Evaluating Federated Learning-Based Intrusion Detection Scheme for Next Generation Networks

Gurpreet Singh, Keshav Sood, *Senior Member, IEEE*, P. Rajalakshmi, *Senior Member, IEEE*, Dinh Duc Nha Nguyen, and Yong Xiang, *Senior Member, IEEE*

***Abstract*—The proliferation of billions of heterogeneous Internet of Things (IoT) devices at a rapid pace has resulted in a marked expansion of attack surfaces. Numerous new attacks are constantly emerging to undermine the network's availability, data confidentiality, and systems' integrity due to inadequate security measures and resource limitations. Intrusion detection systems (IDSs) are used as the first line of defense to identify early instances of cyber-attacks targeting critical points. However, Next-Generation Networks (NGNs) with dense connectivity pose a challenge for traditional IDS approaches, as they raise concerns about users' data privacy. Federated learning-based IDSs (Fed-IDSs) are an emerging and promising solution, as they permit the training of machine learning models on decentralized data stored on devices without compromising privacy. However, Fed-IDSs also have some unique issues. We identified that the existing Fed-IDSs have poor performance since the datasets used for evaluation, or the data in the real world, are highly imbalanced, and classes are not uniformly distributed. Motivated by this, we developed a novel IDS to effectively address the problem of class imbalance in federated learning at both the local and global levels. Following this, we evaluated the performance of our Fed-IDS under both independent and identically distributed (IID) and non-IID data settings and observed its generalizability to detect various attacks improved greatly. Extensive experiments are conducted to illustrate the effectiveness and benefits of this proposal.**

***Index Terms*—Cyber-security, federated learning, intrusion detection, deep learning, anomaly detection.**

I. Introduction

THE INCREASED reliance on smart devices and the Internet has made cyber-security a growing concern [1]. Therefore, it is critical to develop intelligent Intrusion Detection Systems (IDSs) for Next Generation Networks (NGNs) [2]. IDSs are the first line of defense mechanism that monitors all traffic passing a strategic point for malicious activities. Upon detecting unknown or malicious traffic, they generate alerts to the user/admin. IDSs are divided into two main categories: signature-based and anomaly-based. Signature-based IDSs, like Snort and Suricata, use a database of known attack patterns to identify malicious behavior. However, they require regular updates to their attack signatures to detect new known attacks, and they cannot detect previously unknown (zero-day) attacks. The anomaly-based IDSs use machine learning (ML) or deep learning (DL) models to extract and analyze the network traffic data for outlier detection or classification. Although the anomaly-based approaches can detect zero-day attacks, they give high false alarms and poor performance for various reasons, and one of the key reasons is the use of imbalanced datasets for evaluation [3]. Furthermore, traditional centralized machine learning (TCML) approaches for model training, where all devices send their data to one central server, have been widely proposed to improve the performance of IDSs. However, these approaches raise privacy concerns, and in the real world, data is highly imbalanced, and classes are not uniformly distributed. So, there is a strong need to decentralize the ML tasks so that user data can be trained and pre-processed locally while preserving privacy [4], [5].

Federated Learning (FL) is the new dawn of ML for Next Generation Internet of Things (IoT) networks. It is a collaborative learning technique in which many clients (e.g., various IoT devices or even big organizations) train a global ML/DL model under the supervision of a central server [6]. Despite its potential, using FL to create an IDS is hindered by several limitations, such as the absence of suitable intrusion detection datasets, communication efficiency challenges, susceptibility to model and data poisoning attacks, problems with slow-performing clients or stragglers, and lack of trustworthiness. Furthermore, from our literature synthesis, we have found that class imbalance is a critical and significant problem prevalent in numerous networking datasets (from older to recent ones) extensively used by researchers, as depicted in Fig. 1.

We note that the reason for the class imbalance in networking datasets is attributed to the nature of real-world networks, where the volume of benign or normal traffic significantly outweighs the amount of malicious traffic (long-tail data distribution). This disparity in class distribution poses a challenge when training machine learning models as they tend to be biased towards the majority class, leading to reduced performance in detecting and classifying the

Manuscript received 16 August 2023; revised 26 January 2024; accepted 1 April 2024. Date of publication 4 April 2024; date of current version 21 August 2024. The associate editor coordinating the review of this article and approving it for publication was M. Conti. *(Corresponding author: Gurpreet Singh.)*

Gurpreet Singh is with the School of Information Technology, Deakin University, Melbourne, VIC 3125, Australia, and also with the Department of Electrical Engineering, Indian Institute of Technology Hyderabad, Hyderabad 502285, India (e-mail: id21resch11029@iith.ac.in).

Keshav Sood, Dinh Duc Nha Nguyen, and Yong Xiang are with the School of Information Technology, Deakin University, Melbourne, VIC 3125, Australia (e-mail: keshav.sood@deakin.edu.au; dinh.nguyen@deakin.edu.au; yong.xiang@deakin.edu.au).

P. Rajalakshmi is with the Department of Electrical Engineering, Indian Institute of Technology Hyderabad, Hyderabad 502285, India (e-mail: raji@ee.iith.ac.in).

Digital Object Identifier 10.1109/TNSM.2024.3385385

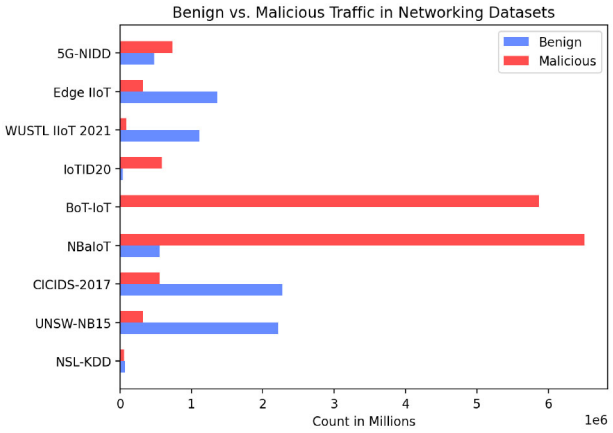


Fig. 1. Binary distribution of widely used IDS datasets.

less frequent but crucial instances of malicious behavior [7], [8], [9]. Unfortunately, to the best of our knowledge, none of the existing studies in Fed-IDSs have looked at this problem in the context of designing network intrusion detection approaches.

The authors in [10] tested the effectiveness of Artificial Neural networks (ANNs) in detecting anomalies in an imbalanced industrial IIoT testbed dataset. They used Synthetic Minority Oversampling Techniques (SMOTE) to balance the data in order to improve the performance of ANN's model. We note that we cannot use this technique in federated settings due to privacy concerns. Further, [9] used Deep and stacked Autoencoders for automatic feature extraction and a deep ensemble of probabilistic neural networks to handle imbalanced data and to detect zero-day attacks, however, it was developed for the SDN context and has not been tested in any federated scenarios. Deep ensemble learning methods are also employed to tackle data imbalance problems, but they are computationally expensive [11]. The authors in [12] recently used both oversampling and ensemble techniques and achieved a high classification performance on the imbalanced dataset.

The problem of class imbalance has been extensively examined in centralized settings; however, adapting those solutions to federated settings is a challenge due to the unknown global data distribution that varies greatly from the local data distributions of each client. Few research works, such as [13], proposed a solution to address the local and global class imbalance by augmenting data in minority classes and redistributing clients through mediators. However, transferring knowledge of client data can lead to privacy leakage and threats to various backdoor attacks. Further, [14] proposed an RL-based solution (using Markov Decision Process (MDP)) to alleviate heterogeneity in federated frameworks. The authors in [15] proposed a new loss function named ratio loss to prioritize minority classes, requiring additional validation data on the server side and assuming a strong correlation between gradient updates and the number of samples.

Notably, class imbalance and non-IID data pose distinct challenges in machine learning, including federated learning. Class imbalance is a scenario where the distribution of classes in the local datasets is not uniform. One or more classes among clients may have significantly fewer samples than others. The associated challenge stems from the bias introduced by the disproportionate representation of classes. Conversely, non-IID data refers to a situation where each client exhibits a unique data distribution, leading to local datasets that may not accurately represent the overall global distribution. The extreme case of non-IID data is the situation when each client has unique class data. Challenges in this context emerge from the imperative to generalize effectively across diverse data distributions. While considerable research efforts have focused on addressing the non-IID (non-independent and identically distributed) data issue in the Fed-IDSs [16], [17], the problem of class imbalance has largely been overlooked.

Motivated by this, we developed a novel Fed-IDS that differs from existing works as it provides insights into handling the class imbalance and non-IID (excluding extreme scenarios) issues simultaneously. Additionally, it is effective in handling both local and global class imbalances and quickly achieves maximum accuracy (high convergence rate) compared to other baselines. In our proposed Federated Intrusion Detection System (Fed-IDS), each client initiates by uniformly sampling network data across all classes, employing Balanced or Uniform Sampling where class weights are inversely proportional to the number of samples. While this ensures equal representation of each class, it can lead to overfitting toward minority classes. To counter this, we used the Feature Space Augmentation [18], a domain-agnostic technique applicable to diverse IoT data modalities. This involves applying arbitrary deviations to the feature space of minority class samples, essentially generating new examples by slightly altering the representations. Each client utilizes a converging Deep Neural Network (DNN) as its deep learning model, with the final layer employing the SoftMax activation for classification. The preceding layers function as a feature extractor, capturing and representing essential features from the input data. To address negative (missed) classes among clients, Knowledge Distillation loss is incorporated during model training alongside standard categorical cross-entropy loss. Additionally, the overall loss calculation employs a smooth regularization (penalty term) to enhance generalization capabilities. Our Contributions are as follows:-

- We have identified that the existing Fed-IDSs overlooked the class imbalance issue while designing and evaluating the IDSs. We are among the early ones proposing to integrate class-balancing approaches while designing next-generation IDSs in the federated learning domain.
- We have evaluated the existing state-of-the-art datasets commonly used by cyber security researchers, and with simulations, we have proved that this is a critical issue affecting the performance and the accuracy of the existing IDSs. To alleviate this problem, we apply an efficient class balancing technique to handle imbalanced data in FL scenarios. Subsequently, we utilized a converging DNN architecture to classify networking attacks.
- We evaluated the performance of our Fed-IDS on four datasets considering both IID and non-IID scenarios. Also, we compared various model training approaches under localized, centralized, and federated settings.

