Practical Privacy-Preserving Federated Learning in Vehicular Fog Computing

Yiran Li, *Graduate Student Member, IEEE*, Hongwei Li, *Senior Member, IEEE*, Guowen Xu, *Member, IEEE*, Tao Xiang, *Member, IEEE*, and Rongxing Lu, *Fellow, IEEE*

***Abstract*—Benefitting from the outstanding capabilities of intelligent controlling and prediction, federated learning (FL) has been widely applied in Internet of Vehicle (IoV). However, applying FL into fog-computing-based IoV still suffers from two crucial problems: (i) how to achieve the privacy-preserving FL under the flexible architecture of fog computing with no assistance of cloud server, and (ii) how to guarantee the privacy-preserving FL to perform with high efficiency and low overhead in fog-computing settings. For addressing the above issues, we propose a practical framework, named GALAXY, the first of its kind in the regime of privacy-preserving FL under the setting of non-cloud-assisted fog computing. Based on the secure multi-party computation (MPC) technology, our framework satisfies the (T, N)-threshold property, permitting N (a scalable number) fog nodes to cooperate with multiple users for implementing privacy-preserving FL, while resisting the collusion up to $T-1$ fog nodes, and being robust to at most $N-T$ fog nodes simultaneously dropping out. Besides, considering the practical scenario that low-quality data may negatively impair the FL model convergence, our scheme can handle users' low-quality data while protecting all user-related information under our secure framework. Based on the above superior properties, our scheme can perform with high scalability, high processing efficiency, and low resource overhead, being practical for fog-computing-based IoV. Extensive experiment results demonstrate our scheme with high-level performance.**

***Index Terms*—Fog computing, IoV, privacy-preserving federated learning, secure multi-party computation.**

I. INTRODUCTION

FEDERATED learning (FL) has attracted a wide attention in both academic and industry fields due to its superiorities in scalability and security. Meanwhile, enormous FL-based applications have been proposed in fog-based internet of vehicles (IoV), since both fog computing and FL are implemented under a similar distributed architecture. For example, applying FL for establishing the automatic-driving platform has been demonstrated high accuracy and efficiency for auto-pilot and real-time driving control. In the malfunction detection domain, FL-based framework, being exploited to detect faults of automobiles, performs with far more precision and lower detection error rate than non-intelligent detection approaches. Undoubtedly, federated learning integrated with fog computing has brought IoV with full of vitality.

Nevertheless, some hard stumbling blocks are still hindering the proliferation of FL in fog-computing-based IoV. One of the most important issues is to protect users' privacy during the FL training process. Since users' raw data is not required to be shared to the service center during the training process, FL can protect users' private information to a certain degree. However, users' private information can still be inferred from the gradients shared by users [1]–[3]. For addressing this issue, many state-of-the-art works have been proposed in various dimensions, which are mainly derived from three technologies: differential privacy (DP), homomorphic encryption (HE), and secure multi-party computation (MPC). For privacy protection, DP technology can perform quite efficiently, since its main idea is to mask related information with noise (such as Laplace noise, Gaussian noise, etc.), requiring only a little additional overhead. However, applying DP to achieve private federated learning is required to balance accuracy and privacy, which means that the higher model accuracy is obtained, the more privacy will be lost. Meanwhile, recent research result [4] demonstrates that current DP-based FL can rarely offer satisfied accuracy while guaranteeing acceptable privacy. Homomorphic encryption, e.g., linear homomorphic encryption (LHE), enables computations on encrypted data, without loss of data utility. However, the privacy of such method depends on enormous modular exponential operations in ciphertext domain, which is computationally intensive. When it is utilized for protecting FL, the large size of ciphertext will become an insurmountable burden for computing participants.

In contrast to DP and HE technologies, MPC allows multiple parties to securely calculate an agreed function. Benefitting from the higher scalability, MPC-based methods seem more powerful for establishing privacy-preserving FL in vehicular fog computing. Currently, several state-of-the-art researches have constructed their frameworks on two [5]–[7], three [8]–[10],

Manuscript received July 29, 2021; revised November 25, 2021 and January 2, 2022; accepted January 31, 2022. Date of publication February 14, 2022; date of current version May 20, 2022. This work was supported in part by the National Natural Science Foundation of China under Grants 62020106013, 61972454, 61802051, 61772121, and 61728102, in part by the Sichuan Science and Technology Program under Grants 2020JDTD0007 and 2020YFG0298, and in part by the Fundamental Research Funds for Chinese Central Universities under Grant ZYGX2020ZB027. The review of this article was coordinated by Dr. Xuanyu Cao. *(Corresponding author: Hongwei Li.)*

Yiran Li and Hongwei Li are with the School of Computer Science and Engineering, University of Electronic Science and Technology of China, Chengdu 611731, China, and also with the Cyberspace Security Research Center, Peng Cheng Laboratory, Shenzhen 518000, China (e-mail: yiranli842@foxmail.com; hongweili@uestc.edu.cn).

Guowen Xu is with the School of Computer Science and Engineering, Nanyang Technological University (NTU), Singapore 639798 (e-mail: guowen.xu@foxmail.com).

Tao Xiang is with the College of Computer Science, Chongqing University, Chongqing 400044, China (e-mail: txiang@cqu.edu.cn).

Rongxing Lu is with the Faculty of Computer Science (FCS), University of New Brunswick (UNB), Fredericton E3B 5G3, Canada (e-mail: rlu1@unb.ca).

Digital Object Identifier 10.1109/TVT.2022.3150806

or four [11], [12] servers. Such methods, nevertheless, limit the number of computing servers in fixed 2, 3, and 4. As we know, fog computing has flexible architectures in IoV, which means that both the participating fog nodes and the number of participating fog nodes are changeable. Obviously, current MPC-based methods, taking fixed number of computing entities, are not suitable for fog computing because of the limited scalability. Besides, almost their schemes require an honest majority among the computing servers, and the collusion between any two servers will thoroughly destroy their secure framework. This might not meet the demands of privacy and confidentiality, let alone more strict protection regulations. Furthermore, due to the instability of equipment or environment, servers may be out of work sometimes. However, in their schemes, once any one of the servers drops out, the whole process will be broken off immediately.

Additionally, current researches [13], [14] have demonstrated that quality-discrepancy actually exists among different data suppliers in practical FL training process. Indeed, utilizing low-quality data for training FL model may decrease model accuracy, and even cause the model divergence. Hence, for enhancing the practicality of FL in fog-computing-based IoV, it is essential to handle these low-quality data shared by users. As far as we know, two recent methods of privacy-preserving FL [15], [16] have considered this issue. *Zhao et al.* [15] proposed the first scheme, named SecProbe, where they construct their secure framework based on the technology of differential privacy (DP) in a single-cloud setting. However, their scheme still suffers from the limited utility of DP, which is mentioned above. Following *Zhao et al.*'s work, *Xu et al.* [16] proposed PPFDL, exploiting MPC to construct their secure framework in a two-server setting. Nevertheless, similar to above two-server MPC-based methods [5]–[7], their scheme is not suitable for fog-computing due to the limitations of scalability and security.

Beyond of the above-mentioned limitations, none of these existing efforts have considered a significant issue, that is, how to satisfy the low latency and low resource overhead in practical fog-computing-based IoV. Implementing cloud-assisted fog computing usually requires multiple interactions between the cloud service center and fog nodes, thus causing much more time delay and resource overhead [17]. Therefore, it is crucial to establish a none-cloud-assisted framework for supplying higher timeliness and freeing participants from the trouble of resources. For addressing the above issues, in this paper, we propose GALAXY, a practical privacy-preserving federated learning in vehicular fog computing. We summarize our contributions as follows:

- A novel framework is proposed to implement privacy-preserving FL in the none-cloud-assisted vehicular fog computing. Through the masterly utilizations of Shamirs secret sharing and Lagrange interpretation, our scheme can guarantee the information-theoretic privacy for all users, while satisfying the (T, N)-threshold property.
- Our GALAXY advances current MPC-based methods for privacy-preserving machine learning, in terms of scalability, security, and robustness.

  Scalability. GALAXY splits the computation among multiple computing nodes, extended to unbounded N, while current MPC-based methods are limited in 2, 3, or 4. Therefore, our GALAXY can perform with more scalability in the flexible fog computing.

  Security. GALAXY resists the collusion among up to $T-1$ computing nodes, while users' private information can still be protected under a semi-honest adversarial model. This is contrast to MPC-based methods, which allows no collusion between servers.

  Robustness. GALAXY supports at most $N-T$ fog nodes simultaneously being off-line, and the remaining fog nodes can still guarantee the training process being executed smoothly, whereas MPC-based methods permit no computing server to drop out.
- Different from DP-based approaches, our GALAXY will not decrease the usability of users' data, performing with a slight impact on the model accuracy.
- Unlike HE-based schemes computing under large-scale ciphertext modes, our GALAXY is executed in a small finite field with the same scale as plaintext, which benefits our scheme with more advantages in resource overhead.
- Our scheme can alleviate the negative impact of low-quality data, while protecting all user-related information under our secure framework, including gradients and aggregation weights.
- We rigorously prove the security of our GALAXY through UC model, and conduct extensive experiments to demonstrate the high-level performance in terms of accuracy, efficiency, and resource overhead.

The remaining parts of this paper are organized as follows. We state the problem and review some primitives in Section II. We introduce our scheme and give a rigorous proof, respectively in Sections III and IV. Then we present a comprehensive analysis of performance in Section V. Next, we discuss some related works in Section VI. Finally, we conclude our work in Section VII.

II. PROBLEM STATEMENT AND PRIMITIVES

A. System Overview

As shown in Fig. 1, we consider that N fog nodes cooperate with M users to achieve the FL training process. To be specific, all the participants hold the same deep neural network (DNN) initialized with the global weight ω. Each users m first calculates its local gradient g_m, throgh training on its own local data. Then g_m, as a secret, will be split to N Shamir's shares, and each n-labeled share will be sent to each fog node P_n. Nextly, through our MPC-based framework, N nodes cooperate to calculate the newest global weight $\boldsymbol{\omega}$ for updating the DNN model. For achieving the training task, nodes and users iteratively execute above processes until the convergence condition is satisfied. Note that, during the training process, users just upload shares of gradients, which contain no useful information of the gradients. Additionally, after receiving the shares, all nodes will execute our MPC-based protocol to obtain shares of the global weight. Whenever it is required, any T out of N fog nodes can reconstruct the global weight.

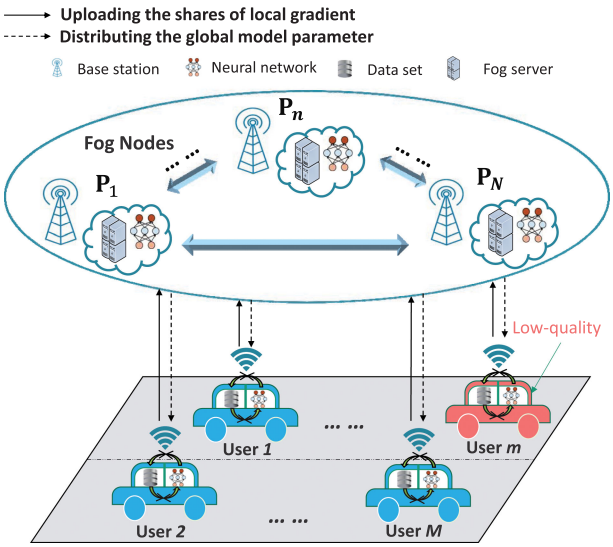


Fig. 1. System overview.

