PoShapley-BCFL: a Fair and Robust Decentralized Federated Learning Based on Blockchain and the Proof of Shapley-Value

Ziwen Cheng, Yi Liu, Chao Wu, Yongqi Pan, Liushun Zhao and Cheng Zhu

October 23, 2023

PoShapley-BCFL: A fair and robust decentralized federated learning based on blockchain and the proof of Shapley-value*

Ziwen Cheng[1][0000−0003−3339−9903], Yi Liu[1], Chao Wu[2], Yongqi Pan[1], Liushun Zhao[3], and Cheng Zhu[1]

[1] National University of Defense Technology, Changsha China
[2] Zhejiang University, Hangzhou China
[3] Xidian University, Xian China

Abstract. Recently, blockchain-based Federated learning(BCFL) has emerged as a promising technology for promoting data sharing in the Internet of Things(IoT) without relying on a central authority, while ensuring data privacy, security, and traceability. However, it remains challenging to design an decentralized and appropriate incentive scheme that should promise a fair and efficient contribution evaluation for participants while defending against low-quality data attacks. Although Shapley-Value(SV) methods have been widely adopted in FL due to their ability to quantify individuals' contributions, they rely on a central server for calculation and incur high computational costs, making it impractical for decentralized and large-scale BCFL scenarios. In this paper, we designed and evaluated PoShapley-BCFL, a new blockchain-based FL approach to accommodate both contribution evaluation and defense against inferior data attacks. Specifically, we proposed PoShapley, a Shapley-value-enabled blockchain consensus protocol tailored to support a fair and efficient contribution assessment in PoShapley-BCFL. It mimics the Proof-of-Work mechanism that allows all participants to compute contributions in parallel based on an improved lightweight SV approach. Following using the PoShapley protocol, we further designed a fair-robust aggregation rule to improve the robustness of PoShapley-BCFL when facing inferior data attacks. Extensive experimental results validate the accuracy and efficiency of PoShapley in terms of distance and time cost, and also demonstrate the robustness of our designed PoShapley-BCFL.

Keywords: Blockchain · Proof of Shapley · Federated learning · Contribution Evaluation.

* Supported by the Strategic Priority Research Program of the Chinese Academy of Sciences (XDA19090105), National Key Research and Development Project of China (2021ZD0110505), National Natural Science Foundation of China (U19B2042), the Zhejiang Provincial Key Research and Development Project (2023C01043 and 2022C03106), the University Synergy Innovation Program of Anhui Province (GXXT-2021-004), Academy Of Social Governance Zhejiang University, Fundamental Research Funds for the Central Universities (226-2022-00064), Hunan Province Graduate Innovation Project Fund (Grant No.CX20200042).

1 Introduction

Nowadays, the proliferation of the Internet of Things(IoT) has led to massive data being generated from various sources. As an emerging distributed learning paradigm, federated learning(FL)[30] has been regarded as a promising solution to promote these data sharing and collaboratively training. However, the conflicts between its centralized framework and the increasing scalability of IoT seriously impede its applications in data sharing. Under this situation, the advent of blockchain-based federated learning(BCFL)[4] has ameliorated the shortcomings. Blockchain emerges as a decentralized ledger technology with the potential to revolutionize the distributed learning paradigm from a centralized design to a decentralized point-to-point collaboration paradigm[29, 8]. Specifically, BCFL works on a P2P communication network, removing the need for centralized servers[10]. In such settings, FL participants are customarily treated as equal blockchain nodes with extensive functions, such as performing local training, recording related transactions, and then making the leader who wins the consensus competition complete the aggregation steps[17]. In this way, BCFL mitigates the concerns about a single point of failure and scalability[9] while also enhancing the protection for data privacy, ownership, and security[14]. To maintain these advantages of BCFL in the long term and better serve IoT data sharing, an efficient, fair and robust incentive mechanism which could always attract participants with high-quality data is critical. Since BCFL-enabled IoT data-sharing tasks are performed on specific tasks in complex networks without inspecting the original data, one of the most direct and effective incentive approaches is to evaluate participants' performance in the global model without third parties' assistance and reward them accordingly. Unfortunately, it is still absent.

The Shapley Value(SV) method[1, 28] has received much attention due to abilities to quantify the contribution of individuals within a group under the Cooperative Game Theory. It calculates the marginal contributions of each participant in all possible subset consortiums to which it belongs and assigns a weighted sum of marginal contributions as total contribution value to each participant[21], thus, ensuring fairness. This method is also commonly used in FL to evaluate model utility[16, 18]. However, the original calculation procedures of SV often incur exponential time concerning the number of participants n, which is not always suited to practical scenarios involving tremendous FL participants, let alone performing a contribution evaluation for participants under the decentralized settings of BCFL. Fairness cannot be guaranteed if the participants upload their self-contribution evaluation results because of self-interest assumptions.

Given the aforementioned dilemmas, a feasible solution for individual performance-based BCFL contribution evaluation is that participants jointly and simultaneously run the calculation of a lightweight Shapley value and mutually oversee without central servers. Fortunately, this idea naturally coincides with the role of blockchain in enabling trusted collaboration with a trusted central authority omitting. More importantly, the consensus mechanism, as one of the critical components of blockchain, defines how different participants collaboratively work to maintain the blockchain networks[2]. With this motivation in mind, we have comprehensively considered combining the blockchain consensus mechanism and the Shapley value, and proposed PoShapley-BCFL, a novel decentralized FL framework with fair and robust contribution evaluation-based incentive designs. Our contributions can be summarized as follows:

- We first developed PoShapley-BCFL, a new combination of federated learning and blockchain technology that guarantees a fair and robust learning process. Our approach achieves this goal by extending a blockchain-consensus-enabled SV calculation procedure and an SV-enabled aggregation procedure into the typical FL framework.
- We proposed a Shapley-value-enabled blockchain protocol, PoShapley, tailored to support the assessment of contributions in decentralized federated learning processes. Our proposed protocol mimics the Proof-of-Work mechanism and allows all participants to compute a monte-carlo-sampling enabled lightweight Shapley value algorithm in parallel until an agreement is achieved, resulting in a more efficient, trustworthy and fair evaluation process.
- Following using the PoShapley protocol, we also developed the fair-robust aggregation method. This method includes a smart-contract-driven client selection process and a Shapley-value-based aggregation process. Specifically, we assigned weights to selected clients based on the ratio of their shapley values, which automatically differentiate low-quality participants and improve model performance when facing inferior data attacks.

