A Privacy Preserving and Byzantine Robust Collaborative Federated Learning Method Design

Nuocheng Yang*, Sihua Wang*†, Mingzhe Chen‡, Changchuan Yin*, and Christopher G. Brinton§

*Beijing Laboratory of Advanced Information Network, Beijing University of Posts and Telecommunications, Beijing, China, Emails: {YangNuoCheng, sihuawang, ccyin}@bupt.edu.cn
†State Key Laboratory of Networking and Switching Technology, Beijing University of Posts and Telecommunications, Beijing, China.
‡Department of Electrical and Computer Engineering and the Institute for Data Science and Computing, University of Miami, Coral Gables, FL USA, Email: mingzhe.chen@miami.edu.
§ School of Electrical and Computer Engineering, Purdue University, West Lafayette, IN, USA, Email: cgb@purdue.edu.

***Abstract*—Collaborative federated learning (CFL) enables device cooperation in training shared machine learning models without reliance on a parameter server. However, the absence of a parameter server also impacts vulnerabilities associated with adversarial attacks, including privacy inference and Byzantine attacks. In this context, this paper introduces a novel CFL framework that enables each device to individually determine the subset of devices to transmit FL parameters to over the wireless network, based on its neighboring devices' location, current loss, and connection information, to achieve privacy protection and robust aggregation. This is formulated as an optimization problem whose goal is to minimize CFL training loss while satisfying the privacy preservation, robust aggregation, and transmission delay requirements. To solve this problem, a proximal policy optimization (PPO)-based reinforcement learning (RL) algorithm integrated with a graph neural network (GNN) is proposed. Compared to traditional algorithms that use global information with high computational complexity, the proposed GNN-RL method can be deployed on devices based on neighboring information with lower computational overhead. Simulation results show that the proposed algorithm can protect data privacy and increase identification accuracy by 15% compared to an algorithm in which devices are partially clustered for model aggregation.**

***Keywords*—Collaborative federated learning, data privacy, graph neural network, reinforcement learning.**

I. Introduction

Federated learning (FL) [1], [2] enables devices to cooperatively train a machine learning (ML) model without data exchange, thus improving data privacy. However, standard FL requires devices to transmit their FL models to a parameter server, which is not always feasible due to limited wireless resources (i.e., energy and bandwidth resources) in practical networks. To address this problem, a novel FL framework that combines the principles of collaborative learning with federated learning, called collaborative federated learning (CFL), has been proposed [3], [4]. In particular, CFL enables devices to collaboratively train an ML model via exchanging their local FL model parameters with their neighboring devices, without the reliance on a parameter server. However, compared with centralized FL, CFL introduces new data leakage and Byzantine attack vulerabilities [5] since a single dishonest device can affect the entire network and final results.

This work was supported in part by Beijing Natural Science Foundation under Grant L223027, in part by the National Natural Science Foundation of China under Grants 61629101 and 61671086, in part by the China 973 Program under Grant 2012CB315801, and in part by the U.S. National Science Foundation under grant CNS-2146171 and CPS-2313109.

Recently, a number of works such as [6]–[8] studied the use of differential privacy (DP) and encryption gradient algorithms, such as Homomorphic encryption and secure multiparty computation (MPC), to address data leakage issues. The authors in [6] introduced proxy and DP schemes into CFL to safeguard privacy guarantees. The authors in [7] designed a privacy-preserving and reliable decentralized FL scheme based on local DP and dynamic encryption. The authors in [8] introduce Gaussian noise into the signal of model parameters to satisfy DP requirements. However, incorporating DP schemes into traditional neural networks introduces noise, thus reducing CFL performance. To overcome this, we consider a Bayesian Neural Network (BNNs) approach to treat parameters as random variables, inherently incorporating sampling bias into CFL model parameters. Thus, BNNs effectively safeguard both the model and its gradients. Additionally, BNNs have been shown to outperform traditional neural networks in certain instances by mitigating overfitting, especially in situations with limited datasets that are susceptible to data leakage.

A number of works such as [5], [9], [10] focused on resisting Byzantine attacks in distributed learning. The authors in [5] proposed a Byzantine-resilient aggregation rule that compares the Euclidean distance of the estimate of each neighbor device with its own estimate to defeat Byzantine adversaries in CFL. The authors in [9] designed a decentralized approach that enables all participating nodes to collaboratively identify malicious entities through an innovative cross-check mechanism. The authors in [10] designed a decentralized blockchain-based FL architecture by using a secure global aggregation algorithm to resist malicious devices. However, these approaches require frequent communication among devices and depend on device cooperation or additional computations to defend against malicious attacks.

The authors in [11], [12] have explored solutions for both privacy leakage and Byzantine attacks. The authors in [11] designed a Byzantine-resilient secure aggregation framework based on integrated stochastic quantization, verifiable outlier detection, and a secure model aggregation approach. The authors in [12] proposed a hierarchical CFL framework to identify harmful devices within a group and regularly share model information with surrounding groups. However, in these studies, frequent and additional communication and computation are necessary to defend against Byzantine attacks, which will bring significant computational and communication overhead. Therefore, they are not well-suited for large-scale

ICC 2024 - IEEE International Conference on Communications | 978-1-7281-9054-9/24/$31.00 ©2024 IEEE | DOI: 10.1109/ICC51166.2024.10622626

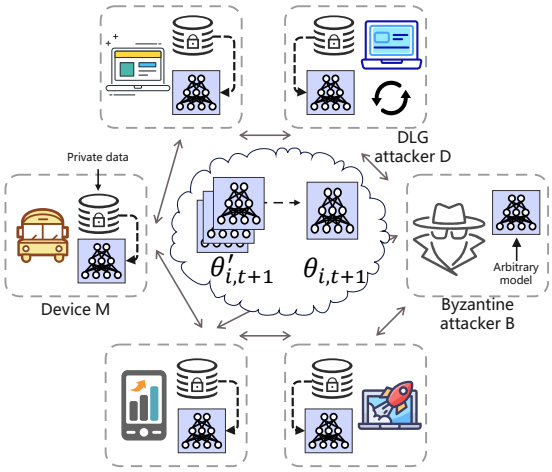


Fig. 1. Illustration of the considered CFL model.

