complexity Analysis of Shard Transmission Mechanism: We assume $\mathcal{T}_{\text{train}}$ is the local training time of each vehicle, $\mathcal{T}_{\text{trans}}$ is the time for any vehicle or RSU to transfer a global model, and $\mathcal{T}_{\text{agg}}$ is the time for a model aggregation. A leader vehicle in a shard needs to obtain the global model of the issued training task from RSUs in blockchain and broadcasts it to followed vehicles. The time complexity is $2\mathcal{T}_{\text{trans}}$ in the task release phase. Vehicles train optimal local parameters and share them with the leader vehicle L_s for aggregation. The time complexity is $\mathcal{T}_{\text{train}} + \mathcal{T}_{\text{trans}} + \mathcal{T}_{\text{agg}}$ in the shard training phase. The leader vehicle L_s received optimal aggregated parameters of L_{s-1} from another shard and broadcasts them to follower in the same shard. The time complexity is $2\mathcal{T}_{\text{trans}}$ in the shard transmission phase. After repeating the above process for $\mathcal{G}-1$ times, leader vehicles $\{L_1, \cdots, L_s\}$ submit optimal parameters to RSUs in the blockchain for a final aggregation. The time complexity of the global aggregation phase is $\mathcal{T}_{\text{trans}} + \mathcal{T}_{\text{agg}}$. Therefore, the time complexity of the proposed mechanism is $O(\mathcal{G}(\mathcal{T}_{\text{train}}+3\mathcal{T}_{\text{trans}}+\mathcal{T}_{\text{agg}})-\mathcal{T}_{\text{train}})$. The difference between the proposed mechanism and ChainsFL lies in the shard transmission phase. In ChainsFL, leader vehicles directly submit parameters to RSUs in the blockchain, receive optimal aggregated parameter from blockchain and broadcast them to follower in the same shard. The time complexity of the shard transmission phase of ChainsFL is $3\mathcal{T}_{\text{trans}}$. The above process need to be repeated for $\mathcal{G}$ times. Therefore, the time complexity of ChainsFL is $O(\mathcal{G}(\mathcal{T}_{\text{train}} + 4\mathcal{T}_{\text{trans}} + 2\mathcal{T}_{\text{agg}})$.

2) Security Analysis of Shard Transmission Mechanism: The framework is divided into two parts, including the blockchain and shard parts. The security analysis of the blockchain part refer to [45]. We only focus on the shard part. The first is intra-shard poisoning attacks. Vehicles within a shard may inject malicious training results. The proposed reputation-based shard selection mechanism can resist to those attacks by decreasing reputations of malicious vehicles. The second is poisoning attacks across shards. There may be at least one unsafe shard to corrupt training tasks while transferring aggregated results to other shards. Assuming there are $\mathcal{N}$ vehicles in all shards, $\mathcal{S}$ shards and $\mathcal{V}$ vehicles in each shard. The number of malicious vehicles in a shard does not exceeds $\mathcal{VH}$, where $\mathcal{H}$ is the largest proportion of malicious vehicles in a shard. We use X as a random variable to identify the number of malicious vehicles in a shard. Therefore, according to [37], the probability of a faulty shard should satisfy $P[X \geq \lceil \mathcal{VH} \rceil] \leq 2^{-\lambda}$, where λ is the security parameter to adjust the upper bound of the probability.

3) Complexity Analysis of DRL-Based Sharding Mechanism: The proposed mechanism introduces DQN to automate the construction of vehicular shards. The proposed mechanism maintains an online Q-learning mechanism for vehicular shard construction and dynamic shard updating. It also constructs two Deep Neural Networks (DNNs) with the same structure for synchronized updates in each time interval and real-time updates. We conduct the algorithm complexity analysis referring to [46] to analyse the time complexity of the proposed mechanism. We assume that there are $\mathcal{T}_{\text{layer}}$ number of layers of the DNN, $\mathcal{T}_{\text{neur}}$ number of neurons in the layer, $\mathcal{T}_{\text{iter}}$ number of local iterations and $\mathcal{N}$ number of vehicles in the network. Therefore, the time complexity of training the DNN is $O(\mathcal{N}\mathcal{T}_{\text{iter}} \sum_{i=1}^{\mathcal{T}_{\text{layer}}-1} \mathcal{T}_{\text{neur}}(\mathcal{T}_{\text{neur}} + 1))$. We assume there are $\mathcal{T}_{\text{step}}$ number of steps per training round. The time complexity of the proposed mechanism in the running phase is $O(\mathcal{T}_{\text{step}}\mathcal{N})$. The time complexity in the training phase is high in deep reinforcement learning mechanisms. However, this mechanism is executed in the shard formation phase for once. Once shards are constructed dynamically among vehicles, the high time complexity can not affect shard processing speed, which do not need to spends unacceptable training time.

