




Please cite the Published Version

Zhu, Jiang, Wu, Jun , Bashir, Ali Kashif , Pan, Qianqian  and Yang, Wu (2023) Privacy-Preserving Federated Learning of Remote Sensing Image Classification With Dishonest Majority. IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing, 16. pp. 4685-4698. ISSN 1939-1404

DOI: <https://doi.org/10.1109/JSTARS.2023.3276781>

Publisher: Institute of Electrical and Electronics Engineers (IEEE)

Version: Published Version

Downloaded from: <https://e-space.mmu.ac.uk/634333/>



Usage rights:  [Creative Commons: Attribution 4.0](https://creativecommons.org/licenses/by/4.0/)

Additional Information: This is an open access article which first appeared in IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing

Enquiries:

If you have questions about this document, contact openresearch@mmu.ac.uk. Please include the URL of the record in e-space. If you believe that your, or a third party's rights have been compromised through this document please see our Take Down policy (available from <https://www.mmu.ac.uk/library/using-the-library/policies-and-guidelines>)

Privacy-Preserving Federated Learning of Remote Sensing Image Classification With Dishonest Majority

Jiang Zhu, Jun Wu , Senior Member, IEEE, Ali Kashif Bashir , Senior Member, IEEE, Qianqian Pan , Member, IEEE, and Wu Yang

Abstract—The classification of remote sensing images can give valuable data for various practical applications for smart cities, including urban planning, construction, and water resource management. The federated learning (FL) solution is often adopted to resolve the problems of limited resources and the confidentiality of data in remote sensing image classification. Privacy-preserving federated learning (PPFL) is a state-of-art FL scheme tailored for the privacy-constrained situation. It is required to address safeguarding data privacy and optimizing model accuracy effectively. However, existing PPFL methods usually suffer from model poisoning attacks, especially in the case of dishonest-majority scenarios. To address this challenge, in this work, we propose a blockchain-empowered PPFL for remote sensing image classification framework with the poisonous dishonest majority, which is able to defend against encrypted model poisoning attacks without compromising users' privacy. Specifically, we first propose the method of proof of accuracy (PoA) aiming to evaluate the encrypted models in an authentic way. Then, we design the secure aggregation framework using PoA, which can achieve robustness in a majority proportion of adversary settings. The experimental results show that our scheme can reach 92.5%, 90.61%, 87.48%, and 81.84% accuracy when the attacker accounts for 20%, 40%, 60%, and 80%, respectively. This is consistent with the FedAvg accuracy when only benign clients own the corresponding proportion of data. The experiment results demonstrate the proposed scheme's superiority in defending against model poisoning attacks.

Index Terms—Artificial intelligence, privacy, remote sensing, security.

Manuscript received 2 January 2023; revised 7 March 2023 and 11 April 2023; accepted 1 May 2023. Date of publication 16 May 2023; date of current version 29 May 2023. This work was supported in part by the JSPS KAKENHI under Grant 23K11072, in part by the National Natural Science Foundation of China under Grant U21B2019, Grant 61972255, Grant 61831007, and Grant U2003206, in part by the Program of Shanghai Academic Young Research Leader under Grant 20XD1422000, and in part by the National Social Science Foundation Major Project under Grant 20&ZD140. (Corresponding author: Jun Wu.)

Jiang Zhu and Jun Wu are with the Graduate School of Information, Production and System, Waseda University, Kitakyushu 808-0135, Japan (e-mail: jiangzhu@suou.waseda.jp; jun.wu@ieee.org).

Ali Kashif Bashir is with the Woxsen School of Business, Woxsen University, Hyderabad 502345, India (e-mail: dr.alikashif.b@ieee.org).

Qianqian Pan is with the Department of Systems Innovation, School of Engineering, The University of Tokyo, Tokyo 113-8654, Japan (e-mail: pan-qianqian@sjtu.edu.cn).

Wu Yang is with the Research Center of Information Security, Harbin Engineering University, Harbin 150009, China (e-mail: yangwu@hrbeu.edu.cn).

Digital Object Identifier 10.1109/JSTARS.2023.3276781

NOMENCLATURE

$\mathbf{w}_i^{(t)}$	Local model.
$\mathbf{w}^{(t)}$	Global model.
$p_i^{(t)}$	Precision rate.
$l_i^{(t)}$	Loss.
$\mu_i^{(t)}$	Trust score.
$\mathcal{C}_i^{(t)}$	Satellite client identity.
\mathcal{D}	Data resource.
σ	Digital signature.
ctr	TEE index.
$[\mathbf{w}_i^{(t)}]$	Encrypted local model.
$[\mathbf{w}^{(t)}]$	Encrypted global model.
n	Number of prediction classes.
m	Number of clients.
sk_t	TEE private key.
vk_t	TEE public key.
pk	Private key.
sk	Public key.
prog	Program in TEE.

I. INTRODUCTION

REMOTE sensing is the process of detecting and monitoring the physical features of a location by measuring its emitted and reflected radiation from a distance, often via a satellite or airplane [1], [2]. Remote sensing applications have diverse domains, with the smart city domain being one of the most prominent ones. This domain encompasses a range of applications, such as monitoring natural catastrophes, weather forecasting, and remote surveillance [3], [4], [5], [6]. A smart city is a modern urban concept that integrates humans, the physical world, and digital technologies. This can improve the quality of life and make our cities more inventive and environmentally friendly [7]. Image classification in remote sensing is a real-world practical application example. It is feasible to apply advanced smart city processes and remote operations for tasks, such as detecting objects, analyzing images, and geographic mapping [8]. However, the variability and complexity of the remote sensing data create a difficult challenge. On the one hand, sensing data is usually scattered across numerous entities due to the increase in data collection from multiple sources [5]. On the other hand, remote sensing data is privacy-sensitive information that may reveal sensitive infrastructure and military facilities [9]. As a result,

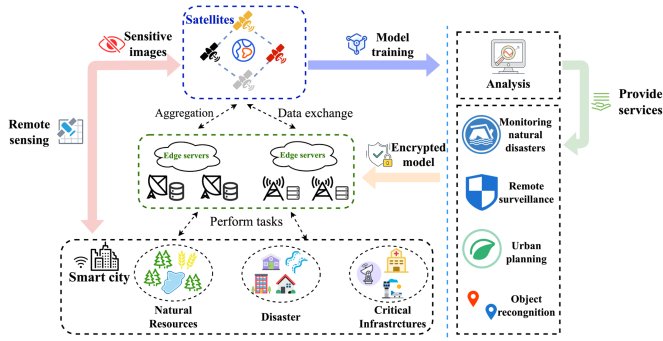


Fig. 1. Remote sensing image classification via PPFL in smart cities.

access to such data is limited to authorized personnel to prevent unauthorized use or disclosure of sensitive information. Hence, from a practical and privacy-concerning perspective, an adaptive paradigm, such as federated learning (FL) [5] is frequently used to address these issues.

FL is an emerging approach in collaborative machine learning, where training is conducted in parallel and distributed among multiple clients. In FL, the training process is carried out on individual devices or nodes, which collect and analyze data locally before sharing only relevant updates with the central server [10]. Nevertheless, traditional FL does not protect the privacy of locally uploaded model parameters, which is also proven to be detrimental to privacy [11], [12]. To alleviate security issues, PPFL has been the topic of extensive academic study. One typical PPFL architecture via homomorphic encryption [13] used in remote sensing image classification is shown in Fig. 1. Satellites capture images in space that carry sensitive data. The satellite then uses these images to train a local model for a specific purpose, encrypts it using a homomorphic encryption algorithm, and delivers it to the edge server. Edge servers collect these encrypted local models, aggregate them, and transfer the global model to satellites for decryption. Finally, the edge servers perform specific actions using the decrypted global model.

Existing PPFL schemes attempted to solve privacy issues, but other problematic issues still exist. Model poisoning attack is the foremost concern [14], [15]. In model poisoning attack, the attackers produce a poisonous local model, which is then uploaded and aggregated by the server to create a poisoned global model, which may make incorrect predictions. Because PPFL frequently encrypts the data before it is delivered to the server, it is more difficult to distinguish between parameters that are harmful and those that are beneficial. To tackle this problem, researchers have proposed a number of secure aggregation solutions to reduce the risk of harmful actions as a defense mechanism against model poisoning attacks. Although these rules can remove malicious parameters through various approaches, they have some restrictions or flaws and do not apply to real-life situations. For example, oblivious defender for private Byzantine-robust federated learning (FLOD) [16] is a novel Hamming distance-based aggregation method, which can resist model poisoning attacks when the Byzantine users are $\geq 50\%$. The approach involves the deployment of two

noncolluding task-specific servers to perform safe aggregation, which can be impractical and does not consider malicious server behavior. Truth discovery-based federated learning (TDFL) [17] is an efficient way to resist model poisoning in a dishonest-majority setting, the solution requires prerequisite knowledge of adversary percentage. Multishuffler secure federated learning (MSFL) framework [18] shuffles users and the local model to build trust between local trainers and aggregators. However, the MSFL involves a hierarchical shuffling process and, hence, cannot avoid the single point of failure issue. To sum up, although existing solutions can solve the problem to some degree, they all have some impractical limitations, such as a single point of failure problem, previous knowledge of adversary percentage, and the requirement of two noncolluding task-specific servers. Therefore, they do not resolve the problem or perfectly fit this situation.

To solve those issues, we propose a homomorphic encryption and blockchain empowered PPFL scheme for remote sensing image analysis. Our proposed scheme involves satellites that capture images and transmit them to edge servers, which act as agents to facilitate FL. Furthermore, we adopt trusted execution environment (TEE) [19] as a mean to generate a credible proof namely PoA for each local model. In addition, we use blockchain to record necessary data and aggregation processes to achieve transparency. The main contributions of our work are shown as follows.

- 1) Inspired by the concept of proof of work, we design a natural construction that provides direct evidence to prove the performance of a local model called PoA. Based on TEE, our strategy can resist model poisoning in a dishonest-majority setting.
- 2) We develop a blockchain-empowered PPFL framework to facilitate transparent processes and protect privacy. In order to guarantee the confidentiality of model parameters on the local data of clients, we encrypt local parameters using the CKKS cryptosystem [20]. The framework allows satellite imagery owners worldwide to jointly train a global image classification model, free from impractical limitations in existing schemes.
- 3) We provide comprehensive security analysis of the scheme and demonstrate extensive experiments using a well-known dataset. The experiments conducted show that the proposed scheme can effectively resist poisonous encrypted models and achieve robust aggregation.