CFL scenarios.

In this paper, we develop a novel, fully distributed CFL framework that enables distributed devices to train an ML model in a privacy-preserving and Byzantine-robust manner. Our key contributions are as follows:

- We propose a novel CFL framework in which mobile devices collaboratively train an ML model via exchanging their FL parameters with their neighboring devices, under the attacks of Byzantine and deep leakage from gradient (DLG) adversaries. In the considered model, due to the limited resources and privacy requirements, each device must select a subset of devices per FL iteration to exchange their FL parameters with, aiming to minimize CFL training loss. This problem is formulated as an optimization problem aiming to minimize the loss function of CFL training while satisfying the privacy and transmission delay requirements, by determining the devices that each device transmits its FL parameters to.
- To solve the formulated problem, we proposed an efficient graph neural network (GNN) reinforcement learning (RL) method where we model devices and connections as the set of nodes and edges in the graph. We search for the adaptive model transmission policy based on the proximal policy optimization (PPO) algorithm. Compared to the traditional solutions for both privacy leakage and Byzantine attacks, the proposed GNN RL method can achieve privacy requirements without the need for encryption or the introduction of noise.

Simulation results show that, compared to a clustered CFL baseline, the proposed PPO-based algorithm can protect data privacy and increase identification accuracy by 15%.

II. System Model and Problem Formulation

Consider a wireless network that consists of a set $\mathcal{M}$ of M mobile devices for collaboratively training an ML model, and a set $\mathcal{B}$ of B Byzantine attackers aiming to disrupt the CFL learning process by transmitting entirely arbitrary model parameters. Additionally, within this network, a subset of D devices, known as honest but curious deep leakage from gradient attackers, are hidden in $\mathcal{M}$. These DLG attackers aim to reconstruct the private data of neighboring devices from the updates of model parameters. This is summarized in Fig. 1.

We assume that each device i in $\mathcal{M}$ has a dataset $\mathcal{D}_i$, containing N_i training data samples with $N = \sum_{i=1}^{M} N_i$ being the total number of training data samples across devices. Each training data sample n consists of an input feature vector $\boldsymbol{x}_{i,n} \in \mathbb{R}^{N_{\mathrm{I}} \times 1}$ and a corresponding label vector $\boldsymbol{y}_{i,n} \in \mathbb{R}^{N_{\mathrm{O}} \times 1}$. In the considered model, the objective of devices in $\mathcal{M}$ is to solve the following optimization problem overall data samples against Byzantine adversaries while safeguarding data privacy:

$$\min_{\boldsymbol{w}_{1,t},\ldots,\boldsymbol{w}_{M,t}} \frac{1}{M} \sum_{i=1}^{M} F\left(\boldsymbol{w}_{i,t}\right), \tag{1}$$

$$\text{s.t. } \boldsymbol{w}_{i,t} = \boldsymbol{w}_{j,t}, \forall i, j \in \mathcal{M}, \tag{1a}$$

where $\boldsymbol{w}_{i,t} \in \mathbb{R}^{V \times 1}$ is a vector to capture the local FL model on device i at iteration t. $F\left(\boldsymbol{w}_{i,t}\right)$ of device i is given by

$$F\left(\boldsymbol{w}_{i,t}\right) = \frac{1}{N_i} \sum_{i=1}^{N_i} f\left(\boldsymbol{w}_{i,t}; \phi\left(\boldsymbol{w}_{i,t}, \boldsymbol{x}_{i,n}\right), \boldsymbol{y}_{i,n}\right), \tag{2}$$

where $\phi(\cdot)$ denotes the neural network function and $f\left(\boldsymbol{w}_{i,t}; \phi\left(\boldsymbol{w}_{i,t}, \boldsymbol{x}_{i,n}\right), \boldsymbol{y}_{i,n}\right)$ is the loss function that measures the difference between the output $\phi\left(\boldsymbol{w}_i, \boldsymbol{x}_{i,n}\right)$ and label $\boldsymbol{y}_{i,n}$.

In contrast to the optimization-based approaches that treat model weights as unknown parameters, Bayesian learning estimates the model weights by a maximum posterior probability (MAP) estimator given the distribution of the weights conditional on the training dataset. Specifically, the local objective function of device i is expressed as

$$\min_{\boldsymbol{w}_{i,t}} F(\boldsymbol{w}_{i,t}) = -\min_{\boldsymbol{w}_{i,t}} \log P\left(\boldsymbol{w}_{i,t}; \mathcal{D}_i\right), \tag{3}$$

where $P\left(\boldsymbol{w}_{i,t}; \mathcal{D}_i\right)$ is the local joint distribution of $\boldsymbol{w}_{i,t}$ and $\mathcal{D}_i$, which is

$$P\left(\boldsymbol{w}_{i,t}; \mathcal{D}_i\right) = P_{\mathrm{l}}(\mathcal{D}_i|\boldsymbol{w}_{i,t}) P_{i,0}(\boldsymbol{w}_{i,t}), \tag{4}$$

where $P_{i,0}\left(\boldsymbol{w}_i\right)$ is the local prior and $P_{\mathrm{l}}(\mathcal{D}_i|\boldsymbol{w}_i)$ is the local likelihood. For each data sample $(\boldsymbol{x}_{i,n}, \boldsymbol{y}_{i,n}) \in \mathcal{D}_i$, the data likelihood can be given by a parameterized Gaussian distribution:

$$P_{\mathrm{l}}(\mathcal{D}_i|\boldsymbol{w}_{i,t}) = \prod_{n=1}^{N_i} P_{\mathrm{l}}\left(\boldsymbol{y}_{i,n}|\phi\left(\boldsymbol{w}_i, \boldsymbol{x}_{i,n}\right)\right). \tag{5}$$

