ESB-FL: Efficient and Secure Blockchain-Based Federated Learning with Fair Payment

Biwen Chen, Honghong Zeng, Tao Xiang, *Senior Member, IEEE*, Shangwei Guo, Tianwei Zhang, and Yang Liu, *Senior Member, IEEE*

Abstract—Federated learning is a technique that enables multiple parties to collaboratively train a model without sharing raw private data, and it is ideal for smart healthcare. However, it raises new privacy concerns due to the risk of privacy-sensitive medical data leakage. It is not until recently that the privacy-preserving FL (PPFL) has been introduced as a solution to ensure the privacy of training processes. Unfortunately, most existing PPFL schemes are highly dependent on complex cryptographic mechanisms or fail to guarantee the accuracy of training models. Besides, there has been little research on the fairness of the payment procedure in the PPFL with incentive mechanisms. To address the above concerns, we first construct an efficient non-interactive designated decryptor function encryption (NDD-FE) scheme to protect the privacy of training data while maintaining high communication performance. We then propose a blockchain-based PPFL framework with fair payment for medical image detection, namely ESB-FL, by combining the NDD-FE and an elaborately designed blockchain. ESB-FL not only inherits the characteristics of the NDD-FE scheme, but it also ensures the interests of each participant. We finally conduct extensive security analysis and experiments to show that our new framework has enhanced security, good accuracy, and high efficiency.

Index Terms—Federated Learning, Privacy Protection, Fair Payment, Function Encryption, Blockchain.

1 Introduction

Federated learning is an emerging and advancing technology that allows users to train a centralized model on separate datasets stored by different entities. It is increasingly prevalent in business and society, and its applications in healthcare drive reforms in the fields such as medical diagnosis and treatment [1], disease risk factor prediction. For example, FL allows medical institutions to train global machine learning models on huge amounts of COVID-19 case data [2] from different areas to predict infectious cases without data sharing. Obviously, the rapid growth of FL benefits from two main drivers: (1) applying machine learning technologies to distributed data scenarios, (2) providing data privacy protection during the data application process.

Although the FL brings great benefits to daily life, it also raises new privacy concerns in practical applications. Recent researches have shown that FL may not always offer enough privacy protection for local training datasets [3]. This is mainly because some malicious adversaries may extract sensitive information about training datasets from the model parameters such as weights or gradients [4]. For example, the membership inference attack [5], [6] could construct a series of shadow models through local gradients to reconstruct original data samples. Obviously, such potential privacy leakage risks in FL may be becoming a serious obstacle for practical applications, particularly the privacy-sensitive scenarios [7] (e.g., healthcare).

To address the privacy concerns, privacy-preserving FL (PPFL) is introduced by applying privacy-preserving mechanisms to FL. Currently, the PPFL schemes can be categorized into two types according to the privacy-preserving methods: (1) non-crypto-based methods such as differential privacy [8], (2) crypto-based methods such as homomorphic encryption (HE) [9] or secure multiparty computation (MPC) [10]. Although non-crypto-based PPFL schemes can provide more efficient performance, their training models may not be as accurate as crypto-based PPFL. This is mainly because the effect of adding noise on the model parameters may be unknown. In contrast to non-crypto-based PPFL, crypto-based PPFL can be used to accurately train general machine learning models while providing an appropriate level of data privacy protection.

Although there have been some researches [11], [12], [13], [14] on crypto-based PPFL, significant computational and communication costs might still be one of the common problems. One reason is that constructing schemes for general machine learning tasks relys on sophisticated technologies, such as homomorphic encryption [12], oblivious transfer [15]. For example, Hao *et al.* [12] adopted the fully homomorphic encryption technology to resist multiple-entities collusion attack, however, it also incurs vast computational and communication overhead. Bonawitz *et al.* [11] had used multiple cryptography tools such as secret sharing, digital signature, authenticated encryption, to maintain high model accuracy and strong privacy guarantee. However,

- *B. Chen is with the College of Computer Science, Chongqing University, Chongqing 400044, China, State Key Laboratory of Cryptology, P.O. Box 5159, Beijing, 100878, and also with the Guangxi Key Laboratory of Trusted Software, Guilin University of Electronic Technology, Guilin 541004, China (e-mail: macrochen@cqu.edu.cn).*
- *H. Zeng, T. Xiang and S. Guo are with the College of Computer Science, Chongqing University, Chongqing 400044, China (e-mail: {hhzeng; txiang; swguo}@cqu.edu.cn).*
- *T. Zhang and Y. Liu are with the School of Computer Science and Engineering, Nanyang Technological University, Singapore (e-mail: {tianwei.zhang; yangliu}@ntu.edu.sg).*
- *Corresponding author: Tao Xiang.*

their scheme requires enormous computing power to train a good model. Obviously, low efficiency may hinder the wide application of crypto-based PPFL schemes.

In addition, achieving fair exchange between task publishers and task participants in the FL framework is also an extremely meaningful challenge. To maintain the function of an FL organization, a financial incentive mechanism is typically needed to motivate task participants to train models actively. Thus, it is critical to ensure each FL participant is treated fairly. Currently, most FL schemes with fair treatment mainly focus on addressing the problem of how to achieve the fair assessment of FL task participants' contribution [16], [17], [18]. That is, they can guarantee the fairness of profit allocation. However, how to ensure fairness of both parties in the trading process is still an open problem. For example, if the task participants are paid in advance by the task publisher, they may be motivated to lower costs by training models lazily or with low accuracy. On the contrary, if the task publisher gets the training model first, he may not pay rewards for the task participants. Therefore, an effective FL framework should ensure that the FL task can be completed correctly and each participant who participates in the task can obtain the task reward timely.

To achieve fair payment, a native solution is to introduce a trusted arbiter, who serves as a middleman to enforce that both the task publisher and task participants follow policies at predefined. However, the role of the trusted arbiter might largely eliminate the benefits of this distributed framework. Over the last few years, blockchain has been proven to be highly effective at financial services. Some blockchain-based FL schemes [19], [20], [21], [22] have been proposed to build a decentralized, healthy FL system with incentive mechanisms. However, they fail to balance the privacy protection of data with fair payment of rewards. For example, to implement transactions between task publishers and task participants, the FLchain proposed by Bao *et al.* [19] requires that all computation results must be consensual on-chain, which however may raise the risk of leakage of personal sensitive data.

To address the above concerns, we introduce an efficient and secure blockchain-based FL system framework. In this framework, we first adopt a new proposed lightweight cryptography tool (i.e. non-interaction designated decryptor function encryption) to encrypt each local gradient. As a remarkable advantage, it can achieve desirable privacy protection and retain the accuracy of the global model while maintaining low communication costs. Meanwhile, the DGC algorithm [23] is introduced to further reduce the communication costs. This is because each local gradient to be sent must be reached a threshold, and thus it greatly reduces the amount of data communicated between nodes. Then, by elaborately designing a blockchain structure and using smart contract technology, fair payment between task publisher and task participants is achieved to ensure that all task participants will get the rewards as long as the task publisher obtains the qualified and correct model.

In summary, our main contributions are as follows.

- We propose a new function encryption scheme, namely non-interactive designated decryptor function encryption (NDD-FE). NDD-FE avoids multiple interactions between the encryptor and the key generator and achieves that only the designated decryptor can decrypt the aggregated global model.
- We design a new block structure of blockchain, which binds the task and model information to the block generation. This guarantees each task participant gets rewards if and only if the trained model satisfies the task requirements, thereby guaranteeing the fairness of the payment process.
- By integrating the proposed NDD-FE and the designed blockchain into federated learning, we propose ESB-FL, an efficient and secure blockchain-based federated learning framework. ESB-FL can not only train a highly accurate model while protecting the privacy of local training data, but also supports the fair payment between the task publisher and all participants.
- We perform a security analysis and effectiveness assessment of the proposed ESB-FL to demonstrate its desired security and efficiency.

The rest of this paper is organized as follows. Section 2 describes related work. Section 3 provides a brief introduction to the preliminaries. Section 4 describes the building blocks that will be used in our framework. Section 5 introduces the proposed framework in detail. Section 6 and 7 present the security and performance analyses about the proposed framework, respectively. Finally, Section 8 presents a conclusion.

2 Related Work

In this section, we mainly review the following two research topics in FL related to this paper.

Privacy-Preserving FL. PPFL aims to collaboratively train a global model while preserving data privacy. Typically, the privacy-preserving mechanisms used by PPFL can be divided into two types: non-cryptographic technology and cryptographic technology. In the non-cryptographic technologies [24], [25], [26], [27], the differential privacy (DP) method has a major share, which provides a privacy guarantee by adding noise to local data or model parameters. However, applying the DP method to FL results in a contradiction between privacy-preserving level and model accuracy [28]. That is, it is hard to achieve high model accuracy and strong privacy simultaneously.

In cryptographic technologies, both homomorphic encryption (HE) [13] and secure multiparty computation (MPC) [29] are two mainstream methods at present, because they can provide stronger privacy protection while retaining the original accuracy. Zhang *et al.* [30] constructed a HE-based PPFL framework, in which each local gradient will be encrypted to avoid information leakage. Zhang *et al.* [31] proposed an efficient HE scheme for FL. In their scheme, a new quantization and encoding scheme is developed to reduce computation and communication costs. Compared with the HE method, the MPC method is considered a promising tool for the privacy protection of FL since it allows distributed participants to securely calculate an objective function. Chaudhari *et al.* [32] proposed a four-party MPC-based PPFL framework, namely Trident. Compared with Gordon *et al.* [33], Trident requires fewer participants to

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TBDATA.2022.3177170, IEEE Transactions on Big Data

be active and has better online communication efficiency. So *et al.* [10] introduced a scalable PPFL framework based on secure MPC protocol, where MPC protocol is exploited to transform the dataset. However, significant communication costs of these solutions is still a challenge to facilitate the wide application [34].

