doi:10.3772/j.issn.1006-6748.2022.04.001

The adaptive distributed learning based on homomorphic encryption and blockchain①

YANG Ruizhe(杨睿哲)*, ZHAO Xuehui*, ZHANG Yanhua②*, SI Pengbo*, TENG Yinglei**
(*Faculty of Information Technology, Beijing University of Technology, Beijing 100124, P. R. China)
(**School of Electronic Engineering, Beijing University of Posts and Telecommunications, Beijing 100876, P. R. China)

Abstract

The privacy and security of data are recently research hotspots and challenges. For this issue, an adaptive scheme of distributed learning based on homomorphic encryption and blockchain is proposed. Specifically, in the form of homomorphic encryption, the computing party iteratively aggregates the learning models from distributed participants, so that the privacy of both the data and model is ensured. Moreover, the aggregations are recorded and verified by blockchain, which prevents attacks from malicious nodes and guarantees the reliability of learning. For these sophisticated privacy and security technologies, the computation cost and energy consumption in both the encrypted learning and consensus reaching are analyzed, based on which a joint optimization of computation resources allocation and adaptive aggregation to minimize loss function is established with the realistic solution followed. Finally, the simulations and analysis evaluate the performance of the proposed scheme.

Key words: blockchain, distributed machine learning (DML), privacy, security

0 Introduction

Today, people and Internet of Things are generating unprecedented amounts of data. Based on these data, machine learning as a data analytics tool learns from data, recognizes patterns, and makes decisions, becoming an important part of artificial intelligence (AI). However, collecting abundant data to the central server is usually expensive and time consuming, or even impossible due to privacy and security issues. Therefore, distributed machine learning (DML) as an alternative approach has attracted more and more attention[1]. DML allows the owner to retain the original data, compute and update the model parameter locally using its data, which is periodically transmitted to the central server. Subsequently, the central server updates the global model by aggregating the received local parameter. In this way, the privacy of the original data is protected, and also the huge cost of data collection is avoided. However, there are still cyber risks in the interaction and message transmission, and potential malicious attacks can still extract some of their data information.

In order to resist attacks and enhance privacy, differential privacy and cryptography technologies are introduced in the existing work. Differential privacy (DP) can protect the sensitive information by adding random noise, therefore, attackers cannot infer from the results of some random algorithms[2-3]. However, this differential noise needs to be smaller than the random noise of the sample, otherwise it will adversely affect the accuracy and privacy of the learning model[4]. On the other side, cryptography with a mechanism of encryption can also protect the confidentiality of sensitive data. Therein, secure multiparty computing (SMC) and homomorphic encryption (HE) are the two main technologies. Comparatively, HE processing the underlying plaintext data in encrypted form without decryption is less complicated. In the current studies[5-7], the solutions using partial homomorphic encryption to protect the privacy in DML are more effective than those based on SMC. Considering the efficiency, Ref. [8] proposed an HE-based batch encryption technology to solve the encryption and communication bottleneck problems.

In the DML system, another challenge is security. Ref. [9] analyzed the potential threats, including

① Supported by the National Natural Science Foundation of China (No. 62171062), Foundation of Beijing Municipal Commission of Education (No. KM202010005017, KM202110005021) and Beijing Natural Science Foundation (No. L211002).
② To whom correspondence should be addressed. E-mail: zhangyh@bjut.edu.cn
Received on Mar. 10, 2022

semi-honest and malicious internal and external attacks. Some researchers have proposed distributed learning systems based on blockchain[10-11]. Blockchain is a representative paradigm of distributed control technology. It provides a peer-to-peer network that enables sensitive participants to communicate and collaborate in a secure manner without the need for a fully trusted third party or an authorized central node. Ref. [10] proposed a blockchain based decentralized learning system for medical data and utilized blockchain as an information exchange platform. Refs[12-14] presented blockchain based distributed learning processes to resist different threats.

Currently the attention to both privacy and security in DML has become a key issue, however, the problems of computation and energy consumption brought by sophisticated privacy protection technology have been ignored in the most existing work. Here, a valuable business model is considered, in which the computing party as the beneficiary hires participants to jointly learn a desired global model. Specifically, the computing party aggregates the local models of the data from participants, but doesn't share the aggregated results with any participants, so that it owns the value of the model. Therefore, an adaptive distributed machine learning scheme based on blockchain and homomorphic encryption is proposed. First, the Paillier encryption algorithm is used in the linear regression mode, where the computing party issues the global ciphertext parameters, and the participants always make ciphertext updates, so that the privacy of both the data and the model are satisfied. Second, the blockchain is introduced to complete the privacy-protected distributed linear regression process, which ensures the integrity and correctness of the updating and the final results of the model. Finally and most importantly, the total energy consumption of the system are analyzed from the perspective of computation complexity and computation resources, based on which a joint optimization of the computation resources allocation and adaptive aggregation in the case of limited system energy is given. Analysis and simulation results show the efficiency of the scheme in terms of security, privacy and learning convergence.

1 System model

In this section, a distributed learning framework is proposed based on homomorphic encryption and blockchain, which can complete the trusted learning process of the model without leaking private information.

