TraCemop: Towards Federated Learning with Traceable Contribution Evaluation and Model Ownership Protection

Jingwei Liu, *Member, IEEE*, Zihan Zhou, *Student Member, IEEE*, Rong Sun, *Member, IEEE*, Lei Liu, *Member, IEEE*, Rongxing Lu, *Fellow, IEEE*, Schahram Dustdar, *Fellow, IEEE*, Dusit Niyato, *Fellow, IEEE*

***Abstract*—Federated Learning (FL) allows multiple clients to collaboratively train machine learning models without the need to share their local private data. As a result, it can effectively address the issue of data fragmentation. Nevertheless, insufficient evaluation of individual contributions and the lack of protections for both the intellectual property rights (IPR) of models and client privacy can greatly reduce clients' motivations in federated training. To address these challenges, this paper introduces the Traceable Contribution Evaluation and Model Ownership Protection (TraCemop) framework for federated learning, which allows each client to swiftly assess the contributions of others in each round, with integrated support for the traceability of evaluation results. To safeguard the intellectual property of models, a collective watermark is embedded in the global model. Additionally, a secure mechanism for verifying model ownership is also available in case of disputes. Security analysis indicates that TraCemop is capable of resisting data reconstruction attacks as well as various types of model copyright infringements. Finally, we evaluate the proposed framework using two commonly-used datasets, and the experimental results show a significant improvement in the efficiency of contribution evaluation compared to existing methods. Meanwhile, IPR infringement tests on TraCemop reveal that the proposed framework is resilient against malicious efforts to monopolize model ownership.**

***Index Terms*—Federated learning, Watermark, Contribution evaluation, Model IPR, Blockchain.**

I. Introduction

This work was supported in part by the Joint Fund of the National Natural Science Foundation of China (No. U24A20241), in part by the Key Research and Development Program of Shaanxi Province (No. 2023-ZDLGY-34), in part by the Key Laboratory of Computing Power Network and Information Security, Ministry of Education under Grant 2023ZD008, and in part by the Innovation Capability Support Program of Shaanxi under Grant 2024RS-CXTD-01. (*Corresponding author: Rong Sun and Lei Liu.*)

Jingwei Liu and Zihan Zhou are with the Engineering Research Center of Blockchain Technology Application and Evaluation, Ministry of Education, and also with the Shaanxi Key Laboratory of Blockchain and Secure Computing, Xidian University, Xi'an, 710071, China (e-mail: jwliu@mail.xidian.edu.cn; zihanchou@stu.xidian.edu.cn).

Rong Sun is with the State Key Laboratory of Integrated Services Networks, Xidian University, Xi'an 710071, China (e-mail: rsun@mail.xidian.edu.cn).

Lei Liu is with the Guangzhou Institute of Technology, Xidian University, Guangzhou 510555,China, and also with the Key Laboratory of Computing Power Network and Information Security, Ministry of Education, Qilu University of Technology (Shandong Academy of Sciences), Jinan 250014, China (e-mail: leiliu@xidian.edu.cn).

Rongxing Lu is with the School of Computing, Queen's University, Kingston, ON K7L 2N8, Canada (e-mail: rongxing.lu@queensu.ca).

Schahram Dustdar is with the Distributed Systems Group, TU Wien, Austria, and also with a part-time ICREA Professor at UPF, Barcelona. (Email: dustdar@dsg.tuwien.ac.at).

Dusit Niyato is with the School of Computer Science and Engineering, Nanyang Technological University, Singapore (e-mail: dniyato@ntu.edu.sg).

FEDERATED learning, as a paradigm within distributed machine learning, has garnered considerable interest owing to its capacity to guarantee data privacy. By completing the model's training process locally and only sharing model updates, FL prevents the privacy risks associated with centralized data storage, offering a high quality solution for the joint training of high precision models among multiple institutions. Up till now, Federated learning has been widely used in various fields such as healthcare services and smart city [1], [2]. In contrast to ordinary Machine Learning (ML), FL boasts the distinct advantage of gathering a vast array of users, who consistently contribute valuable training data and computing resources toward global model updates. These collective efforts are pivotal in achieving FL models with high performance. Hence, sustaining the engagement of clients (participants) in FL training has great significance. Regrettably, two escalating challenges—1) inequitable evaluation of participant contribution and 2) model IPR infringement—pose substantial obstacles to enhancing participants' motivations in FL.

The first challenge arises from profit distribution. A well-performed global model relies on the participation of high quality clients. These clients expect rewards that match their contributions when the trained global model generates revenue. Therefore, it is necessary to assess each participant's contribution to reflect the value of their local models to the global model. Currently, participant contribution in FL is mostly assessed by the Shapley value [3], [4]. However, the high computational complexity of the Shapley value impedes assessment efficiency, which can diminish the contributors' willingness and thereby weaken their commitment to FL training. Consequently, it is imperative to establish an efficient and objective contribution assessment mechanism for fairness in the FL training process.

Moreover, another important challenge is posed by the unauthorized copying and misuse of the well-trained FL model, which is derived from participants' available resources such as substantial computational power and large-scale datasets. To address this issue, the watermarking technique provides an effective solution for IPR protection in FL. In particular, the white-box watermarking technique implicitly embeds the private information of the model owner into the model, which enables the model owners to access the suspicious models in the white-box manner to claim model copyright [5], [6]. Due to direct access to model parameters, white-box watermark-

ing provides a more direct assertion for IPR in the model competition, neural network model markets, and business environments pursuing transparency and trustworthiness. Additionally, adopting white-box watermarking helps enterprises avoid unnecessary legal disputes. Nevertheless, during the verification process, the watermark embedding matrix of the model owner can be exposed. This vulnerability allows a malicious adversary to launch an ownership query against the rightful owner to obtain the watermark embedding matrix. The malicious adversary can then falsely claim ownership of the model and deploy the model as a service to extort money. Most existing schemes presume the trustworthiness of the model ownership verifier. However, in practical applications, the verifier is often not completely trusted. Consequently, ensuring the security of ownership verification remains an open challenge in FL. Besides, the model owners are concerned not only with the ownership of the suspect model but also with the benefit distribution if the model belongs to them. This requires support from the contributions of clients during the training phase. Consequently, obtaining the contribution of clients according to the results of model IPR claims is widely open.

In response to the aforementioned challenges, this paper proposes a solution, named TraCemop, to address fairness and IPR violation issues in Horizontal Federated Learning (HFL). TraCemop measures the quality of local models using artificial samples generated by Differentially Private Generative Adversarial Network (DPGAN) [7] for rapid and accurate contribution assessment. In addition, it employs white-box watermarking to protect IPR and incorporates a secure ownership verification mechanism based on two-party secure computation to maintain the confidentiality of the watermark embedding matrix. To simultaneously achieve model ownership verification and contribution queries, our approach supports the direct retrieval of participants' contribution information from the watermark. Upon confirming the owner of the model, the watermark can be utilized to retrieve the contributions of each FL participant in the training phase, providing proof for the subsequent benefits allocation. To avoid performance degradation due to the limited redundancy in model parameters when directly embedding contribution information, TraCemop leverages blockchain to record contributions from each training round. The watermark is used for the storage and retrieval of contribution values, establishing a unique binding between the model and the corresponding contribution information. Experimental results validate the effectiveness and rationale of TraCemop. Specifically, the key advantages of TraCemop are as follows:

- Efficient and well-founded contribution evaluation: TraCemop reduces the assessment time for FL participants' contribution while providing an approximate evaluation distribution comparable to existing schemes.
- Secure model copyright protection: TraCemop resists watermark obfuscation and removal attacks, supporting model ownership verification without revealing the watermark embedding matrix.
- Preservation of model performance: TraCemop minimizes the impact of watermark embedding on model accuracy through a new regularization term.
- Enhanced information capacity: TraCemop stores information on the blockchain rather than embedding it directly into the watermark, thus effectively and indirectly increasing the amount of information attributed to the watermark.

The remainder of this paper is organized as follows: Section II reviews related work relevant to the challenges addressed by our framework. Section III introduces the basic techniques utilized in this work. Section IV presents the system model, threat model and design goals. In Section V, we detail the TraCemop framework and analyze its security properties in Section VI. Extensive experiments are conducted to evaluate the performance of the proposed framework in Section VII. Finally, Section VIII concludes the paper and outlines future research directions. Besides, we use "client" and "participant" interchangeably in this paper.

II. Related Work

In this section, we review some existing works in Federated Learning (FL).

A. Privacy-Preserving Federated Learning

Federated learning enables participants to collaboratively train a global model without divulging their raw data. However, during training, participants still share model parameters directly with the aggregation server, making them vulnerable to model inversion attacks, which can lead to data leakage. To tackle these privacy concerns, researchers have developed privacy-preserving techniques to prevent information leakage during FL model training. In 2017, Bonawitz *et al.* [8] introduced a secure aggregation protocol using Secure Multi-Party Computation (SMPC) to protect participant gradients. Following this, Phong *et al.* [9] proposed an FL scheme based on Homomorphic Encryption (HE) to prevent servers from inferring client data from model gradients. In 2020, Lu *et al.* [10] designed an FL scheme using Local Differential Privacy (LDP) within vehicular networks, where participants obfuscate their local models with Laplace noise during aggregation. However, the noise introduced from LDP can degrade the accuracy of the global model. In 2022, Ma *et al.* [11] proposed a Byzantine-tolerated aggregation scheme that measures cosine similarity between participants and employs a two-trapdoor Paillier cryptosystem. However, this work converts the float-point gradients to integers to fit the Paillier algorithm, leading to precision loss in the global model.

B. Contribution Evaluation in Federated Learning

To achieve high-performance FL models, an incentive mechanism is necessary to motivate participants to contribute resources. The contribution of each participant is crucial for establishing a fair incentive mechanism, making contribution evaluation a key issue in FL. In 2018, Campen *et al.* [12] first applied the Shapley value to estimate the participant contributions in HFL, establishing it as a mainstream solution in contribution evaluation [13]. In 2019, Wang *et al.*

[3] proposed an incentive mechanism that robustly measures participant contributions in Vertical Federated Learning (VFL), allocating profits based on the importance of data features. In 2023, Jia *et al.* [4] addressed the challenge of improving computational efficiency while maintaining accurate Shapley value estimation. However, the high computational complexity of Shapley values increases the training cost, limiting its practical application.

C. Watermarking Methods in Machine Learning

Developing high-performance ML models often requires substantial computing resources, whereas stealing these models remains relatively straightforward. To protect the model IPR, digital watermarking has been applied to ML models and can be broadly categorized into white-box and black-box watermarking methods.

White-box Watermarking. In 2017, Uchida *et al.* [5] proposed the first white-box watermarking scheme using a projection matrix. Building upon this, researchers have explored various white-box techniques to establish model ownership claims [14]. Since white-box watermarking requires full access to model parameters for embedding and verification, it is particularly suited for scenarios pursuing transparency and credibility, such as model trading markets and competitions [15].

Black-box Watermarking. Inspired by backdoor attacks, researchers introduced black-box watermarking [16], [17]. This method achieves copyright protection by encoding an exceptional input-output relationship on a trigger set. Model owners can verify ownership by querying the model with the trigger set and checking the corresponding outputs. Since black-box watermarking relies on input-output patterns rather than direct access to model parameters, it has been widely applied to various ML models, including classification and self-supervised learning [18].

D. Watermarking Methods in Federated Learning

With the growing demand for distributed training, federated learning (FL) has made significant progress. However, this advancement has also led to increasing concerns regarding IPR infringement [19]. Consequently, there is an urgent need to protect the copyright of FL models. Broadly, existing FL watermarking methods can be categorized into independent and joint watermarking approaches.

Independent Watermarking. In this approach, each client in the FL network embeds a unique watermark to independently assert model ownership [20]. Li *et al.* [6] first introduced the independent watermarking scheme FedIPR that combined both white-box and black-box approaches in FL. In FedIPR, each client can claim ownership using their respective watermarks. However, due to variations in the distribution of embedded watermarks across different clients, some embedded watermarks may become invalid. Besides, malicious clients can collude to introduce backdoors into the model. To address these challenges, Wu *et al.* [21] proposed a multi-party entangled watermark algorithm to enhance watermark embedding effectiveness. Furthermore, Luo *et al.* [22] proposed FedFP to defend against ownership collusion attacks in FL. This method constructs client-embedded watermarks using an anti-collusion code, providing a mechanism to detect collusion. In 2024, Zhang *et al.* [23] proposed FedMark to improve the capacity of white-box watermarking in FL. By incorporating a Bloom filter, FedMark minimizes the impact of watermark embedding on model parameters, thereby enhancing efficiency and significantly increasing watermark capacity.