In addition to the HE and MPC, some special cryptography primitives [35], [36] are also used to provide privacy protection of FL for specific applications. Szatmari *et al.* [36] proposed a secret-sharing-based PPFL for modelling audiological preferences. Xu *et al.* [35] proposed a verifiable PPFL framework, where the key agreement protocol is used to protect the privacy of the local gradients. Guo *et al.* [37] conducted a comprehensive review of research on collaborative learning and introduced in detail the existing attacks against FL and corresponding defense mechanisms. Obviously, exploring different cryptography methods for different applications may be an efficient way to balance the accuracy, privacy, and efficiency of FL.

Blockchain-Based FL. As an emerging technology, blockchain is widely introduced into FL framework to enhance security or service availability due to its decentralization, verifiability, and immutability. Qu *et al.*[38] proposed a blockchain-enabled FL scheme to remedy the privacy and efficiency problems of fog computing, in which a blockchain system replaces the central authority to resist the poisoning attacks. To enhance the reliability and efficiency, Lu *et al.*[39] designed a new architecture for data sharing, based on an asynchronous FL and a hybrid blockchain framework. The hybrid blockchain framework consists of permissioned blockchain and local directed acyclic graphs. To solve the data falsification, Wan *et al.* [40] integrated blockchain with FL to protect the privacy of edge devices in 5G networks.

Different from the above methods to improve the security of FL, some works [41], [42], [17], [43] focus on enhancing the scalability of FL by leveraging blockchain technology. Ramanan *et al.* [42] used the blockchain technology to remove the need for the central FL server but also led to the high computation and communication overhead. To motivate participants, Zhang *et al.* [43] proposed an incentive mechanism for FL, where the blockchain is used to guarantee that the information of reputation cannot be tampered with and can be publicly accessible. Likewise, Gao *et al.* [44] also proposed a blockchain-based fair incentive mechanism for FL to address the profit-sharing problem. Their scheme uses the blockchain to store all intermediate results to prevent fraud and denial. Liu *et al.* [17] built a payment system for FL by adopting the blockchain as a distribution ledger.

Obviously, blockchain has proved to be a powerful tool to enhance the practicality of FL. However, most blockchain-based FL schemes regard the blockchain merely as a decentralized and immutable ledger and use it to build trust. Our work will adopt the blockchain to achieve the fairness of the payment process, namely fair payment, which can ensure the interests of task publishers and task participants.

3 PRELIMINARIES

3.1 Blockchain

Blockchain can be regarded as a decentralized, distributed database. A simplified diagram of the blockchain is shown in Fig. 1, which shows that the blockchain is a collection of a series of blocks connected in chronological order. Blocks are joined by hash pointers, and each block contains the hash value of the previous block. With this structure, if the data in one block is tampered with, all blocks following that block are changed and can be detected immediately. All blocks in a blockchain are verified by the nodes on the chain, and adding a new block to the blockchain requires the consensus of the nodes. If the majority of nodes verify that the new block meets the requirement, the new block is accepted as the next block of the longest legal chain.

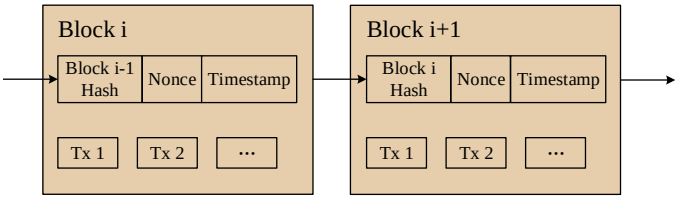


Fig. 1. A simplified blockchain structure.

3.2 Hard Problem Assumption

The privacy of our scheme relies on the CDH assumption. That is, if the CDH assumption holds, then our scheme is secure. The CDH assumption is defined as follows.

Let G denote a cyclic group with prime order p and g be its generator. Let $GGen(1^\lambda)$ be an efficient algorithm that takes a security parameter λ as input and probabilistically outputs (G, p, g).

CDH Assumption. Given parameters (G, p, g) generated by $GGen(1^\lambda)$ and $(g^a, g^b) \in G$, where (a, b) are randomly chosen in $\mathbb{Z}_p^*$, the CDH problem is to compute g^{ab}. We say that the CDH assumption holds if the advantage of solving the CDH problem is negligible.

According to the discrete logairthm, it is impossible to recover the exponent $r \in \mathbb{Z}_p^*$ of $g^r \in G$ if r is large and random enough. Note that it is still possible to recover the exponent κ of g^κ by leveraging the *baby-step giant-step* algorithm [45] if $\kappa \ll r$.

3.3 Function Encryption

Function encryption (FE) is a promising cryptographic primitive that allows authorized users to delegate to third parties the computation of functions of the encrypted data by generating specific secret keys for these functions. It allows us to utilize encrypted data while protecting data privacy. Compared with the MPC and HE, FE is a more lightweight and efficient cryptography solution towards constructing a privacy-preserving FL. We now briefly introduce a FE scheme in [35], [46] that supports basic arithmetic operations. Let $\otimes$ represent multiplication operations in function encryption, the details are defined as follows.

- **FE.Setup**(1^λ): It takes the security parameter λ as input, and generates group sample $(G, p, g) \leftarrow$

$GGen(1^\lambda)$ and selectes a random key $s \leftarrow \mathbb{Z}_p$. Then it outputs the master secret key $msk = s$ and the public key $mpk = (H, g)$, where $H = g^s$.

- **FE.Encrypt**(mpk, x): It takes mpk and the data $x \in \mathbb{Z}_p^*$ that needs to be encrypted as input, and outputs a commitment ct and the ciphertext c. It first randomly selects a random $r \leftarrow \mathbb{Z}_p^*$ and generates the commitment $ct = g^r$ and the ciphertext $c = H^r \cdot g^x$.
- **FE.KeyDerive**$(mpk, msk, ct, \otimes, y)$: It takes msk, the commitment ct and the input of function y as inputs and outputs the special function key sk for $\otimes$ operation. Specifically, the generated function key is $sk_\otimes = (ct^s)^y = g^{rsy}$.
- **FE.Decrypt**$(mpk, sk_\otimes, c, y)$: It takes mpk, the function key $sk_\otimes$ for $\otimes$ operations, the ciphertext c and another input y as input, computes $g^{x \otimes y} = c^y / sk_\otimes = (H^r g^x)^y / ((g^r)^{sy}) = g^{xy}$, and finally extracts the exponent xy.

3.4 Federated Learning

FL is gradually applied to the field of medical image detection by training a global machine learning model on multiple datasets stored by different centers without data sharing. In general, a classic FL framework for medical image detection can be summarized as the following phases. Suppose that there are h medical data centers as the participants, which train their local models based on their local data. Meanwhile, there is an aggregation node A that is responsible for aggregating local models and computing the global model.

Initialization. The aggregation node A publishes the training task T and the parameters of the training process, and then initializes the global model as W_0. Finally, it distributes the t-th global model W_t to all participants, where $t = \{0, 1, ...\}$.

Local Training. After receiving the global model W_t, each participant $i \in \{1, ..., h\}$ sets W_t as his local model, denoted as W_t^i. Then, guided by the task requirement, the participant i optimizes his local model W_t^i by minimizing the loss function $f_{loss}(W_t^i)$ and then obtains an ideal model $(W_t^i)^*$, i.e.

$$(W_t^i)^* = arg\ min_{W_t^i} f_{loss}(W_t^i)$$

Finally, all participants $(1, ..., h)$ send their local models $\{(W_t^i)^*\}_{i=1,...,h}$ to the aggregation node A.

Aggregation and Update. The aggregation node A aggregates all local models as follows:

$$W_{t+1} = \frac{1}{h} \sum_{i=1}^{h} f_{loss}(W_t^i)^*$$

Then, the aggregation node A sets it as the latest global model and determines whether it accords with the need of training accuracy. If not, A distributes the global model to all participants, like the **Initialization** phase.

Note that these steps may be repeated several times until a desirable training model is obtained.

4 Building Blocks

In this section, we first present a new function encryption scheme, and then describe our designed blockchain.

4.1 Non-interactive Designated Decryptor Function Encryption

Although the FE scheme in [35] supports several basic arithmetic computations, as described in Section 3, it is still faced with two challenges if applied directly to the FL framework.

The first challenge is that the multi-round communications between the key generator and the data encryptor may result in high communication costs in the FL framework. In their scheme, the generation of the function key $sk_\otimes$ requires a commitment ct sent by the encryptor, that is, the key generator needs to communicate with each data encryptor for generating a decrypt secret key in each model update process. The second challenge is that relying on a trusted entity further limits the usability and scalability of their scheme. In their scheme, except for the user with the special key $sk_\otimes$, the key generator can also decrypt all ciphertext $c = H^r g^x$ by using the master secret key s since all data are encrypted under the master public key H. Therefore, the key generator must be a trusted entity to ensure the privacy of data. However, the trusted central entity is difficult to be established in the distributed FL framework.

To overcome the above drawbacks, we propose the concept of *non-interactive designated decryptor function encryption (NDD-FE)* supporting *inner-product* computation, in which the encryptor does not need to interact with the key generator and only designated decryptor with function key can decrypt the *inner-product* results on the encrypted data. Suppose the *inner-product* functionality is to compute $f(\mathbf{x}, \mathbf{y}) =< \mathbf{x}, \mathbf{y} >= \sum_{i=1}^{n}(x_i y_i)$, where n denotes the length of the vectors $\mathbf{x}, \mathbf{y}$ and (x_i, y_i) denote the i-th elements of $\mathbf{x}, \mathbf{y}$, respectively.