1.1 System overview

The system completes a secure and private distributed learning based on the blockchain network, as shown in Fig. 1. The system model includes two roles: the participants owning their private data and the computing party having requirements of data analytics (commercial requirements, etc.). Therefore, the computing party initiates the collaboration with participants to derive a global model from their data. Since the local model parameters of each participant are sensitive information having certain commercial value for others, the partial homomorphic encryption algorithm Paillier is used to encrypt them providing privacy protection. Besides, these ciphertext model parameters are uploaded to the blockchain, and the underlying distributed consensus guarantees the data sharing of model parameters in a reliable manner, thereby improving the transparency and trace ability of the learning.

Specifically, it is assumed that the data required by computing party is distributed among participants, then the system nodes composed of the participants $\boldsymbol{P}$ and the computing party C are denoted as $\boldsymbol{I} = \{\boldsymbol{P}, C\}$, $|\boldsymbol{I}| = N + 1$, where $|\cdot|$ represents size of sets. In the set $\boldsymbol{P} = \{P_1, P_2, \cdots, P_N\}$, the element $P_i (i \in 1, 2, \cdots, N)$ represents the participant having a sub-data set $\boldsymbol{D}_i$ and the total data set is $\boldsymbol{D} = \{\boldsymbol{D}_1, \boldsymbol{D}_2, \cdots, \boldsymbol{D}_N\}$.

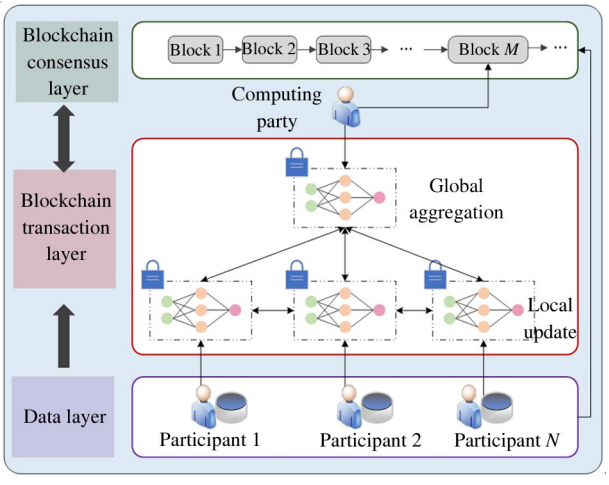

Fig. 1 System model

1.2 Distributed gradient descent

For simplicity, the distributed learning of linear regression model is concerned. For each data sample $(\boldsymbol{x}_{ij}, \boldsymbol{y}_{ij})$, the loss function can be expressed as

$$f(\boldsymbol{w}_i, \boldsymbol{x}_{ij}, \boldsymbol{y}_{ij}) = \frac{1}{2}(\boldsymbol{y}_{ij} - \boldsymbol{w}_i^{\mathrm{T}}\boldsymbol{x}_{ij})^2 \quad (1)$$

where $\boldsymbol{x}_{ij}$ and $\boldsymbol{y}_{ij}$ respectively denote the j-th input vector and output of dataset $\boldsymbol{D}_i$, and $\boldsymbol{w}_i$ is the local parameter. Then the local loss function of P_i based on each dataset

$\boldsymbol{D}_i$ is

$$F_i(\boldsymbol{w}_i) = \frac{1}{|\boldsymbol{D}_i|}\sum_j f(\boldsymbol{w}_i, \boldsymbol{x}_{ij}, \boldsymbol{y}_{ij}) \tag{2}$$

Accordingly, the global loss function is defined as

$$F(\boldsymbol{w}) = \frac{\sum_{i=1}^{N} |\boldsymbol{D}_i| F_i(\boldsymbol{w})}{|\boldsymbol{D}|} \tag{3}$$

where $\boldsymbol{w}$ is the global parameter.

During the learning process, $t(t = 1, 2, \cdots, T)$ is set as the number of iterations, and T is the total number of iterations. In each iteration, it contains a local update and a possible global aggregation. When $t = 0$, the local parameters of all participants are initialized to the same value. When $t > 0$, the local model $\boldsymbol{w}_i(t)$ of P_i is computed according to the gradient descent (GD) rule based on its previous iteration. After certain local updates, the computing party performs global aggregation to derive a global model, which is a weighted average of the local models of all participants.

After global aggregation, the local model $\boldsymbol{w}_i(t)$ at P_i usually changes. For convenience, $\widetilde{\boldsymbol{w}}_i(t)$ represents the local model at P_i after the updating. If no global aggregation is performed at iteration t, $\widetilde{\boldsymbol{w}}_i(t) = \boldsymbol{w}_i(t)$ is set. If global aggregation is performed at iteration t, then $\widetilde{\boldsymbol{w}}_i(t) \neq \boldsymbol{w}_i(t)$ and $\widetilde{\boldsymbol{w}}_i(t) = \boldsymbol{w}(t)$ is set.

Therefore, the local update at P_i using gradient descent algorithm with local loss function can be described as

$$\boldsymbol{w}_i(t) = \widetilde{\boldsymbol{w}}_i(t-1) - \eta \nabla F_i(\widetilde{\boldsymbol{w}}_i(t-1)) \tag{4}$$

where $\eta > 0$ is the learning rate. If global aggregation is performed by computing party C at iteration t, the global model is defined as Eq. (5).

$$\boldsymbol{w}(t) = \frac{\sum_{i=1}^{N} |\boldsymbol{D}_i| \boldsymbol{w}_i(t)}{|\boldsymbol{D}|} \tag{5}$$

1.3 Learning based on blockchain and Paillier

It is defined that there are τ local updates between every two global aggregation. It is assumed that T is a multiple of τ, and G is the total number of global aggregations, then $T = G\tau$.