Joint Watermarking. The joint watermarking method embeds a unified IPR proof into the global model to reduce the impact of multiple copyrights on model performance. In 2023, Wu *et al.* [24] designed a two-stage watermark verification algorithm, that employs black-box watermarking to resist free-rider attacks, while using a white-box watermark as the final proof of model ownership. Yan *et al.* [25] proposed a joint watermark framework for cross-soil FL, utilizing a watermark generation adversarial network (WMGAN) to integrate private client watermarks into a single joint watermark, thereby facilitating seamless integration and verification. Similarly, Yang *et al.* [26] applied black-box watermarking for secure watermark embedding in FL. More recently, in 2025, Shao *et al.* [27] introduced FedTracker, an FL model copyright protection framework designed to enable ownership verification and traceability through a dual-layer protection mechanism. This scheme incorporated a continuous learning-based watermark embedding algorithm that preserved model utility while embedding watermarks. Additionally, it introduced a novel fingerprint similarity score to better distinguish different fingerprints.

However, to the best of our knowledge, none of the existing white-box-based schemes adequately address privacy leakage during ownership verification, which remains a critical open issue in FL copyright claims.

III. Preliminaries

A. Horizontal Federated Learning

Horizontal Federated Learning (HFL) allows participants to jointly train a global model with enhanced generalization ability while preserving the confidentiality of their private data. Similar to traditional machine learning, the goal of HFL is to determine an optimal set of model parameters $\boldsymbol{\omega}$ that maximize the model's prediction accuracy on its main task. To review HFL, we consider the classic algorithm FedAvg [28]. Assume there are K participants $U_i \in \{U_i\}_{i=1}^{K}$, where each participant U_i owns a local dataset D_i. The specific implementation of HFL is outlined below.

- *Local Training:* Each participant U_i trains their local model using the dataset D_i. FedAvg employs stochastic gradient descent (SGD) algorithm, an iterative optimization method, to minimize the loss function $E_o(\boldsymbol{\omega}_i^t, D_i)$. Specifically, each client calculates the gradient $\boldsymbol{g}_{\boldsymbol{\omega}_i^t}$ of $\boldsymbol{\omega}_i^t$ and updates the local model parameters as follows:

$$\boldsymbol{\omega}_i^{t+1} = \boldsymbol{\omega}_i^t - \alpha \boldsymbol{g}_{\boldsymbol{\omega}_i^t}$$

  where α is the learning rate. After $\mathbb{E}$-epoch training, participant U_i sends the updated model parameters $\boldsymbol{\omega}_i^{t+1}$ to the server for aggregation.

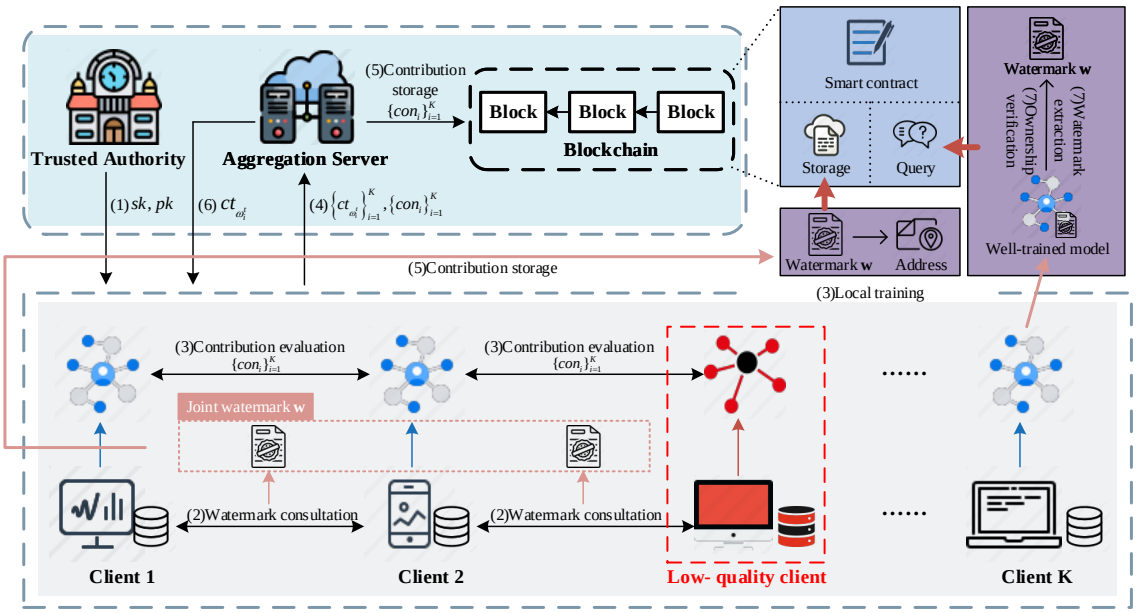


Fig. 1. The workflow of TraCemop, where a framework that integrates model IPR protection and contribution evaluation of models is proposed. The initialization phase includes steps (1)-(2): TA generates private/public key pairs for clients, and clients negotiate a watermark for protecting model IPR. The training phase encompasses steps (3)-(6): Clients train local models using their private data. After each round of local training, clients evaluate their contributions, which are stored on the blockchain during model aggregation. Clients download the model updates after model aggregation and execute the next local training round until the model converges. Step (7) involves copyright verification: The watermark is extracted from the trained model and compared with the original watermark to claim ownership. Furthermore, the watermark can be used to query contribution records from the blockchain.

- *Model Updates:* The server aggregates these local model parameters to obtain the global model parameters $\boldsymbol{\omega}^{t+1}$ as follows:

$$\boldsymbol{\omega}^{t+1} = \sum_{i=1}^{K} p_i \boldsymbol{\omega}_i^{t+1}$$

where p_i is the parameter weight of U_i. Finally, the server sends $\boldsymbol{\omega}^{t+1}$ to each participant for the next round of training until the global model converges.

B. Contribution in FL

Within an FL framework, assessing the contribution of each client accurately is essential for fostering participant enthusiasm. The contributions not only quantify the enhancement of clients to the model performance but also directly impact the fairness of reward allocation. As outlined in [29], an excellent contribution evaluation mechanism should fulfill the following requirements:

- **Fairness**: It should ensure fair measurement of each participant's contribution, and judge all the efforts of participants appropriately.
- **Efficiency**: The mechanism should be computationally efficient, particularly in large-scale FL scenarios.
- **Scalability**: It should dynamically adjust to clients joining and leaving the training process.

Additionally, it is critical to clarify that low-contribution clients are not malicious clients necessarily. Factors such as data quality, device performance, or network conditions can objectively lead to a participant with a lower contribution.

C. Homomorphic Encryption

Homomorphic Encryption (HE) allows any party to perform linear operations on encrypted data without decrypting it first. HE can be categorized into partially homomorphic encryption (PHE) and fully homomorphic encryption (FHE). PHE supports either additive or multiplicative homomorphism, while FHE supports both. The ability of FHE to perform addition and multiplication on encrypted data enables complex mathematical computations without decryption. In this paper, we exploit FHE to guarantee the data security during the communication [30].

In general, an FHE scheme typically consists of four components.

- FHE.KeyGen$(1^\lambda) \rightarrow (pk, sk)$: Given the security parameters λ, this probabilistic polynomial time (PPT) algorithm generates a public/private key pair $\{pk, sk\}$.
- FHE.Enc$_{pk}(m) \rightarrow ct$: This encryption algorithm takes the public key pk and a plaintext m as inputs, and outputs the ciphertext ct.
- FHE.Dec$_{sk}(ct) \rightarrow m$: Input the private key sk and the ciphertext ct, this decryption algorithm returns the plaintext m.
- FHE.Eval$(ct_1, ct_2, f_o) \rightarrow ct'$: Given two ciphertexts ct_1 and ct_2, along with a linear operation function f_o, this algorithm outputs the linear operation results $ct' = f_o(ct_1, ct_2)$ such that $FHE.Dec_{sk}(ct') = f_o(FHE.Dec_{sk}(ct_1), FHE.Dec_{sk}(ct_2))$.

By incorporating Single Instruction Multiple Data (SIMD) technology, FHE supports the encryption and homomorphic operations on vectors of plaintexts, significantly enhancing computational efficiency.

D. White-box-based Watermark

White-box watermarking has emerged as a promising approach to safeguarding the IPR of FL models [5]. This method allows the model owner to assert their ownership of the model by verifying whether a watermark, containing their private information, is embedded in the suspect model. According to the classical white-box watermarking scheme [5], it can be formalized into two main procedures: watermark embedding and watermark extraction.

In the watermark embedding procedure, the model owner embeds a secret T-bit vector $\mathbf{w} \in \{0,1\}^T$ as the watermark into the model using a predefined watermark embedding matrix $\mathbf{X}$. This is achieved by adding a regularization term E_r to the original loss function during the training phase. $\mathbf{X}$ contains the secret information required to extract the watermark and therefore should be kept confidential by the model owner.

In the event of an intellectual property dispute, the owner can initiate the watermark extraction procedure to verify model ownership. The owner extracts the embedded information $\mathbf{w}$ from the model parameters by the watermark embedding matrix $\mathbf{X}$, and compares it with the original watermark. Ownership is determined based on the similarity between the original watermark and the extracted watermark.

E. Secure Two-Party Computation (S2PC)

S2PC enables two participants to securely compute a function without revealing any additional information beyond their own inputs and outputs. In this work, we implement the secure model copyright verification using Additive Secret Sharing (ASS) secure two-party computation protocol [31]. Here, a semi-honest adversary with computational limits attempts to obtain additional information from the messages exchanged during the protocol's execution. The basic semantics of ASS-based S2PC are described as follows:

1) Additive Secret Sharing: Consider two participants U_0 and U_1, who possess two secret values x^0 and x^1, respectively, where both are l-bit integers in the ring $\mathbb{Z}_{2^l}$. x^0 is split into two random values x_0^0 and x_1^0 such that $x_0^0 + x_1^0 \equiv x^0 \ \left(mod\ 2^l\right)$. Similarly, x^1 is also randomly split. Each participant U_i shares x_{1-i}^i with the other participant for secure computation.

2) Addition over ASS: U_i can obtain the shares of $z = x^0 + x^1$ by calculating $z_i = x_i^0 + x_i^1$ locally.

3) Multiplication over ASS: To facilitate multiplication on additive secret shares, Beaver introduced the technique of Beaver triples. Using the Beaver triples, the multiplication result can be obtained in additive share form with only one round of interaction. The methodology for generating Beaver triples is extensively discussed in [31]. The efficient implementation of multiplication computations using the Beaver triples proceeds as follows.

U_0 and U_1 jointly generate a Beaver triple (a, b, c), where $a, b, c \in \mathbb{Z}_{2^l}$ and $c = a \cdot b$. The symbol "$\cdot$" denotes multiplication. U_i can compute its partial multiplication result $\left(x^0 \cdot x^1\right)_i$ using the Beaver triple and its additive secret shares, defined as follows:

$$\left(x^0 \cdot x^1\right)_i = i \cdot d \cdot e + d \cdot x_i^{1-i} + e \cdot x_i^i + c_i$$

where $d_i = x_i^i - a_i$, $e_i = x_i^{1-i} - a_i$, $d = d_i + d_{1-i}$, and $e = e_i + e_{1-i}$. The multiplication on x^0 and x^1 can be converted into the addition of the shares held by U_0 and U_1 through ASS. U_0 and U_1 can obtain the final multiplication result by summing $(x^0 \cdot x^1)_i$ together.

