Federated Learning in Cybersecurity Fields: Applications and Challenges

Beibei Li

Jan. 20, 2022

School of Cyber Science and Engineering, Sichuan University

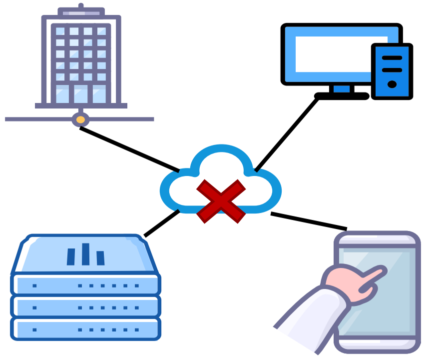
四川大學
SICHUAN UNIVERSITY

CONTENT >>

1. Federated Learning:

A New Distributed Learning Paradigm

- **For person:** People worry about the privacy leakage of their own data, which may result in possible financial crisis.
- **For company:** Companies worry about their trade secrets leakage from their data.
- **For government:** Governments have published some laws or policies to protect the privacy of people and stop the companies from holding too much private data.



Each entity refuses to share its own private data, therefore making itself **"Isolated Island"**. To break this dilemma, the **Federated Learning (called FL)** emerged.

The First FL System

☐ Google built the first Federated Learning system around the world in 2016, and applied the system on its mobile applications.

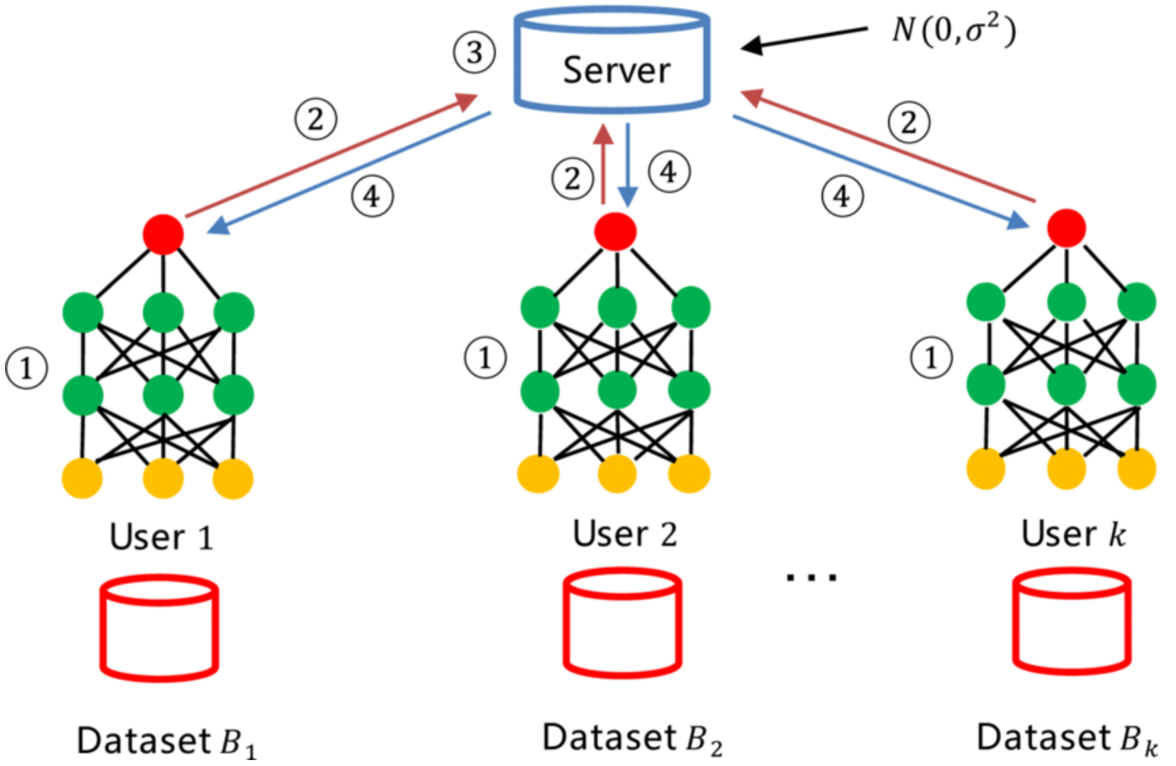
☐ Typical Federated Learning has 4 steps:

① Each client uses its local dataset to update the global model

② Each client sends the model gradient update back to the server

③ The server aggregates all the clients' model gradient updates to a global update and update the global model

④ The server sends the global model to each client



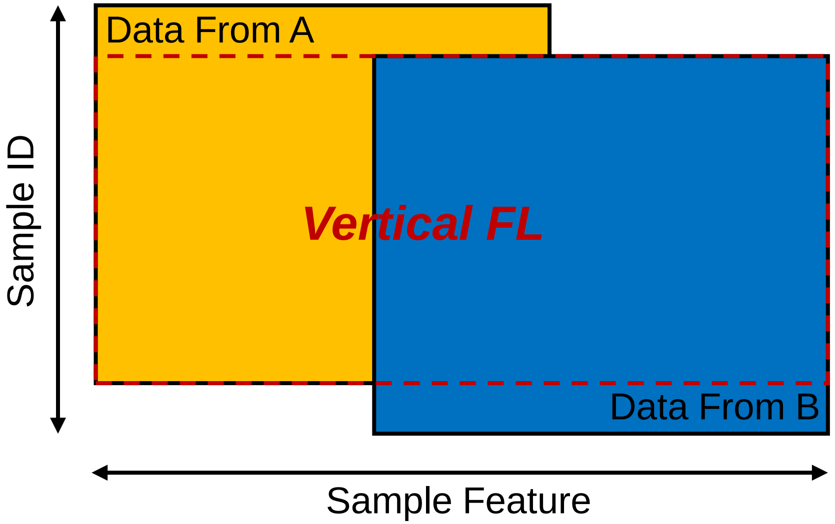
FL Framework

☐ Vertical FL

Vertical FL is also called **Sample-Aligned Federated Learning**.

In Vertical FL, the samples' ID in participants overlap with each other, which means the samples come from the same data source. While the samples' features in different participants are always vary from each other.

For instance, bank and market's client data in the same region suit vertical FL.

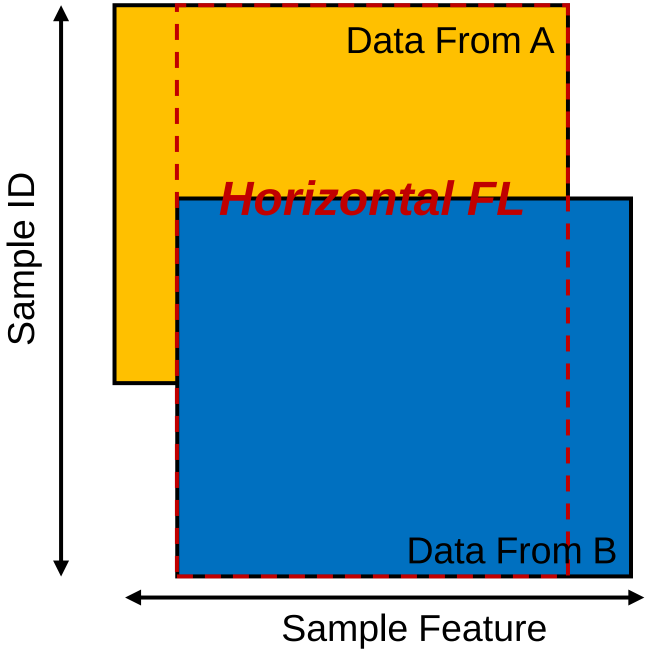


☐ Horizontal FL

Horizontal FL is also called **Feature-Aligned Federated Learning**.

In horizontal FL, the samples' features in different participants overlap with each other. While the samples' ID in different participants vary from each other.

For instance, 2 banks' client data in the different regions are suit horizontal FL.



Research Topics about Federated Learning

Many researchers have started to conduct research about FL.

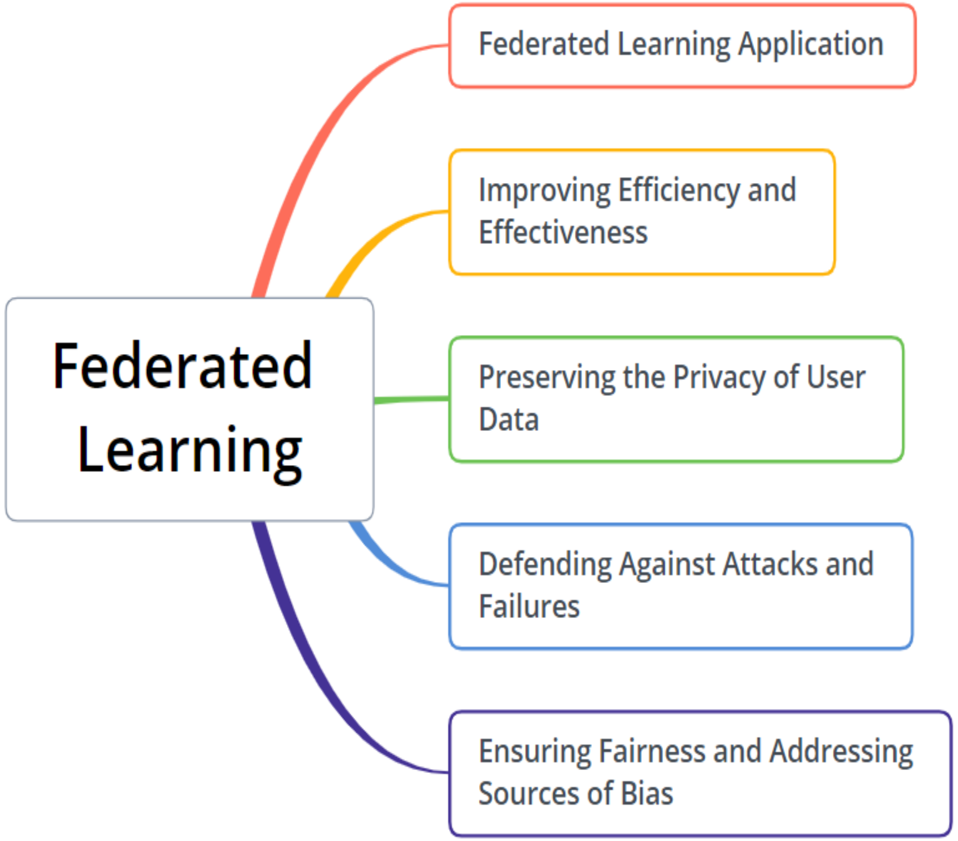
☐ **Federated Learning Application:** Research on Federated Learning application in different industrial scenarios.

☐ **Improving Efficiency and Effectiveness:** Research on how to improve the efficiency and effectiveness of Federated Learning.

☐ **Preserving the Privacy of User Data:** Research on how to enhance the privacy of user data in Federated Learning.

☐ **Defending Against Attacks and Failures:** Research on how to defend against attacks and failures in Federated Learning.