In a cryptographic system, the public key Pk is used for encryption and the private key Sk is used for decryption. Here, $[\![\cdot]\!]$ is used to represent the ciphertext message encrypted by Paillier. During the encrypted learning, the computing party owns the public key and the private key to obtain the global model, while the participants only hold the public key, so that the model parameters are only shared among the participants in the form of ciphertext, ensuring the privacy of the system model.

The encrypted learning process can be divided into model initialization, local update, and global aggregation.

(1) Model initialization. The computing party and participants as the blockchain nodes, form a point-to-point decentralized network. Before learning, the key pair (Pk, Sk) is generated, then all nodes agree on the Pk and the hyperparameters of the learning model, such as the learning rate and the initial local ciphertext model, etc..

(2) Local update. In the ciphertext state, the participants use the gradient descent algorithm to update the local model in the current iteration based on the local ciphertext model completed in the previous iteration. During the encrypted updating, the participants compute local gradient using the homomorphic property $[\![am_1 + bm_2]\!] = [\![m_1]\!]^a \cdot [\![m_2]\!]^b$. At iteration t, the local ciphertext gradient of P_i can be written as

$$[\![\nabla F_i(\boldsymbol{w}_i(t))]\!] = [[\![\nabla_t^{i,1}]\!], [\![\nabla_t^{i,2}]\!], \cdots, [\![\nabla_t^{i,k}]\!] \cdots] \tag{6}$$

where $[\![\nabla_t^{i,k}]\!]$ is the ciphertext gradient of the local loss function $F_i(\boldsymbol{w}_i(t))$ to the k-th element in $\boldsymbol{w}_i(t) = [w_{i,1}(t), w_{i,2}(t), \cdots, w_{i,k}(t), \cdots]$. Then $[\![\nabla_t^{i,k}]\!]$ can be derived as

$$[\![\nabla_t^{i,k}]\!] = \left[\!\!\left[\frac{\partial F_i(\boldsymbol{w}_i(t))}{\partial w_{i,k}(t)} \right]\!\!\right] = \left[\!\!\left[-\frac{1}{|\boldsymbol{D}_i|}\sum_j \nabla_t^{ij,k} \right]\!\!\right] = \left(\prod_{j \in \boldsymbol{D}_i} [\![\nabla_t^{ij,k}]\!]\right)^{\frac{1}{|\boldsymbol{D}_i|}} \bmod n^2 \tag{7}$$

where n is the parameter for the key[15], and $[\![\nabla_t^{ij,k}]\!]$ is the ciphertext gradient corresponding to sample $x_{ij,k}$ and can be obtained by

$$\begin{aligned} [\![\nabla_t^{ij,k}]\!] &= \left[\!\!\left[\frac{\partial f(\boldsymbol{w}_i(t), \boldsymbol{x}_{ij}, y_{ij})}{\partial w_{i,k}(t)} \right]\!\!\right] \\ &= [\![-(y_{ij} - \boldsymbol{w}_i^{\mathrm{T}}(t)\boldsymbol{x}_{ij}) x_{ij,k}]\!] \\ &= [\![y_{ij}]\!]^{x_{ij,k}} \left(\prod_{j \in \boldsymbol{D}_i} [\![w_{i,k}(t)]\!]^{x_{ij,k}}\right)^{-x_{ij,k}} \bmod n^2 \end{aligned} \tag{8}$$

Based on the above theory and Eq. (4), the element $[\![w_{i,k}(t)]\!]$ in the local ciphertext model of P_i at iteration t can be expressed as

$$\begin{aligned} [\![w_{i,k}(t)]\!] &= [\![\widetilde{w}_{i,k}(t-1) - \eta \nabla_{t-1}^{i,k}]\!] \\ &= [\![\widetilde{w}_{i,k}(t-1)]\!] \times [\![\nabla_{t-1}^{i,k}]\!]^{-\eta} \bmod n^2 \end{aligned} \tag{9}$$

Using Eqs (8), (7) and (9), the participants can obtain $[\![\boldsymbol{w}_i(t)]\!]$. After τ local updates, each participant encapsulates local ciphertext model into a transaction $ELW_{P_i} = (t, [\![\boldsymbol{w}_i(t)]\!])$ and broadcasts it to other nodes.

(3) Global aggregation. The computing party C performs global aggregation after τ local updates. Specifically, the computing party collects N transactions $ELW_{P_1}, \cdots, ELW_{P_N}$ from the participants to obtain the

local ciphertext model $[\![\boldsymbol{w}_i(t)]\!]$ at iteration t, and updates the global model in the encryption as

$$[\![\boldsymbol{w}(t)]\!] = \left(\prod_{i=1}^{N} [\![\boldsymbol{w}_i(t)]\!]^{|\boldsymbol{D}_i|} \right)^{\frac{1}{|\boldsymbol{D}|}} \tag{10}$$

Then, the consensus based on PBFT protocol is performed. It is considered that C as the primary node, and it is responsible to pack transactions containing the global ciphertext model and the received local ciphertext model into the new block B_g

$$B_t = ([\![\boldsymbol{w}(t)]\!], ELW_{P_1}, ELW_{P_2}, \cdots, ELW_{P_N}) \tag{11}$$

where $g = 1, 2, \cdots, G$; the number of blocks is the same as the number of global aggregations. As the replicas in the consensus, P_i performs verification, including the signature of each transaction, MAC, and the correctness of Eq. (10) according to the public key.