The data likelihood, related to the cross-entropy loss in classification problems, can be modeled by a categorical distribution:

$$P_{\mathrm{l}}\left(\boldsymbol{y}_{i,n}|\phi\left(\boldsymbol{w}_i, \boldsymbol{x}_{i,n}\right)\right) = \exp\left(-\mathcal{C}_E\left(\boldsymbol{y}_{i,n}; \phi\left(\boldsymbol{w}_i, \boldsymbol{x}_{i,n}\right)\right)\right), \tag{6}$$

where $\mathcal{C}_E(\cdot)$ is the cross-entropy function.

A. CFL Local Training Process

The main task for training Bayesian Neural Networks (BNNs) on device i at iteration t is inferring the posterior distribution $P\left(\boldsymbol{w}_{i,t}|\mathcal{D}_i\right)$ given the dataset $\mathcal{D}_i$ at iteration t, which is given by

$$P\left(\boldsymbol{w}_{i,t}|\mathcal{D}_i\right) = \frac{P_{\mathrm{l}}(\mathcal{D}_i|\boldsymbol{w}_{i,t}) P_{i,0}(\boldsymbol{w}_{i,t})}{\int P_{\mathrm{l}}(\mathcal{D}_i|\boldsymbol{w}_{i,t}) P_{i,0}(\boldsymbol{w}_{i,t}) d\boldsymbol{w}_{i,t}} \propto P\left(\boldsymbol{w}_{i,t}; \mathcal{D}_i\right). \tag{7}$$

Since the denominator of (7) does not have a general closed-form expression, it is infeasible to calculate $P\left(\boldsymbol{w}_{i,t}|\mathcal{D}_i\right)$ directly. Hence, we use Variational Inference (VI) to estimate $P\left(\boldsymbol{w}_{i,t}|\mathcal{D}_i\right)$ by $q\left(\boldsymbol{w}_{i,t}\right)$, which can be represented as

$$q\left(\boldsymbol{w}_{i,t}\right)=\mathcal{N}\left(\boldsymbol{w}_{i,t};\boldsymbol{m}_{i,t},\mathrm{diag}(\boldsymbol{s}_{i,t}^2)\right), \tag{8}$$

where $\boldsymbol{m}_{i,t}$ is the mean of the approximation, $\boldsymbol{s}_{i,t}$ is the standard deviation, and diag is the diagonal matrix. Compared to traditional neural networks that treat $\boldsymbol{w}_{i,t}$ as a fixed constant, we sample $\boldsymbol{w}_{i,t}$ from the distribution $q\left(\boldsymbol{w}_{i,t}\right)$ for the training and inferencing. By initializing $P_{i,0}\left(\boldsymbol{w}_i\right)=1$, each device in set $\mathcal{M}$ at the t-th iteration first approximates $P_{i,t}\left(\boldsymbol{w}_i\right)P\left(\boldsymbol{w}_i;\mathcal{D}_i\right)$ by $q\left(\boldsymbol{w}_{i,t}\right)$, which is given by

$$q\left(\boldsymbol{w}_{i,t}\right)=\arg\min_{q(\boldsymbol{w}_{i,t})}\mathbb{D}\left(q\left(\boldsymbol{w}_{i,t}\right)||P_{i,t}\left(\boldsymbol{w}_i\right)P\left(\boldsymbol{w}_i;\mathcal{D}_i\right)\right), \tag{9}$$

where $\mathbb{D}\left(\cdot\right)$ denotes the Kullback-Leibler (KL) divergence. On the other hand, the Byzantine devices in set $\mathcal{B}$ generate parameters randomly at each iteration. In the next subsection, we will introduce a CFL aggregation method to update the prior distribution $P_{i,t}$ in each iteration.

B. CFL Model Aggregation Process

Given the updated local estimation $P_{j,t}\left(\boldsymbol{w}_{j,t}\right)$, each device i exchanges $q\left(\boldsymbol{w}_{i,t}\right)$ with its neighboring devices. Then, devices in set $\mathcal{M}$ update their local prior distribution $P_{i,t}\left(\boldsymbol{w}_{i,t}\right)$ as follows:

$$P_{i,t}\left(\boldsymbol{w}_{i,t}\right)\propto\exp\left(\sum_{j\in\mathcal{R}}\frac{u_{i,j,t}}{||\boldsymbol{u}_{i,t}||}\log P_{j,t}\left(\boldsymbol{w}_{j,t}\right)\right), \tag{10}$$

where $\boldsymbol{u}_t=[\boldsymbol{u}_{1,t},...,\boldsymbol{u}_{M,t}]$ is the model transmission matrix, $\boldsymbol{u}_{i,t}=[u_{i,1,t},...,u_{i,M,t}]$ being a vector of FL transmission indices of device i. Here, $u_{i,j,t}=1$ implies that device i will exchange its local FL model with device j at iteration t, and $u_{i,j,t}=0$ otherwise. $||\boldsymbol{u}_{i,t}||=\sum_{j\in\mathcal{R}}u_{i,j,t}$ is the number of devices that will transmit FL parameters to device i. Set $\mathcal{R}=\mathcal{M}\cup\mathcal{B}$ contains total R devices from both $\mathcal{M}$ and $\mathcal{B}$. After T iterations, each device estimates the model weights according to its estimated marginal distribution:

$$\boldsymbol{w}_{i,T}^{*}=\arg\max_{\boldsymbol{w}_{i,T}}q\left(\boldsymbol{w}_{i,T}\right). \tag{11}$$