TABLE I
COMPARISON OF STATE-OF-THE-ART PUBLICLY AVAILABLE NETWORK INTRUSION DETECTION DATASETS FROM OLDEST TO RECENT ONES

Year	Dataset	Type	Features	Benign/Total	Summary
2009	KDD Cup 99	Network traffic	41	Train: 0.197/1, Test: 0.195/1	There are 22 attacks in training data and 17 additional attack types in Test data. It contains very large redundant samples.
2009	NSL KDD	Network traffic	42	Train: 0.53/1, Test 0.43/1	Advancement of KDD Cup 99, has widely utilized by researchers. However, it no longer reflects current real-world network scenarios.
2012	ISCX 2012	Network traffic	20	Train: 0.535/1, Test: 0.189/1	It includes real network traffic of HTTP, SMTP, SSH, FTP, POP3, and IMAP protocols and nine diverse attack types.
2015	UNSW NB 15	Network traffic	49	Train: 0.32/1, Test: 0.45/1	There are 49 features and nine kinds of attacks present, excluding normal traffic.
2017	CICIDS 2017	Network traffic	84	0.83/1	It is a big dataset separated into eight large CSV files. There are 14 attacks with large missing samples and high-Class Imbalance.
2018	CSE-CICIDS 2018	Network traffic	80	0.83/1	A collaborative effort of CIC and the University of New Brunswick. The dataset has 80 features and seven attack types.
2018	N-BaIoT	Botnet traffic	115	0.079/1	It is already partitioned into nine clients, making it easy to use in FL. It has been used in recent research works as [21], [23].
2019	BoT- IoT	Botnet traffic	46	Train: 0.0013/1, Test: 0.0013/1	Its reduced version (in 5 CSV files) include 43 features and 3 dependent features (Category, subcategory, and attack).
2019	ToN IoT	IoT traffic	-	-	It is a large heterogeneous dataset that includes 4 different types of traffic with varying features.
2020	IoT-23	IoT traffic	21	-	It include 3 benign, and 20 malware captures. There are 12 attack types: Portscan, Heartbeat, Oriku, benign, and Mirai etc.
2020	IoT-ID20	IoT Traffic	83	0.064/1	It has 80 network features and three label features as binary, category (5 types), and subcategory (9 types).
2021	CIC-ToN-IoT	IoT traffic	83	0.0047/1	It is extracted from the original ToN IoT dataset. In very recent work [16], it is separated into three scenarios according to the IP addresses.
2021	CIC-BoT-IoT	Botnet traffic	83	0.0066/1	It is created from the original BoT IoT dataset. There are four attack categories: DoS, DDoS, Reconnaissance, and Theft.
2021	WUSTL-IIoT-2021	IIoT network	41	0.927/1	It was derived from real IIoT network traffic. It has four attack categories: Command Injection, DoS, Reconnaissance, and Backdoor.
2022	Edge-IIoT	IoT traffic	61	0.715:1/1	This recent, comprehensive, and realistic cyber security dataset is generated from over 10 IoT devices. It consists of 14 attack categories.
2022	5G-NIDD	5G network data	49	0.393:1	It is generated from real 5G test network. It includes various types of Portscan and DoS/DDoS attacks.

The results confirm that our strategy significantly improved the model's generalizability in detecting different types of attacks, under both IID and non-IID scenarios, excluding extreme non-IID use cases. Section II highlights the review of relevant recent works. Section III covers the threat model and motivation of this work. The proposed methodology, model deployment, and performance evaluation are given in Sections IV, V, and VI, respectively. Further discussion is highlighted in Section VII. The paper is concluded in Section VIII.

II. RELATED WORKS

The development of Federated Learning-based Intrusion Detection Systems (Fed-IDSs) has gained significant attention in recent years to enhance security and privacy in distributed IoT networks. Several notable studies have been conducted in this area, which we discussed in this section. Firstly, in Table I, at a high level, we critically analyzed the publicly available, mostly used intrusion detection datasets since 2009. We highlighted significant class imbalance issues in the datasets. Further, Table II provides a high level comparison of the key existing works.

Nguyen et al. [5] proposed one of the early works in the design of a Federated Learning-based IDS in 2019. They developed an autonomous self-learning-based distributed model named 'DIoT.' It was an extension of their earlier work called AuDi [19], used to identify compromised IoT devices within a network by detecting device-specific anomalies. To mitigate DDoS attacks in industrial IoT domains, [20] proposed 'FLEAM' that employed a collaborative approach by implementing their attack mitigation strategy along the path through fog/edge nodes rather than solely on the victim nodes. They increased the cost of launching DDoS attacks and decreased the mitigation response time. However, they used synchronous FL for model training resulted in a higher communication cost.

The authors in [16] utilized the IBM-FL framework to evaluate Fed-IDS under three different data distributions. They divided the original CIC ToN IoT dataset by IP addresses and made it publicly accessible. They evaluated FedAvg against the recent Fed+ algorithm and demonstrated that Fed+ is superior to FedAvg in multiple scenarios. Liu et al. [17] introduced an adaptive federation aggregation algorithm called Batch-Fed to address the straggler problem in Fed-IDSs for maritime transportation. The Batch-Fed algorithm uses a combination of

TABLE II
A HIGH-LEVEL COMPARISON OF OUR WORK WITH THE EXISTING FEDERATED LEARNING-BASED IDSS

Fed-IDSs	Year	Target Domain	Dataset	Classifier	Clients	Aggregation method	IID	Non-IID	Address Class Imbalance
Rahman et al. [4]	2020	IoT Networks	NSL KDD	Deep Neural Networks (DNN)	k = 4	FedAvg	✓	✓	✗
Beibei Li et al. [27]	2020	Industrial Cyber-Physical Systems	Gas Pipeline	CNN, GRU and MLP	k = 3,5,7	FedAvg	✓	✗	✗
Jianhua Li et al. [20]	2021	Industrial IoT	UNSW-NB15	Gated Recurrent Unit (GRU)	k = 4	FedAvg	✓	✗	✗
Campos et al. [16]	2021	IoT Networks	CIC-ToN IoT	Logistic Regression	k = 4, 10	Fed +	✓	✓	✗
Popoola et al. [21]	2021	IoT Networks	BoT-IoT and N-BaIoT	DNN	k = 5	FedAvg	✓	✗	✗
Valerian et al. [23]	2022	IoT Networks	N-BaIoT	MLP, AE	k = 9	FedAvg	✓	✗	✗
Othmane et al. [24]	2022	Agriculture IoT networks	CICIDS 2018, MQTT, InSDN	DNN, CNN, RNN	k = 5,10,15	FedAvg	✓	✓	Oversample Data prior training.
Wentao et al. [17]	2022	Maritime transportation	NSL KDD	CNN, MLP	k = 100	Fed-Batch	✓	✓	✗
Ours	-	IoT Networks	NSLKDD, Edge-IIoT, ToN IoT	DNN	k = 15	FedAvg	✓	✓	✓

prior global model parameters and direct averaging to prevent a rapid decline in global model accuracy. The local models are trained on the NSL-KDD dataset using a combination of Convolutional Neural Networks (CNNs) for feature extraction and Multilayer perception(MLP) for attack classification.

Popoola et al. [21] showcased the efficacy of FL in identifying zero-day botnet attacks using Deep Neural Networks (DNNs). They deliberately exclude one class from each client during the data partitioning process, simulating a zero-day attack scenario when that specific client encounters test data related to the excluded class. They experimented with 16 alternative Deep neural network architectures by utilizing the N-BaIoT and BoT IoT datasets. However, the information regarding selecting class samples from the nine clients of the NBaIoT dataset [22] was not given. Rey et al. [23] presented a Fed-IDS for detecting malware attacks in IoT devices. The authors conducted extensive experiments in both supervised settings (utilizing MLP) and unsupervised settings (leveraging Autoencoders). They compared the performance of the traditional centralized, distributed, and federated approaches using the N-BaIoT dataset. They evaluated the robustness of their proposed federated model by taking into account various adversarial attacks related to data poisoning (label flipping attacks) and model poisoning (gradient factor and model cancelation attacks). To identify countermeasures, they compared the effectiveness of various model aggregation techniques. They demonstrated that the baseline aggregation algorithm (FedAvg) used in most FL algorithms is highly vulnerable to various attacks. However, they incur high server-side computation costs during model training. We have noticed both N-BaIoT and BoT IoT datasets are malware-infected Big datasets. The authors in [21], [23] achieved high accuracy due to an abundant number of malicious classes present in each client dataset (IID settings). Subsequently, they have not evaluated model performance under non-IID settings. Additionally, all the above Fed-IDSs discussed earlier did not address the class Imbalance issue.

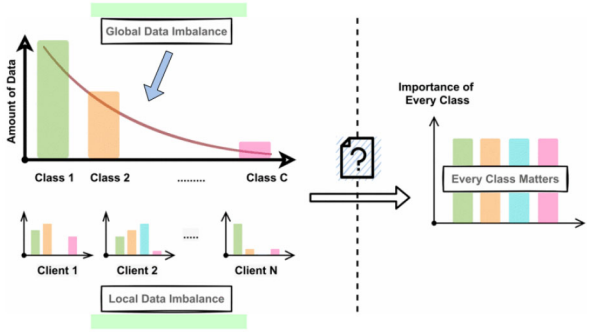


Fig. 2. Local and global level class imbalance in FL [26].

In a recent study, Friha et al. [24] presented a Fed-IDS tailored for agriculture IoT. They assessed the performance of three distinct deep learning models (DNN, CNN, and RNNs) using the FedAvg aggregation method on three networking datasets. The researchers demonstrated the superiority of Fed-IDSs over TCML-based IDSs. However, they employed oversampling to address class imbalance before model training, which may not be effective in real network scenarios. In another work, the authors in [25] proposed a Fed-IDS for vehicular edge computing where Blockchain is utilized to store and share training models, ensuring the security of the aggregate model. However, detection time and computation cost are still ongoing challenges in Blockchain-based solutions.

As shown in Fig. 2, the class imbalance is a significant problem in network traffic that directly affects the model performance at both local and global levels in FL [7], [26]. To our knowledge, the evaluation of class imbalance issues in the context of FL-enabled IDSs has not been thoroughly explored before. Therefore, our work provides a comprehensive evaluation, covering various data distributions using five recent datasets that encompass several IoT-related attacks.