☐ **Ensuring Fairness and Addressing Sources of Bias:** Research on how to ensure fairness and address sources of bias



Federated Learning

Federated Learning Application

Improving Efficiency and Effectiveness

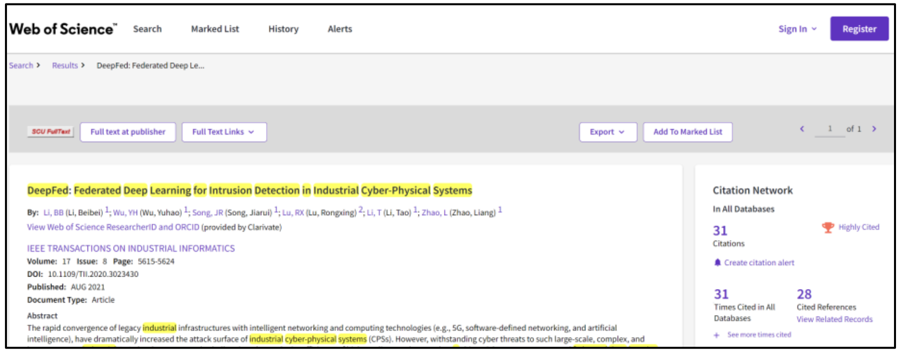
Preserving the Privacy of User Data

Defending Against Attacks and Failures

Ensuring Fairness and Addressing Sources of Bias

2. Federated Learning Application in Cybersecurity: Intrusion Detection in Industrial Cyber-Physical Systems

Beibei Li, Yuhao Wu, Jiarui Song, Rongxing Lu, Tao Li, Liang Zhao, "**DeepFed: Federated Deep Learning for Intrusion Detection in Industrial Cyber–Physical Systems**," in *IEEE Transactions on Industrial Informatics*, vol. 17, no. 8, pp. 5615-5624, Aug. 2021, doi: 10.1109/TII.2020.3023430. (**ESI Highly Cited Paper**)



DeepFed: Federated Deep Learning for Intrusion Detection in Industrial Cyber-Physical Systems

By: Li, BB (Li, Beibei) 1; Wu, YH (Wu, Yuhao) 1; Song, JR (Song, Jiarui) 1; Lu, RX (Lu, Rongxing) 2; Li, T (Li, Tao) 1; Zhao, L (Zhao, Liang) 1

View Web of Science ResearcherID and ORCID (provided by Clarivate)

IEEE TRANSACTIONS ON INDUSTRIAL INFORMATICS

Volume: 17 **Issue:** 8 **Page:** 5615-5624

DOI: 10.1109/TII.2020.3023430

Published: AUG 2021

Document Type: Article

Abstract

The rapid convergence of legacy industrial infrastructures with intelligent networking and computing technologies (e.g., 5G, software-defined networking, and artificial intelligence), have dramatically increased the attack surface of industrial cyber-physical systems (CPSs). However, withstanding cyber threats to such large-scale, complex, and

Citation Network

In All Databases

31 Citations

Highly Cited

Create citation alert

31 Times Cited in All Databases

28 Cited References

View Related Records

See more times cited

Ⅰ. Introduction

- ☐ **Industrial Cyber-Physical Systems (ICPSs):** ICPSs are **large-scale**, **geographically dispersed**, **federated**, **heterogeneous, life-critical** systems that comprise **sensors**, **actuators**, and **control and networking components**. These systems have multiple control loops, strict timing requirements, predictable network traffic, legacy components, and possibly wireless network segments.

- ☐ **Vulnerabilities of ICPSs:** Many industries exist in ICPS with **inherent security vulnerabilities** can be exploited. It give a sophisticated adversary the possibility to launch attacks. If it is serious, it may cause the network of industries to immediately disrupt the concerned processes to a catastrophe.

It is urgent to conduct some research works on intrusion detection in **a privacy-preserving way** and capable of **detecting multiple types of cyberattacks**.

Ⅰ. Introduction

☐ Existing Problems in Intrusion Detection for ICPSs:

a) High quality and large quantities of **training data** are often **hard to obtain** and cannot be easily shared and transmitted.

b) The owner of some highly **sensitive data** will also strongly object to the unrestricted calculation and use of the data, in which case, the data owner will only allow the data to be kept in his own hands, thus creating separate **islands of data**.

☐ Motivation of Federated Learning:

a) **Machine learning or deep learning models** can be trained by each data owner, then with the help of cloud server, a global model can be obtained through **model aggregation**.

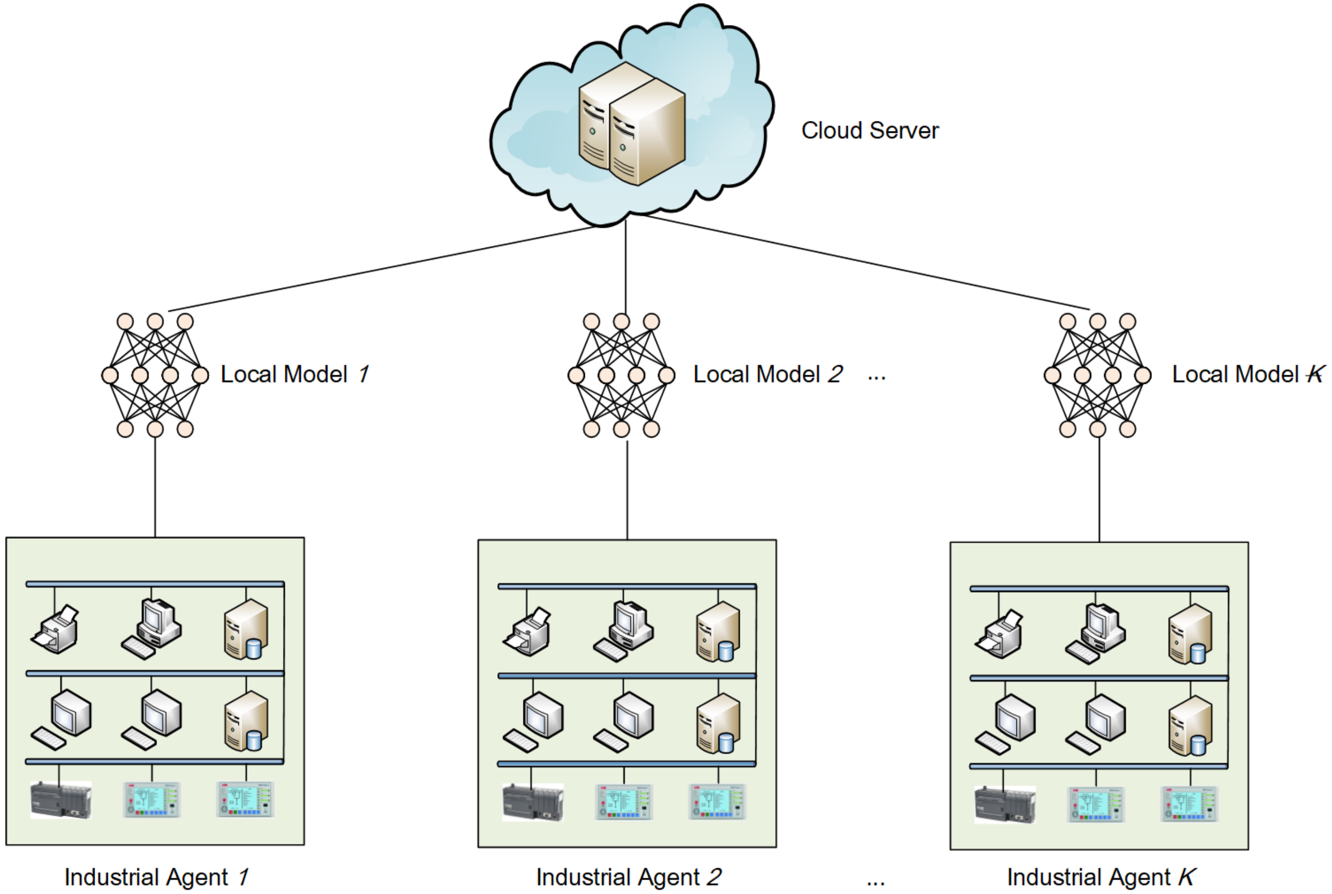
b) If only **parameters** were transferred between the device and the the communication would be extremely efficient.

c) It can **maximize the computing power of terminal devices** in cloud systems.

Ⅱ. System Model and Threat Model

A. System Model:

- ☐ The considered system consist of *k* **industrial agents** and a **cloud server**.
- ☐ **Industrial agents,** owners of ICPSs, with the same network traffic data structure that demand for collaboratively learning a **deep learning model** for **intrusion detection** .
- ☐ A **cloud server** that **aggregates** intrusion detection models trained by industrial agents in the system.



Ⅱ. System Model and Threat Model

B. Threat Model:

☐ **Industrial agents** are exposed to **many severe threats**:

a) **Reconnaissance** attacks, **Response injection** attacks

b) **Command injection** attacks, **Denial-of service** attacks

☐ **Threats during collaboratively model training:**

a) Leakage of industrial agents' **data**

b) Leakage of industrial agents' **model parameters**

☐ **Assumptions:**

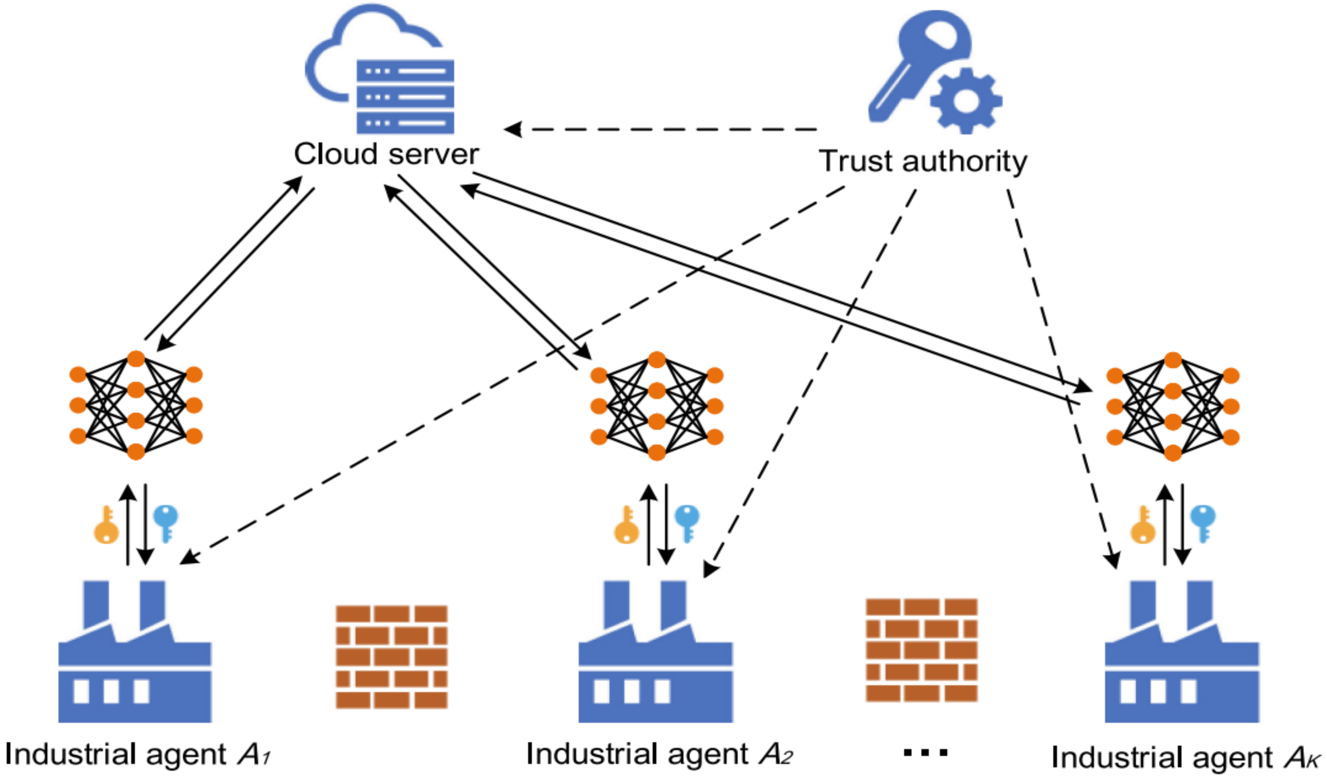
a) All **industrial agents** are **honest**, so that agents follow the negotiated rules and they share the final model parameters.

