Privacy-Preserving Federated Distillation GAN for CIDSs in Industrial CPSs

1st Junwei Liang
College of Software Engineering
Shenzhen Institute of Information Technology
Shenzhen, China
Drimwa.l@gmail.com

2nd Muhammad Sadiq
College of International Relations
Shenzhen Institute of Information Technology
Shenzhen, China
sadiq_paec@yahoo.com

3rd Tie Cai
College of Software Engineering
Shenzhen Institute of Information Technology
Shenzhen, China
cait@sziit.edu.cn

GLOBECOM 2023 - 2023 IEEE Global Communications Conference | 979-8-3503-1090-0/23/$31.00 ©2023 IEEE | DOI: 10.1109/GLOBECOM54140.2023.10437148

***Abstract*—Intrusion Detection System (IDS) is an effective way to detect both internal and external abnormal behaviors, which has been widely deployed in industrial Cyber-Physical Systems (CPSs). However, due to the data island problem caused by the imperativeness of confidentiality of sensitive information, most existing IDSs are limited to be trained and evaluated in isolated CPSs, resulting in the cyber systems vulnerable to various newly-emerging attacks. In this article, a secure and collaborative IDS solution, called PFD-GAN, is proposed. Specifically, we firstly develop a novel semi-supervised IDS model by improving External Classifier (EC)-Generative Adversarial Network (GAN) with Wasserstein distance and label condition, to strengthen the classification performance through the use of synthetic data. Furthermore, Local Differential Privacy (LDP) is adopted to prevent against adversaries learning sensitive information in collaboration. Moreover, a Decentralized Federated Distillation (DFD)-based collaboration is designed, allowing multiple industrial CPSs to collectively build a comprehensive IDS to recognize the threats under the entire cyber systems without sharing a uniform template model. Experimental evaluation and theory analysis demonstrate that the proposed PFD-GAN is secure from the threats of privacy leaking and highly effective in detecting various types of attacks on industrial CPSs.**

***Index Terms*—IDS, CPSs, EC-GAN, LDP, DFD.**

I. Introduction

Industrial Cyber-Physical Systems (CPSs) are a rapid-growing research area in smart grids, autonomous transportation systems, and unmanned factories for recent years [1]. Unsurprisingly, the enormous usefulness of industrial CPSs has been stimulating adversaries to launch attacks on the systems. The deployment of Intrusion Detection Systems (IDSs) is one of the most important approaches to protect industrial CPSs against various threats [2]. In recent years, many studies have been conducted for the IDSs in industrial CPSs. For example, in [3], statistical IDS solutions have been presented by using signature-based analysis to check CPS nodes for compliance. Besides, anomaly-based analysis, i.e., a deep generative adversarial network, is further applied in [4] to recognize the intrusions that are not yet covered by the rules. However, class-imbalanced problem commonly exists in the locally collected datasets of industrial CPSs, thereby the most cutting-edge independent IDSs face a bottleneck in effectively detecting the anomalies with sparse training samples. In [5], a deep-federated learning-based decentralized detection method using an attentive aggregation is exploited, which is capable of parallel computing and can reliably identify the stealthy false data injection attacks on all the nodes simultaneously. Similarly, a federated deep learning scheme, named DeepFed, is proposed to detect cyber threats against industrial CPSs in [6], and a general global detection model, called EEFED, is designed for collaboratively improving the performance of a single local model against cyberattacks in [7]. The most prominent issue faced by the state-of-the-art FL-based IDSs is model homogenization. Employing the same IDS model across all CPSs leads to a loss of model diversity, poor model adaptability, and limited customization.

Even though various IDSs have been deployed in industrial CPSs, there are still some issues remaining unsolved. *i)* The class-imbalanced problem commonly exists the real-world datasets of CPSs, and the well-known datasets are only appropriate in experimental stage as the network traffic in these datasets are classic and widespread but lack of specialization; *ii)* The collaborative IDS (CIDS) is a promising method to solve data island, but the collaboration is crowdsourced and contains sensitive data that are possible to expose some critical and private information; *iii)* The primary challenge confronting modern FL-based CIDSs is the issue of model homogenization, since the utilization of an identical IDS model for all industrial CPSs results in a dearth of model diversity, suboptimal model adaptability, and restricted customization options. These challenges obviously limit the performance of IDSs in CPSs, but few efforts have been done to address them.

For addressing these issues, a secure and collaborative IDS solution (i.e., PFD-GAN) is proposed. To be specific, a developed Generative Adversarial Network (GAN)-based IDS is exploited for semi-supervised classification, which is the first to adopt External Classifier GAN (EC-GAN) mode in intrusion detection and further conduct two distinct improvements in the original model. Wasserstein distance, rather than Jensen-Shannon (JS) divergence, is employed as the objective function to provide a smooth measure for stabilizing the gradient descent process, while label condition is introduced as an extension into the latent space to improve the classification performance of EC-GAN. Furthermore, the Local Differential Privacy (LDP) technique is leveraged to prevent privacy leakage by adding well-designed noise into gradients during the

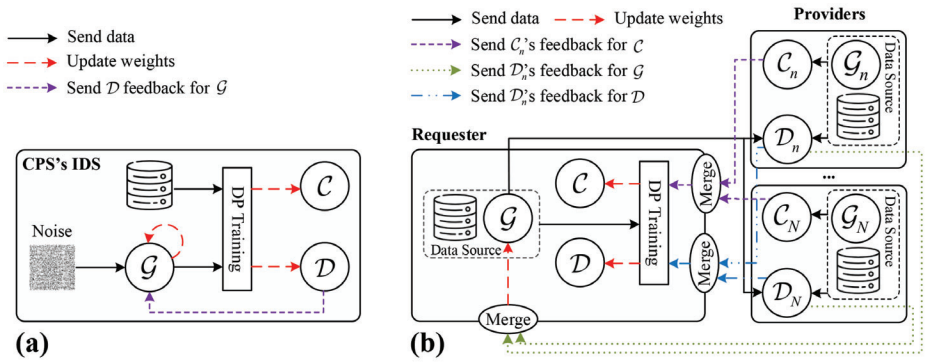

Fig. 1: Scheme of PFD-GAN: *a)* Local Training and *b)* Global Training.