The remainder of this paper is organized as follows. Section 2 provided related work and preliminaries. We proposed our PoShapley-BCFL in Section 3 with the detailed descriptions of the PoShapley consensus protocol and the SV-based aggregation method. After that, we move to experiments in Section 4 to demonstrate the performance of our work. Finally, conclusions are summarized in Section 5.

2 Related Work and Preliminaries

2.1 Related work

Incentive mechanism in BCFL. Existing incentive approaches in BCFL can be broadly categorized into three types: game-based method[20, 14, 26, 27], auction-based method[13, 11, 5], and reputation-based method[12, 3]. Game-based incentives focus on maximizing FL participants' utilities based on Stackelberg games[14], contract-based games[26] or Bayesian game[27, 20]. Auction-based incentives usually reward FL participants with aims of keeping individual rationality and incentive compatibility[13, 11], which are usually adopted in FL-enabled data trading systems[5]. Reputation mechanism was introduced in [12] and [3] to promote honest participation in BCFL to earn higher reputation value in blockchain networks. Generally speaking, these value-driven schemes rely on sophisticated utility functions, pricing strategies, or reputation models, which are important in motivating honest participants, ensuring fair compensation, and preventing malicious attacks. However, they often overlook the evaluation of the model itself and typically have high complexity, making them difficult to apply to large-scale and dynamic IoT environments.

Shapley-value-based incentive mechanism in FL. Recently, Shapley-value-based contribution evaluation stems from cooperative game theory and has been the focus of research due to its remarkable features of fairness[16]. However, the original SV calculation incurs high computational costs, making it challenging to implement in practice. Various approaches were proposed to reduce the time complexity in FL, including methods that aim to decrease the number of permutations sampling for SV calculation, such as Monte-Carlo(MC) sampling-enabled SV methods[6, 24]. Other techniques involve using coalition models to minimize individual redundant re-executions, as demonstrated by the Group-SV protocol[18], or training FL sub-models instead of starting from scratch, as in Truncation Gradient Shapley[16, 25]. In some works investigating SV methods in BCFL, the paper [15] designs a PoSap protocol to properly reward coins to data owners. The work [22] introduced three Shapley-value-based revenue distribution models for blockchain-enabled data sharing. However, these works did not provide implementations, and thus the feasibility of the proposed scheme is not clear. To this end, our research expanded on the findings of existing works and proposed a consensus mechanism that uses proof of Shapley value to optimize fair and robust decentralized FL. Besides, we provide implementations and experiments to illustrate the feasibility and performances of our work.

2.2 Preliminaries

This paper considers the common Horizontal Federated Learning framework, in which FL members with different samples share the same feature space. For the convenience of presentation, we consider a collaborative learning task with N data owners(i.e., FL participants), each with a private local dataset $\mathcal{D}_i$. During each round t, each participant i downloads the global model w^t and trains on local dataset $\mathcal{D}_i$ for multiple local epochs to get a local model w_i^{t+1}. Then, the local updates and global aggregation can be performed as follows:

$$\begin{aligned} \varDelta_i^{t+1} &= w^t - w_i^{t+1}. \\ w^{t+1} &= w^t + \sum_{i=1}^{N} \frac{|\mathcal{D}_i|}{\sum_{i=1}^{N} |\mathcal{D}_i|} \varDelta_i^{t+1}. \end{aligned} \tag{1}$$

The original Shapley value is a solution concept from cooperative game theory, which can be defined as:

$$\phi_i = \frac{1}{N} \sum_{S \subseteq I \setminus \{i\}} \frac{1}{\binom{N-1}{|S|}} \left[U \left(S \cup \{i\} \right) - U \left(S \right) \right]. \tag{2}$$

where S denotes the subset of participants from N, $U(\cdot)$ is the utility function, which can be assumed as any form in FL settings, such as accuracy, loss and F1 scores.

3 The Algorithm design for PoShapley-BCFL

In this section, we proposed a novel blockchain-based serverless federated learning named PoShapley-BCFL. It is expected to effectively complete the contribution evaluation of all participants during the iteration of decentralized collaborative training while also being able to prevent attacks from inferior data sources. We reorganized the entire process of a typical FL and divided PoShapley-BCFL into six procedures. Fig. 1 explains the interactions among these procedures, and Algorithm 1 demonstrates the pseudo-codes of each procedure to reveal the details. Initially, the data requester releases some parameters as inputs for PoShapley-BCFL, including an initialized global w^0, evaluation function $U(w)$, number of FL participants N, total training round T and Mining-success criteria ρ(i.e., the error threshold for consensus judgement). After successfully recruiting N participants, PoShapley-BCFL begins to operate.