III. Threat Model and Motivation

Most datasets cannot be used in FL contexts since they have not separated according to various IP addresses or devices. Our study has found that the N-BaIoT [22] and CIC-ToN IoT [16] intrusion detection datasets are the only ones that are partitioned based on a limited set of clients, can be used for evaluation under cross-silo federated settings. Furthermore, due to the lack of benchmark test datasets for intrusion detection in FL, researchers use their own discretion to divide the dataset, which leads to bias in the results. Many recent studies achieve high accuracy according to their FL settings and data partition. Details of various datasets are shown in Table I. An additional effort is required to produce the IDS datasets that include a wider scenario of IoT devices with various attacks [16], [23].

Fig. 1 shows the Binary class distributions in commonly used networking datasets. The class imbalance issues are clearly visible in this figure, and if we were to consider multiple classes of malicious traffic, the class imbalance would likely be even more pronounced. If this unbalanced data is used to train the ML model, it will perform well at identifying the benign traffic and behave poorly at identifying the attack traffic. In other words, the model will identify benign data as benign data and identify attack data as benign, resulting in high false positives. Recognizing minority (attack) data with accuracy is of greater importance than recognizing majority (benign) data. In FL, data comes from diverse sources (system heterogeneity), which could make the global model's convergence difficult and time-consuming. When statistical heterogeneity originates from multiple clients, class imbalance increases considerably at the local and global levels, which significantly biases the classifier in favor of the majority class [26].

Arguably, the non-parametric supervised learning method (such as Decision Trees) and ensemble techniques (Random Forest, XG boost, and Autoencoders) are effective against outliers detection, and they are good compared to other models against data imbalance. However, we could not eliminate false positives and false negatives after certain K-Fold cross-validations and hyperparameter tuning. Therefore, various resampling (undersampling/oversampling) methods are commonly used to address the class imbalance issue in datasets. Examples include SMOTE, ClusterCentroids, TomekLinks, NearMiss, and Random under/over sampler methods. Generative Adversarial Networks (GANs) are also emerging as a potential method to balance the tabular networking data by generating new synthetic samples from minority classes [28]. Unfortunately, none of these resampling strategies and GANs can be directly applied in Fed-IDSs [26]. Since client data distribution cannot be known due to privacy concerns, only model parameters are shared during model training. Concept Drift as a result of data heterogeneity is another under-explored area in FL [29].

There are some federated aggregation algorithms proposed in the literature which are good at handling non-IID data, such as FedNova [30], FedProx [31], and Scalffold [32]. Personalized FL is also an emerging area in FL that aims to tackle non-IID scenarios effectively. Essentially, personalization FL attempts to answer the very valid question, *"Why train one model for all clients?"* It essentially trains a personalized model for each device while leveraging collaborative learning. However, PFL relies on the assumption of some level of similarity between the distributions of training and test data, which limits its ability to address global imbalances in the overall data distribution [26]. Researchers have made significant efforts to overcome the non-IID data problem in FL, but the class imbalance problem has not been adequately addressed [13], [15], [26]. Hence, developing more resilient and cost-effective class balancing techniques for FL is crucial.

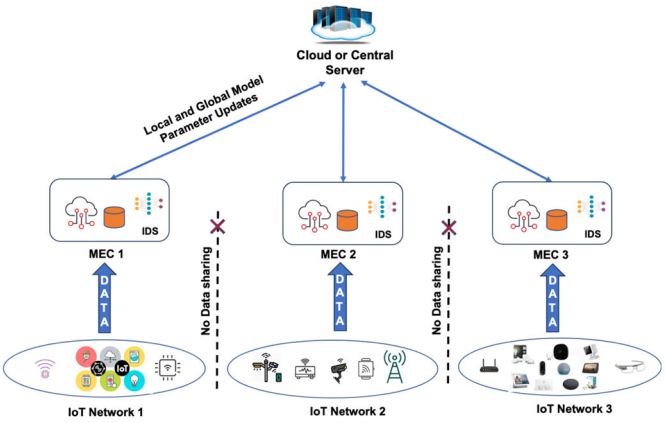


Fig. 3. A high level architecture of the proposed Fed-IDS.

In summary, the effectiveness of next-generation Fed-IDSs cannot be accurately assessed with current datasets, so there is a need to create significantly larger and more diverse networking datasets appropriate for federated settings [16], [23]. Furthermore, incorporating efficient class-balancing methods within the Fed-IDSs can address data-related challenges and improve the performance of IDSs in real-world environments. High-level comparisons of our proposal with the existing Fed-IDSs are shown in Table II.

IV. The Proposed Approach

Conventional wisdom holds that statistical heterogeneity significantly negatively impacts the performance of AI models, which ultimately impacts the performance of ML/DL-based IDSs. We address the critical issue of class imbalance in Fed-IDS by utilizing a recent class balancing technique [26] to classify network traffic. This has been applied to design Fed-IDS, showing a significant novel design feature in the Fed-IDS domain. The approach develops a balanced global model under the imbalanced long-tail data distribution among clients.

Fig. 3 illustrates a high-level deployment framework for the proposed Fed-IDS. In this three-layered architecture, the uppermost layer encompasses a security cloud platform managed by the network operator. The lowest layer represents diverse and intelligent endpoints (IoT devices). The middle layer comprises the N discrete Multiaccess Edge Computing (MEC) platforms, serving as interconnected hosts for the Intrusion Detection System (IDS) component. Each MEC

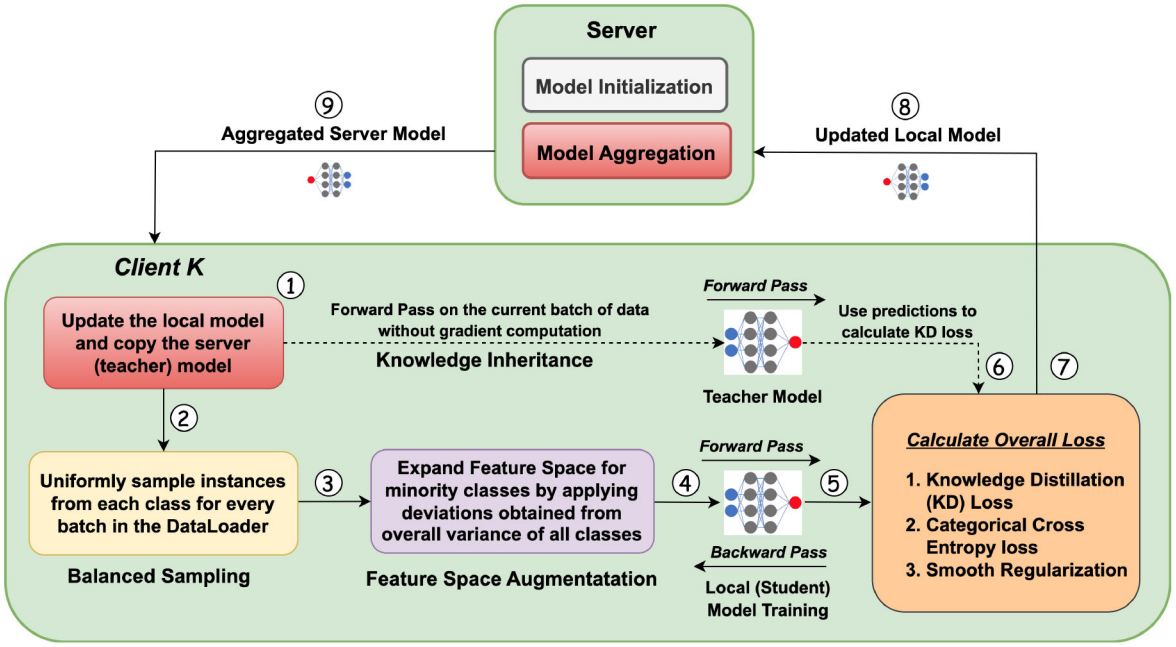


Fig. 4. Workflow of each client in the proposed Fed-IDS.

platform collects traffic data from IoT networks and trains its local model under the orchestration of a central server. These MEC devices possess robust computational capabilities, allowing them to store and process data from various sensor networks efficiently. It is important to note that the proposed IDS is a part of MEC devices.

The high-level working of the proposed approach on each client is illustrated in Fig. 4. Steps 1-9 iteratively performed until the convergence is achieved. For simplicity, the server's initial model-sharing step is omitted. In each global round (r), the aggregating server selects a p fraction of K clients (MEC devices) and sends them its global model with its corresponding weights.

Each client can encounter two types of unbalanced data situations. The first scenario involves missed classes (C^k_{neg}), while the second relates to the long tail distribution of active or positive classes (C^k_{pos}). To tackle missed classes, knowledge is inherited from the global model, referred to as Knowledge Inheritance, while inter-class balancing involves three techniques: Balanced Sampling, Feature Level Data Augmentation, and Smooth Regularization, which are discussed in detail in the next section. The workflow is shown in Algorithm 1, and the key mathematical notations are listed in Table III.

For every input sample x, the local model generates C-dimension predictions, represented as $F(x) = \mathbf{f} = [f_1, f_2, \ldots, f_C] \in \mathbb{R}^C$, where $\mathbb{R}^C$ represents a C-dimensional real space, and $C = |C^k_{\text{pos}}| + |C^k_{\text{neg}}|$, including both positive and negative classes. The learned weights and biases are often called knowledge in neural networks. During the process of federated model training, the global model on the server side (acts as a teacher) gradually gathers the knowledge of all classes from participating clients. This accumulated knowledge serves as the foundation for instructing the clients (act as students). Consequently, even when the local data of these clients lacks exposure to specific classes, they can learn about these classes through a process known as Knowledge Inheritance or Knowledge Distillation.

