FELACS: Federated Learning with Adaptive Client Selection for IoT DDoS Attack Detection

Mulualem Bitew Anley, Pasquale Coscia, Angelo Genovese*, Vincenzo Piuri

Department of Computer Science, Università degli Studi di Milano, Italy

Abstract

Distributed denial-of-service (DDoS) attacks pose a significant threat to network security by overwhelming systems with malicious traffic, leading to service disruptions and potential data breaches. The traditional centralized machine learning (ML) methods for detecting DDoS attacks in Internet of Things (IoT) environments raise privacy and security concerns due to their collection and distribution of data to a central entity that may not be trusted to perform model training. Federated learning (FL) offers a privacy-preserving solution that enables distributed collaboration by training a model only on local clients, without data exchanges, where the central entity only performs global model aggregation. However, the current practice of random client selection, combined with the statistical heterogeneity of client data and the device heterogeneity encountered in IoT environments, requires many training rounds to reach optimal accuracy, increasing the imposed computational overhead. To address these challenges, we propose a multiobjective optimization-based FL with adaptive client selection (FELACS) approach that maximizes client importance scores while satisfying resource, performance, and data diversity constraints. Experiments are carried out on the CIC-IDS2018, CIC-DDoS2019, BoT-IoT, and CIC-IoT2023 datasets, demonstrating that FELACS improves upon the accuracy of the existing approaches while exhibiting increased convergence speed when training a model in an FL scenario, hence reducing the number of communication rounds required to achieve the target accuracy, making it highly effective for performing IoT-based DDoS attack detection in FL scenarios.

*Corresponding author

Email addresses: mulualem.anley@unimi.it (Mulualem Bitew Anley), pasquale.coscia@unimi.it (Pasquale Coscia), angelo.genovese@unimi.it (Angelo Genovese), vincenzo.piuri@unimi.it (Vincenzo Piuri)

Keywords: Adaptive client selection, Federated learning, DDoS attack detection, Cybersecurity, IoT security, Privacy-preserving model.

1. Introduction

Internet of Things (IoT) devices range from personalized health systems and home appliances to smart cities, playing a crucial role in everyday activities while frequently managing sensitive personal data. Safeguarding the privacy and security of these devices thus remains a critical challenge, with distributed denial-of-service (DDoS) attacks currently being among the major threats encountered in IoT environments (Pakmehr et al., 2024). Centralized intrusion detection systems (IDSs) based on machine learning (ML) and deep learning (DL) have been extensively researched for detecting these attacks (Gümüşbaş et al., 2021; Aktar and Nur, 2023). However, centralized DL-based IDSs are often inadequate for scaling and adapting to the distributed nature and privacy requirements of IoT environments (Fotse et al., 2024). Federated learning (FL) has since been considered for detecting cybersecurity threats, enabling collaborative data analyses without requiring the transfer of private data between organizations (McMahan et al., 2017; Nguyen et al., 2025; Mazid et al., 2025; Rahmati, 2025; Olanrewaju-George and Pranggono, 2025).

FL begins with the central server distributing an initial model to a subset of the selected clients. The client selection process plays a crucial role, as it directly influences both the performance and convergence rate of the global model. In IoT-based environments, additional client selection challenges are posed by the high degree of device heterogeneity. In fact, IoT devices produce datasets that can vary significantly in their sizes and feature spaces, resulting in devices that possess many samples for certain types of DDoS attacks while having minimal data for others. Such heterogeneity necessitates distinct representations of data across different organizations, which is a requirement that existing FL frameworks might not fulfill without adaptations for addressing diverse feature spaces and device capabilities. Furthermore, DDoS attacks executed in IoT environments exhibit different characteristics across various devices, further complicating the analysis due to their statistical heterogeneity (Li et al., 2024; Anley et al., 2025).

The client selection approaches described in the literature often do not explicitly account for the relevance and importance of the data held by each client since the selection of clients traditionally occurs via either a random or semirandom approach (McMahan et al., 2017; Nishio and Yonetani, 2019; Wu et al., 2023).

In fact, such approaches can lead to the selection of less valuable data, resulting in suboptimal model performance in the presence of heterogeneous clients (Fu et al., 2023). While more recent works have proposed algorithms that consider criteria such as accuracy, CPU capacity, memory, energy levels, and response times (Zhao et al., 2018; Li et al., 2021; Wang et al., 2020; AbdulRahman et al., 2020; Seo et al., 2022; Maciel et al., 2023; Sun et al., 2024; Nguyen et al., 2025; Mazid et al., 2025), they do not dynamically adapt to changing data quality and device performance conditions throughout the training rounds of FL, resulting in possible model training inefficiencies and slower convergence speeds. In fact, most of the existing methods focus on either data importance or computational efficiency but rarely combine these factors into a unified approach. Furthermore, most current client selection strategies assume static client participation, whereas real-world IoT environments are characterized by high client mobility rates and frequent connectivity disruptions (Zhang et al., 2024).

In this paper, we propose federated learning with adaptive client selection (FELACS): a novel and adaptive framework that is designed to optimize the client participation level for performing DDoS detection in IoT environments.[1] FELACS introduces a multicriteria, multiobjective optimization-based selection strategy that integrates client resource constraints, data diversity, relevance, and system efficiency for each training round. This comprehensive framework evaluates key factors such as data entropy, statistical variance, computational efficiency, energy consumption, and communication latency. By leveraging these diverse criteria, FELACS ensures the adaptive selection of high-impact clients in each round, facilitating robust model updates, a faster convergence speed, and improved detection accuracy. Moreover, our proposed approach addresses the unique challenges posed by DDoS attack detection in IoT-based FL scenarios, where timely and accurate client selection is critical for achieving effective real-time threat detection. In this context, FELACS prioritizes clients with higher-level representations of attack-related data, thus improving the relevance and quality of the updates derived from participating clients. This strategy accelerates the attack detection procedure, reduces the required convergence time, and enhances the responsiveness of the system. By shifting from resource-centric selection to a data-driven and adaptive approach, FELACS establishes a new paradigm for selecting clients in FL settings.

The remainder of the paper is organized as follows. Section 2 provides an

[1] The source code will be made available upon the acceptance of this work at https://github.com/mulerkal/FELACS.

overview of the related works in the field of FL for IoT security, IoT IDSs, and DDoS attack detection, with a focus on client selection strategies. Section 3 describes the system model, problem formulation, and FL learning process of the developed method. Section 4 outlines the methodology and algorithm of our proposed approach, including the experimental design, dataset details, and model architecture. Section 5 presents the results of our proposed methodology and a corresponding performance analysis. Finally, Section 6 summarizes this work and concludes the paper.

2. Related Works

FL in IoT Security and Intrusion Detection Scenarios. Unlike centralized ML-based DDoS attack detection (Agostinello et al., 2023; Anley et al., 2024), which poses challenges such as privacy risks, high latency, and potential network congestion (Feng et al., 2021), an FL-based IDS provides a decentralized alternative in which the model training process is distributed across different IoT devices, thereby minimizing the data transfer requirements and reducing privacy concerns (Li et al., 2020a; Kairouz et al., 2021; Lim et al., 2020). FL-based IDSs can achieve comparable accuracy to that of centralized methods while preserving data privacy (Roy et al., 2023) and simultaneously addressing the distributed nature of IoT networks, where devices may have varying computational capabilities and data characteristics (Ruzafa-Alcázar et al., 2021). In fact, several works have explicitly considered FL tasks with nonindependent and identically distributed (non-IID) data, which are caused by IoT devices often having skewed data distributions, with certain devices collecting extensive data on specific attack types while possessing limited data on other types (Ma et al., 2022; Zhao et al., 2018). Because of the presence of non-IID data, traditional aggregation techniques such as federated averaging (FedAvg), which consider the contributions of each client equally (McMahan et al., 2017), struggle to manage the high degree of data heterogeneity encountered in real-world scenarios. This heterogeneity can hinder the overall performance and convergence speed of the global model.

Client Selection in FL. In FL, the learning process is distributed across multiple clients, which collaboratively train a global model by sharing their locally computed updates instead of raw data. Efficient client selection mechanisms ensure an effective model training procedure while optimizing the incurred communication costs and preserving the diversity of the given data (McMahan et al., 2017; Bonawitz et al., 2019). Studies on the selection of clients in FL tasks have

explored various strategies that consider client availability, data quality, computational capacity, and model performance contributions (Wang et al., 2020; Nishio and Yonetani, 2019).

The simplest way to perform client selection, other than a random-based approach (McMahan et al., 2017; Nishio and Yonetani, 2019; Wu et al., 2023; Ginanjar et al., 2025), consists of considering only the contributions of clients to reducing the loss function of the global model. For example, the approaches described in (Wang et al., 2020; Seo et al., 2022) prioritize clients with higher expected model accuracy contributions, improving their convergence speeds relative to that of random client selection, particularly in heterogeneous data environments where some clients possess more informative or diverse data than others do. However, these approaches overlook the computational capacities of clients and the associated communication costs, potentially leading to increased latency and inefficiencies in resource-constrained IoT networks. To improve this model, more recently developed approaches also consider resources and device capabilities (Maciel et al., 2023). For example, the federated proximal (FedProx) approach was designed to handle systems with varying computational resources by enabling partial client participation during training, thus reducing the update variability caused by heterogeneous datasets (Li et al., 2020b). However, FedProx does not account for the diversity and significance of client data during the selection process. Extensions to resource-aware client selection are represented by the FedCS (Nishio and Yonetani, 2019) and FedMCCS (AbdulRahman et al., 2020) methods, which select clients on the basis of availability, computational power, and communication bandwidth; dynamically adjust the number of selected clients; and allocate more resources to those with higher capabilities. However, these approaches assume that clients are willing to fully disclose their resource states, which may not be feasible because of privacy concerns or the risk of unreliable self-reporting in real-world IoT environments.