C. CFL Model Transmission Process

We adopt an orthogonal frequency division multiple access (OFDMA) transmission scheme. Let W be the bandwidth that the device can use for model parameter transmission and $p_{i,j,t}$ be the transmit power of device i. The data transmission delay of device i transmitting FL parameters to device j is

$$l_{i,j,t}(\boldsymbol{u}_{i,t},\boldsymbol{\phi}_t,W,p_{i,j,t})=\frac{A}{\frac{W}{||\boldsymbol{u}_{i,t}||}\log\left(1+\frac{p_{i,j,t}h_{i,j,t}(\boldsymbol{\phi}_t)}{\sigma_N^2}\right)}, \tag{12}$$

where $h_{i,j,t}=\rho_{i,j,t}d_{i,j,t}^{-2}$ is the channel gain between device i and j with $\rho_{i,j,t}$ being the Rayleigh fading parameters, and $d_{i,j,t}$ is the distance between device i and j. The location of each device i at time t is captured by a vector $\boldsymbol{\phi}_{i,t}=[\phi_{i,t,1},\phi_{i,t,2}]$, with $\boldsymbol{\phi}_t=[\boldsymbol{\phi}_{1,t},...,\boldsymbol{\phi}_{M,t}]$ being the location matrix across users at iteration t. σ_N^2 represents the variance of additive white Gaussian noise. A is the size of the transmitted prior distribution $P_{j,t}\left(\boldsymbol{w}_{j,t}\right)$, which is assumed to be equal for each device.

D. DLG adversaries

The DLG attackers aim to reconstruct the private data from a device i using the received parameters $\boldsymbol{w}'_{i,t-1}$ and the gradients $\nabla\boldsymbol{w}'_{i,t}=\frac{1}{\eta_t}\left(\boldsymbol{w}'_{i,t}-\boldsymbol{w}'_{i,t-1}\right)$. To reconstruct the private data from $\nabla\boldsymbol{w}'_{i,t}$, the DLG attacker first initializes dummy input $\boldsymbol{x}'$ and dummy labels $\boldsymbol{y}'$. Then, the attacker minimizes the distance between $\nabla\boldsymbol{w}'_{i,t}$ and dummy gradients $\nabla\boldsymbol{w}_{i,t}$, which is calculated from $\boldsymbol{x}'$, $\boldsymbol{y}'$, and $\boldsymbol{w}'_{i,t-1}$ via minimizing the following objective function using the BP algorithm:

$$\begin{aligned}\boldsymbol{x}^{*},\boldsymbol{y}^{*}&=\arg\min_{\boldsymbol{x}',\boldsymbol{y}'}\left(||\nabla\boldsymbol{w}'_{i,t}-\nabla\boldsymbol{w}_{i,t}||\right)\\&=\arg\min_{\boldsymbol{x}',\boldsymbol{y}'}\left(||\frac{\partial f\left(\boldsymbol{w}'_{i,t-1},\boldsymbol{x},\boldsymbol{y}\right)}{\partial\boldsymbol{w}_{i,t-1}}-\nabla\boldsymbol{w}'_{i,t}||\right).\end{aligned} \tag{13}$$

Given the optimal reconstruction data $\boldsymbol{x}^{*}$ that satisfies (13), we define the Euclidean distance between $\boldsymbol{x}$ and $\boldsymbol{x}^{*}$ as

$$\ell_2\left(\boldsymbol{x},\boldsymbol{x}^{*}\right)=\sqrt{\sum_{a=1}^{A}\left(\boldsymbol{x}_a-\boldsymbol{x}_a^{*}\right)^2}, \tag{14}$$

where A denotes total number of elements in $\boldsymbol{x}$. Then, the data leakage risk of device i among a batch of data is given by

$$\tau_{i,t}=\frac{1}{S}\sum_{s=1}^{S}\ell_2\left(\boldsymbol{x}_{i,s,t},\boldsymbol{x}_{i,s,t}^{*}\right), \tag{15}$$

where S is the batch size. The device can only ensure that the output does not contain information from (15), as it is impossible to detect deep gradient leakage attacks relying solely on the behavior.

E. Problem Formulation

Our goal is to minimize the FL training loss and total energy consumption while ensuring constraints of privacy preservation, robust aggregation, and transmission delay in each iteration. Minimizing the FL training loss is employed as a strategy to counteract Byzantine attackers when the convergence of devices' loss values is hindered by such attacks. The optimization problem is formulated as

$$\min_{\boldsymbol{U},\boldsymbol{P}}\frac{1}{R}\sum_{i=1}^{R}F\left(\boldsymbol{w}_{i,T}\right), \tag{16}$$

$$\text{s.t. } l_{i,j,t}\left(\boldsymbol{u}_{i,t},\boldsymbol{\phi}_t,B,p_{i,j,t}\right)\leqslant\mathit{\Gamma},\forall i,j\in\mathcal{R},\forall t\in\mathcal{T}, \tag{16a}$$

$$\tau_{i,t}\geqslant\tau^{*},\forall i\in\mathcal{R},\forall t\in\mathcal{T}, \tag{16b}$$

$$\sum_{j=1}^{R}p_{i,j,t}\leqslant p_{\max},\forall i\in\mathcal{R},\forall t\in\mathcal{T}, \tag{16c}$$

where $\boldsymbol{U}=[\boldsymbol{u}_1,...,\boldsymbol{u}_T]^{\top}$ is the FL model transmission matrix, and $\boldsymbol{P}=[\boldsymbol{p}_1,...,\boldsymbol{p}_t,...,\boldsymbol{p}_T]^{\top}$ is the transmit power matrix. $p_{\max}$ is the transmit power constraint. $\mathit{\Gamma}$ is the maximum FL model transmission delay per iteration allowed by the network

operator. (16a) is a constraint on the FL model transmission delay per iteration, (16b) is the privacy-preserving requirement, and (16c) is the transmit power constraint. Here, we assume that each device $i \in \mathcal{B}$ has a constant loss since the model parameters in device i are generated randomly and remain unchanged.