The rest of this article is organized as follows. In Section II, we introduce the related literatures. Then, we overview the preliminaries in Section III. Next, we overview the problem formulation in this work. Section IV elaborates on the proposed scheme. Section V gives the theoretical security and privacy analysis. Section VI evaluates the performance of the proposed scheme. Finally, Section VII concludes this article.

II. RELATED WORK

In this section, we briefly review the state-of-the-art research works focusing on remote sensing image classification and defense against model poisoning attacks.

A. Remote Sensing Image Classification

Remote sensing image classification has attracted a lot of interest and made some impressive strides in recent years. In 2019, with support vector machines (SVM), Geiß et al. [21] followed the idea of learning invariant decision functions for remote sensing image classification. They set up a self-learning process to eliminate useless virtual samples from a process that could generate invariances in any way so that models can be made that are both reliable and sparse. In 2019, Wang et al. [22] investigated the efficiency of the random forest (RF) algorithm for classifying remote sensing images of coastal wetland habitats. Their results suggested that the RF algorithm is superior to the SVM and k-NN algorithms in terms of classification accuracy and suitability for coastal wetland categorization. In 2020, Li et al. [23] suggested an innovative MLRSSC-CNN-GNN scheme, which is capable of mining both the occurrences of visual elements in the scene and their spatial-topological connections. In 2021, Alhichri et al. [24] proposed a deep attention convolutional neural network (CNN) named EfficientNet-B3-Attn-2 in remote sensing to classify scenes. By introducing the attention mechanism, the model takes the original feature maps and creates a new map that is a weighted average of them. In 2021, Cheng et al. [25] developed a straightforward yet successful scheme named Siamese-prototype network to classify remote sensing scenes. Experimented with prototype self-calibration and intercalibration, the authors claimed that the model could improve the accuracies of subsequent inference. Using an evolutionary algorithm to code and search the deep learning network architecture, Ma et al. [26] proposed SceneNet in 2021. As a result, it is possible to extract scene information from remote sensing images in a hierarchical structure that is both flexible and powerful. For the high spatial resolution scene categorization, Xu et al. [27] developed a deep feature aggregation framework powered by graph convolutional network in 2022. In terms of overall accuracy, the authors claimed that the proposed scheme outperforms some state-of-the-art scene classification methods. In 2022, Chen et al. [28] proposed CNSPN, a new few-shot remote sensing scene classification method based on prototype networks, which uses the semantic information of the image class names to achieve better performance. While recent studies have made significant strides in improving the accuracy of remote sensing image classification methods, a more realistic and privacy-conscious approach demands the development of a robust paradigm to address security concerns.

B. Defense Against Model Poisoning Attacks

FL is susceptible to model poisoning attacks during the local learning phase, where malicious individuals intentionally manipulate the local model updates to cause the global model to misclassify a set of targeted inputs. Such attacks can significantly degrade the performance of the trained model and potentially compromise the privacy of the data. To mitigate the impact of these attacks, effective safeguards must be in place to ensure the integrity of the learning process. These techniques can help maintain the robustness and accuracy of the FL model in the presence of adversarial attacks.

FL with secure aggregation has been proposed to further protect the privacy of the datasets. Studies focusing on solutions against model poisoning attacks have been widely reported in the literature. Fang et al. [14] proposed a novel trimmed-means method with validation dataset, which identifies the local model by directly measuring the performance of the model. Yin et al. [29] proposed a scheme based on median statistics, which calculates the median of local gradients as the global update in FL. Euclidean distance represents the length of a line segment between two points in Euclidean space [30]. It is quite popular among researchers to evaluate the similarity between gradients in FL. Krum is a scheme proposed by Blanchard et al. [31] that chose the gradient with the closed Euclidean distance to its neighboring gradients as the global gradient. It is quite obvious that this scheme [31] will face the difficulty of reaching global convergence. Therefore, Mhamdi et al. [32] proposed a new scheme based on Krum [31], which aggregates the gradients chosen by Krum [31] for the global gradient. Shen et al. [33] adopted machine learning method to cluster local gradients and, therefore, to detect the anomaly in the local gradients. Holding the belief that malicious gradients almost always have variants that are similar to one another but different from benign ones, Fung et al. [34] proposed a scheme using cosine similarity to identify outliers in a non-IID setting. Liu et al. [35] proposed the privacy-enhanced federated learning against poisoning attacks, which adopted pearson correction coefficient as the identification method in their work. Ma et al. [36] further proposed ShieldFL, which uses cosine similarity to detect malicious gradients while keeping it encrypted. It allows the aggregation server to compute the similarity on encrypted gradients. We observe that the methods mentioned above [29], [31], [32], [33], [34], [35], [36] compute the similarity between uploaded parameters to assess if a model is malicious. By computing similarity to identify outliers, these approaches rely primarily on the fundamental principle that the majority of clients are harmless. Thus, when adversaries are in the majority, these schemes will fail inexorably.

To address this issue, researchers have proposed many solutions to achieve robust aggregation in a dishonest-majority setting. Dong et al. [16] proposed FLOD, a novel aggregation method, which relies on a small clean dataset to resist $\geq 1/2$ Byzantine users. Due to the dataset's lack of transparency, the suggested solution relies on an essential assumption regarding the dataset's pristine condition. As a result, the plan is sound from an intellectual perspective but fails to take into account its applicability in the real world. In addition, FLOD requires two noncolluding servers to perform secure two-party computation, which naturally introduces single point of failure problem. Similarly, Cao et al. [37] proposed a Byzantine-robust federated learning via trust bootstrapping. By training a benign model on the server using clean validation data, the server can, therefore, identify the malicious model update by measuring the cosine similarity of each uploaded parameters. Yet, the same problems of single point of failure and root data transparency still exist. Xu et al. [17] proposed a Byzantine robust FL method based on truth discovery called TDFL. The technique can function without an external server model or validation dataset. The only

drawback is that protections are only adequate if the proportion of poisoners is known in advance. Miao et al. [38] presented a Byzantine-robust solution via blockchain systems, which cannot only achieve robust aggregation but also address the single point of failure problem. However, it still did not solve the root data transparency problem. MSFL [18] is a framework proposed by Zhou et al. The framework shuffles the users as well as the local model to build mutual confidence between local trainers and aggregators. Nevertheless, the MSFL contains a hierarchical shuffling process and, therefore, cannot avoid a single point of failure problem. To eliminate those impractical limitations, we proposed a robust PPFL scheme that not only resists poisoning attacks with dishonest majority but also avoids limitations from existing schemes.

III. PRELIMINARIES

In this section, we briefly introduce some preliminaries of TEE and PPFL.

A. Trusted Execution Environment

A TEE is a code execution environment, in which the loaded code and data can be protected with the highest confidentiality and integrity [19], [39]. It means that the code and data inside TEE cannot be tampered by any unauthorized entities, including the computer owner itself. However, due to the various side-channel attacks [40], researchers [40], [41] have been building secure application on trusted hardware only leveraging the integrity. There is a clear delineation between the tasks that can be completed with complete transparency and those that must maintain a certain level of secrecy. In this research, we bypass many important and practical worries about side-channel attacks resulting from the execution of arbitrary code on platforms at the expense of compromising the confidentiality. TEE relies on the creation of a protected memory area called enclaves to build an isolated environment. Enclaves with minimal trust assumptions are referred to as transparent enclaves [40]. By only focusing on the integrity, a transparent enclave assumes that the host can inspect all the code and data in the isolated memory area. It is worth noting that the security of TEE platforms like SGX is predicated on the idea that the platform's attestation key would be kept a secret. Xing et al. [19] mentioned techniques to protect cryptographic code against side channels. Therefore, it is reasonable to suppose that the enclave's protection of the attestation key involves the implementation of side-channel protections. When Intel SGX-capable computers are provisioned, attestation data is signed and confirmed with ECDSA-signed collateral acquired from Intel. These documents are then cached in the data center's caching service for safekeeping [19]. Therefore, we do not elaborate on the attestation key management scheme in the following sections. We can presume that the hardware platform handles everything. The abstract functionality for transparent enclave execution \mathcal{F}_{TEE} is usually parameterized by a secure signing scheme $\{\text{Sign}_{\text{sk}_T}, \text{Verify}_{\text{vk}_T}\}$ with a keypair $(\text{sk}_T, \text{vk}_T)$. In order to execute programs in TEE, the client should first use "Install" command on the input prog. Then, it produces a digital signature $\sigma = \text{Sign}_{\text{sk}_T}(\text{Hash}(\text{prog}))$ that is returned to the

server. The generated digital signature will serve as a proof of the successful installation of the specific program prog. The client is then allowed us to run the command "Compute" taking some inputs inp and some fresh randomness rnd to generate the output outp. It is worth noting that the program prog may accept further inputs in various rounds as indexed by a counter ctr and may be stateful. It may then output the corresponding data in accordance with the inputs received. Simply put, we make the assumption that the functionality is capable of computation and can generate signed results. In each round ctr, TEE outputs a state tuple $\{\text{inp}, \text{oup}, \text{ctr}, \sigma_{\text{ctr}}\}$ with $\sigma_{\text{ctr}} = \text{Sign}_{\text{sk}_T}(\text{ctr}, \text{inp}, \text{oup})$. Therefore, by verifying the digital signature every round, the server is convinced that oup is produced correctly by prog without worrying that the client will sabotage the code and data.

In order to implement a realistic instantiation, our scheme makes use of the popular intel SGX [19] as the underlying TEE service.

B. Privacy-Preserving Federated Learning

PPFL is a paradigm that has been widely investigated in recent years. It allows users to train their models locally and upload the local parameters directly to the semihonest server for aggregation without revealing anything private to the third party. Users receive the global model from the server and utilize it to update the global model. As literature suggests [42], it is obviously not safe to share plaintext local models with server. Curious servers can easily recover lots of information from the models. Many approaches have been proposed to ensure that the uploaded local models do not leak anything private. Existing methods can be divided into four varieties, including the techniques of *differential privacy*, *secure multiparty computation*, *secure aggregation*, and *homomorphic encryption*. In this work, we mainly focus on fully homomorphic encryption enabled PPFL due to its properties of strong privacy and no accuracy loss.

Typically, fully homomorphic encryption scheme, such as CKKS [20] consists tuple of algorithms (KeyGenerate, Encrypt, Decrypt) as follows.