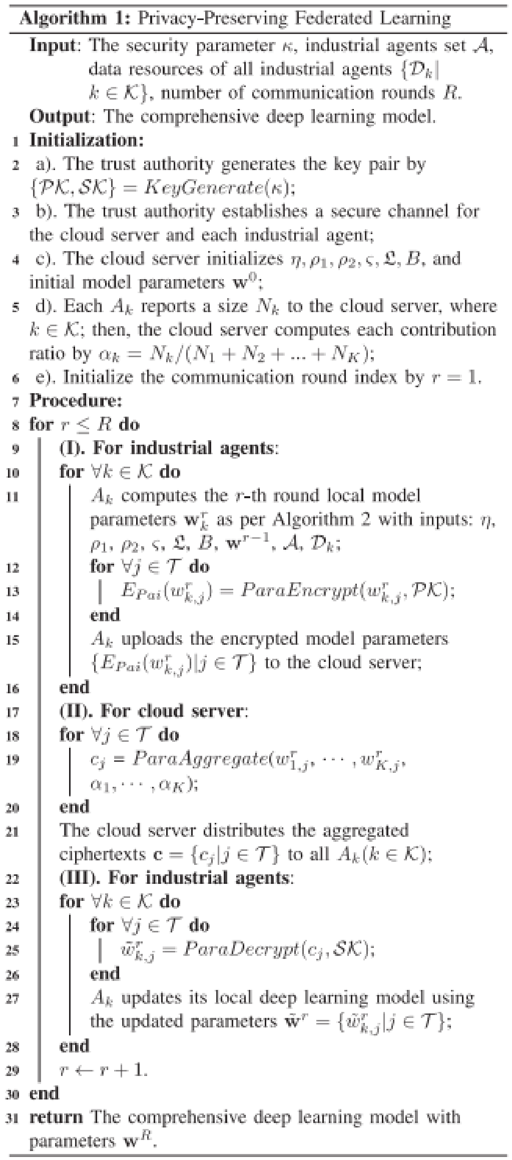
b) The **cloud server** is **semi-trust**, namely honest but curious, in that it perform the actions correctly, but attempt to obtain model parameters of industrial agents. Hence, **no leakage of parameters** from any industrial agents to the server **is allowed**.

Ⅲ. The Proposed DeepFed Scheme

A. Framework of DeepFed:

Industrial agents jointly train a deep learning model for intrusion detection using **federated learning**.





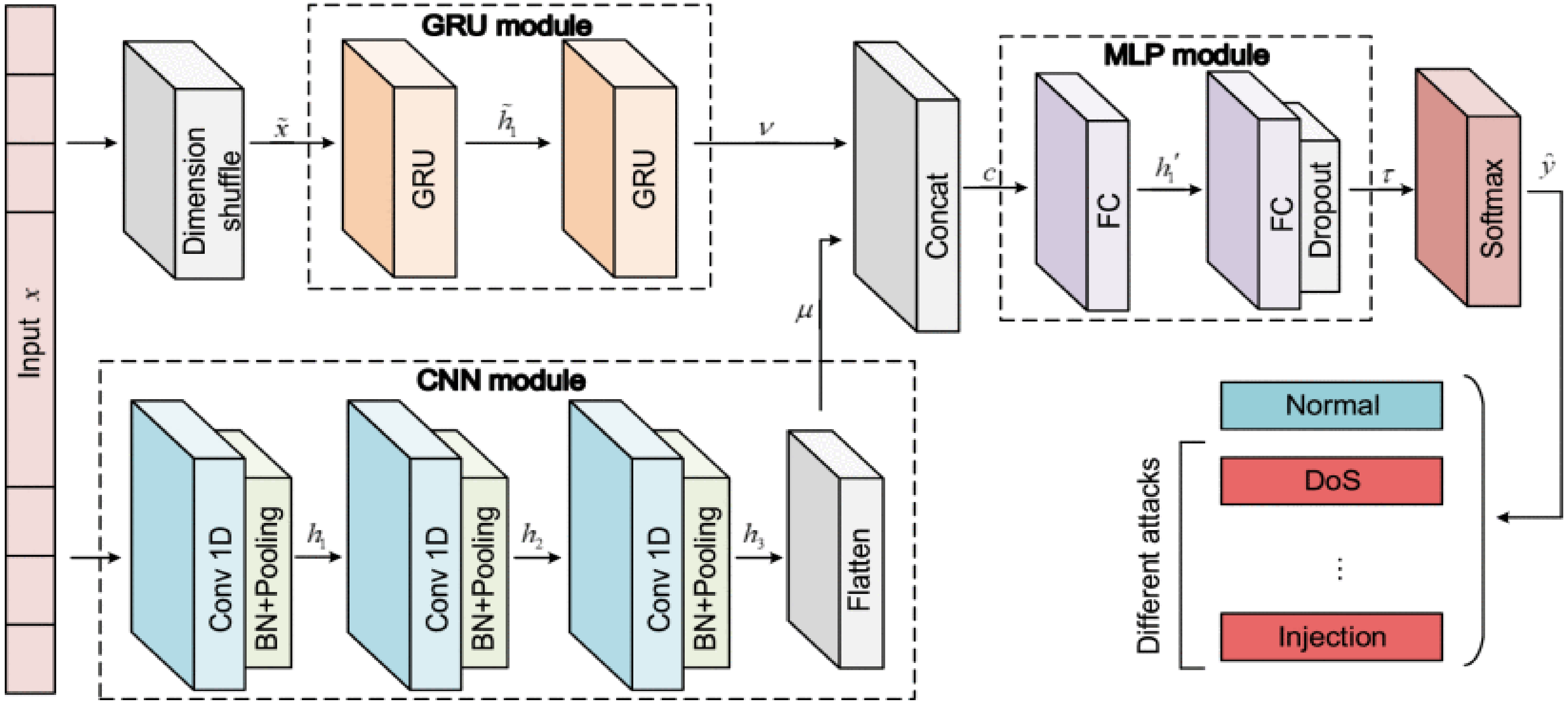
Algorithm 1: Privacy-Preserving Federated Learning

```
Input: The security parameter $\kappa$, industrial agents set $\mathcal{A}$,
       data resources of all industrial agents $\{\mathcal{D}_k |$
       $k \in \mathcal{K}\}$, number of communication rounds $R$.
Output: The comprehensive deep learning model.
Initialization:
 a). The trust authority generates the key pair by
 $\{\mathcal{PK}, \mathcal{SK}\} = KeyGenerate(\kappa)$;
 b). The trust authority establishes a secure channel for
 the cloud server and each industrial agent;
 c). The cloud server initializes $\eta, \rho_1, \rho_2, \varsigma, \mathfrak{L}, B$, and
 initial model parameters $\mathbf{w}^0$;
 d). Each $A_k$ reports a size $N_k$ to the cloud server, where
 $k \in \mathcal{K}$; then, the cloud server computes each contribution
 ratio by $\alpha_k = N_k/(N_1 + N_2 + ... + N_K)$;
 e). Initialize the communication round index by $r = 1$.
Procedure:
for $r \leq R$ do
    (I). For industrial agents:
    for $\forall k \in \mathcal{K}$ do
        $A_k$ computes the $r$-th round local model
        parameters $\mathbf{w}_k^r$ as per Algorithm 2 with inputs: $\eta$,
        $\rho_1$, $\rho_2$, $\varsigma$, $\mathfrak{L}$, $B$, $\mathbf{w}^{r-1}$, $\mathcal{A}$, $\mathcal{D}_k$;
        for $\forall j \in \mathcal{T}$ do
            $E_{Pai}(w_{k,j}^r) = ParaEncrypt(w_{k,j}^r, \mathcal{PK})$;
        end
        $A_k$ uploads the encrypted model parameters
        $\{E_{Pai}(w_{k,j}^r) | j \in \mathcal{T}\}$ to the cloud server;
    end
    (II). For cloud server:
    for $\forall j \in \mathcal{T}$ do
        $c_j = ParaAggregate(w_{1,j}^r, \cdots, w_{K,j}^r,$
        $\alpha_1, \cdots, \alpha_K)$;
    end
    The cloud server distributes the aggregated
    ciphertexts $\mathbf{c} = \{c_j | j \in \mathcal{T}\}$ to all $A_k (k \in \mathcal{K})$;
    (III). For industrial agents:
    for $\forall k \in \mathcal{K}$ do
        for $\forall j \in \mathcal{T}$ do
            $\tilde{w}_{k,j}^r = ParaDecrypt(c_j, \mathcal{SK})$;
        end
        $A_k$ updates its local deep learning model using
        the updated parameters $\tilde{\mathbf{w}}^r = \{\tilde{w}_{k,j}^r | j \in \mathcal{T}\}$;
    end
    $r \leftarrow r + 1$.
end
return The comprehensive deep learning model with
parameters $\mathbf{w}^R$.
```

Ⅲ. The Proposed DeepFed Scheme

B. CNN-GRU Model for Intrusion Detection:

- **Model Architecture:** The proposed CNN-GRU model is mainly composed by a **CNN** (convolutional neural network) module and a **GRU** (gated recurrent unit) module, followed by a **MLP** (multilayer perceptron) module, and then a softmax layer.