Recent advances have also explored client selection approaches based on data diversity to increase the generalizability of FL models. In the method described in (Zhao et al., 2018), the issue of data heterogeneity is addressed through a data-sharing mechanism, where a small amount of globally shared data is used to align the learning objectives of different clients. This method improves the convergence of the global model but undermines the fundamental privacy-preserving principle of FL, as even minimal data sharing could expose sensitive information. To counteract this issue, the method proposed in (Li et al., 2021) considers a diversity-aware client selection scheme that encourages participation from clients with diverse data distributions, thereby enhancing the robustness of the model without

requiring data sharing. While promising, this approach relies heavily on the accurate estimation of data diversity, which may not always be feasible, especially in environments with highly dynamic IoT network data distributions.

To account for both data- and resource-based device client characteristics, multicriteria client selection schemes have been proposed for ensuring reliable and secure participation in FL. As an example, (Tahir et al., 2025) proposed SecureFedPROM, a framework that integrates security, performance, and system efficiency metrics into a weighted multiarmed bandit (MAB) model for dynamically ranking clients. The method introduced in (Ami et al., 2025) further leverages an MAB-based scheduler to learn the computational and network profiles of clients over time, thereby minimizing the training latency induced per round.

3. System Model and Problem Formulation

In this section, we introduce FELACS, an adaptive client selection algorithm that is designed to enhance the FL-based DDoS detection process in IoT environments. We begin by describing the system model and then formalize the FELACS problem. Table 1 summarizes the notations and symbols used in the equations presented in this work.

3.1. System Model

The proposed FELACS framework operates within an FL environment to detect DDoS attacks in IoT networks. The system consists of a central server and a set of distributed clients $\mathcal{N}$, where each client $k \in \mathcal{N}$ owns a local dataset $\mathcal{D}_k$ consisting of traffic from its private network. The central server maintains a global model $\mathbf{w}_t$, which is a classifier of the normal and DDoS traffic encountered in iteration t. In each iteration t, each client downloads the model, trains it on its local data, and transmits updates back to the server.

After an initialization phase, the training process is divided into communication rounds, each of which consists of three phases: *i)* client selection, *ii)* client updating, and *iii)* model aggregation. The process then follows the subsequent steps until the maximum number of communication rounds is reached or until the convergence criteria are satisfied.

Step 1: Client Selection. In each round t, the server assesses the capabilities of each device (i.e., its network latency, power consumption, and computational resources) and uses the proposed FELACS algorithm to select a subset of clients $S_t \subseteq \mathcal{N}$ from the total pool of clients $\mathcal{N}$, with $|\mathcal{N}| = K$ total clients. This step is explained in detail in Section 3.2.

Table 1: Notations and descriptions used throughout the paper.

Symbol	Description
$\mathcal{N}$	Set of all clients participating in FL.
k	Client index, where $k \in \mathcal{N}$.
$\mathcal{D}_k$	Local dataset held by client k.
t	Communication round index.
$S_t \subseteq \mathcal{N}$	Subset of clients selected in round t.
$\mathbf{w_t}$	Global model in round t.
$\mathbf{w_k^t}$	Updated local model parameters after client k trains on $\mathcal{D}_k$ in round t.
$\nabla F_k(\mathbf{w_t})$	Gradient of the loss function F_k of client k evaluated at $\mathbf{w_t}$.
η	Learning rate for local stochastic gradient descent (SGD) updates.
K	Upper bound imposed on the number of clients selected per round.
$R_{\max}$	Total resource budget available for client selection.
$L_{\max}$	Maximum allowable communication latency per client.
$H_{\min}$	Minimum Shannon entropy required for the data of a client to qualify.
$\alpha, \beta, \gamma, \delta$	Hyperparameters for weighting the components of the importance score.
I_i	Importance score of client i, defined as $I_i = \alpha\, H(\mathcal{D}_i) + \beta\, V(\mathcal{D}_i) - \gamma\, P_i - \delta\, L_i$.
E	Number of local training epochs executed on each selected client per round.
y_i^c	One-hot encoded true label of sample i for class c.
$f(x_i; \mathbf{w_t})_c$	Predicted probability of class c for input x_i under model $\mathbf{w_t}$.
C	Total number of classes involved in the classification task.
n_k	Number of data points contained in $\mathcal{D}_k$.
$n \sum_{k \in S_t} n_k$	Total number of samples across all selected clients in round t.
R_i	Resource consumption (e.g., CPU cycles) of client i.
P_i	Power consumption of client i.
L_i	Network latency of client i.
u_i	Utility score of client i.
$H(\mathcal{D}_i)$	Shannon entropy of the data distribution at client i.
$V(\mathcal{D}_i)$	Variance of the data distribution of client i.
T_{dl}	$\frac{\text{model size}}{\text{downlink bandwidth}} + \text{latency}$.
$T_{\text{comp}}^{(i)}$	$E \cdot \frac{N_i}{\text{batch size}} \cdot t_{\text{batch}}^{(i)}$.
$T_{\text{ul}}^{(i)}$	$\frac{\text{update size}}{\text{uplink bandwidth}} + \text{latency}$.
T_{agg}	Aggregation time per round.

Step 2: Client Updating. The selected clients download the current global model $\mathbf{w}_t$. Each client $k \in S_t$ trains the model on its local dataset $\mathcal{D}_k$ for E epochs and computes the updated local model parameters $\mathbf{w}_t^k$. The update process is as follows:

$$\mathbf{w}_t^k = \mathbf{w}_t - \eta \nabla F_k(\mathbf{w}_t) \ , \tag{1}$$

where η is the learning rate and $\nabla F_k(\mathbf{w}_t)$ represents the gradient of the local loss function $F_k(\mathbf{w}_t)$, which quantifies the error induced by the model on the local dataset of the client $\mathcal{D}_k$. We define the local loss function $F_k(\mathbf{w}_t)$ as the cross-entropy loss:

$$F_k(\mathbf{w}_t) = -\frac{1}{|\mathcal{D}_k|} \sum_{i \in \mathcal{D}_k} \sum_{c=1}^{C} y_i^c \log f(x_i; \mathbf{w}_t)^c \ , \tag{2}$$

where y_i^c is the one-hot encoded label produced for class c, $f(x_i; \mathbf{w}_t)^c$ is the predicted probability for class c, and C is the total number of classes.

Step 3: Model Aggregation. The server aggregates the local updates obtained from the selected clients to update the global model. The server-side steps involve collecting updates $\mathbf{w}_t^k$ from each selected client $k \in S_t$ and aggregating the updates to form a new global model, thereby obtaining $\mathbf{w}_{t+1}$:

$$\mathbf{w}_{t+1} = \sum_{k \in S_t} \frac{n_k}{n} \mathbf{w}_t^k \ , \tag{3}$$

where n_k is the number of data points for client k and $n = \sum_{k \in S_t} n_k$. By aggregating these updates from the selected clients, the global model optimizes the overall objective:

$$F(\mathbf{w}_t) = \frac{1}{K} \sum_{k=1}^{K} F_k(\mathbf{w}_t) \ , \tag{4}$$

where K is the total number of clients. The system model is detailed in Algorithm 1.

3.2. FELACS Problem Formulation

The objective of FELACS is to increase the convergence rate and maximize the DDoS attack detection accuracy of the developed model by selecting an optimal subset S_t of clients during each communication round t. The client selection process is framed as an optimization problem aimed at maximizing the achieved

Algorithm 1: FELACS System Model

Input: Client pool $\mathcal{N}$, total number of rounds T, maximum number of clients per round K, number of local epochs E, learning rate η.

Output: Global model $\mathbf{w}_t$.

Initialization.

$\mathbf{w}_0 \leftarrow$ initial model parameters

for $t \leftarrow 0$ **to** $T-1$ **do**

Step 1: Client Selection.

$S_t \leftarrow \textsc{SelectClients}(\mathcal{N}, K)$(see Algorithm 2)

Step 2: Client Updating.

foreach $k \in S_t$ ***in parallel*** **do**

Client k downloads $\mathbf{w}_t$ and trains for E epochs on $\mathcal{D}_k$

$\mathbf{w}_t^k \leftarrow \mathbf{w}_t - \eta \nabla F_k(\mathbf{w}_t)$

Step 3: Model Aggregation.

$n_k =$ number of data points for client k

$$\mathbf{w}_{t+1} = \sum_{k \in S_t} \frac{n_k}{\sum_{j \in S_t} n_j} \mathbf{w}_t^k$$

detection accuracy while respecting the imposed computational and communication constraints, and it is guided by the total importance score I_i of the selected clients. The **importance score** for client i is defined as follows:

$$I_i = \alpha H(\mathcal{D}_i) + \beta V(\mathcal{D}_i) - \gamma P_i - \delta L_i \ , \tag{5}$$

where $H(\mathcal{D}_i)$ is the Shannon entropy of the data of client i, representing the diversity of the data; $V(\mathcal{D}_i)$ is the variance of the local data of client i, which reflects the statistical heterogeneity of the data distribution of this client; P_i represents the power consumption of client i; L_i represents the network latency of client i; and $\alpha, \beta, \gamma, \delta$ are hyperparameters for controlling the relative influence of each factor (see also Table 1).

We model the process of selecting an optimal subset of clients S_t in each round t as a multiobjective optimization problem, specifically a multidimensional knapsack problem (MDKP). In this formulation, each candidate device i is characterized by a cost vector (resource, power, latency) and a utility that reflects its data relevance and expected contribution to the global model. The objective is to maximize the total utility under the constraint that the sum of the costs does not

exceed the available budget.