- 1) $\text{KeyGenerate}(k) \rightarrow (\mathcal{PK}, \mathcal{SK})$. Upon receiving security parameter k , the KeyGenerate function generates a key pair $(\mathcal{PK}, \mathcal{SK})$.
- 2) $\text{Encrypt}(\text{pk}, x) \rightarrow c$. Encrypt is a homomorphic encryption function which takes public key pk and plaintext s as inputs and returns ciphertext c .
- 3) $\text{Decrypt}(\text{sk}, c) \rightarrow x$. Decrypt takes the ciphertext and private key sk as inputs and outputs the corresponding to c .

The key advantage of the homomorphic encryption scheme is that it enables users to perform computations on encrypted data without the need for decryption. Therefore, in PPFL, clients can upload encrypted parameters for aggregation so that they do not need to worry about privacy leakage. In our research, we investigate aggregation rules where users send local models to the master device instead of gradients. We generalize the procedures of PPFL in Algorithm 1.

Algorithm 1: General Procedures of PPFL.

Input: The security parameter λ , clients set \mathcal{C} participating the training process, data resources of clients $\{\mathcal{D}_i | i \in \{1, 2, \dots, m\}\}$, number of total training rounds \mathcal{T} .

Output: The final well-trained global model.

1: **Initialization:**

2: a). The key center generates a key pair by $\{\mathcal{PK}, \mathcal{SK}\} = \text{KeyGenerate}(\lambda)$;

3: b). The key center establishes a secure communication channel and passes the keypair to the aggregation server and each client;

4: c). The server initializes the initial model parameter \mathbf{w}^0 ;

5: e). Initialize the training iteration index by $t = 1$.

6: **Procedure:**

7: **for** $t \leq \mathcal{T}$ **do**

8: **(I). For clients:**

9: **for** $\forall i \in \{1, 2, \dots, m\}$ **do**

10: $C_i^{(t)}$ computes the t -th round local model parameters $\mathbf{w}_i^{(t)}$ with Stochastic Gradient Descent (SGD).

11: $[\mathbf{w}_i^{(t)}] = \text{Encrypt}(\mathbf{w}_i^{(t)}, \mathcal{PK})$;

12: $C_i^{(t)}$ uploads the encrypted model parameters $[\mathbf{w}_i^{(t)}]$ to the server;

13: **end for**

14: **(II). For server:**

15: **for** $\forall t \in \mathcal{T}$ **do**

16: **for** $\forall i \in \{1, 2, \dots, m\}$ **do**

17: $[\mathbf{w}^{(t)}] = \text{ParaAgg}([\mathbf{w}_1^{(t)}], \dots, [\mathbf{w}_i^{(t)}])$;

18: **end for**

19: **end for**

20: The server distributes the aggregated ciphertext $[\mathbf{w}^{(t)}]$ to all C_i ;

21: **(III). For clients:**

22: **for** $\forall t \in \mathcal{T}$ **do**

23: **for** $\forall i \in \{1, 2, \dots, m\}$ **do**

24: $\mathbf{w}^{(t)} = \text{Decrypt}([\mathbf{w}^{(t)}], \mathcal{SK})$;

25: **end for**

26: C_i updates its local model using the distributed parameters $\mathbf{w}^{(t)}$

27: **end for**

28: **end for**

29: **return** The global model with parameters \mathbf{w}^T .

IV. PROBLEM FORMULATION

In this section, we present the system model, threat model, and design goal respectfully.

A. System Model

The system model considered in this article is shown in Fig. 2. It mainly consists of three types of entities, namely the trusted authority (TA), satellites, edge servers, and blockchain system.

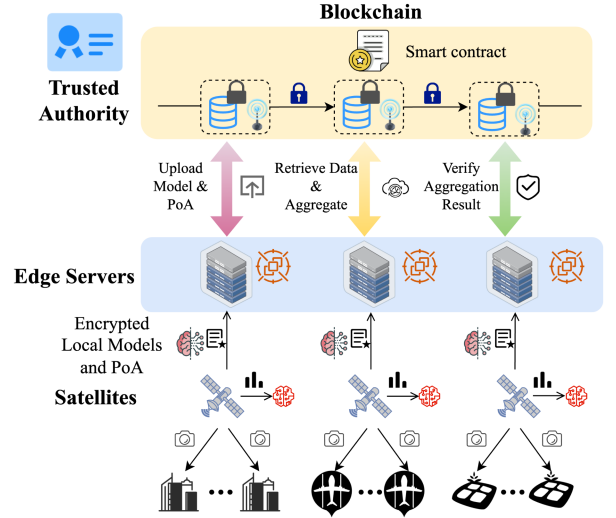


Fig. 2. System model.

- 1) *TA*: TA is responsible for producing the public and private keys (pk, sk) necessary for encryption and decryption based on the homomorphic encryption, and then distributes them to the edge servers so that safe communications may be achieved.
- 2) *Satellites*: While capturing images for remote sensing, each satellite also trains a local image processing model using the image data it collects. During each iteration of FL, satellites generate a PoA, which contains the encrypt the parameters of their local models and send the PoA to the edge servers that correspond to those satellites. In addition, the satellites will select some relatively insensitive data and send it to the edge server. The satellites will be able to utilize the revised model once it has been sent to them by the edge servers once they have updated the global model.
- 3) *Edge servers*: Edge servers, in their capacity as the agents of each satellite cluster, are tasked with the responsibility of publishing to the blockchain the local PoAs as well as the clean auxiliary data. Therefore, edge servers also act as blockchain miners to maintain the network. They are also responsible for PoA verification, aggregating, and updating the latest global model parameters on the blockchain, and then distributing these parameters to the satellites, which is an essential responsibility they take on.
- 4) *Blockchain system*: Blockchain is responsible for recording the collected PoA from the edge servers. Following this, the aggregation process is carried out by edge servers in order to produce a new global model update block based on the verified updates recorded in the blockchain concerning volunteer bias. This new global model is then sent back to the satellites by means of the edge servers.

B. Threat Model

In our proposed system, we assume a fully trusted key center and honest-but-curious edge servers. The edge servers follow the

FL protocol honestly, but are curious about the local datasets of the satellites in order to infer them from the parameters. In our system, we consider two types of satellite clients, i.e., benign clients and malicious clients. Inspired by previous work [36], we define *benign satellite clients* and *malicious satellite clients* as follows.

- 1) *Benign satellite clients*: A satellite client C_i is benign if and only if C_i honestly train and upload local model $w_i^{(t)}$ on its local dataset \mathcal{D}_i .
- 2) *Malicious satellite clients*: A satellite client C_i^* is an adversary such that C_i^* lunches a model poisoning attack by uploading poisonous model $[w_i^{(t)*}]$ to damage to global model. It is worth noticing that in our scheme, we do not limit the total percentage of malicious clients, which means that we can accept arbitrary proportion of malicious clients to achieve better robustness.

As for the adversarial clients who launched model poisoning attacks, we adopt the following settings.

- 1) *Adversarial goals*: \mathcal{A} compromises the integrity of the local training procedure in order to reduce the accuracy of the global model. In order to do this, the parameters of the local model might be deceitfully manipulated.
- 2) *Adversarial capability and knowledge*: \mathcal{A} controls arbitrary malicious clients C^* . C^* has fully control of local dataset, local model w_i , and the training process.

C. Design Goal

Under the abovementioned threat model, our scheme is designed to achieve three design goals.

Goal 1 (Privacy): The confidentiality of our defense scheme must to be ensured. Specifically, it is imperative that no confidential information be shared with any third parties.

Goal 2 (Transparency): Every activity should be recorded on the blockchain to facilitate transparency and stop dishonest clients from denying them.

Goal 3 (Robustness): The proposed scheme should defend against model poisoning attacks from malicious clients. Each client should provide proof to show the correct evaluation result of the local model, which is the direct delivery of the performance. The proof should also satisfy the property of authentication, which ensures that a good process generates the proof. In our scheme, we provide authentication through TEE. Our scheme should resist the model poisoning attacks under homomorphic encryption-based PPFL setting with an arbitrary size of malicious users C^* .

V. PROPOSED SCHEME

In this section, we first introduce the overall workflow to provide comprehensive understanding of our approach. Then, we elaborate on the technical details, including construction of PoA as well as homomorphic encryption and blockchain empowered PPFL framework.

A. Construction of PoA

We propose PoA, a model assessment strategy to evaluate the model based on test accuracy rate and loss function in order to detect model poisoning attacks. Following the completion of the evaluation, a score is assigned to each model, and this score will become an essential component of the aggregation process. However, since each client is responsible for the evaluation, it is a major challenge to prevent the client from making up the result of the evaluation. In order to resolve this issue, it is necessary to use a certain technique that may bind the encrypted model and the evaluation result together. In this article, we solved this problem by putting both the evaluation and the encryption tasks into TEE. After each epoch, each client must put the plain model and the validation data sent from the server inside TEE and upload the following output to the server. The program implementing all these operations within TEE is distributed by the server. After receiving the program prog, each client needs to run the command “Install” and return $\sigma = \text{Sign}_{\text{sk}_i}(\text{Hash}(\text{prog}))$ to the server as a token. By verifying the digital signature, the server is convinced that prog is correctly installed in TEE.

To begin, to complete the assessment process, it is necessary to obtain from the edge server the validation data, which in this instance are the test samples, along with the corresponding digital signature. Prog verifies the integrity of the samples and abort when fails. The samples are then fed to the model to make predictions and calculate the loss function. For the i th model of round t , we use the validation data to derive the accuracy rate and the loss from it, respectfully denoted as $p_i^{(t)}$ and $l_i^{(t)}$. The prediction result and the loss are also considered privacy of each client, therefore, we generalize these two factors using (1) and (2)

$$\mathcal{S}_{p_i}^{(t)} = \log_n \left(\max \left\{ \left(p_i^{(t)} - \frac{1}{n} \right), 0 \right\} \cdot n + 1 \right) \quad (1)$$

$$\mathcal{S}_{l_i}^{(t)} = \frac{2 \cdot e^{-l_i^{(t)}}}{1 + e^{-l_i^{(t)}}} \quad (2)$$

The trust score $\mu_i^{(t)}$, which is the direct evaluation result of the model is then produced using the following equation:

$$\mu_i^{(t)} = (\mathcal{S}_{p_i}^{(t)} + \mathcal{S}_{l_i}^{(t)}) \cdot \mathcal{S}_{p_i}^{(t)} \cdot \mathcal{S}_{l_i}^{(t)} \quad (3)$$

The abovementioned functions are precompiled into prog to compute a trust score $\mu_i^{(t)}$. In the meanwhile, we need to encrypt the model using homomorphic encryption algorithm. Utilizing the CKKS cryptosystem [20], which supports a particular kind of fixed-point arithmetic typically referred to as block floating-point arithmetic, our framework, therefore, is able to perform safe PPFL across the edge servers. The encrypted model and the trust score are packed together as the whole output of prog. After the completion of prog, it will generate a digital signature of inputs, outputs, and the index ctr to guarantee the integrity of the whole computation process.