Remark: In this paper, we consider a practical fog-computing scenario for FL, where no cloud server is supposed to provide any assistance. Unlike cloud-assisted methods [18], [19], no interaction between fog nodes and the cloud server is required in our scheme, thus shortening the processing delay and lowering the resource overhead. Since users are just required to upload some shares, while other computations are outsourced to fog nodes, it is very friendly to edge users with limited resources. Additionally, we consider a practical scenario that some users may achieve the local training based on low-quality data. Therefore, their uploaded gradients may deviate from the convergence trend, specifically, some components of these gradient vectors may emerge with inverse signs or biased values. As a result, these gradients may slow down the convergence of the DNN model, and even ease the model divergence. For addressing this issue, we utilize the methods of removing contradictory components and weighted aggregation to alleviate the negative impact caused by low-quality data. Meanwhile, all user-related private information will be protected under our secure framework. More details are presented in Section III.

B. Threat Model and Goals

In our GALAXY, the main security threats come from the fog nodes, since users will upload the shares of private information to these nodes. In this paper, we consider a *passive* adversary mode, where fog nodes are *semi-honest*, which means that each node will strictly execute the pre-designed protocol, being not allowed to change the input shares by themselves. Nevertheless, some of these nodes may try to infer users' data privacy through exploiting mastered prior information (such as the number of participants, the labels of participants, etc). Besides, our GALAXY permits any less than T fog nodes to collude with each other, while no useful information of users can be disclosed to the colluded group.

Under the above threat model, we formulate the privacy requirements as follows.

- *Confidentiality of users' gradients:* An adversary may compromise fog nodes for inferring the gradients of users, thus disclosing users' private information (e.g., name, address, etc) based on these leaked gradients. Therefore, it is essential to protect users' gradient information from being leaked to fog nodes during the training process.
- *Protection of users' aggregation weights:* For guaranteeing the training to be fair and non-discriminative to all of the users, the information of the aggregation weights, representing "quality of data," should also be guaranteed private against fog nodes.

C. Secret Sharing

In this paper, we utilize the classical Shamir's T-out-of-N secret sharing [20] to establish our MPC-based protocol, which can benefit our framework with abundant functionality, scalability, security, and efficiency.

The main idea of this scheme is derived from how to solve a polynomial. Considering a polynomial of degree at most $T-1$: $f_s(x) = s + a_1 x + \cdots + a_t x^t + \cdots + a_{T-1}x^{T-1}$, where $s \in \mathbb{F}$ is the secret, such that $f_s(0) = s$, and coefficients of $a_1, \ldots, a_t, \ldots, a_{T-1}$ are randomly selected from $\mathbb{F}$, where $\mathbb{F} = \mathbb{Z}_p$ is a finite field, and p is a prime with the restriction condition of $p > N$. Then the secret s will be split into N parts, and any T parts of them can reconstruct the secret s, but any $T-1$ parts can not recover the secret (no useful information of the secret s can be inferred). Note that, all the operations will be over the finite field $\mathbb{Z}_p$. The specific implementation of this secret sharing protocol is introduced as below.

1) **S.share** Based on the polynomial $f_s(x)$, the secret s can be divided into N parts, and the n-th part P_n will privately hold the secret share $s_n = f(n)$, then the whole set of secret shares can be formally denoted as $[s; f_s(n)]_T = (f_s(1), \ldots, f_s(n), \ldots, f_s(N))$. For simplicity, we denote it as $[s]_T$, and each P_n holds $[s]_T^n = f_s(n)$.
2) **S.recon** Randomly selecting T shares, e.g., $[s, f_s(1)]$; $[s, f_s(2)]$; $\cdots$; $[s, f_s(T)]$, from the set of $[s]_T$. Then the secret $s = f_s(0)$ can be obtained through Lagrange interpolation, which will be introduced as follows.

D. Lagrange Interpolation

Lagrange interpolation presents another form of polynomials, which can be utilized for reconstructing the secret s with higher efficiency. Still considering a polynomial $f(x)$ over $\mathbb{Z}_p$ of degree at most $T-1$, and C is subset of $\mathbb{Z}_p$ with $|C| = T$, the polynomial can be defined as

$$f(x) = \sum_{i \in C} f(i)\lambda_i(x) \tag{1}$$

where $\lambda_i(x)$ is a polynomial degree at $T-1$, and for $i, j \in C$, if $i \neq j$, $\lambda_i(j) = 0$, else if $i = j$, $\lambda_i(j) = 1$, in other words,

$$\lambda_i(x) = \prod_{j \in C, j \neq i} \frac{x - j}{i - j} \tag{2}$$

Based on (2), it can be observed that there exists an easily computable set of values $r = \{r_i | i \in C\}$, where $r_i = \lambda_i(0)$, such

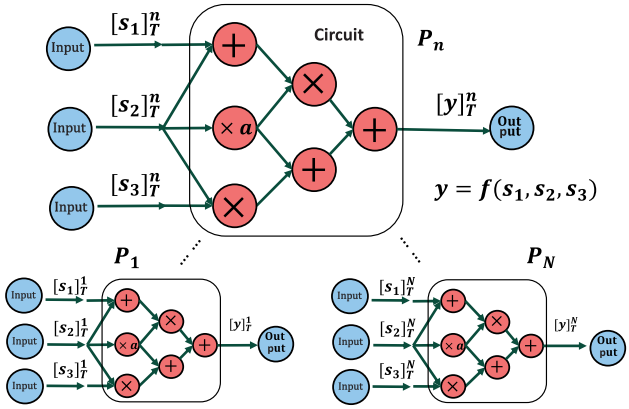


Fig. 2. Arithmetic circuit.

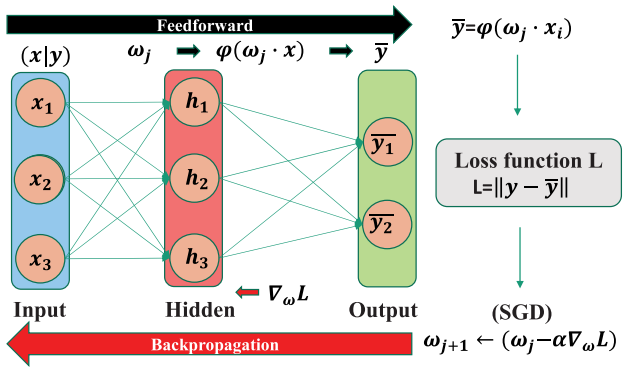


Fig. 3. Neural network.

that $s = f(0) = \sum_{i \in C} r_i f(i)$. The r can be considered as the recombination vector, which is a public information computed and held by all players. Note that $\lambda_i(x)$ does not depend on $f(x)$, so neither does $\lambda_i(0)$. Hence, the same recombination vector r works for all $f(x)$.

E. Arithmetic Circuit

In this paper, we utilize the arithmetic circuit (AC) for securely evaluating the function F, whose operations are all over a finite field $\mathbb{Z}_p$. For the convenience of explanation, we consider a simple function $F(s_1, s_2, s_3) = (s_1 + s_2) * (as_2) + (s_2 * s_3) + (as_2) = y$, taking s_1, s_2, s_3 as input and y as output. According to this function, we construct a simple circuit described by an acyclic directed graph, as shown in Fig. 2, where nodes are denoted as gates, connected by wires. Indeed, our AC supports three types of internal operation gates, i.e., addition, multiplication, and multiply-by-constant ($\times a \in \mathbb{Z}_p$) gates, all of which are presented in the example circuit in Fig. 2. Note that addition and multiplication gates have 2 input wires, while the multiply-by-constant gate has just 1 input wire, and all of these gates have more than 1 output wires. Additionally, one output gate (1 input wire and no output wire) is implemented for each P_n.

For securely executing the AC protocol, each input secret s_i will be divided into N secret shares as $[s_i]_T^1; \ldots; [s_i]_T^n; \ldots; [s_i]_T^N$ based on Shamir's sharing, being respectively taken as the input sharing for each P_n. Then each P_n, holding the unified AC, calculates over these shares based on the internal operation gates. For guaranteeing the correctness of the AC, all the gates in the AC should be processed. After this, each P_n will obtain an output $[y]_T^n$, which is a secret share of y at T-threshold Shamir's sharing.

F. Federated Learning

For clarifying the main idea of deep neural networks (DNNs), we present a fully-connected neural network [21] shown in Fig. 3, which consists of 3 layers, i.e., input layer, hidden layer, and output layer. As depicted in Fig. 3, training a neural network can be divided into two phases, i.e., feed-forward and back-propagation, where the first one aims to obtain an output $\bar{\boldsymbol{y}}$ through a series of operations on the labeled input $(\boldsymbol{x}|\boldsymbol{y})$, while the second one is for updating the weight $\boldsymbol{\omega}$ of the neural network by the algorithm of stochastic gradient descent (SGD) [22]. Specifically, given the input vector $\boldsymbol{x} = \{x_1, x_2, x_3\}$ labeled by $\boldsymbol{y}$, the feed-forward process can be described as a function $\varphi(\boldsymbol{x}, \boldsymbol{\omega}) = \bar{\boldsymbol{y}}$, taking $\boldsymbol{x}$ as input and $\bar{\boldsymbol{y}}$ as output, where $\boldsymbol{\omega}$ denotes the parameter of weight connecting nodes between adjacent layers. Then given the output $\bar{\boldsymbol{y}}$ and label $\boldsymbol{y}$, the loss function can be obtained as a 2 $norm$ of a vector: $\mathcal{L}_\varphi((\boldsymbol{x}, \boldsymbol{y}), \boldsymbol{\omega}) = ||\boldsymbol{y} - \bar{\boldsymbol{y}}||_2 = ||\boldsymbol{y} - \varphi(\boldsymbol{x}, \boldsymbol{\omega})||_2$. During the back-propagation process, we can minimize the loss function to adjust the weight, based on the algorithm of stochastic gradient descent (SGD) [23]–[25]. Specifically, the weight updating process can be described as follows,

$$\boldsymbol{\omega}_{j+1} \leftarrow \boldsymbol{\omega}_j - \alpha \nabla \mathcal{L}_\varphi((\boldsymbol{x}, \boldsymbol{y}), \boldsymbol{\omega}_j) \tag{3}$$

where α denotes the learning rate. Through iteratively executing feed-forward and back-propagation until an approximate minimum is obtained, we can obtain the optimal parameter of weight for the DNNs.

Now we review federated learning as follows. Considering M participating users, each user $m \in [1, M]$ holds a local data set $\mathcal{D}^m$, e.g., $\mathcal{D}^m = \{(\boldsymbol{x}_k, \boldsymbol{y}_k); k = 1, 2 \ldots, K\}$, so the total data set $\mathcal{D}$ can be defined as $\mathcal{D} = \cup_{m \in [1,M]} \mathcal{D}^m$. In j-th iteration, a mini-batch $\mathcal{D}_j^m$ will be randomly chosen by user m, and the total data set is denoted as $\mathcal{D}_j = \cup_{m \in [1,M]} \mathcal{D}_j^m$. Then each user m locally computes $g_j^m = \sum_{(\boldsymbol{x}_k, \boldsymbol{y}_k) \in \mathcal{D}_j^m} \nabla \mathcal{L}_\varphi((\boldsymbol{x}_k, \boldsymbol{y}_k), \boldsymbol{\omega}_j)$, and uploads the g_j^m to the service center. After this, the weight is updated as

$$\boldsymbol{\omega}_{j+1} \leftarrow \boldsymbol{\omega}_j - \alpha \cdot \frac{\sum_{m \in [1,M]} g_j^m}{\sum_{m \in [1,M]} |\mathcal{D}_j^m|} \tag{4}$$

Finally, $\boldsymbol{\omega}_{j+1}$ will be broadcasted to each user for updating their local neural network. The above process will be iteratively executed until satisfying the predefined convergence condition.

III. Our Scheme

In this section, we will introduce the details of our GALAXY, which enables fog nodes and users cooperatively achieve the privacy-preserving FL. We first introduce our secure circuit evaluation protocol (SCEP), the footstone of our secure framework.