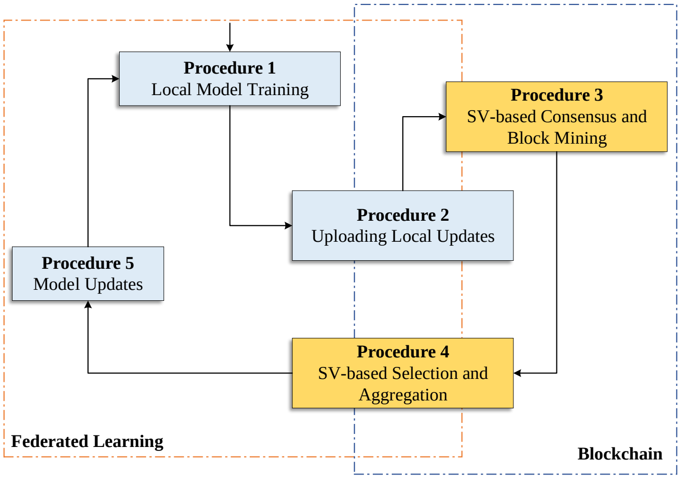


Fig. 1: The modules redesign of proposed PoShapley-BCFL

In Algorithm 1, Lines 2-8 show the local model training process in **Procedure 1** and uploading process of local model updates in **Procedure 2**. Specifically, in each round t, every FL participant i independently obtains a global model w^t and trains it based on local dataset $\mathcal{D}_i$. After some local training iterations, the local model updates Δ_i^{t+1} are generated and transmitted to blockchain as transactions $Tx_{t+1,i}$, in which the pair of Δ_i^{t+1} and $hash\left(\Delta_i^{t+1}\right)$(Line 6) can

```
Algorithm 1: PoShalpey-BCFL Algorithm
input  : initial FL model $w^0$, evaluation function $U(w)$ , $N$ FL participants,
         Total training round $T$, Mining-success criteria $\rho$
output: final FL model $w^T$, SVs for all rounds for $N$ participants
1  for each round $t = 0, 1, 2, \cdots, T - 1$ do
2      for each participant $i$ in parallel do
3          Procedure Local model training($\mathcal{D}_i$)
4              $\Delta_i^{t+1} \leftarrow$ Local Training $\left(\mathcal{D}_i; w^t\right)$
5          Procedure Upload local updates($i, \Delta_i^{t+1}$)
6              $Tx_{t+1,i} = \left\{\Delta_i^{t+1}, hash\left(\Delta_i^{t+1}\right), ID, timestamp\right\}_{Sig_i}$
7              $Tx_{t+1,i} \rightarrow$ upload to BC
8      end
9      $SV^t = \{0, 0, \cdots, 0\} \leftarrow$ initialize SV value list at $t$
10     while Mining-success criteria $\rho$ not met do
11         Procedure SV-based consensus and Block mining($w^t, SV^t$)
12         for each participant $k$ in parallel do
13             $\left\{\mathbb{B}_t, SV^t, M_t\right\}$=PoShapley($w^t, U(w), \left\{\Delta_i^{t+1}\right\}, SV^t$)
14         end
15     end
16     Procedure SV-based Selection and Aggregation($SV^t$)
17     for $M_t$ do
18         $S_a^t \leftarrow$ SV-based-selection-SC($SV^t, key_{M_t}$)
19         $\Delta^{t+1}$= SV-based Aggregation($\left\{\Delta_i^{t+1}\right\}, SV^t$)
20         broadcast to BC and to all participants
21     end
22     for each participant $i$ in parallel do
23         Procedure Models Updates($SV^t$)
24         $w^{t+1} = w^t - \eta\Delta^{t+1}$
25     end
26 end
```

ensure no tampering during transmission. Then, after all participants finish procedure 2, a smart contract deployed on the blockchain triggers the release of an SV list with an initial value of zeros(Line 9), which drives the running of **Procedure 3**. In Lines 10-15, the SV-based consensus procedure begins execution. In this procedure, every FL participant continues to perform the *PoShapley* algorithm(see details in 3.1) to compute each participant's contributions in this training round as long as mining-success criteria ρ is satisfied. After that, the SV list, one of the outputs of *PoShapley* algorithm, is further used in **Procedure 4**(Line 16-20). A winner of the PoShapley competition at round t adapts a fair and robust SV-based aggregation approach(see details in 3.2) to obtain new global model updates, which are then broadcasted to all participants for next training round. Lines 22-25 indicate that every participant performs **Procedure 5** to update a new global model and then restart **Procedure 1**.

3.1 The designs of PoShapley Algorithm

Given the significance of efficiency and attack resistance in contribution evaluation for BCFL systems, we proposed a novel blockchain consensus mechanism named PoShapley. This tailored mechanism facilitates an efficient, fair and robust SV calculation in BCFL, where no central trusted authority exists to evaluate SV utility. The basic concept underlying PoShapley mimics that of PoW[19], which replaces meaningless mathematical puzzles with an improved lightweight SV calculation problem. We present the pseudo-code in Algorithm 2. The initial model w^t represents the initialized global model at training round $t+1$, which also serves as a benchmark for evaluating the training performance. The utility function $U(w)$ can have multiple forms, including accuracy, loss, recall rate, and F1. An illustration of a completed PoShapley loop is presented below.

Before entering the iterative calculation, a participant k should first perform the preparations according to Line 2-4, such as initializes the SV calculation times as $m_k = 1$, computes the utility value of the global model w^t as $v_0^{m_k}$(i.e., $v_0^{m_k} = U(w^t)$), constructs an initialized permutation of received model as L_t, and initializes an SV list as ϕ with all values of 0. Next, at each iterative times m_k , the participant k performs a Monte Carlo sampling[16] on permutation L_t to build a list $\pi^k_{m_k}$. By scanning through the $\pi^k_{m_k}$ from the first entity to the last, the jth FL participant's marginal model contribution can be estimated by participant k following the principle of (3), and then be accumulated into the average Shapley value $V^{t,k}_{m_k}$. We show the calculation steps after disassembling equation (3) in Lines 6-12.

$$
\begin{aligned}
\varDelta v_j &= \mathbb{E}\left[U\left(s \cup \left\{m_k^k\left[j\right]\right\}\right) - U\left(s\right)\right] \\
&= \mathbb{E}\left[U\left(w^t + \sum_{p \in s \cup \left\{m_k^k[j]\right\}} \frac{|\mathcal{D}_p|}{\sum_{p \in s \cup \left\{m_k^k[j]\right\}} |\mathcal{D}_p|} \varDelta_p^{t+1}\right)\right. \\
&\left. -U\left(w^t + \sum_{p \in s} \frac{|\mathcal{D}_p|}{\sum_{p \in s} |\mathcal{D}_p|} \varDelta_p^{t+1}\right)\right].
\end{aligned} \tag{3}
$$

Here, $s = m_k^k\left[1:(j-1)\right]$. And according to index order within SV^t, $V^{t,k}_{m_k}$ is then reordered and denoted as $sv^{t,k}_{m_k}$(Line 13-14). The $sv^{t,k}_{m_k}$ is then used as input for invoking a *Judgement-Smart-Contract*(Line 15), which provides a global signal J_k that indicates whether the loop should be terminated. The automatic judgment operations of this contract are illustrated in Algorithm 3, where the maximum distance between $sv^{t,k}_{m_k}$ and SV^t is calculated. Notably, SV^t refers to the latest average value of all participants' estimated SV, which is updated through smart contracts deployed in blockchain systems in advance(referred in Line 26). When *Judgement-Smart-Contract* returns a $True$ value, that is the maximal distance between the average estimated SV and the estimated SV from participant k is no greater than threshold ρ, the participant k becomes a candidate responsible for generating a new block $\mathbb{B}_{k,t}$ and broadcasting it(Line 16-18).