We formulate the optimization problem to maximize the total importance score of the selected clients while respecting the imposed resource, power, latency, data diversity, and participation constraints (Equation 6):

$$\underset{S_t}{\text{argmax:}} \sum_{i \in S_t} (I_i) \tag{6}$$

$$\text{subject to:} \sum_{i \in S_t} R_i \leq R_{\text{max}}, \tag{7}$$

$$\sum_{i \in S_t} P_i \leq P_{\text{max}}, \tag{8}$$

$$L_i \leq L_{\text{max}}, \ \forall i \in S_t, \tag{9}$$

$$H(\mathcal{D}_i) \geq H_{\text{min}}, \ \forall i \in S_t, \tag{10}$$

$$|S_t| \leq K. \tag{11}$$

The optimization problem must satisfy the following constraints.

1. *Client resource constraints*. The total resource usage $\sum_{i \in S_t} R_i$ of the selected clients must not exceed the available resource budget R_{max}. This constraint helps manage the computational and communication limitations of IoT devices, ensuring an efficient client selection process without overburdening the system (Equation 7).
2. *Power constraint.* The total power consumption level $\sum_{i \in S_t} P_i$ for the selected clients should be bounded by P_{max} to ensure that battery-powered devices retain sufficient energy for completing their training procedures (Equation 8).
3. *Latency constraints*. The latency L_i of each selected client should be bounded by L_{max} to ensure timely model updates (Equation 9).
4. *Data diversity constraints*. To ensure diversity, clients with high data redundancy levels are excluded. The entropy of the data of the selected clients $H(\mathcal{D}_i)$ must exceed a threshold H_{min} (Equation 10).
5. *Client selection constraints*. The number of selected clients $|S_t|$ should not exceed a predefined limit K (Equation 11).

The defined multidimensional client selection problem is a nondeterministic polynomial-time (NP)-hard problem, so we adopt a fast three-step greedy heuristic approach. *i)* First, we discard any client i for which $L_i > L_{\text{max}}$ or $H(\mathcal{D}_i) < H_{\text{min}}$,

thereby enforcing the latency and diversity constraints up front. For each remaining client, we compute the importance-per-resource ratio ρ_i as shown below:

$$\rho_i = \frac{I_i}{R_i} \ , \tag{12}$$

which quantifies the marginal benefit of selecting i relative to its cost. The importance-per-resource ratio balances its potential benefit against its resource consumption level. *ii)* Then, we sort the clients in descending order of ρ_i (an $\mathcal{O}(n \log n)$ operation). *iii)* Finally, we traverse this list from the highest to the lowest values of ρ_i and iteratively add clients to S_t until adding a further client would violate a constraint ($\mathcal{O}(n)$ per pass). In this case, we apply an algorithm to repair constraint violations by removing the violating client and looking for the next-best client in the ρ_i rankings until we find a candidate whose addition does not violate any constraint. Algorithm 2 details the proposed client selection algorithm.

We adopt a three-step greedy heuristic for FELACS because it runs in $\mathcal{O}(n \log n)$ time, uses only linear memory, and imposes negligible computation and communication overhead, which are critical properties for resource-constrained, intermittently connected IoT devices. Alternative approaches for solving the MDKP include integer linear programming (ILP) solvers, LP relaxation with rounding (Anand et al., 2017; Puchinger et al., 2010), population-based metaheuristics (Elfaki et al., 2023), and multiobjective optimization frameworks tailored to FL (Hu et al., 2022). While these approaches offer various optimality or approximation guarantees, they typically exhibit exponential runtime growth, require careful parameter tuning, or demand substantial communication and memory resources, leading to elevated latency and energy consumption levels, which are untenable in intermittently connected IoT networks.

4. Evaluation Methodology

4.1. Experimental Design

We use the Flower[2] FL framework for our simulations because of its versatility and customization options that are tailored to the IoT and client-server configurations (Beutel et al., 2020).

[2] https://flower.ai

Algorithm 2: FELACS Algorithm for Client Selection

Input: Client pool $\mathcal{N}$, maximum number of clients K, resource budget $R_{\max}$, latency $L_{\max}$, entropy threshold $H_{\min}$, weights $(\alpha, \beta, \gamma, \delta)$.

Output: Selected client subset S_t.

Step 1.

Compute importance score I_i.

foreach $i \in \mathcal{N}$ **do**

$I_i \leftarrow \alpha\, H(\mathcal{D}_i) + \beta\, V(\mathcal{D}_i) - \gamma\, P_i - \delta\, L_i$

Compute importance-per-resource ratio ρ_i.

foreach $i \in \mathcal{N}$ **do**

$\rho_i \leftarrow \frac{I_i}{R_i}$

Step 2: Sort clients.

Sort $\mathcal{N}$ in descending order of ρ_i

Step 3.

Client selection under constraints.

$S_t \leftarrow \emptyset$

foreach i *in sorted* $\mathcal{N}$ **do**

if $|S_t| < K$ ***and*** $\sum_{j \in S_t} R_j + R_i \leq R_{\max}$ ***and*** $L_i \leq L_{\max}$ ***and*** $H(\mathcal{D}_i) \geq H_{\min}$ **then**

$S_t \leftarrow S_t \cup \{i\}$

Repair constraint violations.

foreach $j \in S_t$ ***if*** $(L_j > L_{\max}$ ***or*** $H(\mathcal{D}_j) < H_{\min})$ **do**

Find next-best $k \notin S_t$ with the highest I_k satisfying all constraints

Replace j with k in S_t

return S_t

4.2. Datasets

We perform our experiments on widely applied benchmark IDS and DDoS attack detection datasets. Specifically, we consider the CIC-IDS2018 (Sharafaldin et al., 2018), CIC-DDoS2019 (Sharafaldin et al., 2019), BoT-IoT (Koroniotis et al., 2019), and CIC-IoT2023 (Neto et al., 2023) datasets. These datasets enable both binary and multiclass classification, as well as FL model training. Moreover, BoT-IoT and CIC-IoT2023 describe realistic IoT traffic in modern networks. Table 2 lists the different datasets in terms of their classes and numbers of samples.

To simulate an FL scenario for a dataset that does not explicitly divide its data into clients, in this paper, we distribute the preprocessed dataset among five clients

Table 2: Traffic types and their cardinality across four datasets.

CIC-IDS2018		CIC-DDoS2019		BoT-IoT		CIC-IoT2023	
Type	**Card.**	**Type**	**Card.**	**Type**	**Card.**	**Type**	**Card.**
Benign	1 603 315	TFTP	4 396 770	Benign	590 298	DDoS	5 338 243
LoIC-HTTP	143 746	UDP	2 510 574	DDoS	379 302	DoS	1 269 264
HOIC	49 677	NTP	1 112 902	Reconn	108 795	Mirai	413 754
Hulk	36 227	SSDP	891 220	Theft	31 746	Benign	172 642
SlowHTTPTest	10 308	SYN	687 524	Scan	9 164	Spoofing	76 807
GoldenEye	2 419	MSSQL	484 070	–	-	Recon	55 531
Slowloris	433	SNMP	114 179	–	–	Web	3 836
LOIC-UDP	154	DNS	113 252	–	–	BruteForce	1 988
–	–	BENIGN	99 154	–	–	–	–
–	–	LDAP	48 469	–	–	–	–
–	–	NetBIOS	31 052	–	–	–	–
–	–	Portmap	1 638	–	–	–	–
Total: 1 846 279		**Total:** 10 491 218		**Total:** 1 119 305		**Total:** 7 332 065	

Table 3: Data distribution of the CIC-IDS2018 dataset among five clients after performing preprocessing to simulate non-IID data.

Traffic Type	**IoT1**	**IoT2**	**IoT3**	**IoT4**	**IoT5**
Benign	269 431	351 565	503 600	248 003	230 716
LOIC-HTTP	24 155	31 519	45 150	22 234	20 688
HOIC	8 347	10 891	15 602	7 683	7 154
Hulk	6 087	7 943	11 378	5 602	5 217
SlowHTTPTest	1 731	2 260	3 237	1 593	1 487
GoldenEye	406	529	759	373	352
Slowloris	71	94	135	66	67
LOIC-UDP	29	41	17	31	36
Total	310 230	404 804	579 865	285 555	265 687

using random sampling to simulate its non-IID nature. Table 3 shows an example of the data distribution for the CIC-IDS2018 dataset, where each client is assigned a varying number of samples from each class to simulate a realistic non-IID data distribution. The purpose of this setting is to evaluate the performance of our model in terms of handling heterogeneous data distributions, which are commonly encountered in IoT environments. In the following, we briefly describe the datasets and their main characteristics.

CIC-IDS2018. This dataset was specifically designed to simulate realistic network traffic[3], and it includes various attack scenarios captured over five days, including

[3] https://www.unb.ca/cic/datasets/ids-2018.html

Table 4: Data distribution of the BoT-IoT dataset.

Traffic type	IoT1	IoT2	IoT3	IoT4	IoT5
Benign	295 649	148 000	73 000	36 500	37 149
DDoS	189 651	94 825	47 413	23 706	23 707
Reconn	54 395	27 197	13 599	6 800	6 804
Theft	15 873	7 937	3 968	1 984	1 984
Scan	4 582	2 291	1 145	572	574
Total	560 150	280 250	139 125	69 562	70 218

DoS, DDoS, brute force, and infiltration attacks. Each record contained in the dataset comprises 80 features extracted using CICFlowMeter, including details regarding time stamps, source and destination IPs, protocols, and packet lengths (Sharafaldin et al., 2018).