Secure Circuit Evaluation Protocol (SCEP)

Input: Each $user_m$ holds a value $g_m \in \mathbb{Z}_p$, where $m \in [1, M]$.
Output: The output of function F: $y = F(g_1, ..., g_m, ...g_M)$, where F is operated over $\mathbb{Z}_p$.
Procedure:

- **Sharing:** Each $user_m$ generates the Shamir's sharing as $[g_m; f_{g_m}]_T$, for a polynomial f_{g_m} at the degree of $deg(f_{g_m}) = T-1$, and securely sends it to each node P_n, $n \in [1, N]$.
- **Computation:** For correctly evaluating the function F, all the gates of the arithmetic circuit will be processed. Our protocol supports three types of gates as follows.
  Addition: All the nodes hold two shares e.g., $[g_1, f_{g1}]_T$ and $[g_2, f_{g2}]_T$ for the two inputs of g_1 and g_2. All nodes compute $[g_1, f_{g1}]_T + [g_2, f_{g2}]_T = [g_1 + g_2; f_{g1} + f_{g2}]_T$.
  Multiply-by-constant: All the nodes hold $[g_1, f_{g1}]_t$ for the input g_1. Then all the nodes compute $a[g_1, f_{g1}]_T = [ag_1, af_{g1}]_T$, where a is a constant belonging to $\mathbb{Z}_p$.
  Multiplication: All the nodes hold $[g_1, f_{g1}]_T$ and $[g_2, f_{g2}]_T$ for the two inputs of g_1 and g_2. All the nodes first locally compute $[g_1, f_{g1}]_T * [g_2, f_{g2}]_T = [g_1 g_2; f_{g1} f_{g2}]_{2T}$. Then considering $h = f_{g1} f_{g2}$ with degree at $deg(h) = 2(T-1) \leq N-1$, thus $h(0) = f_{g1}(0) f_{g2}(0) = g_1 g_2$, and all the nodes hold $[g_1 g_2; h]_{2T}$, i.e., P_n holds $h(n) = f_{g1}(n) f_{g2}(n)$. Each P_n distributes $[h(n); f_{h(n)}]_T$ for any polynomial $f_{h(n)}$ of degree at $T-1$. Next, considering a recombination vector $r = (r_1, ..., r_N)$, holding $h(0) = \sum_{n=1}^{N} r_n h(n)$ for any polynomial $f_{h(n)}$ of degree $\leq T-1$, all the nodes compute:

$$\sum_{n=1}^{N} r_n [h(n); f_{h(n)}]_T = [\sum_{n=1}^{N} r_n h(n); \sum_{n=1}^{N} r_n f_{h(n)}]_T$$
$$= [h(0); \sum_{n=1}^{N} r_n f_{h(n)}]_T = [g_1 g_2; \sum_{n=1}^{N} r_n f_{h(n)}]_T$$

- **Reconstruction:** After all the gates have been processed in the pre-designed AC, each node P_n holds a secret share of $[y; f_y]_T$ for y, which is the output value of function F. Then, each P_n securely sends $f_y(n)$ to an assigned node P_i, who uses Lagrange interpolation to compute $y = f_y(0)$ from $f_y(1), ..., f_y(T)$, or any other T nodes.

Fig. 4. SCEP: Secure circuit evaluation protocol.

Then we present a method to handle low-quality data. Finally, we introduce how to implement our GALAXY.

A. Secure Circuit Evaluation Protocol

Overall, our secure circuit evaluation protocol (SCEP) is derived from Shamir's secret sharing, Lagrange interpolation, and arithmetic circuit, being executed over the finite field $\mathbb{Z}_p$. In our framework, a function F, operated over $\mathbb{Z}_p$, can be described as an arithmetic circuit (AC). For securely calculating the function F, users' private data will be divided into N labelled Shamir's secret shares and respectively sent to N labelled fog nodes, while each fog node P_n, holding the unified AC, will calculate a final share of the output of F based on the unified AC and the input shares. Finally, an assigned fog node P_i ($i \in [1, N]$) utilizes Lagrange interpolation to reconstruct the output of F. The details are presented in Fig. 4.

Proposition 1: Considering a corrupted (semi-honest) node group $\boldsymbol{A} \subset \{P_1, \ldots P_N\}$, where $|\boldsymbol{A}| < T < N/2$. After executing our $SCEP$, each user's $g_m \in G_M = (g_1, \ldots, g_m, \ldots, g_M)$ will be protected against all nodes in $\boldsymbol{A}$ in an information-theoretic sense.

Proof: As shown in Fig. 4, considering two input vectors $G_M = (g_1, \ldots, g_m, \ldots, g_M)$ and $G'_M = (g'_1, \ldots, g'_m, \ldots, g'_M)$, we prove the security for each part of the protocol.

Sharing: Each $P_n \in \boldsymbol{A}$ just holds shares of G_M. If we replace G_M with G'_M, based on Shamir's secret sharing, the views of $P_n \in \boldsymbol{A}$ will have the same distribution for both G_M and G'_M, so that no useful information of each g_m will be leaked.

Addition and Multiply-by-constant: Here no participant sends or receives anything, so that the privacy will still be guaranteed.

Multiplication: Before the current multiplication, we can consider that all values held by $P_n \in \boldsymbol{A}$ have the same distribution for G_M and substitution G'_M. After the local multiplication by each $P_n \in \boldsymbol{A}$, these values will be supposed to still have the same distribution. Thereby, the generated values by each $P_n \in \boldsymbol{A}$ still have the same distribution, while the honest nodes just send the shares seeming random to each $P_n \in \boldsymbol{A}$. Additionally, $\boldsymbol{A}$ can only see $T-1$ shares, so the final values held by $\boldsymbol{A}$ have the same distribution for both G_M and G'_M.

Output Reconstruction: Here all shares of $[y; f_y]_T$ are held by P_i. If P_i is honest, nothing will be leaked to nodes in $\boldsymbol{A}$. If $P_i \in A$ is corrupted, then nodes in $\boldsymbol{A}$ see all shares in $[y; f_y]_T$. However, all these shares just can reconstruct the final output of y, while no useful information of G_M will be leaked. ■

B. Scheme for Handling Low-Quality Data

As mentioned in Section II-F, for achieving the FL training, it is required to average the gradients uploaded from users. However, in real-world training process, users may upload some low-quality gradients due to unstable equipment or irregular operations. Recent research results [26], [27] have shown that the low-quality gradients may differ far from the collaboratively generated gradient which determines the convergence trend. As a result, the components of these gradient vectors may perform with opposite signs or biased values, thus negatively impacting the model convergence, decreasing the convergence rate, and even causing divergence. Therefore, it is essential to handle these low-quality-data users. To do so, our first intuition is to remove these components of inverse signs (named as contradictory components) and find an optimal aggregation mechanism (not just averaging) for the remaining components. Therefore, we propose a novel method combining "Removing Contradictory Components (RCC)" and "Weighted Aggregation (WA)," introduced below.

1) Removing Contradictory Components: In our settings, we consider that M users participate in the FL training. Through pre-training the DNN model, the weight of the model is initialized as $\omega_{(0)} = (\omega^1_{(0)}, \ldots, \omega^l_{(0)}, \ldots, \omega^L_{(0)})$ with L dimensions, accordingly, the initial global gradient is obtained as $g_{(0)} = (g^1_{(0)}, \ldots, g^l_{(0)}, \ldots, g^L_{(0)})$. In each j-th iteration, each user m trains their local model to obtain the local gradient $g_{m,(j)} = (g^1_{m,(j)}, \ldots, g^l_{m,(j)}, \ldots, g^L_{m,(j)})$, which will be shared for calculating current global gradient $g_{(j)} = (g^1_{(j)}, \ldots, g^l_{(j)}, \ldots, g^L_{(j)})$ to further update the model. Besides, we consider that only a small part of users may upload low-quality gradients.

Algorithm 1: Removing Contradictory Components.

Input: User m's gradient $g_m=(g_m^1,\ldots,g_m^l,\ldots,g_m^L)$, Global gradient $g=(g^1,\ldots,g^l,\ldots,g^L)$.

1: Initialize global gradient $g_{(0)}=(g_{(0)}^1,\ldots,g_{(0)}^l,\ldots,g_{(0)}^L)$;
2: In j-th iteration, given previous global gradient $g_{(j-1)}$, a initialized counter $C_{m,(j)}=0$, each user m does below;
3: **for** each $g_{m,(j)}^l \in g_{m,(j)}$ **do**
4: **if** Sign$(g_{m,(j)}^l) \neq$ Sign$(g_{(j-1)}^l)$ **then**
5: $g_{m,(j)}^l \leftarrow null$; ▷ Remove contradictory components
6: $C_{m,(j)}=C_{m,(j)}+1$;
7: **end if**
8: **end for**
9: **if** $\frac{C_{m,(j)}}{L} > v_{(j)}$ **then**
10: $g_{m,(j)} \leftarrow null$; ▷ Remove the whole gradient
11: **end if**

As discussed above, the gradient is a vector in nature, whose direction is jointly determined by the two factors of sign and value of the components constituting the gradient. In one dimension of the gradient, the sign of the component determines the direction (increase or decrease) which the model should be improved along. Indeed, in each j-th iteration, through comparing local gradient $g_{m,(j)}$ with current global gradient $g_{(j)}$, each user can identify the components which have the opposite signs. However, the comparing process can just be achieved after all the local gradients have been uploaded and aggregated, thus causing redundant works, being not efficient. Recent work [28] has found that there exists only a small discrepancy between the gradients of two sequential iterations. Therefore, it is reasonable for each user to compare their current local gradients with previous global gradient for achieving our goals. More details of this method are presented through the pseudo-code in Algorithm 1.

Remark: Each user will locally execute above comparison processes, sharing no private information to fog nodes, so that here no privacy issue should be considered. Besides, the contradictory components labelled $null$, and the labels will be shared to fog nodes for guaranteeing the robustness of the gradient aggregation, since in our framework, the gradient aggregation will not be achieved until all participating users have shared their gradient information. Additionally, if the proportion of the components with opposite signs is larger than a tuned threshold $v_{(j)}$, the whole gradient $g_{m,(j)}$ of user m will be excluded in this iteration. Furthermore, since we consider that only a small part of low-quality gradients exist in the practical training process, the convergence of the model can still be guaranteed, even if some contradictory components are removed. This result can also be clarified in our experiments in Section. V.

2) Weighted Aggregation: In this part, we first introduce how to update the weight of each component, then we present the method for weighted aggregation. Considering each l-th component of global gradient g^l, each user's weight of the l-th component will be updated as follows:

$$\mathcal{W}_m^l = log(\frac{\sum_{m=1}^{M} dis(g_m^l, g^l)}{dis(g_m^l, g^l)}) \tag{5}$$

where $dis(\cdot)$ is a function for measuring the distance between g_m^l and g^l. Here we utilize a squared distance function [29] as follows.

$$dis(g_m^l, g^l) = (g_m^l - g^l)^2 \tag{6}$$

Then given each user's weight $\mathcal{R}_m^l$, each l-th component of the global gradient is updated as follows:

$$g^l = \frac{\sum_{m=1}^{m=M} \mathcal{W}_m^l g_m^l}{\sum_{m=1}^{m=M} \mathcal{W}_m^l} \tag{7}$$

Remark: Utilizing (6) to measure the weight requires that g_m^l and g^l have the same sign, thus guaranteeing the convergence of the aggregation process. To do so, we can remove the components with opposite signs based on Algorithm 1 before executing the weighted aggregation. Besides, as shown in (6), the closer between the local component and the global component, the higher weight of the local component is set. This is reasonable. Recent work [27] has found that if two gradients are generated from two datasets with similar quality, the value discrepancy between the components in these two gradients will be quite small. Additionally, in this paper, the component g_m^l satisfying $\frac{\mathcal{W}_m^l - \min_{j\in[1,M]} \mathcal{W}_j^l}{\max_{s\in[1,M]} \mathcal{W}_s^l - \min_{j\in[1,M]} \mathcal{W}_j^l} < \theta$ will be considered low-quality, where θ is a threshold value negotiated by each user. This setting was also applied in works [27], [28]. Furthermore, for achieving the FL training while handling low-quality-data users, during the training process, we will iteratively execute " RCC" and "WA" until the optimal global gradient is obtained.