Since current algorithms often require extensive device interaction to detect Byzantine attacks or extensive encryption operations to resist data leakage, which may increase communication overhead and complexity, (16) is challenging to solve through current algorithms. To overcome this challenge, we propose a fully distributed algorithm based on GNNs that enables each device to determine the FL model transmission matrix $\boldsymbol{U}$ only with its neighboring devices' location information, current loss, and connection information. In the following, we will explain the details of the GNN and RL agents within our approach.

III. GNN-based RL for CFL Transmission Optimization

We now introduce a GNN based proximal policy optimization (PPO) algorithm to solve problem (16). Compared to standard PPO algorithms [13], our proposed PPO approach uses GNNs to extract network topological features including both wireless channel and edge device features for device connection probability prediction. Here, the use of PPO instead of value-based deep reinforcement learning (RL) approaches follows from the output (i.e., connection probability of each device) of a GNN being continuous, since PPO can process continuous data (particularly continuous actions).

We first introduce the use of the GNNs in the PPO algorithm to calculate the connection probability of each device. Here, the connection probability of GNNs will be optimized by the PPO algorithm. Then, we explain the components of the PPO algorithm. Finally, we show the entire procedure of using our GNN-based PPO algorithm to optimize the connection probabilities for each device.

A. Graph Neural Network

In the proposed PPO method, GNNs will be used for policy and critic functions. Hence, the input of GNNs will be the state of the PPO algorithm. To determine the input of the GNN in the PPO algorithm, we first construct a k nearest neighbor graph [14]. Here, we use a k nearest neighbor graph instead of a fully connected graph for the input of the GNN. Let $\mathcal{L}^1(i,k)$ be the set of the first hop devices that can directly connect to device i and $\mathcal{L}^2(i,k)$ be the set of the second hop devices that can connect to device i via $\mathcal{L}^1(i,k)$. Let $\mathcal{L}(i,k) = \mathcal{L}^1(i,k) \cup \mathcal{L}^2(i,k) \cup \{i\}$, with $|\mathcal{L}(i,k)|$ being the number of devices in $\mathcal{L}(i,k)$. The input to the GNN model for device i comprises a vector that includes concatenated locations and loss values for the devices in $\mathcal{L}^2(i,k)$ at iteration t, which is represented as

$$\boldsymbol{\nu}_{i,t} = \left(\left[\boldsymbol{\phi}_{1,t}, ..., \boldsymbol{\phi}_{|\mathcal{L}(i,k)|,t}\right], \left[F\left(\boldsymbol{w}_{1,t}\right), ..., F\left(\boldsymbol{w}_{|\mathcal{L}(i,k)|,t}\right)\right]\right)^{\top}, \tag{17}$$

where $\boldsymbol{\nu}_{i,t} = \left[\nu_{i,1,t}, ..., \nu_{i,|\mathcal{L}(i,k)|,t}\right] \in \mathbb{R}^{|\mathcal{L}(i,k)|\times 3}$, and $(\cdot,\cdot)$ is the operation of concatenating two vectors into a new vector by stacking them vertically. Here, $\boldsymbol{\nu}_{i,t}$ will be used as the input of both the policy and critic functions. The hidden layer of each GNN consists of two graph convolutional network layers (GCNs). The output of the GNN based policy function is a probability distribution $\boldsymbol{\mu}_{i,t} = \left[\mu_{i,1,t}, ..., \mu_{i,|\mathcal{L}^1(i,k)|,t}\right]$ of device i connecting to its first hop devices. The output of the GNN based critic function is an estimate of the value function corresponding to the current policy $\boldsymbol{\mu}_{i,t}$ for a given state $\boldsymbol{\nu}_{i,t}$.

B. Components of RL Method

The proposed PPO algorithm consists of six components: a) agent, b) action, c) state, d) policy, e) critic, and f) reward, which are specified as follows:

- **Agent:** The agents are mobile devices in set $\mathcal{R}$.
- **Action:** The continuous action of device i at time slot t is $\boldsymbol{\mu}_{i,t}$ which is the output of the GNN based policy function. The action jointly considers the FL performance optimization, privacy-preserving, and robust aggregation requirements. Given $\boldsymbol{\mu}_{i,t}$, the next step is to determine the FL model transmission matrix $\boldsymbol{u}_i$ of each device i. Then, each device i will exchange their local prior distribution based on $\boldsymbol{u}_i$. We first use a set $\mathcal{S}_i = \left\{\mu_{i,1}, ..., \mu_{i,|\mathcal{L}^1(i,k)|}\right\}$ to represent the set of elements in $\boldsymbol{\mu}_i$. We denote $\mu_i^{'}$ as the $\lfloor \frac{k}{2} \rfloor$-th largest element in $\mathcal{S}_i$. The symbol $\lfloor x \rfloor$ represents the floor function, which gives the largest integer that is less than or equal to the real number x. Then we can define set $\mathcal{S}_i^{'} = \left\{j | \mu_{i,j} \geqslant \mu_i^{'}, j, z \in \mathcal{L}^1(i,k)\right\}$. Hence, element $u_{i,j}$ in FL model transmission matrix is determined by

$$u_{i,j} = \begin{cases} 1, & \text{if } j \in \mathcal{S}_i^{'} \text{ and } i \in \mathcal{S}_j^{'}, \\ 0, & \text{otherwise.} \end{cases} \tag{18}$$