training processes of PFD-GAN. Moreover, a newly-designed Decentralized Federated Distillation (DFD)-based collaboration with advanced compatibility and interoperability is presented for the developed EC-GAN, in which the generator is jointly trained by dispatching its generated data and leveraging the feedback losses from the distributed discriminators for augmenting local data, while the classifier and discriminator are updated based on the Decentralized Knowledge Distillation (DKD) protocol that does not distribute and aggregate an isomorphic neural network for general applicability.

The remainder of this paper is organized as follows. The collaboration of PFD-GAN, including the workflow, the developed EC-GAN and DFD-based collaboration, is presented in Sections II, III and IV progressively. Section V demonstrates the experimental results and theoretical analysis in detail. At last, conclusions are presented in Section VI.

II. Collaboration of PFD-GAN

The collaboration of PFD-GAN is illustrated in Fig. 1. The complete workflow of a PFD-GAN collaboration session can be described in two processes: (i) *local training* inside of an industrial CPS (Fig. 1(a)); (ii) *global training* across multiple CPSs (Fig. 1(b)). In the beginning, every CPS leverages its local dataset to train its own developed EC-GAN that integrates with Wasserstein distance and label condition in a privacy-preserving way during the local training process. Once completing the local training, any collaboration-oriented CPS or a cooperative service requester can initiate the global training process by networking with multiple peers, i.e., its corresponding providers, to build a comprehensive IDS model based on the newly-designed DFD-based collaboration.

Particularly, in a collaboration session of PFD-GAN, there are a cooperative service requester u and N corresponding providers $\{v_n\}_n^N$. The service requester and n-*th* provider v_n are equipped with a local dataset S and S_n respectively, and the entire dataset is donated by $\bigcup_{n=1}^{N} S_n \cup S$. A generator $\mathcal{G}$, a discriminator $\mathcal{D}$, and an external classifier $\mathcal{C}$ are hosted in the requester with a privacy-preserving layer, while v_n operates its own discriminator $\mathcal{D}_n$ and classifier $\mathcal{C}_n$ in a peer-to-peer fashion between them. The weights of $\mathcal{G}$, $\mathcal{D}$, and $\mathcal{C}$ are φ, ω, and θ respectively, and locally renamed as φ_n, ω_n, and θ_n for the n-*th* provider. The detailed procedures of the two processes are elaborated in the following sections.

III. Local Training of PFD-GAN

Here, a generic architecture of the developed EC-GAN is illustrated in Fig. 2 (notice: customized EC-GAN models can be employed for IDSs in industrial CPSs on demand). In the architecture, there are three main modules, i.e., the generator, discriminator, and classifier, and each module has an independent architecture as follows:

i) Generator $\mathcal{G}(\cdot)$: The generator hosts Fully-Connected (FC) layers, Dropout (Dt) layers, and LeakyReLU (LReLU) activation functions, and the FC layers are sorted from the smallest scale to the largest along with the last FC layer that contains the equal number of neurons as the dimension of input data. A randomly sampled vector $z \sim p(z)$ embedded with generated class information l (i.e., a certain class label) in the final dimension is inputted to condition $\mathcal{G}(\cdot)$ to generate the data sample of a certain class as $\hat{x} = \mathcal{G}(z|l)$ for semi-supervised learning.

ii) Wasserstein Discriminator $\mathcal{D} \triangleq f_\omega(\cdot)$: The discriminator uses a similar architecture to the generator, but lines up the FC combos from the largest scale to the smallest with no Dt layer, and it ends with a 1×1 dimension FC layer to differentiate synthetic samples from real data. $f_\omega(x|y) = 1$ and $f_\omega(\hat{x}|l) = 1$ indicate that the condition inputs, including the real condition input $(x, y) \sim p(S)$ and the synthetic one $(\hat{x}, l)$, are both classified as real by $f_\omega(\cdot)$. Oppositely, the condition input is considered as fake when $f_\omega(x|y) = 0$ or $f_\omega(\hat{x}|l) = 0$.

iii) Classifier $\mathcal{C}(\cdot)$: The classifier is mainly composed of three convolutional blocks, followed by two FC layers, a dropout layer, and a softmax layer. Each convolutional block consists of a 1-Dimensional Convolutional (Conv 1D) layer, a Batch Normalization (BN) layer, and a max-pooling layer. The softmax layer is exploited to map the nonnormalized output of $\mathcal{C}(\cdot)$ to a probability distribution over predicted classes (e.g., "Normal", "DoS", "Injection", etc.) in accordance with $y = \mathcal{C}(x)$ or $y = \mathcal{C}(\mathcal{G}(z|l))$.

The procedures of the local training on the requester are summarized in Algorithm 1, and are visualized in Fig. 1(a).

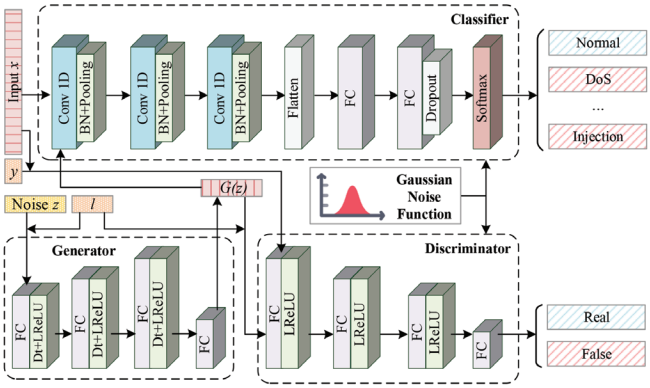

Fig. 2: Generic Architecture of Developed EC-GAN Model.