B. Performance of the Adaptive Blockchain Empowered FL Framework

We simulated a layered sharding network in go1.17 linux/amd64. It is deployed on Ubuntu 16.04.7 LTS, Intel(R) Xeon(R) CPU E5-2620v4@2.10GHz with 8 core, 64G memory and 1000Mb/s. The essence of the architecture



(a) Node=32 (b) Node=36 (c) Node=40 (d) Node=44 (e) Node=48 (f) Node=52

Fig. 5. Processing time in the blockchain consensus when δ = DAG.

proposed by ChainsFL [5], TwoLayer [7] and our proposed method are the combination of shard consensus and blockchain consensus. Therefore, we divide the architecture performance into shard consensus time and blockchain consensus time. ChainsFL adopts the method of DAG [47] and RAFT [14]. TwoLayer adopts the method of PBFT [43]. To better analyze the impact of different consensus algorithms on architecture performance, the consensus adopted in the shard is HOTSTUFF [42], PBFT, RAFT or VRF [44]. HOTSTUFF reduces the communication complexity of consensus linearly as the number of replicas increase. PBFT provides a four-phase state transition mechanism to prevent malicious nodes in the network, which is a Byzantine fault tolerance algorithm. RAFT ensures each node in the network to agree with the same series of state transitions, which is not a Byzantine fault tolerance algorithm. VRF produces a pseudorandom output and a proof that can be verified by anyone in the network. The consensus of the blockchain adopts DAG and DPoS [48]. DAG allows transactions recorded as vertices and connected directly or indirectly. DPoS elects delegates by pooling token in a centralized pool to serve as block producers.

Fig. 3 is the processing time in the shard consensus while concurrent transactions are [10,80,10] and the number of vehicles is [24,52,4] in the network. The orange line represents HOTSTUFF, the red line represents PBFT, the green line represents RAFT and the blue line represents VRF. We can see from the figure that HOTSTUFF and VRF have better performance than PBFT and RAFT. PBFT and RAFT require more communication rounds to reach a consensus between vehicles, while HOTSTUFF and VRF reduce communication complexity to a linear extent as the total number of vehicles increases.

TABLE II
REPUTATION SETTINGS

Variable	Setting
T_{ab}^{disc}, T_{bc}^{disc}	$\Upsilon_{ab}^{disc} = (0.7T_{ab}, 0.7(1-T_{ab}), 0, 0.3)$ $\Upsilon_{bc}^{disc} = (0.7T_{bc}, 0.7(1-T_{bc}), 0, 0.3)$
T_{ab}^{disc}, e_{ab}^{disc}	$\Upsilon_{ab}^{disc} = (T_{ab}(1-e_{ab}), (1-T_{ab})(1-e_{ab}), 0, e_{ab})$ $\Upsilon_{bc}^{disc} = (0.7, 0.1, 0, 0.2)$
T_{bc}^{disc}, e_{bc}^{disc}	$\Upsilon_{ab}^{disc} = (0.7, 0.1, 0, 0.2)$ $\Upsilon_{bc}^{disc} = (T_{bc}(1-e_{bc}), (1-T_{bc})(1-e_{bc}), 0, e_{bc})$
e_{ab}^{disc}, e_{bc}^{disc}	$\Upsilon_{ab}^{disc} = (0.875(1-e_{ab}), 0.125(1-e_{ab}), 0, e_{ab})$ $\Upsilon_{bc}^{disc} = (0.875(1-e_{bc}), 0.125(1-e_{bc}), 0, e_{bc})$
T_{ac}^{para}, T_{bc}^{para}	$\Upsilon_{ac}^{para} = (0.7T_{ac}, 0.7(1-T_{ac}), 0, 0.3)$ $\Upsilon_{bc}^{para} = (0.7T_{bc}, 0.7(1-T_{bc}), 0, 0.3)$
T_{ac}^{para}, e_{ac}^{para}	$\Upsilon_{ac}^{para} = (T_{ac}(1-e_{ac}), (1-T_{ac})(1-e_{ac}), 0, e_{ac})$ $\Upsilon_{bc}^{para} = (0.7, 0.1, 0, 0.2)$
T_{bc}^{para}, e_{bc}^{para}	$\Upsilon_{ac}^{para} = (0.7, 0.1, 0, 0.2)$ $\Upsilon_{bc}^{para} = (T_{bc}(1-e_{bc}), (1-T_{bc})(1-e_{bc}), 0, e_{bc})$
e_{ac}^{para}, e_{bc}^{para}	$\Upsilon_{ac}^{para} = (0.875(1-e_{ac}), 0.125(1-e_{ac}), 0, e_{ac})$ $\Upsilon_{bc}^{para} = (0.875(1-e_{bc}), 0.125(1-e_{bc}), 0, e_{bc})$

Fig. 4 and Fig. 5 are the processing time in the blockchain consensus while concurrent transactions are [10,50,10], data sizes increase from 100KB to 10MB, the number of vehicles is [24,52,4] in the network and the consensus algorithm is DAG and DPoS, respectively. We can see from the figure that as the concurrent transactions and data sizes increase, the processing time in the blockchain consensus is increasing stably when concurrent transactions are up to 40. The sharding network does not affect the blockchain network. When the concurrent transactions are equal with 50, the processing time becomes unstable limited by the consensus algorithm. Therefore, it is not suitable to interact with blockchain network frequently.

Besides, we implemented the proposed scheme in go1.17.1 darwin/amd64. We utilize the expected reputation of a malicious vehicle to illustrate the performance of the reputation-based shard selection scheme MWMVSL, which is compared with TSL [39] and MWSL [40]. We simulated the cases where the malicious vehicle randomly feeds back distrust results with the probability of 0.7, neutral results with the probability of 0.1, and belief results with the probability of 0.2. Since TSL and MWSL are not able to classify the neutral results, we divide neutral results into honest and dishonest cases to compare their expected reputation. TABLE II is the specific reputation settings in discount and parallel operations with $q_{ij} = 0.8$, $a = 0.5$, $\theta = 0.8$, $t_e = 100$min, and $\mathcal{F}_1 = \mathcal{F}_2 = 0.5$, which varies T_{ab}, T_{bc}, T_{ac} and e_{ab}, e_{bc}, e_{ac} from 0 to 1 [17], [40], [41].