TABLE III
KEY MATHEMATICAL NOTATIONS

w^k_{r+1}	Updated k^{th} client model after local training
C^k_{neg}	Absent classes on k^{th} client
C^k_{pos}	Active classes on k^{th} client
s^k_j	Total samples of j^{th} class present on k^{th} client
s^k_{max}	Total samples of majority class on k^{th} client
$\frac{n_k}{n}$	Number of samples in client k^{th} dataset out of total samples from all the clients
P^k_{aug,c_j}	Augmentation probability of j^{th} class for $k^{th} client$
y	Ground truth or actual labels
f	Model predictions (after softmax) for each class
f^t,f^s	Soft predictions of teacher and student models

The primary goal of employing knowledge distillation is to transfer the knowledge learned by the teacher model (comprising rich representations and patterns of clients' data) to the student models, making the student model more efficient and able to generalize better. The process of knowledge distillation involves training the student model to minimize a specific loss function, known as the knowledge distillation loss [33]. This loss function quantifies the disparity between the outputs of the teacher model and those of the student model. The knowledge distillation loss comprises two key components, i.e., Soft Targets Loss and Hard Targets Loss.

In the Soft Targets Loss, this component measures the difference between the softened (probabilistic) predictions of the teacher model and the student model as expressed

Algorithm 1: Working of Proposed Fed-IDS. K Clients, Denoted by Index k, With a Selection Ratio p in Each Round. The System Executes a Total of R Global Rounds (Indexed by r)

Input: Initial global model with random weights
Output: Final global model with updated weights
Server executes:
- Initialize global model w_0^g
- **for** *each communication round* $r \leq R$ **do**
  - Randomly choose set $\{S_r\}$ of $p \cdot K$ clients
  - Broadcast the server model to clients
  - **for** *each client* $k \in S_r$ *in parallel* **do**
    - $w_{r+1}^k \leftarrow$ **Client**(w_r^g, k)
  - **end**
  - $w_{r+1} = \sum_{k=1}^{K} \frac{n_k}{n} w_{r+1}^k$
- **end**

Client (w_r^g, k)**:**
- Copy the server (teacher) model for knowledge inheritance and update the local model: $w_r^k \leftarrow w_r^g$
- Determine P_{aug,c_j}^k and store it for later use
- Obtain features (θ_r^k) by passing data through the feature extractor of w_r^k
- Compute the covariance matrix for each class σ_j^k by analyzing the class-specific features, $\text{cov}(\theta_r^k \mid y = c_j)$, for $j = c_1, c_2, \ldots, c_m$
- Determine σ^k, the weighted average of σ_j^k
- Generate random vectors (deviations) Δ_r^k by sampling from $\mathcal{N}(0, \sigma^k)$ for feature space augmentation during local model training
- `// Local Model Training loop`
- **for** *each local epoch from 1 to E* **do**
  - **for** *each batch (applied Balanced sampling to the DataLoader)* **do**
    - Perform a forward pass through w_r^k and store logits
    - Apply (add) Δ_r^k to the original features (θ_r^k) selectively based on P_{aug,c_j}^k
    - Calculate logits for augmented features, f_{aug}
    - Perform forward pass through the teacher model (without gradient computation) and store predictions
    - Use ground truth labels (y), teacher predictions (f^t), student logits (f^s), and augmented logits (f_{aug}) to compute overall loss (cross-entropy loss, knowledge distillation loss, and penalty term for smooth regularization)
    - Perform backward pass to compute gradients
    - Update model parameters using the optimizer: $w_{r+1}^k \leftarrow w_r^k - \eta \nabla l(w_r^k; b)$
  - **end**
- **end**
- Return updated model weights w_{r+1}^k to the server

by equation (1). The softening process involves applying a temperature parameter to the logits (pre-softmax outputs) before computing the softmax function. It enables the student model to learn from the teacher model's uncertainty and better capture intricate decision boundaries. During this process, forward propagation is executed on both the teacher and student models, while backward propagation is performed exclusively on the student model, as shown in Fig. 4. In the Hard Targets Loss, this component compares the output predictions of the student model with the actual ground truth labels. This is similar to the standard classification loss (cross-entropy loss) used in training traditional neural networks, as shown in equation (6).

Thus, the total knowledge distillation loss is a weighted sum of these two components. The weights assigned to each component can be adjusted to balance the influence of teacher knowledge and ground truth labels on the student model's training. Consequently, during local model updates, clients are motivated to contribute to activated classes (C_{pos}^k) while simultaneously minimizing changes to predictions related to absent classes (C_{neg}^k) to preserve the knowledge imparted by the teacher model. This goal is achieved through the joint optimization of two loss functions: the distillation loss (as expressed in equation (2)) applied to the absent classes and a regularized variant of the cross-entropy loss (as shown in equation (6)) applied to the activated classes.

$$f_j^t = \frac{\left(f_j^t\right)^{1/T}}{\sum_{j=1}^{C}\left(f_j^t\right)^{1/T}}, f_j^s = \frac{\left(f_j^s\right)^{1/T}}{\sum_{j=1}^{C}\left(f_j^s\right)^{1/T}}, \tag{1}$$

T is a hyperparameter here, also called distillation temperature. Its value greater than one is used to amplify the originally small probabilities from the teacher model.

Mathematically, the Distillation loss function for k^{th} client is given by,

$$L_{KI}^k(f^t, f^s) = \sum_{f \in C_{neg}^k} f_j^t \log f_j^s \tag{2}$$

here, $f^t = [f_1^t, f_2^t, \ldots, f_C^t]$ and $f^s = [f_1^s, f_2^s, \ldots, f_C^s]$ are the modified version of predicted probabilities (after softmax) of j^{th} negative class by the teacher and student models, respectively. In steps 2–3 of Fig. 4, *Balanced Sampling* is used to increase the likelihood that data from tail classes to be chosen. This involves assigning probabilities to each class based on the number of samples in that class, which is inversely proportional to the sample count. The data loader uniformly selects samples from each class for each training batch, utilizing the probability of selection $P_{s,j}^k$ as defined in Equation (3).

The premise is that if each client has balanced pseudo-local data, then the global data as a whole will also be pseudo-balanced. Suppose if there are total m classes, then the probability of selection of j^{th} class by k^{th} client is,

$$P_{s,j}^k = \frac{\left(s_j^k\right)^{\lambda}}{\sum_{j=c_1}^{c_m}\left(s_j^k\right)^{\lambda}} \tag{3}$$

Here, $\lambda = 0$ for giving equal importance to each class. Consequently, the probability $P_{s,j}^k$ can be expressed as $1/m$, where m represents the number of classes.

Next, steps 3–4 involve the implementation of *Feature Space Augmentation*, a process in which augmentations are applied to the feature space rather than the input data, as outlined by [18]. This approach is domain agnostic and does not depend on input data shape as augmentations (deviations) are applied directly to the feature space, making it highly useful for a large number of data modalities. The process includes computing the weighted covariance matrix of all active classes (σ^k). Further, before training, the desired deviations are generated by sampling from a multivariate Gaussian

distribution with a zero mean and a computed variance σ^k. The number of deviations that need to be generated is decided first; it is a hyperparameter. The probability of applying these deviations is inversely proportional to the number of samples in each class. These deviations are then randomly applied to the features of underrepresented classes according to calculated augmentation probability during each batch of model training. The overall variance (σ^k) in all classes of k^{th} client is utilized to expand the feature space for classes with limited samples. Mathematically,

$$\sigma^k = \frac{\sum_{j=c_1}^{c_m} s_j^k \sigma_j^k}{\sum_{j=c_1}^{c_m} s_j^k} \tag{4}$$

here, $\sigma_j^k \in R^{dXd}$ denotes covariance matrix of feature vectors of class j. Classes with fewer samples in the dataset are augmented with higher probability to reduce the overfitting due to *Balanced Sampling* described in a previous section. The formula below provides the class-specific probability for augmentation of class c_j on k^{th} client.

$$P_{aug,c_j}^k = \frac{s_{max}^k - s_j^k}{s_{max}^k} \tag{5}$$

In summary, feature space augmentation is performed by selectively applying deviations to the features of certain samples in the training data, enhancing the model's ability to handle variations in the input space.

Following this, steps 1, 6, 7 and 4, 5, 7 correspond to the knowledge distillation and local model training processes, respectively. The computation of knowledge distillation and categorical cross-entropy losses takes place after the forward pass of both the teacher and student models. In addition to this, an extra *Smooth Regularization* (penalty term) is incorporated during the calculation of the categorical cross-entropy loss to enhance its generalization capabilities. This regularization serves to address the bias introduced by the standard categorical cross-entropy loss function due to imbalanced data distribution, as illustrated in Equation (7). Its objective is to penalize overconfident classes, thereby promoting improved generalization. Further model gradient parameters are updated in the backward pass of the local model.

The formula for conventional cross-entropy loss is given as,

$$L_{CE}^k(y,f) = -\sum_{j \in C_{pos}^k} y_j log f_j \tag{6}$$

here, y $= [y_1, y_2, \ldots, y_c] \in R^C$ represents the ground truth vector and $\mathbf{f} \in R^C$ is the model prediction described in earlier section. The formula for smooth regularization (also called negative entropy) is given by

$$L_{Smooth}^k(f) = \sum_{j \in C_{pos}^k} f_j \log f_j \tag{7}$$

Finally, by putting Equations (2), (6), and (7) together, the overall loss for an input sample on k^{th} client is equal to

$$L_{KI}^k(f^t, f^s) + L_{CE}^k(y,f) + \lambda_1 L_{Smooth}^k(f) \tag{8}$$

λ_1 is a balancing factor here.

Steps 2–7 are repetitively performed for each batch of data. After a few local epochs (E = 5 in our case), each client sends its updated model parameters (w_{r+1}^k) back to the server for aggregation (as shown in step 8 in Figure 4). The server objective is to reduce the overall global loss (Fedavg is used for aggregation), which is the weighted average from individual k client losses given as $\sum_{k=1}^{K} \frac{n_k}{n} F_k(w)$. Here, $F_k(w) = w_{r+1}^k$ represents k^{th} client loss function. The dataset size (n_k) of k^{th} client decides its weight. To ensure equal importance for each class sample, we used balanced sampling (explained earlier). After collecting and aggregating model parameter updates from all the selected clients, the updated global model is sent to another set of random clients in its next round, as shown in Step 9. As such, one global round is completed. This process repeats until the model convergence or algorithm completes the selected number of global training rounds.