In the beginning, the requester initializes $\mathcal{G}$, $\mathcal{D}$, and $\mathcal{C}$ with weights φ, ω, and θ in the line 1. When $t_1 \leqslant T_\varphi$, where T_φ is the iterations of $\mathcal{G}$, $\mathcal{G}$ and $\mathcal{D}$ are trained firstly in the lines 3-15 to generate practicable samples. In a loop (the lines 5-10), $\mathcal{D}$ is trained iteratively with $\{x^{(b)}, y^{(b)}\}_b^B$ and $\{z^{(b)}\}_b^B$, until $t_2 > T_\omega$. Specifically, when computing a $\mathcal{D}$'s gradient with respect to a real sample $(x^{(b)}, y^{(b)})$ and a random noise $z^{(b)}$, the gradient is pruned by injecting the well-designed Gaussian noise in the lines 7-8, ensuring that the sensitivity is bounded by ϵ. A $\mathcal{D}$'s gradient for each b can be obtained by Eq. (1):

$$g_\omega(x^{(b)}, y^{(b)}, z^{(b)}) = \nabla_\omega[f_\omega(x^{(b)}|y^{(b)}) - f_\omega(\mathcal{G}(z^{(b)}|y^{(b)})|y^{(b)})] \quad (1)$$

where $f_\omega(\cdot)$ and $\mathcal{G}(\cdot)$ are conditioned on auxiliary information, i.e., a class label $y^{(b)}$. In the line 9, the $RMSProp(\cdot)$ is an optimization function that can adaptively adjust ω_{t_2} (ω in t_2 epoch) according to the magnitude of the gradient $\tilde{g}_\omega$. The $clip(\cdot)$ function is further adopted to guarantee that $\{f_\omega\}_\omega$ are all K_ω-Lipschitz and act in a way to bound the gradient from each data in the line 10. Out of the loop, the average gradient of $\mathcal{G}$ is calculated with regard to $\{z^{(b)}, l^{(b)}\}_{b=1}^B$ in the line 14 as Eq. (2):

$$\bar{g}_\varphi = -\nabla_\varphi \frac{1}{B} \sum_{b=1}^{B} f_\omega(\mathcal{G}(z^{(b)}|l^{(b)})|l^{(b)}) \quad (2)$$

where $f_\omega(\cdot)$ and $\mathcal{G}(\cdot)$ are conditioned on a randomly generated label $l^{(b)}$, and then φ_{t_1} is updated to φ_{t_1+1} by using the $RMSProp(\cdot)$ with the learning rate α_φ and the average gradient $\bar{g}_\varphi$ in the line 15. When $T_\varphi < t_1 \leqslant T_\theta$, the local training of the external classifier $\mathcal{C}$ is performed in the lines 16-24. In the line 20, every $\mathcal{C}$'s gradient $g_\theta(x^{(b)}, y^{(b)}, z^{(b)}, l^{(b)})$ is computed for a pair of real samples $(x^{(b)}, y^{(b)})$ and synthetic data $(\mathcal{G}(z^{(b)}|l^{(b)}), l^{(b)})$ by minimizing the empirical loss function $\mathcal{L}(\theta)$ as Eq. (3):

$$g_\theta(x^{(b)}, y^{(b)}, z^{(b)}, l^{(b)}) = \nabla_\theta[\mathcal{L}_{ce}(\mathcal{C}(x^{(b)}), y^{(b)}) + \lambda \mathcal{L}_{ce}(\mathcal{C}(\mathcal{G}(z^{(b)}|l^{(b)})), argmax(\mathcal{C}(\mathcal{G}(z^{(b)}|l^{(b)})) > \tau)] \quad (3)$$

where $\mathcal{L}_{ce}(\cdot)$ is cross-entropy loss, τ is the pseudo-label threshold. Similarly, in the lines 21-23, we clip the norm of each g_θ along with the adding noise in order to protect the privacy and take a step in the opposite direction of this average noisy gradient $\tilde{g}_\theta$ for updating θ_{t_1} to θ_{t_1+1}. The procedures above repeat iteratively until the desired level convergence for φ, ω, and θ is achieved.

Algorithm 1 Local Training on Requester

Input: $\alpha_\varphi, \alpha_\omega, \alpha_\theta$ % Learning Rates of $\mathcal{G}$, $\mathcal{D}$, and $\mathcal{C}$
B % Batch Size of Data Sampling
$T_\varphi, T_\omega, T_\theta$ % Iterations of $\mathcal{G}$, $\mathcal{D}$, and $\mathcal{C}$
σ_ω, σ_θ % Noise Levels of $\mathcal{D}$ and $\mathcal{C}$
C_ω, C_θ % Norm Bounds of $\mathcal{D}$ and $\mathcal{C}$
Output: Locally trained $\mathcal{G}$, $\mathcal{D}$, and $\mathcal{C}$ with DP