TABLE I
NOTATIONS AND DESCRIPTION

Notation	Description
AS_i	the artificial sample set generated by DPGAN of U_i
as_i	the artificial samples released to the other clients
β	a hyperparameter to control the impact of regularization term
$ct_{\boldsymbol{\omega}_i^t}$	the ciphertext of $\boldsymbol{\omega}_i^t$
$ct_{\boldsymbol{\omega}^{t+1}}$	the ciphertext of $\boldsymbol{\omega}^{t+1}$
c_i^j	the raw contribution value of U_j to U_i
con_i	the normalized contribution value of U_i
D_i	the private dataset of U_i
$E(\cdot)$	the total loss function
$E_o(\cdot)$	the raw loss function
$E_r(\cdot)$	the watermarking regularization term
$H(\cdot)$	the Hamming distance
K	the number of clients in FL
L	the length of $\boldsymbol{\theta}$
$M_i(\cdot)$	the local learning objective of U_i
N	the number of as_i
p_i	the aggregation weight of $\boldsymbol{\omega}_i$
sk, pk	the private/public key pairs of homomorphic encryption
T	the bit length of $\mathbf{w}$
$\{U_i\}_{i=1}^K$	the client set
$\mathbf{w}$	the watermark
$\hat{\mathbf{w}}$	the watermark extracted from model
$\mathbf{X}$	the watermark embedding matrix
λ	the security parameter
$\boldsymbol{\omega}$	the global model parameters
$\boldsymbol{\omega}_i^t$	the model parameters of U_i in round t
$\boldsymbol{\theta}$	the specific layer of $\boldsymbol{\omega}$ with watermark
$[\cdot]_i$	the i-th element of a vector
$[\cdot]_{i,j}$	the element in the i-th row and the j-th column of a matrix

IV. System Overview

In this section, we focus on the system model and the threat model of TraCemop. Additionally, we discuss the design goals of the proposed framework. The basic notations are shown in TABLE I. Note that in this paper, we use non-bold lowercase symbols to represent the scalar variables, and use bold lowercase and bold uppercase symbols to represent vectors and matrices unless otherwise specified.

A. System Model

The system model consists of five entities: Trusted Authority, Aggregation Server, Blockchain, Clients, and the model user.

- *Trusted Authority (TA).* The *TA* is responsible for generating homomorphic encryption key pairs for clients.
- *Aggregation Server (AS).* The *AS* aggregates the local models from the clients to form the global model.
- *Blockchain (BC).* The *BC* serves as a storage center for the contribution values from each training round. Clients can retrieve their contribution values from the *BC* by the watermark for profit allocation[1]. We design

[1]This paper primarily focuses on the evaluation of clients' contributions. The aspect of profit distribution is beyond the scope of this paper and will not be discussed.

and deploy a smart contract on the *BC* for secure and traceable contribution values storage and query, defined in Algorithm 2.

- *Clients.* Clients (participants) have more interest in models with superior generalization ability. They collaborate to jointly train a global model using their local data while seeking information from others without compromising their training data.
- *Model User.* The model users do not participate directly in the model training, and they acquire the trained model with a watermark $\mathbf{w}$ to meet their needs or generate business profit. These models may be obtained through legal purchase or illegal theft. During the model copyright proof phase, the model user acts as the "verifier" and leads the entire verification process.

B. Threat Model

For convenience, we define the threat model to analyze the potential security threats: *TA* and *BC* are assumed to be trustworthy and execute their duties without infringing upon users' privacy. The *AS* is considered to be honest-but-curious. It performs aggregation honestly but may attempt to obtain additional privacy information, such as local model parameters from clients. The model user is honest-but-curious, who collaborates in the model copyright verification but tries to seek information on the watermark embedding matrix. We consider two types of clients with different behaviors in the face of the privacy-preserving issue and the model IPR issue:

- **Privacy-Preserving Issue**: In this issue, we consider the clients to be honest-but-curious. During the training phase, data security is the main concern. In this situation, the clients are honest-but-curious, aiming to obtain training data or model parameters from other clients while adhering to the rules.
- **Model IPR Issue**: In this issue, we consider there exist malicious clients in our framework. All clients aim to obtain a high-performance model through FL, which is the result of collective effort. Therefore, all clients should share the IPR of the model. However, some malicious clients attempt to monopolize the model. They generate an illegitimate watermark, and attempt to gain exclusive control of the model by removing or obfuscating legitimate watermarks while embedding his/her illegitimate watermark.

Potential threats from these entities are described below:

- *Illegal model possession*. A model user might attempt to retain the watermark embedding matrix during extraction, intending to unlawfully claim ownership of the model.
- *Data reconstruction attack*. During the training phase, the *AS* may infer a client's local data by analyzing the model parameters or gradients uploaded by the client.
- *Watermark removal or obfuscation attacks*. This attack is initiated by malicious clients targeting model IPR issues. The adversary may attempt to falsify or remove the watermark to make unauthorized claims of model ownership.

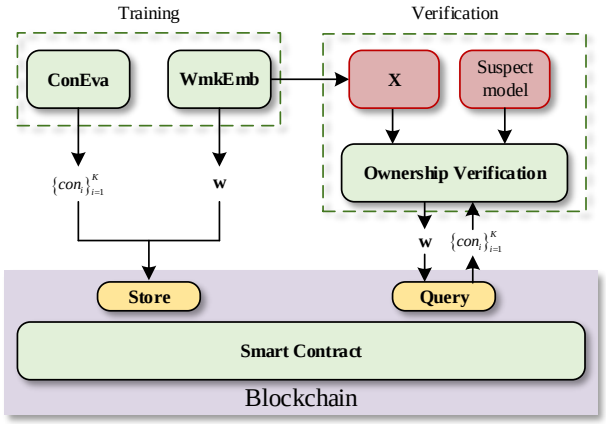


Fig. 2. The algorithmic procedures of TraCemop.

C. Design Goals

Based on the system model and threat model described above, TraCemop aims to achieve the following design goals:

- *Privacy preservation*: TraCemop should protect participants' training data from data inference attacks during aggregation. Additionally, it should safeguard the watermark embedding matrix of the owner during the IPR verification phase.
- *Efficient contribution evaluation*: The framework should ensure that contribution evaluation is conducted efficiently.
- *Robust watermark*: The embedded watermark should remain robust against attacks, including removal and obfuscation attempts.

V. The Proposed Framework

In this section, we begin with an overview of TraCemop, followed by a detailed description of its fundamental components.

A. Overview

The workflow of TraCemop is illustrated in Fig. 1. A concrete example of TraCemop is the smart healthcare system. Assume some healthcare institutions, such as hospitals, can collaboratively train an FL model for medical data analysis using our framework without sharing their data. During training, TraCemop provides a mechanism to assess the contribution of each hospital, and records the contribution on the blockchain, serving as the criterion for subsequent benefit allocation. Moreover, a negotiated watermark is embedded into the model. The watermark serves as critical evidence of model ownership and acts as the index to contributions recorded on the blockchain. Consequently, after verifying model ownership, specific contributions from each hospital can be retrieved from the blockchain, enabling precise and fair profit distribution based on their actual contributions. Algorithm 1 depicts the proposed scheme, which consists of two phases: initialization and training. A detailed description of them is as follows. Besides, the algorithmic procedures involved in the approach are illustrated in Fig. 2.

Algorithm 1: TraCemop

Input: $\boldsymbol{\omega}_i$, $con_i\ (i = 1, 2, ..., K)$, D_i, local model learning rate α
Output: $\boldsymbol{\omega}$

The *TA* generates sk and pk, and sends them to clients in secure channel
The *AS* transmits $\boldsymbol{\omega}^0$ to all clients
$\{U_i\}_{i=1}^K$ negotiates $(\mathbf{w}, \mathbf{X})$ by their $(\mathbf{w}^i, \mathbf{X}^i)$
for *each round* **do**
  for *each client* U_i **do**
    for *each local epoch* **do**
      U_i calculates main task loss and watermarking regularization term:

$$E_o(\boldsymbol{\omega}_i^t) \leftarrow M_i(\boldsymbol{\omega}_i^t, D_i)$$
$$E_r(\boldsymbol{\theta}_i^t) \leftarrow \mathbf{WmkEmb}(\mathbf{w}, \mathbf{X}, \boldsymbol{\theta}_i^t)$$

      U_i calculates the total loss:

$$E(\boldsymbol{\omega}_i^t) = E_o(\boldsymbol{\omega}_i^t) + \beta E_r(\boldsymbol{\theta}_i^t)$$

      Update the local model $\boldsymbol{\omega}_i$ with regard to the total loss:

$$\boldsymbol{\omega}_i^{t+1} \leftarrow \boldsymbol{\omega}_i^t - \alpha \cdot \nabla E(\boldsymbol{\omega}_i^t)$$

    end
  end
  $\{con_i\}_{i=1}^K \leftarrow$**ConEva**$(D_i, M_i(\boldsymbol{\omega}_i^t))$
  U_i encrypts $\boldsymbol{\omega}_i^t$ by pk
  U_i transmits $ct_{\boldsymbol{\omega}_i^t}$ to the *AS*
  The *AS* updates the global model:

$$ct_{\boldsymbol{\omega}^{t+1}} = \sum_{i=1}^{K} con_i ct_{\boldsymbol{\omega}_i^t}$$

  The *AS* sends $ct_{\boldsymbol{\omega}^{t+1}}$ to clients
  U_i decrypts $ct_{\boldsymbol{\omega}^{t+1}}$ by sk to get $\boldsymbol{\omega}^{t+1}$
end

Algorithm 2: Smart Contract

Data: state value $st \in \{0, 1, 2\}$, contribution values $\{con_i^t\}_{i=1}^K$ in t-th round, watermark $\mathbf{w}$, current round t
Result: response res

Function Login(st, $\mathbf{w}$):
  conAddr $\leftarrow \mathbf{w}$
  if *conAddr is empty* **then**
    Create con_list$_{\mathbf{w}} = []$ in conAddr
    return 1
  end
  else
    return 0
  end
Function Store(st, $\mathbf{w}$, $\{con_i^t\}_{i=1}^K$, t):
  conAddr $\leftarrow \mathbf{w}$
  In conAddr, con_list$_{\mathbf{w}}$.$append(\{con_i^t\}_{i=1}^K)$
  return 1
Function Query(st, $\mathbf{w}$):
  conAddr $\leftarrow \mathbf{w}$
  que_list $\leftarrow$ con_list$_{\mathbf{w}} = [\{con_i^1\}_{i=1}^K, \{con_i^2\}_{i=1}^K, \ldots, \{con_i^t\}_{i=1}^K]$ in conAddr
  return que_list

Main (st, $\mathbf{w}$, $\{con_i^t\}_{i=1}^K$, t):
  while 1 do
    The *BC* receives message from the *AS*
    if st**==0 then**
      res $\leftarrow$ **Login**(st, $\mathbf{w}$)
    end
    if st**==1 then**
      res $\leftarrow$ **Store**(st, $\mathbf{w}$, $\{con_i^t\}_{i=1}^K$)
    end
    if st**==2 then**
      res $\leftarrow$ **Query**(st, $\mathbf{w}$)
    end
    return res
  end

1) Initialization: The *TA* generates a key pair sk/pk for homomorphic encryption and sends them to the clients through a secure channel. The *AS* initializes the global model $M(\boldsymbol{\omega}^0)$ and publishes it to the clients. All clients negotiate a T-bit watermark and the watermark embedding matrix $(\mathbf{w}, \mathbf{X})$, and broadcast $\mathbf{w}$ to the *AS*. Finally, the *AS* uploads the watermark and a state value $st = 0$ to the *BC*, and the *BC* performs the **Login** function in Algorithm 2. Specifically, the *BC* allocates a storage area for the current training and uses the watermark $\mathbf{w}$ as the data storage address to record the contribution values for each training round according to the smart contract. This achieves watermark-based storage of contribution values, while preventing malicious tampering of records and ensuring traceability for all contribution records.

2) Training: U_i trains the local model $M_i(\boldsymbol{\omega}_i, D_i)$ using local dataset and embeds the watermark $(\mathbf{w}, \mathbf{X})$ into it. The process of **Watermark Embedding** is illustrated in Section V-C. The objective of local training is to minimize the total loss $E(\boldsymbol{\omega}_i)$, which includes the main task loss $E_o(\boldsymbol{\omega}_i)$ and the regularization term $E_r(\boldsymbol{\theta}_i)$ for the embedded watermark:

$$E(\boldsymbol{\omega}_i) = E_o(\boldsymbol{\omega}_i) + \beta E_r(\boldsymbol{\theta}_i)$$

where β is a hyperparameter.