Fig. 6 compares MWMVSL with TSL and MWSL to show the expected reputation of a malicious vehicle. The reputation of MWMVSL has a sharper and larger decrease than those in TSL and MWSL which classify neutral results as honest or dishonest ones. When a time slot is equal to 100, reputations tend to be stable and close to the probability of dishonest behaviors since reputation opinions require more operations to get stable expected reputations.

Since there are three variables influencing reputation opinions of shards, we simplify the opinion vector to evaluate the influence of belief and priori uncertainty on discount and parallel operations of inter-shard reputations. According to [41], we can transfer the reputation opinions of shards into a pair variable referring to T and e as

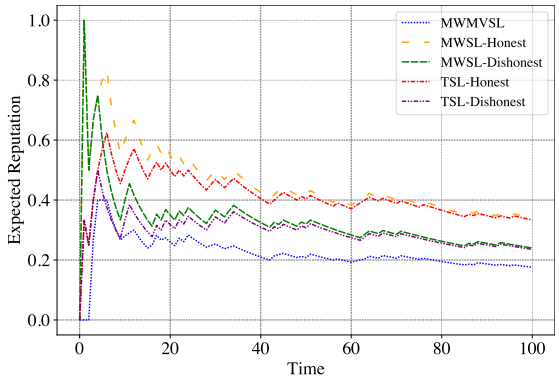
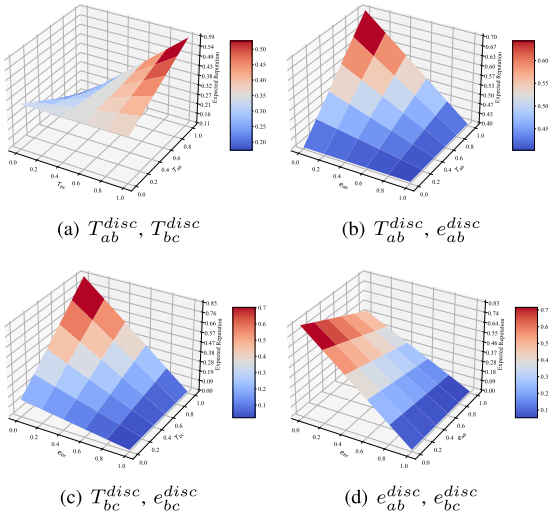


Fig. 6. Expected reputation of a malicious vehicle.



(a) T_{ab}^{disc}, T_{bc}^{disc} (b) T_{ab}^{disc}, e_{ab}^{disc}

(c) T_{bc}^{disc}, e_{bc}^{disc} (d) e_{ab}^{disc}, e_{bc}^{disc}

Fig. 7. Influence on discount reputation.

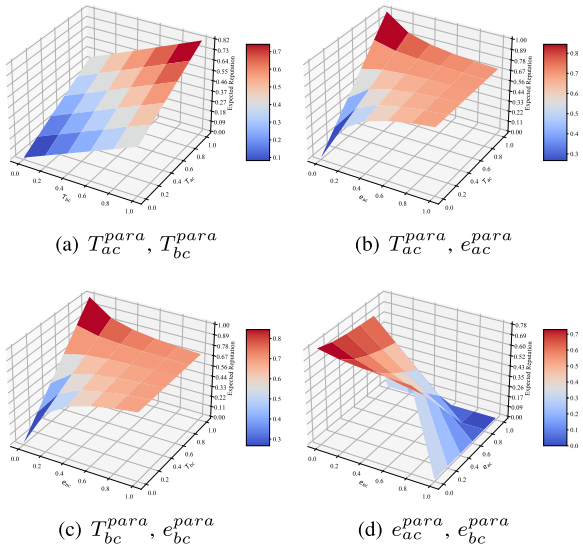
$\Upsilon_{ab}^{t_e} = (T(1-e), (1-T)(1-e), 0, e)$, while T and e represent the average observations of belief events and priori uncertainty of opinion metric, respectively.

For the discount reputation, we obtain the reputation opinions of Υ_{ab}^{disc} and Υ_{bc}^{disc} to infer Υ_{ac}^{disc}. Fig. 7 compares the impact of belief and priori uncertainty on discount reputations. The expected reputation between sequential shards highly depends on T_{ab}^{disc} and e_{bc}^{disc} from Fig. 7, which means that the reputation opinion from shard a on shard c is decided by the belief between shards a and b and priori uncertainty between shards b and c. If the belief between shards a and b is high, and the priori uncertainty between shards b and c is low, the posteriori uncertainty and priori uncertainty will be high, which is benefited to the expected reputation.

For the parallel reputation, we obtain the reputation opinions of Υ_{ac}^{para} and Υ_{bc}^{para} to infer Υ_{ab}^{para}. Fig. 8 compares the influence of belief and priori uncertainty on parallel reputations. The expected reputation between parallel streamlines highly depends on e_{ac}^{para} and e_{bc}^{para} from Fig. 8, which means that the reputation opinion from shard a on shard b is determined by the priori uncertainties of shard a on shard c and shard b on shard c.