2 Joint optimization for computation resources and adaptive aggregation

The distributed learning, especially the computation and learning in encryption, will consume lots of computing and energy resources. For the computing party C to obtain the value of its data analysis, it is necessary to keep operating costs low. Therefore, how to effectively use the given resources to optimize the learning effect (minimize the loss function) has become an important issue. To solve this problem, the computation cost and energy consumption at each node is analyzed, based on which the convergence performance of the model is improved by optimizing the computation resources allocation and the adaptive aggregation (dynamically adjust aggregation frequency τ).

2.1 Resources consumption and analysis

For resources consumption, from the aspect of learning (including local learning and global learning) and the blockchain consensus respectively to analyze it, it is assumed that the computing resources used by node $i (i \in \boldsymbol{I})$ for learning and consensus are f_i^L (CPU cycle frequency) and f_i^S, respectively.

(1) In learning (subsection 1.3), it is assumed that the average of the CPU cycles consumed by the participants and the computing party to complete a step of ciphertext calculation is μ_1 CPU cycles, and the average of the CPU cycles consumed to complete a step of plaintext calculation is μ_2 CPU cycles.

Local update. $P_{i'} (i' \in \boldsymbol{I}, i' = 1, \cdots, N)$ updates the local ciphertext model as shown in Eq. (9), and broadcasts it to the blockchain in the form of $ELW_{P_{i'}}$ transaction. In this step, the computation complexity is $O(|\boldsymbol{w}||\boldsymbol{D}_{i'}|)$. Accordingly, the computation costs $\Delta_{1i'}^L$ and computation time $\varepsilon_{1i'}^L$ are

$$\begin{aligned} \Delta_{1i'}^L &= \mu_1 (|\boldsymbol{w}||\boldsymbol{D}_{i'}|) \\ \varepsilon_{1i'}^L &= \frac{\Delta_{1i'}^L}{f_{1i'}^L} \end{aligned} \tag{12}$$

Global aggregation. The computing party $C (C = i'' \in \boldsymbol{I}, i'' = N + 1)$ collects $ELW_{P_{i'}}$ from $P_{i'}$, and updates the global ciphertext model as shown in Eq. (10). In this step, the computation complexity is $O(N|\boldsymbol{w}|)$. Since Paillier cannot handle the problem of ciphertext multiplication, the computing party needs to decrypt the $ELW_{P_{i'}}$ transaction, and computes the optimization parameter φ, δ (see subsection 2.2 for details). Its computation complexity is $O(\sum_{i'=1}^{N} N|\boldsymbol{w}||\boldsymbol{D}_{i'}|)$. The computation costs $\Delta_{2i''}^L$ and computation time $\varepsilon_{2i''}^L$ are

$$\begin{aligned} \Delta_{2i''}^L &= \mu_1 N|\boldsymbol{w}| + \mu_2 \sum_{i'=1}^{N} N|\boldsymbol{w}||\boldsymbol{D}_{i'}| \\ \varepsilon_{2i''}^L &= \frac{\Delta_{2i''}^L}{f_{2i''}} \end{aligned} \tag{13}$$

(2) In the PBFT consensus run between participants and the computing party, it can resist at most $F = 3^{-1}(N - 1)$ failed nodes. It is assumed that the average value of the CPU cycles consumed by computing tasks is α, and generating or verifying a signature and generating or verifying a MAC require β and θ CPU cycles, respectively. The consensus protocol consists of five steps.

Step 1 Request. Participants submit K transactions containing model parameters to computing party C (primary node i''). The primary node i'' packs the verified transactions into a new block. Afterwards, it broadcasts the produced block and pre-prepare messages to other nodes for verification. The computation costs $\Delta_{1i''}^S$ and computation time $\varepsilon_{1i''}^S$ are

$$\begin{aligned} \Delta_{1i''}^S &= K(\beta + \theta) + \beta + N\theta \\ T_{1i''}^S &= \frac{\Delta_{1i''}^S}{f_{1i'}^S} \end{aligned} \tag{14}$$

Step 2 Pre-prepare. $P_{i'} (i' \neq i'')$ as replica receives a new block of pre-prepare message, then verifies the signature and MAC of the block and the signature and MAC of each transaction in turn. Finally, the computation result is verified according to the smart contract. If the result is verified successfully, $P_{i'}$ will send the prepare messages to all the others. The computation costs $\Delta_{2i'}^S$ and computation time $\varepsilon_{2i'}^S$ are

$$\begin{aligned} \Delta_{2i'}^S &= \beta + \theta + K(\beta + \theta) + \alpha + \beta + N\theta \\ \varepsilon_{2i'}^S &= \frac{\Delta_{2i'}^S}{f_{2i'}^S} \end{aligned} \tag{15}$$

Step 3 Prepare. All the nodes receive and check the prepare message to ensure that it is consistent with the pre-prepare message. Once the node receives $2F$ prepare messages from others, it will send the commit messages to all the others. The computation costs Δ_{3i}^{S} and computation time ε_{3i}^{S} are

$$\begin{aligned} \Delta_{3i}^{S} &= 2F(\beta+\theta)+\beta+N\theta \\ T_{3i}^{S} &= \frac{\Delta_{3i}^{S}}{f_{3i}^{S}} \end{aligned} \tag{16}$$