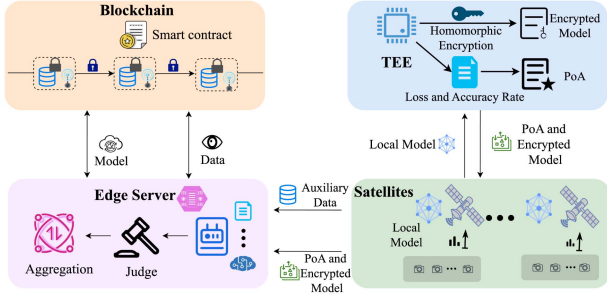


Fig. 3. Work flow of homomorphic encryption and blockchain empowered PPFL framework.

B. Homomorphic Encryption and Blockchain Empowered PPFL Framework

In this article, we develop a blockchain-based PPFL framework for remote sensing image processing to facilitate cooperative global model update aggregation and verification. In this architecture, edge servers establish a decentralized network in place of the central server, typically in standard FL designs. We adopt the blockchain as the underlying infrastructure since it is more resistant to attacks due to its immutable and auditable properties. Chain-based PoAs and auxiliary data may be publicly validated, minimizing poison attacks' effects. Furthermore, model parameters information is always kept in ciphertext on the chain, guaranteeing that it is never accessible to unauthorized or untrusted devices and avoiding single point of failure problem. The proposed architecture holds potential for significantly enhancing the security and resiliency of the FL process. In this framework, each cluster of satellites is assigned an edge server to act as an agent, which is consistent with real-life scenarios. The proposed scheme consists of four processes, system initialization, framework setup, local computation, and global model aggregation. The proposed homomorphic encryption and blockchain empowered PPFL framework is shown in Fig. 3. Here, we break down each step of the procedures in depth.

1) *System Initialization*: First and foremost, TA implements key generation and distribute them to the satellites in the initialization process. Each satellite gets a keypair (pk, sk) for digital signature and homomorphic encryption. It is worth noticing that the asymmetric key pair (pk_T, sk_T) used in TEE to generate digital signature is obtained from TEE service provider when the capable systems are provisioned [19]. Therefore, it is not the KGC's responsibility to distribute the TEE asymmetric key.

2) *Framework Setup*: As shown in Algorithm 2, the whole process of framework setup consists of three stages, and we will introduce them one by one.

a) *Validation data preparation*: Before we can get started with the actual training, there is some groundwork that needs to be laid out to ensure that our plan will go off without a hitch. Since our method requires a dataset that is both small and clean in order to generate a PoA, it is vital to maintain track of the dataset and make sure that it has not been tainted in any way. We achieve this by recording the auxiliary data on blockchain.

Algorithm 2: Framework Setup.

```

1: for  $i \in \{1, 2, \dots, m\}$  do
2:   Validation data preparation
3:    $C_i$  selects less sensitive images  $d_i$  from its own data resource  $\mathcal{D}_i$  where  $d_i \in \mathcal{D}_i$ ;
4:    $C_i$  use  $sk$  to sign the data  $d_i$  and generates a digital signature  $\sigma_{d_i}$ ;
5:    $C_i$  sends tuple  $(\sigma_{d_i}, d_i)$  to the corresponding edge server which will publish the tuple to the blockchain;
6:   The edge server maintains a clean validation dataset  $\mathcal{D}_v$  which is  $\sum_{i=1}^m (\sigma_{d_i}, d_i)$ ;
7:   TEE program installation
8:    $C_i$  compiles essential program prog according to the private key  $pk$ ;
9:    $C_i$  invokes  $\mathcal{F}_{TEE}(\text{"Install"}, \text{prog})$  and receives  $\sigma = \text{Sign}_{sk_T}(\text{Hash}(\text{prog}))$ ;
10:   $C_i$  sends tuple  $(\sigma, \text{prog})$  to the miner which will publish the tuple to the blockchain;
11:  Global model initialization
12:   $C_i$  initializes and encrypts the random global model  $[\mathbf{w}^0]$ ;
13:   $C_i$  sends  $[\mathbf{w}^0]$  to the interrelated server;
14:  if there exists a  $[\mathbf{w}^0]$  on blockchain then
15:    The edge server returns the formerly published  $[\mathbf{w}^0]$ ;
16:  else
17:    The edge server uploads  $[\mathbf{w}^0]$  and return it back to the satellites;
18:  end if
19: end for

```

Every satellite will select specific images that are less sensitive and sends them to the edge server to store those records in the blockchain network voluntarily.

b) *TEE program installation*: The satellites need to compile the corresponding prog based on pk. Then, each client invokes \mathcal{F}_{TEE} to install the program and stores prog along with the token on the blockchain network, which proves the validation of the correctness of the TEE procedure.

c) *Global model initialization*: Furthermore, the global model's random initialization $[\mathbf{w}^0]$ is generated by the satellite clients and uploaded to the blockchain to ensure that it is accessible and, therefore, verifiable to everyone. If multiple records are uploaded, the clients will select the earliest uploaded one.

3) *Local Computation*: In Algorithm. 3, the local training and the PoA generation steps are performed iteratively until it reaches global convergence.

a) *Local training*: The t th training iteration entails each edge server retrieving the latest global model from the blockchain and passing it to the satellite clients. Then, each satellite $C_i^{(t)}$ locally trains the individual model $\mathbf{w}_i^{(t)}$ on their own data resource.

b) *PoA generation*: After the completion of the local training, the edge servers collect data from the blockchain network, organizes it into a validation dataset, and distributes it to the

Algorithm 3: Local Computation.

```

1: for  $t \leq \mathcal{T}$  do
2:   for  $i \in \{1, 2, \dots, m\}$  do
3:     Local training
4:     Edge server sends the latest global model  $[\mathbf{w}^{(t-1)}]$ 
       back to  $\mathcal{C}_i^{(t)}$ ;
5:      $\mathcal{C}_i^{(t)}$  decrypts  $[\mathbf{w}^{(t-1)}]$  and uses its data resource to
       compute the  $t$ -th round local model parameters
        $\mathbf{w}_i^{(t)}$ ;
6:     PoA generation
7:      $\mathcal{C}_i^{(t)}$  initiate a request to its responding edge server
       to retrieve the validation dataset  $\mathcal{D}_v$ ;
8:      $\mathcal{C}_i^{(t)}$  invokes  $\mathcal{F}_{\text{TEE}}$  on ("Compute", ctr,  $\mathcal{C}_i^{(t)}$ ,  $\mathbf{w}_i^{(t)}$ ),
       and receives PoA (ctr,  $[\mathbf{w}_i^{(t)}]$ ,  $\mu_i^{(t)}$ ,  $\sigma_{\text{ctr}}$ ) from TEE;
9:   end for
10: end for

```

Algorithm 4: Global Model Aggregation.

```

1: for  $t \leq \mathcal{T}$  do
2:   Edge servers receive PoAs from satellites and
       publish them to the blockchain network;
3:   if there are  $m$  PoA blocks on the blockchain then
4:     Aggregate PoAs
5:      $[\mathbf{w}^{(t)}] = \eta \cdot \frac{\mu_i^{(t)}}{\sum_{i \in [1, m]} \mu_i^{(t)}} \cdot [\mathbf{w}_i^{(t)}]$ ;
6:     Send  $[\mathbf{w}^{(t)}]$  to blockchain network;
7:   end if
8:   if there is a new global model on the blockchain then
9:     Fetch the encrypted global model  $[\mathbf{w}^{(t)}]$  and send
       it to each satellite  $\mathcal{C}_i^{(t)}$ ;
10:  end if
11: end for

```

satellites. The client invokes \mathcal{F}_{TEE} to run the previously installed prog to generate a PoA, which is later uploaded to blockchain network by edge server.

4) *Global Model Aggregation*: The detailed process of global model aggregation is described in Algorithm 4. In theory, all participating edge servers are permitted to update the global model. The intended server will upload the computation result to the blockchain network in order to urge them to do the computation correctly. After each edge server uploads its PoA to the blockchain network, one voluntary server will collect all blocks and aggregate the ciphertext as follows:

$$[\mathbf{w}^{(t)}] = \eta \cdot \frac{\mu_i^{(t)}}{\sum_{i \in [1, m]} \mu_i^{(t)}} \cdot [\mathbf{w}_i^{(t)}]. \quad (4)$$

VI. THEORETICAL ANALYSIS AND PROOF

In this section, we provide comprehensive analysis of the security property and briefly analyze the privacy of our proposed scheme.

A. Security Property

In the our setting, the secure properties of fully homomorphic encryption-based PPFL has been widely discussed in previous work already [42]. To prove our aggregation scheme achieve robustness and does not reveal any private information about local models, we only discuss the secure properties of PoA in this section.

1) *Correctness Analysis*: To ensure that the aggregation scheme can effectively identify malicious models in an arbitrary proportion of adversary setting, we need to make sure that PoA provides reasonable weights for individual models.

Theorem 1: There is an error term that occurs between the honest model and the dishonest ones as $\sum_{i \in \text{Adversary}} \mathbf{w}_i^T = \sum_{i \in \text{Benign}} \mathbf{w}_i + \Delta$.