Algorithm 2: PoShapley Algorithm

```
input  : initial FL model $w^t$, evaluation function $U(w)$ , participants' model
         updates $\left\{\mathit{\Delta}_i^{t+1}, \cdots, \mathit{\Delta}_n^{t+1}\right\}$, initial $SV^t$ for all participants,
         Mining-success criteria $\rho$
output: $SV^t = \left\{\phi_i^{t+1}, \cdots, \phi_n^{t+1}\right\}$ for all participants, new block $\mathbb{B}_t$, the
         winner $M_t$
for each participant $k$ in parallel do
    initialize
    $m_k = 1$; $v_0^{m_k} = U\left(w^t\right)$
    $L_t = \left\{\mathit{\Delta}_i^{t+1}, \cdots, \mathit{\Delta}_n^{t+1}\right\}$; $\phi = \{0, 0, \cdots, 0\}$ $(|\phi| = |L_t|)$
    while Mining-success criteria not met do
        $\pi_{m_k}^k \leftarrow$ Monte Carlo sampling permutation of $L_t$
        for $q = 1, 2, \cdots, \left|\pi_{m_k}^k\right|$ do
            $S = \left\{\pi_{m_k}^k[1], \pi_{m_k}^k[2], \cdots, \pi_{m_k}^k[q]\right\}$
            $w_S^{t+1} = w^t + \sum_{p \in S} \frac{|\mathcal{D}_p|}{\sum_{p \in S} |\mathcal{D}_p|} \mathit{\Delta}_p^{t+1}$
            $v_q^{m_k} = U\left(w_S^{t+1}\right)$
            $\phi_{\pi_{m_k}^k[q]} = \frac{1}{m_k}\left((m_k - 1)\,\phi_{\pi_{m_k}^k[q]} + v_q^{m_k} - v_{q-1}^{m_k}\right)$
        end
        $V_{m_k}^{t,k} = \left\{\phi_{\pi_{m_k}^k[1]}, \phi_{\pi_{m_k}^k[2]}, \cdots, \phi_{\pi_{m_k}^k\left[\left|\pi_{m_k}^k\right|\right]}\right\}$
        $sv_{m_k}^{t,k} = sort\left\{V_{m_k}^{t,k}\right\}$ by index in $SV^t$
        $J_k \leftarrow$ Judgement-SC($k, sv_{m_k}^{t,k}, \rho$)
        if $J_k == True$ then
            $\mathbb{B}_{k,t} \leftarrow$generate block(Txs,$sv_{m_k}^{t,k}$)
            broadcast $\mathbb{B}_{k,t}$ to all participants
            if verify($\mathbb{B}_{k,t}$)== True then
                for all participants :
                   stop PoShapley at this round $t$
                blockchain add $\mathbb{B}_{k,t}$
                return $SV^t$ and break
            end
        else
            $SV^t \leftarrow$ SV-Update-SC($k, sv_{m_k}^{t,k}$)
            $m_k = m_k + 1$
        end
    end
end
```

Algorithm 3: Judgement Smart-Contract

```
input  : $sv_{m_k}^{t,i}, \rho$
output: $J_i$
if invoke successful then
    for $i$ automatically do
        $\rho_k = \max \left| sv_{m_i}^{t,i} - SV^t \right|$
        if $\rho_i \leq \rho$ then
            $J_i = True$
        else
            $J_i = False$
        end
    end
    Return $J_i$
end
```

Algorithm 4: SV-Update Smart-Contract

```
input  : $sv_{m_k}^{t,i}$
output: $SV^t$
if invoke successful then
    for $k$ automatically do
        for $j = 1, 2, \cdots, n$ do
            $SV^t[j] = \frac{1}{2}\left(SV^t[j] + sv_{m_k}^{t}[j]\right)$
        end
    end
    Return $SV^t$
end
```

$\mathbb{B}_{k,t}$ contains all transactions of this training round, and its calculation result $sv_{m_k}^{t,k}$. Line 19-24 show if the verification of $\mathbb{B}_{k,t}$ passes, all participants would stop the PoShapley procedure at this round and append the newest block to the blockchain. Meanwhile, the consensus winner and the final approximated SV results can be acknowledged by all participants. Whereas, if *Judgement-Smart-Contract* returns a $False$ value, the participant k should invoke *SV-Update-Smart-Contract* to update the SV^t(Line 26). The automatic updating operations of computing the average value between $sv_{m_k}^{t,k}$ and SV^t are introduced in Algorithm 4. After that, the iteration time m_k is incremented by one, driving participant k to continue the loop at round t.

3.2 Fair and robust Aggregation

Through leveraging results of the PoShapley protocol, we designed the *SV-based aggregation*(i.e., Line 19 in Algorithm 1) to perform global updates with fairness and inferior data attack tolerance. Specifically, in each round t, after the election of a consensus winner M_t, a *SV-based-selection-Smart-Contract* is triggered by the M_t to generate clients set S_a whose corresponding model updates are to be selected for global model aggregation. The workflows of this contract are illustrated in Algorithm 5.

Algorithm 5: SV-based-selection-Smart-Contract

```
   input  : $SV^t$, $Private_{key}^{M_t}$
   output: $S_a^t$
1  if invoke successful then
2      verify identity
3      if $Public_{key}^{M_t} = f(Private_{key}^{M_t})$ then
4          for $M_t$ automatically do
5              for $i = 1, 2, \cdots, m$ do
6                  $v = max(SV^t)$, add the indies of corresponding participant
                    into combination $(v, k)$
7                  $S_a^t = append(S_a^t, (v, k))$
8                  $SV^t$=remove $v$ from $SV^t$
9              end
10         end
11         Return $S_a^t$
12     else
13         Return Error
14     end
15 end
```

To ensure security and fairness during the smart contract invocation, we first use the RSA encryption algorithm to verify the identities of M_t(Lines 2-3). Each participant's corresponding public key is submitted and held in the PoShapley-BCFL system when forming the collaborative group. The public key of a winner at each round t can be encapsulated into *SV-based-selection-Smart-Contract* as soon as the consensus process is finished. Only the private key from M_t can pass the verification and triggers the running of automatic selection for S_a, shown in Line 4-11. That is, top m values from the list of SV_t are selected, and their corresponding local updates are accepted for aggregation to get rid of some modifying updates or malicious inferior updates in each global iteration. Here $m \leq N$.

After that, the winner M_t performs the fair and robust aggregation using the formula (4). Unlike the simple average aggregation in (1), we assign aggregation weights based on the shapley value of selected participants and represent the aggregation formula in (4). Finally, the M_t delivers the newest global model updates to blockchain for the next round of training.

$$\Delta^{t+1} = \sum_{k \in S_a^t} \frac{SV_k^t}{\sum_{k \in S_a^t} SV_k^t} \Delta_k^{t+1}. \tag{4}$$

4 Experimental Results And Evaluations

4.1 Experimental Settings

This section introduces the experiment settings, including PoShapley-BCFL components in our experimental setup, dataset settings, evaluation metrics and other learning parameters.