To show the Proof-of-Concept (PoC), we conducted several experiments on five recent network intrusion detection datasets. Six baseline aggregation approaches are used in our comparative study. In addition to our proposed approach, the other five baselines used for comparison are as follows:

1) *Local* - No global training rounds. The model trained locally on each client for 200 training rounds after distributing data among clients. After calculating the model performance for each client, the mean values from all clients have been computed to represent the final metrics.
2) *Central* - Clients individually send their data to a server for model training in each global round (TCML method). Unfortunately, privacy is not maintained in this process.
3) *Fedavg* [6] - a standard FL approach, which extends the traditional stochastic gradient descent (SGD) algorithm. It involves each client performing a few local training rounds on its own data, then sending the updates to a central server for aggregation. The resulting model is then broadcast back to all clients for further training.
4) *Fedprox* [31] - FedProx incorporates an additional regularization term, in contrast to FedAvg, to manage non-IID data. This regularization term balances the trade-off between global and local information in the model updates.
5) *Balanced Meta Softmax (BALMS):* - Ren et al. [34] proposed Balanced Meta Softmax, also called BALMS, to alleviate the Class Imbalance issues in long tail data distributions in *centralized settings*. So, privacy is not maintained in this process. As it is an effective class-balancing technique for TCML methods, hence we used it in our comparative analysis.

We have seen a reasonable amount of improvement in our comparative study. Using a converging DNN (discussed in the next section) on four recent network intrusion detection datasets considering different data distributions (IID and non-IID), we demonstrated that resolving class imbalance improves the generalizability of Fed-IDS to categorize multiple attacks.

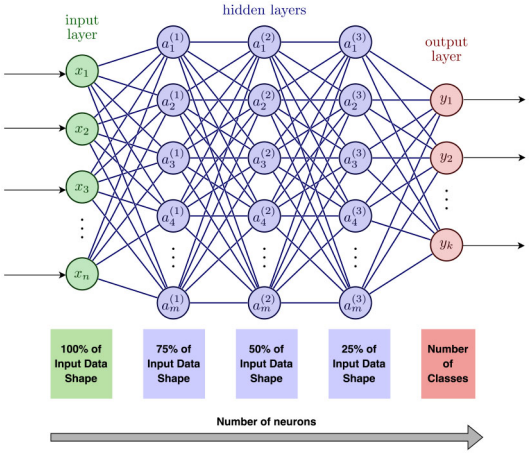


Fig. 5. Architecture of our DL Model.

V. Model Development

To extract meaningful information from each layer of the network, we used a converging deep neural network architecture. We could obtain much higher accuracy if we used other cutting-edge deep learning models, but that was not the goal of our experimentation. After class balancing, we observed a significant improvement in the generalizability of Fed-IDS.

A. Model Architecture

Our proposed deep neural network architecture is inspired from [22], [23]. Basically, we used the encoder part of the autoencoder as autoencoders are effective in dimensionality reduction and good in handling class imbalanced data [9]. Our network includes five layers: one input, one output, and three hidden layers. The number of neurons in the input layer is equal to the number of features in the dataset. The number of neurons in each hidden layer is 75%, 50%, and 25% of the input layer for extracting useful features from the data. Finally, there is an output layer equal to the number of classes in the dataset. We employed the last layer of the Deep Neural Network (DNN) model as the classifier, whereas the remaining part was used as a feature extractor. Notably, the features for feature space augmentation were derived by passing data through the feature extractor segment of the model. A mathematical representation of the proposed deep neural network's input layer $(h^0(x))$, multiple hidden layers $(h^k(x))$, and output layer $(h^{n+1}(x))$ is as follows:

$$h^0(x) = x \tag{9}$$

$$h^k(x) = f\left(w^k h^{k-1}(x) + b(k)\right) \tag{10}$$

$$h^{n+1}(x) = f\left(w^{n+1} h^n(x) + b^{n+1}\right) \tag{11}$$

Here, x is the input (d-dimensional data) vector represented as $X = (x_1, x_2, x_3, \ldots, x_d) \in R^d$, w means weight vector denoted as $w_j = (w_{j,1}, w_{j,2}, \ldots, w_{j,d})$, b means bias vector for each hidden layer denoted as $(b_1, b_2, .., b_j)$, $f(.)$ is the activation function. We used the Rectified Linear Units (ReLU) activation function, which equals $f(x) = max(0, x)$. k= Number of hidden layers = 1, 2,..n. In our case, $n = 3$. To transform the model outcomes into probabilities, the softmax function used at the last layer is given by

$$\text{Softmax}\,\sigma(z)_i = \frac{\exp^{z_i}}{\sum_{j=1}^{m} \exp^{z_j}} \tag{12}$$

here, m is the number of classes in the dataset. The Softmax $\sigma(z)_i$ can be thought of as the predicted probability that the test input would belong to class i. z is the vector of raw outputs from the neural network known as logits, and z_i represents the i^{th} value of the softmax output vector.

B. Simulation Settings

The experiments are conducted on a Linux server (Ubuntu 18.04.6 LTS) using an Intel Xeon Platinum 8168 CPU running at 2.70 GHz with an NVIDIA Tesla V100 SXM3, 32 GB GPU. PyTorch, a deep learning framework, is used to simulate experiments, and Tensorboard is used to visualize the outcomes. The number of global training rounds is 200, and local epochs are 5. The data was partitioned among clients using a random seed value of 13. DL model parameters such as dropout are 0.01, and the learning rate is set to 0.001. We use Adam optimizer, Batch size set to 64, and the regularization term (μ) was taken as 0.05 for Fedprox. The temperature parameter is set to 2 for knowledge distillation, and the balancing factor (λ_1) equals 1. The number of deviations generated from the overall covariance matrix is 2000. These parameters are used throughout our tests as a basis for comparison.

C. Datasets Used

As old datasets are not reliable for designing IDSs for next-generation networks mentioned in the recent work [2], we have evaluated the proposed Fed-IDS rigorously on four recent datasets containing modern network traffic footprints under various data distribution settings. The class distribution in all the datasets is shown in Fig. 6. We used IoT-ID-20, Edge-IIoT, and 5G-NIDD datasets to evaluate the performance of proposed Fed-IDS under IID settings. Further, to evaluate the Fed-IDS performance under non-IID data settings, we used the Dirichlet criteria [35] to distribute data among clients on a realistic WUSTL-IIoT-2021 dataset. We evaluated its performance across three different alpha values. The alpha parameter in the Dirichlet distribution plays a crucial role in determining the concentration and shape of data distribution among clients. A small alpha value represents the sparse or highly skewed distribution. Non-iidness increases as going towards zero. Conversely, as alpha approaches ∞, the distribution becomes more balanced, with equal representation across all categories. We performed experiments utilizing three distinct alpha values (0.3, 0.5, 1) to assess the model's performance across three diverse non-iid scenarios, similar to other works [35], [36]. Considering computational constraints, we opted to work with 15 clients in our federated learning framework, representing a cross-silo federated scenario [37]. All datasets are separated into a train-test ratio of 80:20 utilizing stratified sampling with a random seed of 13.

1) IoT-ID-20 Dataset [38]: The dataset originated from the raw network packet files generated by the IoTID dataset [39].

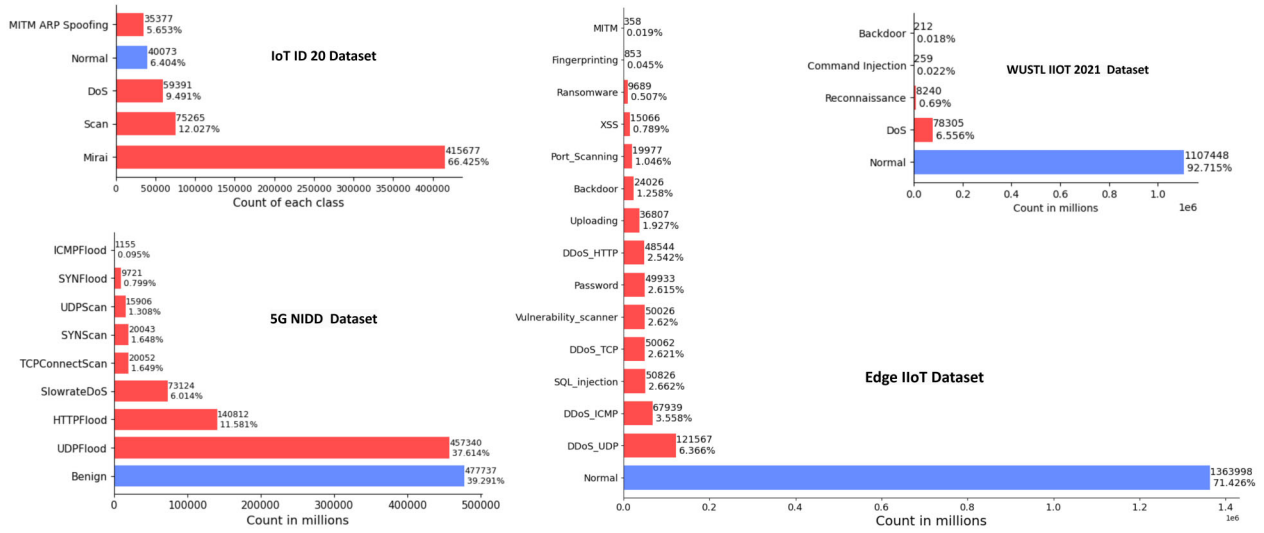


Fig. 6. Class Distribution in the four recent Network Intrusion Detection Datasets.

The experimental configuration emulates a smart home network comprising the AI-based speaker SKT 21 NUGU (NU 100), the EZVIZ Wi-Fi Camera (C2C Mini O Plus 1080P), and different smartphones and laptops. All these devices were interconnected through a shared wireless network facilitated by a smart home Wi-Fi router. Within this setup, the AI-based speaker SKT NGU and the EZVIZ Wi-Fi camera were designated as victim IoT devices, while all other devices within the testbed were configured as the attacking devices. During data processing, we eliminated the following redundant categorical attributes from the dataset: 'Flow_ID,' 'Src_IP,' 'Dst_IP,' 'Timestamp,' 'Label,' and 'Sub_Cat.' Consequently, the final dataset comprises 80 features encompassing a dependent attribute with five distinct classes.

2) WUSTL-IIoT-2021 Dataset [40]: The WUSTL-IIoT-2021 dataset comprises network data collected from a real Industrial Internet of Things (IIoT) environment. This dataset was created using the IIoT testbed described in [41]. During data preprocessing, we eliminated redundant, as well as host-specific features, including StartTime,' LastTime,' SrcAddr,' DstAddr,' sIpId,' and dIpId.The final Dataset includes 43 features.