Ⅲ. The Proposed DeepFed Scheme

C. Paillier-Based Secure Communication Protocol:

- **This protocol is designed to achieve model parameter aggregation over ciphertexts, and protect the model parameters while transmission between the cloud server and industrial agents.**

- **Encryption:**

$$E_{Pai}(m) = g^{f(m)} \cdot r^{n} \bmod n^{2} = g^{m'} \cdot r^{n} \bmod n^{2}$$

- **Aggregation:**

$$\begin{aligned} c &= \prod_{i=1}^{K} E_{Pai}^{\alpha_i}(m_i) \\ &= g^{\alpha_1 m'_1} r_1^{\alpha_1 n} \cdot g^{\alpha_2 m'_2} r_2^{\alpha_2 n} \cdot g^{\alpha_K m'_K} r_K^{\alpha_K n} \bmod n^{2} \\ &= g^{\sum_{i=1}^{K} \alpha_i m'_i} \cdot \prod_{i=1}^{K} r_i^{\alpha_i n} \bmod n^{2}. \end{aligned}$$

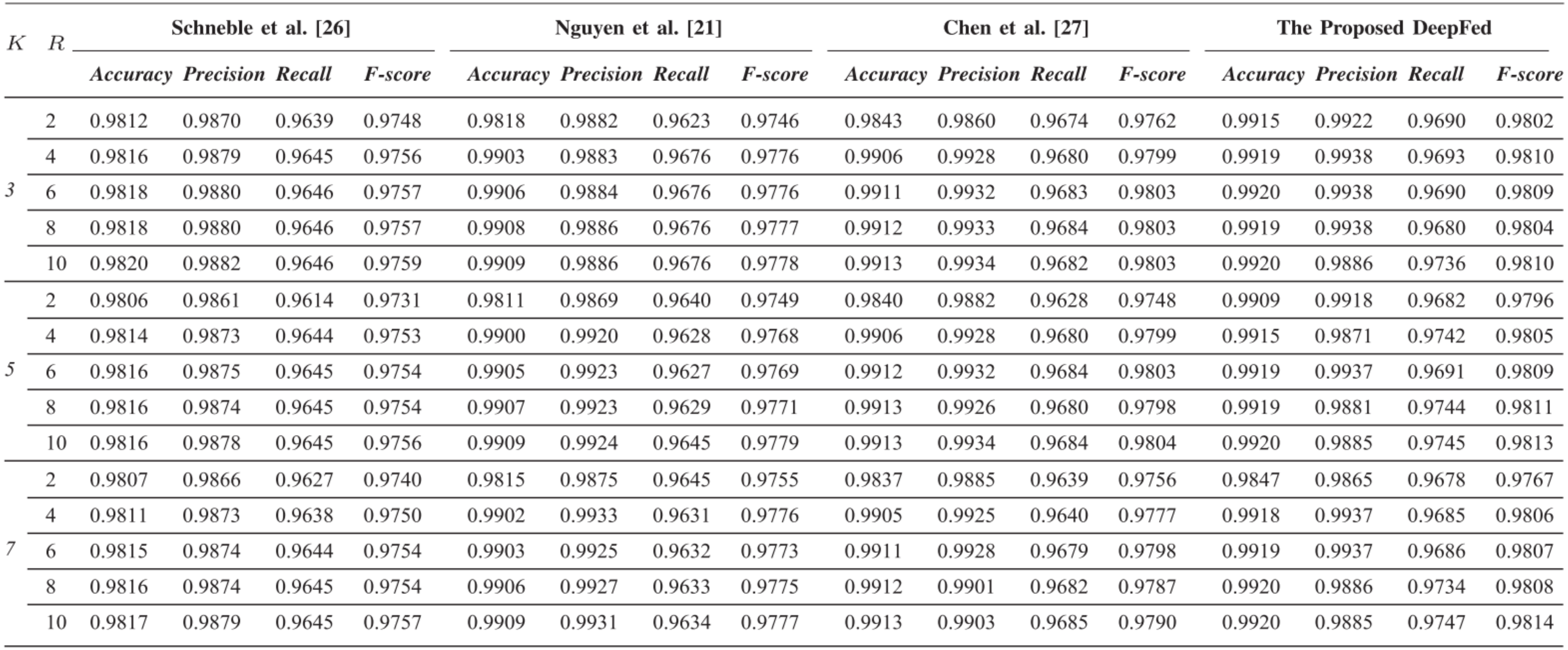
Decryption:

$$\begin{aligned} \tilde{m}'_{\text{sum}} &= L(c \bmod n^{2}) \cdot \mu \bmod n \\ &= \frac{L(g^{\sum_{i=1}^{K} \alpha_i m'_i} \cdot \prod_{i=1}^{K} r_i^{\alpha_i n} \bmod n^{2})}{L(g^{\lambda} \bmod n^{2})} \bmod n \\ &= \sum_{i=1}^{K} \alpha_i m'_i \bmod n. \end{aligned}$$

Ⅳ. Experiments and Evaluation

A. Performance Comparison with State-of-the-Art Studies: The performance of different models, including *FL-MLP, FL-CNN,FL-GRU and FL-CNN-GRU,* with different communication rounds under 3 different scenarios: 3 agents, 5 agents and 7 agents

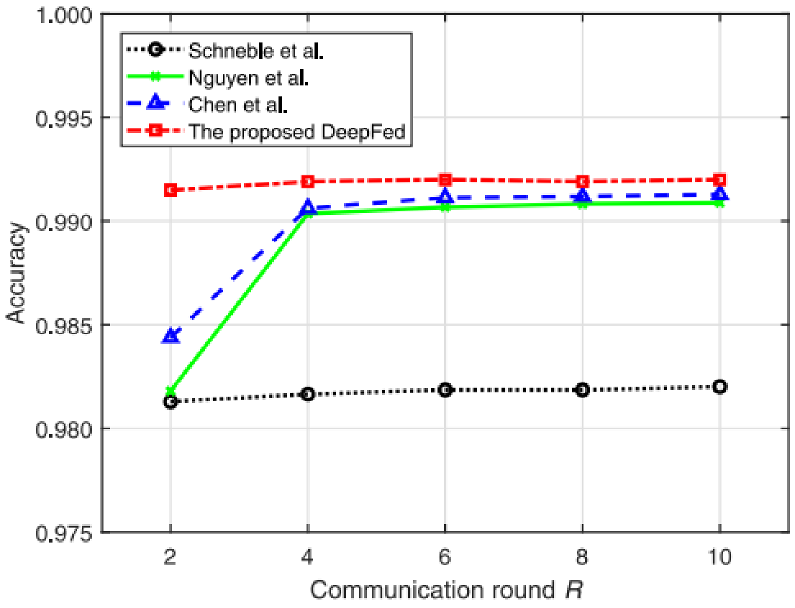
NUMERICAL RESULTS OF INTRUSION DETECTION MODELS WITH VARYING COMMUNICATION ROUNDS UNDER THREE DIFFERENT SCENARIOS



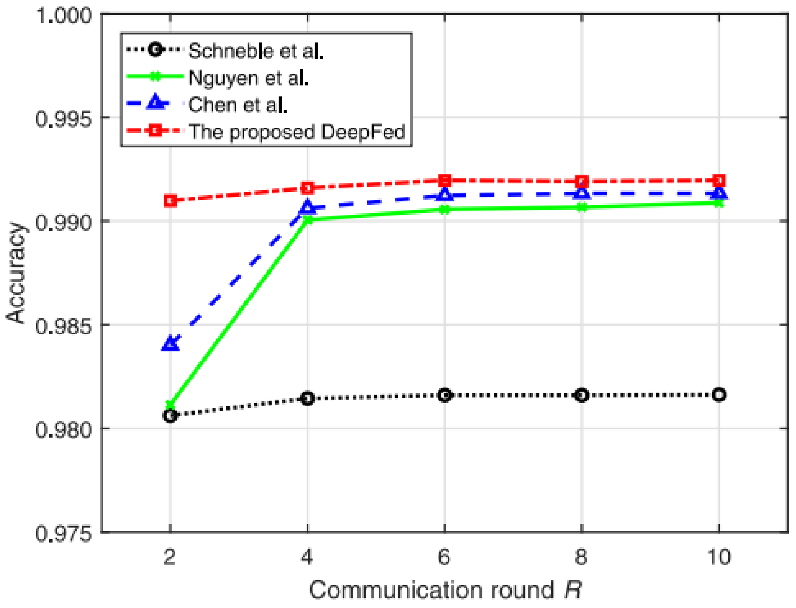
K	R	Schneble et al. [26]				Nguyen et al. [21]				Chen et al. [27]				The Proposed DeepFed			
		Accuracy	*Precision*	*Recall*	*F-score*	*Accuracy*	*Precision*	*Recall*	*F-score*	*Accuracy*	*Precision*	*Recall*	*F-score*	*Accuracy*	*Precision*	*Recall*	*F-score*
3	2	0.9812	0.9870	0.9639	0.9748	0.9818	0.9882	0.9623	0.9746	0.9843	0.9860	0.9674	0.9762	0.9915	0.9922	0.9690	0.9802
	4	0.9816	0.9879	0.9645	0.9756	0.9903	0.9883	0.9676	0.9776	0.9906	0.9928	0.9680	0.9799	0.9919	0.9938	0.9693	0.9810
	6	0.9818	0.9880	0.9646	0.9757	0.9906	0.9884	0.9676	0.9776	0.9911	0.9932	0.9683	0.9803	0.9920	0.9938	0.9690	0.9809
	8	0.9818	0.9880	0.9646	0.9757	0.9908	0.9886	0.9676	0.9777	0.9912	0.9933	0.9684	0.9803	0.9919	0.9938	0.9680	0.9804
	10	0.9820	0.9882	0.9646	0.9759	0.9909	0.9886	0.9676	0.9778	0.9913	0.9934	0.9682	0.9803	0.9920	0.9886	0.9736	0.9810
5	2	0.9806	0.9861	0.9614	0.9731	0.9811	0.9869	0.9640	0.9749	0.9840	0.9882	0.9628	0.9748	0.9909	0.9918	0.9682	0.9796
	4	0.9814	0.9873	0.9644	0.9753	0.9900	0.9920	0.9628	0.9768	0.9906	0.9928	0.9680	0.9799	0.9915	0.9871	0.9742	0.9805
	6	0.9816	0.9875	0.9645	0.9754	0.9905	0.9923	0.9627	0.9769	0.9912	0.9932	0.9684	0.9803	0.9919	0.9937	0.9691	0.9809
	8	0.9816	0.9874	0.9645	0.9754	0.9907	0.9923	0.9629	0.9771	0.9913	0.9926	0.9680	0.9798	0.9919	0.9881	0.9744	0.9811
	10	0.9816	0.9878	0.9645	0.9756	0.9909	0.9924	0.9645	0.9779	0.9913	0.9934	0.9684	0.9804	0.9920	0.9885	0.9745	0.9813
7	2	0.9807	0.9866	0.9627	0.9740	0.9815	0.9875	0.9645	0.9755	0.9837	0.9885	0.9639	0.9756	0.9847	0.9865	0.9678	0.9767
	4	0.9811	0.9873	0.9638	0.9750	0.9902	0.9933	0.9631	0.9776	0.9905	0.9925	0.9640	0.9777	0.9918	0.9937	0.9685	0.9806
	6	0.9815	0.9874	0.9644	0.9754	0.9903	0.9925	0.9632	0.9773	0.9911	0.9928	0.9679	0.9798	0.9919	0.9937	0.9686	0.9807
	8	0.9816	0.9874	0.9645	0.9754	0.9906	0.9927	0.9633	0.9775	0.9912	0.9901	0.9682	0.9787	0.9920	0.9886	0.9734	0.9808
	10	0.9817	0.9879	0.9645	0.9757	0.9909	0.9931	0.9634	0.9777	0.9913	0.9903	0.9685	0.9790	0.9920	0.9885	0.9747	0.9814