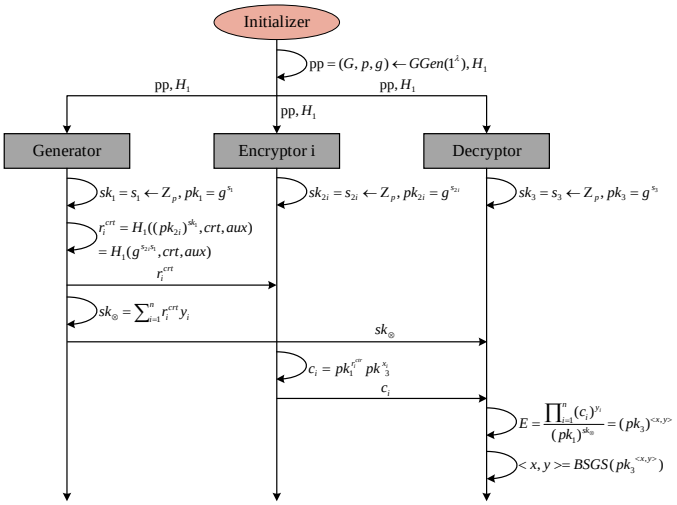


Fig. 2. System model of NDD-FE.

For the convenience of description, we describe the NDD-FE scheme using three roles, namely *generator, encryptor* and *decryptor*. The system model is shown in Fig. 2, and the construction is defined as follows.

- **NDD-FE.Setup**$(1^\lambda) \rightarrow pp$: It takes the security parameter λ as input and generates system public parameter $pp = (G, p, g) \leftarrow GGen(1^\lambda)$ and a secure hash function $H_1 : G \rightarrow \mathbb{Z}_p^*$.

- **NDD-FE.KeyGen**$(pp) \rightarrow (pk, sk)$: It is executed by all participants, including *generator, encryptor* and *decryptor*. It takes the system public parameter pp as input, and selectes a random number $s \leftarrow \mathbb{Z}_p$ as the secret key and the public key $pk = H = g^s$. Let $(pk_1 = g^{s_1}, sk_1 = s_1)$, $(pk_{2i} = g^{s_{2i}}, sk_{2i} = s_{2i})_{i=1,\ldots,n}$ and $(pk_3 = g^{s_3}, sk_3 = s_3)$ denote the public/secret key pairs of the *generator*, the i-th *encryptor* and the *decryptor*, respectively.
- **NDD-FE.KeyDerive**$(pk_1, sk_1, \{pk_{2i}\}_{i=1,2,\ldots,n}, ctr, \mathbf{y}, aux) \rightarrow sk_{\otimes}$: It is executed by the *generator*. It takes the public/secret key pair $(pk_1 = g^{s_1}, sk_1 = s_1)$ of the *generator*, the public keys $\{pk_{2i} = g^{s_{2i}}\}_{i=1,2,\ldots,n}$ of n encryptors, an incremental counter ctr, a vector $\mathbf{y}$ that needs to be computed with ciphertext and an auxiliary information aux as input, and outputs the function key $sk_{\otimes} = \sum_{i=1}^{n} r_i^{crt} y_i$, where $r_i^{crt} = H_1(pk_{2i}^{sk_1}, crt, aux) = H_1(g^{s_{2i}s_1}, crt, aux) \in \mathbb{Z}_p^*$ for each encryptor pk_{2i}.
- **NDD-FE.Encrypt**$(pk_1, sk_{2i}, pk_3, ctr, x_i, aux) \rightarrow c_i$: It is executed by n *encryptors*. It takes the public key pk_1 of the *generator*, the public/secret key pair (pk_{2i}, sk_{2i}) of the i-th *encryptor*, the public key pk_3 of the *decryptor*, and the data x_i as input, and outputs the ciphertext $c_i = pk_1^{r_i^{crt}} \cdot pk_3^{x_i}$, where $r_i^{crt} = H_1(pk_1^{sk_{2i}}, crt, aux) = H_1(g^{s_1 s_{2i}}, crt, aux) \in \mathbb{Z}_p^*$.
- **NDD-FE.Decrypt**$(pk_1, sk_{\otimes}, sk_3, \{c_i\}_{i=1,\ldots,n}, \mathbf{y}) \rightarrow < \mathbf{x}, \mathbf{y} >$: It is executed by the *decryptor*. It takes the public key pk_1 of the *generator*, the function key $sk_{\otimes}$, the secret key sk_3 of the *decryptor*, the ciphertexts $\{c_i\}_{i=1,\ldots,n}$ and the vector $\mathbf{y}$ as input, and computes:

$$E = \frac{\prod_{i=1}^{n} c_i^{y_i}}{(pk_1)^{sk_{\otimes}}} = \frac{\prod_{i=1}^{n} ((pk_1)^{r_i^{crt}} pk_3^{x_i})^{y_i}}{(pk_1)^{\sum_{i=1}^{n} r_i^{crt} y_i}}$$
$$= \frac{(pk_1)^{\sum_{i=1}^{n} r_i^{crt} y_i} pk_3^{\langle x,y \rangle}}{(pk_1)^{\sum_{i=1}^{n} r_i^{crt} y_i}} = pk_3^{<\mathbf{x},\mathbf{y}>}$$

  and computes $E^* = E^{\frac{1}{sk_3}} = g^{<\mathbf{x},\mathbf{y}>}$. Finally, it can recover the *inner-product* of vectors $(\mathbf{x}, \mathbf{y})$ by using the *baby-step giant-step (BSGS)* algorithm.

Scheme Analysis. Here, we analyze the proposed FE scheme in terms of security and performance.

For security, it is straightforward to see that the security of the NDD-FE scheme is the same as that of the scheme in [35]. This is because the ciphertext $c_i = pk_1^{r_i^{crt}} pk_3^{x_i}$ in our scheme and the ciphertext $c_i = H^{r_i} g^{x_i}$ in [35] can be viewed as the standard ElGamal ciphertext generated by the ElGamal encryption scheme, and thus any outside adversaries cannot obtain the information of data. However, the main difference between our scheme and [35] is that our scheme ensures that only the designated *decryptor* can decrypt the ciphertext due to the usage of the secret key sk_3 of the designated decryptor in the decryption phase. Moreover, NDD-FE supports secure multiple rounds of encryption, where the participants can automatically update the encrypted random number r_i^{crt} by introducing the incremental parameter crt, thereby updating the ciphertexts c_i.

For performance, our scheme removes the interactions between the *generator* and the *encryptors* since the generation of the function key does not require the information sent by the *encryptor*. In the FL framework, training a good model usually takes multiple rounds of updates, that is, the FE scheme will be executed multiple times. Therefore, the performance advantage will become more meaningful as the number of times the FE scheme increases.

4.2 Designed Blockchain

To achieve fair payment and exploit the computation power of miners, we design an elaborate blockchain by modifying the traditional blockchain from two following aspects.

Block Header. Different from the traditional blockchain, we define a new block header structure, as shown in Fig. 3. In addition to the general attributes such as the block version, the hash value of the previous block, the difficulty value, and so on, we introduce two new attributes to the block header in our blockchain: (1) digital signature of task information and participating miners information, (2) digital signature of model hash, model link and model accuracy. The two new attributes bind the task information to the block and establish a relationship between the task and the training model.

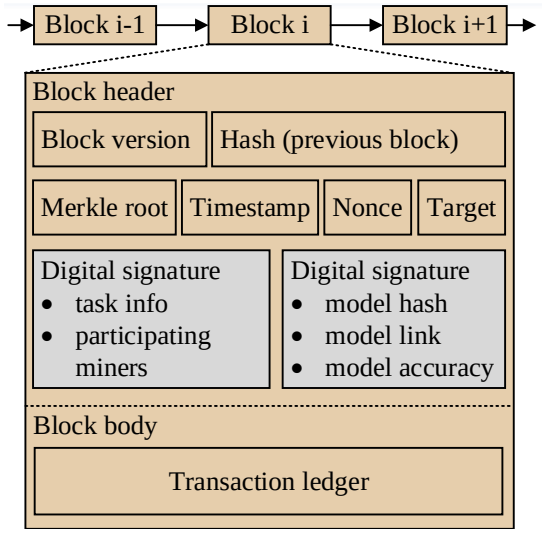


Fig. 3. The designed block structure.

New Block Generation. In the typical blockchains (i.e., Bitcoin), new blocks are generated by solving hard computation problems, thereby resulting in the energy-wasting issue. Thus, to address this drawback, our mechanism asks miners to perform meaningful FL tasks and presents a global model.

To generate a new block, our mechanism needs to complete two following phases. In the first phase, the task publisher publishes information about the task and participating miners, the task publisher will sign this information as a payment token. Then, all participants execute their works based on their roles, respectively. After completing the model training task, a special miner builds a new block based on the above-defined structure and sends it to other miners for verification. If the trained model satisfies the requirements of the task, then the new block can pass validation and is added to the blockchain networks. Note

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TBDATA.2022.3177170, IEEE Transactions on Big Data

that all new blocks contain the $timestamp$ attribute, which can effectively avoid the case of blockchain forks in multi-task parallel processing.

5 Design of ESB-FL

5.1 System Overview

We outline the architecture of our efficient and secure blockchain-based FL framework (ESB-FL), as shown in Fig. 4. The proposed framework consists of three roles, i.e., task publisher, miners, and aggregation node.

- **Task publishers (TP).** They may be enterprises, research institutes, or healthcare research units that try to obtain a disease detection model for a medical disease. However, since the lack of real medical data, they have to outsource their tasks and pay for model training and data services.
- **Miners (M).** They are the service providers (e.g., hospitals), who hold various types of medical datasets and have the different computing power to train models, respectively. To earn monetary rewards, the miners collaborate on training a global model according to the published tasks, verify the final model and the newly generated block.
- **Aggregation node (A).** It is a special type of miner, which is responsible for aggregating the local gradients submitted by all other participating miners and returning the aggregated gradient to them for the next iteration. And beyond that, it is also responsible for sending the final qualified model to the task publisher to gain permission to publish a new block.

The high-level workflow in our ESB-FL framework is as follows. *First*, by leveraging blockchain, TP publishes a medical training task which includes the relevant parameters such as task requirements, rewards, etc. *Second*, once receiving new tasks, miners M who meet the requirement of the same task respond by sending their personal information such as the public key, computing power and so on. After identifying the participants, TP needs to generate the keys and encryption parameters of FE for the participating miners, while generating the decryption keys with the special function for the aggregation node A. Then, the participating miners will individually train the model based on their local data. *Third*, when the training is temporarily over, the miners need to encrypt the compressed training model using the encryption algorithm of FE and send the encrypted model to A, respectively. Once receiving the models from different miners, A will aggregate them and decrypt the aggregation model using the decryption key of FE. If the accuracy of the decrypted model does not meet the task requirement, A returns it to the participating miners for the next iteration, while the participating miners will start all over again. Note that the process may require potentially numerous iterations until the accuracy is enough. *Fourth*, After the above phase, A will return the final model to TP for the permission to publish a new block. *Finally*, TP will reward the participating miners, while A will get the block rewards.