C. Performance of the Streamline-Based Shard Transmission Mechanism

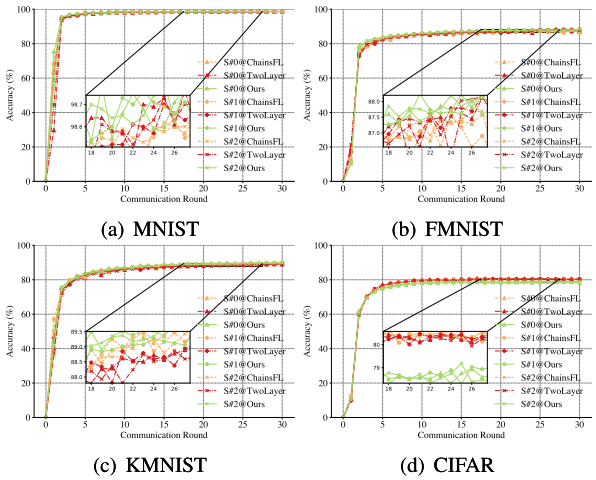
We implemented the proposed approach in PyTorch 1.8.0 and go1.17 linux/amd64. All experiments are performed



(a) T_{ac}^{para}, T_{bc}^{para} (b) T_{ac}^{para}, e_{ac}^{para}

(c) T_{bc}^{para}, e_{bc}^{para} (d) e_{ac}^{para}, e_{bc}^{para}

Fig. 8. Influence on parallel reputation.



(a) MNIST (b) FMNIST

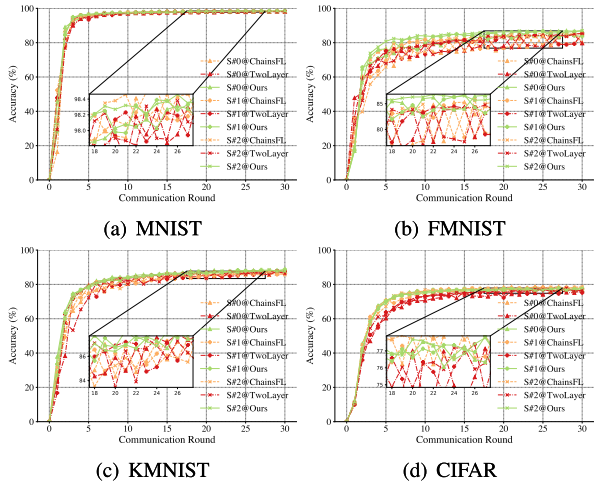
(c) KMNIST (d) CIFAR

Fig. 9. Accuracy of IID datasets.

on a server with CentOS Linux release 7.9.2009 with 6 core, 60G memory, 25000Mb/s and 4 NVIDIA RTX 3090 GPUs. We exploit multiple network ports to simulate the proposed mechanism while the base port is 12080 to avoid port conflicts. The consensus algorithms used in the proposed mechanism are VRF and DAG mentioned in the experiments of the architecture. We set the shard number as 3, the number of vehicles in the shard layer as 12, the number of RSUs in the blockchain layer as 4, the local epoch as 5, and the global round as 30. We compared our mechanism with ChainsFL [5] and TwoLayer [7] to show the difference in the accuracy of shards and the communication overhead of the mechanism.

We use MNIST,[1] KMNIST,[2] Fashion-MNIST (denoted as FMNIST)[3] and CIFAR-10[4] as the primary dataset of the training task. Moreover, we divide those datasets into IID (Independently Identically Distribution) and Non-IID situations. IID is conducted by randomly extracting the same number of samples from training sets and distributing them

[1]http://yann.lecun.com/exdb/mnist/
[2]https://github.com/rois-codh/kmnist
[3]https://github.com/zalandoresearch/fashion-mnist
[4]https://www.cs.toronto.edu/ kriz/cifar.html



(a) MNIST (b) FMNIST

(c) KMNIST (d) CIFAR

Fig. 10. Accuracy of Non-IID datasets.

to vehicles. Non-IID is conducted by sorting training sets, dividing them into multiple slices, and distributing the same number of slices to vehicles. We apply the proposed method on CNN (Convolutional Neural Network)[5] [6] to execute training tasks.

Fig. 9 and Fig. 10 indicate the accuracy of three shards [S#0, S#1, S#2] in ChainsFL, TwoLayer and Ours with datasets MNIST, KMNIST, FMNIST and CIFAR under IID and Non-IID situations, and the consensus mechanisms are VRF and DAG. As seen from those figures, after the training period, ChainsFL, TwoLayer and Ours can bring the model to a similar accuracy in IID and Non-IID situations. Moreover, Ours is inferior to ChainsFL and TwoLayer in CIFAR, while ChainsFL and TwoLayer are inferior to Ours for datasets MNIST, FMNIST and KMNIST. Since ChainsFL and TwoLayer aggregate parameters from all shards, it can reduce the impact of differences among datasets of vehicles. While Ours aggregates parameters from neighbour shards, it can reduce interactions between blockchain and shard almost without compromising the FL accuracy.