Step 4 Commit. All the nodes receive and check the commit message to ensure that it is consistent with the prepare message. Once the node receives $2F$ commit messages from others, it will send the reply message to the primary node. The computation costs Δ_{4i}^{S} and computation time ε_{4i}^{S} are

$$\begin{aligned} \Delta_{4i}^{S} &= 2F(\beta+\theta)+K(\beta+\theta) \\ \varepsilon_{4i}^{S} &= \frac{\Delta_{4i}^{S}}{f_{4i}^{S}} \end{aligned} \tag{17}$$

Step 5 Reply. The primary node receives and checks the reply messages. After receiving $2F$ reply messages, the new block will take effect and be added to the blockchain. The computation cost $\Delta_{5i''}^{S}$ and computation time $\varepsilon_{5i''}^{S}$ are

$$\begin{aligned} \Delta_{5i''}^{S} &= 2F(\beta+\theta) \\ \varepsilon_{5i''}^{S} &= \frac{\Delta_{5i''}^{S}}{f_{i''}^{S}} \end{aligned} \tag{18}$$

As analyzed above, the efficiency (computation time) of a node in each step is constrained by computation cost and computation resources. Following the model in Ref. [14], the energy consumption of local update can be given as

$$F^{J}(f) = \sum_{i=1}^{N+1} \delta_{1i}^{L} \Delta_{1i}^{L} \gamma (f_{1i}^{L})^{2} \tag{19}$$

the energy consumption of global aggregation can be given as

$$F^{Q}(f) = \sum_{i=1}^{N+1} \delta_{2i}^{L} \Delta_{2i}^{L} \gamma (f_{2i}^{L})^{2} + \sum_{i=1}^{N+1} \sum_{\sigma=1}^{5} \delta_{\sigma i}^{S} \Delta_{\sigma i}^{S} \gamma (f_{\sigma i}^{S})^{2} \tag{20}$$

where γ is a constant related to the hardware, $\delta_{i} = [\delta_{1i}^{L}, \delta_{2i}^{L}, \delta_{\sigma i}^{S}]$, $\sigma = 1,2,3,4,5$; and the element $\delta_{\cdot,i} = 0,1$ indicates whether the node i participates in each step. For example, $\delta_{i''} = [0,1,1,0,1,1,1]$ and $\delta_{i' \neq i''} = [1,0,0,1,1,1,0]$ represent the participation of the computing party as a primary node and the participates as replicas at each step in the learning and consensus process, respectively.

Therefore, the energy consumption of the system is expressed as

$$F_{e}(\boldsymbol{f}, T, \tau) = T\left(F^{J}(\boldsymbol{f}) + \frac{F^{Q}(\boldsymbol{f})}{\tau}\right) \tag{21}$$

where $\boldsymbol{f} = \{f_{i}^{L}, f_{i}^{S}\}$ is the overall computation resources.

2.2 Optimization model and solutions

For the loss function $F(\boldsymbol{w})$, linear regression satisfies the following assumptions[16]. $F_{i}(\boldsymbol{w}')$ and $F(\boldsymbol{w})$ are convex, φ-smooth. For any $\boldsymbol{w}$, $\boldsymbol{w}'$. $\| \nabla F_{i}(\boldsymbol{w}) - \nabla F_{i}(\boldsymbol{w}') \| \leqslant \varphi \| \boldsymbol{w} - \boldsymbol{w}' \|$ to capture the divergence between the gradient of a local loss function and the gradient of the global loss function. This divergence has to do with how data is distributed across different nodes, and the measure factors φ, δ will affect the optimization model. For any t, $\hat{\varphi} = \sum_{i} |\boldsymbol{D}_{i}| \hat{\varphi}_{i} / |\boldsymbol{D}|$ is defined to approximate φ, where

$$\hat{\varphi}_{i} = \frac{\| \nabla F_{i}(\boldsymbol{w}_{i}(t)) - \nabla F_{i}(\boldsymbol{w}(t)) \|}{\| \boldsymbol{w}_{i}(t) - \boldsymbol{w}(t) \|} \tag{22}$$

Similarly, $\hat{\omega} = \sum_{i} |\boldsymbol{D}_{i}| \hat{\omega}_{i} / |\boldsymbol{D}|$ is approximated, where

$$\hat{\omega}_{i} = \| \nabla F_{i}(\boldsymbol{w}(t)) - \nabla F(\boldsymbol{w}(t)) \| \tag{23}$$

In view of the learning effect, the loss function $F(\boldsymbol{w}^{*})$ is introduced ($\boldsymbol{w}^{*}$ represents the ideal model). For any τ, $h(\tau) = \frac{\omega}{\varphi}((\eta\varphi+1)^{\tau}-1) - \eta\omega\tau$. If $\eta\varphi \leqslant 1$ is set, then $\eta(1-\frac{\varphi\eta}{2}) - \frac{B_{0}h(\tau)}{\tau} > 0$ is satisfied, where B_{0} is control parameter and is constant for the same dataset. In the above theorem, the upper bound on the convergence after T iterations can be obtained as Eq. (24).