3) Edge-IIoT Dataset [42]: It is a recent, realistic IoT intrusion detection dataset. It includes network traffic from more than ten types of IoT devices, such as ultrasonic sensors, heart rate sensors, water level detection sensors, low-cost IoT sensors to measure temperature and humidity, pH sensors, soil moisture sensors, flame sensors, etc. The dataset contains fourteen attacks on IoT and IoT communication protocols. A recently proposed Fed-IDS by Friha et al. [24] expressed their intent to leverage this dataset for their forthcoming research. After eliminating some redundant features, including 'icmp.unused', 'http.tls_port', 'dns.qry.type', and 'mqtt.msg_decoded_', resulting in a final dataset with 91 features.

4) 5G-NIDD Dataset [43]: It is a very recent intrusion detection dataset produced using Motorola g50 5G devices in a real 5G test network. The original CSV file contained 50 features, excluding two output features named 'Label' and 'Attack Type. Firstly, the features 'Dur,' 'Runtime,' 'Mean,' 'Sum,' 'Min,' and 'Max' were removed because they had the same covariance. After removing a few more redundant features such as 'sDSb, dDSb, 'Label'and 'Attack Tool, ' the dataset was left with 46 feature columns. We also noticed that many features have a large number of null values, so we replaced features with large null values such as SToS, dToS, sTtl, dTtl, sHops, dHops, SrcGap, DstGap, sVid, dVid, SrcTCPBase, and DstTCPBase are replaced with their respective median or zero values. Finally, independent nominal categorical features ('Proto,' 'Cause,' 'State') were encoded using one-hot encoding.

TABLE IV
PERFORMANCE EVALUATION OF DIFFERENT BASELINES ON IOT-ID-20 DATASET

Per-class Accuracy	Local	Central	FedAvg	Fedprox	BALMS	Ours
Normal	0.946	0.963	0.917	0.940	0.960	0.967
Mirai	0.952	0.981	0.958	0.956	0.834	0.849
Scan	0.931	0.952	0.940	0.944	0.944	0.948
DoS	0.995	0.996	0.989	0.992	0.989	0.994
MITM	0.500	0.745	0.199	0.643	0.966	0.976
Macro Accuracy	0.865	0.927	**0.800**	**0.895**	0.938	**0.947**
Macro Precision	0.931	0.971	0.934	0.949	0.877	0.886
Macro Recall	0.919	0.956	0.884	0.935	0.956	0.961
Macro F1-score	0.921	0.963	0.886	0.940	0.901	0.910

VI. PERFORMANCE EVALUATION

We performed several experiments using four recent network intrusion detection datasets and explored three different model training settings: Local, TCML (Traditional Centralized Machine Learning), and FL (Federated Learning). We used the macro average as our performance measure to compute accuracy, precision, recall, and F1-score for all baselines as it provides model evaluation for each class.

We observed a significant increase in the generalizability of Fed-IDS to identify multiple attacks with our approach.

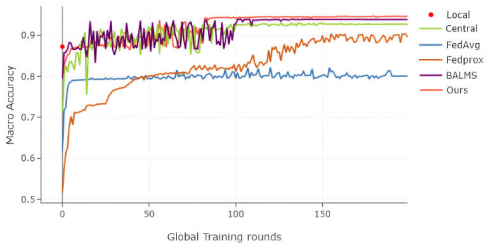
TABLE V
PERFORMANCE EVALUATION OF VARIOUS BASELINES ON EDGE-IIOT DATASET

Per-class Accuracy	Local	Central	FedAvg	Fedprox	BALMS	Ours
Normal	1	1	1	1	1	1
DDoS UDP	1	1	0.995	0.983	0.972	1
DoS ICMP	0.998	0.999	0.985	0.989	0.937	0.986
SQL Injection	0.286	0.219	0.171	0.343	0.190	0.190
DDoS TCP	0.993	0.996	1	1	0.552	0.570
Vulnerability Scanner	0.842	0.842	0.841	0.842	0.842	0.842
Password	0.732	0.805	0.843	0.648	0.885	0.837
DDoS HTTP	0.955	0.937	0.986	0.986	0.606	0.606
Uploading	0.367	0.378	0.377	0.371	0.378	0.380
Backdoor	0.951	0.971	0.931	0.931	0.922	0.931
Port Scanning	0	0	0	0	0.489	0.489
XSS	0.168	0.257	0	0	0.781	0.781
Ransomware	0.899	0.880	0	0.026	0.969	0.935
Fingerprinting	0	0	0	0	0.743	0.819
MITM	1	1	0	0	1	**1**
Macro Accuracy	0.679	0.686	**0.542**	**0.542**	0.751	**0.758**
Macro Precision	0.849	0.843	0.751	0.780	0.858	0.863
Macro Recall	0.838	0.841	0.769	0.769	0.873	0.876
Macro F1-score	0.828	0.835	0.751	0.755	0.843	0.853

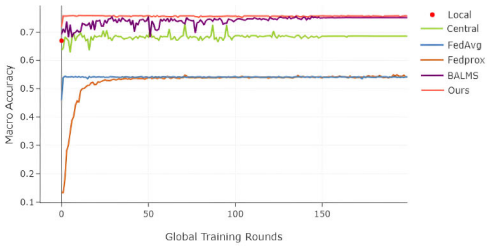
TABLE VI
PERFORMANCE EVALUATION OF DIFFERENT BASELINES ON 5G-NIDD DATASET

Per-class Accuracy	Local	Central	FedAvg	Fedprox	BALMS	Ours
Benign	0.998	0.999	0.997	0.999	0.994	0.998
UDP Flood	0.999	0.999	0.998	0.999	0.999	0.999
HTTP Flood	0.995	0.997	0.987	0.995	0.989	0.991
Slowrate DoS	0.982	0.990	0.984	0.984	0.989	0.993
TCP Connect Scan	0.996	0.998	0.996	0.996	0.997	0.997
Syn Scan	0.998	0.997	0.999	0.998	0.999	0.998
UDP Scan	0.995	0.998	0.995	0.995	0.995	0.995
Syn Flood	0.999	1	0.850	0.998	0.855	1
ICMP Flood	1	0.995	1	1	1	1
Macro Accuracy	0.996	0.997	**0.978**	**0.996**	0.979	**0.997**
Macro Precision	0.998	0.998	0.992	0.998	0.956	0.997
Macro Recall	0.998	0.998	0.989	0.997	0.989	0.998
Macro F1-score	0.998	0.998	0.990	0.998	0.965	0.998

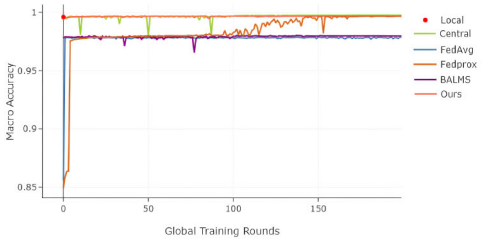
Firstly, we ran a simple experiment for a reader to show the class distribution in all datasets, as shown in Figures 6. To assess the model's performance in IID settings, we randomly partitioned the data among clients, ensuring each client had some representation from all classes. We employed the IoT-ID-20, Edge-IIoTset, and 5G-NIDD datasets for this evaluation. As shown in Tables VII(a), V, and VII(c), ours (proposed Fed-IDS) demonstrates accurate detection of minority class samples while maintaining performance on other classes. Compared to FedAvg, there are significant improvements in macro accuracy, with gains of 14.7%, 21.6%, and 1.9% observed on IoT-ID-20, Edge-IIoT, and 5G-NIDD datasets, respectively. Moreover, compared to Fedprox, it exhibits enhancements in macro accuracy, with gains of 5.2%, 21.6%, and 0.1% on the same three datasets, respectively. This shows that both FedAvg and Fedprox lack the ability to deal with class unbalancing issues. Also, we observed that in contrast to FedAvg and FedProx, our proposed Fed-IDS quickly achieves its maximum accuracy within just a few rounds of model training, resulting in reduced model training time at the server, making it highly effective for real-world deployment. Notably, ours (proposed Fed-IDS) has a very high convergence rate.



(a) Global test accuracy on IoT-ID-20 Dataset



(b) Global test accuracy on Edge-IIoT dataset



(c) Global test accuracy on 5G-NIDD dataset

Fig. 7. Comparative analysis of global test accuracies of various baselines on three recent Network Intrusion Detection Datasets.

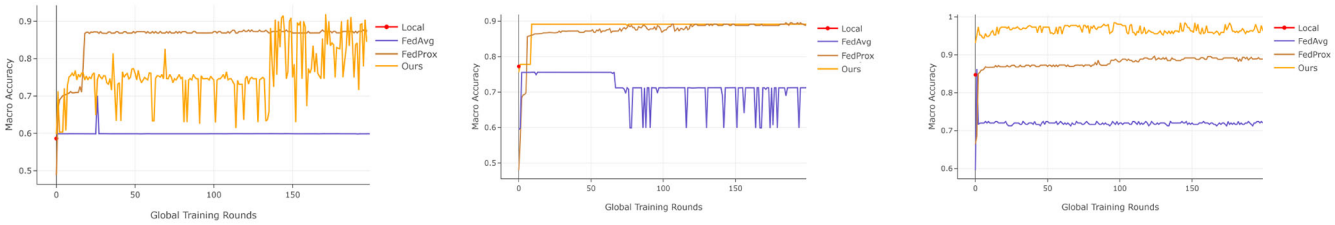
Furthermore, it has also demonstrated superior performance compared to local and TCML methods, such as Central and BALMS, across all datasets, as illustrated in Fig. 7.

To evaluate the performance of our Fed-IDS under various non-IID settings, we used the real-world WUSTL-IIoT-2021 dataset. Employing the Dirichlet criterion, we distributed data among clients and scrutinized its performance relative to other baseline models across three distinct alpha (α) values: 0.3, 0.5, and 1. Central and BALMS are not evaluated across different alpha values. Since these are centralized approaches, all data from all devices must be aggregated in one location before model training. Based on the results shown in Table VII and Fig. 8, the Central approach outperforms BALMS. It is observed that when dealing with highly long-tailed data, BALMS sacrifices the efficiency of the majority class to enhance the performance of other classes. Nonetheless, these methods are based on TCML, requiring all the data at a central place for model training, which is not an effective way to train models in next-generation networks due to privacy concerns being the main issue. Moreover, the performance of the local method exhibited an increase with the rise in the alpha value. While privacy is assured in this approach, it is not an efficient means of training models, as it lacks collaborative learning among clients.