Nextly we introduce how to apply our SCEP to protect all users' related information (including each user's gradient and aggregation weight values) in the above training process.

C. Construction of GALAXY

We now describe the workflow of our GALAXY. As described in Fig. 5, it contains three phases: initialization, gradient generation and sharing, and model update.

1) Initialization: In this phase, we establish an initialization for our MPC-based secure environment and the DNN model. We first construct an MPC-based framework, which allows all the computations to be executed in a finite field $\mathbb{F}=\mathbb{Z}_p$, where p is a prime. Then we initialize the (T, N) parameter for the MPC algorithm, where N denotes the total number of fog nodes, and T is the threshold value. For initializing the DNN model, we consider the global weight vector ω in L dimensions, and then we pre-train the DNN model, initializing ω as $\omega_{(0)}=(\omega_{(0)}^1,\ldots,\omega_{(0)}^l,\ldots,\omega_{(0)}^L)$ and generating the global gradient initialized as $g_{(0)}=(g_{(0)}^1,\ldots,g_{(0)}^l,\ldots,g_{(0)}^L)$.

2) Gradient Generation and Sharing: In j-th iteration, each user m will first train their local DNN model with local dataset to generate a local gradient. Then each user m utilizes the quantization method [30] to transfer the local gradient to an integer vector, denoted as $g_{m,(j)}=(g_{m,(j)}^1,\ldots,g_{m,(j)}^l,\ldots,g_{m,(j)}^L)$.

Implementation of GALAXY

Initialization: For constructing the computation environment, we first set a finite field $\mathbb{F} = \mathbb{Z}_p$, where p is a prime. Then we initialize the (T, N) parameter for the MPC algorithm, where N denotes the total number of fog nodes, T is the threshold value. Consider the global weight vector in L dimensions, we pre-train the DNN model, obtaining initialized $\omega_{(0)}$=$(\omega^1_{(0)}, ..., \omega^l_{(0)}, ..., \omega^L_{(0)})$ and $g_{(0)}$=$(g^1_{(0)}, ..., g^l_{(0)}, ..., g^L_{(0)})$.

Gradient generation and sharing:

In j-th iteration.

$\underline{user_m}$:

1. Each user m trains on their own local data and obtain the local gradient quantized as $g_{m,(j)} = (g^1_{m,(j)}, ..., g^l_{m,(j)}, ..., g^L_{m,(j)})$.
2. Each user m randomly generates a polynomial $f_{g_{m,(j)}}$ for $g_{m,(j)}$ at degree $T-1$, taking $g_{m,(j)}$ as the secret, i.e., the constant term.
3. Each user m generates the secret set $[g_{m,(j)}, f_{g_{m,(j)}}]_T$ based on the generated polynomial $f_{g_{m,(j)}}$, and respectively sends each $[g_{m,(j)}]^n_T = f_{g_{m,(j)}}(n)$ to each node P_n.

Model update:

$\underline{user_m}$:

4. Each user m locally removes the components with opposite signs based on Alg. 1.
5. For these remaining components, each user m locally calculates $D^l_{m,(j)} = dis(g^l_{m,(j)}, g^l_{(j-1)})$, generates the secret share set $[D^l_{m,(j)}, f_{D^l_{m,(j)}}]_T$ for the polynomial $f_{D^l_{m,(j)}}$, and then sends each $[D^l_{m,(j)}]^n_T = f_{D^l_{m,(j)}}(n)$ to each node P_n.
6. Each user m locally calculates $logD^l_{m,(j)}$, and generates the secret share set $[logD^l_{m,(j)}, f_{logD^l_{m,(j)}}]_T$ for the polynomial $f_{logD^l_{m,(j)}}$, and then sends each $[logD^l_{m,(j)}]^n_T = f_{logD^l_{m,(j)}}(n)$ to each node P_n.

$\underline{\text{Node } P_n}$:

7. Each fog node P_n calculates $[sum^l_{dis,(j)}]^n_T = \sum_{m=1}^M [D^l_{m,(j)}]^n_T$, and sends each $[sum^l_{dis,(j)}]^n_T$ to P_i.

$\underline{\text{Node } P_i}$:

8. P_i randomly selects T shares in $[sum^l_{dis,(j)}]_T$ to reconstruct $sum^l_{dis,(j)}$ based on Lagrange interpretation.
9. Then P_i calculates $logsum^l_{dis,(j)}$ and generates the secret share set $[logsum^l_{dis,(j)}]_T$, and sends each labelled share $[logsum^l_{dis,(j)}]^n_T$ to each node P_n.

$\underline{\text{Node } P_n}$:

10. Each fog node P_n calculates $[\mathcal{W}^l_{m,(j)}]^n_T = [logsum^l_{dis,(j)}]^n_T - [logD^l_{m,(j)}]^n_T$ for each user m.
11. Each fog node P_n calculates $[\mathcal{W}^l_{m,(j)} * g^l_{m,(j)}]^n_T$ through the multiplication gates in SCEP, and obtains $\sum_{m=1}^M [\mathcal{W}^l_{m,(j)} * g^l_{m,(j)}]^n_T$ and $\sum_{m=1}^M [\mathcal{W}^l_{m,(j)}]^n_T$ through the addition gates.
12. Each fog node P_n sends $\sum_{m=1}^M [\mathcal{W}^l_{m,(j)} * g^l_{m,(j)}]^n_T$ and $\sum_{m=1}^M [\mathcal{W}^l_{m,(j)}]^n_T$ to P_i.

$\underline{\text{Node } P_i}$:

13. P_i reconstructs the values of $\sum_{m=1}^{m=M} \mathcal{W}^l_{m,(j)} * g^l_{m,(j)}$ and $\sum_{m=1}^{m=M} \mathcal{W}^l_{m,(j)}$, and obtains the l-th global gradient component $g^l_{(j)} = \frac{\sum_{m=1}^{m=M} \mathcal{W}^l_{m,(j)} g^l_{m,(j)}}{\sum_{m=1}^{m=M} \mathcal{W}^l_{m,(j)}}$.

$\underline{P_n, P_i, user_m}$:

14. Each P_n, P_i, and each $user_m$ execute step 1∼13 iteratively until the optimal global gradient $g_{(j)}$ is obtained.
15. P_i calculates ω_{j+1}=$\omega_j - \alpha g_{(j)}$ based on Eqn. (4), and broadcasts the global gradient $g_{(j)}$ and weight ω_{j+1} to all other fog nodes and each user m.

$\underline{P_n, user_m}$:

16. Each P_n and $user_m$ execute step 1∼15 iteratively until the minimum of the loss function is obtained.

Fig. 5. Workflow of GALAXY.

Next each user m generates the shares of g_m designated for each fog nodes P_n, where $n \in [1, N]$, via (T, N)-threshold Shamirs secret sharing. This can protect each user's private $g_{m,(j)}$ against fog nodes, even at most T fog nodes collude with each other. Specifically, each user m randomly generates a polynomial $f_{g_{m,(j)}}(x) = g_{m,(j)} + a^1_{m,(j)} x + \cdots + a^t_{m,(j)} x^t + \cdots + a^{T-1}_{m,(j)} x^{T-1}$, where each $a^t_{m,(j)}$ is randomly chosen from $\mathbb{Z}_p$. Based on the polynomial, each user m creates a share group $[g_{m,(j)}; f_{g_{m,(j)}}(x)]_T$, and sends each share $[g_{m,(j)}; f_{g_{m,(j)}}(x)]^n_T = f_{g_{m,(j)}}(n)$ to each fog node P_n.

3) Model Update: For accomplishing the model update, we will iteratively remove contradictory components and execute the private weighted gradient aggregation. The details are presented as follows. First of all, each user m locally removes the components with opposite signs based on Algorithm 1. This process will be executed locally by each user, so that here no privacy issue should be considered.

Then we introduce how to privately achieve the weighted gradient aggregation for the remaining components. For each l-th component, each user m first locally calculates $D^l_{m,(j)} = dis(g^l_{m,(j)}, g^l_{(j-1)})$, and generates the secret share set $[D^l_{m,(j)}, f_{D^l_{m,(j)}}]_T$ for the polynomial $f_{D^l_{m,(j)}}$, and then sends each $[D^l_{m,(j)\frac{a}{b}}]^n_T = f_{D^l_{m,(j)}}(n)$ to each node P_n. Next, each user m locally calculates $logD^l_{m,(j)}$, and generates the secret share set $[logD^l_{m,(j)}, f_{logD^l_{m,(j)}}]_T$ for the polynomial $f_{logD^l_{m,(j)}}$, sending each share of $[logD^l_{m,(j)}]^n_T = f_{logD^l_{m,(j)}}(n)$ to each node P_n. After achieving the sharing task, users will be freed intermittently until receiving the global parameters from fog nodes to start next iteration.

When receiving all the shares uploaded by users, all fog nodes calculate $[sum^l_{dis,(j)}, f_{sum^l_{dis,(j)}}]_T = \sum_{m=1}^M [D^l_{m,(j)}]_T$, and each P_n sends each $[sum^l_{dis,(j)}]^n_T = f_{sum^l_{dis,(j)}}(n)$ to P_i.

Then P_i reconstructs $sum^l_{dis,(j)}$ based on Lagrange interpretation. Next, P_i calculates $logsum^l_{dis,(j)}$, generates the secret share set $[logsum^l_{dis,(j)}]_T$, and sends each labelled share $[logsum^l_{dis,(j)}]^n_T$ to each node P_n. After receiving the share of $[logsum^l_{dis,(j)}]^n_T$, each node P_n calculates $[\mathcal{W}^l_{m,(j)}]^n_T = [logsum^l_{dis,(j)}]^n_T - [logD^l_{m,(j)}]^n_T$ for each user m. Then each fog node P_n calculates $[\mathcal{W}^l_{m,(j)} * g^l_{m,(j)}]^n_T$ through the multiplication gates in SCEP, and obtain $\sum_{m=1}^{M}[\mathcal{W}^l_{m,(j)} * g^l_{m,(j)}]^n_T$ and $\sum_{m=1}^{M}[\mathcal{W}^l_{m,(j)}]^n_T$ through the addition gates. Next each fog node P_n sends $\sum_{m=1}^{M}[\mathcal{W}^l_{m,(j)} * g^l_{m,(j)}]^n_T$ and $\sum_{m=1}^{M}[\mathcal{W}^l_{m,(j)}]^n_T$ to P_i. After receiving these shares from all fog nodes, based on Lagrange interpretation, P_i reconstructs the value of $\sum_{m=1}^{m=M}\mathcal{W}^l_{m,(j)} * g^l_{m,(j)}$ and $\sum_{m=1}^{m=M}\mathcal{W}^l_{m,(j)}$, and obtains the l-th global gradient component $g^l_{(j)} = \frac{\sum_{m=1}^{m=M}\mathcal{W}^l_{m,(j)}g^l_{m,(j)}}{\sum_{m=1}^{m=M}\mathcal{W}^l_{m,(j)}}$. P_n and $user_m$ iteratively remove contradictory components and privately execute the weighted gradient aggregation until the optimal global gradient is obtained.