1: initialize φ, ω, and θ for $\mathcal{G}$, $\mathcal{D}$, and $\mathcal{C}$
2: **for** $t_1 = 1, ..., T_\varphi, ..., T_\theta$ **do**
3: **if** $t_1 \leqslant T_\varphi$ **then**
4: **for** $t_2 = 1, 2, ..., T_\omega$ **do**
5: sample $\{z^{(b)}\}_{b=1}^B \sim p(z)$
6: sample $\{x^{(b)}, y^{(b)}\}_{b=1}^B$ from S
7: for each b, compute $g_\omega(x^{(b)}, y^{(b)}, z^{(b)})$ as Eq. (1)
8: $\tilde{g}_\omega \leftarrow \frac{1}{B}(\sum_b g_\omega(x^{(b)}, y^{(b)}, z^{(b)}) + \mathcal{N}(0, \sigma_\omega^2 C_\omega^2 \mathbf{I}))$
9: $\omega_{t_2+1} \leftarrow \omega_{t_2} + \alpha_\omega \cdot RMSProp(\omega_{t_2}, \tilde{g}_\omega)$
10: $\omega_{t_2+1} \leftarrow clip(\omega_{t_2+1}, -C_\omega, C_\omega)$
11: **end for**
12: sample $\{z^{(b)}\}_{b=1}^B \sim p(z)$
13: generate $\{l^{(b)}\}_{b=1}^B$ randomly
14: compute $\bar{g}_\varphi$ according to Eq. (2)
15: $\varphi_{t_1+1} \leftarrow \varphi_{t_1} - \alpha_\varphi \cdot RMSProp(\varphi_{t_1}, \bar{g}_\varphi)$
16: **else**
17: sample $\{z^{(b)}\}_{b=1}^B \sim p(z)$
18: sample $\{x^{(b)}, y^{(b)}\}_{b=1}^B$ from S
19: generate $\{l^{(b)}\}_{b=1}^B$ randomly
20: for each b, obtain $g_\theta(x^{(b)}, y^{(b)}, z^{(b)}, l^{(b)})$ by Eq. (3)
21: $\tilde{g}_\theta \leftarrow \frac{1}{B}(\sum_b g_\theta(x^{(b)}, y^{(b)}, z^{(b)}, l^{(b)}) + \mathcal{N}(0, \sigma_\theta^2 C_\theta^2 \mathbf{I}))$
22: $\theta_{t_1+1} \leftarrow \theta_{t_1} - \alpha_\theta \tilde{g}_\theta$
23: $\theta_{t_1+1} \leftarrow clip(\theta_{t_1+1}, -C_\theta, C_\theta)$
24: **end if**
25: **end for**

Based on the above procedures, the weights of the discriminator and classifier can be shown to guarantee DP with respect to the training data, and the privacy of the data that have not been sampled for training is guaranteed naturally as replacing these data does not cause any change in output distribution. The weights of the generator can also guarantee DP with respect to the training data. This is because there is a post-processing property of DP, which proves that any mapping after a differentially private output does not invade privacy. It is safe for the generator to generate data after the local training since the mapping here is in fact the computation of weights of the generator and the output is the differentially private weights of the discriminator.

IV. Global Training of PFD-GAN

The pseudo-code of the global training on the requester's $\mathcal{G}$ is presented in Algorithm 2 and Fig. 1(b). For every global iteration t, the requester's $\mathcal{G}$ firstly produces a batch

Algorithm 2 Global Training on Requester's $\mathcal{G}$

repeat
  PROCEDURE: TRAINING AT REQUESTER
  for $n = 1, 2, ..., N$ **do**
    sample $\{z^{(b)}\}_{b=1}^{B} \sim p(z)$
    generate $L_{n,t} = \{l^{(b)}\}_{b=1}^{B}$ randomly
    send $(\widehat{X}_{n,t}, L_{n,t})$ to *n-th* provider
  end for
  receive $\{\{e_{n,t}^{(b)}\}_{b=1}^{B}\}_{n=1}^{N}$ from *n-th* provider
  for each $\varphi_t^{(i)} \in \varphi_t$ **do**
    $\triangle\varphi_t^{(i)} \leftarrow \frac{1}{NB}\sum_{n=1}^{N}\sum_{\hat{x}^{(b)}\in\widehat{X}_{n,t}} e_{n,t}^{(b)} \frac{\partial \hat{x}^{(b)}}{\partial \varphi_t^{(i)}}$
    $\varphi_{t+1}^{(i)} \leftarrow \varphi_t^{(i)} + Adam(\triangle\varphi_t^{(i)})$
  end for
  PROCEDURE: TRAINING AT PROVIDERS (*n-th*)
  receive $(\widehat{X}_{n,t}, L_{n,t})$ from requester
  for $b = 1, 2, ..., B$ **do**
    compute $e_{n,t}^{(b)}$ according to Eq. (4)
  end for
  send $\{e_{n,t}^{(b)}\}_{b=1}^{B}$ back to requester
until φ_{t+1} is converged or $++t > T$

of synthetic samples of size B ($\widehat{X}_{n,t} = \{\mathcal{G}(z^{(b)}|l^{(b)})\}_{b=1}^{B}$) with randomly generated labels $L_{n,t} = \{l^{(b)}\}_{b=1}^{B}$. Then, each provider in CPSs is sent with $(\widehat{X}_{n,t}, L_{n,t})$ for the computation of the gradients for $\mathcal{G}$. The error terms $\{e_{n,t}^{(b)}\}_{b=1}^{B}$ of the *n-th* provider can be calculated once receiving $(\widehat{X}_{n,t}, L_{n,t})$ from the requester, as Eq. (4):

$$e_{n,t}^{(b)} = -\frac{\partial log(f_{\omega_n}(\hat{x}^{(b)}|l^{(b)}))}{\partial \hat{x}^{(b)}} \tag{4}$$