We noticed that compared to FedAvg, our approach shows significant improvement in the classification performance in

TABLE VII
PERFORMANCE EVALUATION OF VARIOUS BASELINES AFTER SPLITTING WUSTL-IIOT-2021 DATASET ACCORDING TO DIRICHLET CRITERIA [35]

Per-class Accuracy	Central	BALMS	$\alpha = 0.3$				$\alpha = 0.5$				$\alpha = 1$			
			Local	FedAvg	FedProx	Ours	Local	FedAvg	FedProx	Ours	Local	FedAvg	FedProx	Ours
Normal	1	0.988	0.862	1	1	1	0.999	1	1	1	0.999	1	1	0.999
DoS	0.999	0.996	0.665	0.996	0.996	0.999	0.912	0.996	0.997	0.999	0.995	0.997	0.997	0.999
Reconnaissance	0.999	0.998	0.755	0.998	0.999	0.999	0.910	0.998	0.998	0.999	0.998	1	1	0.998
Command Injection	0.877	0.785	0.342	0	0.785	0.554	0.569	0	0.861	0.892	0.627	0	0.861	0.985
Backdoor	0.830	0.849	0.307	0	0.623	0.755	0.470	0.566	0.585	0.566	0.615	0.603	0.585	0.906
Macro Accuracy	0.941	0.923	0.586	**0.599**	**0.881**	**0.861**	0.772	**0.712**	**0.888**	**0.891**	0.847	**0.720**	**0.889**	**0.977**
Macro Precision	0.993	0.792	0.715	0.797	0.980	0.977	0.867	0.836	0.992	0.990	0.934	0.836	0.991	0.913
Macro Recall	0.970	0.961	0.756	0.799	0.940	0.931	0.878	0.855	0.944	0.945	0.923	0.860	0.944	0.989
Macro F1-Score	0.981	0.803	0.702	0.798	0.957	0.947	0.856	0.844	0.963	0.962	0.924	0.845	0.963	0.928



(a) $\alpha = 0.3$ (b) $\alpha = 0.5$ (c) $\alpha = 1$

Fig. 8. Comparative analysis of global test accuracies of various baselines across three non-IID scenarios in the WUSTL-IIoT-2021 dataset.

all three non-IID scenarios, i.e., 26.2%, 17.9%, and 25.7% at $\alpha = 0.3$, 0.5, and 1, respectively. Moreover, compared to Fedprox, it still exhibits enhancements in macro accuracy, with gains of 0.3% and 8.8% at $\alpha = 0.5$ and 1, respectively. Additionally, Fedprox, which is specifically designed to handle non-IID data, performed reasonably well in all cases and exhibiting enhanced performance as non-iidness escalated with alpha set at 0.3. Our approach exhibited commendable performance, closely aligned with FedProx, in all three scenarios. However, it faced challenges in achieving convergence, particularly noticeable in Fig. 8(a), as non-iidness increased. We acknowledge that our approach may not perform as effectively in extreme non-IID cases due to substantial variations in local data distribution among clients. The divergence, particularly in minority or majority classes, poses challenges for convergence. However, it performs remarkably well under scenarios with long-tail imbalanced data distributions, which is common in real-world network traffic.

VII. FURTHER DISCUSSION

In many research works, the researchers have focused only on the overall model's accuracy while comparing their work with other models in a multiclass classification problem. Arguably, the per-class accuracy, confusion matrix results (low false positives and false negatives), macro precision, macro recall, and macro F1 score are the real evaluation metrics to compare the model's generalizability against class imbalance.

Besides, we would also like to emphasize that handling class imbalance strengthens Fed-IDSs' capabilities in detecting zero-day attacks since it allows the model to generalize better. For example, Popoola et al. [21] recently showcased the efficacy of FL in identifying zero-day botnet attacks using deep neural networks. They deliberately excluded one class from each client during the data partitioning, simulating a zero-day attack scenario when that specific client encounters test data related to the excluded class. They have used the FedAvg aggregation method. Note that in our work, we have also shown that FedAvg performs better after handling class Imbalance. However, further evaluation is indeed essential.

We reiterate that the scope of this paper is to address the class imbalance issues in Fed-IDSs. It is important to note that the class imbalance issue is a non-ignorable factor. In our future work, we aim to incorporate the pragmatic dynamics of today's edge networks for further deep analyses of the proposed approach. Additionally, to enhance the robustness of the proposed Fed-IDS against various adversarial attacks, we intend to explore the methodology outlined in the recent work by Valadi et al. [36] as a future scope of this work.

VIII. CONCLUSION

We discussed a key issue of class imbalance in the design of novel Fed-IDSs and have applied a potential solution to alleviate this issue. We have emphasized or proved experimentally that class imbalance critically impacts the performance or generalizability of Fed-IDSs. By employing a recently developed class balancing technique in Federated Learning (FL), we have demonstrated a notable enhancement in the performance of Fed-IDS across various recent datasets with distinct distributions. To increase the performance of Fed-IDSs, we are actively encouraging cyber-security researchers to develop robust solutions or incorporate effective class-balancing techniques into their designs. In the future, we plan

to evaluate the effectiveness of Fed-IDS by implementing robust aggregation techniques to mitigate class imbalance and protect against adversarial (Byzantine) attacks. Assessing how well the proposed IDS performs in Personalized Federated settings is another potential direction for future work.

REFERENCES

[1] K. Sood, S. Yu, D. D. N. Nguyen, Y. Xiang, B. Feng, and X. Zhang, "A tutorial on next generation heterogeneous IoT networks and node authentication," *IEEE Internet Things Mag.*, vol. 4, no. 4, pp. 120–126, Dec. 2021.

[2] K. Sood, M. R. Nosouhi, D. D. N. Nguyen, F. Jiang, M. Chowdhury, and R. Doss, "Intrusion detection scheme with dimensionality reduction in next generation networks," *IEEE Trans. Inf. Forensics Security*, vol. 18, pp. 965–979, 2023.

[3] A. Khraisat and A. Alazab, "A critical review of intrusion detection systems in the Internet of Things: Techniques, deployment strategy, validation strategy, attacks, public datasets and challenges," *Cybersecurity*, vol. 4, no. 1, pp. 1–27, 2021.

[4] S. A. Rahman, H. Tout, C. Talhi, and A. Mourad, "Internet of Things intrusion detection: Centralized, on-device, or federated learning?" *IEEE Netw.*, vol. 34, no. 6, pp. 310–317, Nov./Dec. 2020.

[5] T. D. Nguyen, S. Marchal, M. Miettinen, H. Fereidooni, N. Asokan, and A.-R. Sadeghi, "DÏoT: A federated self-learning anomaly detection system for IoT," in *Proc. IEEE 39th Int. Conf. Distrib. Comput. Syst. (ICDCS)*, 2019, pp. 756–767.

[6] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas, "Communication-efficient learning of deep networks from decentralized data," in *Proc. Artif. Intell. Stat.*, 2017, pp. 1273–1282.

[7] D. Y. Zhang, Z. Kou, and D. Wang, "FedSens: A federated learning approach for smart health sensing with class imbalance in resource constrained edge computing," in *Proc. IEEE Conf. Comput. Commun. (INFOCOM)*, 2021, pp. 1–10.

[8] S. K. Amalapuram, A. Tadwai, R. Vinta, S. S. Channappayya, and B. R. Tamma, "Continual learning for anomaly based network intrusion detection," in *Proc. 14th Int. Conf. COMmun. Syst. NETw. (COMSNETS)*, 2022, pp. 497–505.

[9] E. Tsogbaatar et al., "Del-IoT: A deep ensemble learning approach to uncover anomalies in IoT," *Internet Things*, vol. 14, Jun. 2021, Art. no. 100391.

[10] M. Zolanvari, M. A. Teixeira, and R. Jain, "Effect of imbalanced datasets on security of industrial IoT using machine learning," in *Proc. IEEE Int. Conf. Intell. Security Informat. (ISI)*, 2018, pp. 112–117.

[11] O. Sagi and L. Rokach, "Ensemble learning: A survey," *Wiley Interdiscip. Rev. Data Min. Knowl. Discov.*, vol. 8, no. 4, 2018, Art. no. e1249.

[12] S. Kaisar and A. Chowdhury, "Integrating oversampling and ensemble-based machine learning techniques for an imbalanced dataset in dyslexia screening tests," *ICT Express*, vol. 8, no. 4, pp. 563–568, 2022.

[13] M. Duan et al., "Astraea: Self-balancing federated learning for improving classification accuracy of mobile deep learning applications," in *Proc. IEEE 37th Int. Conf. Comput. Design (ICCD)*, 2019, pp. 246–254.

[14] J. Pang, Y. Huang, Z. Xie, Q. Han, and Z. Cai, "Realizing the heterogeneity: A self-organized federated learning framework for IoT," *IEEE Internet Things J.*, vol. 8, no. 5, pp. 3088–3098, Mar. 2021.

[15] L. Wang, S. Xu, X. Wang, and Q. Zhu, "Addressing class imbalance in federated learning," in *Proc. AAAI Conf. Artif. Intell.*, 2021, pp. 10165–10173.

[16] E. M. Campos et al., "Evaluating federated learning for intrusion detection in Internet of Things: Review and challenges," *Comput. Netw.*, vol. 203, Feb. 2022, Art. no. 108661.

[17] W. Liu et al., "Intrusion detection for maritime transportation systems with batch federated aggregation," *IEEE Trans. Intell. Transp. Syst.*, vol. 24, no. 2, pp. 2503–2514, Feb. 2023.

[18] T. DeVries and G. W. Taylor, "Dataset augmentation in feature space," 2017, *arXiv:1702.05538*.

[19] S. Marchal, M. Miettinen, T. D. Nguyen, A.-R. Sadeghi, and N. Asokan, "AuDI: Toward autonomous IoT device-type identification using periodic communication," *IEEE J. Sel. Areas Commun.*, vol. 37, no. 6, pp. 1402–1412, Jun. 2019.