  The FL model transmission matrix $\boldsymbol{u}_t$ is completed based on each device executing its action.
- **State:** The state, defined as $\boldsymbol{\nu}_{i,t}$, consists of: 1) the devices' position $\left[\boldsymbol{\phi}_{1,t}, ..., \boldsymbol{\phi}_{|\mathcal{L}(i,k)|,t}\right]$ and 2) the loss value of each device $\left[F\left(\boldsymbol{w}_{1,t}\right), ..., F\left(\boldsymbol{w}_{|\mathcal{L}(i,k)|,t}\right)\right]$.
- **Policy:** The policy is the probability of the agent choosing each action given the state $\boldsymbol{\nu}_{i,t}$. The PPO algorithm uses a GNN parameterized by $\boldsymbol{\theta}_t$ to build the relationship between the input state $\boldsymbol{\nu}_{i,t}$ and the output policy that can achieve the maximum total reward, which is also called the actor. Then, the policy can be expressed as $\boldsymbol{\pi}_{\boldsymbol{\theta}_t}(\boldsymbol{\nu}_{i,t}, \boldsymbol{\mu}_{i,t}) = P(\boldsymbol{\mu}_{i,t} | \boldsymbol{\nu}_{i,t})$.
- **Critic:** The critic $\boldsymbol{V}_{\boldsymbol{\theta}_t^{'}}(s_t)$ in the proposed method is a function to estimate the value-function of the current policy for a given state, which is a GNN parameterized by $\boldsymbol{\theta}_t^{'}$.
- **Reward:** The reward of choosing action $\boldsymbol{\mu}_{i,t}$ based on state $\boldsymbol{\nu}_{i,t}$ is

$$r\left(\boldsymbol{\mu}_{i,t} | \boldsymbol{\nu}_{i,t}\right) = -F(\boldsymbol{w}_{i,t}) - \lambda_i \sum_{j=1}^{M} \mathbb{1}_{\{l_{i,j,t}(\boldsymbol{u}_{i,t}, \boldsymbol{\phi}_t, B, p_{i,j,t}) - \Gamma\}} - \varrho_i \mathbb{1}_{\left\{\sum_{j=1}^{M} \tau_{i,j,t} - \tau^*\right\}} - \xi_i \mathbb{1}_{\left\{\sum_{j=1}^{M} p_{i,j,t} - p_{\max}\right\}}, \tag{19}$$

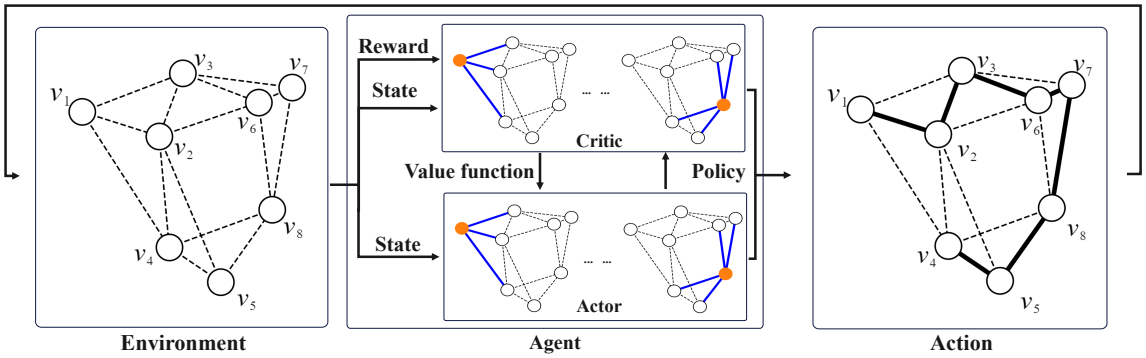


Fig. 2. Illustration of our proposed GNN-based algorithm.

where $\mathbb{1}_{\{x\}} = x$ if $x > 0$, $\mathbb{1}_{\{x\}} = 0$, otherwise. $r\left(\boldsymbol{\mu}_{i,t}|\boldsymbol{\nu}_{i,t}\right)$ is the loss of device i resulting from action $\boldsymbol{\mu}_{i,t}$ at state $\boldsymbol{\nu}_{i,t}$.

C. PPO Algorithm for Total Reward Maximization

Next, we introduce the entire procedure of training the proposed PPO algorithm for solving problem (16). In the proposed PPO algorithm, the objective function of the actor is

$$\mathfrak{L}\left(\boldsymbol{\nu}_{i,t}, \boldsymbol{\mu}_{i,t}|\boldsymbol{\theta}_t\right) = \mathbb{E}\left[p_{\boldsymbol{\theta}_t} A\left(\boldsymbol{\nu}_{i,t}, \boldsymbol{\mu}_{i,t}\right)\right], \tag{20}$$

where

$$A\left(\boldsymbol{\nu}_{i,t}, \boldsymbol{\mu}_{i,t}\right) = r\left(\boldsymbol{\nu}_{i,t}, \boldsymbol{\mu}_{i,t}\right) + \gamma \boldsymbol{V}_{\boldsymbol{\theta}_t'}\left(\boldsymbol{\nu}_{i,t+1}\right) - \boldsymbol{V}_{\boldsymbol{\theta}_t'}\left(\boldsymbol{\nu}_{i,t}\right), \tag{21}$$

and $p_{\boldsymbol{\theta}_t} = \boldsymbol{\pi}_{\boldsymbol{\theta}_t}/\boldsymbol{\pi}_{\boldsymbol{\theta}_{t-1}}$ is the advantage function and the probability ratio of the current policy and the old policy function, respectively. To satisfy the trust region constraint in PPO, the proposed PPO-based approach maximizes a clipping surrogate objective function, which is expressed as