$$F(\boldsymbol{w}(T,\tau)) - F(\boldsymbol{w}^{*}) \leqslant \frac{1}{T\left(\eta\left(1-\frac{\varphi\eta}{2}\right) - \frac{B_{0}h(\tau)}{\tau}\right)} \tag{24}$$

Therefore, the optimization model is defined as

$$\min_{T,\tau} \frac{1}{T\left(\eta\left(1-\frac{\varphi\eta}{2}\right) - \frac{B_{0}h(\tau)}{\tau}\right)} \tag{25}$$

$$\begin{aligned} \text{s.t.}\quad & C1: F_{e}(\boldsymbol{f}, T, \tau) \leqslant E \\ & C2: f_{i}^{L} + f_{i}^{S} \leqslant f_{i} \\ & C3: \frac{T}{\tau}\left(\max \frac{\tau\Delta_{1i'}^{L}}{f_{1i'}^{L}} + \frac{\Delta_{2N+1}^{L}}{f_{2N+1}^{L}}\right) \leqslant \xi,\ i' = 1,\cdots,N \\ & C4: \max \frac{T}{\tau} \sum_{\sigma=1}^{5} \frac{\Delta_{\sigma i}^{S}}{f_{\sigma i}^{S}} \leqslant \xi \\ & C5: \tau \leqslant \tau_{\max} \end{aligned}$$

where the loss function is minimized under the constraint of $C1 - C5$. The total system energy E is limited by $C1$, and the computation resources of the node is limited by $C2$. Moreover, the total time in the learning process and the consensus process are limited by $C3$ and $C4$, respectively, where ξ makes the time of the

two processes consistent. Since τ is unbounded and the problem is difficult to solve, the search space is restricted by $\tau_{\max}$ in $C5$ and on integers.

Since the denominator of Eq. (25) is always positive, then combining Eq. (21) and $C1$, the parameter T can be replaced by the optimal value $T = (E\tau)(\tau F^J(f) + F^Q(f))^{-1}$. Note that φ and δ in Eq. (25) are ideal values, and only the alternative approximated $\hat{\varphi}$ and $\hat{\delta}$ can be updated along the learning. In each global aggregation, the computation resources allocated for each node to complete each step and the aggregation frequency τ are optimized by the convex optimization theory. The joint optimization is repeated to decide the global aggregation until the energy of the system is consumed out, as shown in algorithm 1.

Algorithm 1 Process of learning and optimization

Input: E, η, B_0, ξ
Output: $[\![\boldsymbol{w}(T)]\!]$
1. Initial $t = 0$, $\tau = 1$, $R = 0$ // R is the energy count
2. Initial $E' = E$ and use Pk to obtain $[\![\boldsymbol{w}(0) = 0]\!]$
3. Initial $\boldsymbol{f}$, evenly distributed to each process
4. while ($t \geqslant 0 \& E > 0$) do
5. for $i = 1, 2, 3, \cdots, N$ do
6. P_i downloads global parameter from blockchain
7. P_i completes τ local updates
8. P_i broadcasts ELW_{P_i}
9. end for
10. $t = t + \tau$
11. C receives $[\![\boldsymbol{w}_i(t)]\!]$ from P_i
12. C aggregates $[\![\boldsymbol{w}(t)]\!]$ and calculates δ, $\hat{\varphi}$
13. C calculates $R = R + \tau F^J(f) + F^Q(f)$
14. C updates $E = E' - R$
15. C uses Eq. (25) to obtain the optimal $\boldsymbol{f}$ and τ, where $\tau \in [1, \tau_{\max}]$
16. if $R > E'$ then
17. reduce to τ the maximum value within E'
18. set $E = 0$
19. end if
20. C creates a block and upload to blockchain
21. end while

3 Simulation and analysis

3.1 Security and privacy analysis

The system addresses data privacy and model security by combining distributed machine learning with homomorphic encryption and blockchain.

For privacy, the Palillier encryption algorithm is used to protect the model parameters, where the encrypted parameters prevent attackers from obtaining local data by eavesdropping on the broadcast and also protect every model parameter from other participants. Only the computing party as the sponsor can know the parameters.

For security, blockchain is used to make point-to-point interaction between computing party and participants to ensure the reliability of the system. Through the PBFT consensus, the participants are able to verify transactions and the updated ciphertext model from the beginning to the end results, which confirms the contribution of each participant.

3.2 Simulation

In this section, the proposed joint optimization scheme for computation resources and adaptive aggregation is simulated. To verify its performance, the proposed joint optimization is compared with the other two optimization cases using the loss function value as a metric.

Only-aggregation. Ref. [16] proposed an adaptive aggregation algorithm. Assuming that the computation resources are fixed (the computation resources allocated to each node in different steps are equal and fixed), only the aggregation frequency is adaptively optimized so that the loss function reaches minimum.

Only-resource. A comparison scheme for joint optimization, where the aggregation frequency is fixed, and only the allocation of computation resources is optimized.

In order to prove the generality of the joint optimization, it is validated on the Boston house price dataset with 14 features and the fish toxicity dataset with 7 features, respectively. 506 pieces of sample data are selected on each dataset and all sample data are preprocessed to the interval $[-1, 1]$. 80% of them are selected for training and others for testing. For simulations, the parameter settings are shown in Table 1.