Design Goals and Assumptions. Our design goals are to enforce the following security and performance guarantees.

- **Confidentiality.** The confidentiality of medical data stored by each miner is the first and most important security requirement. Our framework should ensure that any unauthorized adversary cannot learn the privacy information of training data. That is, the adversary cannot reconstruct private medical data samples from local models acquired during task processing.
- **Efficiency.** Efficiency is the key property in practical application. Our ESB-FL framework should ensure that the task can be processed effectively and be completed in time. Besides, we aim to provide another practical notion: parallelism, indicating that multiple FL tasks can be executed simultaneously.
- **Fair Payment.** Fair payment is a key financial property of the incentive mechanism. It guarantees the interests of each TP and M and promotes the long-term development of ESB-FL. More specifically, TP can obtain the correct model that meets his requirements as long as he pays for his request, while the miners can earn the corresponding rewards as long as they do correct computations.

Availability Assumptions. To guarantee the functionality and security of our framework, the following reasonable assumptions should be ensured.

- **Assumption 1.** The networks of each participant in the framework should be stable enough to work. For example, the miners should be able to receive several task updates and perform task selection and processing immediately.
- **Assumption 2.** Both TP and A will not collude with each other. That is, the TP has stayed out of the model training process. Fortunately, it can be easily guaranteed by delegating an agent to publish tasks.

5.2 ESB-FL Framework

By combining the above NDD-FE scheme and our designed blockchain, we proposed an efficient and secure blockchain-based FL framework, i.e. ESB-FL. It consists of five main modules: task publishing, model training, model aggregation, block publishing, and task reward releasing.

5.2.1 Task Publishing

In this module, TP with public/secret key pair (pk_{TP}, sk_{TP}) first issues a service request by publishing a medical training task T_i to the blockchain, where T_i contains the task publisher identity pk_{TP}, the task status S, dataset requirements D, initialization model link L, the expected model accuracy acc, the expected processing time T_{exp} and task rewards R, where S is set to $publishing$. Note that multiple task publishers can publish their tasks $T_{i=1,2,...}$ at the same time.

According to the published tasks, the miners $\{M_j\}_{j=0,...,h}$ who meet D of T will respond to the request by sending the relevant proofs that can prove their ability and their public keys $(pk_0, ..., pk_h)$, respectively. Once receiving enough replies from miners, the task publisher modifies S to $processing$ and broadcasts the list of participating miners. The task publisher TP performs

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TBDATA.2022.3177170, IEEE Transactions on Big Data

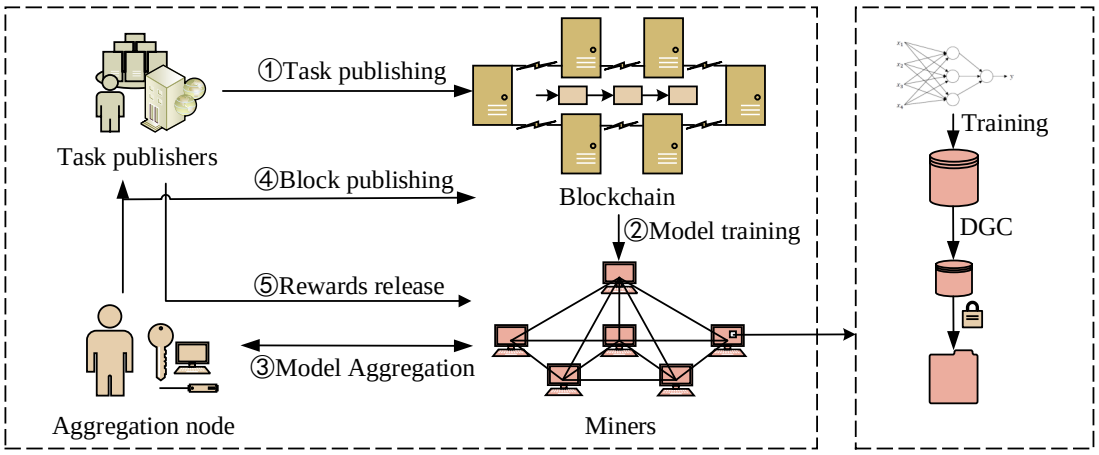


Fig. 4. Framework of ESB-FL.

Proof of State (PoS) algorithm [47] to select a special miner as the aggregation node $A \in \{M_j\}_{j=0,...,h}$, which ensures that the aggregation node is chosen at random. Suppose that M_0 is selected as the aggregation node.

Then, the task publisher TP executes the **NDD-FE.KeyDerive**$(pk_{TP}, sk_{TP}, \{pk_i\}_{i=1,2,...,h}, ctr, \mathbf{y}, T) \rightarrow sk_{\otimes}$ algorithm to generate the function key $sk_{\otimes}$ for the aggregation node A, where $ctr = \{1, 2, ...,\}$ is an incremental counter, $\mathbf{y} = \{y_1, ..., y_h\}$ is a $h-$dimentional weight vector and $y_i = \frac{1}{h}$. Suppose that the function key $sk_{\otimes} = \sum_{i=1}^{h} r_i y_i$, which means $r_i = H_1(pk_i^{sk_{TP}}, crt, T)$.

Finally, the task publisher TP publishes the signature σ_1 that contains task information and participating miner list as the payment token. Also, he deploys a smart contract to specify payment policies that are used to pay rewards to miners. Please refer to Algorithm 2 for more details. We also show the process for the task publishing phase in Fig. 5.

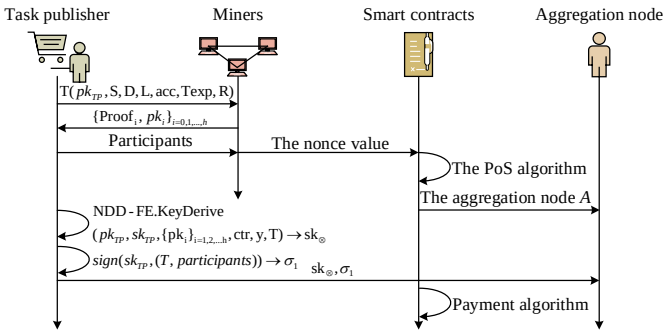


Fig. 5. Task publishing process.

5.2.2 Model Training

This module roughly consists of three steps: (1) local training, (2) gradient compression, (3) gradient encryption, as shown in the right half of Figure 4.

After obtaining the published global model W_t at t-round, each miner $M_j, j = 1, ..., h$ begins to train the local model based on the local data D_j and the loss function $f_{loss}(W_t)$, and then obtain the updated model W_t^j. The details of the training model are not the focus of this work.

Algorithm 1 DGC algorithm

Input: dataset D, minibatch size b per node, the number of nodes N, init parameters $w = \{w_i[0], w_i[1], ..., w_i[n]\}$
Output: W_{t+1}^j

$G_0 \leftarrow 0$
for $t = 1, ...$ **do**
  $G_t \leftarrow G_{t-1}$
  for $i = 1$ to b **do**
    Sample data d from D_j
    $G_t \leftarrow G_t + \frac{1}{Nb} \nabla f(d; w)$
  end for
  for $j = 0$ to n **do**
    Select threshold: $\tau \leftarrow s\%$ of $|G_t[j]|$
    $Tmp \leftarrow |G_t[j]| > \tau$
    $\widetilde{G}_t[j] \leftarrow G_t[j] \odot Tmp$
    $G_t[j] \leftarrow G_t[j] \odot \neg Tmp$
  end for
  $W_{t+1}^j \leftarrow encode(\widetilde{G}_t)$
end for

When completing the local model training, M_j parses the model W_t^j as $W_t^j = \{w_j[0], w_j[1], ..., w_j[n]\}$ to compresses it. In our framework, the DGC algorithm $W_{t+1}^j = DGC(W_t^j, b, N, n, s)$ (see Algorithm 1) is chosen to improve communication efficiency, where n denotes the number of total parameters and s denotes the compression rate set by the task publisher. In the process of gradient compression, there exists a threshold τ to determine whether the gradient meets the requirements. That is, only the gradient greater than the threshold τ can be selected for transmission. Meanwhile, to prevent data loss, miners whose gradient W_{t+1}^j does not reach the threshold need to accumulate the remaining gradients locally until the gradient becomes large enough. In addition to improving performance, the DGC algorithm transforms high-dimensional data to low-dimensional, thereby solving the input format problem of the encryption algorithm to be used.

Finally, to prevent the gradient leakage, each miner M_j encrypts the compressed gradient W_{t+1}^j using **NDD-**

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TBDATA.2022.3177170, IEEE Transactions on Big Data

$\mathbf{FE.Encrypt}(pk_{TP}, sk_j, pk_0, ctr, W_{t+1}^j, \mathbf{T})$.

$$U_{t+1}^j = \mathbf{NDD\text{-}FE.Encrypt}(pk_{TP}, sk_j, pk_0, ctr, W_{t+1}^j, crt, \mathbf{T}) = pk_{TP}^{r_j} \cdot pk_0^{W_{t+1}^j}$$

where $r_j = H_1(pk_{TP}^{sk_j}, crt, \mathbf{T})$. Then, each miner M_j sends the encrypted update U_{t+1}^j to the aggregation node A as the $t+1$-th round response.