PoShapley-BCFL components in the experimental setup Fig. 2 shows the arrangement components in the experimental setup of our PoShapley-BCFL. We used the Go language(version 1.15.7) and Hyperledger-Fabric-enabled channels, gossip and gRPC protocols to simulate the p2p communications and public ledgers among FL participants. For ease of implementation of PoShapley, the block structure and its chain-based generation process were reprogrammed by Go. Go language was also used to implement the PoShapley-BCFL smart contract, which was deployed to the PoShapley-BCFL blockchain networks using docker. As for the FL participant side, we used Go as the primary programming language, and multithreading settings and goroutine channels were utilized for networking, connections, and coordination among simulated participants. In this process, Pytorch 1.10 from Python(version 3.6) was used as the local training architecture to develop the learning behaviors in FL, which then communicated with some public and trust storage systems(such as IPFS) for model parameter exchange and storage.For the local experiments on the blockchain, the participant invoked the smart contracts through the Go SDK interface and then interacted with the PoShapley-BCFL blockchain network. PoShapley consensus protocol was implemented by the joint of Go and Python and embedded into the PoShapley-BCFL system.

Datasets and Evaluation metrics (1) **Datasets**. The dataset used in the experiments is based on the MNIST dataset. To evaluate the proposed algorithms under different FL settings, we designed IID and NIID FL scenarios with 10 participants as follows:

- *IID datasets−−Same Size and Same Distribution*: The MNIST dataset, which contains 60000 training samples of ten digits and 10000 testing samples, was evenly divided into ten parts as every participant's local training dataset(i.e., each participant has 6000 training samples and 10000 testing samples).
- *NIID datasets*:
  - *NIID-1−−Same Size and Different Distributions*: We allocate the same number of MNIST samples for every participant. However, different distributions are set as follows: participant 1 & 2's datasets contain 40% of digits '0' and '1', respectively. The other 8 participants evenly divide the remaining 20% of of digits '0' and '1'. Participant 3 & 4's datasets contain 40% of digits '2' and '3', respectively. The other 8 participants evenly divide the remaining 20% of of digits '2' and '3'. Similar procedures are applied to the rest of the samples.

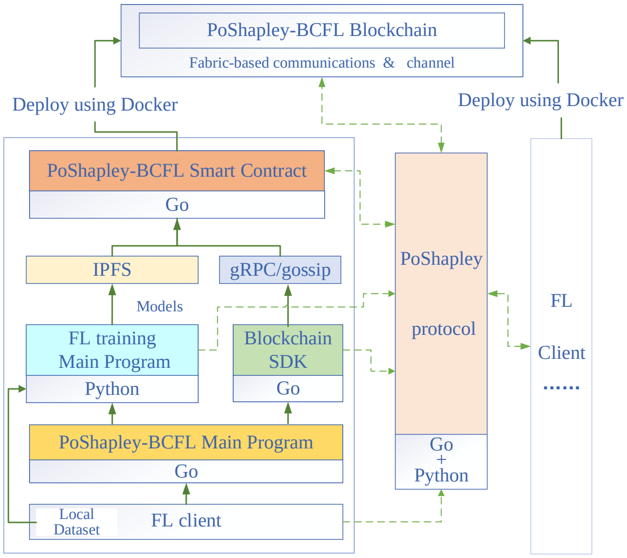


Fig. 2: PoShapley-BCFL components in the experimental setup.

- *NIID-2——Different Sizes and Same Distribution*: We randomly sample from the entire MNIST dataset following pre-defined ratios to achieve NIID-2 settings: The proportions are 5% for participants 1 and 2, respectively; 7.5% for participants 3 and 4, respectively; 10% for participants 5 and 6, respectively; 12.5% for participants 7 and 8, respectively; and 15% for participants 9 and 10, respectively.

(2) **Evaluation metrics**. To comprehensively test the performance of PoShapley, we chose the original shapley algorithm, following the principle of Equation (2), as a benchmark. We also used Adjust-SV from [23] and TMC-SV from [7] as comparison approaches. In addition, inspired by [16], we introduced the following evaluation metrics:

- *Distance metrics*: We used the results of the Original-SV algorithm as a baseline, with distance metric referring to the deviation from the results produced by the Original-SV algorithm. For any participant i, We denote its model contributions calculated by Original-SV in all training rounds as a vector $\phi_i^* = \langle \phi_{i,1}^*, \cdots, \phi_{i,T}^* \rangle$, and the estimated results calculated by any other approach are denoted as $\phi_i = \langle \phi_{i,1}, \cdots, \phi_{i,T} \rangle$. Three distances are introduced as follows.
  - *Euclidean Distance*: The Euclidean Distance for any participant i is defined as:

$$ED_i = \sqrt{\sum_{t=1}^{T} \left(\phi_{i,t}^* - \phi_{i,t}\right)^2}. \tag{5}$$

  - *Cosine Distance*: The Cosine Distance for any participant i is defined as:

$$CD_i = 1 - \cos\left(\phi_i^*, \phi_i\right). \tag{6}$$

  - *Maximum Distance*: The Maximum Distance for any participant i is defined as:

$$MD_i = \max{}_{t=1}^{T} \left|\phi_i^* - \phi_i\right|. \tag{7}$$

- *Time analysis*: The total time cost of calculating SVs and time complexity is used to evaluate the efficiency of each approach.
- *Accuracy analysis*: The accuracy metrics are used to evaluate the effectiveness of our PoShapley-BCFL with an SV-based aggregation rule, particularly in scenarios where adversarial nodes upload inferior model updates.

Other learning parameters We implemented a MLP neural network architecture as the training model and set learning rate $\eta = 0.01$, total training round $T = 10$, and mining-success criteria $\rho = 0.01$. As for evaluation function $U(w)$, since the *F1 Score* can better measure the performance of the models in most scenarios[22], we use it as the measure of contribution, i.e., $U(w) = F1(w)$.

4.2 Experimental Results Analysis

Firstly, we analyzed the accuracy and time performance of the PoShapley protocol and compared it to state-of-the-art baselines under various FL settings, including both IID and Non-IID (NIID) data silos. Next, we investigated the performance of the PoShapley-BCFL algorithm against inferior data attacks when using the SV-based aggregation. ***1) Accuracy analysis of PoShapley***:

Table 1: The Average SV Distance.