TABLE III shows the communication overhead of ChainsFL, TwoLayer and Ours while the datasets and consensus are the same as above. It can be seen from the table that Ours can reduce communication overhead compared with ChainsFL and TwoLayer. Since ChainsFL and TwoLayer require frequent interactions between blockchain and shards, the communication overhead is increased. Ours requires the leader vehicle of each shard to pass parameters to the next shard, and needs blockchain to aggregate parameters from all shards in the final communication round, which reduces high communication overhead.

D. Performance of the DRL-Based Sharding Mechanism

We implemented the proposed approach in Pytorch 1.7.0 and go1.17 linux/amd64. It is deployed on a single server the same as the architecture performance. The experiment environment is based on the gym framework [49] for simulation. TABLE IV is the specific parameter configuration. This paper uses DQN to adjust parameters to accommodate dynamic environments. The policy network of all agents adopts a two-layer fully connected network and the ReLU activation function. The learning rate is set to 0.002, the discount factor is 0.98, the exploration rate is 0.1, the number of DRL training epochs is 10000, the buffer size is 100, the batch size is 64, and the step interval for target network update is 10 [34], [35], [37].

Fig. 11 shows the convergence performance of our proposed mechanism. Two baseline schemes are considered for comparison in the performance analysis. *Proposed mechanism with fixed shards* means the number of shards is set to 16. *Proposed mechanism with fixed size* indicates that the data size is fixed to 8. As seen from the figure, as the number of episodes increases, the rewards increase and reach a stable state after 4000 episodes. Meanwhile, the proposed algorithm

[5]CNN for MNIST, KMNIST and FMNIST is with the following structure: 10 × 5 × 1/20 Convolutional → 2 × 2 MaxPool → 320 × 10 Linear.

[6]CNN for CIFAR-10 is with the following structure: 3 × 3 × 64/128/256/256/512/512/512/512 Convolutional → 2 × 2 MaxPool → 512 × 10 Convolutional.

TABLE III
COMMUNICATION OVERHEAD (S)

Type	Dataset	Ours	ChainsFL	TwoLayer
IID	MNIST	1031.230	1387.859	1199.347 (↓ 14.01%)
	FMNIST	978.361	1314.983	1186.729 (↓ 17.55%)
	KMNIST	982.845	1343.168	1254.282 (↓ 21.64%)
	CIFAR	1194.003	1928.936	1583.551 (↓ 24.59%)
Non-IID	MNIST	984.995	1321.967	1180.599 (↓ 16.56%)
	FMNIST	969.677	1312.089	1173.305 (↓ 17.35%)
	KMNIST	977.183	1341.176	1172.542 (↓ 16.66%)
	CIFAR	1237.831	1952.307	1685.833 (↓ 26.57%)

TABLE IV
DRL SIMULATION PARAMETERS

Parameter	Value
The maximum number of vehicles, $\mathcal{N}$	200
The global communication round, $\mathcal{G}$	30
The message verification time, T_v	0.1s
The message transmission rate, R_t	10-100Mbps
The local training epoch, $epoch$	5
The latency for configuring shards, T_{reco}	0.001s
The training time, T^s_{comp}	20s
Average data size, M_t	1MB
The threshold of T_{inter} to T_{intra}, γ	$\frac{1}{6}$
The maximum data size, $M_{\max}$	16MB
The maximum number of shards, $\mathcal{S}_{\max}$	25

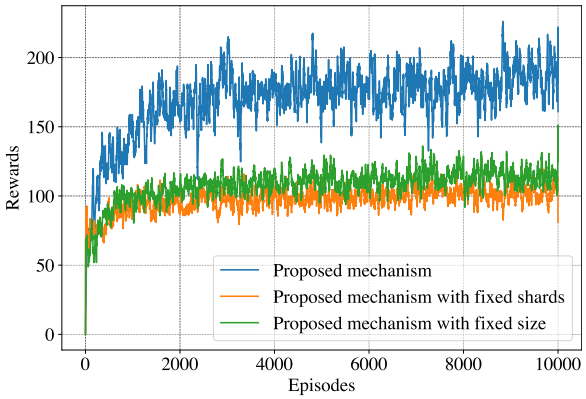


Fig. 11. Convergence performance of different mechanisms.

can obtain higher rewards, which shows the advantage of our proposed framework. Since the compared algorithms are with fixed number of shards and message sizes, their rewards are limited by these factors, which can not release their throughput according to dynamic requirements.

VII. Conclusion and Future Works

The two-layer framework consisting of a mainchain and subchains strongly depends on predefined vehicular blockchain parameters, as well as suffers from huge communication costs and feeds by false data, which limits the throughput, reliability and credibility of the blockchain-enabled FL framework for ITS. In this paper, we propose an adaptive blockchain-enabled FL framework based on blockchain sharding and DRL to obtain vehicular data flow between a blockchain and multiple shards. We also introduce a streamline-driven shard transmission mechanism to improve communication efficiency between shards. Moreover, we develop a reputation-based shard selection mechanism to improve data credibility. Finally, we design an adaptive DRL-based sharding mechanism to enable adaptive scalability. Experimental results show that our framework can be adaptive, reduce communication overhead, and perform more effectively than the baseline mechanisms. We plan to consider intelligent dynamic collaboration among vehicles in a cross-chain environment in future work.

References

[1] S. Nakamoto, "Bitcoin: A peer-to-peer electronic cash system," Decentralized Bus. Rev., White Paper, Oct. 2008.