Fig. 2 and Fig. 3 show the relationship between the loss value and the aggregation frequency τ. In order to find the optimal fixed τ in the only-resources optimization through simulations, it is considered that the value of τ is from 10 to 90. It can be seen that the optimal value of τ varies with different number of participants and dataset, and a fixed value of τ will not be suitable for all cases. Note that this optimal value can not be obtained in practical work. In order to facilitate comparison, according to the changing trend of the curve, $\tau = 25$ is set in the Boston dataset and $\tau = 30$ is set in the fish dataset.

Table 1 The simulation parameters

Simulation parameters	House dataset	Fish dataset
The learning rate η	0.001	
The total time ξ	300 s	
The maximum computation resource f_i	1500 M cycles	
CPU cycles for completing a step ciphertext calculation μ_1	10 M cycles	
CPU cycles for completing a step plaintext calculation μ_2	0.5 M cycles	
CPU cycles for computing tasks α	0.2 M cycles	
CPU cycles for generating or verifying a signature β	8 M cycles	
CPU cycles for generating or verifying a MAC θ	0.05 M cycles	
The hardware parameter γ	10^{-28}	
The maximum value of aggregation frequency $\tau_{\max}$	10τ	
The control parameter B_0	1.5	2

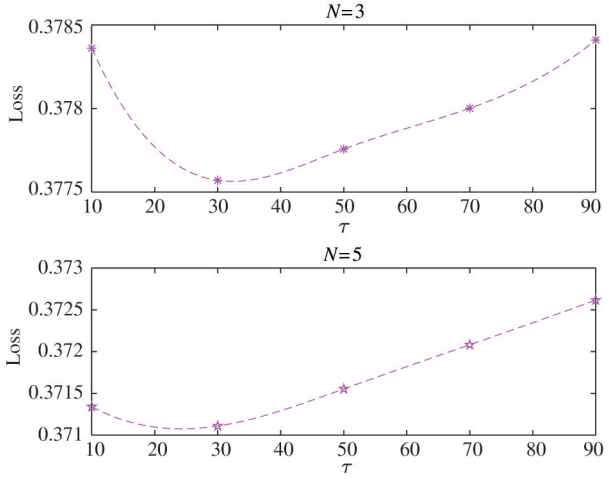

Fig. 2 Loss value with different τ in Boston dataset

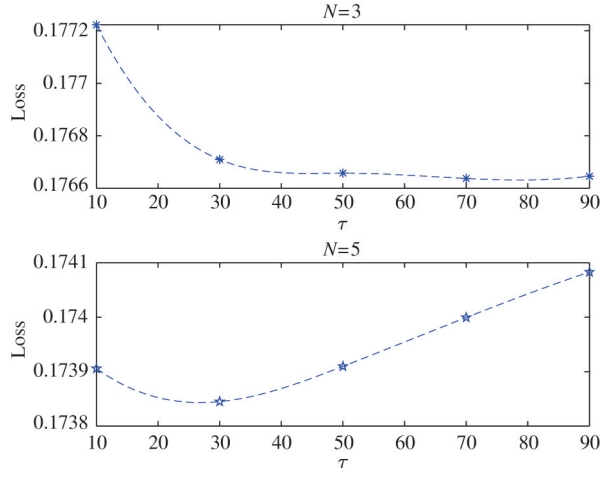

Fig. 3 Loss value with different τ in fish dataset

Fig. 4 shows the relationship between system energy and loss value when the number of participants is $N = 5$. Since the Boston dataset has more feature dimensions than the fish dataset, it consumes more energy during learning and requires more energy to converge (i.e. different orders of magnitude on the horizontal axis). It can be seen that with the increase of the system energy, the loss values of the three cases decrease, and the joint optimization scheme has the smallest loss value. It adjusts the computation resources allocation of each node in each step according to the complexity, so that the energy consumption of each iteration reaches an ideal value and the distributed learning process can complete more iterations under the constraint of limited system energy. When the system energy is small, the effect of the joint optimization scheme is more obvious. In addition, there is a small difference between the loss value of joint optimization and only-resources optimization. It can be seen that in the distributed machine learning, optimizing computation resources allocation plays a more important role than optimizing aggregation frequency.

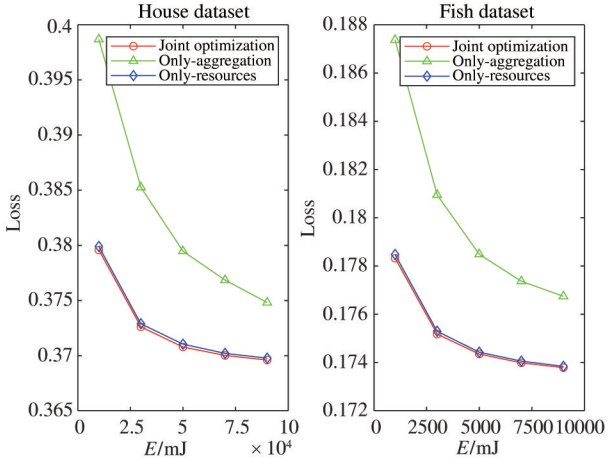

Fig. 4 Loss value with different energy consumption

Fig. 5 shows the relationship between the number of participants and the loss value. It can be seen that as the number of participants increases, the change trend of the loss value under different datasets is different. The reason is that the loss value is not only affected by the number of participants, but also related to the data distribution. With the same number of participants, the proposed joint optimization scheme has the smallest loss value.