After local training, clients perform **Contribution Evaluation** to estimate their contribution level and submit the assessment results to the *AS*. Subsequently, the *AS* transmits the contribution values to the *BC* for storage. After the training, the client can query the contribution values for the training task by executing the **Query** function in the smart contract.

Clients encrypt the local model parameters with pk and transmit them to the *AS*. The *AS* updates the global model by aggregating the local model weighted by the contribution

values[2]. The *AS* then sends the updated global model to the clients, who update their local models by decrypting the global model parameters with sk. The training process continues until the global model converges.

B. Contribution Evaluation

At the end of each local training phase, clients present artificial samples derived from their local datasets to other participants. These samples are employed to assess the predictive capabilities of the other participants' models. The algorithm is divided into three stages: sample generation, contribution initialization, and contribution normalization. **Contribution Evaluation** is shown in Algorithm 3.

1) Sample Generation: Participants use their local datasets to generate artificial samples through a well-trained Differentially Private Generative Adversarial Network (DPGAN) [7]. The generated sample set from each client is proportional to the class distribution in its local dataset, ensuring consistent data distribution across the artificial samples. They label these artificial samples locally.

2) Contribution Initialization: Each participant randomly selects N artificial samples and releases them to other clients. Upon receiving these artificial samples, each participant labels them using their own model. Subsequently, each participant collects the predicted results for its artificial samples from other participants and determines the majority vote for each sample. This majority vote is deemed as the correct label for the sample. Accordingly, participants can determine the correct label set for the selected N artificial samples and calculate the raw contribution value c_i^j for others as follows:

$$c_i^j = cor_i/N$$

where c_i^j is U_i's raw contribution value calculated by U_j, and cor_i is the number of correct predictions. Finally, the participants send all their c_i^j to the *AS*.

3) Contribution Normalization: The server computes the normalized contribution value as follows:

$$con_i = \frac{\frac{\sum_{j\neq i} c_i^j}{K-1}}{\sum_{i=1}^{K} \frac{\sum_{j\neq i} c_i^j}{K-1}}$$

where con_i represents the normalized contribution of U_i. This value reflects the engagement of participants in model training and is used as a weight for model parameters during aggregation. The *AS* submits the contribution values $\{con_i^t\}_{i=1}^K$ of this round t, the joint watermark $\mathbf{w}$, and a state value st=1 to the *BC*. Then, the $\{con_i^t\}_{i=1}^K$ is stored in the corresponding address $\mathbf{w}$ by the function **Store** in the smart contract. The blockchain inherently possesses tamper-resistant and traceable characteristics. Therefore, storing contribution values on *BC* ensures the traceability of contribution in each training round and prevents tampering with the contribution information. The *AS* confirms that the response res from *BC* is 1. If not, the *AS* retries the upload of $\{con_i^t\}_{i=1}^K$ until res $= 1$.

[2]Our contribution assessment mechanism objectively quantifies the data quality among participants, serving as weights for model aggregation and facilitating corresponding model adjustments and optimization.

Algorithm 3: Contribution Evaluation—$\mathbf{ConEva}(\cdot)$

Input: $\boldsymbol{\omega}_i^t$, D_i, $M_i(\cdot)$
Output: $\{con_i\}_{i=1}^K$

/* **Sample generation** */
U_i generates their own artificial sample set $ASam_i$ through DPGAN
U_i labels all samples in $ASam_i$ with $M_i(\boldsymbol{\omega}_i^t, ASam_i)$
/* **Contribution initialization** */
for i *in* $1, 2, \ldots, K$ **do**
  $\{U_j\}_{j\neq i}^K$ releases a portion of their artificial samples $as_j \subseteq ASam_j$ to U_i
  U_i labels these samples with $M_i(\boldsymbol{\omega}_i^t, as_j)$, and sends the results to the corresponding owners
end
for j *in* $1, 2, \ldots, K$ **do**
  U_j calculates the raw contribution value c_i^j for $\{U_i\}_{i\neq j}^K$ from the labeling results, and sends $\{c_i^j\}_{i=1}^K$ to the *AS*
end
/* **Contribution normalization** */
for i *in* $1, 2, \ldots, K$ **do**
  The *AS* computes the average of $\{c_i^j\}_{j\neq i}^K$ of U_i:

$$avg_i = \frac{\sum_{j=1, j\neq i}^{K} c_i^j}{K-1}$$

end
The *AS* forms the normalized contribution $\{con_i\}_{i=1}^K$ of each client
for i *in* $1, 2, \ldots, K$ **do**

$$con_i = \frac{avg_i}{\sum_{j=1}^{K} avg_j}$$

end

C. Ownership Protection

The declaration of model IPR is achieved by embedding a watermark into the model. The workflow of the ownership protection mechanism is outlined as follows.

1) Watermark Consultation: Since the FL model is collaboratively trained by clients, the IPR of the model should be jointly held by all participants. Therefore, participants need to collaboratively negotiate a legal watermark and the corresponding watermark embedding matrix. Specifically, each client $U_i \in \{1, 2, \ldots, K\}$ generates a T-bit local watermark $\mathbf{w}^i \in \{0, 1\}^T$, and a watermark embedding matrix $\mathbf{X}^i$. Each client shares $(\mathbf{w}^i, \mathbf{X}^i)$ to other clients through a secure channel. Upon receiving the watermark information from other clients, each client locally computes the legal watermark information $(\mathbf{w}, \mathbf{X})$ as follows:

$$\mathbf{w} = \left(\sum_{i=1}^{K} \mathbf{w}^i\right) \mod 2$$

Algorithm 4: Watermark Embedding—$\mathbf{WmkEmb}(\cdot)$

Input: $M_i(\cdot)$, $\boldsymbol{\omega}_i$, $\boldsymbol{\theta}_i$ D_i, $(\mathbf{w}, \mathbf{X})$
Output: The local model $M_i(\boldsymbol{\omega})$ with a watermark B embedded

for *each round* **do**
  U_i trains $M_i(\boldsymbol{\omega}_i)$ with D_i, and computes original loss $E_o(\boldsymbol{\omega}_i)$ of the main task
  U_i computes the watermark loss $E_r(\boldsymbol{\theta}_i)$, where

$$E_r(\boldsymbol{\theta}) = \begin{cases} 0.5 \cdot \delta \left(\mathrm{w}_t - y_t\right)^2 & |\mathrm{w}_t - y_t| < \delta \\ \delta \cdot |\mathrm{w}_t - y_t| - 0.5 \cdot \delta^2 & otherwise \end{cases}$$

  Optimize the total loss $E(\boldsymbol{\omega}_i) = E_o(\boldsymbol{\omega}_i) + \beta E_r(\boldsymbol{\theta}_i)$
  Update model parameters $\boldsymbol{\omega}_i$
end

$$\mathbf{X} = \sum_{i=1}^{K} \mathbf{X}^i$$

where

$$\mathbf{w} = \left(\mathrm{w}_0, \mathrm{w}_1, \ldots, \mathrm{w}_{T-1}\right), \ \mathrm{w}_t \in \{0, 1\}$$

$$\mathbf{X} = \left[\mathbf{x}_0, \mathbf{x}_1, \ldots, \mathbf{x}_{T-1}\right]' = \begin{bmatrix} x_{01} & \cdots & x_{0L} \\ \vdots & \ddots & \vdots \\ x_{(T-1)1} & \cdots & x_{(T-1)L} \end{bmatrix}.$$

Here, $\mathbf{w}$ is the watermark, and $\mathbf{X}$ is the watermark embedding matrix. Model owners should keep $\mathbf{X}$ confidential as it serves as the credential for model ownership verification.

The size of $\mathbf{X}$ is $T \times L$, where L is related to the layer $\boldsymbol{\theta}$ containing the watermark. In TraCemop, the watermark is embedded into the first convolutional layer to mitigate the impacts of model pruning and fine-tuning on the embedded watermark.

2) Watermark Embedding: During the training phase, participants embed the target watermark $\mathbf{w}$ into their local model by adding a regularization term that measures the similarity between the target watermark $\mathbf{w}$ and the extracted watermark $\hat{\mathbf{w}}$. The regularization term $E_r(\boldsymbol{\theta})$ in TraCemop is L1-smooth and is defined as follows.

$$E_r(\boldsymbol{\theta}) = \begin{cases} 0.5 \cdot \delta \left(\mathrm{w}_t - y_t\right)^2 & |\mathrm{w}_t - y_t| < \delta \\ \delta \cdot |\mathrm{w}_t - y_t| - 0.5 \cdot \delta^2 & otherwise. \end{cases}$$

In the regularization term, $y_t = \sigma\left(\langle \mathbf{X}_t, \boldsymbol{\theta} \rangle\right)$, where $\langle \cdot \rangle$ is the inner product, $\mathbf{X}_t$ is the t-th row of $\mathbf{X}$, and $\sigma\left(\cdot\right)$ is the sigmoid function defined as $\sigma(x) = \frac{1}{1+e^{-x}}$. δ is a hyperparameter that controls the similarity between w_t and y_t. The primary goal of iterative optimization of the regularization term is to ensure that the influence of the watermark embedding matrix on the model parameters eventually converges to the target watermark. This process ultimately facilitates the effective embedding of the watermark into the model. The watermark embedding process is detailed in Algorithm 4.

D. Ownership Verification

Model owners can assert their ownership of potentially stolen models by evaluating the similarity between the watermark retrieved from the model parameters and the original watermark. According to the ownership verification phase in [6], the model owner provides its watermark embedding matrix $\mathbf{X}$ to the model user, who then extracts the watermark $\hat{\mathbf{w}}$ from the model as follows:

$$\hat{w}_t = s(\langle \mathbf{X}_t, \hat{\boldsymbol{\theta}} \rangle).$$

Here, $s(\cdot)$ is a step function:

$$s(x) = \begin{cases} 1 & x \geq 0 \\ 0 & x < 0. \end{cases}$$

Subsequently, the model user supplies the original watermark $\mathbf{w}$ and compares it with the extracted watermark $\hat{\mathbf{w}}$ to verify ownership. The white-box watermark ownership verification can be directly determined by watermark detection rate η_s, which is defined as follows:

$$\eta_s = H(\mathbf{w}, \hat{\mathbf{w}})/T.$$

If η_s is larger than the given threshold th_{wm}, which is set to 0.8 in this paper, the model is confirmed to belong to the provider of $\mathbf{X}$. However, in the standard ownership verification process, $\mathbf{X}$ would be exposed to the model user, potentially leading to privacy concerns. To prevent this, TraCemop incorporates a privacy-preserving copyright verification mechanism based on Secure Two-Party Computation (S2PC). In this approach, the provider of $\mathbf{X}$ is referred to as the "prover", while the model user assumes the role of the "verifier". The prover must demonstrate to the verifier that he/she is indeed the legitimate owner of the model. To ensure the effectiveness and credibility of the verification process, the model IPR verification should be led by the model user and the verification results are ultimately obtained by the model user. This not only guarantees the high trust of the model user in the verification process but also mitigates the risk of potential cheating by the prover. In particular, the model owner and the model user generate L Beaver triples $(\mathbf{a}, \mathbf{b}, \mathbf{c})$, where $\mathbf{a} = [a_1, a_2, \ldots, a_L]$, $\mathbf{b} = [b_1, b_2, \ldots, b_L]$, $\mathbf{c} = [c_1, c_2, \ldots, c_L]$, and $[c]_i = [a]_i \cdot [b]_i$. They perform additive secret sharing on their respective $\mathbf{X}$ and $\boldsymbol{\theta}$ and then conduct the watermark extraction computations on these shared values. As a result, the model user securely obtains the extraction outcome without any leakage of $\mathbf{X}$ and $\boldsymbol{\theta}$. The specific steps of this privacy-preserving ownership verification are detailed in Algorithm 5, where the provider of $\mathbf{X}$ is referred to as the "prover". This algorithm guarantees that neither $\mathbf{X}$ nor $\boldsymbol{\theta}$ is exposed during the ownership verification process.