[2] G. Wood, "Ethereum: A secure decentralised generalised transaction ledger," *Ethereum Project Yellow Paper*, vol. 151, pp. 1–32, Apr. 2014.

[3] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. Y. Arcas, "Communication-efficient learning of deep networks from decentralized data,," in *Proc. Artif. Intell. Statist.*, 2017, pp. 1273–1282.

[4] X. Wang, X. Ren, C. Qiu, Z. Xiong, H. Yao, and V. C. M. Leung, "Integrating edge intelligence and blockchain: What, why, and how," *IEEE Commun. Surveys Tuts.*, vol. 24, no. 4, pp. 2193–2229, 4th Quart., 2022.

[5] S. Yuan, B. Cao, M. Peng, and Y. Sun, "ChainsFL: Blockchain-driven federated learning from design to realization," in *Proc. IEEE Wireless Commun. Netw. Conf. (WCNC)*, Mar. 2021, pp. 1–6.

[6] S. Yuan, B. Cao, Y. Sun, Z. Wan, and M. Peng, "Secure and efficient federated learning through layering and sharding blockchain," 2021, *arXiv:2104.13130*.

[7] L. Feng, Z. Yang, S. Guo, X. Qiu, W. Li, and P. Yu, "Two-layered blockchain architecture for federated learning over the mobile edge network," *IEEE Netw.*, vol. 36, no. 1, pp. 45–51, Jan. 2022.

[8] H. Chai, S. Leng, Y. Chen, and K. Zhang, "A hierarchical blockchain-enabled federated learning algorithm for knowledge sharing in Internet of Vehicles," *IEEE Trans. Intell. Transp. Syst.*, vol. 22, no. 7, pp. 3975–3986, Jul. 2021.

[9] X. Xu, S. Duan, J. Zhang, Y. Luo, and D. Zhang, "Optimizing federated learning on device heterogeneity with a sampling strategy," in *Proc. IEEE/ACM 29th Int. Symp. Quality Service (IWQOS)*, Jun. 2021, pp. 1–10.

[10] Q. Xia, Z. Tao, Z. Hao, and Q. Li, "FABA: An algorithm for fast aggregation against Byzantine attacks in distributed neural networks," in *Proc. IJCAI*, 2019, pp. 1–7. [Online]. Available: https://par.nsf.gov/biblio/10119268

[11] S. Warnat-Herresthal et al., "Swarm learning for decentralized and confidential clinical machine learning,," *Nature*, vol. 594, no. 7862, pp. 265–270, 2021.

[12] S. Zhou, H. Huang, W. Chen, P. Zhou, Z. Zheng, and S. Guo, "PIRATE: A blockchain-based secure framework of distributed machine learning in 5G networks," *IEEE Netw.*, vol. 34, no. 6, pp. 84–91, Nov. 2020.

[13] H. Jin, X. Dai, J. Xiao, B. Li, H. Li, and Y. Zhang, "Cross-cluster federated learning and blockchain for Internet of medical things," *IEEE Internet Things J.*, vol. 8, no. 21, pp. 15776–15784, Nov. 2021.

[14] D. Ongaro and J. Ousterhout, "In search of an understandable consensus algorithm,," in *Proc. USENIX Annu. Tech. Conf. (USENIX ATC)*, 2014, pp. 305–319.

[15] B. Cao et al., "Performance analysis and comparison of PoW, PoS and DAG based blockchains," *Digit. Commun. Netw.*, vol. 6, no. 4, pp. 480–485, Nov. 2020.

[16] X. Huang, R. Yu, J. Kang, and Y. Zhang, "Distributed reputation management for secure and efficient vehicular edge computing and networks," *IEEE Access*, vol. 5, pp. 25408–25420, 2017.

[17] J. Kang, Z. Xiong, D. Niyato, S. Xie, and J. Zhang, "Incentive mechanism for reliable federated learning: A joint optimization approach to combining reputation and contract theory," *IEEE Internet Things J.*, vol. 6, no. 6, pp. 10700–10714, Dec. 2019.

[18] J. Kang, Z. Xiong, D. Niyato, Y. Zou, Y. Zhang, and M. Guizani, "Reliable federated learning for mobile networks," *IEEE Wireless Commun.*, vol. 27, no. 2, pp. 72–80, Apr. 2020.

[19] J. Kang et al., "Blockchain for secure and efficient data sharing in vehicular edge computing and networks," *IEEE Internet Things J.*, vol. 6, no. 3, pp. 4660–4670, Jun. 2019.

[20] J. Zhang, Y. Huang, F. Ye, and Y. Yang, "A novel proof-of-reputation consensus for storage allocation in edge blockchain systems," in *Proc. IEEE/ACM 29th Int. Symp. Quality Service (IWQOS)*, Jun. 2021, pp. 1–10.

[21] N. Gao, R. Huo, S. Wang, T. Huang, and Y. Liu, "Sharding-hashgraph: A high-performance blockchain-based framework for industrial Internet of Things with hashgraph mechanism," *IEEE Internet Things J.*, vol. 9, no. 18, pp. 17070–17079, Sep. 2022.

[22] H. Moudoud, S. Cherkaoui, and L. Khoukhi, "Towards a secure and reliable federated learning using blockchain," in *Proc. IEEE Global Commun. Conf. (GLOBECOM)*, Dec. 2021, pp. 1–6.

[23] J. C. Corbett, "Spanner: Google's globally distributed database," *ACM Trans. Comput. Syst. (TOCS)*, vol. 31, no. 3, pp. 1–22, 2013.