4 Conclusions

A distributed learning scheme is proposed using homomorphic encryption and blockchain as the privacy

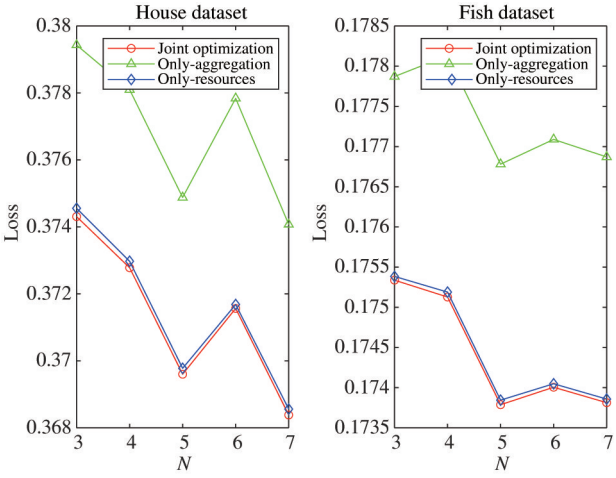

Fig. 5 Loss value with different number of participants

and security guarantee. In the scheme, data is leaved to its owner and only the encrypted model parameters derived from data are transmitted to the global aggregation, all of which are recorded and verified by blockchain consensus. In this way, the privacy of both the data and model as well as the security of the learning are ensured. Most importantly, the computation resources and the adaptive aggregation in the distributed learning and consensus are optimized. Simulation results show the efficiency of the scheme.

References

[1] VERBRAEKEN J, WOLTING M, KATZY J, et al. A survey on distributed machine learning[J]. *ACM Computing Surveys* (*CSUR*), 2020, 53(2): 1-33

[2] DE C E. An overview of privacy in machine learning[EB/OL]. https://arxiv.org/abs/2005.08679: arXiv, (2020-05-18),[2022-03-10]

[3] HUANG Z, HU R, GUO Y, et al. DP-ADMM: ADMM-based distributed learning with differential privacy[J]. *IEEE Transactions on Information Forensics and Security*, 2019, 15: 1002-1012

[4] JI Z, LIPTON Z C, ELKAN C. Differential privacy and machine learning: a survey and review[EB/OL]. https://arxiv.org/abs/1412.7584: arXiv, (2014-12-24), [2022-03-10]

[5] SUN X, ZHANG P, LIU J K, et al. Private machine learning classification based on fully homomorphic encryption[J]. *IEEE Transactions on Emerging Topics in Computing*, 2018, 8(2): 352-364

[6] AONO Y, HAYASHI T, WANG L, et al. Privacy-preserving deep learning via additively homomorphic encryption [J]. *IEEE Transactions on Information Forensics and Security*, 2017, 13(5): 1333-1345

[7] FANG H, QIAN Q. Privacy preserving machine learning with homomorphic encryption and federated learning[J]. *Future Internet*, 2021, 13(4): 94

[8] ZHANG C, LI S, XIA J, et al. {BatchCrypt}: efficient homomorphic encryption for {Cross-Silo} federated learning[C]//2020 USENIX Annual Technical Conference (USENIX ATC 20), Carlsbad, USA, 2020: 493-506

[9] LYU L, YU H, YANG Q. Threats to federated learning: a survey[EB/OL]. https://arxiv.org/abs/2003.02133: arXiv, (2020-03-04),[2022-03-10]

[10] KUO T T, OHNO-MACHADO L. Modelchain: decentralized privacy-preserving healthcare predictive modeling framework on private blockchain networks[EB/OL]. https://arxiv.org/abs/1802.01746: arXiv, (2018-02-06),[2022-03-10]

[11] MENDIS G J, WU Y, WEI J, et al. A blockchain-powered decentralized and secure computing paradigm[J]. *IEEE Transactions on Emerging Topics in Computing*, 2021, 9(4): 2201-2222

[12] ZHOU S, HUANG H, CHEN W, et al. Pirate: a blockchain-based secure framework of distributed machine learning in 5g networks[J]. *IEEE Network*, 2020, 34 (6): 84-91

[13] SHEN M, TANG X, ZHU L, et al. Privacy-preserving support vector machine training over blockchain-based encrypted IoT data in smart cities[J]. *IEEE Internet of Things Journal*, 2019, 6(5): 7702-7712

[14] CHEN X, JI J, LUO C, et al. When machine learning meets blockchain: a decentralized, privacy-preserving and secure design[C]//2018 IEEE International Conference on Big Data, Seattle, USA, 2018: 1178-1187

[15] PAILLIER P. Public-key cryptosystems based on composite degree residuosity classes[C]//International Conference on the Theory and Applications of Cryptographic Techniques, Berlin, Germany, 1999: 223-238

[16] WANG S, TUOR T, SALONIDIS T, et al. When edge meets learning: adaptive control for resource-constrained distributed machine learning [C]//IEEE INFOCOM 2018-IEEE Conference on Computer Communications, Honolulu, USA, 2018: 63-71

YANG Ruizhe, born in 1982. She received the Ph. D. degree from the Beijing University of Posts and Telecommunications and joined the Beijing University of Technology in 2009. Her research interests include blockchain, resource management, and channel estimation.