After obtaining all the global components of $g_{(j)}$, P_i calculates $\omega_{j+1}=\omega_j - \alpha g_{(j)}$ based on (4), and broadcasts the current global gradient and weight of the DNN model to all other fog nodes and each user m. Above processes will be executed iteratively until the loss function reaches the minimum.

Remarks: During the above processes, users are only required to upload secret shares, and other computations are outsourced to fog nodes. Therefore, after achieving the sharing task, users can be off-line until next iteration starts, which can minimize the overhead of edge users. Besides, based on the (T, N)-threshold property of our framework, even if at most $T-1$ fog nodes collude with each other, each user's privacy will still be guaranteed. Meanwhile, even if at most $N-T$ fog nodes simultaneously drop out during the aggregation process, the final aggregation result can still be reconstructed by the remaining fog nodes.

IV. Privacy Guarantees

As mentioned in Section II-B, the privacy threats are mainly derived from the fog nodes. Thereby, our goal of GALAXY is to guarantee all user-related information, including gradient and aggregation weight, from being disclosed to any fog node. Intuitively, in our protocol, each user m will just upload the shares of gradients, e.g., $[g_m, f_{g_m}]_T$, while each fog node P_n will implement a series of operations on these shares until a final result is obtained. Through the utilization of our SCEP, users' privacy will be protected from semi-honest fog nodes, even if at most $T-1$ of them collude with each other. We present the formal proof based on UC model as follows.

Proposition 1: (Against semi-honest fog nodes) Considering a corrupted node group $\boldsymbol{A} \subset \{P_1, \ldots P_N\}$, where $|\boldsymbol{A}| < T < N/2$, there exists a simulator $\mathbf{SIM}$ with unlimited computing power such that for the given security parameter p, threshold value T, and total number of fog nodes N, in the execution of our Galaxy, the view of $\boldsymbol{A}$ in the simulator $\mathbf{SIM}^{p,T,N}_{\boldsymbol{A}}$ is indistinguishable from view of $\boldsymbol{A}$ in the real protocol $\mathbf{REAL}^{p,T,N}_{\boldsymbol{A}}$ in an information-theoretic sense, even if fog nodes $\in \boldsymbol{A}$ collude with each other:

$$\mathbf{SIM}^{p,T,N}_{\boldsymbol{A}} \equiv \mathbf{REAL}^{p,T,N}_{\boldsymbol{A}} \tag{8}$$

Proof: We define such a simulator $\mathbf{SIM}$, which takes input of the gradient from users randomly assigned as $G'_M=(g'_1, \ldots, g'_m, \ldots, g'_M)$ instead of the input of real protocol $\mathbf{REAL}$ $G_M=(g_1, \ldots, g_m, \ldots, g_M)$, rigorously, both of G'_M and G_M belong to $\mathbb{Z}_p$, holding the same distribution. It runs our Galaxy as same as the real protocol. Then we prove indistinguishability between $\mathbf{SIM}^{p,T,N}_{\boldsymbol{A}}$ and $\mathbf{REAL}^{p,T,N}_{\boldsymbol{A}}$ in a hybrid argument [31], [32], as follows.

$\mathbf{hyb_1}$ In this hybrid, each user m uploads the share of $[g_{m,(j)}]^n_T$ to each fog node P_n in real protocol, while in the simulator each fog node takes the share of $[g'_{m,(j)}]^n_T$. Based on the property of Shamir's secret sharing, $[g_{m,(j)}]^n_T$ and $[g'_{m,(j)}]^n_T$ have the same distribution, which can further prove that $\mathbf{SIM}^{p,T,N}_{\boldsymbol{A}} \equiv \mathbf{REAL}^{p,T,N}_{\boldsymbol{A}}$.

$\mathbf{hyb_2}$ In this hybrid, each user m uploads the share of $[D^l_{m,(j)}]^n_T$ and $[logD^l_{m,(j)}]^n_T$ to each fog node P_n in real protocol, while in the simulator each fog node takes the share of $[D'^l_{m,(j)}]^n_T$ and $[logD'^l_{m,(j)}]^n_T$. Based on the property of Shamir's secret sharing, $[D^l_{m,(j)}]^n_T$ and $[D'^l_{m,(j)}]^n_T$ have the same distribution, while $[logD^l_{m,(j)}]^n_T$ and $[logD'^l_{m,(j)}]^n_T$ have the same distribution, such guaranteeing the indistinguishability $\mathbf{SIM}^{p,T,N}_{\boldsymbol{A}} \equiv \mathbf{REAL}^{p,T,N}_{\boldsymbol{A}}$.

$\mathbf{hyb_3}$ In this hybrid, each fog node P_n calculates $[sum^l_{dis,(j)}]^n_T$ in the real protocol, and calculates $[sum'^l_{dis,(j)}]^n_T$ in the simulator. $[sum^l_{dis,(j)}]^n_T$ and $[sum'^l_{dis,(j)}]^n_T$ have the same distribution, therefore, the indistinguishability $\mathbf{SIM}^{p,T,N}_{\boldsymbol{A}} \equiv \mathbf{REAL}^{p,T,N}_{\boldsymbol{A}}$ is guaranteed in this hybrid.

$\mathbf{hyb_4}$ In this hybrid, P_i reconstructs $sum^l_{dis,(j)}$ through the reconstruction function in our SCEP. Based on the security of the reconstruction function, the indistinguishability $\mathbf{SIM}^{p,T,N}_{\boldsymbol{A}} \equiv \mathbf{REAL}^{p,T,N}_{\boldsymbol{A}}$ is guaranteed in this hybrid.

$\mathbf{hyb_5}$ In this hybrid, P_i sends share of $[logsum^l_{dis,(j)}]^n_T$ to each fog node P_n in real protocol, while in the simulator each fog node takes the share of $[logsum'^l_{dis,(j)}]^n_T$. Based on the property of Shamir's secret sharing, $[logsum^l_{dis,(j)}]^n_T$ and $[logsum'^l_{dis,(j)}]^n_T$ have the same distribution. Therefore, the indistinguishability $\mathbf{SIM}^{p,T,N}_{\boldsymbol{A}} \equiv \mathbf{REAL}^{p,T,N}_{\boldsymbol{A}}$ is guaranteed in this hybrid.

$\mathbf{hyb_6}$ In this hybrid, each fog node P_n calculates $[\mathcal{W}^l_{m,(j)}]^n_T$ in the real protocol, and calculates $[\mathcal{W}'^l_{m,(j)}]^n_T$ in the simulator. Based on the security of addition and multiply-by-constant gates, $[\mathcal{W}^l_{m,(j)}]^n_T$ and $[\mathcal{W}'^l_{m,(j)}]^n_T$ have the same distribution. Therefore, the indistinguishability $\mathbf{SIM}^{p,T,N}_{\boldsymbol{A}} \equiv \mathbf{REAL}^{p,T,N}_{\boldsymbol{A}}$ is guaranteed in this hybrid.

$\mathbf{hyb_7}$ In this hybrid, each fog node P_n calculates $[\mathcal{W}^l_{m,(j)} * g^l_{m,(j)}]^n_T$ in the real protocol, and calculates $[\mathcal{W}'^l_{m,(j)} *$

TABLE I
SYSTEM PARAMETERS

Parameter	Value Setting	Description of Parameter
p	$2^{26}-5$	Prime for the finite field $\mathbb{Z}_p$
α	0.01	Learning rate
N	10	Total number of fog nodes
T	4	Threshold value

$g'^{l}_{m,(j)}]^n_T$ in the simulator. Based on the security of our multiplication gates, $[\mathcal{W}^l_{m,(j)} * g^l_{m,(j)}]^n_T$ and $[\mathcal{W}'^{l}_{m,(j)} * g'^{l}_{m,(j)}]^n_T$ have the same distribution. Therefore, the indistinguishability $\mathbf{SIM}^{p,T,N}_{\boldsymbol{A}} \equiv \mathbf{REAL}^{p,T,N}_{\boldsymbol{A}}$ is guaranteed in this hybrid.

$\mathbf{hyb_8}$ In this hybrid, each fog node P_n calculates $\sum_{m=1}^{M}[\mathcal{W}^l_{m,(j)} * g^l_{m,(j)}]^n_T$ and $\sum_{m=1}^{M}[\mathcal{W}^l_{m,(j)}]^n_T$ through the addition gates in the real protocol, and calculates $\sum_{m=1}^{M}[\mathcal{W}'^{l}_{m,(j)} * g'^{l}_{m,(j)}]^n_T$ and $\sum_{m=1}^{M}[\mathcal{W}'^{l}_{m,(j)}]^n_T$ in the simulator. Based on the security of our addition gates, the indistinguishability $\mathbf{SIM}^{p,T,N}_{\boldsymbol{A}} \equiv \mathbf{REAL}^{p,T,N}_{\boldsymbol{A}}$ is guaranteed in this hybrid.

$\mathbf{hyb_9}$ In this hybrid, P_i reconstructs the values of $\sum_{m=1}^{m=M} \mathcal{W}^l_{m,(j)} * g^l_{m,(j)}$ and $\sum_{m=1}^{m=M} \mathcal{W}^l_{m,(j)}$. Based on the security of our reconstruction function, the indistinguishability $\mathbf{SIM}^{p,T,N}_{\boldsymbol{A}} \equiv \mathbf{REAL}^{p,T,N}_{\boldsymbol{A}}$ is guaranteed in this hybrid.

As discussed above, based on the security of our SCEP, we prove that there exist a simulator $\mathbf{SIM}$ with unlimited computing power such that in the execution of our GALAXY, the view of $\boldsymbol{A}$ in the simulator $\mathbf{SIM}^{p,T,N}_{A}$ is indistinguishable from view of $\boldsymbol{A}$ in the real protocol $\mathbf{REAL}^{p,T,N}_{A}$ in an information-theoretic sense. That means users' gradient and gradient aggregation weight plaintexts will not be exposed to fog nodes in the execution of our protocol. ■

V. PERFORMANCE EVALUATION

In this section, we introduce how to construct experiments to evaluate the performance of our GALAXY, in terms of functionality, model accuracy, and resource overhead.

A. Experiment Setup

We set the main system parameters as shown in Table I, and other experiment settings are presented as follows. We implement our experiments in a Java environment, where we conduct a platform for supporting (T, N)-Shamir secret sharing and Lagrange interpolation. We build a decentralized computing environment, simulating fog nodes on 5 Ubuntu servers with Intel(R) Xeon(R) running at 2.10 GHz on 6 cores and 32 GB RAM, simulating each edge user by Huawei nova3 running at 2.36 GHz on 4 cores and 6 GB RAM, and achieving the communication between participants through TCP/IP with secure channels based on transport layer security (TLS) protocol [33]. Additionally, to achieve the federated learning, we select the raw data from the MNIST [1] dataset, which has 60,000 training and 10,000 testing examples, and we conduct the DNN model with 5 layers including 2 convolutional layers, 1 average pooling layer, and 2 fully connected layers.

[1] http://yann.lecun.com/exdb/mnist/

B. Functionality

For specifying the functionality advantages of our GALAXY, we make a comparison with four state-of-the-art works focusing on privacy-preserving FL, i.e., SecProbe [15], PPFDL [16], Secureml [6], and FALCON [10].