$$\begin{aligned}&\mathfrak{L}^{\mathrm{c}}\left(\boldsymbol{\nu}_{i,t}, \boldsymbol{\mu}_{i,t}|\boldsymbol{\theta}_t\right)\\ &= \mathbb{E}[\min\{p_{\boldsymbol{\theta}_t} A\left(\boldsymbol{\nu}_{i,t}, \boldsymbol{\mu}_{i,t}\right), \zeta\left(p_{\boldsymbol{\theta}_t}, 1-\epsilon, 1+\epsilon\right) A\left(\boldsymbol{\nu}_{i,t}, \boldsymbol{\mu}_{i,t}\right)\}],\end{aligned} \tag{22}$$

where ϵ is a hyper-parameter that adjusts the clipping fraction of the clipping range, and $\zeta(\cdot)$ is the clip function. Then, we can update the actor model $\boldsymbol{\theta}_t$ by mini-batch stochastic gradient descent (SGD) method which can be represented as

$$\boldsymbol{\theta}_t = \boldsymbol{\theta}_{t-1} - \frac{1}{B} \sum_{\left(s_t, a_t, r_t, s_{t+1}\right)} \nabla_{\boldsymbol{\theta}_{t-1}} \mathfrak{L}^{\mathrm{c}}\left(s_t, a_t|\boldsymbol{\theta}_{t-1}\right). \tag{23}$$

After conducting n iterations of policy function optimization, we utilize regression on the mean-squared error to adjust the value function $\boldsymbol{V}_{\boldsymbol{\theta}'}$ based on the actual rewards, which is given by

$$\boldsymbol{\theta}_t' = \boldsymbol{\theta}_{t-1}' - \frac{1}{B} \sum_{\left(\boldsymbol{\nu}_{i,t}, \boldsymbol{\mu}_{i,t}, r_{i,t+1}, \boldsymbol{\nu}_{i,t+1}\right)} \nabla_{\boldsymbol{\theta}_{t-1}'} \left(\boldsymbol{V}_{\boldsymbol{\theta}_{t-1}'}\left(\boldsymbol{\nu}_{i,t}\right) - r_{i,t}\right)^2. \tag{24}$$

By iteratively running the policy updating step (23) and the state-value updating step (24), the parameters $\boldsymbol{\theta}_t$ and $\boldsymbol{\theta}_t'$ of the policy and state-value can find the relation between the device connections and the total reward, jointly considering the CFL performance, privacy protection and robust aggregation. The specific training process of the proposed PPO algorithm is summarized in Algorithm 1.

Algorithm 1 GNN based PPO Algorithm for FL model transmission optimization

1: **Initialize** policy parameters $\boldsymbol{\theta}_t$, initial state-value function parameters $\boldsymbol{\theta}_t'$.
2: **for** $t = 1, 2, \ldots, T$ **do**
3: Collect $\left(s_t, a_t, r_t, s_{t+1}\right)$ by running policy $\boldsymbol{\pi}_{\boldsymbol{\theta}_t}$ in the environment;
4: Compute advantage estimates, $A\left(\boldsymbol{s}_t, \boldsymbol{a}_t\right)$ based on the current state-value function $V_{\boldsymbol{\theta}_t'}$;
5: Update the policy by maximizing the (22);
6: Fit state-value function by regression on mean-squared error based on (24);
7: **end for**

D. Optimization of Transmit Power Vector with Fixed FL Model Transmission Matrix

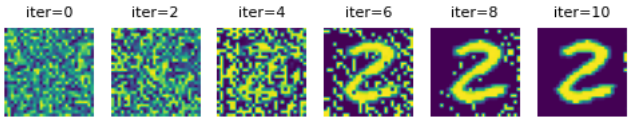
Once the FL model transmission matrix $\boldsymbol{U}$ has been determined, the minimal transmit power vector $\boldsymbol{p}_{i,t}$ of each device can be obtained through the following lemma, similar to [15].

Lemma 1. The optimal transmit power $p_{i,j,t}$ of each device i transmitting its model to device j is given by

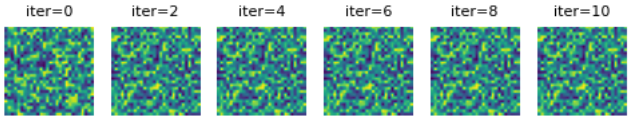
$$p_{i,j,t}^* = \frac{u_{i,j,t}\sigma_N^2}{h_{i,j,t}\left(\boldsymbol{\phi}_t\right)}\left(2^{\frac{A||\boldsymbol{u}_i||}{BI}} - 1\right). \tag{25}$$

IV. Simulation Results

For our simulations, we consider a network with a circular area having a radius $r = 1000$ m, comprising 8 devices in set $\mathcal{M}$ and 4 devices in set $\mathcal{B}$. The other parameters used in simulations are the same as in [15]. We consider the use of FL for handwritten digit identifications based on the MNIST dataset [16]. For comparison, we evaluate the proposed algorithm by comparing it with a clustered CFL baseline [12] that does not involve encryption. Within the clustered CFL, clusters are formed by grouping together k nearest neighboring devices. During each iteration of the



(a) Clustered CFL.



(b) Proposed method.

Fig. 3. Reconstruction results of the DLG attack.

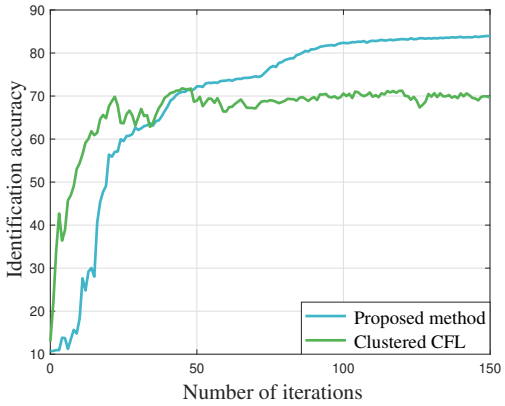


Fig. 4. Test accuracy vs. the number of iterations.

clustered CFL, one device within the cluster is chosen for model aggregation and detecting potential Byzantine attackers.