[24] L. Luu, V. Narayanan, C. Zheng, K. Baweja, S. Gilbert, and P. Saxena, "A secure sharding protocol for open blockchains," in *Proc. ACM SIGSAC Conf. Comput. Commun. Secur.*, Oct. 2016, pp. 17–30.

[25] E. Kokoris-Kogias, P. Jovanovic, L. Gasser, N. Gailly, E. Syta, and B. Ford, "OmniLedger: A secure, scale-out, decentralized ledger via sharding," in *Proc. IEEE Symp. Secur. Privacy (SP)*, May 2018, pp. 583–598.

[26] M. Zamani, M. Movahedi, and M. Raykova, "RapidChain: Scaling blockchain via full sharding," in *Proc. ACM SIGSAC Conf. Comput. Commun. Secur.*, Oct. 2018, pp. 931–948.

[27] M. Al-Bassam, A. Sonnino, S. Bano, D. Hrycyszyn, and G. Danezis, "Chainspace: A sharded smart contracts platform," 2017, *arXiv:1708.03778*.

[28] J. Wang and H. Wang, "Monoxide: Scale out blockchains with asynchronous consensus zones,," in *Proc. 16th USENIX Symp. Netw. Syst. Design Implement. (NSDI)*, 2019, pp. 95–112.

[29] H. Dang, T. T. A. Dinh, D. Loghin, E.-C. Chang, Q. Lin, and B. C. Ooi, "Towards scaling blockchain systems via sharding," in *Proc. Int. Conf. Manage. Data*, Jun. 2019, pp. 123–140.

[30] Z. Hong, S. Guo, P. Li, and W. Chen, "Pyramid: A layered sharding blockchain system," in *Proc. IEEE INFOCOM Conf. Comput. Commun.*, May 2021, pp. 1–10.

[31] H. Huang et al., "Elastic resource allocation against imbalanced transaction assignments in sharding-based permissioned blockchains," *IEEE Trans. Parallel Distrib. Syst.*, vol. 33, no. 10, pp. 2372–2385, Oct. 2022.

[32] V. Mnih et al., "Human-level control through deep reinforcement learning," *Nature*, vol. 518, pp. 529–533, Feb. 2015. [Online]. Available: http://www.nature.com/articles/nature14236.

[33] Q. Yang and P. Li, "Deep reinforcement learning based energy scheduling for edge computing," in *Proc. IEEE Int. Conf. Smart Cloud (SmartCloud)*, Nov. 2020, pp. 175–180.

[34] M. Liu, F. R. Yu, Y. Teng, V. C. M. Leung, and M. Song, "Performance optimization for blockchain-enabled industrial Internet of Things (IIoT) systems: A deep reinforcement learning approach," *IEEE Trans. Ind. Informat.*, vol. 15, no. 6, pp. 3559–3570, Jun. 2019.

[35] J. Yun, Y. Goh, and J. Chung, "DQN-based optimization framework for secure sharded blockchain systems," *IEEE Internet Things J.*, vol. 8, no. 2, pp. 708–722, Jan. 2021.

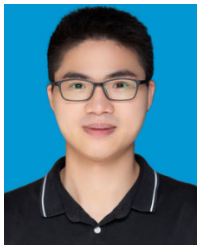
[36] Y. Lu, X. Huang, K. Zhang, S. Maharjan, and Y. Zhang, "Blockchain empowered asynchronous federated learning for secure data sharing in Internet of vehicles," *IEEE Trans. Veh. Technol.*, vol. 69, no. 4, pp. 4298–4311, Apr. 2020.

[37] J. Zhang, Z. Hong, X. Qiu, Y. Zhan, S. Guo, and W. Chen, "SkyChain: A deep reinforcement learning-empowered dynamic blockchain sharding system," in *Proc. 49th Int. Conf. Parallel Process. (ICPP)*, Aug. 2020, pp. 1–11.
[38] T. P. Lillicrap et al., "Continuous control with deep reinforcement learning," 2015, *arXiv:1509.02971*.
[39] A. Jøsang, "A logic for uncertain probabilities," *Int. J. Uncertainty, Fuzziness Knowledge-Based Syst.*, vol. 9, no. 3, pp. 279–311, Jun. 2001.
[40] M. Sohail, L. Wang, S. Jiang, S. Zaineldeen, and R. U. Ashraf, "Multi-hop interpersonal trust assessment in vehicular ad-hoc networks using three-valued subjective logic," *IET Inf. Secur.*, vol. 13, no. 3, pp. 223–230, May 2019.
[41] G. Liu, Q. Yang, H. Wang, X. Lin, and M. P. Wittie, "Assessment of multi-hop interpersonal trust in social networks by three-valued subjective logic," in *Proc. IEEE INFOCOM Conf. Comput. Commun.*, Apr. 2014, pp. 1698–1706.
[42] M. Yin, D. Malkhi, M. K. Reiter, G. G. Gueta, and I. Abraham, "HotStuff: BFT consensus with linearity and responsiveness," in *Proc. ACM Symp. Princ. Distrib. Comput.*, Jul. 2019, pp. 347–356.
[43] M. Castro and B. Liskov, "Practical Byzantine fault tolerance," in *Proc. OSDI*, vol. 99, Feb. 1999, pp. 173–186.
[44] Y. Gilad, R. Hemo, S. Micali, G. Vlachos, and N. Zeldovich, "Algorand: Scaling Byzantine agreements for cryptocurrencies," in *Proc. 26th Symp. Operating Syst. Princ.*, Oct. 2017, pp. 51–68.
[45] W. Li, C. Feng, L. Zhang, H. Xu, B. Cao, and M. A. Imran, "A scalable multi-layer PBFT consensus for blockchain," *IEEE Trans. Parallel Distrib. Syst.*, vol. 32, no. 5, pp. 1146–1160, May 2021.
[46] Y. Ren, X. Chen, S. Guo, S. Guo, and A. Xiong, "Blockchain-based VEC network trust management: A DRL algorithm for vehicular service offloading and migration," *IEEE Trans. Veh. Technol.*, vol. 70, no. 8, pp. 8148–8160, Aug. 2021.
[47] S. Popov, "The tangle," IOTA Found., Berlin, Germany, White Paper 3.02, 2018, vol. 1, no. 3. [Online]. Available: https://www.iota.org/connect/contact
[48] D. Larimer, "Delegated proof-of-stake (DPoS)," *Bitshare Whitepaper*, vol. 81, p. 85, Jun. 2014.
[49] G. Brockman et al., "OpenAI gym," OpenAI, San Francisco, CA, USA, Tech. Rep., 2016.