Ⅳ. Experiments and Evaluation

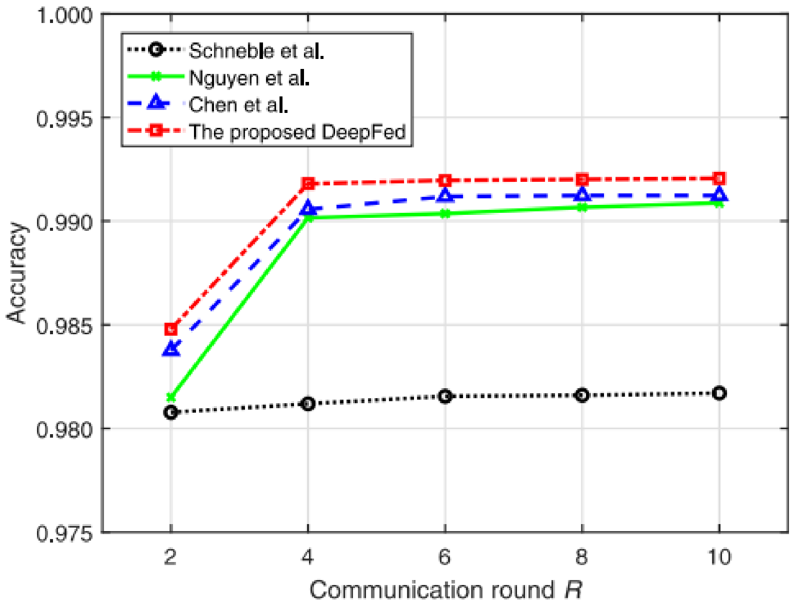
A. Performance Comparison with State-of-the-Art Studies: The accuracy of different models, including *FL-MLP, FL-CNN,FL-GRU and FL-CNN-GRU,* with different communication rounds under 3 different scenarios: 3 agents, 5 agents and 7 agents



3 agents



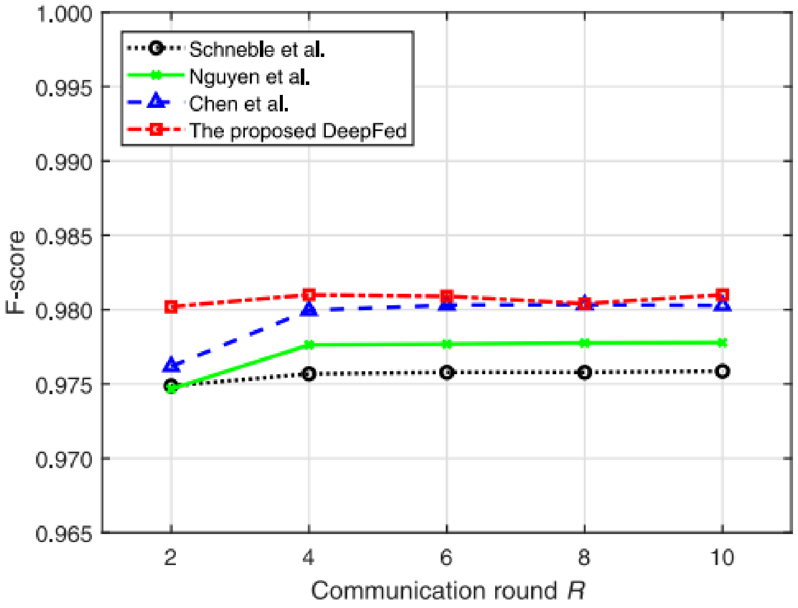
5 agents



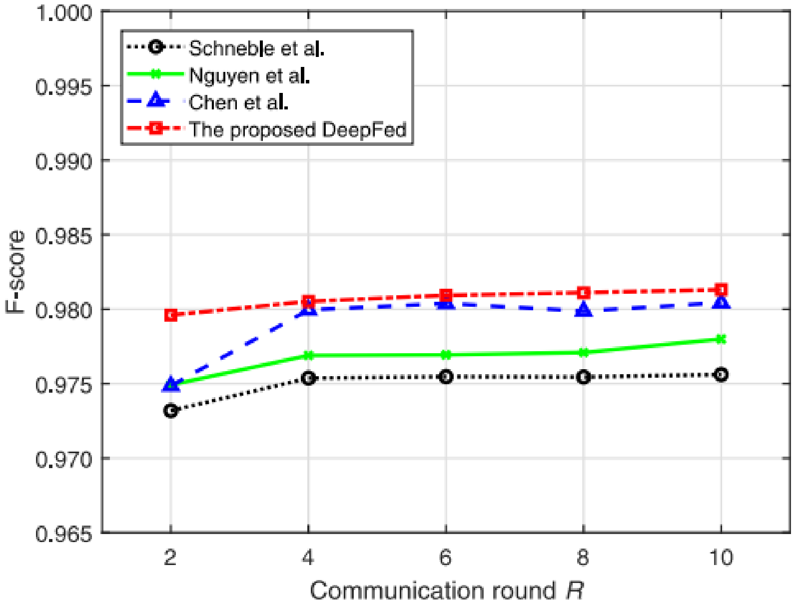
7 agents

Ⅳ. Experiments and Evaluation

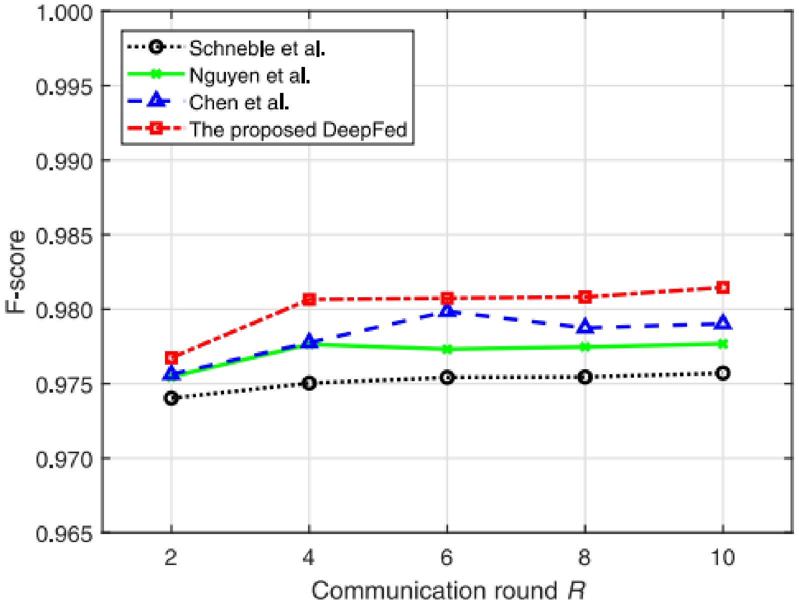
A. Performance Comparison with State-of-the-Art Studies: The F1-score of different models, including *FL-MLP, FL-CNN,FL-GRU and FL-CNN-GRU,* with different communication rounds under 3 different scenarios: 3 agents, 5 agents and 7 agents



3 agents



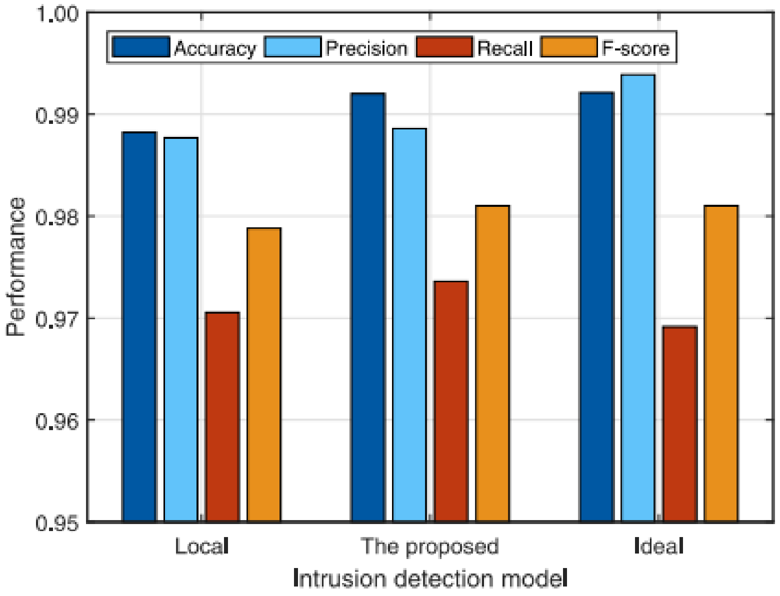
5 agents



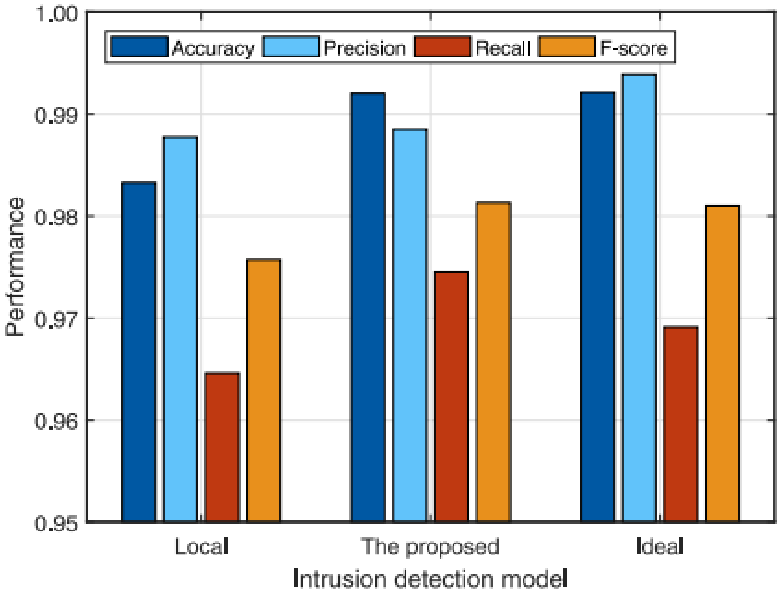
7 agents

Ⅳ. Experiments and Evaluation

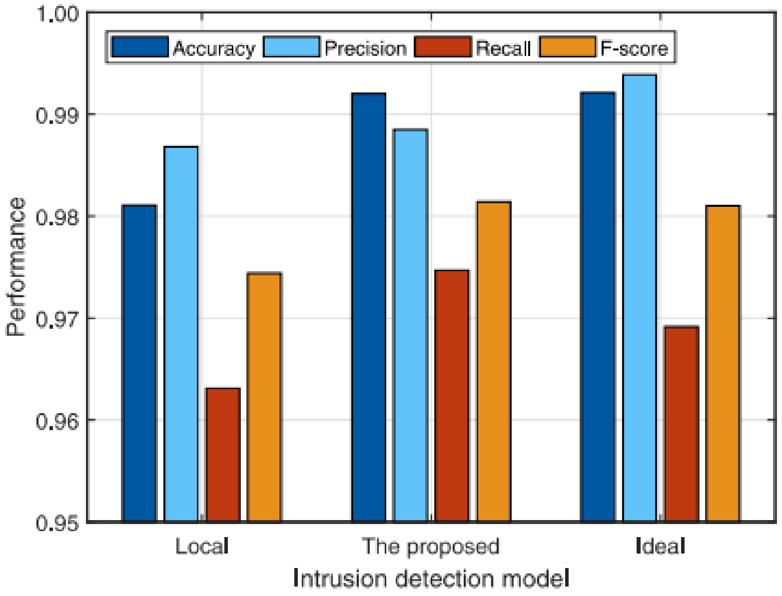
B. Performance comparison with local and ideal models: Performance comparison of local and ideal models under 3 different scenarios: 3 agents, 5 agents and 7 agents