5.2.3 Model Aggregation

Once receiving all encrypted updates $U_{t+1} = (U_{t+1}^1, U_{t+1}^2, ..., U_{t+1}^h)$ at the $t+1$ round, the aggregation node A with the function key $sk_\otimes$ performs the decryption algorithm of NDD-FE scheme to obtain the aggregation model θ_{t+1}. The aggregation node A first performs

$$\begin{aligned} E_{t+1} &= \mathbf{NDD\text{-}FE.Decrypt}(pk_{TP}, sk_\otimes, sk_0, U_{t+1}, \mathbf{y}) \\ &= \frac{\prod_{j=1}^h (U_{t+1}^j)^{y_j}}{(pk_{TP})^{sk_\otimes}} = \frac{\prod_{j=1}^h ((pk_{TP})^{r_j} pk_0^{W_{t+1}^j})^{y_j}}{(pk_{TP})^{\sum_{j=1}^h r_j y_j}} \\ &= \frac{(pk_{TP})^{\sum_{j=1}^h r_j y_j} (pk_0)^{\sum_{j=1}^h W_{t+1}^j y_j}}{(pk_{TP})^{\sum_{j=1}^h r_j y_j}} \\ &= pk_0^{<\mathbf{W},\mathbf{y}>} \end{aligned}$$

where $\mathbf{W} = \{W_{t+1}^1, ..., W_{t+1}^h\}$. Then, A executes the *baby-step giant-step* algorithm to obtain $< \mathbf{W}, \mathbf{y} >$ as the updated global model θ_{t+1}.

Finally, the aggregation node A needs to verify whether the accuracy of the model θ_{t+1} meets the requirement of the task. If the accuracy of θ_{t+1} reaches the value defined by the task publisher, A modifies the status of the current task and returns the model to the task publisher to move into the next phase. Otherwise, the aggregation node A distributes the model update θ_{t+1} to each miner M_i for the next iteration. Therefore, the model training and model aggregation phases may be repeated several times until the accuracy of the model reaches the requirements.

5.2.4 Block Publishing

When completing a training task, the aggregation node A gets the rewards by publishing a new block in blockchain networks. First, the aggregation node A computes the hash $mh = h(\theta)$ of the trained global model θ and sends θ to an accessible platform (e.g., cloud server) while keeping the access link ml, which aims to allow each participant to verify the validity of θ.

Then, using the secret key sk_0, A generates a signature σ_2 about the model information, including the model hash mh, the access link ml, and the model accuracy ma. By leveraging two signatures (σ_1, σ_2), A builds a new block and broadcasts it to the participating miners. Receiving the new block B, each miner M_i verifies the correctness of the new block. If A publishes an error block, which means that the block may contain an error participant list or the accuracy does not meet the requirement, the new block will not be passed. After all participants have validated that the new block is correct, the block will be added to the blockchain networks in series. Obviously, if the trained model is valid, the block can be generated correctly, and the aggregation node A then obtains the rewards. Otherwise, A cannot get any rewards. Note that multiple tasks may be completed simultaneously, the aggregation node A needs to collect all running tasks locally and records the task status. Then A builds the block in order of task end time (timestamp) to prevent forks. We also show the process for the block publishing phase in Fig. 6.

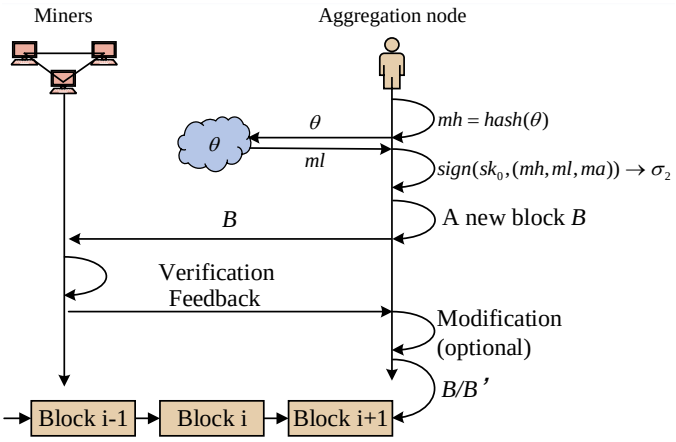


Fig. 6. Block publishing process.

5.2.5 Task Reward Releasing

When a new block is added to the blockchain network successfully, the participants contained by the new block will get rewards automatically by executing the smart contract deployed by the task publisher TP at the task publishing phase.

Each participant M_j can trigger the smart contract by sending the task information T, the new block B published by A, and the trained model θ, the pseudocode is shown in Algorithm 2. Based on the status of the training task $taskPool[T]$, the smart contract determines whether to set the rewards or issue the rewards. If $taskPool[T]$ is true, then the contract will create the transactions for each miner M_j (Line 5) when the $model.accuracy$ meets $T.acc$. If $taskPool[T]$ is false, it indicates that the task T has been resolved. After that, the smart contract will invite TP to initialize a new request T'. It is straightforward to see that each participant can get rewards as long as the accuracy of the trained model is valid and qualified.

6 Security Analysis

In this section, we theoretically analyze that the ESB-FL achieves the following design goals: confidentiality, efficiency, and fair payment.

6.1 Confidentiality

The confidentiality guarantees the privacy of miners' local data and prevents attackers from reconstructing the private medical data from data acquired task processing. The confidentiality of the ESB-FL relies on the security of the NDD-FE scheme used to encrypt local gradients.

Theorem 1. If the underlying NDD-FE scheme is secure, then the confidentiality of local training models sent to the aggregation node is guaranteed.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TBDATA.2022.3177170, IEEE Transactions on Big Data

Algorithm 2 The pseudocode of smart contract that guarantees task reward payment

Input: $T(pk_{TP}, S, D, L, acc, T_{exp}, R), model, B, taskPool, publishBlock$

if $taskPool[T]$ **then**
  $require(model)$
  if $model.accuracy > T.acc$ **then**
    $publishBlock[msg.sender] \leftarrow true$
    $MultiTransfer(T.pk_{TP}, T.R, B.participants)$
    $T.S \leftarrow finished$
    $taskPool[T] \leftarrow false$
  end if
else
  init T'
  $T'.pk_{TP} \leftarrow TP.ID$
  $T'.S \leftarrow publishing$
  $T'.D \leftarrow COVID19\ CT\ image\ datasets$
  $T'.L \leftarrow aa.bb.com$
  $T'.acc \leftarrow 97\%$
  $T'.T_{exp} \leftarrow 48\ h$
  $T'.R \leftarrow 1\ BTC$
  $require(!taskPool[T'])$
  if $getBalance(T'.pk_{TP}) \geq R$ **then**
    $taskPool[T'] \leftarrow true$
  else
    init faild
  end if
end if

$Proof$. In the model training phase, each miner M_j encrypts the gradient W_{t+1}^j by using the encryption algorithm of the NDD-FE scheme before sending to the aggregation node A. Based on the description of NDD-FE, the ciphertext of each gradient W_{t+1}^j is $U_{t+1}^j = pk_{TP}^{r_j} \cdot pk_0^{W_{t+1}^j}$, where $r_j = H_1(pk_{TP}^{sk_j}, crt, \mathbf{T})$. For any adversary, r_j is random and unknown since $pk_{TP}^{sk_j}$ is difficult to be calculated and $(pk_{TP}, pk_j, pk_{TP}^{sk_j})$ forms a CDH hard problem. Thus, U_{t+1}^j can be regarded as a standard ElGamal ciphertext so that any adversary cannot recover the information about W_{t+1}^j. Note that for the task publisher, although he can obtain $pk_0^{W_{t+1}^j}$ by using his secret key sk_{TP}, no information about W_{t+1}^j is leaked since sk_0 is unknown for him.

In the model aggregation phase, the aggregation node A can obtain all encrypted gradients $U_{t+1} = (U_{t+1}^1, U_{t+1}^2, ..., U_{t+1}^h)$ from h miners and holds the function key $sk_{\otimes} = \sum_{j=1}^{h} r_j y_j$. Based on the correctness and functionality of the FE scheme, the aggregation node A only obtain $\sum_{j=1}^{h} y_j W_{t+1}^j$ by decrypting the ciphertext U_j using the corresponding secret key $sk_{\otimes}$. However, since $y_i = \frac{1}{h}$, the probability of the aggregation node A guessing the specific gradient W_{t+1}^j correctly is negligible.

To sum up, the security of the NDD-FE scheme guarantees that any adversary cannot obtain any information about the local gradients of each miner, and thus the confidentiality of our framework is achieved.

6.2 Efficiency

The efficiency of the FL framework is the key factor for its wide application, and thus we have several ways to optimize the efficiency of our ESB-FL. On the one hand, ESB-FL avoids using computation-intensive tools such as secure MPC and HE, and uses a relatively lightweight function encryption scheme to protect data privacy. Compared with existing MPC-based or HE-based FL frameworks, ESB-FL is considerably more efficient in terms of computation and communication costs. For example, the size of the ciphertext in our framework is only $U_{t+1}^j \in G$, about 512 bits (ECDSA parameters). Apart from interaction for model training, each participant does not need to interact with others for achieving privacy protection, thereby reducing the communication costs. In addition, our framework adopts the DGC algorithm to further reduce communication costs. The gradients will be transmitted if and only if they become large enough, which can significantly reduce the number of rounds of interaction.

On the other hand, ESB-FL supports parallel publishing and processing of multiple FL tasks. Multiple tasks are effectively arranged for each miner M_i to work on, and the statuses of them will be broadcast in time. Finally, the aggregation node A will be rewarded by publishing new blocks, where multiple tasks are organized by the deadline of tasks.

6.3 Fair Payment

Fair payment is achieved by combining our designed blockchain and smart contracts. It mainly shows in the following two aspects: (1) the task publisher can obtain a valid model as long as he pays for his request, (2) each participant can get rewards as long as he participates in model training.

For the task publisher, at the task publishing phase, he first needs to publish the payment token and deploy a pay smart contract to complete the task release. Note that when deploying the smart contract, he must make sure that the corresponding account has a sufficient balance. When the trained model is published, the smart contract will judge whether the accuracy of the model meets the task requirements. If yes, the smart contract will create a reward transaction from the task publisher to the task participants. Otherwise, the new block cannot be generated and the smart contract is not triggered, and thus all participants cannot get any rewards.