CIC-DDoS2019. This dataset, which was released by the Canadian Institute for Cybersecurity, offers a realistic benchmark for performing DDoS detection by combining benign background traffic synthesized via the B-Profile system to emulate 25 users across HTTP, HTTPS, FTP, SSH, and email, with a diverse mix of reflection and amplification attacks[4]. The training set includes 12 distinct attack variants, whereas the testing set covers 7 variants. All traffic was captured as raw PCAPs per host and processed with CICFlowMeter to extract over 80 flow-based features, yielding tens of millions of labeled flows for both binary and multiclass DDoS research (Sharafaldin et al., 2019).

BoT-IoT. This dataset was collected at the University of New South Wales (UNSW) Canberra's Cyber Range Lab and describes normal and malicious DDoS traffic in a realistic network environment[5]. The data were collected from IoT devices, including refrigerators, sound systems, thermostats, lights, and locks. These devices are represented as IoT1 through IoT5, as depicted in Table 4. Each device simulates different network environments and attack patterns, allowing for a comprehensive evaluation of the proposed approach in an FL scenario. This dataset covers various cyber threats, including DDoS, DoS, reconnaissance, service scan, and theft attacks (Koroniotis et al., 2019).

CIC-IoT2023. This dataset describes large-scale IoT attack data captured on a realistic testbed consisting of 105 heterogeneous devices, featuring 33 distinct

[4] https://www.unb.ca/cic/datasets/ddos-2019.html
[5] https://research.unsw.edu.au/projects/bot-iot-dataset

attack variants across seven categories (DDoS, DoS, reconnaissance, web exploit, brute-force, spoofing, and Mirai botnet commands) against benign traffic[6] (Neto et al., 2023).

4.3. Data Distributions

For each of the considered datasets, we randomly allocate 80% of the records to the training partition and reserve the remaining 20% of the data for centralized testing. The training subsets are then distributed across K federated nodes to approximate realistic IoT deployments. Each node holds a distinct slice of the data that reflects its local traffic conditions. Moreover, the held-out 20% remains the same across all the experiments to ensure a fair performance comparison for the global model.

To induce non-IID heterogeneity, we employ Dirichlet sampling $\mathrm{Dir}(\alpha)$ over the class labels, which is a commonly used technique in FL studies (Hsu et al., 2019). Specifically, for each node k and each traffic class c (benign or one of the attack types), we draw proportions $p_{k,c} \sim \mathrm{Dir}(\alpha)$ and assign roughly $p_{k,c} \times N_{\mathrm{train}}$ samples from class c to node k, where N_{train} is the total number of training records. By varying α, we control the skewness, thus defining three levels of heterogeneity: mild, moderate, and severe. With $\alpha = 10$, the class distributions across the nodes remain close to IID *(mild heterogeneity)*, $\alpha = 1$ produces a noticeable imbalance between the benign and malicious traffic *(moderate heterogeneity)*, and $\alpha = 0.1$ yields highly skewed partitions in which some nodes see almost exclusively benign samples or almost exclusively attack samples *(severe heterogeneity)*.

4.4. Model Architecture

The convolutional neural network (CNN) architecture used in our experiments is specifically designed for detecting DDoS attacks in IoT environments. It uses two Conv1D layers, each of which is followed by rectified linear unit (ReLU) activation and batch normalization functions, to enhance its feature stability. Maximum pooling layers are interleaved to reduce the spatial dimensionality of the network, and dropout layers are applied to mitigate overfitting. The output of the convolutional blocks is then flattened and passed through a dense layer with 256 neurons, culminating in a Softmax layer for multiclass classification (Table 5). This design is chosen for its demonstrated ability to capture hierarchical patterns in network traffic, which is critical for accurately distinguishing between benign and malicious flows in IoT settings (Anley et al., 2024).

[6] https://www.unb.ca/cic/datasets/iotdataset-2023.html

Table 5: Proposed 1D-CNN architecture of the model.

Layer	Filters/Units	Output Shape	Activation	Parameters
Input window	–	(|input|)	–	0
Conv1D	64	(254, 64)	Sigmoid	256
MaxPool1D	–	(127, 64)	–	0
Dropout	–	(127, 64)	–	0
Conv1D	128	(125, 128)	ReLU	24 704
MaxPool1D	–	(62, 128)	–	0
Dropout	–	(62, 128)	–	0
Conv1D	256	(60, 256)	ReLU	98 560
MaxPool1D	–	(30, 256)	–	0
Dropout	–	(30, 256)	–	0
Flatten	–	(7 680)	–	0
Dense	256	(256)	ReLU	983 168
Dropout	–	(256)	–	0
Dense (output)		(|output|)	Softmax	256

For comparison purposes, we also consider Visual Geometry Group 16 (VGG16), which is a 16-layer convolutional network composed of five convolutional blocks, each comprising consecutive 3×3 convolutions followed by maximum pooling, and three fully connected layers at the top. Its deeper architecture allows for a more hierarchical feature extraction process than that performed by our CNN, exhibiting high accuracy in DDoS attack detection scenarios (Anley et al., 2024; Agostinello et al., 2023). In our evaluations, we integrate FELACS with both the shallow CNN and VGG16, with the resulting models referred to as FELACS+CNN and FELACS+VGG16, respectively, to assess how increased depth can influence the convergence speed and classification accuracy of a model under heterogeneous IoT traffic distributions.

During the model evaluation procedure, we apply an early stopping criterion at the local client level to halt the training process when the performance plateaus. In the FL framework, we vary the number of communication rounds to observe its update dynamics, and the pool of participating clients for each round is determined according to our FELACS-based selection strategy. The learning rate is carefully tuned to balance the convergence and stability of the model, and a dropout rate of 20% is enforced to enhance its robustness against noisy or incomplete IoT data. For the loss computation, we use the binary cross-entropy loss to distinguish between benign and DDoS traffic and employ the categorical cross-entropy loss when classifying specific DDoS attack types.

4.5. Evaluation Metrics

To assess the performance of our FL models, we consider the following metrics: *i)* accuracy, which is defined as the proportion of correctly classified instances out of the total number of instances; *ii)* precision, which is defined as the ratio of true-positive predictions to the total number of predicted positives; *iii)* recall, which is expressed as the ratio of true-positive predictions to the total number of actual positives; *iv)* the F1 score, which is computed as the harmonic mean of precision and recall; and *v)* the training time, which is measured as the total time required to complete the training rounds, including the communication and computation times.

We define the total training time T_{total} as the sum of the per-round computation and communication latencies over all R federated rounds:

$$T_{\text{total}} = \sum_{r=1}^{R} \left[T_{\text{dl}} + \max_{i \in \mathcal{S}} \left(T_{\text{comp}}^{(i)} + T_{\text{ul}}^{(i)} \right) + T_{\text{agg}} \right] \quad . \tag{13}$$

In each federated round, client i performs local computations on its own data in time $T_{\text{comp}}^{(i)} \left(E \times \frac{N_i}{\text{batch size}} \times t_{\text{batch}}^{(i)} \right)$. The server first incurs a download delay of T_{dl} when the global model is sent to clients (server(client). After the local training process, client i uploads its update in time $T_{\text{ul}}^{(i)}$ (client(server). Since the server waits for the slowest client, the combined communication and computation cost in that round is $\max_{i \in \mathcal{S}} \left(T_{\text{comp}}^{(i)} + T_{\text{ul}}^{(i)} \right) + T_{\text{dl}}$. Once all the updates arrive, the server aggregates them in time T_{agg}.

5. Results and Discussion

This section includes an evaluation of the proposed adaptive client selection approach for conducting FL to detect DDoS attacks within IoT environments. We first assess the performance of the model in client-side evaluations. Next, we conduct experiments within FL settings, comparing the results with those of the centralized model. We apply the proposed client selection algorithm and compare its performance with that of the state-of-the-art methods, analyzing the accuracies of the tested models under both IID and non-IID data settings. Finally, we evaluate the communication overhead, latency, sensitivity, and convergence time of the proposed model throughout the communication rounds.

5.1. Classification Accuracy Achieved on the Client Side

In this experiment, we first establish a local training baseline by measuring the detection accuracies of the CNN architecture trained independently and in a centralized manner on the different partitions of the dataset, each of which resembles the data captured by a separate and heterogeneous IoT client. For the CIC-IDS2018 dataset, the evaluation is conducted across five distributed clients. Table 6 presents the corresponding results. The table shows that the model demonstrates robust performance in terms of detecting various IoT attacks across multiple clients, achieving accuracy, precision, recall, and F1 score values of 100.00% for benign traffic across all clients in the CIC-IDS2018 dataset. For the DDoS and other DoS attack types, the model maintains high precision and recall levels, with an average precision of 99.28% across the different attack types. However, the SlowHTTPTest attack detection results show a minimum recall of 44.57% because these attacks have low cardinality. Overall, the average accuracy of 99.28% achieved by the model across diverse clients and attack types underscores its effectiveness for detecting DDoS attacks in IoT environments. In client-side evaluations, it achieves an average accuracy of 99.28%, a precision of 98.05%, a recall of 97.86%, and an F1 score of 96.54%.

For the CIC-DDoS2019 dataset, we report the per-class performance achieved across 12 DDoS attack types and benign traffic. As shown in Table 7, our client-side evaluation yields an overall accuracy of 95.85%. Benign traffic is classified with 98.10% accuracy, whereas the average accuracy attained for the DDoS attack classes is 95.60%. Across those attack categories, the model attains a mean precision level of 95.47%, a mean recall of 95.40%, and a mean F1 score of 95.50%.

For the BoT-IoT dataset, the results in Table 8 describe the classification performance metrics achieved in the IoT DDoS attack detection scenarios. The results demonstrate that most metrics reach values that are close to or equal to 100%. In particular, the models achieve perfect or almost perfect scores for the *Benign* and *Reconn* classes across all the metrics and clients, indicating their robustness and reliability in terms of detecting these types of traffic. The *DDoS* and *Theft* classes also exhibit high accuracy levels, consistently yielding scores $> 99.00\%$ for most metrics, underscoring the effectiveness of the proposed model in identifying these attacks. Even the *Scan* class–which is often considered more challenging–provides excellent performance, with a classification accuracy of approximately 98%. Overall, the client-side evaluation of the model demonstrates an average accuracy of 99.20%, a precision of 99.70%, a recall of 99.85%, and an F1 score of 99.58%.