3 agents



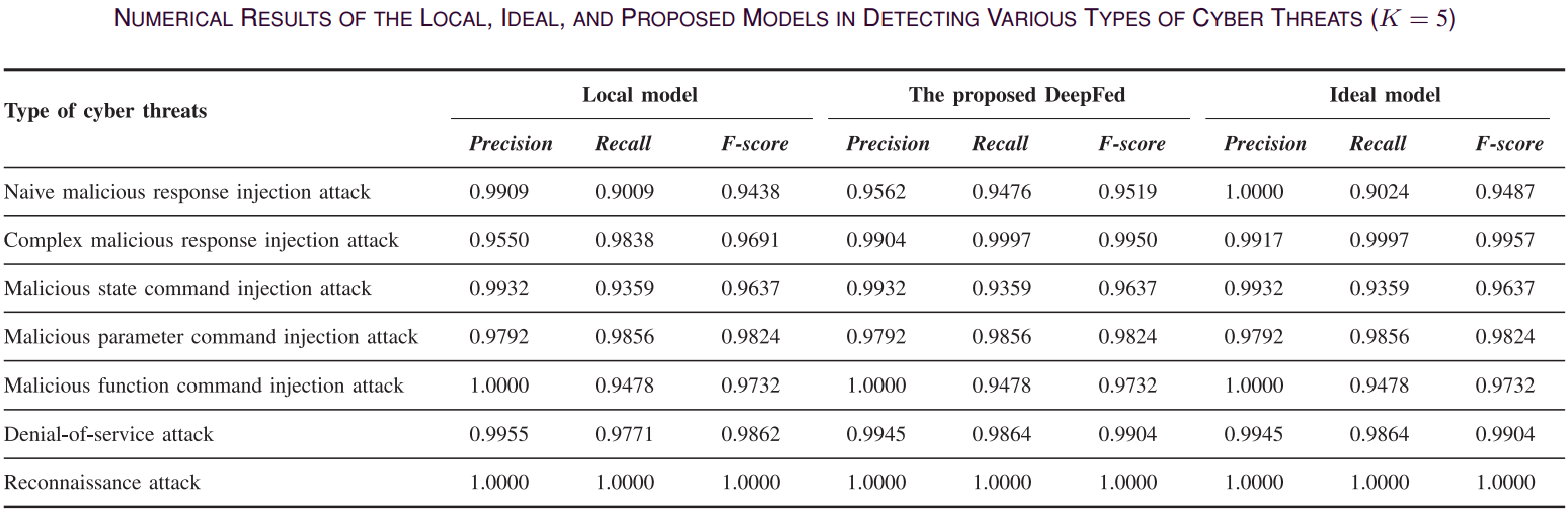
5 agents



7 agents

Ⅳ. Experiments and Evaluation

B. Performance comparison with local and ideal models: Performance comparison of local and ideal models in detecting various types of cyber threats



NUMERICAL RESULTS OF THE LOCAL, IDEAL, AND PROPOSED MODELS IN DETECTING VARIOUS TYPES OF CYBER THREATS ($K = 5$)

Type of cyber threats	**Local model**			**The proposed DeepFed**			**Ideal model**		
	Precision	***Recall***	***F-score***	***Precision***	***Recall***	***F-score***	***Precision***	***Recall***	***F-score***
Naive malicious response injection attack	0.9909	0.9009	0.9438	0.9562	0.9476	0.9519	1.0000	0.9024	0.9487
Complex malicious response injection attack	0.9550	0.9838	0.9691	0.9904	0.9997	0.9950	0.9917	0.9997	0.9957
Malicious state command injection attack	0.9932	0.9359	0.9637	0.9932	0.9359	0.9637	0.9932	0.9359	0.9637
Malicious parameter command injection attack	0.9792	0.9856	0.9824	0.9792	0.9856	0.9824	0.9792	0.9856	0.9824
Malicious function command injection attack	1.0000	0.9478	0.9732	1.0000	0.9478	0.9732	1.0000	0.9478	0.9732
Denial-of-service attack	0.9955	0.9771	0.9862	0.9945	0.9864	0.9904	0.9945	0.9864	0.9904
Reconnaissance attack	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000

Ⅴ. Conclusion

- ☐ First, a novel deep learning-based **intrusion detection scheme**, namely **DeepFed** is proposed for **ICPSs**, which can identify multiple attacks existing in ICPSs.

- ☐ Second, **federated learning** is employed to jointly train a deep learning model with good performance that takes advantages of **CNN** and **GRU**. Meanwhile, **data privacy** of industrial agents can be **protected** during the collaborative training process.

- ☐ Third, model parameters of all industrial agents are preserved by using **homomorphic encryption**, so that **model parameter leakage** can be **avoided** during the communication between industrial agents and cloud server.

3. Security Challenges in Federated Learning: Byzantine Attacks and Defenses

Beibei Li, Peiran Wang, Hanyuan Huang, Shang Ma, Yukun Jiang, "**FLPhish: Reputation-based Phishing Byzantine Defense in Ensemble Federated Learning**," *2021 IEEE Symposium on Computers and Communications (ISCC)*, 2021, pp. 1-6, doi: 10.1109/ISCC 53001.2021.9631506. (**Best Paper Award**)

IEEE ISCC 2021

BEST PAPER AWARD

Beibei Li, Peiran Wang, Hanyuan Huang, Shang Ma and Yukun Jiang

as Authors of the paper entitled

"FLPhish: Reputation-Based Phishing Byzantine Defense in Ensemble Federated Learning"

26th IEEE Symposium on Computers and Communications
(IEEE ISCC 2021)
September 5-8, 2021 Athens, Greece

Eirini Eleni Tsiropoulou

Prof. Gang Li
Program Committee Co-Chair

Prof. Eirini Eleni Tsiropoulou
Program Committee Co-Chair

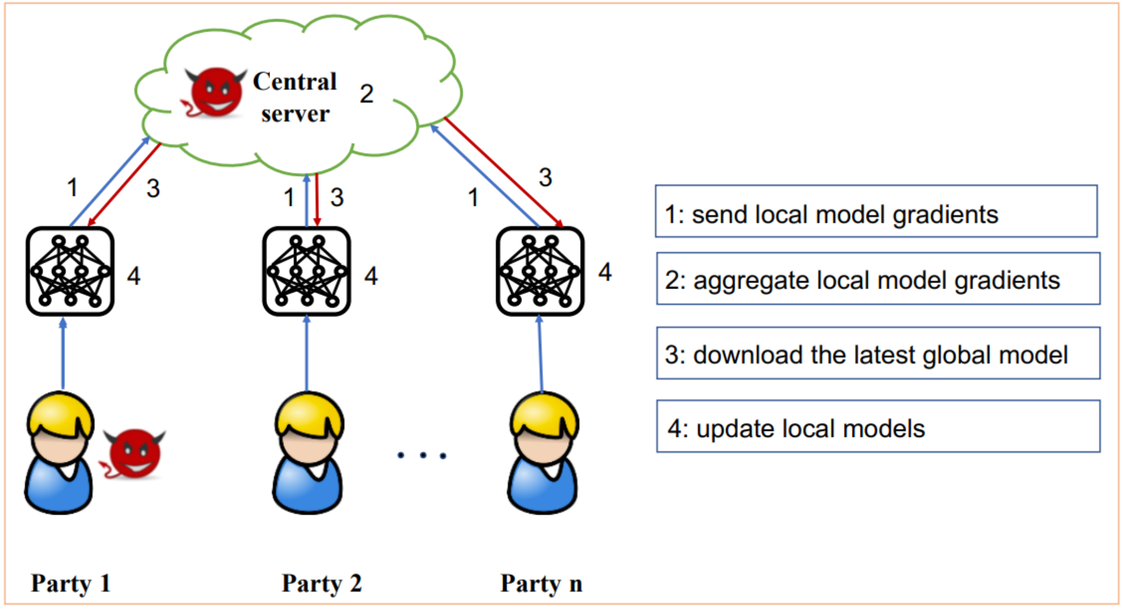
Prof. Theofanis Raptis
Program Committee Co-Chair

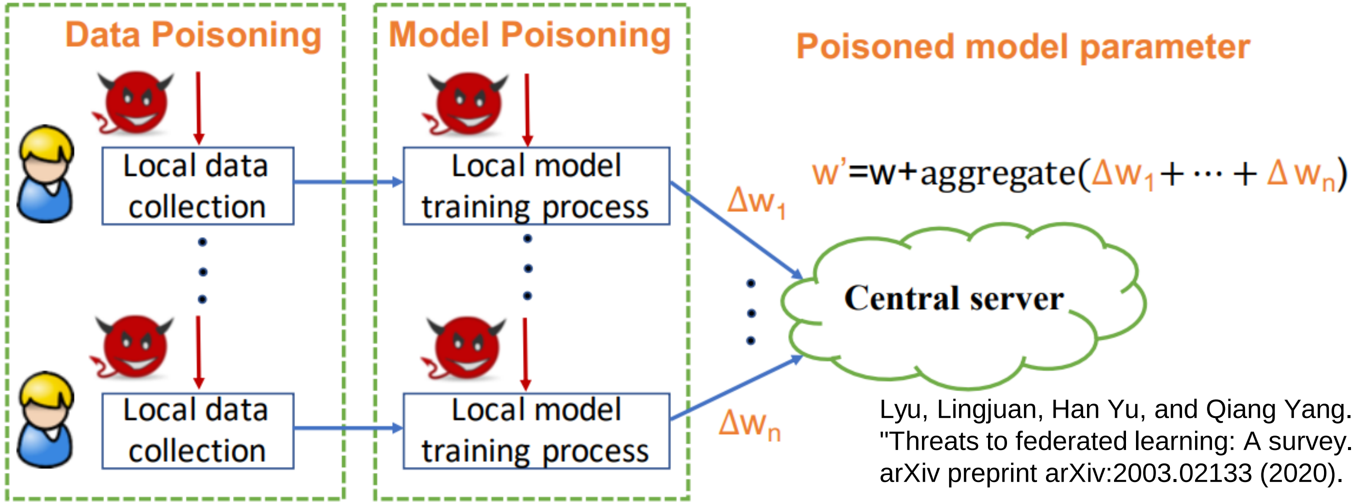
Ⅰ. Introduction

Typical FL faces challenges from Byzantine attacks. Malicious Byzantine attackers can launch attacks in two major way:

- ☐ Data Poisoning Attack: Data poisoning attack is launched during local data collection by label flipping attacks etc.
- ☐ Model Poisoning Attack: model poisoning attack is launched during local model training process