As shown in Table II, all the works can protect the privacy of users' gradients in a semi-honest threat model. SecProbe [15] is the first work, which can resist low-quality-data users while guaranteeing the confidentiality of users' gradients. However, SecProbe cannot protect the privacy of the aggregation results, nor can it allow users or the server to drop out in the training process. Following SecProbe [15], PPFDL [16] can handle low-quality-data users while supporting users dropping out during the training process. However, PPFDL requires two servers to cooperatively establish the secure scheme, such that if any one of these two servers drops out, the training process will be suspended. Meanwhile, PPFDL permits no collusion between these two servers. Secureml [6] respectively utilized technologies of oblivious transfer (OT) and linearly homomorphic encryption (LHE) to construct their framework of secure 2-party computation (2-PC). Although their scheme can be robust to users being off-line during the training process, it cannot support any server dropping out, nor can it permit these 2 servers to collude with each other. FALCON [10] proposed a hybrid scheme based on secure 3-party computation (3-PC). Nevertheless, in their scheme, at least one server should be considered trusted, and any two servers cannot collude with each other, while no server is allowed drop out during the training process. Additionally, since both of Secureml [6] and FALCON [10] focus on privacy protection for original FL, the case of handling low-quality-data users is out of the scope of their work.

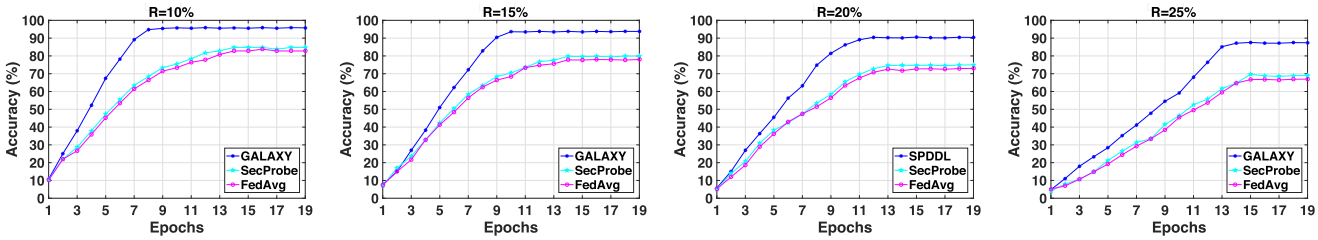
In contrast to above works, our GALAXY utilizes Shamir's secret sharing and Lagrange interpolation, combined with arithmetic circuit, to conduct the secure multi-party computation framework, based on which, not only users' gradient information can be protected, but also the confidentiality of users' gradient aggregation weights can be guaranteed. Meanwhile, the (T, N)-threshold property allows our GALAXY to resist the collusion among up to $T-1$ fog nodes, while supporting at most $N-T$ fog nodes simultaneously being off-line. Besides, similar to PPFDL [16], our GALAXY can also alleviate the negative impact of low-quality-data users.

C. Accuracy

In this section, we discuss the model accuracy of our GALAXY. We implement the training task securely with a series of epochs (number of iterations), and observe the testing results (accuracy) after each epoch. Besides, two representative works [15], [34] are compared with our GALAXY. One is FedAvg [34], an original federated learning, averages shared gradients to achieve the training task, without any consideration of handling low-quality data, implemented in a plaintext mode. Another one, SecProbe [15], is the first work of privacy-preserving FL with

TABLE II
FUNCTIONALITY COMPARISON

Scheme \ Function	Protection of gradient privacy	Protection of aggregation results	Resistance to servers' collusion	Robustness to servers dropping out	Robustness to users dropping out	Resistance to low-quality data	Server setting	Adversary model
SecProbe [15]	✔	✗	–	✗	✗	✔	**Single-Server**	**Semi-honest**
PPFDL[16]	✔	✔	✗	✗	✔	✔	**Dual-Server**	**Semi-honest**
Secureml [6]	✔	✔	✗	✗	✔	✗	**Dual-Server**	**Semi-honest**
FALCON [10]	✔	✔	✗	✗	✔	✗	**Triple-Server**	**Semi-honest**
GALAXY	✔	✔	✔	✔	✔	✔	**Multi-Server**	**Semi-honest**



(a) (b) (c) (d)

Fig. 6. Comparison of accuracy with different number of epochs.

low-quality-data users, utilizing differential privacy to protect users' privacy. This comparison is quite meaningful. Based on the comparison between FedAvg and our GALAXY, we can measure the effect on the accuracy caused by our method of handling low-quality-data users. Through comparing our GALAXY with SecProbe, our advantage (better trade-off between privacy and accuray) over the DP-based method can be demonstrated.

Considering the practical training process, we implement our experiments over different proportions (R) of low-quality-data users. For simulating these users, we randomly select R of the participating users, adding random noise $(0-1)$ to their data to simulate the low-quality data. To demonstrate the robustness to low-quality-data users in our scheme, we change R from 10% to 25%, with interval of 5%, and observe the results of accuracy in different R, which is also utilized in FedAvg [34] and SecProbe [15] for the comparison. Besides, for implementing SecProbe in our experiments, we set privacy budget ϵ as 10 for its technology of differential privacy, which is as same as the setting in SecProbe [15].

As shown in Fig. 6, all the subgraphs present us the accuracy will increase with the increasing epochs, and gradually reach a max value after a certain epoch. It can also be observed that the accuracy of our GALAXY is always much higher than FedAvg [34] and SecProbe [15] in each subgraph (R=10%, 15%, 20%, 25%), finally reaching up to 95.73%, 93.77%, 90.36%, and 87.38% respectively. Besides, it is shown that at the point of the same accuracy value, less epochs are required in our GALAXY compared with FedAvg and SecProbe. This means our GALAXY has a higher convergence rate in the training process. Additionally, we can observe that the accuracy in our GALAXY just has a little decrease when R reaches 25%, however it decreases greatly in FedAvg and SecProbe. We can attribute above results into two main factors: (i) In the FedAvg [34], simply utilizing the method of averaging gradients nearly cannot mitigate the negative impact caused by low-quality-data users, and (ii) SecProbe, applying the technology of differential privacy to protect users' privacy, cannot obtain abundant benefit of accuracy, even if the negative impact of low-quality-data users can be mitigated to a certain degree.

In contrast, our GALAXY utilizing MPC technology to protect all user-related information hardly decreases the accuracy. Meanwhile, based on the hybrid method of handling low-quality-data users, our GALAXY can further obtain an improvement on accuracy. All of these factors benefit our GALAXY with a superior performance in terms of accuracy.

D. Computation Overhead

In this section, we present both the complexity analysis and experiment results for specifying the computation overhead of our scheme. Besides, we make a comparison with an MPC-based scheme PPFL [32], which also utilizes Shamir' secret sharing as one of the technologies for constructing their framework.

1) Complexity Analysis: Considering M participating users, N participating fog nodes, K gradient components per user, we conduct the complexity analysis for each user m, each fog node P_n, and the fog node P_i, as follows.

User m: $\mathrm{O(KN^2 + K)}$. Each $user_m$'s computation overhead can be broken up into 3 main parts: (i) generating $[g_{m,(j)}]_T^n$ for each P_n, which takes $\mathrm{O(KN^2)}$, (ii) generating $[D^l_{m,(j)}]_T^n$ and $[logD^l_{m,(j)}]_T^n$ for each P_n, which takes $\mathrm{O(KN^2)}$, and (iii) locally calculating $D^l_{m,(j)} = dis(g^l_{m,(j)}, g^l_{(j-1)})$ and $logD^l_{m,(j)}$, which takes $\mathrm{O(K)}$. Overall, each $user_m$ costs $\mathrm{O(KN^2 + K)}$.

Node $\mathbf{P_n}$($\mathrm{n \neq i}$): $\mathrm{O(KMN^2 + KM)}$. Each P_n's computation overhead can be broken up into 4 main parts: (i) calculating $[sum^l_{dis,(j)}]_T^n = \sum_{m=1}^{M}[D^l_{m,(j)}]_T^n$, which takes $\mathrm{O(KM)}$, (ii) calculating $[\mathcal{W}^l_{m,(j)}]_T^n$, which takes $\mathrm{O(KM)}$, (iii) calculating

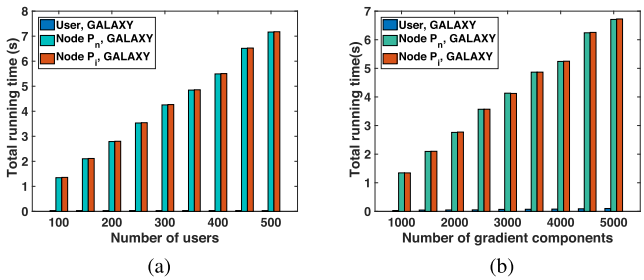


Fig. 7. Computation overhead of GALAXY.

$[\mathcal{W}^l_{m,(j)} * g^l_{m,(j)}]^n_T$, which takes $\mathrm{O(KMN^2 + KM)}$, and (iv) calculating $\sum_{m=1}^{M}[\mathcal{W}^l_{m,(j)} * g^l_{m,(j)}]^n_T$ and $\sum_{m=1}^{M}[\mathcal{W}^l_{m,(j)}]^n_T$, which takes $\mathrm{O(KM)}$. Overall, the computation complexity of each P_n is $\mathrm{O(KMN^2 + KM)}$.

Node $\mathbf{P_i}$: $\mathrm{O(KMN^2 + KM + KN^2 + K)}$. In addition to achieving the task undertaken by P_n, P_i is also responsible for completing the reconstruction. Therefore, P_i will take additional cost as follows: (i) reconstructing $sum^l_{dis,(j)}$, which takes $\mathrm{O(K)}$, (ii) calculating $[logsum^l_{dis,(j)}]_T$, which takes $\mathrm{O(KN^2)}$, and (iii) reconstructing $\sum_{m=1}^{m=M}\mathcal{W}^l_{m,(j)} * g^l_{m,(j)}$ and $\sum_{m=1}^{m=M}\mathcal{W}^l_{m,(j)}$, which takes $\mathrm{O(KM)}$. Overall, P_i's computation complexity takes $\mathrm{O(KMN^2 + KM + KN^2 + K)}$.

2) Experiment Results: As shown in Fig. 7, the computation overhead of each user increases linearly with the increasing number of gradient components per user, but stays the same with the increasing number of users. That means the increase of users will not bring additional overhead to each single user. Besides, it can be observed that the overhead of each user is far less than the overhead of fog nodes, being up to 0.028 s and 0.101 s respectively in Fig. 7(a) and (b). This result is reasonable, since our scheme outsources most of the computation tasks to fog nodes, thus mitigating the computation pressure on user side. This is very friendly for the edge users with limited hardware resources in IoV. Additionally, since P_i is responsible for achieving the reconstruction missions, the computation overhead of P_i is a little higher than $P_n (n \neq i)$.

As shown in Fig. 8, we compare the computation overhead between GALAXY and PPFL [32]. The result shows that, for both user side and server side, our scheme can perform much better than PPFL in terms of computation overhead, especially on user side. The reasons fall into 2 main factors: (i) the complexity of PPFL takes $\mathrm{O(M^2 + M \cdot K)}$ and $\mathrm{O(K \cdot M^2)}$ respectively on user side and server side, which are quadratic functions of the number of users, while our complexity is just linear with the number of users on server side and has no relation with the number of users on user side. (ii) although PPFL also utilizes Shamir' secret sharing as one of the technologies for constructing their framework, their method requires much more computations for implementing their masking scheme in the single server setting. In contrast to PPFL, our GALAXY just requires simpler operations in a finite field. Obviously, our GALAXY can be more suitable for Internet of Vehicles, which consists of tens of thousands of edge users.

E. Communication Overhead

1) Complexity Analysis: In this part, we present the communication complexity analysis for each user m, each fog node P_n, and the fog node P_i, as follows.

User m: $\mathrm{O(KN)}$. The communication cost of each user $user_m$ falls into 3 parts: (i) sending each $[g_{m,(j)}]^n_T$ to each P_n, which takes $\mathrm{O(KN)}$, (ii) sending each $[D^l_{m,(j)}]^n_T$ to each P_n, which takes $\mathrm{O(KN)}$, and (iii)sending each $[logD^l_{m,(j)}]^n_T$ to each P_n, which takes $\mathrm{O(KN)}$. Overall, each $user_m$ costs $\mathrm{O(KN)}$.