where $\hat{x}^{(b)}$ is the *b-th* data of batch $\widehat{X}_{n,t}$. When the requester obtains the error terms $\{\{e_{n,t}^{(b)}\}_{b=1}^{B}\}_{n=1}^{N}$ from all provides, the weight update of $\mathcal{G}$, i.e., $\triangle\varphi_t = -\frac{\partial log(f_{\omega_n}(\hat{x}^{(b)}|l^{(b)}))}{\partial \varphi_t}$, is deduced from all $\{\{e_{1,t}^{(b)}\}_b^B, \{e_{2,t}^{(b)}\}_b^B, ..., \{e_{N,t}^{(b)}\}_b^B\}$ as $\triangle\varphi_t^{(i)} = \frac{1}{NB}\sum_{n=1}^{N}\sum_{\hat{x}^{(b)}\in\widehat{X}_{n,t}} e_{n,t}^{(b)} \frac{\partial \hat{x}^{(b)}}{\partial \varphi_t^{(i)}}$, where $\triangle\varphi_t^{(i)}$ is the *i-th* element of $\triangle\varphi_t$. After computing $\triangle\varphi_t^{(i)}$, the Adam optimizer, the most common method to aggregate updates processed in parallel, is adopted to update $\varphi_t^{(i)}$ as $\varphi_{t+1}^{(i)} = \varphi_t^{(i)} + Adam(\triangle\varphi_t^{(i)})$. The iterative process is recited continuously until φ_{t+1} is converged or the maximum number of iterations is reached. For the global training, the requester's $\mathcal{G}$ is updated using the providers' $\{\mathcal{D}_n\}_n$ and their local shares $\{S_n\}_n$. It is a 1-versus-N game, in which $\mathcal{G}$ is optimized to generate synthetic data considered as real by all the providers while every provider's $\mathcal{D}_n$ tries to differentiate the generated data of $\mathcal{G}$ from the real data in $\{S_n\}_n$.

For better understanding the DKD protocol, the newly-defined notations related to Algorithm 3 are explained. $m \in M = \{"Normal", "DoS", "Injection", ...\}$ is assumed to be an alphabet of $|M|$ labels under consideration. The function $F(\theta_n, s_n^{(b)})$ (or $F(\omega_n, s_n^{(b)})$) is the logit vector normalized by the softmax function, where θ_n (or ω_n) and $s_n^{(b)} \in S_n$ are

Algorithm 3 Global Training on Requester's $\mathcal{D}$ & $\mathcal{C}$

repeat
  PROCEDURE: TRAINING AT PROVIDERS (*n-th*)
  for each $s_n^{(b)} = (x_n^{(b)}, y_n^{(b)}) \in S_n$ **do**
    $F_{\theta_n,t}^{m} \leftarrow F_{\theta_n,t}^{m} + F(\theta_n, s_n^{(b)}), cnt_{\theta_n,t}^{m} += 1$
    $F_{\omega_n,t}^{1} \leftarrow F_{\omega_n,t}^{1} + F(\omega_n, s_n^{(b)}), cnt_{\omega_n,t}^{1} += 1$
  end for
  for each $\hat{s}_n^{(b)} = (\hat{x}_n^{(b)}, l_n^{(b)}) \in \widehat{S}_n$ **do**
    $F_{\omega_n,t}^{0} \leftarrow F_{\omega_n,t}^{0} + F(\omega_n, \hat{s}_n^{(b)}), cnt_{\omega_n,t}^{0} += 1$
  end for
  $\bar{F}_{\omega_n,t}^{0} \leftarrow \frac{F_{\omega_n,t}^{0}}{cnt_{\omega_n,t}^{0}}, \bar{F}_{\omega_n,t}^{1} \leftarrow \frac{F_{\omega_n,t}^{1}}{cnt_{\omega_n,t}^{1}}$
  for each m, $\bar{F}_{\theta_n,t}^{m} \leftarrow \frac{F_{\theta_n,t}^{m}}{cnt_{\theta_n,t}^{m}}$
  send $(\{\bar{F}_{\omega_n,t}^{0}, \bar{F}_{\omega_n,t}^{1}\}, \{\bar{F}_{\theta_n,t}^{m}\}_m^M)$ to requester
  PROCEDURE: TRAINING AT REQUESTER
  receive $\{\{\bar{F}_{\omega_n,t}^{0}, \bar{F}_{\omega_n,t}^{1}\}, \{\bar{F}_{\theta_n,t}^{m}\}_m^M\}_{n=1}^{N}$
  $\bar{F}_t^0 \leftarrow \sum_n^N \bar{F}_{\omega_n,t}^{0}/N, \bar{F}_t^1 \leftarrow \sum_n^N \bar{F}_{\omega_n,t}^{1}/N$
  for each m, $\bar{F}_t^m \leftarrow \sum_n^N \bar{F}_{\theta_n,t}^{m}/N$
  for each b, compute $g_{\theta_t}(s^{(b)})$ by Eq. (8)
  for each b, compute $g_{\omega_t}(s^{(b)}, \hat{s}^{(b)})$ by Eq. (9)
  for each g_{θ_t}, compute $\hat{g}_{\theta_t}(s^{(b)})$ by Eq. (10)
  for each g_{ω_t}, compute $\hat{g}_{\omega_t}(s^{(b)}, \hat{s}^{(b)})$ by Eq. (11)
  $\tilde{g}_{\theta_t} \leftarrow \frac{1}{B}\sum_b(\hat{g}_{\theta_t}(s^{(b)}) + \mathcal{N}(0, \sigma_\theta^2 C_\theta^2 \mathbf{I}))$
  $\tilde{g}_{\omega_t} \leftarrow \frac{1}{B}\sum_b(\hat{g}_{\omega_t}(s^{(b)}, \hat{s}^{(b)}) + \mathcal{N}(0, \sigma_\omega^2 C_\omega^2 \mathbf{I}))$
  $\omega_{t+1} \leftarrow \omega_t - \alpha_\omega \tilde{g}_{\omega_t}, \theta_{t+1} \leftarrow \theta_t - \alpha_\theta \tilde{g}_{\theta_t}$
until $\omega_{t+1}, \theta_{t+1}$ are converged or $++t > T$