Table 6: Client-side evaluation results obtained on the CIC-IDS2018 dataset.

Class	Metric	Client accuracy [%] IoT1	IoT2	IoT3	IoT4	IoT5
Benign	Precision	100.00	100.00	100.00	100.00	100.00
	Recall	100.00	100.00	100.00	100.00	100.00
	F1 score	100.00	100.00	100.00	100.00	100.00
DDOS attack–HOIC	Precision	97.12	97.23	97.14	97.18	97.34
	Recall	97.59	97.36	98.23	99.49	98.16
	F1 score	97.86	97.66	97.09	98.12	98.54
DDOS attack–LOIC-UDP	Precision	78.03	76.38	77.98	74.86	74.22
	Recall	87.20	89.10	84.18	85.00	87.00
	F1 score	83.34	84.48	84.08	84.80	84.00
DDOS attack–LOIC-HTTP	Precision	92.26	92.22	94.86	94.59	94.93
	Recall	98.21	99.87	98.49	98.99	98.62
	F1 score	95.12	95.10	96.91	96.87	96.93
DoS attack–GoldenEye	Precision	95.66	95.09	98.80	98.81	98.02
	Recall	98.21	99.02	99.87	99.49	99.99
	F1 score	96.04	97.64	99.13	99.81	99.64
DoS attack–Hulk	Precision	97.07	98.42	98.08	98.76	98.74
	Recall	99.27	98.82	99.41	98.51	99.36
	F1 score	98.19	99.85	99.30	99.87	99.15
DoS attack–SlowHTTPTest	Precision	77.11	67.48	68.82	78.37	79.02
	Recall	44.57	44.46	44.41	44.87	44.05
	F1 score	61.13	61.81	61.64	61.51	61.64
DoS attack–Slowloris	Precision	82.11	89.48	79.62	84.37	82.02
	Recall	77.40	75.62	76.49	76.98	76.50
	F1 score	88.07	85.42	86.08	86.64	86.32
Avg. accuracy				99.28		

For the CIC-IoT2023 dataset, which includes eight DDoS attack types and benign traffic, the client-side evaluation yields an overall accuracy of 96.95% (Table 9). Benign samples are detected with 97.50% accuracy on average, whereas the average accuracy attained for the IoT attack types is 96.80%. Across these attack categories, the mean precision is 96.70%, the mean recall is 96.60%, and the mean F1 score is 96.65%.

5.2. Comparison with Centralized with Federated Learning Models

In this experiment, we compare the accuracy of the proposed FELACS approach with those of centralized standalone models and FL-based aggregation strategies, including random selection (Ruan et al., 2021; Ginanjar et al., 2025), FedAvg (McMahan et al., 2017), FedProx (Li et al., 2020b), and FedMCCS (AbdulRahman et al., 2020).

Table 7: Client-side evaluation results obtained on the CIC-DDoS2019 dataset.

Class	**Metric**	**Client accuracy [%]**				
		IoT1	**IoT2**	**IoT3**	**IoT4**	**IoT5**
Benign	Precision	98.98	99.08	98.85	98.75	98.72
	Recall	99.60	99.55	99.34	99.73	99.86
	F1 score	99.29	99.31	99.09	99.24	99.29
DNS	Precision	71.75	80.00	75.95	73.41	74.09
	Recall	67.65	76.33	75.16	78.87	79.67
	F1 score	69.86	78.93	73.85	74.81	76.98
SNMP	Precision	83.64	69.20	65.72	78.94	73.88
	Recall	86.28	74.71	63.55	75.14	74.61
	F1 score	85.70	77.85	62.59	76.05	74.15
LDAP	Precision	51.75	65.37	73.15	69.98	67.27
	Recall	52.28	71.28	86.41	90.25	65.53
	F1 score	56.83	68.03	79.34	81.16	66.26
MSSQL	Precision	89.22	85.16	93.15	92.69	88.67
	Recall	94.38	93.19	93.89	95.57	93.89
	F1 score	95.69	89.00	93.48	93.82	90.86
NTP	Precision	99.51	99.63	99.63	99.54	99.32
	Recall	99.05	99.44	99.35	99.06	99.34
	F1 score	99.28	99.54	99.49	99.37	99.33
NetBIOS	Precision	87.14	81.48	78.57	76.84	82.75
	Recall	70.94	82.30	73.18	72.72	80.67
	F1 score	81.36	85.69	75.78	74.78	81.54
SSDP	Precision	76.45	71.39	83.15	76.37	73.41
	Recall	82.28	73.23	93.89	73.72	71.29
	F1 score	79.52	72.27	88.20	74.82	72.37
Syn	Precision	98.50	97.25	97.63	98.62	95.24
	Recall	96.12	95.83	98.73	98.72	99.17
	F1 score	97.73	96.79	98.18	98.44	97.17
TFTP	Precision	100.00	99.69	99.98	100.00	99.93
	Recall	99.32	99.34	99.37	99.20	99.22
	F1 score	99.66	99.65	99.67	99.60	99.57
UDP	Precision	91.21	95.79	97.57	97.43	97.82
	Recall	97.34	93.87	95.23	95.83	95.82
	F1 score	94.05	94.65	96.40	96.62	96.89
WebDDoS	Precision	77.14	73.66	76.85	82.96	76.68
	Recall	69.43	71.28	74.18	95.57	74.34
	F1 score	73.63	72.28	75.27	88.82	75.57
Avg. accuracy				95.85		

Table 8: Client-side evaluation results obtained on the BoT-IoT dataset.

Class	Metric	Client accuracy [%]				
		IoT1	**IoT2**	**IoT3**	**IoT4**	**IoT5**
Benign	Accuracy	100.00	100.00	100.00	100.00	100.00
	Precision	100.00	100.00	100.00	100.00	100.00
	Recall	100.00	100.00	100.00	100.00	100.00
	F-measure	100.00	100.00	100.00	100.00	100.00
DDoS	Accuracy	99.01	99.28	99.28	99.23	99.25
	Precision	99.31	99.49	99.63	98.77	99.10
	Recall	99.77	99.60	99.88	99.87	99.86
	F-measure	99.73	99.72	99.82	99.84	99.86
Reconn	Accuracy	100.00	100.00	100.00	100.00	100.00
	Precision	99.73	98.59	99.70	99.60	99.29
	Recall	100.00	100.00	100.00	100.00	100.00
	F-measure	100.00	100.00	100.00	100.00	100.00
Theft	Accuracy	99.00	99.00	99.00	99.00	99.00
	Precision	99.03	99.10	99.11	99.11	99.37
	Recall	100.00	100.00	100.00	100.00	100.00
	F-measure	99.15	99.22	99.44	99.55	99.75
Scan	Accuracy	97.00	98.00	98.00	98.00	98.00
	Precision	98.00	98.08	98.15	98.08	98.15
	Recall	98.83	99.93	99.93	99.93	99.93
	F-measure	97.38	99.00	99.03	99.00	99.03
Avg. accuracy				99.20		

For the CIC-IDS2018 dataset, the results show that the proposed FELACS method achieves $\approx 95.0\%$ accuracy, as shown in Figure 1a, outperforming the other methods and even the centralized version. Similarly, for the CIC-DDoS2019 dataset, the results are shown in Figure 1b. Among the federated strategies, FedMCCS and FELACS achieve the highest accuracies at $\approx 92.0\%$, also outperforming the centralized baseline. On the BoT-IoT dataset, as illustrated in Figure 1c, FELACS also consistently outperforms the compared approaches, with $\approx 92.0\%$ accuracy. Considering the CIC-IoT2023 dataset, Figure 1d shows that, on a more complex split, CL-sin reaches a top accuracy of approximately $\approx 99.0\%$; FELACS has the best performance among the FL-based approaches, with an average accuracy of $\approx 98.0\%$. Although the centralized model retains a small lead in this scenario, FELACS matches or exceeds all other federated baselines.

The results obtained on the considered datasets confirm that our methodology effectively adapts to changing traffic conditions, maintaining high accuracy levels under all conditions, and is superior to other FL-based approaches, in some cases surpassing the centralized version. This finding underscores two key points. First, FL can approach or surpass the performance of centralized methods when

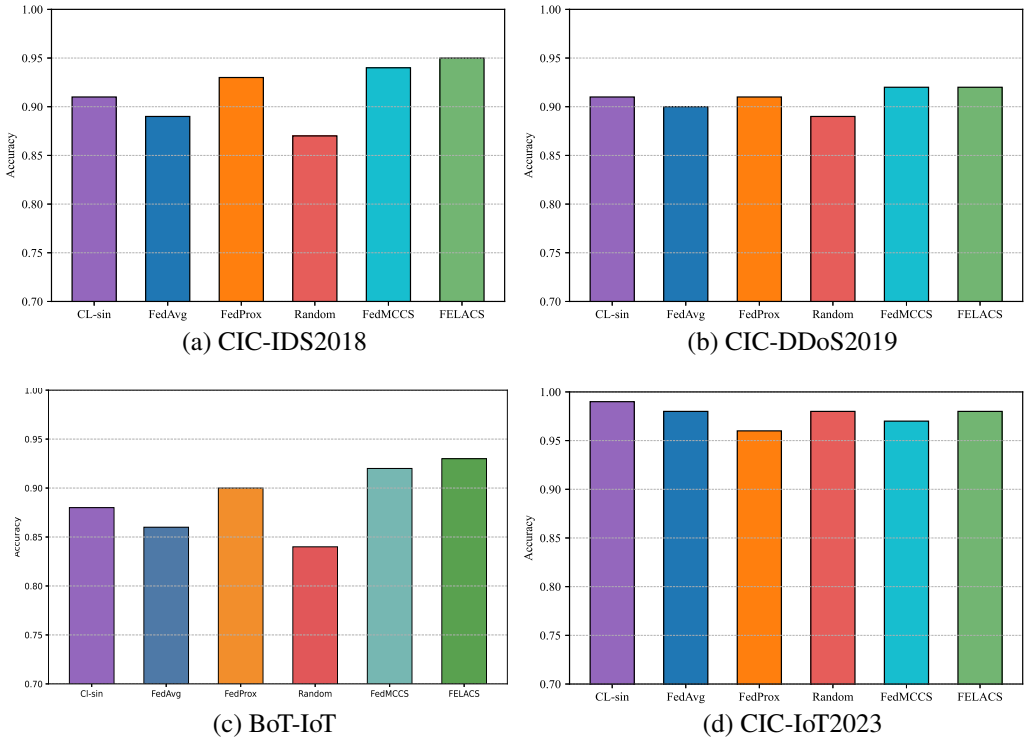
Table 9: Client-side evaluation results obtained on the CIC-IoT2023 dataset.