Proof: We generalize the model poisoning attack approach such that we may construct a set of false model parameters that are distinct from the honest users' local model. Consider the following unbiased estimator Δ . $E[\mathbf{w}] = g$ as the difference between them. We define the actual vector received by the server as being when a malicious user is present to be

$$\tilde{\mathbf{w}}_i = \begin{cases} \mathbf{w}_i, & \text{for honest } i\text{th user} \\ \mathbf{w}_i^T, & \text{for dishonest } i\text{th user.} \end{cases} \quad (5)$$

In accordance with the SGD technique that was established and utilized in order to update the model parameters, we are able to get: $\mathbf{w}^{(t)} = \mathbf{w}^{(t-1)} - \eta \frac{\sum_{i \in [1, m]} \tilde{\mathbf{w}}_i}{m}$. Here, we define the model of the honest users as $\{\mathbf{w}_i, i \in H\}$, where H stands for the set of honest users. On the other hand, we define the models of the adversary as $\{\mathbf{w}_i^T, i \in A\}$, where A represents the malicious users. Considering δ as the difference between the dishonest model parameters and the benign target model, we have $\delta = \eta \frac{\sum_{i \in A} \mathbf{w}_i^T}{|A|} - \eta \frac{\sum_{i \in H} \mathbf{w}_i}{|H|}$. As a result, we have

$$\begin{aligned} \sum_{i \in A} \mathbf{w}_i^T &= \frac{|A|}{\eta} \delta + \frac{\sum_{i \in H} \mathbf{w}_i}{|H|} |A| \\ &= \frac{|A|}{\eta} \delta + \left(\frac{|A|}{|H|} - 1 \right) \sum_{i \in H} \mathbf{w}_i + \sum_{i \in H} \mathbf{w}_i \\ &= \frac{|A|}{\eta} \delta + \left(\frac{|A|}{|H|} - 1 \right) |H|g + \sum_{i \in H} \mathbf{w}_i. \end{aligned} \quad (6)$$

We have, therefore, proved that there is an error term between malicious and benign models as $\Delta = \frac{|B|}{\eta} \delta + \left(\frac{|A|}{|H|} - 1 \right) |H|g$.

Theorem 2: For an arbitrary proportion of malicious satellites, there will be a gradual convergence between the global model learnt by our scheme based on PoA and the optimal global model \mathbf{w}^* over T iterations.

Proof: It is obvious that benign models tend to have less loss and more precision, whereas poisonous models tend to have the opposite characteristics. As a consequence, the following tendencies are satisfied by both honest and malicious users:

Tendency 1: $\forall \mathcal{C}_i \in H, p_i^{(t)} \rightarrow 1$ and $l_i^{(t)} \rightarrow 0$

Tendency 2: $\forall \mathcal{C}_i \in A, p_i^{(t)} \rightarrow 0$ and $l_i^{(t)} \rightarrow \infty$

In accordance with (1) and (2), when it comes to perfect local model, we have limit equations as follows:

$$\begin{aligned} \lim_{p_i^{(t)} \rightarrow 1} \mathcal{S}_{p_i}^{(t)} &= \lim_{p_i^{(t)} \rightarrow 1} \log_n \left(\max \left\{ \left(p_i^{(t)} - \frac{1}{n} \right), 0 \right\} \cdot n + 1 \right) \\ &= \lim_{p_i^{(t)} \rightarrow 1} \log_n \left(1 - \frac{1}{n} \right) \cdot n + 1 \\ &= \lim_{p_i^{(t)} \rightarrow 1} \log_n n = 1 \end{aligned} \quad (7)$$

$$\begin{aligned} \lim_{l_i^{(t)} \rightarrow 0} \mathcal{S}_{l_i}^{(t)} &= \lim_{l_i^{(t)} \rightarrow 0} \frac{2 \cdot e^{-l_i^{(t)}}}{1 + e^{-l_i^{(t)}}} \\ &= \frac{2 \cdot 1}{1 + 1} = 1. \end{aligned} \quad (8)$$

On the other hand, poisonous models aim to decrease the accuracy of the federated model. As a result, the following is the equations at the limit of malicious local models:

$$\begin{aligned} \lim_{p_i^{(t)} \rightarrow 0} \mathcal{S}_{p_i}^{(t)} &= \lim_{p_i^{(t)} \rightarrow 0} \log_n \left(\max \left\{ \left(p_i^{(t)} - \frac{1}{n} \right), 0 \right\} \cdot n + 1 \right) \\ &= \lim_{p_i^{(t)} \rightarrow 0} \log_n 0 \cdot n + 1 \\ &= \lim_{p_i^{(t)} \rightarrow 0} \log_n 1 = 0 \end{aligned} \quad (9)$$

$$\begin{aligned} \lim_{l_i^{(t)} \rightarrow \infty} \mathcal{S}_{l_i}^{(t)} &= \lim_{l_i^{(t)} \rightarrow \infty} \frac{2 \cdot e^{-l_i^{(t)}}}{1 + e^{-l_i^{(t)}}} \\ &= \frac{2 \cdot 0}{1 + 0} = 0. \end{aligned} \quad (10)$$

The trust score $\mu_i^{(t)}$ is consequently greater for healthy models and less for hostile ones. According to Theorem 1, we can conclude that $\sum_{i \in A} \mathbf{w}_i^T = \sum_{i \in H} \mathbf{w}_i + \Delta$. In this respect, the proposed scheme is able to recognize anomalies based on differences and lower the weight of malicious local models. If the harmful objective has been accomplished, the value Δ will grow as the number of the iteration increases. However, with limitation equations from above, the weights of poisoners are close to 0. As a result, we have

$$\lim_{t \rightarrow \infty} \Delta = 0. \quad (11)$$

That is, the effect on honest users satisfies

$$\lim_{t \rightarrow \infty} \sum_{C_i \in H} \frac{\mu_i^{(t)}}{\sum_{C_i \in [1, m]} \mu_i^{(t)}} = 0. \quad (12)$$

2) *Privacy Analysis*: We provide provable guarantees on the privacy of PoA against honest-but-curious servers. PoA can protect the confidentiality of the users' models, the prediction results, and the associated loss.

For any honest-but-curious edge server, the PoA consists a trust score and encrypted local model parameters. The homomorphic encryption approach is expected to prevent any privacy of the local model from being revealed in the ciphertext of the

model parameters. In accordance with (3), the trust score is calculated from two secret variables $p_i^{(t)}$ and $l_i^{(t)}$. Therefore, it is implausible for the server to determine the loss and prediction outcome merely from the trust score.

As a result, the proposed scheme has the property of maintaining users' privacy by preventing any parties from compromising clients' privacy.

VII. EXPERIMENTAL ANALYSIS

In this section, we evaluate our scheme against both targeted and untargeted attacks and show the results of our experiments. All experiments are run in a high-performance server with the configuration of Ubuntu 20.04, Intel Xeon Gold 6226R 3.90 GHz CPU, and 256 GB RAM.

A. Experimental Setup

1) *Datasets and Settings*: The dataset used is the real-world dataset named EuroSat [43] to evaluate our scheme. Using Sentinel-2 satellite images, the difficulty of accurately classifying land use and land cover is one that EuroSAT [43] attempts to solve. The maps represent all of Europe's 34 nations and are divided into ten categories that correlate to the continent's various land uses. There are a total of 27 000 captioned images, with each category including between 2000 and 3000 pictures. The dimensions of the image are 64×64 pixels and they span an area that is 640×640 m. Included in this analysis are all thirteen of the Sentinel-2's spectral bands. In Fig. 4, selected representative samples are presented to give a more direct understanding of the dataset.

On a private blockchain infrastructure, we put our plan into action and conduct an evaluation of it. The private blockchain is hosted on an Intel processor that runs at 3.90 GHz and has 16 cores with 32 threads each core. In addition, we deploy CKKS using the Pyfhel library, which is a library that serves as an efficient Python API for the most powerful C++ HE libraries. Pyfhel is a library for homomorphic encryption that is free to use, and its source code can be found on GitHub (<https://github.com/ibarrond/Pyfhel>).

2) *Poisoning Attacks*: In this particular experiment, we consider both targeted and untargeted attacks. Our approach assumes that the malicious clients upload arbitrary model parameters as part of untargeted attacks in order to influence the global model. We adopt label-flipping attack in terms of targeted attacks. As part of our simulation of the label-flipping attack, we have relabeled the source class, which is now being possessed by malicious users, as the target class.

3) *Evaluation Metrics*: Since the accuracy of the global model is the primary focus of our scheme, we make use of test accuracy and test loss as indicators of the models. The FedAvg [44] approach is widely used in FL environments that do not include poisoners. Therefore, we use the FedAvg scheme with only benign clients owning the corresponding proportion of data as a baseline for our evaluation, and then we present the outcomes of the proposed scheme with varying percentages of malicious clients.



Fig. 4. Sample images of all ten classes in the dataset EuroSAT. (a) Annual crop. (b) Forest. (c) Herbaceous vegetation. (d) Highway. (e) Industrial. (f) Pasture. (g) Permanent Crop. (h) Residential. (i) River. (j) Sea lake.

TABLE I
MODEL SUMMARY

Type	Output shape	Parameters
Rescaling	(None, 64, 64, 3)	0
Conv2D	(None, 64, 64, 16)	448
MaxPooling2D	(None, 32, 32, 16)	0
Conv2D	(None, 32, 32, 32)	4640
MaxPooling2D	(None, 16, 16, 32)	0
Conv2D	(None, 16, 16, 64)	18496
MaxPooling2D	(None, 8, 8, 64)	0
Flatten	(None, 4096)	0
Dense	(None, 128)	524416
Dense	(None, 10)	1290

4) *FL Settings*: The cross-silo configuration is the one that we use for our evaluation. Throughout the process of the training, we will choose all clients in each iteration, with the number of clients being set at $n = 20$. For the purpose of model selection and training using the dataset using TensorFlow as the backend. The parameters of the model we adopted are shown in Table I. The table contains the name and type of all layers, the output shape, and the number of weight parameters for each layer. After then, we split the data in an equal manner across each client; for the EuroSat dataset, this means that each client has around 1350 data distributed equally. When it comes to the proportion of the malicious clients, we use a number of possible situations, ranging from 20% to 80%. We decided to use a batch size of 32 for this operation. In addition, we find that the loss function arrives at a stable value after 40 rounds; hence, we decide to make the total training epoch to 40.

B. Experimental Results

Experiments described here are carried out to demonstrate that our framework is capable of meeting the design goals of privacy, robustness, and transparency. We do each experiment several times in order to acquire trial findings and then calculate an average. The results of the experiments reveal that our scheme

is capable of capturing the desired aims, which is discussed in Section IV. It is worth noting that in the case of different proportions of attackers, the test accuracy and loss of the final model will be different. This is because the proportion of benign users decreases, resulting in fewer datasets for training. In order to correctly demonstrate the correctness of our scheme, we use the training results part excluding malicious users as a baseline for comparison. Next, we first verify the practicability of the scheme. In order to accomplish that, we evaluate the scheme from two different attack perspectives, namely targeted attacks and untargeted attacks. In addition, we analyze the additional time overhead of the scheme and conduct an experimental evaluation.