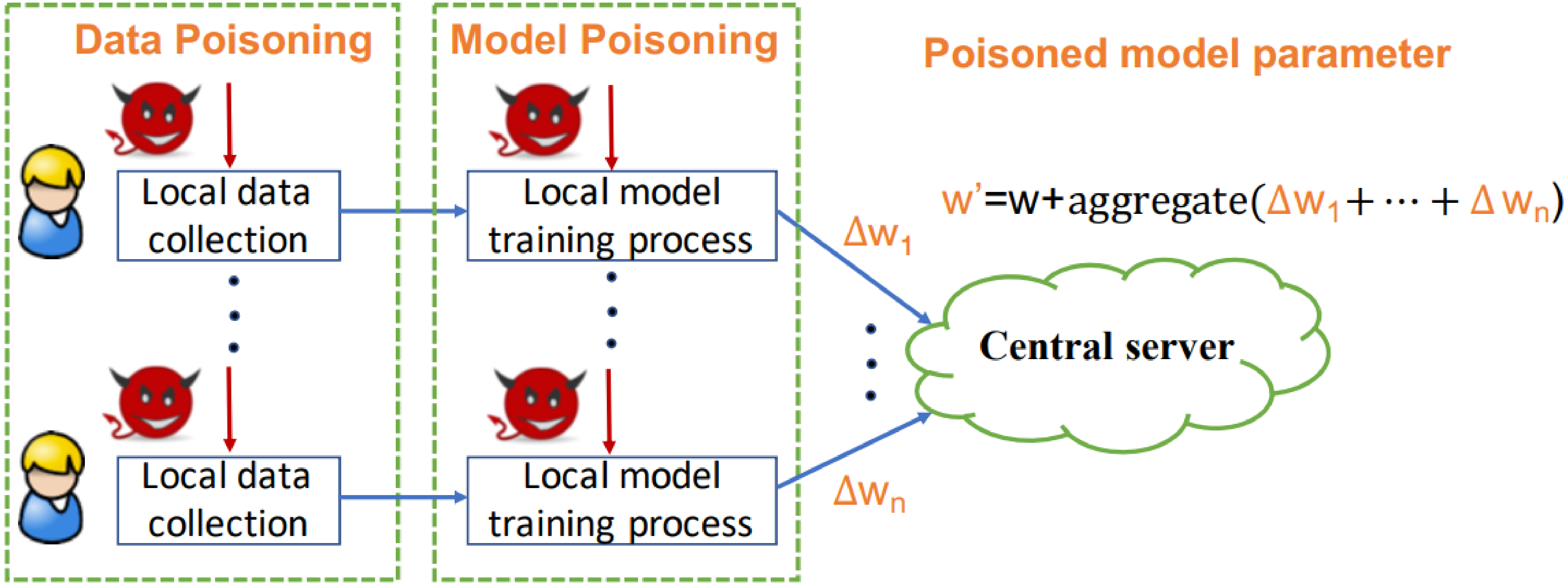
The intention of the attack is to violate central model





Lyu, Lingjuan, Han Yu, and Qiang Yang. "Threats to federated learning: A survey." arXiv preprint arXiv:2003.02133 (2020).

Ⅰ. Introduction



Poisoned model parameter

$w' = w + \text{aggregate}(\Delta w_1 + \cdots + \Delta w_n)$

Typical FL has many drawbacks:

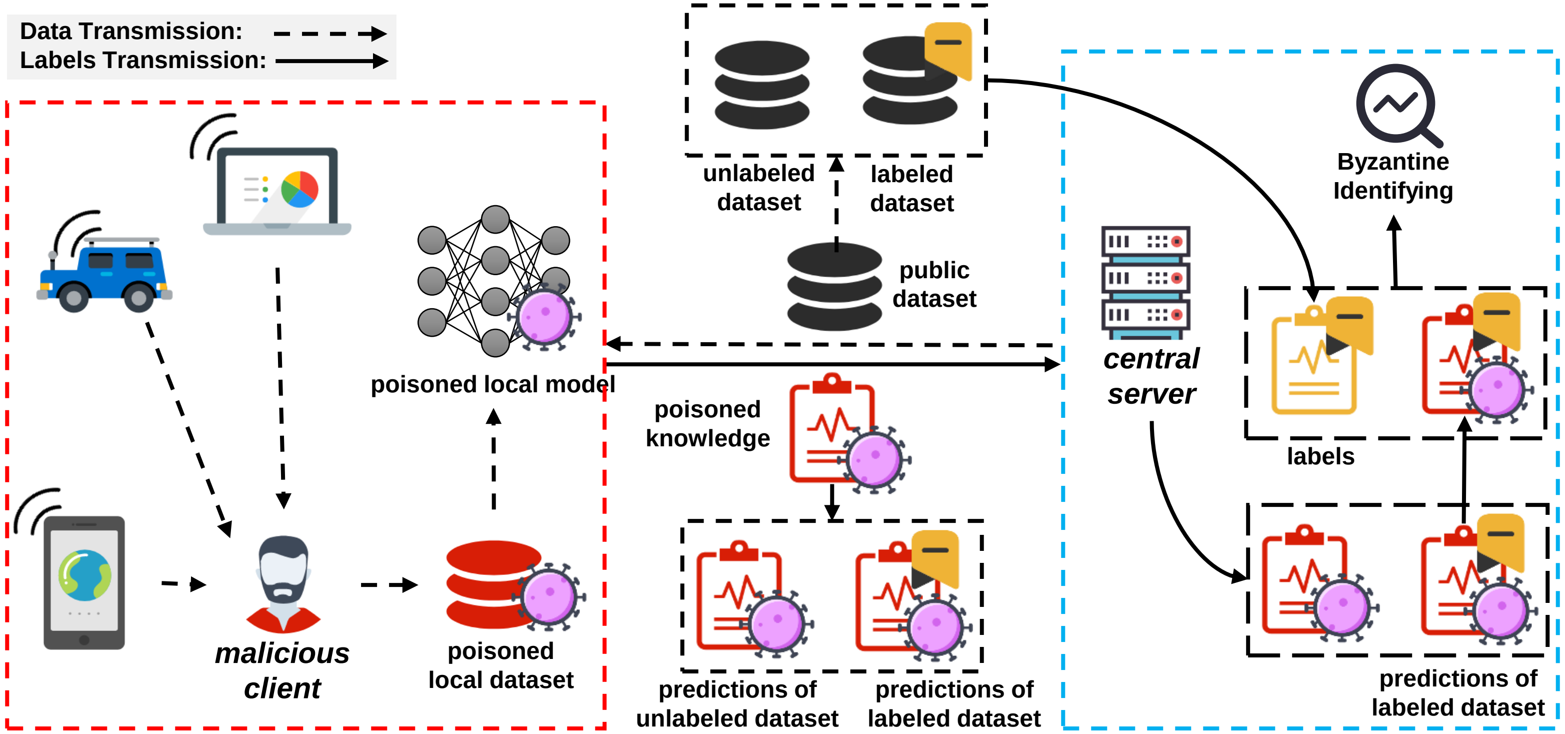
☐ Heavy network cost

☐ Each FL client should use the same type of deep learning model, such as CNN, LSTM ...

☐ Vulnerable to malicious Byzantine attacks

Lyu, Lingjuan, Han Yu, and Qiang Yang. "Threats to federated learning: A survey." arXiv preprint arXiv:2003.02133 (2020).

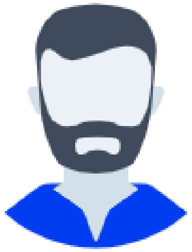
Ⅱ. The Proposed FLPhish Scheme



```
Algorithm 3 Phishing Mechanism
Input: the ensemble of clients $C$ with local dataset $d_i$, $i = 1, 2, 3, ..., u$; a central server $s$ with unlabeled dataset $D$ and labeled dataset $B$; number of training iterations $T$; unlabeled batch size $n$;labeled batch size $m$.
Output: output result
$\mathbf{m_i} \leftarrow$ each client $c_i$ train a local model using its local dataset $d_i$.
for $t$=1,2,3,...,$T$ do
    $s$ selects $D_t$ (containing $n$ samples) from $D$ and $B_t$ (containing $m$ samples) from $B$.
    for $i$=1,2,3,...,$u$ do
        $s$ sends $D_t$ and $B_t$ to $c_i$.
        $c_i$ makes predictions $\mathbf{k_i^t}$ of the $D_t$ and $B_t$.
        $c_i$ sends $\mathbf{k_i^t}$ to $s$.
        $s$ calculates the accuracy $a_i^t$ of the predictions of $B_t$ made by $c_i$ in $t$th procedure.
        if $a_i^t > r_q$ then
            $q_i = 1$.
            for $j$=$i$,$i$+1,$i$+2,...,$u$-1 do
                $(d, c, \mathbf{k^t}, q, a^t)_j \leftarrow (d, c, \mathbf{k^t}, q, a^t)_{j-1}$.
            end for
        end if
    end for
    $\mathbf{K_t} = KnowledgeEnsemble(\mathbf{k_1^t}, \mathbf{k_2^t}, \mathbf{k_3^t}, ..., \mathbf{k_u^t})$.
    $\mathbf{M} = ModelUpdate(\mathbf{K_t}, D_t, \mathbf{M})$.
end for
return $\mathbf{M}$
```

- ☐ To defend against Byzantine attacks, we present a Byzantine defense scheme, FLPhish. FLPhish combines the phishing mechanism and reputation mechanism to defend Byzantine attacks in Ensemble FL.

Ⅱ. Proposed FLPhish Scheme



FL Client:

- ☐ Collect local data
- ☐ Train a local model
- ☐ Make predictions of the broadcast dataset
- ☐ Transfer the predictions back to the FL server

FL Server:

- ☐ Preserve a public dataset
- ☐ Broadcast the public dataset
- ☐ Receive the predictions of the public dataset
- ☐ Aggregate the predictions of the public dataset

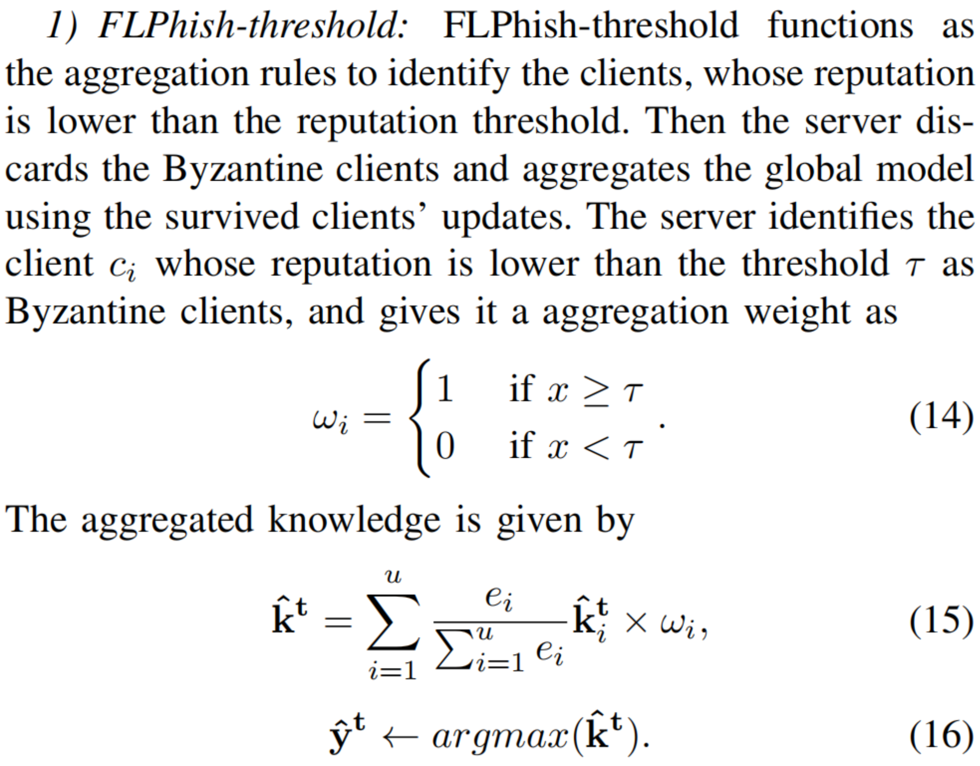
Algorithm 1 Ensemble FL

Input: the ensemble of clients C with local dataset d_i, $i = 1, 2, 3, ..., u$; a central server s with unlabeled dataset D; number of training iterations T; unlabeled batch size n;