VI. Analysis

We first validate the correctness of the watermark extraction phase employed by S2PC and examine the security properties of TraCemop. Furthermore, we provide an informal analysis to demonstrate the privacy-preserving aspects of the ownership verification phase. Finally, we analyze the complexity of TraCemop.

Algorithm 5: Ownership Verification

Input: $\mathbf{X}$, $\boldsymbol{\theta}$, $\mathbf{w}$
Output: η_s

Model user and prover agree on L Beaver triples $(\mathbf{a}, \mathbf{b}, \mathbf{c})$, where the size of $\mathbf{a}$ and $\mathbf{b}$ is the same as $\boldsymbol{\theta}$
Model user randomly divides its $\boldsymbol{\theta}$ into two shares $\boldsymbol{\theta}_1$ and $\boldsymbol{\theta}_2$, where $\boldsymbol{\theta} = \boldsymbol{\theta}_1 + \boldsymbol{\theta}_2$
Prover randomly divides its $\mathbf{X}$ into $\mathbf{X}_1$ and $\mathbf{X}_2$, where $\mathbf{X} = \mathbf{X}_1 + \mathbf{X}_2$
Prover sends the share $\mathbf{X}_2$ to model user. Similarly, model user sends the share $\boldsymbol{\theta}_1$ to the prover
/* **Parameter negotiation** */
// **Prover:**
$\mathbf{e}_1 = \boldsymbol{\theta}_1 - \mathbf{b}_1$
for *i in each row of* $\mathbf{X}_1$ **do**
  $[\mathbf{I}_1]_i = [\mathbf{X}_1]_i - \mathbf{a}_1$
end
// **Model user:**
$\mathbf{e}_2 = \boldsymbol{\theta}_2 - \mathbf{b}_2$
for *i in each row of* $\mathbf{X}_2$ **do**
  $[\mathbf{I}_2]_i = [\mathbf{X}_2]_i - \mathbf{a}_2$
end
The prover and model user jointly calculate $\mathbf{I}$ and $\mathbf{e}$ by the following formula:
$\mathbf{I} = \mathbf{I}_1 + \mathbf{I}_2$
$\mathbf{e} = \mathbf{e}_1 + \mathbf{e}_2$
/* **Partial calculation** */
// **Prover:**
for *i in each row of* $\mathbf{X}$ **do**
  $[PC_1]_i = \sum_{j=1}^{|\boldsymbol{\theta}_1|}([\mathbf{I}]_{i,j} \cdot [\mathbf{e}]_j + [\mathbf{I}]_{i,j} \cdot [\mathbf{b}_1]_j + [\mathbf{a}_1]_j \cdot [\mathbf{e}]_j + [\mathbf{c}_1]_j)$
end
// **Model user:**
for *i in each row of* $\mathbf{X}$ **do**
  $[PC_2]_i = \sum_{j=1}^{|\boldsymbol{\theta}_2|}([\mathbf{I}]_{i,j} \cdot [\mathbf{b}_2]_j + [\mathbf{a}_2]_j \cdot [\mathbf{e}]_j + [\mathbf{c}_2]_j)$
end
/* **Watermark extraction** */
The prover sends PC_1 to the model user
// **Model user:**
$\mathbf{X}\boldsymbol{\theta}_{mul} = \mathbf{X} \cdot \boldsymbol{\theta} = PC_1 + PC_2$
for *i in the row of* $\mathbf{X}\boldsymbol{\theta}_{mul}$ **do**
  $\hat{\mathrm{w}}_i = s([\mathbf{X}\boldsymbol{\theta}_{mul}]_i)$
end
Compare the extracted watermark $\hat{\mathbf{w}} = (\hat{\mathrm{w}}_0, \hat{\mathrm{w}}_1, \ldots, \hat{\mathrm{w}}_{\hat{T}-1})$ with the target watermark $\mathbf{w} = (\mathrm{w}_0, \mathrm{w}_1, \ldots, \mathrm{w}_{T-1})$ by calculating the Hamming distance $H(\mathbf{w}, \hat{\mathbf{w}})$
Calculate the watermark detection rate $\eta_s = H(\mathbf{w}, \hat{\mathbf{w}})/T$

A. Correctness

Theorem 1. *During the IPR verification, the embedded watermark can be correctly extracted by S2PC.*

Proof. The detailed proof is given in Appendix A of the supplemental material, available online. □

B. Rationality Analysis

In Section III, we establish fairness as a key criterion for assessing the reasonableness of contribution evaluations. In this paper, our contribution assessment is grounded in the prediction accuracy achieved by local models during the training process. We provide proof of the fairness of the contribution evaluation mechanism in TraCemop.

Theorem 2. In TraCemop, the contribution evaluation mechanism is fair and can effectively measure the contribution of each client.

Proof. The detailed proof is given in Appendix B of the supplemental material, available online. □

C. Formal Security Analysis

A public-key cryptosystem $\sigma = (\mathsf{Gen}, \mathsf{En}, \mathsf{Dec})$ is Indistinguishability under Chosen Plaintext Attack (*IND-CPA*) if a probabilistic polynomial-time (PPT) adversary $\mathcal{A}$ cannot distinguish between two messages of equivalent length m_1 and m_2 captured from the message domain M.

Definition 1 (IND-CPA Security)**.** *Let σ be operations in a homomorphic encryption scheme. We define a game $\mathsf{Expr}_{\mathsf{CPA}}[\mathcal{A}]$ parameterized by $coin \in \{0, 1\}$ and a probabilistic polynomial-time adversary $\mathcal{A}$. The game played by $\mathcal{A}$ is as follows:*

$$\begin{aligned}\mathsf{Expr}_{\mathsf{CPA}}[\mathcal{A}](1^k) :& (sk, pk) \leftarrow \mathsf{Gen}(1^k)\\ & (m_1, m_2) \leftarrow \mathcal{A}(1^k, pk)\\ & ct = \mathsf{En}_{pk}(m_{coin})\\ & coin' \leftarrow \mathcal{A}(ct)\\ & \textit{return } coin'.\end{aligned}$$

The encryption scheme is considered to be IND-CPA *secure if, for any $\mathcal{A}$, the advantage*

$$\begin{aligned}\mathsf{Adv}_{\mathsf{CPA}}[\mathcal{A}](k) = \big|&\Pr[\mathsf{Expr}_{\mathsf{CPA}}[\mathcal{A}](1^k) = 1 \mid coin = 0]\\ &- \Pr[\mathsf{Expr}_{\mathsf{CPA}}[\mathcal{A}](1^k) = 1 \mid coin = 1]\big|\end{aligned}$$

is negligible.

In TraCemop, data security during transmission is ensured using the CKKS homomorphic encryption scheme [30], which is proven secure under *IND-CPA* [32]. We demonstrate that TraCemop can guarantee the data security through the following theorem.

Theorem 3. In TraCemop, assuming that the AS and the client are honest-but-curious and that no collusion occurs between them, the data encrypted by CKKS is secure during transmission.

Proof. The detailed proof is given in Appendix C of the supplemental material, available online. □

D. Informal Security Analysis

1) Resistance to Data Reconstruction Attack: During the training process, all the transmitted model parameters $\boldsymbol{\omega}$ are encrypted by pk. According to *Theorem* 2, the transmitted

information is secure, so that the adversary is unable to access plaintext model parameters, preventing them from reconstructing the participant's training data D_i based solely on the model parameters. Consequently, TraCemop effectively mitigates the risk of data reconstruction attacks.

2) Resistance to Watermark Removal Attack: Throughout the training phase, all model parameters are encrypted during transmission, preventing adversaries from directly accessing the model parameters and thwarting any attempts to remove the embedded joint watermark $\mathbf{w}$. The malicious client may attempt to collude with each other, abstaining from embedding the watermark $\mathbf{w}$ during local training, thereby reducing its presence in the global model. However, the joint watermark $\mathbf{w}$ embedded by honest clients will still be integrated into the global model, guiding the model toward maintaining the watermark during global aggregation. Consequently, watermark removal attacks are ineffective in TraCemop.

3) Resistance to Watermark Obfuscation Attack: Adversaries may seek to gain exclusive control over the model by embedding a new watermark to obscure the legitimate one. However, since most participants will embed the legitimate watermark $\mathbf{w}$, the influence of the malicious watermark is diminished during the aggregation phase. Such an attack is unlikely to succeed unless more than half of the participants collude to embed an identical malicious watermark.

Furthermore, during ownership verification, the watermark detection rate η_s between the extracted watermark $\hat{\mathbf{w}}$ and the original watermark $\mathbf{w}$ must meet a specified threshold th_{wm}. This ensures that the legitimate watermark is accepted while the malicious watermark is rejected. In Section VII, we demonstrate the robustness of our framework against obfuscation attacks. Extensive experimental results illustrated in Fig. 8 show that TraCemop can effectively distinguish the legitimate and malicious watermark under the threshold th_{wm}.

4) Privacy-preserving Model Ownership Verification: In TraCemop, the copyright proof requires the model owner to provide secret shares of $\mathbf{X}$ to facilitate watermark extraction. At this stage, the model user cannot reconstruct $\mathbf{X}$ from the secret shares, preventing the leakage of $\mathbf{X}$. Moreover, the extraction process involves computing the dot product between $\mathbf{X}$ and $\boldsymbol{\theta}$. Although the model user obtains the final result $\mathbf{X}\cdot\boldsymbol{\theta}$, they cannot derive any information about $\mathbf{X}$. This is because numerous studies have shown that under the semi-honest threat model, adversaries cannot recover vector information from the dot product result if the number of unknown variables far exceeds the number of linear equations [33]. In our framework, the number of elements in $\mathbf{X}$ is much larger than the length of $\mathbf{w}$, preventing the model user from inferring information about $\mathbf{X}$ from $\hat{\mathbf{w}}$.

E. Complexity Analysis

1) Computation Overhead: We discuss the computational complexity of TraCemop, focusing on each client U_i and the *AS* during the training phase. The analysis is as follows.

- *Client*: $O(K\cdot N+K+1)$. The computation cost of each client U_i consists of three parts.
  - Label $K\cdot N$ artificial samples in the contribution initialization stage, which takes $O(K\cdot N)$.
  - Calculate $K-1$ c_i^j for other clients in the contribution normalization stage, which takes $O(K)$.
  - Encrypt model parameters $\boldsymbol{\omega}^t$ and obtains the CKKS ciphertext $ct_{\boldsymbol{\omega}^t}$ during the global model updates phase, which takes $O(1)$.

  Overall, the computation overhead of U_i is $O(K\cdot N+K+1)$.
- *AS*: $O(K)$. The computation overhead of *AS* falls into two parts.
  - Compute the average of the raw contribution values of each client in the contribution normalization stage, which takes $O(K)$. Besides, it should normalize the contribution values for all clients, which takes $O(K)$.
  - Aggregate the K model into the global model during the aggregation phase, which takes $O(K)$.

  Overall, the computation overhead of *AS* is $O(K)$.

2) Communication Overhead: We present the analysis of the communication overhead for U_i, *AS*, and *BC*, as follows.

- *Client*: $O(K\cdot N+1)$. The communication overhead of U_i can be divided into three parts.
  - Receive $(K-1)\cdot N$ artificial samples from other clients, which takes $O(K\cdot N)$.
  - Send $(K-1)\cdot N$ predicted results to the corresponding owners, which takes $O(K\cdot N)$.
  - Upload the encrypted model parameters to *AS*, which takes $O(1)$.

  Overall, the communication overhead of U_i is $O(K\cdot N+1)$.
- *AS*: $O(K^2+K+1)$. The communication overhead of *AS* falls into four parts.
  - Receive the raw contribution value $\{con_i\}_{i=1}^K$ list from K clients, which takes $O(K^2)$.
  - Upload the normalized contribution to *BC*, which takes $O(1)$.
  - Receive K encrypted model parameters $\{ct_{\boldsymbol{\omega}^t}\}_1^K$, which takes $O(K)$
  - Send the ciphertext of updated model parameters $ct_{\boldsymbol{\omega}^{t+1}}$ to K clients, which takes $O(K)$

  Overall, the communication overhead of *AS* is $O(K^2+K+1)$.
- *BC*: $O(1)$. The communication overhead of *BC* is receiving the normalized contribution from *AS*, which takes $O(1)$.