For the participating miners, there are two types of roles: the aggregation node and the miner node. The aggregation node is rewarded by publishing a new block, while all miners are rewarded by smart contracts. If the trained model is correct and has been determined to meet the requirements, a new block is generated and the smart contract has triggered automatically, thereby completing reward payment. Note that the payment process cannot be stopped even for the creator of smart contracts.

Due to the characteristics of blockchain and smart contracts, such as immutability and automatic execution, the interests of both the task publisher and miners are guaranteed without a trusted party. Therefore, our framework achieves fair payment and sustains the activity levels of users.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TBDATA.2022.3177170, IEEE Transactions on Big Data

7 EXPERIMENTS

We conduct experiments in this section to check the performance of our proposed ESB-FL. We first introduce our experimental settings, and then give the experimental results. Meanwhile, we compare our FL scheme with existing representative work.

7.1 Experimental Settings

7.1.1 Configuration

We implement our ESB-FL by Python on a Linux server with Intel Xeon CPU E5-1650 v4 @ 3.60GHz, 64Gb RAM, GTX 1080 Ti. We adopt an open-source project[1] to provide the blockchain service. Our blockchain network topology is shown in Fig. 7, in our experiments, the number of registered miners is set to 100 and our medical image dataset is divided equally among these 100 miners. The DGC algorithm is implemented based on [23], the compression threshold of the DGC algorithm is set to 90%. The NDD-FE scheme is implemented based on FE[2].

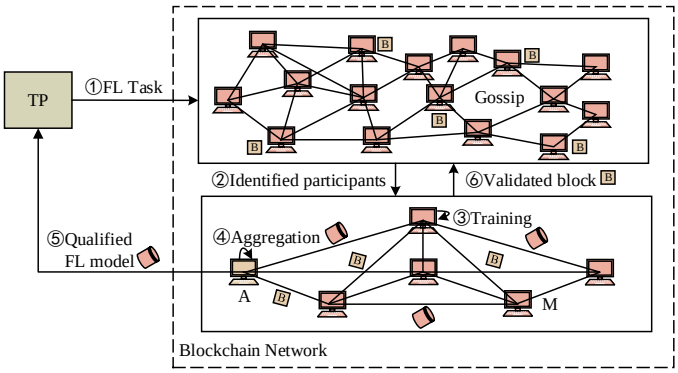


Fig. 7. Blockchain network topology.

7.1.2 Model

We refer to some convolutional neural network models to classify medical images. First, we use denseblock [48] network structure to extract features from medical images, the network contains 4 convolutional layers, the numbers of channels in each convolutional layer are 1×16, 16×16, 32×16, 48×16, and the size of the convolution kernel is 3×3. Then, we use SSD [3] network to locate and classify extracted features, which contains five localization and classification layers. The learning rate and batch size of the training model are set to 10^{-4} and 2, respectively. Meanwhile, we set the number of iterations to be large enough (such as 1000) to keep the miners training so that the aggregation model meets the required accuracy.

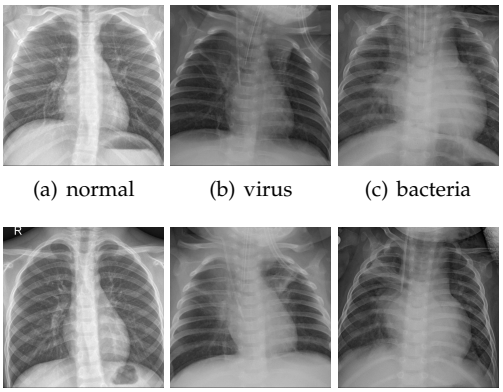
7.1.3 Dataset

In our framework, we use the chest X-ray image (pneumonia) dataset [4], partial samples are shown in Fig. 8. We use training samples as the basis of the generator and randomly generate 80,000 images for model training. We greyscale each image and set its size to 200 × 200. We test our trained model with 1,000 test images from this dataset, where normal, virus, and bacteria images are considered.

1. https://github.com/guanchao/mini_blockchain
2. https://github.com/OpenMined/PyFE
3. https://github.com/AIZOOTech/FaceMaskDetection
4. https://www.kaggle.com/paultimothymooney/chest-xray-pneumonia

TABLE 1
Time costs of blockchain operations

Blockchain operation	Ours	PEFL
Task publishing	< 1h	< 1h
Model Training (80,000 images)	4.3h	7.4h
Model Aggregation (10 nodes)	6.44 min	8.03min
Block Publishing	< 1h	< 1h
Task Reward	< 1min	< 1min



(a) normal (b) virus (c) bacteria

(d) normal (e) virus (f) bacteria

Fig. 8. The chest X-ray image dataset used in our proposed framework. (a) and (d), (b) and (e), (c) and (f) are normal images, virus images and bacteria images, respectively.

7.1.4 Comparatives

We compare the performance of our FL scheme with several representative approaches, including PEFL [49], Lu's [50] and Kumar's [51]. We use the same dataset to train their FL frameworks and analyze their experimental results.

7.2 Results

7.2.1 Blockchain Operations

Blockchain technology is applied to ensure security and solve the payment problem, its performance has a significant impact on the practicality of our ESB-FL framework. We first test the time costs of main blockchain operations in each phase, the results are the average values of running 100 times over 80,000 images. As shown in Table 1, in the phases such as task publishing, blockchain publishing, and task reward, our framework and the FL scheme based on homomorphic encryption (PEFL) take approximately the same length of time. However, in terms of model training and aggregation, our framework saves more time cost compared with PEFL. For example, the model training in our framework only takes 4.3 hours, while it takes about 7.4 hours in PEFL. In addition, since the task selection and new block publishing do not involve any encryption operations, the time costs of these stages are not affected in either case.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TBDATA.2022.3177170, IEEE Transactions on Big Data

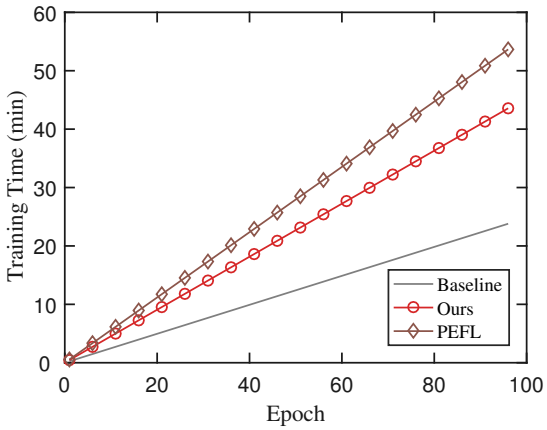


Fig. 9. Training time with different epoches.

TABLE 2
Time costs of encryption scheme

Operations (for a gradient)	Ours (NDD-FE)	PEFL (HE)
Encryption	23.33 s	45.28 s
Decryption	37.44 s	47.51 s

7.2.2 FL Training

We compare the efficiency of our framework based on NDD-FE with the PEFL scheme. The baseline scheme is an FL framework that does not use encryption algorithms. Fig. 9 shows the training time accumulated as the epoch increased in FL. We can find that the time cost of our framework is higher than the baseline, which can be regarded as a tradeoff to improve security. However, the performance of our framework is higher than that of the PEFL framework. This is because the aggregation node in the PEFL framework needs to interact with the task publisher to decrypt the aggregation model frequently. While the aggregation node in our framework can directly decrypt the aggregation model while maintaining the privacy of the model.

Besides, we also test the security costs of our framework and the PEFL framework, the results are shown in Table 2. Clearly, the NDD-FE scheme proposed in this paper is more efficient than the FE scheme adopted by PEFL in terms of performance. For example, the encryption algorithm of the NDD-FE scheme only takes 23.33 seconds, while that of the HE scheme takes about 45.28 seconds.

7.2.3 FL Testing

The normal, virus and bacteria images are considered to be tested in the experiment. Using deep learning models to detect lung medical images is helpful for rapid localization and accurate identification of lung lesions in patients with pneumonia, which greatly reduces the burden of doctors and is of great significance for clinical treatment.

Fig.10 shows the test results of our model after training on the complete data set. Columns 1, 2, and 3 are normal images, virus images, and bacteria images, respectively. We can observe that the test results for each set of images are very accurate.

Fig. 11 further shows the results of our FL model on 1,000 test datasets. We can discover that the test accuracy

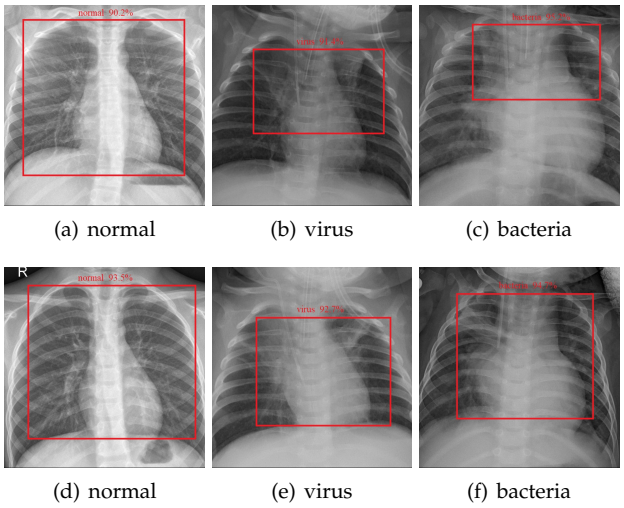


Fig. 10. Test results of our FL model. (a) and (d), (b) and (e), (c) and (f) are the test results of normal images, virus images and batceria images respectively.

of normal images, virus images and bacteria images are 93.3%, 90%, and 96.1% respectively, which further verified the effectiveness of our FL scheme.