$\mathcal{C}_n$'s (or $\mathcal{D}_n$'s) weights and input respectively. The function $\mathcal{L}_{ce}(\cdot)$ is the cross-entropy loss that is used for both loss function and distillation regularizer. The term γ_θ (or γ_ω) is a weight parameter of θ (or ω). $\bar{F}_{\theta_n,t}^{m}$ (or $\{\bar{F}_{\omega_n,t}^{0}, \bar{F}_{\omega_n,t}^{1}\}$) is the local-average logit vector of θ_n (or ω_n) at the *t-th* iteration when the training sample belongs to the *m-th* (or $\{0,1\}$-*th*) ground truth label, $\bar{F}_t^m$ (or $\{\bar{F}_t^0, \bar{F}_t^1\}$) is the global-average logit vector that equals to the average of $\{\bar{F}_{\theta_n,t}^{m}\}_n$ (or $\{\{\bar{F}_{\omega_n,t}^{0}\}_n, \{\bar{F}_{\omega_n,t}^{1}\}_n\}$). $cnt_{\theta_n,t}^{m}$ (or $\{cnt_{\omega_n,t}^{0}, cnt_{\omega_n,t}^{1}\}$) counts the number of the samples whose ground-truth labels are m (or $\{0,1\}$). As shown in Algorithm 3 and Fig. 1(b), the global training on Requester's $\mathcal{D}$ & $\mathcal{C}$ at every *t-th* epoch mainly contains the following steps:

Step 1: the *n-th* provider utilizes its local dataset S_n to compute the local-aggregate logit vectors $\{F_{\theta_n,t}^{m}\}_m^M$ of $\mathcal{C}_n$ for every label m ($m \in M$), and counts the corresponding amount for each m into $cnt_{\theta_n,t}^{m}$, as Eqs. (5-6):

$$F_{\theta_n,t}^{m} = F_{\theta_n,t}^{m} + F(\theta_n, s_n^{(b)}) \tag{5}$$

$$cnt_{\theta_n,t}^{m} = cnt_{\theta_n,t}^{m} + 1 \tag{6}$$

Similarly, a set of synthetic samples $\widehat{S}_n = \{\hat{s}_n^{(b)} | \hat{s}_n^{(b)} \in \{\hat{x}_n^{(b)}, l_n^{(b)}\}_{b=1}^{B}\}$ is generated as a supplement to calculate the local-aggregate logit vectors $\{F_{\omega_n,t}^{0}, F_{\omega_n,t}^{1}\}$ of $\mathcal{D}_n$ and the counts $\{cnt_{\omega_n,t}^{0}, cnt_{\omega_n,t}^{1}\}$ for real and fake data.

TABLE I: MSTS Results in Effectiveness

IDS Solutions	*Precision*	*Recall*	*F1-Score*	*FAR*
i) w/o Differential Privacy				
CNN	95.62	94.56	94.92	5.41
EC-GAN	96.55	95.89	96.11	3.93
Dev EC-GAN	97.13	96.76	96.81	2.20
PFD-GAN	98.65	98.62	98.60	1.14
ii) w Differential Privay				
CNN	93.76	92.77	93.00	7.85
EC-GAN	95.63	94.14	94.68	4.57
Dev EC-GAN	96.41	95.60	95.82	3.03
PFD-GAN	97.94	97.75	97.77	1.54

Step 2: the n-*th* provider uploads its all local-average logit vectors $(\{\bar{F}^0_{\omega_n,t}, \bar{F}^1_{\omega_n,t}\}, \{\bar{F}^m_{\theta_n,t}\}^M_m)$, which are calculated as Eq. (7):

$$\bar{F}^m_{\theta_n,t} = \frac{F^m_{\theta_n,t}}{cnt^m_{\theta_n,t}}, \bar{F}^{0/1}_{\omega_n,t} = \frac{F^{0/1}_{\theta_n,t}}{cnt^{0/1}_{\theta_n,t}} \tag{7}$$

Then, the requester averages the uploaded local-average logit vectors from all the providers separately for each label, and constructs the global-average logit vectors of all labels as $\bar{F}^{0/1}_t = \frac{\sum_n \bar{F}^{0/1}_{\omega_n,t}}{N}$ and $\{\bar{F}^m_t = \frac{\sum_n \bar{F}^m_{\theta_n,t}}{N} | m \in M\}$.

Step 3: As similar to Section III, the LDP technology is leveraged in updating the requester's $\mathcal{D}$ & $\mathcal{C}$. The gradient g_{θ_t} of $\mathcal{C}$ per $s^{(b)} \in S$ and the gradient g_{ω_t} of $\mathcal{D}$ for each $(s^{(b)}, \hat{s}^{(b)})$ are computed with using $(\bar{F}^0_t, \bar{F}^1_t, \{\bar{F}^m_t\}^M_m)$, as Eqs. (8-9):

$$g_{\theta_t}(s^{(b)}) = \nabla_\theta[\mathcal{L}_{ce}(F(\theta_t, s^{(b)}), m) + \gamma_\theta \mathcal{L}_{ce}(F(\theta_t, s^{(b)}), \bar{F}^m_t)] \tag{8}$$

$$\begin{aligned} g_{\omega_t}(s^{(b)}, \hat{s}^{(b)}) = \nabla_\omega[&\mathcal{L}_{ce}(F(\omega_t, s^{(b)}), 1) + \mathcal{L}_{ce}(F(\omega_t, \hat{s}^{(b)}), 0) \\ &+ \gamma_\omega(\mathcal{L}_{ce}(F(\omega_t, s^{(b)}), \bar{F}^1_t) + \mathcal{L}_{ce}(F(\omega_t, \hat{s}^{(b)}), \bar{F}^0_t))] \end{aligned} \tag{9}$$