Class	Metric	Client accuracy [%]				
		IoT1	**IoT2**	**IoT3**	**IoT4**	**IoT5**
Benign	Precision	92.43	95.48	93.32	94.57	94.06
	Recall	95.75	97.24	96.22	97.87	98.23
	F1 score	93.08	93.20	93.20	93.20	93.20
BruteForce	Precision	97.57	95.89	96.94	94.87	94.87
	Recall	91.83	95.68	93.72	95.68	95.68
	F1 score	95.32	95.40	95.34	94.64	94.96
DDoS	Precision	99.67	97.64	98.39	98.74	99.82
	Recall	99.94	98.30	98.92	97.34	97.07
	F1 score	99.55	99.54	98.02	98.90	97.48
DoS	Precision	97.56	95.47	89.43	94.23	95.92
	Recall	87.57	89.66	87.76	93.64	96.04
	F1 score	94.70	93.68	89.86	92.72	95.99
Mirai	Precision	100.00	100.00	100.00	100.00	100.00
	Recall	99.45	99.78	99.00	99.92	99.32
	F1 score	99.78	99.89	99.58	99.98	99.74
Recon	Precision	98.95	98.00	97.62	96.89	97.70
	Recall	100.00	100.00	100.00	100.00	99.74
	F1 score	99.08	99.01	98.76	98.46	98.68
Spoofing	Precision	83.79	83.80	83.00	83.00	83.00
	Recall	97.88	97.00	97.00	97.00	96.00
	F1 score	89.90	89.34	89.20	89.32	89.67
Web	Precision	74.64	89.00	78.27	76.66	74.64
	Recall	79.69	72.62	78.68	80.70	81.71
	F1 score	76.66	79.69	78.68	68.68	77.34
Avg. accuracy				98.86		

heterogeneity is properly managed. Second, by dynamically prioritizing clients with the most informative data under resource constraints, FELACS injects higher-quality updates into each aggregation step, increasing the accuracy of the global model without sacrificing privacy or incurring centralization overhead. As a consequence, we believe that the proposed methodology is able to adapt to modern IoT networks under a variety of network conditions.

5.3. Impact of the Number of Clients on Global Accuracy

In this section, we evaluate the accuracy achieved by the proposed method when varying the number of clients. Table 10 shows the average accuracy attained by the global model over 50 communication rounds as we vary the number of participating clients (10, 25, and 50) on four benchmarks. Across all the datasets, both FELACS+CNN and FELACS+VGG16 consistently outperform FedAvg, FedProx,



(a) CIC-IDS2018 (b) CIC-DDoS2019

(c) BoT-IoT (d) CIC-IoT2023

Figure 1: Accuracies achieved by various aggregation strategies and centralized standalone models on the considered datasets. It is possible to observe how the proposed FELACS method achieves the highest accuracy among the tested FL-based techniques, in some cases surpassing the centralized version *(CL-sin)*.

FedMCCS, and random selection, with the VGG16 variant achieving the highest accuracy in every setting. In particular, on CIC-IDS2018, FELACS+CNN attains 98.20%, 99.80%, and 99.40% accuracy values with 10, 25, and 50 clients, respectively, whereas FELACS+VGG16 further improves the performance to 98.50%, 99.90%, and 99.60%–up to 1.40% higher than the metrics of the best baseline (FedProx). On CIC-DDoS2019, FELACS+CNN achieves 93.00%, 94.00%, and 94.50% accuracy values, whereas FELACS+VGG16 reaches 93.50%, 94.50%, and 95.00% accuracy values, exceeding those of FedProx by up to 1.40%. On BoT-IoT, the accuracies increase from 97.10%, 97.90%, and 98.30% for FELACS+CNN to 97.40%, 98.10%, and 98.50% with FELACS+VGG16, surpassing the results of FedProx by as much as 1.40%. Finally, on CIC-IoT2023, FELACS+CNN delivers accuracies of 95.20%, 95.80%, and 96.00%, and FELACS+VGG16 reaches values 95.50%, 96.10%, and 96.30%, outperforming all the baselines by $0.60 - -1.00\%$.

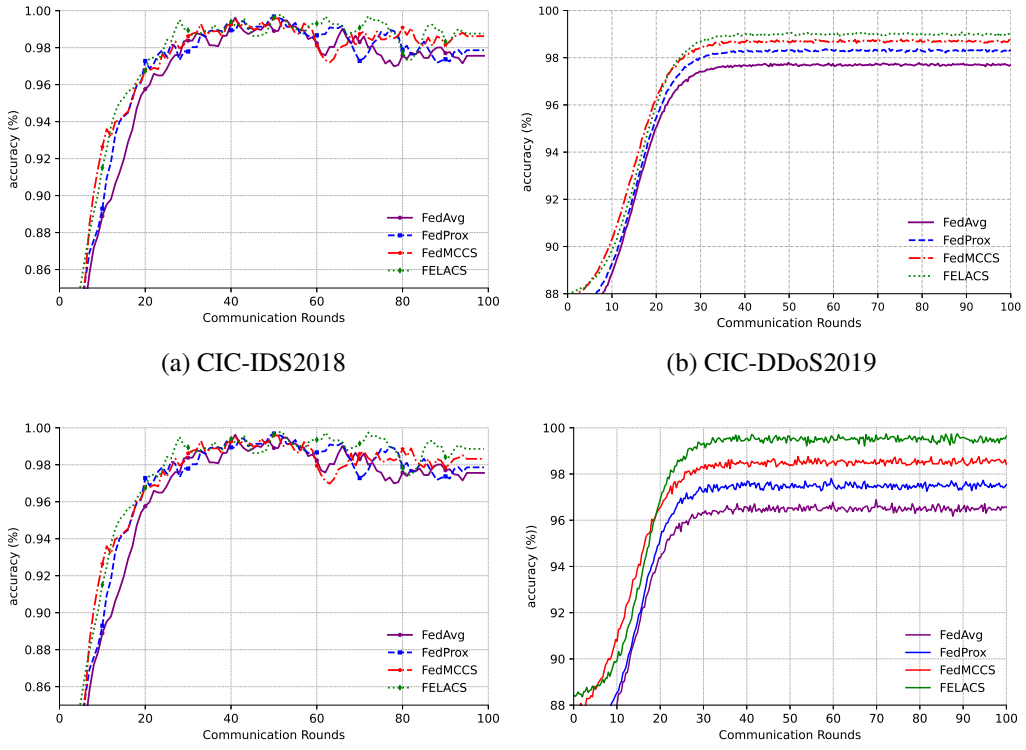
Table 10: Impact of the number of clients on the accuracy of the global model.

Algorithm	CIC-IDS2018			CIC-DDoS2019			BoT-IoT			CIC-IoT2023		
	10	25	50	10	25	50	10	25	50	10	25	50
FedAvg	97.00	97.80	98.20	91.80	92.40	93.00	96.80	97.60	98.00	94.20	94.80	95.10
FedProx	97.10	97.90	98.30	92.30	93.00	93.60	96.90	97.70	98.10	94.50	95.10	95.40
Random	96.80	97.50	97.90	91.20	91.80	92.40	96.60	97.30	97.80	93.80	94.30	94.60
FedMCCS	93.92	94.06	94.26	90.10	90.70	91.20	93.12	93.86	94.88	92.50	93.00	93.40
FELACS+CNN	98.20	99.80	99.40	93.00	94.00	94.50	97.10	97.90	98.30	95.20	95.80	96.00
FELACS+VGG16	98.50	99.90	99.60	93.50	94.50	95.00	97.40	98.10	98.50	95.50	96.10	96.30

The results produced with different numbers of clients confirm two key findings. First, enlarging the client population improves the performance of the global model by increasing the diversity of the samples. Second, integrating a deeper VGG16 backbone with FELACS yields additional accuracy gains over the CNN baseline, demonstrating robust and high-quality aggregated updates even under severe non-IID conditions. As the client pool further expands, the gains decrease: selecting up to 100 clients yields only marginal accuracy improvements over the results obtained using fewer participants. Although a larger cohort can capture more diverse data, this advantage plateaus beyond a certain point. Moreover, integrating VGG16, which has a substantially larger parameter count and a deeper architecture, increases the computational and communication burdens imposed on resource-constrained edge devices and IoT nodes. In our experiments, selecting approximately 25 clients strikes the best balance between global accuracy and system efficiency, providing sufficient data diversity to avoid both underfitting and overfitting without incurring the excessive training time, memory usage, and bandwidth demands that arise when many devices must process or transmit updates for a heavyweight model such as VGG16. This underscores the importance of a strategic client selection strategy that jointly considers model complexity and limited edge resources.