Yijing Lin (Graduate Student Member, IEEE) received the bachelor's degree from the North China Electric Power University (NCEPU), Beijing, China. She is currently pursuing the Ph.D. degree with the State Key Laboratory of Networking and Switching Technology, Beijing University of Posts and Telecommunications (BUPT), Beijing. Her current research interests include blockchain and edge computing.

Zhipeng Gao (Member, IEEE) received the Ph.D. degree from the Beijing University of Posts and Telecommunications (BUPT), Beijing, China, in 2007. He is currently a Professor with the State Key Laboratory of Networking and Switching Technology, BUPT. He presides over a series of key research projects on network and service management, including the projects supported by the National Natural Science Foundation and the National High-Tech Research and Development Program of China. His research interests include edge computing and blockchain.

Hongyang Du (Graduate Student Member, IEEE) received the B.Sc. degree from Beijing Jiaotong University, Beijing, China, in 2021. He is currently pursuing the Ph.D. degree with the School of Computer Science and Engineering, Energy Research Institute@NTU, Nanyang Technological University, Singapore, under the Interdisciplinary Graduate Program. His research interests include semantic communications, reconfigurable intelligent surfaces, and communication theory. He was a recipient of the IEEE Daniel E. Noble Fellowship Award in 2022. He was recognized as an Exemplary Reviewer of the IEEE TRANSACTIONS ON COMMUNICATIONS in 2021.



Jiawen Kang received the M.S. and Ph.D. degrees from the Guangdong University of Technology, Guangzhou, China, in 2015 and 2018, respectively. From 2018 to 2021, he was a Post-Doctoral Fellow with Nanyang Technological University, Singapore. He is currently a Full Professor with the Guangdong University of Technology. His research interests include blockchain, security, and privacy protection in wireless communications and networking.

Dusit Niyato (Fellow, IEEE) received the B.Eng. degree from the King Mongkut's Institute of Technology Ladkrabang (KMITL), Thailand, in 1999, and the Ph.D. degree in electrical and computer engineering from the University of Manitoba, Canada, in 2008. He is currently a Professor with the School of Computer Science and Engineering, Nanyang Technological University, Singapore. His research interests include sustainability, edge intelligence, decentralized machine learning, and incentive mechanism design.

Qian Wang received the Ph.D. degree from the Beijing University of Posts and Telecommunications (BUPT), Beijing, China, in 2020. She is currently a Teacher with the Beijing University of Technology. Her current research interests include wireless networks and opportunistic communications.

Jingqing Ruan received the bachelor's degree from North China Electric Power University (NCEPU), Beijing, China. She is currently pursuing the Ph.D. degree with the Institute of Automation, Chinese Academy of Sciences, Beijing. Her research interests include deep reinforcement learning.

Shaohua Wan received the Ph.D. degree from the School of Computer, Wuhan University, in 2010. From 2016 to 2017, he was a Visiting Professor with the Department of Electrical and Computer Engineering, Technical University of Munich, Germany. He is currently a Professor with the Shenzhen Institute for Advanced Study, University of Electronic Science and Technology of China. He is the author of over 150 peer-reviewed research articles and books, including over 40 IEEE/ACM TRANSACTIONS articles, such as IEEE TRANSACTIONS ON INDUSTRIAL INFORMATICS, IEEE TRANSACTIONS ON INTELLIGENT TRANSPORTATION SYSTEMS, *ACM Transactions on Internet Technology* (TOIT), IEEE TRANSACTIONS ON NETWORK SCIENCE AND ENGINEERING, IEEE TRANSACTIONS ON MULTIMEDIA, IEEE TRANSACTIONS ON COMPUTATIONAL SOCIAL SYSTEMS, *ACM Transactions on Multimedia Computing, Communications, and Applications* (TOMM), IEEE TRANSACTIONS ON EMERGING TOPICS IN COMPUTATIONAL INTELLIGENCE, and *Pattern Recognition*, and many top conference papers in the fields of edge intelligence. His main research interests include deep learning for the Internet of Things.