1) *Impact of Targeted Attacks*: We examine the accuracy loss under targeted attacks from malicious users. In targeted attacks, we flip the label of the training data for each malicious clients so that the trained models make a wrong judgment on the test data. The experiment demonstrates that the presented PPFL scheme is acceptable and provides the essential robustness required under targeted attack despite the proportion of adversaries. Fig. 5 shows the testing accuracy and loss under targeted attacks with honest majority setting. To demonstrate the superiority and novelty of our proposed scheme, we compare the accuracy and loss of our defense approach with the traditional FedAvg algorithm. By doing so, we aim to provide evidence of the effectiveness of our approach in enhancing the security and privacy of the data in the FL process. The figure clearly shows that the trained global model suffers from a loss of accuracy and precision when the attacker's percentage is at 20% and 40%. Therefore, when it comes to honest majority setting, the attacker can only affect the global model to a limited extent but cannot cause fatal damage. Even though there is a certain gap between the attacked model and the regular trained global model, it is still relatively correct. On the other hand, when it comes to the dishonest-majority setting, shown in Fig. 6, the global model will be damaged if malicious users' misleading data overwhelms the truths of honest users. It can be concluded

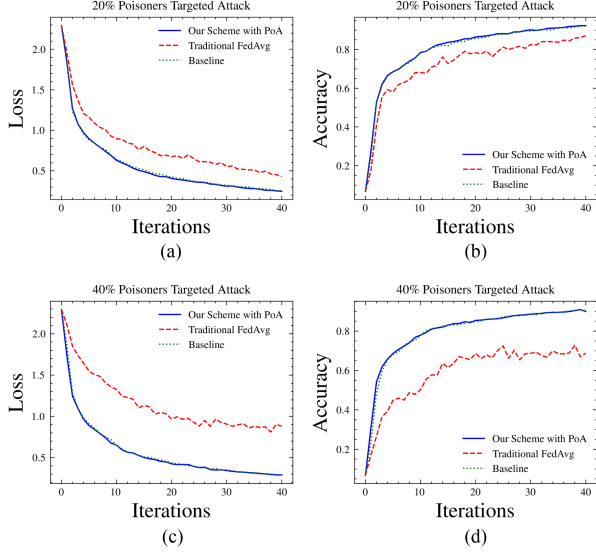


Fig. 5. Comparison between FedAvg and our scheme under honest majority setting in targeted attacks. (a) Loss curve, Att = 20%. (b) Accuracy curve, Att = 20%. (c) Loss curve, Att = 40%. (d) Accuracy curve, Att = 40%.

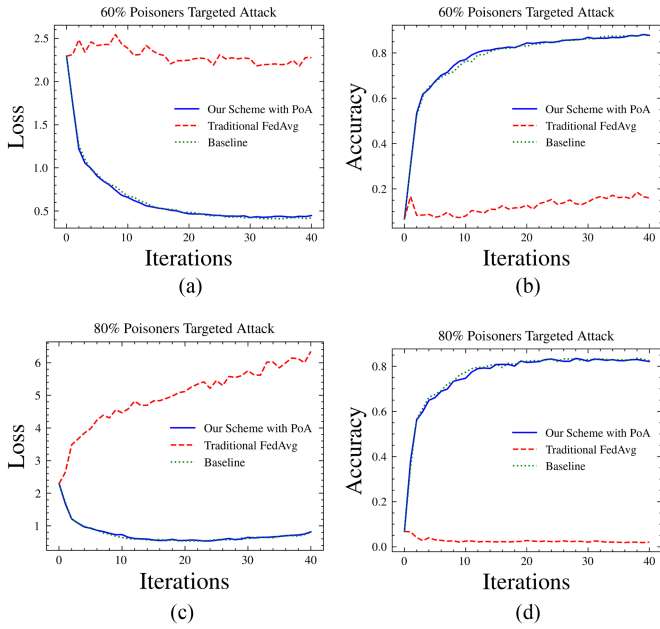


Fig. 6. Comparison between FedAvg and our scheme under dishonest-majority setting in targeted attacks. (a) Loss curve, Att = 60%. (b) Accuracy curve, Att = 60%. (c) Loss curve, Att = 80%. (d) Accuracy curve, Att = 80%.

from the figure that in the presence of an attacker, there is a significant gap in terms of both accuracy and loss between the global model without a defense mechanism and the normally trained model. The accuracy and loss curves of the model trained using our scheme in both settings roughly coincide with the baseline, which is the embodiment of the correctness of our scheme. Naturally, the final global models in dishonest-majority settings have slightly less accuracy and more loss due to the relative scarcity of training data. Furthermore, to better evaluate the model's performance when adopting the PoA, we calculate

TABLE II
F-SCORE IN TRAGETED ATTACKS

Attakcers percentage	F-score
20%	0.922
40%	0.902
60%	0.870
80%	0.810

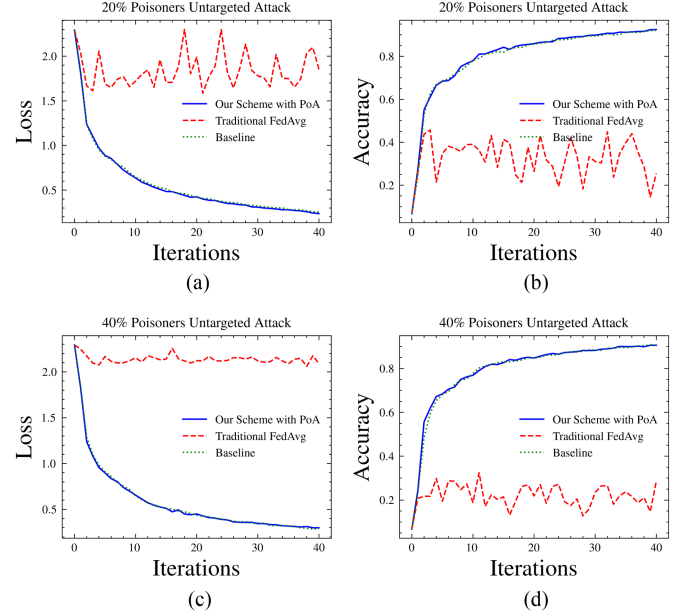


Fig. 7. Comparison between FedAvg and our scheme under honest majority setting in untargeted attacks. (a) Loss curve, Att = 20%. (b) Accuracy curve, Att = 20%. (c) Loss curve, Att = 40%. (d) Accuracy curve, Att = 40%.

the F-scores of the model under targeted attacks, which are listed in Table II. As we can see, under the protection of our scheme, the F-scores can still achieve the desired effect, which indicates that the model is performing well in terms of precision and recall.

2) *Impact of Untargeted Attacks:* We examine the accuracy loss under untargeted attacks from malicious users. Similarly, to demonstrate the superiority of our method, we also compare its accuracy and loss to the FedAvg algorithm under untargeted attacks. The results of both the honest majority setting and dishonest-majority setting experiments are shown in Figs. 7 and 8, respectively. In untargeted assaults, we make the assumption that the malicious clients try to sabotage the global model by uploading arbitrary local models. During the experiments, we make the observation that the arbitrary models generally have low accuracy, but do not have exaggeratedly large losses. Therefore, we adjust the aggregation weights by exploiting the two parameters from loss and accuracy according to (3). A lower value for either of both parameters will have a more significant effect on aggregation weights. As shown in Figs. 7 and 8, different from the previous scenario, we can observe from the experimental results that there exists larger gaps between unprotected PPFL and the baseline in both honest and dishonest settings. The attackers upload random parameters despite the

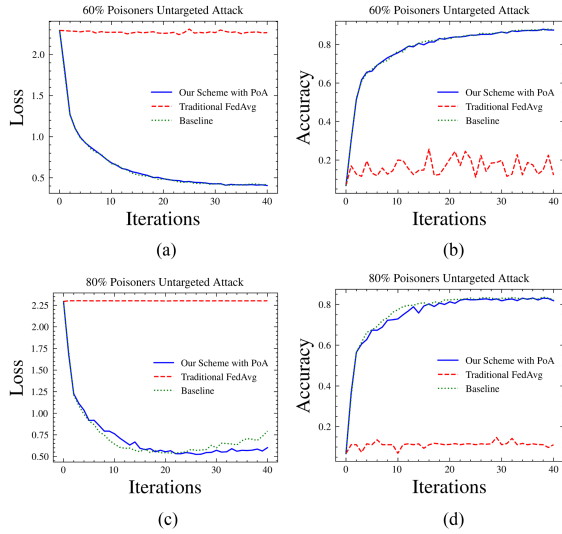


Fig. 8. Comparison between FedAvg and our scheme under dishonest-majority setting in untargeted attacks. (a) Loss curve, $Att = 60\%$. (b) Accuracy curve, $Att = 60\%$. (c) Loss curve, $Att = 80\%$. (d) Accuracy curve, $Att = 80\%$.

TABLE III
F-SCORE IN UNTARGETED ATTACKS

Attackers percentage	F-score
20%	0.920
40%	0.896
60%	0.873
80%	0.815

global model, making rectification from benign clients more difficult. Therefore, the global model suffers more from untargeted attacks. Still, our approach can reduce the aggregation weight of malicious models, which increase the weights of correct models to improve accuracy and reduce loss. The robustness in this article depends on its capacity to resist against poisoning attacks. By measuring the accuracy rates and losses under the attacks, we assess the robustness of our strategy. Thus, the proposed scheme is robust against untargeted attacks and the impact from the data of low quality. Similarly, we also calculated the f-scores of the model under untargeted attacks, which are shown in Table III. The results show that the model has a low number of false positives (i.e., instances where it incorrectly predicts a positive result) and a low number of false negatives.

In addition, during our experiments, the verification process of PoA can increase confidence while guaranteeing that the aggregation is carried out properly. Following each iteration of the aggregation process, the results are published to the blockchain in order to ensure that they can be tracked back to any point in time. This helps achieve the aim of transparency while also eliminating the problem of a single point of failure and the server's potential for malicious conduct.

3) *Computation Overhead*: Compared with standard FL methods, the computing overhead of our proposed scheme is mainly concentrated on homomorphic encryption and model evaluation. We use homomorphic encryption to protect the model's parameters while allowing computation tasks to be performed on the data. However, homomorphic encryption algorithms usually need to consume a lot of resources and time.