5.4. Classification Results

The FL model classification results obtained on the CIC-IDS2018 dataset when our FELACS client selection strategy is applied are presented in Figure 2a. The performance metrics produced across 100 communication rounds demonstrate that the proposed FELACS approach outperforms the FedAvg, FedProx, FedMCCS, and Random methods, achieving higher accuracy throughout all the training rounds. Similarly, Figure 2b presents the accuracy curves obtained for CIC-DDoS2019. The figure shows that FELACS consistently outperforms the baselines, exceeding



(a) CIC-IDS2018

(b) CIC-DDoS2019

(c) BoT-IoT

(d) CIC-IoT2023

Figure 2: Detection accuracies attained over 100 communication rounds on the considered datasets with different client selection strategies. The proposed FELACS approach outperforms the other FL-based methods.

98.8% accuracy and maintaining stable performance. Figure 2c illustrates the accuracy achieved on the BoT-IoT dataset. The accuracy curves converge toward values near 100%, with the FELACS method reaching the highest final accuracy. On the CIC-IoT2023 dataset (Figure 2d), FELACS achieves over 98% accuracy by round 20 while also attaining the highest steady-state accuracy of approximately 99.5%.

5.5. Communication Overhead

In this section, we evaluate the number of communication rounds needed to achieve convergence. In particular, Table 11 illustrates the numbers of rounds needed to reach 90% and 95% accuracy under both IID and non-IID splits for all

Table 11: Numbers of communication rounds required to reach 95% and 90% accuracy (IID / non-IID) on the considered datasets.

Approach	CIC-IDS2018		CIC-DDoS2019		BoT-IoT		CIC-IoT2023	
	ToA@95	ToA@90	ToA@95	ToA@90	ToA@95	ToA@90	ToA@95	ToA@90
FedAvg	38/43	32/36	50/58	42/50	46/54	40/46	44/52	36/44
FedProx	36/40	30/34	48/54	40/46	44/50	38/43	42/48	35/40
Random	50/58	42/48	65/75	56/64	60/68	52/56	58/68	50/56
FedMCCS	34/38	28/32	46/52	38/44	42/48	36/40	40/46	34/38
FELACS+CNN	30/35	25/28	40/46	32/38	38/45	32/38	36/42	30/34
FELACS+VGG16	28/32	23/26	38/44	30/36	36/43	30/36	34/40	28/32

four datasets. As expected, heterogeneous distributions incur higher communication costs: each method requires several more rounds in the non-IID setting than in the IID setting. Traditional schemes such as FedAvg and FedProx generally converge in the mid-30s to mid-40s (rounds) when aiming for 90% accuracy and the high-30s to mid-50s for 95%. Random selection is the least efficient approach, often demanding upward of 60 rounds to reach the same thresholds under skewed data. In contrast, the FELACS approaches substantially reduce this overhead, achieving both targets in the low-30s to low-40s even when the data are highly imbalanced. These results demonstrate that by prioritizing clients whose updates carry maximal utility while respecting resource constraints, FL can preserve the fidelity of a model with far fewer communication exchanges, which is an essential property for real-time IoT deployments.

5.6. Model Convergence Time Analysis

In this section, we evaluate the convergence times required by the considered approaches. Table 12 provides a comparative analysis of the temporal efficiency levels of various FL methods across 100 communication rounds on the examined datasets. The results indicate that the proposed FELACS method has better performance than the existing approaches do, achieving lower average communication times and requiring fewer rounds for convergence. These findings show the potential of FELACS to reduce the imposed communication overhead and enhance the overall efficiency of FL in IoT environments, positioning it as a promising approach for real-world applications. However, when applying FELACS to the deeper VGG16 architecture, the increased parameter count and layer depth of this network introduce significant communication and computational overhead for resource-constrained IoT and edge devices. In our experiments, FELACS+VGG16 yields the highest accuracy but requires substantially more bandwidth per round

Table 12: Per-round time (minutes) required by different FL client selection methods for four datasets.

Method	CIC-IDS2018		CIC-DDoS2019		BoT-IoT		CIC-IoT2023	
	Mean	Std	Mean	Std	Mean	Std	Mean	Std
FedAvg	5.45	0.32	6.00	0.35	5.12	0.28	5.00	0.30
FedProx	5.20	0.25	5.80	0.30	4.98	0.22	4.80	0.24
Random	6.30	0.50	7.00	0.55	6.05	0.45	6.10	0.48
FedMCCS	9.20	0.87	9.80	1.00	8.76	0.79	9.00	0.85
FELACS+CNN	4.82	0.20	5.30	0.22	4.65	0.18	4.50	0.16
FELACS+VGG16	9.50	0.86	11.80	1.35	8.20	0.56	9.00	0.98

than does FELACS+CNN. To mitigate this overhead in real-world deployments, it may be possible to use model compression techniques, such as 8-bit quantization and pruning, to reduce payload sizes and computational loads, thereby preserving the convergence gains provided by FELACS+VGG16 while satisfying IoT/edge resource constraints.

We also theoretically analyze the convergence of the proposed approach by considering the demonstration presented in (Li and Lyu, 2023) , which states that sequential FL converges on heterogeneous data at an $O(1/T)$ rate for strongly convex objectives, provided that each client has a positive probability of being selected in each round. Moreover, the works of (Cho et al., 2020; Wang and Ji, 2022) demonstrated the convergence of client selection schemes with bounded gradient estimation. In this context, the multicriteria optimization scheme of FELACS ensures that every client retains a strictly positive selection probability and keeps the gradient estimation variance bounded. Moreover, the federated multiobjective learning framework (Yang et al., 2023; Vardhan et al., 2025) extends identical guarantees to multigradient aggregation scenarios. FELACS satisfies all core assumptions of these convergence theories and therefore exhibits the predicted convergence behavior in our IoT DDoS detection experiments.

5.7. Sensitivity Analysis

We assess the robustness of the proposed approach by varying each weight (α, β, γ, δ) by $\pm 20\%$ around its default value and then measure the detection accuracy and F1 score values achieved on the CIC-IDS2018, CIC-DDoS2019, BoT-IoT, and CIC-IoT2023 datasets. Across all the perturbations and datasets, the maximum accuracy and F1 score deviations remain within $\pm 5\%$. These findings confirm that the importance scoring function of our proposed FELACS approach tolerates moderate hyperparameter shifts without significant performance losses, reducing the

burden of fine-tuning in real-world federated IoT deployments. The demonstrated robustness suggests that practitioners can confidently apply our method under various device conditions and network characteristics with minimal recalibration.

6. Conclusion

In this paper, we propose FELACS, an adaptive client selection approach for detecting IoT DDoS attacks in FL scenarios. Our approach considers multiobjective optimization, considering both the diversity and importance of client data, as well as the real-time performance of client devices. In contrast with the existing methods in the literature, our approach integrates a multiobjective optimization framework that evaluates key factors, including data entropy, variance, power consumption, and latency, for each client. This comprehensive approach ensures an effective and balanced client selection process during FL, mitigating client drift by favoring the clients with the most representative data and sufficient computational resources.

We validate the developed method on four publicly available datasets and demonstrate that our algorithm achieves better classification accuracy than that of a nonfederated model and does so in fewer communication rounds. Moreover, we evaluate the performance of the method in comparison with that of FedAvg, FedProx, FedMCCS, and random client selection, with our approach achieving better accuracy and faster convergence than those of the existing methods in the literature. Future works will focus on developing robust, scalable, and interpretable models using federated transfer learning to enhance the DDoS attack detection process in IoT environments.

Acknowledgments

This work was supported in part by the EC under the EdgeAI (101097300) and GLACIATION (101070141) projects and by the SERICS project (PE00000014) under the MUR NRRP funded by the EU–NGEU. We also thank the NVIDIA Corporation for the donated GPU. Project EdgeAI is supported by the Chips Joint Undertaking and its members, including top-up funding by Austria, Belgium, France, Greece, Italy, Latvia, Netherlands, and Norway under grant agreement no. 101097300. However, the views and opinions expressed are those of the authors only and do not necessarily reflect those of the European Union, the Chips Joint Undertaking, or the Italian MUR. Neither the European Union nor the granting authority or the Italian MUR can be held responsible for them.

References

AbdulRahman, S., Tout, H., Mourad, A., Talhi, C., 2020. FedMCCS: Multicriteria client selection model for optimal IoT federated learning. IEEE Internet of Things Journal 8, 4723–4735.

Agostinello, D., Genovese, A., Piuri, V., 2023. Anomaly-based intrusion detection system for DDoS attack with deep learning techniques, in: Proc. of SECRYPT, pp. 267–275.

Aktar, S., Nur, A.Y., 2023. Towards DDoS attack detection using deep learning approach. Computers & Security 129, 103251.

Ami, D.B., Cohen, K., Zhao, Q., 2025. Client selection for generalization in accelerated federated learning: A multi-armed bandit approach. IEEE Access 13, 33697–33713.

Anand, R., Aggarwal, D., Kumar, V., 2017. A comparative analysis of optimization solvers. Journal of Statistics and Management Systems 20, 623–635.

Anley, M.B., Genovese, A., Agostinello, D., Piuri, V., 2024. Robust DDoS attack detection with adaptive transfer learning. Computers & Security 144, 103962.

Anley, M.B., Genovese, A., Piuri, V., 2025. TransferEdge: Transfer learning approach to detect evolving DDoS threats in Edge-IIoT, in: Proc. of RTSI, pp. 1–6.