Node $\mathbf{P_n}(\mathrm{n} \neq \mathrm{i})$: $\mathrm{O(KM + K)}$. The communication cost of each fog node P_n can be broken up into 4 parts: (i) receiving each $user_m$'s $[g_{m,(j)}]^n_T$, $[D^l_{m,(j)}]^n_T$, and $[logD^l_{m,(j)}]^n_T$, all of which take $\mathrm{O(KM)}$, (ii) sending $[sum^l_{dis,(j)}]^n_T$ to P_i, which takes $\mathrm{O(K)}$, (ii) receiving $[logsum^l_{dis,(j)}]^n_T$ from P_i, which takes $\mathrm{O(K)}$, and (iv) sending $\sum_{m=1}^{M}[\mathcal{W}^l_{m,(j)} * g^l_{m,(j)}]^n_T$ and $\sum_{m=1}^{M}[\mathcal{W}^l_{m,(j)}]^n_T$ to P_i, which takes $\mathrm{O(KM)}$. Overall, each $P_n (n \neq i)$ costs $\mathrm{O(KM + K)}$.

Node $\mathbf{P_i}$: $\mathrm{O(KMN + KN + KM + K)}$. As the reconstruction node, P_i takes additional communication cost as follows: (i) receiving $[sum^l_{dis,(j)}]^n_T$ from each P_n, which takes $\mathrm{O(KN)}$, (ii) sending $[logsum^l_{dis,(j)}]^n_T$ to each P_n, which takes $\mathrm{O(KN)}$, and (iii) receiving $\sum_{m=1}^{M}[\mathcal{W}^l_{m,(j)} * g^l_{m,(j)}]^n_T$ and $\sum_{m=1}^{M}[\mathcal{W}^l_{m,(j)}]^n_T$ from each P_n, which takes $\mathrm{O(KMN)}$. Overall, P_i's computation complexity takes $\mathrm{O(KMN + KN + KM + K)}$.

2) Experiment Results: As shown in Fig. 9, on user side, the communication overhead is much smaller compared with server side. The reason is that in each iteration, our scheme only requires each user to upload some shares to the fog nodes at the beginning and receive the final global parameters at the end, while more interactions are achieved among fog nodes for achieving the secure multi-party computation. Besides, on user side, the communication overhead will just increase with the increasing number of gradient components per user, but keep constant when number of users increases. That is to say, even if the number of users is extended, the communication overhead of a single user will not increase. This is a fantastic result, especially for IoV applications with enormous participating edge users.

As shown in Fig. 10, we also present some comparisons between GALAXY and PPFL [32] in terms of communication overhead. It is observed that on both user side and server side, GALAXY can outperform PPFL in terms of communication overhead. Especially, the communication overhead of PPFL is very sensitive to the number of users, which means that even a little increase in number of users will cause much increase of communication overhead. In contrast, the communication overhead of our scheme increases a little on server side and keeps constant on user side. The reason is explained as follows. First of all, in PPFL [32], each user is required to send both the encrypted gradients and secret keys to the cloud server, respectively for aggregation and for encryption. In contrast, our GALAXY just requires each user to upload some shares to the fog nodes. Besides, for guaranteeing the robustness to users dropping out during the execution of their protocol, PPFL will invoke their secret sharing protocol repeatedly to recover off-line

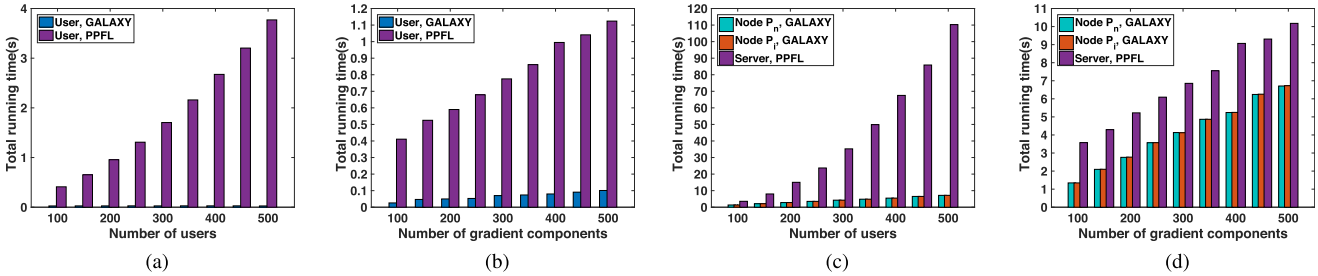


Fig. 8. Computation comparison between GALAXY and PPML. (a), (b) user side. (c), (d) server side.

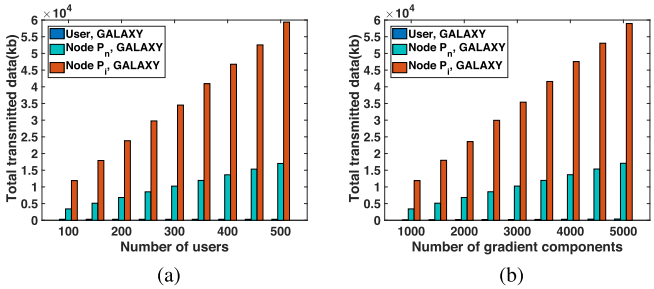


Fig. 9. Communication overhead of GALAXY.

users' shares, which will cause more additional communication overhead. However, our scheme requires no additional communication overhead, even if users are off-line during the training process. Similarly, on server side, the cloud server in PPFL will be responsible for receiving encrypted gradients as well as secret shares from users, and handling off-line users, all of which make the communication complexity of PPFL up to $\mathrm{O}(\mathrm{M}^2 + \mathrm{M} \cdot \mathrm{K})$, while the communication complexity of our scheme keeps linear with M. Overall, the above experiment results show that our GALAXY performs with a lower communication overhead than PPFL.

VI. RELATED WORK

For addressing the issue of privacy-preserving FL, many approaches have been proposed, spanning various technology domains. Meanwhile, considering the practicality of FL, how to handle low-quality-data users in the privacy-preserving FL has also attracted many attentions. In this section, we discuss both the methods of privacy-preserving FL without and with low-quality-data users as follows.

A. Conventional Privacy-Preserving FL

For addressing the conventional issue of privacy-preserving FL, existing methods are mainly derived from three technologies, i.e., differential privacy (DP), homomorphic encryption (HE), and secure multi-party computation (MPC). Recently, to address the issue with the technology of differential privacy, *Yu* et al. [35] proposed a concentrated differential privacy (CDP), which can outperform the traditional DP-based framework in terms of model accuracy. Besides, their scheme is also a universal DP method, which can track the cumulative privacy loss for FL. However, DP-based methods are required to balance privacy and accuracy, which means that the higher privacy is obtained the more accuracy is lost. Indeed, the work [4] specified that current DP-based methods rarely offer acceptable trade-off between privacy and accuracy for complex learning tasks. Besides, the research [2] has claimed that the approach for federated learning can be fundamentally destroyed even if all exchanged parameters are perturbed via DP-based technology. Therefore, utilizing current DP-based methods to protect users privacy is not suitable in this paper.

Based on HE technologies, *Phong* et al. [36] respectively exploited fully homomorphic encryption (FHE) and Linear homomorphic encryption (LHE) to construct their two privacy-preserving FL frameworks, while utilizing asynchronous SGD and secure channel to improve training efficiency and guarantee secure communication. To conclude, FHE, such as LWE-based homomorphic encryption, holding both multiplication and addition homomorphisms, can effectively achieve private federated learning. However, FHE would consume huge computation resources for achieving the FL training tasks. LHE, just like Paillier encryption, has smaller key sizes than FHE. Nevertheless, its security depends on enormous modular exponential operations in ciphertext domain, which is also computationally intensive. When it is utilized for protecting FL, the large size of ciphertext will cause an insurmountable overhead pressure for computing participants, especially for the setting with large-scale users and training samples. thus making it unpractical for real-world federated learning.

Through the utilization of MPC, *Keith Bonawitz* et al. [32] proposed a practical secure aggregation protocol for federated training in a single-server setting, where they combined secret sharing and masking technology to protect users' gradient information. Considering the sporadic instability of users, their scheme has the robustness to users being off-line during the training procedure. However, their solution requires enormous rounds of interactions. Especially, when some users are off-line during the process, their masking scheme needs to be reconstructed, thus causing huge communication overhead. More recently, several state-of-the-art researches employ MPC technologies to construct privacy-preserving schemes for machine learning, and operate their methods on two [5]–[7], three [8]–[10], or four [11], [12] servers. Such methods, nevertheless, limit the number of computing participants in fixed 2, 3, and 4. Besides, their schemes cannot defend against the collusion between servers, nor can they support servers dropping out during the training process. Therefore, these methods are not suitable in the practical setting of non-cloud-assisted fog computing.

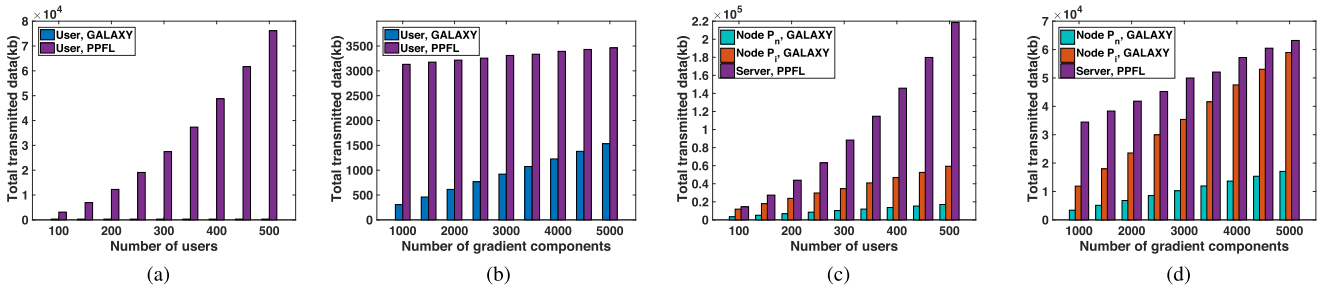


Fig. 10. Communication comparison between GALAXY and PPML. (a), (b) user side. (c), (d) server side.

B. Privacy-Preserving FL With Low-Quality-Data Users

For addressing the issue of privacy-preserving FL with low-quality-data users, two state-of-the-art works have been proposed. Specifically, *Zhao* et al. [15] proposed the first scheme, i.e., SecProbe. They utilized the technology of differential privacy to protect users' privacy, focusing on how to perturb the loss function during the model training process. Following *Zhao* et al.'s work, *Xu* et al. [16] proposed a novel scheme, named PPFDL, where they integrated technologies of garble circuit (GC) and oblivious transfer (OT) to construct a secure two-party computation (2-PC) cryptosystem. Besides, they created a method of $\texttt{Meth}_{\texttt{IU}}$ to mitigate the negative impact of low-quality-data users while protecting the $\texttt{Meth}_{\texttt{IU}}$ under their proposed secure framework.

However, the DP-based SecProbe [15] cannot supply enough privacy guarantee, if a high-level accuracy of the DNN model is required. Moreover, their scheme lacks the mechanism for protecting users' information aggregation weight, such that users' fairness cannot be guaranteed during the training process. Additionally, the MPC-based PPFDL [16] suffers from limited scalability, security, and functionality. That is, there is no permission of collusion between these two fixed cloud servers, and none of them is allowed to be off-line during the process. Therefore, both their schemes [15], [16] are unsuitable in the setting of this paper.