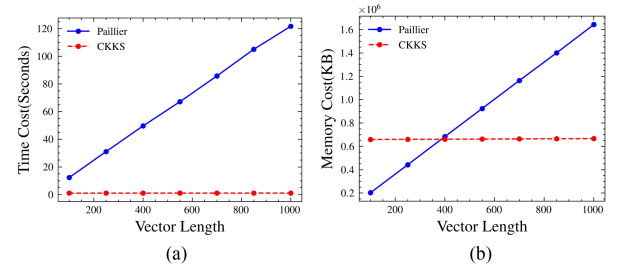


Fig. 9. Homomorphic algorithm performance. (a) Time consumption. (b) Memory usage.

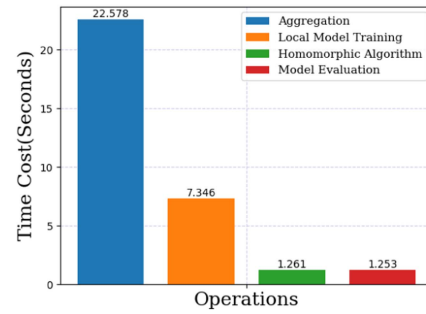


Fig. 10. Overhead time of different operations.

Here, we experiment to compare the two most commonly used homomorphic algorithms' efficiency. Paillier [45] and CKKS are the most popular homomorphic encryption algorithms currently used in PPFL to protect the confidentiality of the model parameters. CKKS is a homomorphic algorithm particularly well-suited for encrypting and processing data in the context of machine learning and other mathematical computations. Also, since the aggregation process only requires an additive operation, a simple cryptosystem, such as Paillier is sufficient and adopted by many PPFL schemes. As shown in Fig. 9, we conduct an experiment to measure the time consumption as well as the peak memory usage of both schemes. We record the data for both schemes to encrypt and decrypt vectors of different lengths ranging from 100 to 1000. The results show that CKKS is a more efficient algorithm in terms of time and memory as the vector length grows. Moreover, the time and memory consumption of the CKKS algorithm is also acceptable. It only takes about 600 MB memory and less than 2 s for CKKS to encrypt and decrypt a 1000-float vector. Therefore, to increase computing efficiency, we encrypt the local parameters using CKKS.

On the other hand, to calculate the aggregated weights for each model, each client still needs to evaluate the model on the auxiliary data after completing the local training, which is also a significantly resource-intensive process. Therefore, we calculate the average elapsed time for model evaluation and compare it to the elapsed time for other heavy calculations per epoch in the scenario pipeline. As shown in Fig. 10, the average model evaluation time only takes about 1 s, while the average local training time takes about 7 s for each epoch. Model evaluation takes roughly the same time as using the homomorphic algorithm. While our proposed scheme may impact the efficiency of the FL process, its impact is relatively minor compared to other computing tasks involved in the process. Therefore, the

proposed scheme does not significantly affect the overall task, and the benefits of enhancing the security and privacy of the data outweigh the potential decrease in efficiency.

VIII. CONCLUSION

In this article, we present a novel aggregation framework by making use of a new approach called PoA to resist model poisoning attacks in PPFL. Additionally, we evaluate our method via a real-world remote sensing image dataset. We judge the effectiveness of our method based on how evenly the customer data is distributed. And our comprehensive trials on the dataset have shown that our system is resistant to being poisoned by malicious users. When the attacker is assumed to be 20%, 40%, 60%, or 80% of the time, our strategy achieves 92.5%, 90.61%, 87.48%, and 81.84% accuracy in experiments. This coordinates with the FedAvg results when only benign clients hold the equivalent data share. Therefore, compared with the no-adversary baseline, our proposed aggregation scheme can achieve equitable test accuracy. In this work, we use a private blockchain to record the model information for research purposes. However, in real life, uploading data to the blockchain takes time and gas, requiring careful consideration for efficiency. As for the time and gas consumption of the blockchain, we leave them for future exploration.

REFERENCES

- [1] Q. Yuan et al., "Deep learning in environmental remote sensing: Achievements and challenges," *Remote Sens. Environ.*, vol. 241, 2020, Art. no. 111716.
- [2] G. Cheng, X. Xie, J. Han, L. Guo, and G.-S. Xia, "Remote sensing image scene classification meets deep learning: Challenges, methods, benchmarks, and opportunities," *IEEE J. Sel. Topics Appl. Earth Observ. Remote Sens.*, vol. 13, pp. 3735–3756, Jun. 2020. [Online]. Available: <https://ieeexplore.ieee.org/abstract/document/9127795/>
- [3] B. Bischke, P. Bhardwaj, A. Gautam, P. Helber, D. Borth, and A. Dengel, "Detection of flooding events in social multimedia and satellite imagery using deep neural networks," in *Proc. MediaEval*, 2017. [Online]. Available: https://www-live.dfki.de/fileadmin/user_upload/import/9269_bischke_MediaEval_2017_MST_solution.pdf
- [4] M. M. U. Rathore, A. Paul, A. Ahmad, B.-W. Chen, B. Huang, and W. Ji, "Real-time big data analytical architecture for remote sensing application," *IEEE J. Sel. Topics Appl. Earth Observ. Remote Sens.*, vol. 8, no. 10, pp. 4610–4621, Oct. 2015.
- [5] P. Tam, S. Math, C. Nam, and S. Kim, "Adaptive resource optimized edge federated learning in real-time image sensing classifications," *IEEE J. Sel. Topics Appl. Earth Observ. Remote Sens.*, vol. 14, pp. 10929–10940, Oct. 2021. [Online]. Available: <https://ieeexplore.ieee.org/abstract/document/9580651/>
- [6] B. Zhang et al., "Progress and challenges in intelligent remote sensing satellite systems," *IEEE J. Sel. Topics Appl. Earth Observ. Remote Sens.*, vol. 15, pp. 1814–1822, Feb. 2022. [Online]. Available: <https://ieeexplore.ieee.org/abstract/document/9705087/>
- [7] F. Zhao, O. I. Fashola, T. I. Olarewaju, and I. Onwumere, "Smart city research: A holistic and state-of-the-art literature review," *Cities*, vol. 119, 2021, Art. no. 103406.
- [8] H. Yoo, R. C. Park, and K. Chung, "IoT-based health big-data process technologies: A survey," *KSII Trans. Internet Inf. Syst.*, vol. 15, no. 3, pp. 974–992, 2021.
- [9] K. Li, G. Wan, G. Cheng, L. Meng, and J. Han, "Object detection in optical remote sensing images: A survey and a new benchmark," *ISPRS J. Photogrammetry Remote Sens.*, vol. 159, pp. 296–307, 2020.
- [10] Q. Pan, J. Wu, A. K. Bashir, J. Li, W. Yang, and Y. D. Al-Otaibi, "Joint protection of energy security and information privacy for energy harvesting: An incentive federated learning approach," *IEEE Trans. Ind. Informat.*, vol. 18, no. 5, pp. 3473–3483, May 2022.
- [11] B. Li, Y. Wu, J. Song, R. Lu, T. Li, and L. Zhao, "Deepfed: Federated deep learning for intrusion detection in industrial cyber-physical systems," *IEEE Trans. Ind. Informat.*, vol. 17, no. 8, pp. 5615–5624, Aug. 2021.
- [12] G. Li, J. Wu, S. Li, W. Yang, and C. Li, "Multitentacle federated learning over software-defined industrial Internet of Things against adaptive poisoning attacks," *IEEE Trans. Ind. Informat.*, vol. 19, no. 2, pp. 1260–1269, Feb. 2023.
- [13] F. Armknecht et al., "A guide to fully homomorphic encryption," *Cryptol. ePrint Arch.*, vol. 1192, 2015.
- [14] M. Fang, X. Cao, J. Jia, and N. Z. Gong, "Local model poisoning attacks to Byzantine-robust federated learning," in *Proc. 29th USENIX Conf. Secur. Symp.*, 2020, pp. 1623–1640.
- [15] V. Tolpegin, S. Truex, M. E. Gursoy, and L. Liu, "Data poisoning attacks against federated learning systems," in *Proc. Comput. Secur.—ESORICS: 25th Eur. Symp. Res. Comput. Secur.*, 2020, pp. 480–501.
- [16] Y. Dong, X. Chen, K. Li, D. Wang, and S. Zeng, "FLOD: Oblivious defender for private Byzantine-robust federated learning with dishonest-majority," in *Proc. Comput. Secur.—ESORICS: 26th Eur. Symp. Res. Comput. Secur.*, 2021, pp. 497–518.
- [17] C. Xu, Y. Jia, L. Zhu, C. Zhang, G. Jin, and K. Sharif, "TDFL: Truth discovery based Byzantine robust federated learning," *IEEE Trans. Parallel Distrib. Syst.*, vol. 33, no. 12, pp. 4835–4848, Dec. 2022.
- [18] Z. Zhou, C. Xu, M. Wang, X. Kuang, Y. Zhuang, and S. Yu, "A multi-shuffler framework to establish mutual confidence for secure federated learning," *IEEE Trans. Dependable Secure Comput.*, to be published, 2022, doi: [10.1109/TDSC.2022.3215574](https://doi.org/10.1109/TDSC.2022.3215574).
- [19] B. C. Xing, M. Shanahan, and R. Leslie-Hurd, "Intel software guard extensions (intel SGX) software support for dynamic memory allocation inside an enclave," in *Proc. Hardware Architectural Support Secur. Privacy*, 2016, pp. 1–9.
- [20] J. H. Cheon, A. Kim, M. Kim, and Y. Song, "Homomorphic encryption for arithmetic of approximate numbers," in *Proc. Adv. Cryptol.—ASIACRYPT: 23rd Int. Conf. Theory Appl. Cryptol. Inf. Secur.*, Hong Kong, China, 2017, pp. 409–437.
- [21] C. Geiß, P. A. Pelizari, L. Blickensdörfer, and H. Taubenböck, "Virtual support vector machines with self-learning strategy for classification of multispectral remote sensing imagery," *ISPRS J. Photogrammetry Remote Sens.*, vol. 151, pp. 42–58, 2019.
- [22] X. Wang et al., "Land-cover classification of coastal wetlands using the RF algorithm for worldview-2 and landsat 8 images," *Remote Sens.*, vol. 11, no. 16, 2019, Art. no. 1927.
- [23] Y. Li, R. Chen, Y. Zhang, M. Zhang, and L. Chen, "Multi-label remote sensing image scene classification by combining a convolutional neural network and a graph neural network," *Remote Sens.*, vol. 12, no. 23, 2020, Art. no. 4003.
- [24] H. Alhichri, A. S. Alswayed, Y. Bazi, N. Ammour, and N. A. Alajlan, "Classification of remote sensing images using efficientnet-b3 CNN model with attention," *IEEE Access*, vol. 9, pp. 14078–14094, 2021.
- [25] G. Cheng et al., "SPNet: Siamese-prototype network for few-shot remote sensing image scene classification," *IEEE Trans. Geosci. Remote Sens.*, vol. 60, Jul. 2022, Art. no. 5608011. [Online]. Available: <https://ieeexplore.ieee.org/abstract/document/9501951/>
- [26] A. Ma, Y. Wan, Y. Zhong, J. Wang, and L. Zhang, "Scenenet: Remote sensing scene classification deep learning network using multi-objective neural evolution architecture search," *ISPRS J. Photogrammetry Remote Sens.*, vol. 172, pp. 171–188, 2021.
- [27] K. Xu, H. Huang, P. Deng, and Y. Li, "Deep feature aggregation framework driven by graph convolutional network for scene classification in remote sensing," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 33, no. 10, pp. 5751–5765, Oct. 2022.
- [28] J. Chen et al., "Improving few-shot remote sensing scene classification with class name semantics," *IEEE Trans. Geosci. Remote Sens.*, vol. 60, Nov. 2022, Art. no. 5633712. [Online]. Available: <https://ieeexplore.ieee.org/document/9940200/>
- [29] D. Yin, Y. Chen, R. Kannan, and P. Bartlett, "Byzantine-robust distributed learning: Towards optimal statistical rates," in *Proc. Int. Conf. Mach. Learn.*, 2018, pp. 5650–5659.
- [30] K. J. Smith, *Precalculus: A Functional Approach to Graphing and Problem Solving*. Boston, MA, USA: Jones & Bartlett, 2011.
- [31] P. Blanchard, E. M. El Mhamdi, R. Guerraoui, and J. Stainer, "Machine learning with adversaries: Byzantine tolerant gradient descent," *Adv. Neural Inf. Process. Syst.*, vol. 30, 2017. [Online]. Available: <https://proceedings.neurips.cc/paper/2017/hash/f4b9ec30ad9f68f89b29639786cb62ef-Abstract.html>
- [32] E. M. E. Mhamdi, R. Guerraoui, and S. Rouault, "The hidden vulnerability of distributed learning in byzantium," 2018, *arXiv:1802.07927*.
- [33] S. Shen, S. Tople, and P. Saxena, "Auror: Defending against poisoning attacks in collaborative deep learning systems," in *Proc. 32nd Annu. Conf. Comput. Secur. Appl.*, 2016, pp. 508–519.