[20] J. Li, L. Lyu, X. Liu, X. Zhang, and X. Lyu, "Fleam: A federated learning empowered architecture to mitigate DDoS in industrial IoT," *IEEE Trans. Ind. Informat.*, vol. 18, no. 6, pp. 4059–4068, Jun. 2022.

[21] S. I. Popoola, R. Ande, B. Adebisi, G. Gui, M. Hammoudeh, and O. Jogunola, "Federated deep learning for zero-day botnet attack detection in IoT-edge devices," *IEEE Internet Things J.*, vol. 9, no. 5, pp. 3930–3944, Mar. 2022.

[22] Y. Meidan et al., "N-BaIoT–network-based detection of IoT botnet attacks using deep autoencoders," *IEEE Pervasive Comput.*, vol. 17, no. 3, pp. 12–22, Jul.–Sep. 2018.

[23] V. Rey, P. M. S. Sánchez, A. H. Celdrán, and G. Bovet, "Federated learning for malware detection in IoT devices," *Comput. Netw.*, vol. 204, Feb. 2022, Art. no. 108693.

[24] O. Friha, M. A. Ferrag, L. Shu, L. Maglaras, K.-K. R. Choo, and M. Nafaa, "FELIDS: Federated learning-based intrusion detection system for agricultural Internet of Things," *J. Parallel Distrib. Comput.*, vol. 165, pp. 17–31, Jul. 2022.

[25] H. Liu et al., "Blockchain and federated learning for collaborative intrusion detection in vehicular edge computing," *IEEE Trans. Veh. Technol.*, vol. 70, no. 6, pp. 6073–6084, Jun. 2021.

[26] X. Shuai, Y. Shen, S. Jiang, Z. Zhao, Z. Yan, and G. Xing, "BalanceFL: Addressing class imbalance in long-tail federated learning," in *Proc. 21st ACM/IEEE Int. Conf. Informat. Process. Sens. Netw. (IPSN)*, 2022, pp. 271–284.

[27] B. Li, Y. Wu, J. Song, R. Lu, T. Li, and L. Zhao, "DeepFed: Federated deep learning for intrusion detection in industrial cyber–physical systems," *IEEE Trans. Ind. Informat.*, vol. 17, no. 8, pp. 5615–5624, Aug. 2021.

[28] R. Sauber-Cole and T. M. Khoshgoftaar, "The use of generative adversarial networks to alleviate class imbalance in tabular data: A survey," *J. Big Data*, vol. 9, no. 1, p. 98, 2022.

[29] E. Jothimurugesan, K. Hsieh, J. Wang, G. Joshi, and P. B. Gibbons, "Federated learning under distributed concept drift," 2023, *arXiv:2206.00799*.

[30] J. Wang, Q. Liu, H. Liang, G. Joshi, and H. V. Poor, "Tackling the objective inconsistency problem in heterogeneous federated optimization," in *Proc. 34th Adv. Neural Inf. Process. Syst.*, 2020, pp. 7611–7623.

[31] T. Li, A. K. Sahu, M. Zaheer, M. Sanjabi, A. Talwalkar, and V. Smith, "Federated optimization in heterogeneous networks," in *Proc. Mach. Learn. Syst.*, 2020, pp. 429–450.

[32] S. P. Karimireddy, S. Kale, M. Mohri, S. Reddi, S. Stich, and A. T. Suresh, "Scaffold: Stochastic controlled averaging for federated learning," in *Proc. Int. Conf. Mach. Learn.*, 2020, pp. 5132–5143.

[33] G. Hinton, O. Vinyals, and J. Dean, "Distilling the knowledge in a neural network," 2015, *arXiv:1503.02531*.

[34] J. Ren, C. Yu, X. Ma, H. Zhao, and S. Yi, "Balanced meta-Softmax for long-tailed visual recognition," in *Proc. Adv. Neural Inf. Process. Syst.*, 2020, pp. 4175–4186.

[35] T.-M. H. Hsu, H. Qi, and M. Brown, "Measuring the effects of non-identical data distribution for federated visual classification," 2019, *arXiv:1909.06335*.

[36] V. Valadi, X. Qiu, P. P. B. de Gusmão, N. D. Lane, and M. Alibeigi, "FedVal: Different good or different bad in federated learning," 2023, *arXiv:2306.04040*.

[37] P. Kairouz et al., "Advances and open problems in federated learning," *Founda. Trends® Mach. Learn.*, vol. 14, nos. 1–2, pp. 1–210, 2021.

[38] I. Ullah and Q. H. Mahmoud, "A scheme for generating a dataset for anomalous activity detection in IoT networks," in *Proc. 33rd Canadian Conf. Artif. Intell.*, 2020, pp. 508–520.

[39] H. Kang, D. H. Ahn, G. M. Lee, J. D. Yoo, K. H. Park, and H. K. Kim, 2019, "Iot network intrusion dataset," Dataset, IEEE DataPort, 2019. [Online]. Available: https://dx.doi.org/10.21227/q70p-q449

[40] M. Zolanvari, L. Gupta, K. Khan, and R. Jain, "*Wustl-iiot-2o2l Dataset for IIoT Cybersecurity Research*, Washington Univ. St. Louis, St. Louis, MO, USA, 2021.

[41] M. Zolanvari, M. A. Teixeira, L. Gupta, K. M. Khan, and R. Jain, "Machine learning-based network vulnerability analysis of industrial Internet of Things," *IEEE Internet Things J.*, vol. 6, no. 4, pp. 6822–6834, Aug. 2019.

[42] M. A. Ferrag, O. Friha, D. Hamouda, L. Maglaras, and H. Janicke, "Edge-IIoTset: A new comprehensive realistic cyber security dataset of IoT and IIoT applications for centralized and federated learning," *IEEE Access*, vol. 10, pp. 40281–40306, 2022.

[43] S. Samarakoon et al., "5G-NIDD: A comprehensive network intrusion detection dataset generated over 5G wireless network," 2022, *arXiv:2212.01298*.

Gurpreet Singh received the bachelor's and master's degrees in mechanical engineering with the specialization in Manufacturing Systems Engineering from the Sant Longowal Institute of Engineering and Technology, Punjab, India, in 2019 and 2021, respectively. He is currently pursuing the Ph.D. degree with the Indian Institute of Technology Hyderabad, India, and Deakin University, Australia. His doctoral research is centered on network security, particularly intrusion detection systems for next generation networks, employing advanced techniques, such as deep learning and federated learning.

Keshav Sood (Senior Member, IEEE) received the Bachelor of Technology degree (Hons.) in electronics and communications engineering with distinction and the Master of Technology degree in optical fiber engineering in 2007 and 2012, respectively, and the Ph.D. degree in information technology (software-defined networking security) from Deakin University, Melbourne, in 2018, under the supervision of Prof. Shui Yu. He was a Trainee with the Terminal Ballistic Research Laboratory, Defence Research and Development Organization, Ministry of Defense, Chandigarh, India. He completed the Postdoctoral with the University of Newcastle, NSW, Australia. He is currently working as a Lecturer with Cyber Security, Deakin University School of Information Technology. His work in cyber security for next-generation networks has been published in top-notch security and networking venues, i.e., IEEE TRANSACTIONS ON INFORMATION FORENSICS AND SECURITY, IEEE TRANSACTIONS ON DEPENDABLE AND SECURE COMPUTING, IEEE/ACM TRANSACTIONS ON NETWORKING, IEEE TRANSACTIONS ON NETWORK AND SERVICE MANAGEMENT, and IEEE NETWORK MAGAZINE. He has been successful in obtaining a Research Funding with the Department of Defence, Australia, Cyber CRC, and industry partners. He was the recipient of the Professor of Information Technology Award given by the School of Information Technology for his outstanding academic achievements during the Ph.D. degree. He is an Associate Editor of IEEE INTERNET OF THINGS JOURNAL and *IET Networks*.

P. Rajalakshmi (Senior Member, IEEE) received the Ph.D. in electrical engineering from the Indian Institute of Technology Madras, India, in 2008. She is currently holds the esteemed position of Professor with the Department of Electrical Engineering, Indian Institute of Technology Hyderabad, India, where she serves as the Cyient Chair Professor of Future Communication and is the Director of NMICPS TiHAN Foundation. She actively contributes to advancing these fields and exploring innovative solutions in wireless technology. She leads the Wireless Networks Lab, focusing on wireless communications, wireless sensor networks, UAV-based sensing, embedded systems, cyber-physical systems/Internet of Things, converged network modeling, energy efficiency, and green communications.

Dinh Duc Nha Nguyen received the master's degree from the Queensland University of Technology, Australia, and the Ph.D. degree from Deakin University, VIC, Australia. He is currently an Associate Research Fellow with the Socrates Project (Software Security with a focus on critical technologies), Deakin University. During the Ph.D., he published papers in Q1 cybersecurity journals and some of his work was funded by the Department of Defense, Australia. He has been awarded a place on the QUT Dean's list for excellent academic performance two consecutive times in 2020, and the School of Information Technology Higher Degree by Research Award for the 2022 Academic Year, recognizing the highest research, and academic merit. He also secured a prize in the 2022 Country to Country CTF Cybersecurity Challenge, a prestigious international competition hosted by MIT and organized by INCS-CoE with leading universities.

Yong Xiang (Senior Member, IEEE) received the Ph.D. degree in electrical and electronic engineering from the University of Melbourne, Australia. He is a Professor with the School of Information Technology, Deakin University, Australia. He has published 7 authored books, over 260 refereed journal articles, and over 110 conference papers in these areas. His research interests include distributed computing, cybersecurity and privacy, machine learning and AI, and communications engineering. He is the Senior Area Editor of IEEE SIGNAL PROCESSING LETTERS, an Associate Editor of IEEE COMMUNICATIONS SURVEYS AND TUTORIALS, and an Associate Editor of *Computer Standards and Interfaces*. He was an Associate Editor of IEEE SIGNAL PROCESSING LETTERS and IEEE ACCESS, and the Guest Editor of IEEE TRANSACTIONS ON INDUSTRIAL INFORMATICS and IEEE MULTIMEDIA. He has served as a Honorary Chair, the General Chair, the Program Chair, the TPC Chair, the Symposium Chair, and the Track Chair for many conferences, and was invited to give keynotes at numerous international conferences.