Since there is no a priori bound on the size of the gradients, each g_{θ_t} and g_{ω_t} in L2-norm are clipped to $\hat{g}_{\theta_t}$ and $\hat{g}_{\omega_t}$ with their clipping thresholds C_θ and C_ω as Eqs. (10-11):

$$\hat{g}_{\theta_t}(s^{(b)}) = \frac{g_{\theta_t}(s^{(b)})}{max(1, ||g_{\theta_t}(s^{(b)})||_2 / C_\theta)} \tag{10}$$

$$\hat{g}_{\omega_t}(s^{(b)}, \hat{s}^{(b)}) = \frac{g_{\omega_t}(s^{(b)}, \hat{s}^{(b)})}{max(1, ||g_{\omega_t}(s^{(b)}, \hat{s}^{(b)})||_2 / C_\omega)} \tag{11}$$

Step 4: In order to protect privacy, the requester adds the well-designed Gaussian noise, i.e., $\mathcal{N}(0, \sigma^2_\theta C^2_\theta \mathbf{I})$ and $\mathcal{N}(0, \sigma^2_\omega C^2_\omega \mathbf{I})$, to each g_{θ_t} and g_{ω_t} when computing the average noisy gradients $\tilde{g}_{\theta_t}$ and $\tilde{g}_{\omega_t}$. At last, the weights ω_t and θ_t of $\mathcal{D}$ & $\mathcal{C}$ are updated in the opposite direction of $\tilde{g}_{\theta_t}$ and $\tilde{g}_{\omega_t}$ as $\omega_{t+1} = \omega_t - \alpha_\omega \tilde{g}_{\omega_t}$ and $\theta_{t+1} = \theta_t - \alpha_\theta \tilde{g}_{\theta_t}$.

The DFD-collaboration can enhance the classification performance of the requester's discriminator and classifier model since the predictions of the providers' models are able to give more helpful information (soft targets) than one-hot labels (hard targets) as a regularizer, which will be proved in Section V and the theory analysis will be discussed in detail.

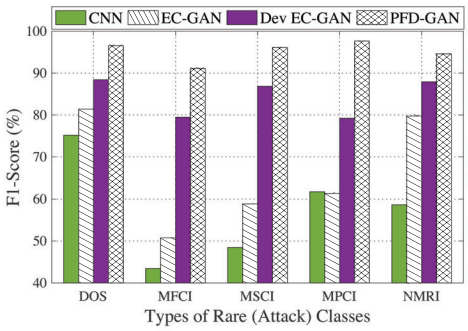

Fig. 3: *F1-Score* under Different Types of Rare Classes.

V. Performance Evaluation

A. Simulation Environment

Our experiments emulate 5 collaborative-oriented IDSs, and a real data resource, i.e., Water Storage Tank System (WSTS, one significant example of industrial CPSs) dataset [8], is used. The WSTS dataset is divided into two major parts, i.e., 80% for training and 20% for testing, and the training part is further split into even partitions for each IDS along with using the same testing data to evaluate all trained models. For simulation parameters, we equally set the learning rates of generator α_φ, discriminator α_ω, and classifier α_θ to 5×10^{-4} and the prune constants, i.e., C_ω and C_θ, to 0.01. The number of local iterations $T_{[\varphi,\omega,\theta]}$ and global iterations T are set to [50,5,100] and 50, where T is equal to T_φ and $\frac{T_\theta}{2}$. The dimension of z is 24, and every coordinate is within [-1,1]. A developed EC-GAN model similar to which presented in Section III is adopted along with using the batch size B calculated by $B = |S|/T$, while its unsupervised Loss-Weight λ and pseudo-label threshold τ are set to 0.1 and 0.7 respectively. In the end, the noise scale of LDP, i.e., (ϵ, δ), is set as $(6, 1 \times 10^{-5})$.

B. Effectiveness Evaluation

In order to demonstrate the effectiveness of our PFD-GAN, we monitor the performance of the IDSs with no proposed enhancement, with the generative capacity from EC-GAN and developed EC-GAN, and with the composite contributions in this paper. The results are presented under two different scenarios, i.e., with or without LDP technology, and the comparison is carried out in terms of *Precision*, *Recall*, *F1-Score*, and *FAR* in TABLE I. In both situations (i.e., *w* or *w/o* DP), the Precision, *Recall*, *F1-Score*, and *FAR* of PFD-GAN are the most advantageous, and those of Dev EC-GAN reach significant increments than those of EC-GAN and CNN. Specifically, the comparison among the different IDS solutions for rare classes is further shown in Fig. 3. It is obvious that the *F1-Score* for the rare classes, i.e., DOS, MFCI, MSCI, MPCI, and NMRI, gets evident improvement when adopting the proposed methods, and PFD-GAN achieves the highest values of *F1-Score*. This is mainly because the developed EC-GAN is designed in PFD-GAN, which inherits and develops the capability of generating usable data to mitigate the imbalanced problem that existed in the WSTS dataset. Apart from that, the

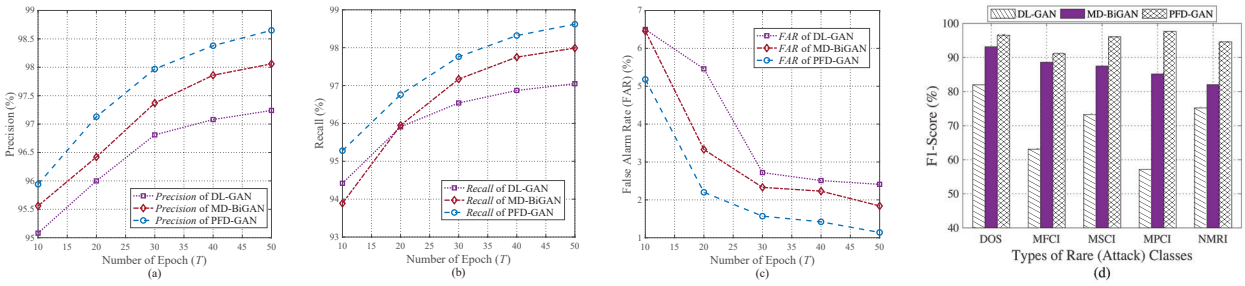

Fig. 4: Comparison among Three IDSs: *a) Precision*, *b) Recall*, *c) FAR*, and *d) F1-Score* among Three IDSs.