VII. Performance Evaluation

In this section, we conduct experiments to evaluate the performance of TraCemop. We compare our contribution evaluation mechanism against a Shapley-based scheme to demonstrate its accuracy and efficiency. Additionally, we assess the efficacy and robustness of IPR protection within TraCemop.

A. Experiment Settings

- *Evaluation Environment.* We implement TraCemop using Python 3.7 and conduct experiments on a desktop equipped with an Intel i7-10700 8-Core processor, 32GB of RAM, and the Windows 10 operating system. The

TABLE II
CONTRIBUTION DISTRIBUTION

Number of clients	Scheme	Client 1	Client 2	Client 3	Client 4	Client 5	Client 6	Cosine Similarity
3	Shapley-Based [34]	0.369	0.344	0.286	/	/	/	99.987%
	TraCemop	0.363	0.343	0.293	/	/	/	
4	Shapley-Based [34]	0.253	0.251	0.258	0.240	/	/	99.977%
	TraCemop	0.260	0.251	0.259	0.232	/	/	
5	Shapley-Based [34]	0.205	0.204	0.176	0.192	0.223	/	99.889%
	TraCemop	0.202	0.204	0.190	0.196	0.208	/	
6	Shapley-Based [34]	0.164	0.167	0.165	0.150	0.183	0.171	99.735%
	TraCemop	0.165	0.159	0.168	0.174	0.170	0.163	

TABLE III
ACCURACY OF DIFFERENT SCHEMES

Model	Dataset	10 Clients						100 Clients					
		10%		20%		30%		10%		20%		30%	
		Ours	FedAvg	Ours	FedAvg	Ours	FedAvg	Ours	FedAvg	Ours	FedAvg	Ours	FedAvg
CNN	*MNIST*	92.3%	92.7%	90.2%	91.6%	90.5	89.8%	92.9%	13.2%	91%	89.2%	90.7%	19.7%
AlexNet	*Cifar10*	96.3%	92.8%	88.4%	88.8%	87.1%	86.6%	95.5%	96.1%	88.2%	87.1%	78.6%	72.9%

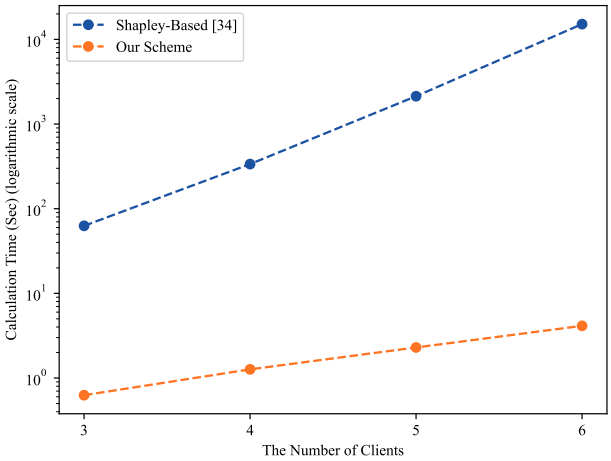
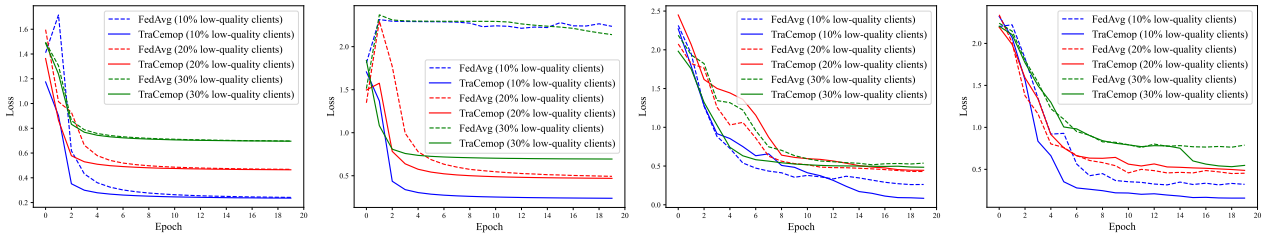


Fig. 3. The running time of contribution evaluation in logarithmic scale.

training is conducted in Paddle and PyTorch. Besides, the blockchain is deployed on a virtual machine equipped with two Cores, 4GB of RAM, and the Centos 7.0 operating system.

- *Model.* In all experiments, we use the AlexNet and a Convolutional Neural Network (CNN) as the network architecture in our framework. The CNN model consists of two convolutional layers, each followed by a pooling layer, and concludes with two fully-connected layers. The Stochastic Gradient Descent (SGD) optimizer is employed with a learning rate of 0.01, and the mini-batch size is set to 64.
- *Datasets.* The experiments are conducted on three well-known datasets including *MNIST*, *FashionMNIST* and *Cifar10*. *MNIST* and *FashionMNIST* contains 60,000 training samples and 10,000 test samples, categorized into 10 classes. *Cifar10* consists of 50,000 colored images for training and 10,000 for testing. In this paper, we focus on low-quality data clients. Since the impacts of Non-Independent and Identically Distributed (Non-IID) and low-quality data clients on model performance are difficult to distinguish, we consider only the Independent and Identically Distributed (IID) setting to ensure a clear clarification of experimental results. The extension to Non-IID scenarios will be addressed in future work.
- *Baselines.* To verify the utility of our contribution evaluation mechanism, we choose the scheme proposed by Ghorbani et al. [34] as the baseline. To explore the aggregation performance of TraCemop, we select FedAvg [28] as the baseline. For the watermarking performance, we present two baselines for comparison. The first is the scheme proposed by Uchida et al. [5], the first white-box watermarking scheme in machine learning. The second baseline is FedIPR [6]. which is a copyright protection solution combining white-box and black-box watermarking in the FL scenario.
- *Evaluation Metrics.* We use cosine similarity as the metric to assess the utility of the contribution evaluation mechanism. To evaluate the performance of the TraCemop, we calculate the loss and accuracy on the aforementioned training and testing set as the metric. To assess the robustness of our watermarking method, we use the similarity between the original watermark and the extracted watermark, named watermark detection rate as an indicator.
- *Federated Learning Settings.* We simulate a horizontal federated learning scenario with default number of clients to 100. The clients upload their local models during each



(a) 10 clients with CNN on *MNIST* (b) 100 clients with CNN on *MNIST* (c) 10 clients with AlexNet on *Cifar10* (d) 100 clients with AlexNet on *Cifar10*

Fig. 4. The training loss of FedAvg and TraCemop with different ratio of low-quality data clients in FL system.

communication round. The server then aggregates these models by taking the contribution values of clients as the aggregation weight.

- *Blockchain Settings.* We implement the blockchain on FISCO BCOS and write the smart contract in Solidity. The blockchain consists of four default nodes and uses the Practical Byzantine Fault Tolerance (PBFT) algorithm for consensus.
- *Watermark Settings.* The watermark is embedded into the first convolutional layer, with the hyperparameter β set to 10.
- *WLAN Settings.* The model owner and model user are simulated on two desktops within the same region, with an average network delay of 31.83 ms and a bandwidth of 18.3 Mbps.
- *Encryption Parameters.* TenSEAL is used for simulating the homomorphic encryption, with the CKKS algorithm [30]. The basic encryption settings are as follows:
  - Algorithm: CKKS
  - Polynomial modulus degree = 8192
  - Coefficient modulus degree = 218
  - Coeff_modulus size: 200 (60 + 40 + 40 + 60) bits
  - Amplification factor = pow (2.0, 40)

B. Utility and Efficiency

To evaluate the utility of the contribution assessment in TraCemop, we conduct a comparative analysis of the contribution evaluation results across different numbers of clients, contrasting our approach with the Shapley-based scheme [34]. Cosine similarity is employed to measure the similarity of contribution evaluation results, as presented in TABLE II. When the total number of clients increases from 3 to 6, the similarity of the contribution evaluation results between TraCemop and the Shapley-based scheme remains consistently high, with the lowest recorded similarity being 99.74%. This high degree of similarity indicates that our framework provides an accurate contribution assessment.

Additionally, we assess the contribution evaluation time of TraCemop and the Shapley-based scheme [34] across various numbers of clients to gauge the efficiency of the proposed framework. As depicted in Fig. 3, the computation time for our framework remains minimal as the total number of clients increases, whereas the Shapley-based scheme exhibits exponential growth in computation time. The time required for contribution evaluation is relatively similar between the two schemes when there are 3 clients. However, as the number of clients increases to 6, the Shapley-based scheme requires up to 15155.99 seconds to complete the evaluation, while our framework completes it in just 4.13 seconds.

Overall, our framework demonstrates a comparable contribution distribution to the existing approach while requiring significantly less computation time. Hence, the contribution evaluation criteria proposed in TraCemop are both reasonable and efficient.

C. Aggregation Robustness

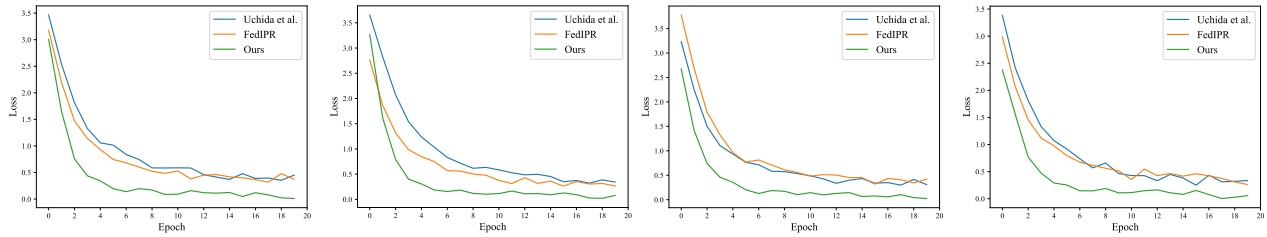
TraCemop leverages contribution evaluation scores from each training round as weights for model aggregation, aiming to mitigate the impact of clients with low-quality data on the global model. To assess the effectiveness of this method, we use FedAvg [28] as a baseline and evaluate model performance in the presence of clients with low-quality data by comparing training loss and accuracy. During the training phase, we set the low-quality client ratio at 10%, 20% as well as 30%, whose training data is covered by the Gaussian noise with a mean of 0 and a variance of 256. As depicted in Fig. 4 and TABLE III, our aggregation method is more stable than FedAvg. In particular, when the proportion of low-quality clients is 10% and 30% in the 100 clients setting, the training loss of FedAvg stabilizes around 2.23, whereas the training loss obtained using our aggregation method stabilizes at 0.231 and 0.694, respectively. Furthermore, the accuracy of TraCemop on the test set is substantially higher than that of FedAvg. This disparity arises because the presence of clients with poor data quality significantly impairs the training performance of FedAvg, often leading to non-convergence results. In contrast, TraCemop effectively identifies clients with poor data quality and dynamically adjusts their aggregation weights, thereby reducing their impact on the global model and ensuring stable convergence. Therefore, the aggregation method proposed in TraCemop demonstrates robust performance against the perturbations caused by low-quality data, maintaining the integrity and accuracy of the global model.