Dataset	Distance	Standard deviation of Distance		
		AdjustSV	TMC_SV	Poshapley
IID	$\overline{ED}$	0.0709	0.0921	**0.0558**
	$\overline{CD}$	0.4584	0.4169	**0.3229**
	$\overline{MD}$	0.0464	0.0546	**0.333**
NIID-1	$\overline{ED}$	0.0595	0.0842	**0.0520**
	$\overline{CD}$	0.2877	0.3077	**0.2054**
	$\overline{MD}$	0.0477	0.0639	**0.0401**
NIID-2	$\overline{ED}$	0.0707	0.0617	**0.0456**
	$\overline{CD}$	0.1296	0.1588	**0.1020**
	$\overline{MD}$	0.0560	0.0521	**0.0368**

We analyzed the experimental results under the three aforementioned dataset settings. In each case, 10 participants were involved in BCFL with 10 training rounds. And the average distances of all participants' evaluation under different dataset settings were calculated to represent the accuracy performance of PoShapley, shown in Table 1. Under IID data settings, PoShapley achieves the lowest average distance under all three distance metrics, demonstrating that PoShapley achieves the best contribution accuracy. And under Niid-1 settings, the results show that PoShapley still performs with the best accuracy according to the average distances. Notably, the average accuracy gap among the three algorithms is less than that under the IID settings. The results under NIID-2 situations show a similar pattern as in NIID-1, where PoShapley continues outperforming Adjust-SV and TMC-SV approaches regarding average distance. We attribute this advantage to the fact that PoShapley generates more permutations and calculation times of SV due to all participants working in parallel. Moreover, Table 2 compares the standard deviation of SV distance to indicate the stability of different algorithms. PoShapley achieves the slightest standard deviation of SV distances under all metrics and all datasets, making SV approximation more stable and fair for all participants. This advantage is more pronounced under the NIID-1 settings, illustrating that PoShapley is well-suited for these NIID settings.

Table 2: Standard deviation of SV Distance.

Dataset	Distance	Standard deviation of Distance		
		AdjustSV	TMC_SV	Poshapley
IID	Euclidean	0.0244	0.0375	**0.0172**
	consine	0.2391	0.2198	**0.2103**
	max	0.0156	0.0392	**0.011**
NIID-1	Euclidean	0.022	0.0332	**0.0134**
	consine	0.1887	0.2029	**0.0855**
	max	0.0165	0.0217	**0.0087**
NIID-2	Euclidean	0.0314	0.0328	**0.0146**
	consine	0.1569	0.1331	**0.11**
	max	0.0304	0.0229	**0.014**

2) Time cost and complexity analysis: To investigate the time cost of our PoShapley algorithm concerning the number of participants n, we varied n from 2 to 10 when performing PoShapley and other compared algorithms under all three dataset settings. The time cost values of each method were determined by calculating their average time under three data settings, and these values were shown in Fig. 3. The original SV method involves training and evaluating additional $2^n - 1$ models, resulting in exponential growth with the number of participants. In contrast, the other three algorithms significantly reduce computational time. It is notable that the runtime of TMC-SV is not significantly affected by an increase in the number of participants. This is because TMC-SV uses the Truncation Monte Carlo policy to drop models with a small marginal

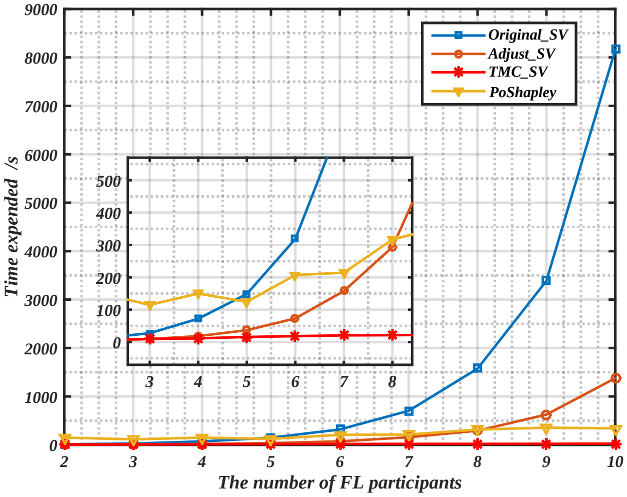
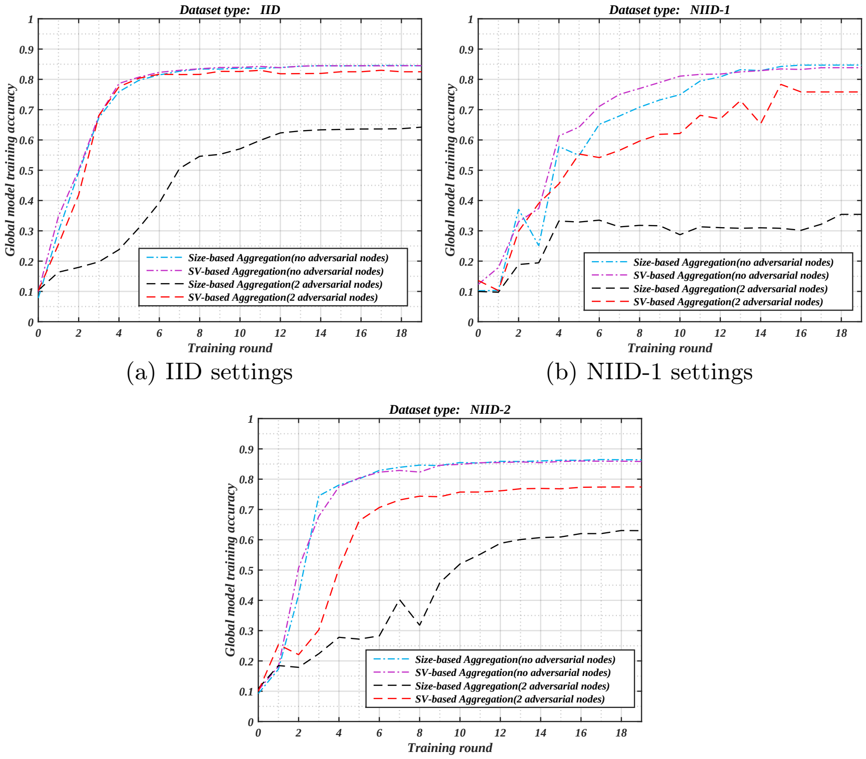


Fig. 3: Time costs with respect to the number of participant.

utility gain and keeps a low number of models in each round. However, TMC-SV performs poorly in terms of accuracy in our experiments. Adjust-SV uses an approximating algorithm during model reconstruction and outperforms PoShapley in terms of time when $n \leq 7$. However, Adjust-SV still relies on the principle of equation (2) when calculating SV, resulting in exponential time consumption as n increases(such as $n \geq 8$ in Fig. 3). As n increases, PoShapley saves even more time than Adjust-SV. In each calculation round of PoShapley, every participant involves in training and evaluating additional n models at each iteration m_k(Line 27 in Algorithm 2). Assuming the maximum m_k among all participants is m and the total training round is T, the number of evaluations of PoShapley is expressed as $\mathcal{O}\left(\sum_{t=1}^{T} mn\right)$, which indicates that the time complexity of PoShapley increases linearly with the number of participants..