- [34] C. Fung, C. J. Yoon, and I. Beschastnikh, "The limitations of federated learning in sybil settings," in *Proc. RAID*, 2020, pp. 301–316.
- [35] X. Liu, H. Li, G. Xu, Z. Chen, X. Huang, and R. Lu, "Privacy-enhanced federated learning against poisoning adversaries," *IEEE Trans. Inf. Forensics Secur.*, vol. 16, pp. 4574–4588, Aug. 2021. [Online]. Available: <https://ieeexplore.ieee.org/abstract/document/9524709>
- [36] Z. Ma, J. Ma, Y. Miao, Y. Li, and R. H. Deng, "Shieldfl: Mitigating model poisoning attacks in privacy-preserving federated learning," *IEEE Trans. Inf. Forensics Secur.*, vol. 17, pp. 1639–1654, 2022.
- [37] X. Cao, M. Fang, J. Liu, and N. Z. Gong, "Fltrust: Byzantine-robust federated learning via trust bootstrapping," 2020, *arXiv:2012.13995*.
- [38] Y. Miao, Z. Liu, H. Li, K.-K. R. Choo, and R. H. Deng, "Privacy-preserving Byzantine-robust federated learning via blockchain systems," *IEEE Trans. Inf. Forensics Secur.*, vol. 17, pp. 2848–2861, Aug. 2022. [Online]. Available: <https://ieeexplore.ieee.org/abstract/document/9849010>
- [39] F. McKeen et al., "Innovative instructions and software model for isolated execution," *Hasp, isca*, vol. 10, no. 1, 2013. [Online]. Available: <https://ieeexplore.ieee.org/abstract/document/7961949/>
- [40] F. Tramer, F. Zhang, H. Lin, J.-P. Hubaux, A. Juels, and E. Shi, "Sealed-glass proofs: Using transparent enclaves to prove and sell knowledge," in *Proc. IEEE Eur. Symp. Secur. Privacy*, 2017, pp. 19–34.
- [41] N. Kumar, M. Rathee, N. Chandran, D. Gupta, A. Rastogi, and R. Sharma, "Cryptflow: Secure tensorflow inference," in *Proc. IEEE Symp. Secur. Privacy*, 2020, pp. 336–353.
- [42] Y. Aono et al., "Privacy-preserving deep learning via additively homomorphic encryption," *IEEE Trans. Inf. Forensics Secur.*, vol. 13, no. 5, pp. 1333–1345, May 2017.
- [43] P. Helber, B. Bischke, A. Dengel, and D. Borth, "EuroSAT: A novel dataset and deep learning benchmark for land use and land cover classification," *IEEE J. Sel. Topics Appl. Earth Observ. Remote Sens.*, vol. 12, no. 7, pp. 2217–2226, Jul. 2019.
- [44] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. Y. Arcas, "Communication-efficient learning of deep networks from decentralized data," in *Proc. Artif. Intell. Statist.*, 2017, pp. 1273–1282.
- [45] P. Paillier and D. Pointcheval, "Efficient public-key cryptosystems provably secure against active adversaries," in *Proc. Adv. Cryptol.-ASIACRYPT'99: Int. Conf. Theory Appl. Cryptol. Inf. Secur.*, 1999, pp. 165–179.



Jiang Zhu is currently working toward the master's degree with the Graduate School of Information, Production and System, Waseda University, Fukuoka, Japan.

His current research interests include blockchain and privacy protection.



Jun Wu (Senior Member, IEEE) received the Ph.D. degree in information and telecommunication studies from Waseda University, Tokyo, Japan, in 2011.

He is currently a Professor with the Graduate School of Information, Production and Systems, Waseda University. He is the author or coauthor of more than 200 peer-reviewed journal/conference papers within the abovementioned topics. His research interests include the intelligence and security techniques of Internet of Things (IoT), edge computing, big data, 5G/6G, etc.

Dr. Wu is the Chair of IEEE P21451-1-5 Standard Working Group for Internet of things. His publications was the recipient of a few distinctions, which includes the Best Paper Award of IEEE TRANSACTIONS ON EMERGING TOPICS IN COMPUTING, in 2020, Best Paper Award of International Conference on Telecommunications and Signal Process in 2019, Best Conference Paper Award of the IEEE ComSoc Technical Committee on Communications Systems Integration and Modeling in 2018. He was the Track Chair for Vehicular Technology Conference (VTC) 2019, VTC 2020, and the TPC Member of more than ten international conferences including IEEE International Conference on Communications, GLOBECOM, etc. He is currently an Associate Editor for the IEEE SYSTEMS JOURNAL and IEEE NETWORKING LETTERS. He was the as a Guest Editor for the IEEE TRANSACTIONS ON INDUSTRIAL INFORMATICS, IEEE TRANSACTIONS ON INTELLIGENT TRANSPORTATION, IEEE SENSORS JOURNAL, *Sensors*, *Frontiers of Information Technology & Electronic Engineering* (FI-TEE), etc.



Ali Kashif Bashir (Senior Member, IEEE) received the Ph.D. degree in computer science and engineering from Korea University, Seoul, South Korea, in 2012.

He is the leader with the Department of Computing and Mathematics, Manchester Metropolitan University, Manchester, U.K., and also an Adjunct Professor with the Woxsen School of Business, Woxsen University, Hyderabad, India. His research interests include Internet of Things, wireless networks, distributed systems, network/cyber security, network function virtualization, machine learning, etc.

Dr. Bashir is a member of IEEE Industrial Electronic Society, member of ACM, and Distinguished Speaker of ACM. He authored or coauthored more than 200 research articles; and was the recipient of more than 3 million USD funding as PI and Co-PI from research bodies of South Korea, Japan, EU, U.K., and Middle East. He is currently the Editor-in-Chief of the IEEE FUTURE DIRECTIONS NEWSLETTER. He is also an Area Editor for *KSII Transactions on Internet and Information Systems*; an Associate Editor for IEEE INTERNET OF THINGS MAGAZINE, IEEE ACCESS, *Peer J Computer Science*, *IET Quantum Computing*, and *Journal of Plant Disease and Protection*. He is leading many conferences as a Chair (program, publicity, and track) and had organized workshops in flagship conferences, such as IEEE Infocom, IEEE Globecom, and IEEE Mobicom.



Qianqian Pan (Member, IEEE) received B.S. and M.S. in information and communication engineering from the School of Information Science and Engineering, Southeast University, Nanjing, China, in 2015 and 2018, respectively. She received the Ph.D. degree in cyberspace security from the School of Electronic Information and Electrical Engineering, Shanghai Jiao Tong University, Shanghai, China, in 2023.

From July 2022 to December 2022, she visited the Muroran Institution of Technology, Muroran, Japan, supported by the China Scholarship Council Program. She is currently a Researcher with the Department of Systems Innovation, School of Engineering, University of Tokyo, Tokyo, Japan. Her research interests include blockchain, privacy protection, and next-generation network security.

Dr. Pan was the recipient of the IEEE best student paper award runner-up from the IEEE CPSCOM 2021. She is the Reviewer for IEEE TRANSACTIONS ON DEPENDABLE AND SECURE COMPUTING, IEEE TRANSACTIONS ON INDUSTRIAL INFORMATICS, IEEE JOURNAL ON SELECTED AREAS IN COMMUNICATIONS, and TPC member for IEEE Vehicular Technology Conference 2023.



Wu Yang received the Ph.D. degree in computer system architecture specialty from the Computer Science and Technology School, Harbin Institute of Technology, Harbin, China, in 2005.

He is currently a Professor and a Doctoral Supervisor with Harbin Engineering University, Harbin, China. His main research interests include wireless sensor network, peer-to-peer network, and information security.

Dr. Yang is a member of ACM and a Senior Member of CCF.