D. Comparison

To assess the impact of the watermark embedding on the main task of the model, we evaluate the convergence and accuracy of TraCemop and compare them with the scheme in [5]

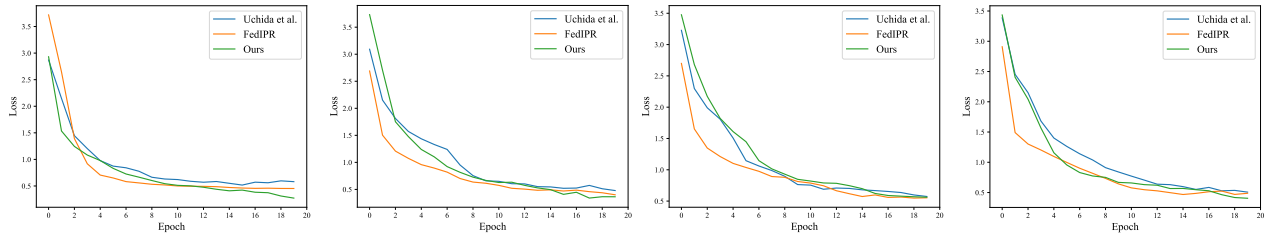
TABLE IV
ACCURACY OF DIFFERENT SCHEMES

Model	Dataset	10 Clients			50 Clients			80 Clients			100 Clients		
		Ours	Uchida et al.	FedIPR	Ours	Uchida et al.	FedIPR	Ours	Uchida et al.	FedIPR	Ours	Uchida et al.	FedIPR
CNN	*MNIST*	99.1%	98.7%	98.8%	98.5%	98.7%	98.5%	98.8%	98%	99.1%	97.2%	97.4%	96.8%
	FashionMNIST	84.2%	83.8%	84.3%	82.6%	82.1%	82%	77.9%	78.1%	78.2%	75.5%	74.6%	75.3%
AlexNet	*Cifar10*	97.1%	97.7%	96.4%	99.1%	96.7%	97.6%	98.7%	98.3%	97.9%	98.3%	97.4%	98.5%



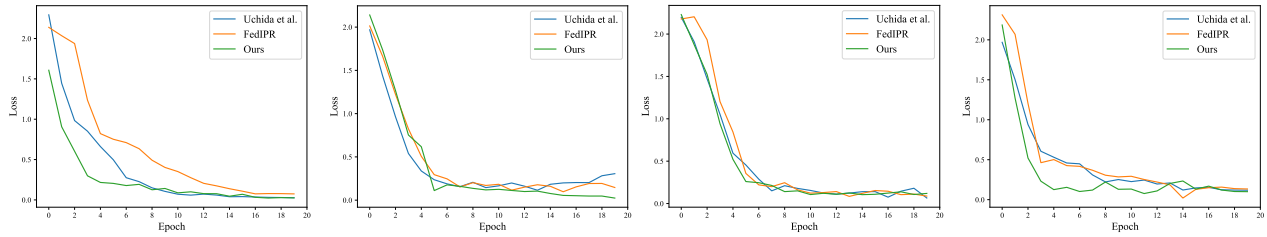
(a) 10 clients (b) 50 clients (c) 80 clients (d) 100 clients

Fig. 5. The loss of different watermarking schemes on *MNIST* with CNN.



(a) 10 clients (b) 50 clients (c) 80 clients (d) 100 clients

Fig. 6. The loss of different watermarking schemes on *FashionMNIST* with CNN.



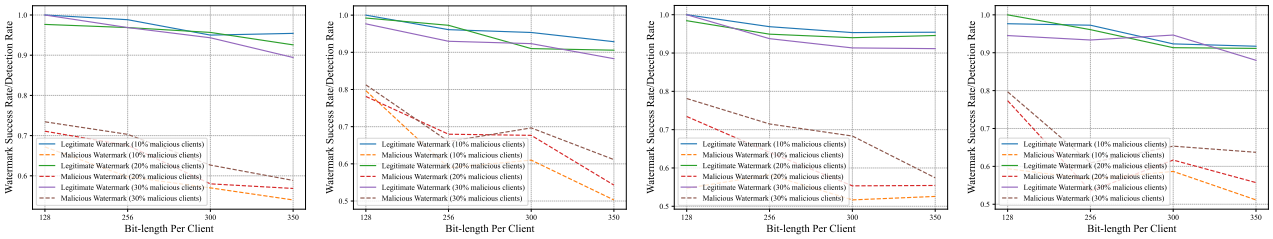
(a) 10 Clients (b) 50 Clients (c) 80 Clients (d) 100 Clients

Fig. 7. The loss of different watermarking schemes on *Cifar10* with AlexNet.

and [6]. As illustrated in Fig. 5, Fig. 6, and Fig. 7, TraCemop demonstrates quicker convergence compared to [5], which is attributed to the use of smooth-L1 as the regularization term in our framework. This term ensures a constant convergence rate, especially when the embedded watermark significantly deviates from the target watermark, leading to faster and more stable convergence during the early stages of training. Upon completion of the training, TraCemop achieves minimum loss across all training datasets compared to [5] and [6]. Taking *MNIST* dataset for example, it is observed that our scheme could achieve a mean loss of 0.043 for different client ratios, which are significantly better than the baselines. Additionally, as shown in TABLE IV, TraCemop attains prediction accuracy of 98.35% on *MNIST*, 80.05% on *FashionMNIST*, and 98.3% on *Cifar10* on average across different client ratio, making 0.2%, 0.4% and 0.775% improvements over those datasets in [6]. This improvement is due to our watermark embedding regularization term, which tends to zero once the embedded watermark is effectively classified as the target watermark using a step function. At this point, the model shifts its focus to optimizing the main task, resulting in enhanced overall performance. Thus, compared to [5] and [6], TraCemop not only achieves precise watermark embedding but also preserves high model accuracy, demonstrating its effectiveness in balancing watermarking with model performance.

E. Watermark Robustness

In an FL system comprising 100 clients, we conduct an analysis of the watermark detection rate of TraCemop under



(a) *CNN on MNIST* (b) *CNN on FashionMNIST* (c) *AlexNet on MNIST* (d) *AlexNet on FashionMNIST*

Fig. 8. The watermark detection rate of different bit length of watermark on different datasets.

watermark obfuscation attacks to demonstrate its robustness. We evaluate the watermark detection rate for different watermark lengths when the proportion of malicious clients in the system is 10%, 20%, and 30%. As shown in Fig. 8, the detection rate of the legitimate watermark remains at or above 0.9, with the worst-case scenario being 0.87, which still exceeds the preset threshold th_{wm} = 0.8. Conversely, the detection rate of malicious watermarks decreases sharply as the bit length increases, remaining below 0.75 when the watermark length exceeds 256 bits. This disparity makes it easier for the model ownership verifier to detect and reject the IPR verification requests involving malicious watermarks. Moreover, on *FashionMNIST* dataset, the minimum differences between the detection rate for malicious watermarks and that for the legitimate watermark of CNN and AlexNet are 0.1484375 and 0.1640625 respectively with a bit length of 128. This gap becomes even more pronounced on the *MNIST*, and continues to widen as the watermark bit length increases on these two datasets. Consequently, the presence of malicious watermarks does not interfere with the legitimate watermark, ensuring that TraCemop maintains strong robustness against watermark obfuscation attacks.

F. Communication Efficiency

Compared with existing watermarking schemes, TraCemop introduces additional interaction rounds to ensure privacy-preserving verification. We further analyze the impact of this increased communication overhead on the efficiency of the verification process. The simulation, conducted on two desktops with identical configurations, reveals that the communication time accounts for no more than 18.6% of the total verification time, as shown in Fig. 9. Although each additional 100 bits of watermark increases the overall running time by approximately one second, requiring a total of 4.373 seconds at 350 bits, the secure verification mechanism added by TraCemop does not significantly contribute to the overall running overhead. Given the privacy protection it offers, this communication overhead is considered to be acceptable. Therefore, TraCemop successfully balances efficiency and security in ownership verification.

G. Blockchain Efficiency

We assume the *BC* platform is an open-source system capable of serving various FL training tasks. To ensure that the consensus process for each FL task does not impact the training efficiency, we test the efficiency of *BC*. Specifically,

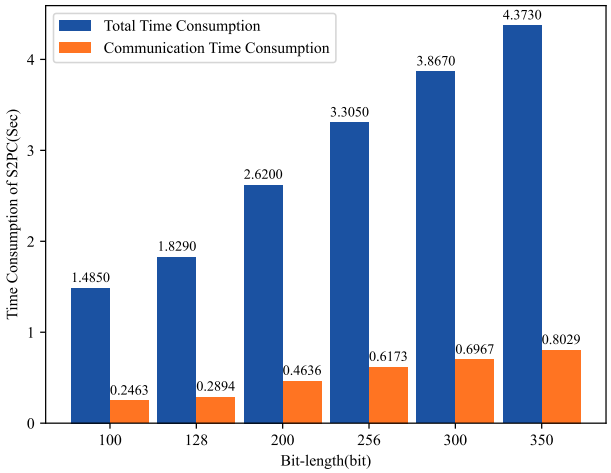


Fig. 9. The running time of ownership verification.

TABLE V
THE BLOCKCHAIN EFFICIENCY

Indicators	Training Time (s)	Consensus Time (s)	TPS
Value	66.07	6.11	221.73

we initiate 2,000 contribution storage requests, where the data stored in the *BC* is in a key-value structure, with the key being the watermark $\mathbf{w}$ and the value being the contribution values $\{con_i^t\}_{i=1}^K$ for each round. The average consensus time and Transactions Per Second (TPS) are presented in TABLE V. Since FL training and *BC* are two independently operating systems, the *AS* can continue to the next training round after uploading the contribution values to the *BC*. Therefore, as long as the response of contribution storage from *BC* arrives at the corresponding *AS* before the end of the next training round, it will not affect the training efficiency. To this end, we measure the average time of one training round and compare it with the average consensus time. As shown in TABLE V, the average time of one training round is 66.07 seconds, significantly greater than the average consensus time of 6.11 seconds. Thus, storing contributions on the blockchain via the smart contract does not impact the efficiency of FL training. Additionally, the average TPS of the entire *BC* system is 221.73, indicating that up to 221 concurrent FL tasks can receive the response of storage request within 1 second, demonstrating that *BC* is in high performance.

VIII. Limitations and Future Work

In this work, we address the challenges of contribution evaluation and model IPR infringement in FL. However, several limitations remain, presenting opportunities for future research.

First, although white-box watermarking provides credible IPR verification, it relies on full access to model parameters. If a model user refuses to cooperate with the model owner, verification becomes infeasible. While Side Channel Analysis (SCA) offers a solution for extracting model parameters under black-box access to a suspect model [35], obtaining full parameter access remains a fundamental challenge for white-box watermarking. Future research could explore effective methods to verify white-box-based model ownership in a black-box manner.

Second, our approach assumes that all FL participants share model copyright equally. However, this assumption overlooks potential IPR disparities arising from varying client contributions. In future work, we aim to investigate dynamic IPR allocation mechanisms that proportionally distribute ownership based on each client's contribution to the global model.

IX. Conclusion

In this paper, we have proposed a federated learning framework called TraCemop, which achieves efficient contribution evaluation and IPR protection. The framework employs homomorphic encryption to protect clients' model parameters in transmission and aggregation. During the verification of model copyright, the S2PC mechanism in TraCemop safeguards the watermark embedding matrix of the model owner. In addition, we have integrated blockchain technology to ensure the transparent recording of contribution evaluation results from each round. Our security analysis demonstrated that TraCemop effectively protects the privacy of federated learning participants during both the training process and model ownership verification. Experimental results across different datasets confirmed that TraCemop is efficient and reliable in assessing user contributions, while also demonstrating resilience against different model ownership attacks.

References

[1] J. Wicaksana, Z. Yan, D. Zhang, X. Huang, H. Wu, X. Yang, and K. Cheng, "FedMix: Mixed Supervised Federated Learning for Medical Image Segmentation," *IEEE Transactions on Medical Imaging*, vol. 42, no. 7, pp. 1955-1968, 2023.

[2] Z. A. El Houda, B. Brik, A. Ksentini, and L. Khoukhi, "A MEC-Based Architecture to Secure IoT Applications using Federated Deep Learning," *IEEE Internet of Things Magazine*, vol. 6, no. 1, pp. 60-63, 2023.

[3] G. Wang, C. X. Dang, and Z. Zhou, "Measure Contribution of Participants in Federated Learning," in Proc. of 2019 IEEE International Conference on Big Data, 2019, pp. 2597-2604.

[4] R. Jia, D. Dao, B. Wang, F. A. Hubis, N. Hynes, N. M. Gurel, B. Li, C. Zhang, D. Song, and C. Spanos, "Towards Efficient Data Valuation Based on the Shapley Value," 2023, arXiv:1902.10275.

[5] Y. Uchida, Y. Nagai, S. Sakazawa, and S. Satoh, "Embedding Watermarks into Deep Neural Networks," in Proc. of 2017 ACM on International Conference on Multimedia Retrieval, 2017, pp. 269-277.

[6] B. Li, L. Fan, H. Gu, J. Li, and Q. Yang, "FedIPR: Ownership Verification for Federated Deep Neural Network Models," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 45, no. 4, pp. 4521-4536, 2023.