3) Accuracy Performance of PoShapley-BCFL: In Fig. 4, we investigated the accuracy performance of PoShapley-BCFL with SV-based aggregation procedure, which is achieved by comparing with a typical weighted aggregation process based on data size(referred to as size-based aggregation). We also set up two groups of experimental comparisons: group 1, with 10 regular participants and no adversarial nodes, and group 2, with 8 regular participants and 2 adversarial participants(return randomized parameters). From Fig. 4, we can observe that the proposed SV-based aggregation method achieves almost the same performance in terms of model accuracy under all data settings when there are no adversarial participants. Notably, PoShapley-BCFL exhibits faster convergence than size-based aggregation under the NIID-1 scenario, which can be attributed to the reason that the SV-based approach encourages models with more remarkable contributions to occupy more weight in the aggregated models. Moreover, under all three data settings, PoShapley-BCFL significantly improves accuracy



(a) IID settings

(b) NIID-1 settings

(c) NIID-2 settings

Fig. 4: Accuracy convergence performance under various settings.

compared to baselines(size-based aggregation) when adversarial participants are attacking the collaborative learning process. Its performance is nearly as close to the settings without adversarial nodes under the IID scenario, and is slightly worse than that without adversarial nodes under the NIID settings, but remarkably outperforms the size-based method. This advantage is attributed to that the SV computing process naturally and automatically detect adversarial workers with lower or no contributions, allocating them with lower or no weights and no longer allowing them to participate in the current aggregation round. The above results confirm that the SV-based aggregation procedure is more robust than size-based methods and that PoShapley-BCFL is more effective when encountering malicious attacks.

5 Conclusion

This paper addresses the challenge of providing fair and robust incentives for blockchain-based decentralized federated learning services that support edge data sharing. We presented insights into designing a new blockchain-based serverless federated learning named PoShapley-BCFL, which has a modular design capable of evaluating model contributions while facilitating robust learning. To meet the lightweight calculation requirements and offer self-assessment in a decentralized setting, we proposed PoShapley. This Shapley-value-enabled blockchain consensus protocol provides fair and efficient contribution evaluation. Based on the results from PoShapley, we further design a fair-robust model aggregation algorithm that can tolerate inferior data attacks. Extensive experiments demonstrated that our proposed methods could promote fair and efficient contribution evaluation during decentralized collaborative learning and improve the final model performance through robust aggregation. For future work, since we observe that the marginal model contribution becomes smaller and smaller during the late stage of model convergence, we plan to study adaptive PoShapley, which can adjust the mining-success threshold during the learning process to prevent degradation of Shapley-value results.

References

1. An, Q., Wen, Y., Ding, T., Li, Y.: Resource sharing and payoff allocation in a three-stage system: Integrating network dea with the shapley value method. Omega **85**, 16–25 (2019). https://doi.org/https://doi.org/10.1016/j.omega.2018.05.008
2. Chen, S., Mi, H., Ping, J., Yan, Z., Shen, Z., Liu, X., Zhang, N., Xia, Q., Kang, C.: A blockchain consensus mechanism that uses proof of solution to optimize energy dispatch and trading. Nature Energy **7**(6), 495–502 (Jun 2022). https://doi.org/10.1038/s41560-022-01027-4
3. Chen, Y., Zhang, Y., Wang, S., Wang, F., Li, Y., Jiang, Y., Chen, L., Guo, B.: Dim-ds: Dynamic incentive model for data sharing in federated learning based on smart contracts and evolutionary game theory. IEEE Internet of Things Journal p. 1–1 (2022). https://doi.org/10.1109/JIOT.2022.3191671
4. Cheng, Z., Pan, Y., Liu, Y., Wang, B., Deng, X., Zhu, C.: Vflchain: Blockchain-enabled vertical federated learning for edge network data sharing. In: 2022 IEEE International Conference on Unmanned Systems (ICUS). p. 606–611. IEEE, Guangzhou, China (Oct 2022). https://doi.org/10.1109/ICUS55513.2022.9987097, https://ieeexplore.ieee.org/document/9987097/
5. Fan, S., Zhang, H., Zeng, Y., Cai, W.: Hybrid blockchain-based resource trading system for federated learning in edge computing. IEEE Internet of Things Journal **8**(4), 2252–2264 (2021). https://doi.org/10.1109/JIOT.2020.3028101
6. Ghorbani, A., Zou, J.: Data Shapley: Equitable Valuation of Data for Machine Learning. No. arXiv:1904.02868 (Jun 2019). https://doi.org/10.48550/arXiv.1904.02868, http://arxiv.org/abs/1904.02868, arXiv:1904.02868 [cs, stat] type: article
7. Ghorbani, A., Zou, J.: Data shapley: Equitable valuation of data for machine learning. In: Chaudhuri, K., Salakhutdinov, R. (eds.) Proceedings of the 36th International Conference on Machine Learning. Proceedings of Machine Learning Research, vol. 97, pp. 2242–2251. PMLR (09–15 Jun 2019), https://proceedings.mlr.press/v97/ghorbani19c.html
8. Hu, D., Chen, J., Zhou, H., Yu, K., Qian, B., Xu, W.: Leveraging blockchain for multi-operator access sharing management in internet of vehi-

cles. IEEE Transactions on Vehicular Technology **71**(3), 2774–2787 (Mar 2022). https://doi.org/10.1109/TVT.2021.3136364