Beutel, D.J., Topal, T., Mathur, A., Qiu, X., Fernandez-Marques, J., Gao, Y., Sani, L., Li, K.H., Parcollet, T., de Gusmão, P.P.B., et al., 2020. Flower: A friendly federated learning research framework, in: arXiv:2007.14390, pp. 1–15.

Bonawitz, K.A., Eichner, H., Grieskamp, W., Huba, D., Ingerman, A., Ivanov, V., Kiddon, C., Konečný, J., Mazzocchi, S., McMahan, B., Overveldt, T.V., Petrou, D., Ramage, D., Roselander, J., 2019. Towards federated learning at scale: System design, in: Proc. of SysML, pp. 1–15.

Cho, Y.J., Wang, J., Joshi, G., 2020. Client selection in federated learning: Convergence analysis and power-of-choice selection strategies, in: arXiv:2010.01243, pp. 1–22.

Elfaki, M.A., Alshahrani, H.M., Mahmood, K., Alabdan, R., Alymani, M., Alshahrani, H., Motwakel, A., Alneil, A.A., 2023. Metaheuristics algorithm-based minimization of communication costs in federated learning. IEEE Access 11, 81310–81317.

Feng, J., Liu, L., Pei, Q., Li, K., 2021. Min-max cost optimization for efficient hierarchical federated learning in wireless edge networks. IEEE Transactions on Parallel and Distributed Systems 33, 2687–2700.

Fotse, Y.S.N., Tchendji, V.K., Velempini, M., 2024. Federated learning based DDoS attacks detection in large scale software-defined network. IEEE Trans. on Computers 74, 101–115.

Fu, L., Zhang, H., Gao, G., Zhang, M., Liu, X., 2023. Client selection in federated learning: Principles, challenges, and opportunities. IEEE Internet of Things Journal 10, 21811–21819.

Ginanjar, A., Li, X., Singh, P., Hua, W., 2025. Random client selection on contrastive federated learning for tabular data, in: arXiv:2505.10759, pp. 1–5.

Gümüşbaş, D., Yıldırım, T., Genovese, A., Scotti, F., 2021. A comprehensive survey of databases and deep learning methods for cybersecurity and intrusion detection systems. IEEE Systems Journal 15, 1717–1731.

Hsu, T.M.H., Qi, H., Brown, M., 2019. Measuring the effects of non-identical data distribution for federated visual classification, in: arXiv:1909.06335, pp. 1–5.

Hu, Z., Shaloudegi, K., Zhang, G., Yu, Y., 2022. Federated learning meets multi-objective optimization. IEEE Trans. on Network Science and Engineering 9, 2039–2051.

Kairouz, P., McMahan, H.B., Avent, B., Bellet, A., Bennis, M., Bhagoji, A.N., Bonawitz, K., Charles, Z., Cormode, G., Cummings, R., et al., 2021. Advances and open problems in federated learning. Foundations and trends® in machine learning 14, 1–210.

Koroniotis, N., Moustafa, N., Sitnikova, E., Turnbull, B., 2019. Towards the development of realistic botnet dataset in the internet of things for network forensic analytics: BoT-IoT dataset. Future Generation Computer Systems 100, 779–796.

Li, B., Wu, Y., Song, J., Lu, R., Li, T., Zhao, L., 2020a. DeepFed: Federated deep learning for intrusion detection in industrial cyber–physical systems. IEEE Transactions on Industrial Informatics 17, 5615–5624.

Li, J., Chen, T., Teng, S., 2024. A comprehensive survey on client selection strategies in federated learning. Computer Networks 251, 1–15.

Li, J., Zhang, Z., Li, Y., Guo, X., Li, H., 2021. FIDS: Detecting DDoS through federated learning based method, in: Proc. of TrustCom, pp. 856–862.

Li, T., Sahu, A.K., Zaheer, M., Sanjabi, M., Talwalkar, A., Smith, V., 2020b. Federated optimization in heterogeneous networks. Proc. of Machine Learning and Systems 2, 429–450.

Li, Y., Lyu, X., 2023. Convergence analysis of sequential federated learning on heterogeneous data, in: Proc. of NIPS, pp. 56700–56755.

Lim, W.Y.B., Luong, N.C., Hoang, D.T., Jiao, Y., Liang, Y.C., Yang, Q., Niyato, D., Miao, C., 2020. Federated learning in mobile edge networks: A comprehensive survey. IEEE communications surveys & tutorials 22, 2031–2063.

Ma, X., Zhu, J., Lin, Z., Chen, S., Qin, Y., 2022. A state-of-the-art survey on solving non-IID data in federated learning. Future Generation Computer Systems 135, 244–258.

Maciel, F., De Souza, A.M., Bittencourt, L.F., Villas, L.A., 2023. Resource aware client selection for federated learning in IoT scenarios, in: Proc. of DCOSS-IoT, pp. 1–8.

Mazid, A., Kirmani, S., Manaullah, Yadav, M., 2025. FL-IDPP: A federated learning based intrusion detection approach with privacy preservation. Trans. on Emerging Telecommunications Technologies 36, e70039.

McMahan, B., Moore, E., Ramage, D., Hampson, S., y Arcas, B.A., 2017. Communication-efficient learning of deep networks from decentralized data, in: Artificial Intelligence and Statistics, pp. 1273–1282.

Neto, E.C.P., Dadkhah, S., Ferreira, R., Zohourian, A., Lu, R., Ghorbani, A.A., 2023. CICIoT2023: A real-time dataset and benchmark for large-scale attacks in IoT environment.

Nguyen, Q.H., Hore, S., Shah, A., Le, T., Bastian, N.D., 2025. FedNIDS: A federated learning framework for packet-based network intrusion detection system. Digital Threats: Research and Practice 6, 1–23.

Nishio, T., Yonetani, R., 2019. Client selection for federated learning with heterogeneous resources in mobile edge, in: Proc. of ICC, pp. 1–7.

Olanrewaju-George, B., Pranggono, B., 2025. Federated learning-based intrusion detection system for the internet of things using unsupervised and supervised deep learning models. Cyber Security and Applications 3, 100068.

Pakmehr, A., Aßmuth, A., Taheri, N., Ghaffari, A., 2024. Ddos attack detection techniques in IoT networks: a survey. Cluster Computing 27, 14637–14668.

Puchinger, J., Raidl, G.R., Pferschy, U., 2010. The multidimensional knapsack problem: Structure and algorithms. INFORMS Journal on Computing 22, 250–265.

Rahmati, M., 2025. Federated learning-driven cybersecurity framework for IoT networks with privacy-preserving and real-time threat detection capabilities, in: arXiv:2502.10599, pp. 1–16.

Roy, S., Li, J., Bai, Y., 2023. Federated learning-based intrusion detection system for IoT environments with locally adapted model, in: Proc. of CSCloud/EdgeCom, pp. 203–209.

Ruan, Y., Zhang, X., Liang, S.C., Joe-Wong, C., 2021. Towards flexible device participation in federated learning, in: Proc. of ICAIS, PMLR. pp. 3403–3411.

Ruzafa-Alcázar, P., Fernández-Saura, P., Mármol-Campos, E., González-Vidal, A., Hernández-Ramos, J.L., Bernal-Bernabe, J., Skarmeta, A.F., 2021. Intrusion detection based on privacy-preserving federated learning for the industrial IoT. IEEE Trans. on Industrial Informatics 19, 1145–1154.

Seo, S., Lee, J., Ko, H., Pack, S., 2022. Performance-aware client and quantization level selection algorithm for fast federated learning, in: Proc. of WCNC, pp. 1892–1897.

Sharafaldin, I., Lashkari, A.H., Ghorbani, A.A., 2018. Toward generating a new intrusion detection dataset and intrusion traffic characterization. ICISSp 1, 108–116.

Sharafaldin, I., Lashkari, A.H., Hakak, S., Ghorbani, A.A., 2019. Developing realistic distributed denial of service (DDoS) attack dataset and taxonomy, in: Proc. of ICCST, pp. 1–8.

Sun, S., Sharma, P., Nwodo, K., Stavrou, A., Wang, H., 2024. FedMADE: Robust federated learning for intrusion detection in IoT networks using a dynamic aggregation method, in: Proc. of ICIS, pp. 286–306.

Tahir, M., Mawla, T., Awaysheh, F., Alawadi, S., Gupta, M., Ali, M.I., 2025. SecureFedPROM: A zero-trust federated learning approach with multi-criteria client selection. IEEE Journal on Selected Areas in Communications 43, 2025–2041.

Vardhan, H., Yu, X., Rosing, T., Mazumdar, A., 2025. Client selection in federated learning with data heterogeneity and network latencies, in: arXiv:2504.01921, pp. 1–23.

Wang, H., Yurochkin, M., Sun, Y., Papailiopoulos, D.S., Khazaeni, Y., 2020. Federated learning with matched averaging, in: Proc. of ICLR, pp. 1–16.

Wang, S., Ji, M., 2022. A unified analysis of federated learning with arbitrary client participation, in: Proc. of NIPS, pp. 19124–19137.

Wu, H., Tang, X., Zhang, Y.J.A., Gao, L., 2023. Incentive mechanism for federated learning with random client selection. IEEE Trans. on Network Science and Engineering 11, 1922–1933.

Yang, H., Liu, Z., Liu, J., Dong, C., Momma, M., 2023. Federated multi-objective learning. Advances in neural information processing systems 36, 39602–39625.

Zhang, J., Li, S., Wang, C., 2024. Utility aware optimal data selection for differentially private federated learning in IoV. IEEE Internet of Things Journal 11, 33326–33336.

Zhao, Y., Li, M., Lai, L., Suda, N., Civin, D., Chandra, V., 2018. Federated learning with non-IID data, in: arXiv:1806.00582, pp. 1–12.