[7] X. Zhang, S. Ji, and T. Wang, "Differentially Private Releasing via Deep Generative Model," 2018, arXiv:1801.01594.

[8] 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 Proc. of 2017 ACM SIGSAC Conference on Computer and Communications Security, 2017, pp. 1175-1191.

[9] L. T. Phong, Y. Aono, T. Hayashi, L. Wang, and S. Moriai, "Privacy-Preserving Deep Learning via Additively Homomorphic Encryption," *Transactions on Information Forensics and Security*, vol. 13, no. 5, pp. 1333-1345, 2018.

[10] Y. Lu, X. Huang, Y. Dai, S. Maharjan, and Y. Zhang, "Federated Learning for Data Privacy Preservation in Vehicular Cyber-Physical Systems," *IEEE Network*, vol. 34, no. 3, pp. 50-56, 2020.

[11] Z. Ma, J. Ma, Y. Miao, Y. Li, and R. H. Deng, "ShieldFL: Mitigating Model Poisoning Attacks in Privacy-Preserving Federated Learning," *Transactions on Information Forensics and Security*, vol. 17, pp. 1639-1654, 2022.

[12] T. van Campen, H. Hamers, B. Husslage, and R. Lindelauf, "A New Approximation Method for The Shapley Value Applied to The WTC 9/11 Terrorist Attack," *Social Network Analysis and Mining*, vol. 8, no. 1, pp. 1-12, 2018.

[13] R. H. L. Sim, Y. Zhang, M. C. Chan, and B. K. H. Low, "Collaborative Machine Learning with Incentive-Aware Model Rewards," in Proc. of the 37th International Conference on Machine Learning, 2020, pp.8927-8936.

[14] H. Chen, B. D. Rouhani, C. Fu, J. Zhao, and F. Koushanfar, "DeepMarks: A Secure Fingerprinting Framework for Digital Rights Management of Deep Learning Models," in Proc. of the 2019 on International Conference on Multimedia Retrieval, 2019, pp. 105-113.

[15] P. Lv, P. Li, S. Zhang, K. Chen, R. Liang, H. Ma, Y. Zhao, and Y. Li, "A Robustness-Assured White-Box Watermark in Neural Networks," *IEEE Transactions on Dependable and Secure Computing*, vol. 20, no. 6, pp. 5214-5229, 2023.

[16] Y. Adi, C. Baum, M. Cisse, B. Pinkas, and J. Keshet, "Turning Your Weakness into A Strength: Watermarking Deep Neural Networks by Backdooring," in Proc. of the 27th USENIX Security Symposium, 2018, pp. 1615-1631.

[17] H. Jia, C. A. Choquette-Choo, V. Chandrasekaran, and N. Papernot, "Entangled Watermarks as a Defense against Model Extraction," in Proc. of the 30th USENIX Security Symposium, 2021, pp. 1937-1954.

[18] P. Lv, P. Li, S. Zhu, S. Zhang, K. Chen, R. Liang, C. Yue, F. Xiang, Y. Cai, H. Ma, Y. Zhang, and G. Meng, "SSL-WM: A Black-Box Watermarking Approach for Encoders Pre-trained by Self-Supervised Learning," in Proc. of Network and Distributed System Security, 2024, pp. 1-16.

[19] M. Lansari, R. Bellafqira, K. Kapusta, V. Thouvenot, O. Bettan, and G. Coatrieux, "When Federated Learning Meets Watermarking: A Comprehensive Overview of Techniques for Intellectual Property Protection," *Machine Learning and Knowledge Extraction*, vol. 5, no. 4, pp. 1382-1406, 2023.

[20] X. Yuan, J. Liu, B. Wang, W. Wang, B. Wang, T. Li, X. Ma, and W. Pedrycz, "FedComm: A Privacy-Enhanced and Efficient Authentication Protocol for Federated Learning in Vehicular Ad-Hoc Networks," *Transactions on Information Forensics and Security*, vol. 19, pp. 777-792, 2024.

[21] T. Wu, X. Li, Y. Miao, M. Xu, H. Zhang, X. Liu, and K. -K. R. Choo, "CITS-MEW: Multi-Party Entangled Watermark in Cooperative Intelligent Transportation System," *IEEE Transactions on Intelligent Transportation Systems*, vol. 24, no. 3, pp. 3528-3540, 2023.

[22] Y. Luo, Y. Li, S. Qin, Q. Fu, and J. Liu, "Copyright Protection Framework for Federated Learning Models against Collusion Attacks," *Information Sciences*, vol. 680, p. 121161, 2024.

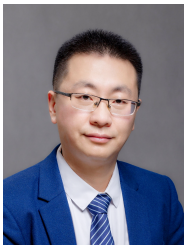
[23] L. Zhang, C. Tang, H. Liu, H. Yu, X. Zhuang, Q. Zhao, L. Wang, W. Fang, and X. Li, "FedMark: Large-Capacity and Robust Watermarking in Federated Learning," in Proc. of 2024 IEEE 44th International Conference on Distributed Computing Systems, 2024, pp. 821-832.

[24] Y. Wu, H. Cheng, L. Guan, P. Liu, F. Chen and M. Wang, "VPFL: A Verifiable Property Federated Learning Framework Against Invisible Attacks in Distributed IoT," in Proc. of 2023 IEEE International Conference on High Performance Computing & Communications, Data Science & Systems, Smart City & Dependability in Sensor, Cloud & Big Data Systems & Application, 2023, pp. 671-678.

[25] M. Yan, Z. Su, Y. Wang, X. Ran, Y. Liu, and T. H. Luan, "Shared DNN Model Ownership Verification in Cross-Silo Federated Learning: A GAN-Based Watermark Approach," in Proc. of 2023 IEEE Global Communications Conference, 2023, pp. 1807-1811.

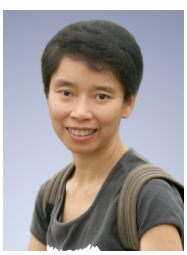
[26] W. Yang, S. Shao, Y. Yang, X. Liu, X. Liu, Z. Xia, G. Schaefer, and H. Fang, "Watermarking in Secure Federated Learning: A Verification

Framework Based on Client-Side Backdooring," *ACM Transactions on Intelligent Systems and Technology*, vol. 15, no. 1, pp. 1-25, 2024.
[27] S. Shao, W. Yang, H. Gu, Z. Qin, L. Fan, Q. Yang, and K. Ren, "FedTracker: Furnishing Ownership Verification and Traceability for Federated Learning Model," *IEEE Transactions on Dependable and Secure Computing*, vol. 22, no. 1, pp. 114-131, 2025.
[28] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas, "Communication-Efficient Learning of Deep Networks from Decentralized Data," in Proc. of 20th International Conference on Artificial Intelligence and Statistics, 2017, pp. 1273-1282.
[29] Y. Liu, S. Chang, Y. Liu, B. Li, and C. Wang, "FairFed: Improving Fairness and Efficiency of Contribution Evaluation in Federated Learning via Cooperative Shapley Value," in Proc. of IEEE INFOCOM 2024, 2024, pp. 621-630.
[30] J. H. Cheon, A. Kim, M. Kim, and Y. Song, "Homomorphic Encryption for Arithmetic of Approximate Numbers," in Proc. of ASIACRYPT, 2017, pp. 409-437.
[31] D. Demmler, T. Schneider, and M. Zohner, "ABY - A framework for efficient mixed-protocol secure two-party computation," in Proc. of The Network and Distributed System Security Symposium 2015, 2015, pp. 1-15.
[32] B. Li and D. Micciancio, "On the security of homomorphic encryption on approximate numbers," in Proc. of EUROCRYPT 2021, 2021, pp. 648-677.
[33] Sheng. G, Wen. T, Guo. Q, and Yin. Y, "Privacy Preserving Inner Product of Vectors in Cloud Computing," *International Journal of Distributed Sensor Networks*, vol. 10, no. 5, pp. 1-6, 2014.
[34] A. Ghorbani and J. Zou, "Data Shapley: Equitable Valuation of Data for Machine Learning," in Proc. of the 36th International Conference on Machine Learning, 2019, pp.2242-2251.
[35] Y. Zheng, S. Wang, and C. -H. Chang, "A DNN Fingerprint for Non-Repudiable Model Ownership Identification and Piracy Detection," textitIEEE Transactions on Information Forensics and Security, vol. 17, pp. 2977-2989, 2022.



Jingwei Liu (Member, IEEE) received the B.S. degree majoring in applied mathematics, and the M.S. and Ph.D. degrees majoring in communication and information systems from Xidian University, Xi'an, China, in 2001, 2004, and 2007, respectively. He is currently a Professor with the School of Telecommunications Engineering, Xidian University. He has published more than 70 papers in journals and conference proceedings and authored or coauthored two books. His research interests include big data security and privacy preservation, cloud and edge computing, and cryptography. He is a member of the Chinese Association for Cryptologic Research.

Zihan Zhou (Student Member, IEEE) received the B.S. degree in information engineering from Xidian University, Xi'an, China, in 2022. He is currently pursuing the master's degree with the school of Communication Engineering, Xidian University, Xi'an, China. His research interests include federated learning, homomorphic encryption, and privacy protection.



Rong Sun (Member, IEEE) received the B.E. degree in telecommunications engineering, the M.E. degree in communications and information systems, and the Ph.D. degree in communications and information systems from Xidian University, Xi'an, China, in 1998, 2001, and 2008, respectively. She is currently an Associate Professor with the School of Telecommunications Engineering, Xidian University. Her research interests include wireless communications, channel coding design, and information theory. She is a member of IEICE.

Lei Liu (Member, IEEE) received the B.Eng. degree in communication engineering from Zhengzhou University, Zhengzhou, China, in 2010, and the M.Sc. and Ph.D. degrees in communication engineering from Xidian University, Xi'an, China, in 2013 and 2019, respectively. From 2013 to 2015, he was employed a subsidiary of China Electronics Corporation, Beijing, China. From 2018 to 2019, he was supported by the China Scholarship Council to be a visiting Ph.D. student with the University of Oslo, Oslo, Norway. He is currently an Associate Professor with the Guangzhou Institute of Technology, Xidian University. His research interests include vehicular ad hoc networks, intelligent transportation, mobile-edge computing, and Internet of Things.

Rongxing Lu (Fellow, IEEE) received the Ph.D. degree from the Department of Electrical and Computer Engineering, University of Waterloo, Canada, in 2012.

He was a Post-Doctoral Fellow with the University of Waterloo from May 2012 to April 2013. He is currently the Acting Director of Canadian Institute for Cybersecurity (CIC), a Mastercard IoT Research Chair, and a full professor with the School of Computing, Queen's University, Canada. Before that, he was an Assistant Professor with the School of Electrical and Electronic Engineering, Nanyang Technological University (NTU), Singapore, from April 2013 to August 2016. His research interests include applied cryptography, privacy enhancing technologies, the IoT-big data security, and privacy. He serves as the Chair for the IEEE Communications and Information Security Technical Committee (ComSoc CISTC) and the Founding Co-Chair for the IEEE TEMS Blockchain and Distributed Ledgers Technologies Technical Committee (BDLT-TC).

Schahram Dustdar (Fellow, IEEE) is currently a full professor of computer science (informatics) with a focus on Internet Technologies heading the Distributed Systems Group at the TU Wien. He is the chairman of the Informatics Section of the Academia Europaea (since December 9, 2016). He is elevated to IEEE fellow (since January 2016). From 2004-2010 he was the honorary professor of Information Systems at the Department of Computing Science, University of Groningen (RuG), The Netherlands. From December 2016 until January 2017, he was a visiting professor with the University of Sevilla, Spain, and from January until June 2017, he was a visiting professor at UC Berkeley, Berkeley, California. He is a member of the IEEE Conference Activities Committee (CAC) (since 2016), of the Section Committee of Informatics of the Academia Europaea (since 2015), a member of the Academia Europaea: The Academy of Europe, Informatics Section (since 2013). He is the recipient of the ACM Distinguished Scientist Award (2009) and the IBM Faculty Award (2012). He is an associate editor of the IEEE Transactions on Services Computing, ACM Transactions on the Web, and ACM Transactions on Internet Technology and on the editorial board of the IEEE Internet Computing. He is the editor-in-chief of the Computing (an SCI-ranked journal of Springer).

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