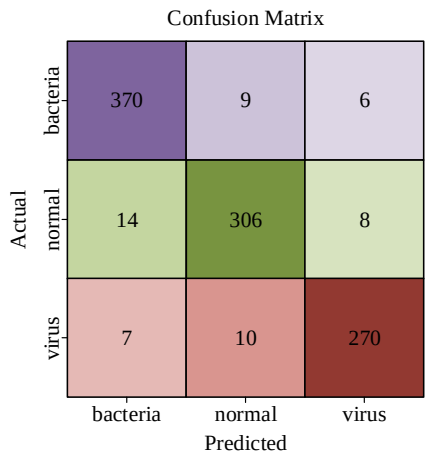


Fig. 11. Confusion matrix of test dataset.

7.3 Comparative Fusion Performance

7.3.1 Privacy-preserving Cost

We test the cumulative model accuracy of our framework and other advanced FL frameworks[49], [50], [51], as shown in Fig. 12. Compared with the unencrypted FL scheme, our framework will lose 1-2% accuracy due to the introduction of the DGC algorithm and NDD-FE, while the PEFL framework will lose 5-6% accuracy due to the polynomial activation function and the encryption algorithm error. Other FL frameworks use non-cryptographic DP technology to protect data privacy, their model accuracies are affected by the introduced DP noise. Therefore, while under the same security condition, the accuracy of our framework will be

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TBDATA.2022.3177170, IEEE Transactions on Big Data

higher than that of the HE-based framework. Furthermore, our framework still performs better than some advanced FL frameworks based on DP technology.

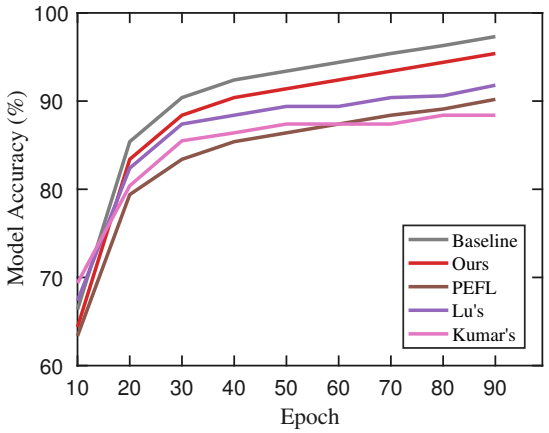


Fig. 12. Model accuracy with different epoches.

7.3.2 Communication and Computation consumption

We compare the communication and computation costs of our scheme with PEFL, Lu's and Kumar's. The baseline scheme is our FL framework without the DGC algorithm. Fig. 13 shows the communication and computation consumption of our framework and some other advanced FL frameworks. As can be seen from the figure, the performance of baseline is worse than PEFL and Lu's scheme, requiring longer communication and computation time. After adopting the DGC compression algorithm, our framework becomes the most efficient method, which further confirms that our framework can control communication and computing costs well, and can be applied in practical applications to improve model accuracy and computing efficiency.

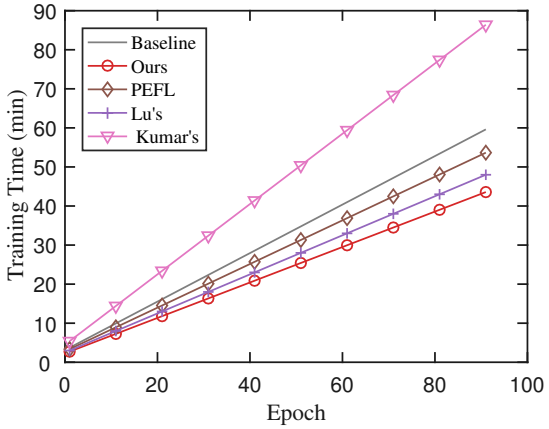


Fig. 13. Running time of different FL schemes.

Through the above evaluation, we can observe that our scheme not only improves communication efficiency, but also ensures the secure aggregation of gradient data, which well solves the privacy leakage and communication problems that may be encountered in FL. Parallel local training in FL can greatly improve the efficiency of centralized training and ensure stable model accuracy. With the increase of epoch, the time of encryption and update transmission in the model training process also accumulates, but it increases the quality of the detection model.

8 Conclusion

Leveraging FL into smart healthcare offers new ways to improve the quality of medical services. FL can train a high-precision detection model while keeping all the training data on local devices. This paper presents an efficient and secure blockchain-based FL framework called ESB-FL. ESB-FL effectively solves the privacy, communication, and payment problems of the existing FL frameworks. The computation and communication costs are reduced by adopting the NDD-FE scheme and DGC algorithm. The blockchain technology in our framework is used to address the fair payment problem between FL task publishers and miners. The security analysis and extensive experiments of ESB-FL are conducted in this paper. The results show that ESB-FL achieves enhanced security and efficient communication in implementing FL for multiple hospital nodes without the involvement of a third party.

Acknowledgments

This work was supported by the National Natural Science Foundation of China under Grants U20A20176, U21A20463, 62102050, 62102052, China Postdoctoral Science Foundation under Grant BX2021399, Guangxi Key Laboratory of Trusted Software under Grant KX202043, and State Key Laboratory of Cryptology under Grant MMKFKT202118, the Natural Science Foundation of Chongqing, China, under Grant cstc2021jcyj-msxmX0744.

References

[1] D. Sui, Y. Chen, J. Zhao, Y. Jia, Y. Xie, and W. Sun, "Feded: Federated learning via ensemble distillation for medical relation extraction," in *Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2020, pp. 2118–2128.

[2] Y.-H. Wu, S.-H. Gao, J. Mei, J. Xu, D.-P. Fan, R.-G. Zhang, and M.-M. Cheng, "JCS: An explainable COVID-19 diagnosis system by joint classification and segmentation," *IEEE Transactions on Image Processing*, vol. 30, pp. 3113–3126, 2021.

[3] J. Geiping, H. Bauermeister, H. Dröge, and M. Moeller, "Inverting gradients-how easy is it to break privacy in federated learning?" *Advances in Neural Information Processing Systems (NeurIPS)*, vol. 33, pp. 16 937–16 947, 2020.

[4] L. Zhu, Z. Liu, and S. Han, "Deep leakage from gradients," *Advances in Neural Information Processing Systems (NeurIPS)*, vol. 32, 2019.

[5] R. Shokri, M. Stronati, C. Song, and V. Shmatikov, "Membership inference attacks against machine learning models," in *IEEE Symposium on Security and Privacy (SP)*, 2017, pp. 3–18.

[6] H. Chen, H. Li, G. Dong, M. Hao, G. Xu, X. Huang, and Z. Liu, "Practical membership inference attack against collaborative inference in industrial iot," *IEEE Transactions on Industrial Informatics*, vol. 18, no. 1, pp. 477–487, 2022.

[7] M. Zhang, Y. Zhang, and G. Shen, "Ppdds: A privacy-preserving disease diagnosis scheme based on the secure mahalanobis distance evaluation model," *IEEE Systems Journal*, pp. 1–11, 2021.

[8] M. Gong, Y. Xie, K. Pan, K. Feng, and A. K. Qin, "A survey on differentially private machine learning," *IEEE Computational Intelligence Magazine*, vol. 15, no. 2, pp. 49–64, 2020.

[9] M. Hao, H. Li, X. Luo, G. Xu, H. Yang, and S. Liu, "Efficient and privacy-enhanced federated learning for industrial artificial intelligence," *IEEE Transactions on Industrial Informatics*, vol. 16, no. 10, pp. 6532–6542, 2020.

[10] J. So, B. Guler, and S. Avestimehr, "A scalable approach for privacy-preserving collaborative machine learning," *Advances in Neural Information Processing Systems (NeurIPS)*, vol. 33, pp. 8054–8066, 2020.

[11] K. Bonawitz, V. Ivanov, B. Kreuter, A. Marcedone, H. B. McMahan, S. Patel, D. Ramage, A. Segal, and K. Seth, "Practical secure aggregation for privacy-preserving machine learning," in *ACM SIGSAC Conference on Computer and Communications Security (CCS)*, 2017, pp. 1175–1191.

[12] M. Hao, H. Li, X. Luo, G. Xu, H. Yang, and S. Liu, "Efficient and privacy-enhanced federated learning for industrial artificial intelligence," *IEEE Transactions on Industrial Informatics*, vol. 16, no. 10, pp. 6532–6542, 2019.

[13] Y. Chen, X. Qin, J. Wang, C. Yu, and W. Gao, "Fedhealth: A federated transfer learning framework for wearable healthcare," *IEEE Intelligent Systems*, vol. 35, no. 4, pp. 83–93, 2020.

[14] V. Mugunthan, A. Polychroniadou, D. Byrd, and T. H. Balch, "Smpai: Secure multi-party computation for federated learning," in *NeurIPS Workshop on Robust AI in Financial Services*, 2019.

[15] V. Goyal, A. Jain, Z. Jin, and G. Malavolta, "Statistical zaps and new oblivious transfer protocols," in *International Conference on the Theory and Applications of Cryptographic Techniques (EUROCRYPT)*, 2020, pp. 668–699.

[16] T. Song, Y. Tong, and S. Wei, "Profit allocation for federated learning," in *IEEE International Conference on Big Data*, 2019, pp. 2577–2586.

[17] Y. Liu, Z. Ai, S. Sun, S. Zhang, Z. Liu, and H. Yu, "Fedcoin: A peer-to-peer payment system for federated learning," in *Federated Learning*, 2020, pp. 125–138.

[18] Y. Zhan, J. Zhang, Z. Hong, L. Wu, P. Li, and S. Guo, "A survey of incentive mechanism design for federated learning," *IEEE Transactions on Emerging Topics in Computing*, 2021.

[19] X. Bao, C. Su, Y. Xiong, W. Huang, and Y. Hu, "Flchain: A blockchain for auditable federated learning with trust and incentive," in *International Conference on Big Data Computing and Communications (BIGCOM)*, 2019, pp. 151–159.

[20] H. Kim, J. Park, M. Bennis, and S.-L. Kim, "Blockchained on-device federated learning," *IEEE Communications Letters*, vol. 24, no. 6, pp. 1279–1283, 2019.

