Enhancing Aggregation Efficiency and Training Stability in Heterogeneous Federated Learning Using the Adaptive LoRA FL Framework

Lin-Huang Chang, Yu-Sheng Hsieh, and Tsung-Han Lee*

Department of Computer Science, National Taichung University of Education, Taiwan
Email: lchang@mail.ntcu.edu.tw (L.H.C.); bcs113101@gm.ntcu.edu.tw (Y.S.H.); thlee@mail.ntcu.edu.tw (T.H.L.)
*Corresponding author

***Abstract*—In recent years, Low-Rank Adaptation (LoRA) has emerged as a highly effective parameter-efficient fine-tuning technique for Transformer-based models. This paper extends the applicability of LoRA to traditional Deep Neural Networks (DNNs) by integrating it into a Federated Learning (FL) framework. The principal aim of this study is to facilitate data privacy protection while simultaneously minimizing weight transfer overhead in heterogeneous environments. To address the challenges of Non-Independent Identically Distribution (Non-IID) data, this study proposes the Adaptive LoRA FL framework, which incorporates feature mapping and category unification mechanisms to harmonize highly heterogeneous datasets. By integrating LoRA fine-tuning with feature mapping and category unification mechanisms, it effectively integrates highly heterogeneous datasets (e.g., CIC-IDS2017, Edge-IIoTset, and TON-IoT datasets). Furthermore, the performance of Adaptive LoRA FL framework is evaluated against state-of-the-art FL methods and demonstrate its generalizability across diverse application domains. Experimental results demonstrate that compared to full-weight aggregation, the Adaptive LoRA FL framework significantly reduces client communication costs while maintaining improved global accuracy. By integrating an early stopping mechanism, the training process terminates once convergence is reached, effectively minimizing redundant computation and communication overhead.**

***Keywords*—Low-Rank Adaptation (LoRA), heterogeneous federated learning, feature mapping**

I. Introduction

The paradigm of edge computing has become a pivotal paradigm in modern data processing, driven by rapid advancements in the Internet of Things (IoT) and Artificial Intelligence (AI). To enable effective model training in distributed environments, Federated Learning (FL) [1] has been extensively adopted for collaborative learning across devices and network nodes. The fundamental principle of FL is the training of local models independently on multiple clients without centralizing raw data, followed by the server-side aggregation of model parameters to enhance both privacy preservation and model performance.

In the context of IoT applications, such as smart manufacturing, Industrial Internet of Things (IIoT), and Artificial Intelligence of Things (AIoT) scenarios, deep learning clients are generally implemented on various edge devices and network nodes. Consequently, it is essential for FL clients to periodically transmit full model weights to central servers. These edge devices typically encounter constraints, such as high data heterogeneity, constrained computational resources, and limited bandwidth availability. Therefore, these limitations hinder the effective execution of model synchronization and weight transmission during training. Furthermore, in Non-Independent Identically Distribution (Non-IID) data environments, federated learning systems continue to encounter the challenge of slow global convergence.

Recent research introduces Parameter Efficient Fine-Tuning (PEFT) techniques into on-site distributed learning frameworks to mitigate these limitations. Low-Rank Adaptation (LoRA) [2] has gained attention as an efficient fine-tuning method. It injects trainable low-rank matrices into frozen weight layers, achieving comparable performance to full fine-tuning while training only a small number of additional parameters. Although LoRA is effective in Transformer-based models, its use with traditional Deep Neural Networks (DNN) is limited. Furthermore, research into LoRA's efficacy within Non-IID and heterogeneous edge environments remains nascent.

To address these challenges, this paper proposes an Adaptive LoRA FL framework specifically designed for heterogeneous federated learning environments. This novel approach integrates the LoRA technology into traditional DNN architectures, employing an adaptive mechanism that combines LoRA fine-tuning parameters with layer freezing strategies and early stopping mechanisms to balance communication efficiency and model performance in heterogeneous edge scenarios. Specifically, three public intrusion detection datasets, CIC-IDS2017 [3], Edge-IIoTset [4] and TON-IoT [5] are

Manuscript received November 3, 2025; revised November 21, 2025; accepted January 13, 2026; published April 16, 2026.

employed to emulate a realistic heterogeneous federated learning environment across multiple edge nodes. The proposed Adaptive LoRA FL framework demonstrates its effectiveness by significantly reducing both communication and training overheads while simultaneously achieving high global accuracy.

The experimental results indicate that the proposed Adaptive LoRA FL framework achieved a global accuracy of 98.58%, while the traditional Weight FL baseline attained 98.61% after 300 global rounds. In terms of resource consumption, the average communication cost decreased from 278.49 MB in Weight FL to 39.26 MB in Adaptive LoRA FL, representing a reduction of 85.9%. Additionally, the average training time for the Adaptive LoRA FL framework is 3927.8 s, which is 61.68% lower than the 10,248.8 s required by Weight FL. These empirical findings suggest that the proposed Adaptive LoRA FL framework yields lower communication and training requirements while maintaining a performance level closely aligned with full-weight federated learning aggregation methods.