Output:

```
m_i ← each client c_i train a local model using its local dataset d_i;
for t=1,2,3,...,T do
    s selects D_t (containing n samples) from D;
    for i=1,2,3,...,u do
        $s \xrightarrow{D_t} c_i$;
        c_i makes predictions $\mathbf{k_i^t}$ of the D_t;
        $c_i \xrightarrow{\mathbf{k_i^t}} s$;
    end for
    $Y_t = KnowledgeEnsemble(\mathbf{k_1^t}, \mathbf{k_2^t}, \mathbf{k_3^t}, ..., \mathbf{k_u^t})$;
    $\mathrm{M} = ModelUpdate(Y_t, D_t, \mathrm{M})$;
end for
return M.
```

Ⅱ. Proposed FLPhish Scheme



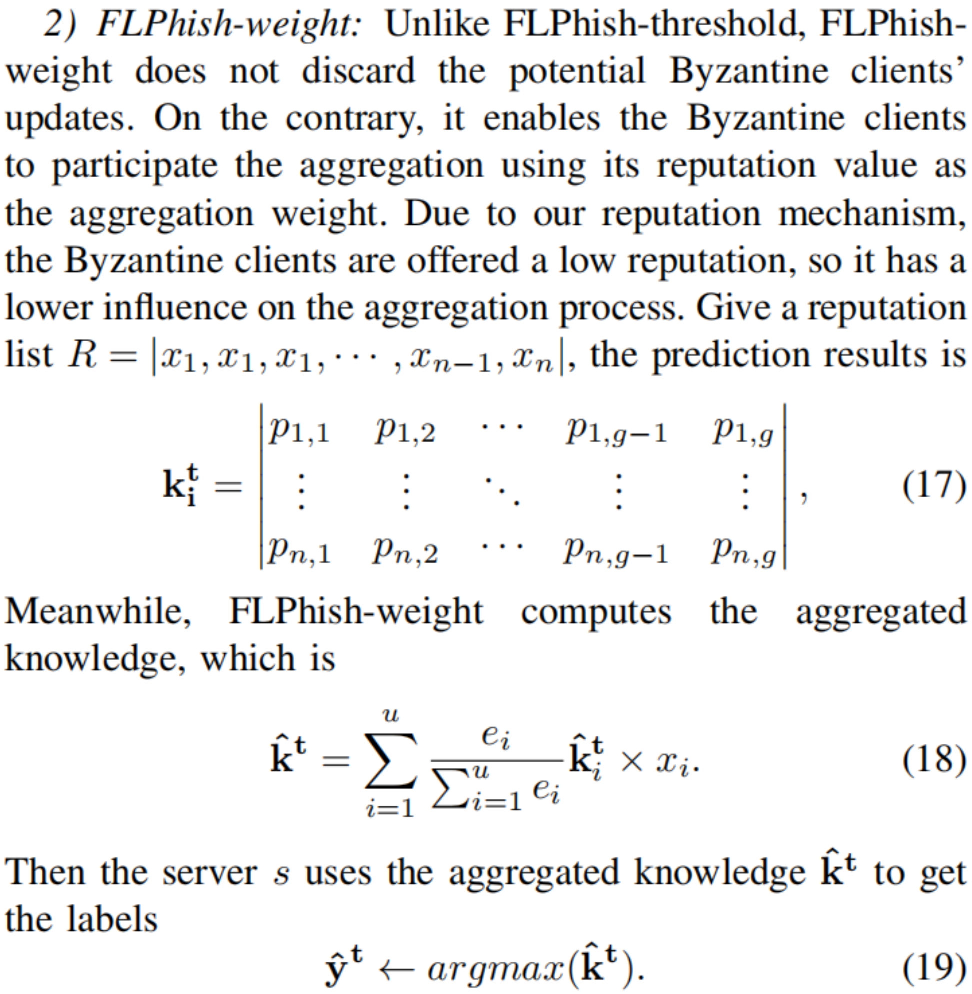
1) FLPhish-threshold: FLPhish-threshold functions as the aggregation rules to identify the clients, whose reputation is lower than the reputation threshold. Then the server discards the Byzantine clients and aggregates the global model using the survived clients' updates. The server identifies the client c_i whose reputation is lower than the threshold τ as Byzantine clients, and gives it a aggregation weight as

$$\omega_i = \begin{cases} 1 & \text{if } x \geq \tau \\ 0 & \text{if } x < \tau \end{cases}. \tag{14}$$

The aggregated knowledge is given by

$$\hat{\mathbf{k}}^{\mathbf{t}} = \sum_{i=1}^{u} \frac{e_i}{\sum_{i=1}^{u} e_i} \hat{\mathbf{k}}_i^{\mathbf{t}} \times \omega_i, \tag{15}$$

$$\hat{\mathbf{y}}^{\mathbf{t}} \leftarrow argmax(\hat{\mathbf{k}}^{\mathbf{t}}). \tag{16}$$



2) FLPhish-weight: Unlike FLPhish-threshold, FLPhish-weight does not discard the potential Byzantine clients' updates. On the contrary, it enables the Byzantine clients to participate the aggregation using its reputation value as the aggregation weight. Due to our reputation mechanism, the Byzantine clients are offered a low reputation, so it has a lower influence on the aggregation process. Give a reputation list $R = |x_1, x_1, x_1, \cdots, x_{n-1}, x_n|$, the prediction results is

$$\mathbf{k}_\mathbf{i}^\mathbf{t} = \begin{vmatrix} p_{1,1} & p_{1,2} & \cdots & p_{1,g-1} & p_{1,g} \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ p_{n,1} & p_{n,2} & \cdots & p_{n,g-1} & p_{n,g} \end{vmatrix}, \tag{17}$$

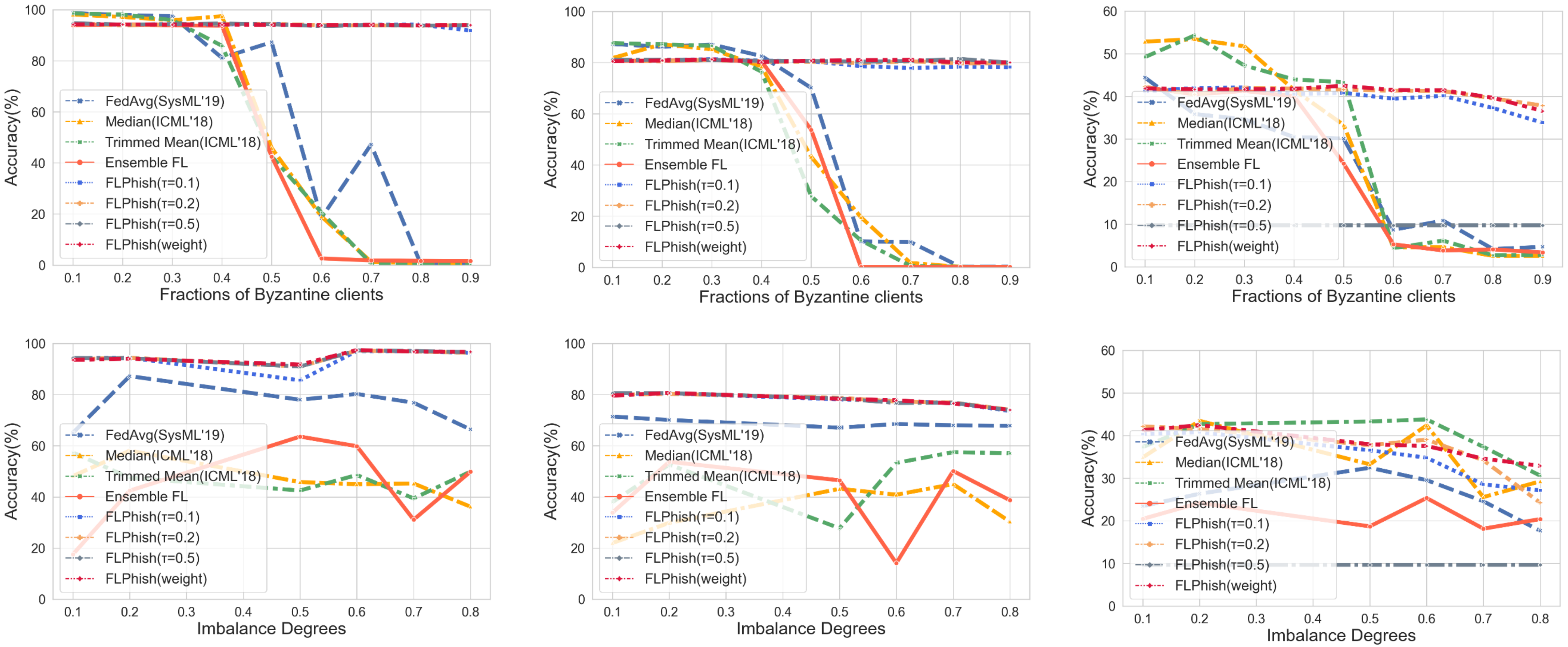
Meanwhile, FLPhish-weight computes the aggregated knowledge, which is

$$\hat{\mathbf{k}}^{\mathbf{t}} = \sum_{i=1}^{u} \frac{e_i}{\sum_{i=1}^{u} e_i} \hat{\mathbf{k}}_i^{\mathbf{t}} \times x_i. \tag{18}$$

Then the server s uses the aggregated knowledge $\hat{\mathbf{k}}^{\mathbf{t}}$ to get the labels

$$\hat{\mathbf{y}}^{\mathbf{t}} \leftarrow argmax(\hat{\mathbf{k}}^{\mathbf{t}}). \tag{19}$$

Ⅲ. Experiments Results



MNIST Fashion-MNIST CIFAR-10

Ⅳ. Conclusions

- In this paper, we have **designed an FL architecture**, Ensemble Federated Learning in this study, which allows knowledge transferring between the FL server and FL clients via the unlabeled dataset and FL clients' predictions of it.

- We have **crafted the FLPhish technique** to make Ensemble FL resistant to Byzantine attacks by using a labeled dataset as `bait' to detect malicious Byzantine clients.

- Furthermore, we have **proposed a reputation technique based on Bayesian inference** to determine a client's level of trust.

- We have also **presented two aggregation techniques, FLPhish-threshold and FLPhish-weight**, to improve FLPhish's performance.

Thanks for Listening