Different from the above two works, our GALAXY proposes a novel MPC-based framework for non-cloud-assisted fog computing through the masterly utilization of Shamir's secret sharing, Lagrange interpolation, and arithmetic circuit. Based on our secure framework, both users' gradient information and the confidentiality of users' gradient aggregation weights can be protected. Besides, the (T, N)-threshold property of our GALAXY can defend against the collusion among multiple fog nodes while supporting some fog nodes dropping out during the training process. Furthermore, as same as PPFDL [16], our GALAXY can effectively handle low-quality-data users.

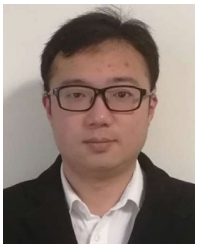
VII. CONCLUSION

In this paper, we have proposed a practical privacy-preserving federated learning GALAXY in vehicular fog computing, the first of its kind in the regime of privacy-preserving FL in the setting of non-cloud-assisted fog computing. For enhancing the practicality in real-world applications, our scheme can also mitigate the negative impact of low-quality data during FL training process, while all user-related information can be protected under our secure framework. We demonstrate the performance of our GALAXY through extensive experiments.

REFERENCES

[1] R. Shokri, M. Stronati, C. Song, and V. Shmatikov, "Membership inference attacks against machine learning models," in *Proc. IEEE Symp. Secur. Privacy*, 2017, pp. 3–18.
[2] B. Hitaj, G. Ateniese, and F. Pérez-Cruz, "Deep models under the gan: Information leakage from collaborative deep learning," in *Proc. ACM SIGSAC Conf. Comput. Commun. Secur.*, 2017, pp. 603–618.
[3] G. Xu, H. Li, H. Ren, K. Yang, and R. H. Deng, "Data security issues in deep learning: Attacks, countermeasures, and opportunities," *IEEE Commun. Mag.*, vol. 57, no. 11, pp. 116–122, Nov. 2019.
[4] B. Jayaraman and D. Evans, "Evaluating differentially private machine learning in practice," in *Proc. 28th USENIX Secur. Symp.*, Santa Clara, CA, 2019, pp. 1895–1912.
[5] N. Agrawal, A. S. Shamsabadi, M. J. Kusner, and A. Gascón, "QUOTIENT: Two-party secure neural network training and prediction," in *Proc. ACM SIGSAC Conf. Comput. Commun. Secur.* New York, NY, USA:, 2019, pp. 1231–1247.
[6] P. Mohassel and Y. Zhang, "SecureML: A. system for scalable privacy-preserving machine learning," in *Proc. IEEE Symp. Secur. Privacy*, 2017, pp. 19–38.
[7] I. Damgård, D. Escudero, T. Frederiksen, M. Keller, P. Scholl, and N. Volgushev, "New primitives for actively-secure MPC over rings with applications to private machine learning," in *Proc. IEEE Symp. Secur. Privacy*, 2019, pp. 1102–1120.
[8] P. Mohassel and P. Rindal, "ABY3: A mixed protocol framework for machine learning," in *Proc. ACM SIGSAC Conf. Comput. Commun. Secur.*, New York, NY, USA, 2018, pp. 35–52.
[9] S. Wagh, D. Gupta, and N. Chandran, "SecureNN: 3-party secure computation for neural network training," *Proc. Privacy Enhancing Technol.*, vol. 2019, no. 3, pp. 26–49, 2019.
[10] S. Wagh, S. Tople, F. Benhamouda, E. Kushilevitz, P. Mittal, and T. Rabin, "FALCON: Honest-majority maliciously secure framework for private deep learning," *Proc. Privacy Enhancing Technol.*, vol. 2021, no. 1, pp. 188–208, 2020.
[11] M. Byali, H. Chaudhari, A. Patra, and A. Suresh, "FLASH: Fast and robust framework for privacy-preserving machine learning," *Proc. Privacy Enhancing Technol.*, vol. 2020, no. 2, pp. 459–480, 2020.
[12] H. Chaudhari, R. Rachuri, and A. Suresh, "Trident: Efficient 4PC framework for privacy preserving machine learning," in *Proc. 27th Annu. Netw. Distrib. Syst. Secur. Symp.*, 2020.
[13] H. Sun, X. Ma, and R. Q. Hu, "Adaptive federated learning with gradient compression in uplink NOMA," *IEEE Trans. Veh. Technol.*, vol. 69, no. 12, pp. 16325–16329, Dec. 2020.
[14] P. Luo, F. R. Yu, J. Chen, J. Li, and V. C. M. Leung, "A novel adaptive gradient compression scheme: Reducing the communication overhead for distributed deep learning in the Internet of Things," *IEEE Internet Things J.*, vol. 8, no. 14, pp. 11476–11486, Jul. 2021.
[15] L. Zhao, Q. Wang, Q. Zou, Y. Zhang, and Y. Chen, "Privacy-preserving collaborative deep learning with unreliable participants," *IEEE Trans. Inf. Forensics Secur.*, vol. 15, pp. 1486–1500, 2020.
[16] G. Xu, H. Li, Y. Zhang, S. Xu, J. Ning, and R. Deng, "Privacy-preserving federated deep learning with irregular users," *IEEE Trans. Dependable Secure Comput.*, to be published, doi: 10.1109/TDSC.2020.3005909.

[17] N. Kato, B. Mao, F. Tang, Y. Kawamoto, and J. Liu, "Ten challenges in advancing machine learning technologies toward 6G," *IEEE Wireless Commun.*, vol. 27, no. 3, pp. 96–103, Jun. 2020.
[18] C. Zhou, A. Fu, S. Yu, W. Yang, H. Wang, and Y. Zhang, "Privacy-preserving federated learning in fog computing," *IEEE Internet Things J.*, vol. 7, no. 11, pp. 10782–10793, Nov. 2020.
[19] Y. Li, H. Li, G. Xu, T. Xiang, X. Huang, and R. Lu, "Toward secure and privacy-preserving distributed deep learning in fog-cloud computing," *IEEE Internet Things J.*, vol. 7, no. 12, pp. 11460–11472, Dec. 2020.
[20] A. Shamir, "How to share a secret," *Commun. ACM*, vol. 22, pp. 612–613, 1979.
[21] N. Kato *et al.*, "The deep learning vision for heterogeneous network traffic control: Proposal, challenges, and future perspective," *IEEE Wireless Commun.*, vol. 24, no. 3, pp. 146–153, Jun. 2017.
[22] G. Xu, H. Li, S. Liu, K. Yang, and X. Lin, "VerifyNet: Secure and verifiable federated learning," *IEEE Trans. Inf. Forensics Secur.*, vol. 15, pp. 911–926, 2020.
[23] S. Chang and C. Li, "Privacy in neural network learning: Threats and countermeasures," *IEEE Netw.*, vol. 32, no. 4, pp. 61–67, Jul./Aug. 2018.
[24] T. Zhu, G. Li, W. Zhou, and S. Y. Philip, *Differential Privacy and Applications*. Berlin, Germany: Springer, 2017.
[25] N. Kato *et al.*, "Optimizing space-air-ground integrated networks by artificial intelligence," *IEEE Wireless Commun.*, vol. 26, no. 4, pp. 140–147, Aug. 2019.
[26] Y. Lin, S. Han, H. Mao, Y. Wang, and B. Dally, "Deep gradient compression: Reducing the communication bandwidth for distributed training," in *Proc. Int. Conf. Learn. Representations*, 2018. [Online]. Available: https://openreview.net/forum?id=SkhQHMW0W
[27] K. Hsieh *et al.*, "Gaia: Geo-distributed machine learning approaching LAN speeds," in *Proc. 14th USENIX Symp. Networked Syst. Des. Implementation*, 2017, pp. 629–647.
[28] L. Wang, W. Wang, and B. Li, "CMFL: Mitigating communication overhead for federated learning," in *Proc. IEEE 39th Int. Conf. Distrib. Comput. Syst.*, 2019, pp. 954–964.
[29] G. Xu, H. Li, S. Liu, M. Wen, and R. Lu, "Efficient and privacy-preserving truth discovery in mobile crowd sensing systems," *IEEE Trans. Veh. Technol.*, vol. 68, no. 4, pp. 3854–3865, Apr. 2019.
[30] J. So, B. Guler, and S. Avestimehr, "A scalable approach for privacy-preserving collaborative machine learning," in *Proc. Adv. Neural Inf. Process. Syst.*, 2020, pp. 8054–8066.
[31] C. Gentry, J. Groth, Y. Ishai, C. Peikert, A. Sahai, and A. Smith, "Using fully homomorphic hybrid encryption to minimize non-interative zero-knowledge proofs," *J. Cryptol.*, vol. 28, no. 4, pp. 820–843, 2015.
[32] K. Bonawitz *et al.*, "Practical secure aggregation for privacy-preserving machine learning," in *Proc. ACM SIGSAC Conf. Comput. Commun. Secur.*, 2017, pp. 1175–1191.
[33] F. Fischer *et al.*, "Stack overflow considered harmful? The impact of copy&paste on Android application security," in *Proc. IEEE Symp. Secur. Privacy*, 2017, pp. 121–136.
[34] V. Smith, C.-K. Chiang, M. Sanjabi, and A. S. Talwalkar, "Federated multi-task learning," in *Proc. 31st Int. Conf. Neural Inf. Process. Syst.*, 2017, pp. 4424–4434.
[35] L. Yu, L. Liu, C. Pu, M. E. Gursoy, and S. Truex, "Differentially private model publishing for deep learning," in *Proc. IEEE Secur. Privacy*, 2019, pp. 309–326.
[36] L. T. Phong, Y. Aono, T. Hayashi, L. Wang, and S. Moriai, "Privacy-preserving deep learning via additively homomorphic encryption," *IEEE Trans. Inf. Forensics Secur.*, vol. 13, no. 5, pp. 1333–1345, May 2018.



Yiran Li (Graduate Student Member, IEEE) received the M.S. degree in communication and information system in 2009 from the University of Electronic Science and Technology of China, Chengdu, China, where he is currently working toward the Ph.D. degree with the School of Computer Science and Engineering. His research interests include cryptography, privacy-preserving deep learning, and data security.

Hongwei Li (Senior Member, IEEE) received the Ph.D. degree from the University of Electronic Science and Technology of China, Chengdu, China, in 2008. He is currently the Head and a Professor with the Department of Information Security, School of Computer Science and Engineering, University of Electronic Science and Technology of China. From October 2011 to October 2012, he was a Postdoctoral Fellow with the University of Waterloo, Waterloo, ON, Canada. He is currently the Secretary of IEEE ComSoc CIS-TC.

Guowen Xu (Member, IEEE) received the Ph.D. degree in cyberspace security from the University of Electronic Science and Technology of China, Chengdu, China, in 2020. He is currently a Research Fellow with Nanyang Technological University, Singapore. His research interests include secure outsourcing computing and privacy-preserving issues in deep learning. He was the recipient of the Best Paper Award of the 26th IEEE International Conference on Parallel and Distributed Systems (ICPADS 2020).

Tao Xiang (Member, IEEE) received the Ph.D. degree in computer science from Chongqing University, Chongqing, China, in 2008. He is currently a Professor of the College of Computer Science, Chongqing University. His research interests include cryptography, multimedia security, cloud security, and data privacy.

Rongxing Lu (Fellow, IEEE) received the Ph.D. degree from the Department of Electrical & Computer Engineering, University of Waterloo, Waterloo, ON, Canada, in 2012. He is currently an Associate Professor with the Faculty of Computer Science, University of New Brunswick, Saint John, NB, Canada. He was an Assistant Professor with the School of Electrical and Electronic Engineering, Nanyang Technological University, Singapore, from April 2013 to August 2016. He was a Postdoctoral Fellow with the University of Waterloo, from May 2012 to April 2013. He is currently a Fellow of IEEE Communications Society. He is currently the Vice-Chair (Publication) of IEEE ComSoc CIS-TC.