DFD-based collaboration in PFD-GAN is proposed enabling multiple industrial CPSs to collectively build a comprehensive IDS over the entire dataset. Therefore, the proposed PFD-GAN is demonstrated perfectly fit as an advanced solution for intrusion detection in industrial CPSs.

C. Performance Comparison

Here, two representatives, i.e., Deep Learning (DL)-GAN [9] and MD-BiGAN [10], are compared with the proposed PFD-GAN. As shown in Fig. 4(a-c), the *Precision*, *Recall*, and *FAR* of PFD-GAN are superior than those of DL-GAN [9] and MD-BiGAN [10], when the number of epochs (T) increases from 10 to 50. Similarly, in Fig. 4(d), the *F1-Score* for the proposed model is better than those of the two references. This is mainly because the developed EC-GAN of PFD-GAN serves as not only the classifier with using a deep learning network to differentiate attacks from normal flows but also the generator to produce synthetic samples as the supplement for rare classes to strengthen the classification performance. Most importantly, the DFD-based collaboration is further designed in PFD-GAN to jointly train the IDS model over the entire industrial CPSs. However, DL-GAN [9] has no any collaboration mechanism to handle the imbalanced problem for the WSTS dataset, while the collaboration of MD-BiGAN [10] only takes place unidirectionally from the multiple generators and encoders to the central discriminator. Therefore, the proposed PFD-GAN has better performance than the two representatives in terms of *Precision*, *Recall*, *F1-Score*, and *FAR*.

Furthermore, the comparison among the existing IDSs in industrial CPSs is provided in TABLE II. According to the figures, the proposed IDS solution, i.e., PFD-GAN, reaches the highest values of *Precision*, *Recall*, and *F1-Score* while securing the lowest *FAR* compared with other prevailing IDS solutions. Therefore, the performance of our proposed PFD-GAN has demonstrated superiorities over the state-of-art IDS solutions in industrial CPSs.

TABLE II: Comparison of Existing IDSs.

IDS Solutions	*Precision*	*Recall*	*F1-Score*	*FAR*
Markov Model	91.33	93.17	92.00	10.40
Hypothesis Testing	94.28	94.00	93.96	7.15
Dolphine + SVM	95.99	95.97	95.87	5.59
GHSOM	96.17	96.11	95.99	5.48
t-test + Bayesian	96.46	96.18	96.08	5.42
GHSOM + MOEA	97.44	97.03	97.01	2.72
Proposed PFD-GAN	98.65	98.62	98.60	1.14

ACKNOWLEDGMENT

This work was supported by the funding of Shenzhen Science and Technology Program (Grand No. RCBS20221008093252092 and No. 20220820003203001).

VI. CONCLUSIONS

In this paper, PFD-GAN is proposed for IDSs in industrial CPSs. First, the novel semi-supervised IDS model is presented that develops EC-GAN with Wasserstein distance and label condition, to improve the classification performance in imbalanced datasets. Furthermore, the LDP is leveraged to prevent against adversaries learning sensitive information in collaboration. Moreover, the DFD-based collaboration is designed, allowing multiple CPSs to build a comprehensive IDS without sharing a uniform template model. Simulation experiments demonstrate the superiorities of PFD-GAN over state-of-the-art IDS solutions.

REFERENCES

[1] Diego GS Pivoto, etc., "Cyber-physical systems architectures for industrial internet of things applications in Industry 4.0: A literature review," *Journal of manufacturing systems*, vol. 58, pp. 176-192, 2021.
[2] Junwei Liang and Maode Ma, "Co-maintained database based on blockchain for IDSs: A lifetime learning framework," *IEEE Transactions on Network and Service Management*, vol. 18, pp. 1629-1645, 2021.
[3] Izhar Ahmed Khan, etc., "Enhancing IIoT networks protection: A robust security model for attack detection in Internet Industrial Control Systems," *Ad Hoc Networks*, vol. 134, p. 102930, 2022.
[4] Weijie Hao, etc., "Hybrid statistical-machine learning for real-time anomaly detection in industrial cyber-physical systems," *IEEE Transactions on Automation Science and Engineering*, 2021.
[5] Bushra Tahir, etc., "Experience-Driven Attack Design and Federated-Learning-Based Intrusion Detection in Industry 4.0." *IEEE Transactions on Industrial Informatics*, vol. 18, no. 9, pp. 6398-6405, 2021.
[6] Beibei Li, etc., "DeepFed: Federated deep learning for intrusion detection in industrial cyberCphysical systems," *IEEE Transactions on Industrial Informatics*, vol. 17, no. 8, pp. 5615-5624, 2020.
[7] Xianting Huang, etc., "EEFED: Personalized Federated Learning of Execution&Evaluation Dual Network for CPS Intrusion Detection," *IEEE Transactions on Information Forensics and Security*, 2022.
[8] ICS Cyber Attack Datasets. [Online]. Available: https://sites.google.com/a/uah.edu/tommy-morris-uah/ics-data-sets, Accessed on: 2022.
[9] Qingling Zhao, etc., "CAN bus intrusion detection based on auxiliary classifier GAN and out-of-distribution detection," *ACM Transactions on Embedded Computing Systems (TECS)*, vol. 21, no. 4, pp. 1-30, 2022.
[10] Weili Wang, etc., "Collaborative intrusion detection for VANETs: A deep learning-based distributed SDN approach," *IEEE Transactions on Mobile Computing*, 2022.