In Fig. 3, we show how the reconstructed image changes as the number of DLG attack iterations varies. From Fig. 3(a), we observe that with the clustered CFL baseline, the DLG attackers can recover the raw data of edge devices. This is due to the fact that a device in a cluster knows the parameters of other devices in the same cluster at each training iteration, which may lead to data leakage. In contrast, in Fig. 3(b), the proposed algorithm prevents data leakage, and the DLG attackers cannot recover raw data of the devices. This is due to the fact that the proposed method aggregates the prior information from different neighboring devices in each iteration, and the model update is composed of the likelihood from local data and the KL divergence with the aggregated prior and local likelihood.

In Fig. 4, we show how the average identification accuracy of both considered algorithms changes as the number of iterations varies. In particular, in Fig. 4, compared to clustered CFL, the proposed method can improve the identification accuracy by up to 15% over time. This improvement stems from the fact that the proposed method detects Byzantine attackers, thereby reducing the probability of exchanging model parameters to minimize Byzantine attackers' impact.

V. Conclusion

In this paper, we proposed a novel CFL framework in which mobile devices collaboratively train an ML model via exchanging their FL parameters with their neighboring devices, in the presence of Byzantine and DLG adversaries. We combined PPO with GNN to capture the connections between neighboring devices' information and the probability of device connections. The proposed method enables each device to individually determine its FL parameter transmission matrix using its neighboring devices' location information, current loss, and connection information. Simulation results demonstrated that the proposed algorithm can achieve robust aggregation compared to clustered CFL, while ensuring data privacy without any encryption or additional noise.

References

[1] M. Chen, D. Gündüz, K. Huang, W. Saad, M. Bennis, A. V. Feljan, and H. V. Poor, "Guest editorial special issue on distributed learning over wireless edge networks-part II," *IEEE Journal on Selected Areas in Communications*, vol. 40, no. 2, pp. 445–448, Jan. 2022.

[2] M. Chen, Z. Yang, W. Saad, C. Yin, H. V. Poor, and S. Cui, "A joint learning and communications framework for federated learning over wireless networks," *IEEE Transactions on Wireless Communications*, vol. 20, no. 1, pp. 269–283, Oct., 2021.

[3] M. Chen, H. V. Poor, W. Saad, and S. Cui, "Wireless communications for collaborative federated learning," *IEEE Communications Magazine*, vol. 58, no. 12, pp. 48–54, Dec. 2020.

[4] S. Wang, S. Hosseinalipour, V. B. Aggarwal, C. G. Brinton, D. J. Love, W. Su, and M. Chiang, "Towards cooperative federated learning over heterogeneous edge/fog networks," *IEEE Communications Magazine*, vol. 61, no. 12, pp. 54–60, May 2023.

[5] S. Guo, T. Zhang, H. Yu, X. Xie, L. Ma, T. Xiang, and Y. Liu, "Byzantine-resilient decentralized stochastic gradient descent," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 32, no. 6, pp. 4096–4106, Oct. 2022.

[6] S. Kalra, J. Wen, J. C. Cresswell, M. Volkovs, and H. R. Tizhoosh, "Decentralized federated learning through proxy model sharing," *Nature Communications*, vol. 14, pp. 2899, Mar. 2023.

[7] Y. Gao, L. Zhang, L. Wang, K-K Choo, and R. Zhang, "Privacy-preserving and reliable decentralized federated learning," *IEEE Transactions on Services Computing*, vol. 16, no. 4, pp. 2879–2891, March 2023.

[8] S. Chen, D. Yu, Y. Zou, J. Yu, and X. Cheng, "Decentralized wireless federated learning with differential privacy," *IEEE Transactions on Industrial Informatics*, vol. 18, no. 9, pp. 6273–6282, Jan. 2022.

[9] A. Gouissem, K. Abualsaud, E. Yaacoub, T. Khattab, and M. Guizani, "Collaborative byzantine resilient federated learning," *IEEE Internet of Things Journal*, vol. 10, no. 18, pp. 15887–15899, Apr. 2023.

[10] Z. Yang, Y. Shi, Y. Zhou, Z. Wang, and K. Yang, "Trustworthy federated learning via blockchain," *IEEE Internet of Things Journal*, vol. 10, no. 1, pp. 92–109, Aug. 2023.

[11] J. So, B. Güler, and A. S. Avestimehr, "Byzantine-resilient secure federated learning," *IEEE Journal on Selected Areas in Communications*, vol. 39, no. 7, pp. 2168–2181, Dec. 2021.

[12] Y. Li, X. Wang, R. Sun, X. Xie, S. Ying, and S. Ren, "Trustiness-based hierarchical decentralized federated learning," *Knowledge-Based Systems*, vol. 276, pp. 110763, Sept. 2023.

[13] Y. Wang, M. Chen, T. Luo, W. Saad, D. Niyato, H. V. Poor, and S. Cui, "Performance optimization for semantic communications: An attention-based reinforcement learning approach," *IEEE Journal on Selected Areas in Communications*, vol. 40, no. 9, pp. 2598–2613, July. 2022.

[14] M. Lee, G. Yu, and G. Y. Li, "Graph embedding-based wireless link scheduling with few training samples," *IEEE Transactions on Wireless Communications*, vol. 20, no. 4, pp. 2282–2294, Dec. 2021.

[15] N. Yang, S. Wang, M. Chen, C.G. Brinton, and C. Yin, "Energy efficient collaborative federated learning design: A graph neural network based approach," in *2023 IEEE Global Communications Conference (Globecom)*, Kuala Lumpur, Malaysia, Dec. 2023.

[16] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, Nov. 1998.