9. Imteaj, A., Thakker, U., Wang, S., Li, J., Amini, M.H.: A survey on federated learning for resource-constrained iot devices. IEEE Internet of Things Journal **9**(1), 1–24 (Jan 2022). https://doi.org/10.1109/JIOT.2021.3095077
10. Issa, W., Moustafa, N., Turnbull, B., Sohrabi, N., Tari, Z.: Blockchain-based federated learning for securing internet of things: A comprehensive survey. ACM Computing Surveys **55**(9), 1–43 (Sep 2023). https://doi.org/10.1145/3560816
11. Jiang, L., Zheng, H., Tian, H., Xie, S., Zhang, Y.: Cooperative federated learning and model update verification in blockchain-empowered digital twin edge networks. IEEE Internet of Things Journal **9**(13), 11154–11167 (Jul 2022). https://doi.org/10.1109/JIOT.2021.3126207
12. Kang, J., Xiong, Z., Niyato, D., Zou, Y., Zhang, Y., Guizani, M.: Reliable federated learning for mobile networks. IEEE Wireless Communications **27**(2), 72–80 (2020). https://doi.org/10.1109/MWC.001.1900119
13. Li, D., Guo, Q., Yang, C., Yan, H.: Trusted data sharing mechanism based on blockchain and federated learning in space-air-ground integrated networks. Wireless Communications and Mobile Computing **2022**, 1–9 (Oct 2022). https://doi.org/10.1155/2022/5338876
14. Lin, X., Wu, J., Bashir, A.K., Li, J., Yang, W., Piran, M.J.: Blockchain-based incentive energy-knowledge trading in iot: Joint power transfer and ai design. IEEE Internet of Things Journal **9**(16), 14685–14698 (Aug 2022). https://doi.org/10.1109/JIOT.2020.3024246
15. Liu, Y., Ai, Z., Sun, S., Zhang, S., Liu, Z., Yu, H.: FedCoin: A Peer-to-Peer Payment System for Federated Learning, Lecture Notes in Computer Science, vol. 12500, p. 125–138. Springer International Publishing, Cham (2020). https://doi.org/10.1007/978-3-030-63076-8_9, `http://link.springer.com/10.1007/978-3-030-63076-8\_9`
16. Liu, Z., Chen, Y., Yu, H., Liu, Y., Cui, L.: Gtg-shapley: Efficient and accurate participant contribution evaluation in federated learning. ACM Trans. Intell. Syst. Technol. **13**(4) (may 2022). https://doi.org/10.1145/3501811, `https://doi.org/10.1145/3501811`
17. Lo, S.K., Liu, Y., Lu, Q., Wang, C., Xu, X., Paik, H.Y., Zhu, L.: Toward trustworthy ai: Blockchain-based architecture design for accountability and fairness of federated learning systems. IEEE Internet of Things Journal **10**(4), 3276–3284 (Feb 2023). https://doi.org/10.1109/JIOT.2022.3144450
18. Ma, S., Cao, Y., Xiong, L.: Transparent contribution evaluation for secure federated learning on blockchain. In: 2021 IEEE 37th International Conference on Data Engineering Workshops (ICDEW). p. 88–91. IEEE, Chania, Greece (Apr 2021). https://doi.org/10.1109/ICDEW53142.2021.00023, `https://ieeexplore.ieee.org/document/9438754/`
19. Nakamoto, S.: Bitcoin: A peer-to-peer electronic cash system. Cryptography Mailing list at https://metzdowd.com (03 2009)
20. Qinnan, Z., Jianming, Z., Sheng, G., Zehui, X., Qingyang, D., Guirong, P.: Incentive mechanism for federated learning based on blockchain and bayesian game. SCIENTIA SINICA Informationis **52**(6), 971– (2022). https://doi.org/https://doi.org/10.1360/SSI-2022-0020, `http://www.sciengine.com/publisher/ScienceChinaPress/journal/SCIENTIASINICAInformationis/52/6/10.1360/SSI-2022-0020`
21. Shapley, L.S.: A value for n-person games. Contributions to the Theory of Games (1953)

22. Shen, M., Duan, J., Zhu, L., Zhang, J., Du, X., Guizani, M.: Blockchain-based incentives for secure and collaborative data sharing in multiple clouds. IEEE Journal on Selected Areas in Communications **38**(6), 1229–1241 (Jun 2020). https://doi.org/10.1109/JSAC.2020.2986619
23. Song, T., Tong, Y., Wei, S.: Profit allocation for federated learning. In: 2019 IEEE International Conference on Big Data (Big Data). pp. 2577–2586 (2019). https://doi.org/10.1109/BigData47090.2019.9006327
24. TOUATI, S., RADJEF, M.S., SAIS, L.: A bayesian monte carlo method for computing the shapley value: Application to weighted voting and bin packing games. Computers & Operations Research **125**, 105094 (2021). https://doi.org/https://doi-org-s.libyc.nudt.edu.cn:443/10.1016/j.cor.2020.105094, `https://www-sciencedirect-com-s.libyc.nudt.edu.cn:443/science/article/pii/S0305054820302112`
25. Wang, T., Rausch, J., Zhang, C., Jia, R., Song, D.: A Principled Approach to Data Valuation for Federated Learning. No. arXiv:2009.06192 (Sep 2020). https://doi.org/10.48550/arXiv.2009.06192, `http://arxiv.org/abs/2009.06192`, arXiv:2009.06192 [cs, stat] type: article
26. Wang, X., Zhao, Y., Qiu, C., Liu, Z., Nie, J., Leung, V.C.M.: Infedge: A blockchain-based incentive mechanism in hierarchical federated learning for end-edge-cloud communications. IEEE Journal on Selected Areas in Communications p. 1–1 (2022). https://doi.org/10.1109/JSAC.2022.3213323
27. Weng, J., Weng, J., Huang, H., Cai, C., Wang, C.: Fedserving: A federated prediction serving framework based on incentive mechanism. In: IEEE INFOCOM 2021 - IEEE Conference on Computer Communications. pp. 1–10 (2021). https://doi.org/10.1109/INFOCOM42981.2021.9488807
28. Wu, W., Fu, Y., Wang, Z., Liu, X., Niu, Y., Li, B., Huang, G.Q.: Consortium blockchain-enabled smart esg reporting platform with token-based incentives for corporate crowdsensing. Computers & Industrial Engineering **172**, 108456 (Oct 2022). https://doi.org/10.1016/j.cie.2022.108456
29. Xu, L., Bao, T., Zhu, L.: Blockchain empowered differentially private and auditable data publishing in industrial iot. IEEE Transactions on Industrial Informatics **17**(11), 7659–7668 (Nov 2021). https://doi.org/10.1109/TII.2020.3045038
30. Yang, Q., Liu, Y., Chen, T., Tong, Y.: Federated machine learning: Concept and applications. ACM Transactions on Intelligent Systems and Technology **10**(2), 1–19 (Mar 2019). https://doi.org/10.1145/3298981