[21] R. Kumar, A. A. Khan, J. Kumar, N. A. Golilarz, S. Zhang, Y. Ting, C. Zheng, W. Wang *et al.*, "Blockchain-federated-learning and deep learning models for COVID-19 detection using ct imaging," *IEEE Sensors Journal*, vol. 21, no. 14, pp. 16 301–16 314, 2021.

[22] Y. Qu, L. Gao, T. H. Luan, Y. Xiang, S. Yu, B. Li, and G. Zheng, "Decentralized privacy using blockchain-enabled federated learning in fog computing," *IEEE Internet of Things Journal*, vol. 7, no. 6, pp. 5171–5183, 2020.

[23] Y. Lin, S. Han, H. Mao, Y. Wang, and W. J. Dally, "Deep gradient compression: Reducing the communication bandwidth for distributed training," *arXiv preprint arXiv:1712.01887*, 2017.

[24] L. Zelei, C. Yuanyuan, Z. Yansong, Y. Han, L. Yang, B. Renyi, J. Jinpeng, N. Zaiqing, X. Qian, and Y. Qiang, "Contribution-aware federated learning for smart healthcare," in *Proceedings of the 34th Annual Conference on Innovative Applications of Artificial Intelligence (IAAI-22)*, 2022.

[25] K. Wei, J. Li, M. Ding, C. Ma, H. H. Yang, F. Farokhi, S. Jin, T. Q. Quek, and H. V. Poor, "Federated learning with differential privacy: Algorithms and performance analysis," *IEEE Transactions on Information Forensics and Security*, vol. 15, pp. 3454–3469, 2020.

[26] H. Zheng, H. Hu, and Z. Han, "Preserving user privacy for machine learning: local differential privacy or federated machine learning?" *IEEE Intelligent Systems*, vol. 35, no. 4, pp. 5–14, 2020.

[27] M. Kim, O. Günlü, and R. F. Schaefer, "Federated learning with local differential privacy: Trade-offs between privacy, utility, and communication," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2021, pp. 2650–2654.

[28] Y. Li, H. Li, G. Xu, T. Xiang, and R. Lu, "Practical privacy-preserving federated learning in vehicular fog computing," *IEEE Transactions on Vehicular Technology*, 2022.

[29] I. Damgård, D. Escudero, T. Frederiksen, M. Keller, P. Scholl, and N. Volgushev, "New primitives for actively-secure mpc over rings with applications to private machine learning," in *IEEE Symposium on Security and Privacy (SP)*, 2019, pp. 1102–1120.

[30] J. Zhang, B. Chen, S. Yu, and H. Deng, "PEFL: A privacy-enhanced federated learning scheme for big data analytics," in *IEEE Global Communications Conference (GLOBECOM)*, 2019, pp. 1–6.

[31] C. Zhang, S. Li, J. Xia, W. Wang, F. Yan, and Y. Liu, "{BatchCrypt}: Efficient homomorphic encryption for {Cross-Silo} federated learning," in *USENIX Annual Technical Conference (USENIX ATC)*, 2020, pp. 493–506.

[32] H. Chaudhari, R. Rachuri, and A. Suresh, "Trident: Efficient 4pc framework for privacy preserving machine learning," *arXiv preprint arXiv:1912.02631*, 2019.

[33] S. D. Gordon, S. Ranellucci, and X. Wang, "Secure computation with low communication from cross-checking," in *International Conference on the Theory and Application of Cryptology and Information Security (ASIACRYPT)*, 2018, pp. 59–85.

[34] M. S. Riazi, K. Laine, B. Pelton, and W. Dai, "HEAX: An architecture for computing on encrypted data," in *International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS)*, 2020, pp. 1295–1309.

[35] R. Xu, J. B. Joshi, and C. Li, "CryptoNN: Training neural networks over encrypted data," in *International Conference on Distributed Computing Systems (ICDCS)*, 2019, pp. 1199–1209.

[36] T.-I. Szatmari, M. K. Petersen, M. J. Korzepa, and T. Giannetsos, "Modelling audiological preferences using federated learning," in *ACM Conference on User Modeling, Adaptation and Personalization (UMAP)*, 2020, pp. 187–190.

[37] S. Guo, X. Zhang, F. Yang, T. Zhang, Y. Gan, T. Xiang, and Y. Liu, "Robust and privacy-preserving collaborative learning: A comprehensive survey," 2021. [Online]. Available: https://arxiv.org/abs/2112.10183

[38] Y. Qu, L. Gao, T. H. Luan, Y. Xiang, S. Yu, B. Li, and G. Zheng, "Decentralized privacy using blockchain-enabled federated learning in fog computing," *IEEE Internet of Things Journal*, vol. 7, no. 6, pp. 5171–5183, 2020.

[39] 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 Transactions on Vehicular Technology*, vol. 69, no. 4, pp. 4298–4311, 2020.

[40] Y. Wan, Y. Qu, L. Gao, and Y. Xiang, "Privacy-preserving blockchain-enabled federated learning for B5G-driven edge computing," *Computer Networks*, p. 108671, 2021.

[41] 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 of Things Journal*, vol. 6, no. 6, pp. 10 700–10 714, 2019.

[42] P. Ramanan and K. Nakayama, "Baffle: Blockchain based aggregator free federated learning," in *2020 IEEE International Conference on Blockchain (Blockchain)*, 2020, pp. 72–81.

[43] J. Zhang, Y. Wu, and R. Pan, "Incentive mechanism for horizontal federated learning based on reputation and reverse auction," in *The Web Conference (WWW)*, 2021, pp. 947–956.

[44] L. Gao, L. Li, Y. Chen, W. Zheng, C. Xu, and M. Xu, "FIFL: A fair incentive mechanism for federated learning," in *International Conference on Parallel Processing (ICPP)*, 2021, pp. 1–10.

[45] A. Abdaoui, A. Erbad, A. Al-Ali, A. Mohamed, and M. Guizani, "Fuzzy elliptic curve cryptography for authentication in internet of things," *IEEE Internet of Things Journal*, 2021.

[46] E. Dufour-Sans, R. Gay, and D. Pointcheval, "Reading in the dark: Classifying encrypted digits with functional encryption," Cryptology ePrint Archive, Report 2018/206, 2018.

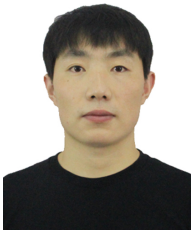
[47] A. Kiayias, A. Russell, B. David, and R. Oliynykov, "Ouroboros: A provably secure proof-of-stake blockchain protocol," in *Annual International Cryptology Conference (CRYPTO)*, 2017, pp. 357–388.

[48] H. Li and X.-J. Wu, "DenseFuse: A fusion approach to infrared and visible images," *IEEE Transactions on Image Processing*, vol. 28, no. 5, pp. 2614–2623, 2019.

[49] M. Hao, H. Li, X. Luo, G. Xu, H. Yang, and S. Liu, "Efficient and privacy-enhanced federated learning for industrial artificial intelligence," *IEEE Transactions on Industrial Informatics*, vol. 16, no. 10, pp. 6532–6542, 2020.

[50] Y. Lu, X. Huang, Y. Dai, S. Maharjan, and Y. Zhang, "Blockchain and federated learning for privacy-preserved data sharing in industrial IoT," *IEEE Transactions on Industrial Informatics*, vol. 16, no. 6, pp. 4177–4186, 2020.

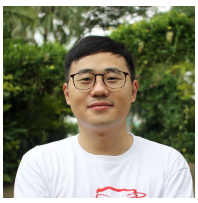
[51] R. Kumar, A. A. Khan, J. Kumar, Zakria, N. A. Golilarz, S. Zhang, Y. Ting, C. Zheng, and W. Wang, "Blockchain-federated-learning and deep learning models for COVID-19 detection using CT imaging," *IEEE Sensors Journal*, vol. 21, no. 14, pp. 16 301–16 314, 2021.



Biwen Chen received the Ph.D. degree from School of Computer, Wuhan University in 2020. He is currently an assistant professor of the School of Computer, Chongqing University. His main research interests include cryptography, information security and blockchain.

Honghong Zeng received the B.E. degree from the School of Information Engineering, Nanchang University, China, where she is currently pursuing the master's degree at the College of Computer Science at Chongqing University. Her research interests include blockchain and secure medical image processing.

Tao Xiang received the BEng, MS, and PhD degrees in computer science from Chongqing University, China, in 2003, 2005, and 2008, respectively. He is currently a professor with the College of Computer Science at Chongqing University, China. His research interests include multimedia security, cloud security, data privacy and cryptography. He has published more than 100 papers on international journals and conferences. He also served as a referee for numerous international journals and conferences.



Shangwei Guo is an associate professor in College of Computer Science, Chongqing University. He received the Ph.D. degree in computer science from Chongqing University, Chongqing, China at 2017. He worked as a postdoctoral research fellow at Hong Kong Baptist University and Nanyang Technological University from 2018 to 2020. His research interests include secure deep learning, secure cloud/edge computing, and database security.

Tianwei Zhang is an assistant professor in School of Computer Science and Engineering, at Nanyang Technological University. His research focuses on computer system security. He is particularly interested in security threats and defenses in machine learning systems, autonomous systems, computer architecture and distributed systems. He received his Bachelor's degree at Peking University in 2011, and the Ph.D degree in at Princeton University in 2017.

Yang Liu received the B.Comp. degree (Hons.) from the National University of Singapore (NUS) in 2005 and the Ph.D. degree from NUS and MIT, in 2010. He started his postdoctoral work in NUS and MIT. In 2012, he joined Nanyang Technological University (NTU). He is currently a Full Professor and the Director of the Cybersecurity Laboratory, NTU. He specializes in software verification, security, and software engineering. His research has bridged the gap between the theory and practical usage of formal methods and program analysis to evaluate the design and implementation of software for high assurance and security. By now, he has more than 270 publications in top tier conferences and journals. He received a number of prestigious awards, including the MSRA Fellowship, the TRF Fellowship, the Nanyang Assistant Professor, the Tan Chin Tuan Fellowship, the Nanyang Research Award, and eight best paper awards in top conferences, such as ASE, FSE, and ICSE.