The rest of this paper is organized as follows: Section II reviews the related work and theoretical background. Section III presents the proposed Adaptive LoRA FL framework and its methodology. Section IV discusses the experimental setup, evaluation metrics, and results. Section V concludes the paper and outlines future research directions.

II. Literature Review

A. Federated Learning in Heterogeneous Environments

Federated Learning (FL) is a distributed training paradigm that facilitates collaborative model development while precluding the central aggregation of sensitive raw data. This approach is intended to achieve a balance between data privacy and model efficacy. Since the introduction of FedAvg [1], research has focused on improving convergence stability under Non-IID data conditions. Specifically, FedProx [6] addresses client drift by incorporating a proximal term into the loss function, while frameworks such as FedNova [7] and SCAFFOLD [8] utilize adaptive normalization and variance reduction techniques to address gradient inconsistency.

While these conventional approaches manage data heterogeneity by modifying optimization and aggregation mechanics, they are predicated on the exchange of the full set of model parameters. As the scale of modern architectures (e.g., Large Language Models) expands, these methods encounter limitations regarding communication overhead and client-side computational demands.

The Adaptive LoRA FL framework proposed in this paper differs from the approaches in Refs. [6–8] in that it uses low-rank parameter updates rather than full parameter updates. This mechanism only requires clients to compute and transmit the LoRA parameters, with the aim of reducing communication and training overheads while maintaining fast convergence and model stability in Non-IID environments.

Nevertheless, despite these advances, the application of deep learning in AIoT and IIoT systems continues to face numerous challenges, such as data imbalance, bandwidth constraints, and device heterogeneity [9, 10]. To reduce communication overhead, multiple studies have proposed using model compression, partial layer aggregation, and layer freezing strategies [11, 12]. However, these approaches often require additional computational steps, thereby limiting their scalability and feasibility in resource-constrained environments.

B. Parameter-Efficient Fine-Tuning and Low-Rank Adaptation

To enhance model training and deployment efficiency, researchers have developed various PEFT techniques. These techniques aim to reduce the number of trainable parameters while maintaining model performance. In the machine learning domain, the LoRA method proposed by Hu *et al.* [2] has made significant contributions. This approach achieves fine-tuning by injecting a low-rank matrix into the original weight matrix, thereby requiring adjustment of only a limited number of trainable parameters matrices A and B. This method significantly reduces the number of parameters requiring adjustment while maintaining performance comparable to full fine-tuning. As described in Ref. [2], LoRA enables efficient fine-tuning and knowledge transfer without modifying pre-trained model weights, substantially reducing memory consumption and computation costs during training. LoRA has demonstrated effectiveness across diverse domains, including natural language generation, visual recognition, and multimodal tasks, while exhibiting high scalability and architectural compatibility. Specifically, in Refs. [13–17], LoRA has exhibited remarkable efficacy in large language models, visual transformers, and multimodal learning tasks.

The LoRA parameterization is defined by Eq. (1), where $W_0 \in \mathbb{R}^{d\times d}$ represents the frozen pre-trained weight matrix, and $A \in \mathbb{R}^{r\times d}$, $B \in \mathbb{R}^{d\times r}$ are trainable low-rank matrices. When $r \ll d$, the number of trainable parameters can be significantly reduced.

$$W = W_0 + \Delta W = W_0 + BA \tag{1}$$

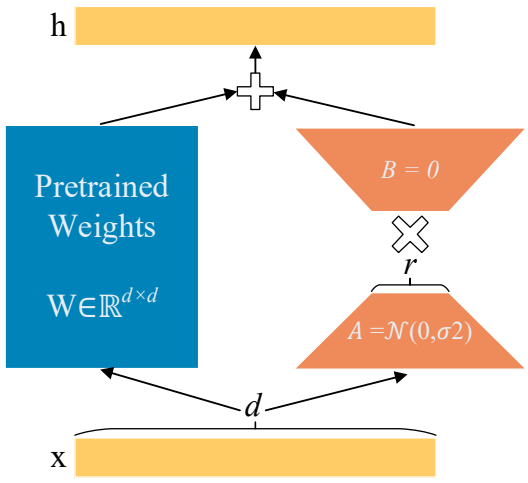

Fig. 1. LoRA Reparameterization.

As illustrated in Fig. 1, LoRA introduces low-rank matrices into each linear layer. Based on the offset, it computes the matrices A and B and adds them to the main weight outputs before passing them to the next layer. During initialization, matrix A is sampled from a normal distribution $N(0, \sigma^2)$, while matrix B is initialized as a zero matrix to ensure that the initial output aligns with the pre-trained model.

C. *Applications of LoRA in Federated Learning*

The application of LoRA has recently been introduced into the domain of federated learning with the explicit goal of mitigation communication costs. Yi *et al.* [18] and Wang *et al.* [19] demonstrated that aggregating LoRA parameters effectively minimizes the substantial communication overhead associated with Large Language Models (LLMs) in decentralized federated architectures. Crucially, the underlying principles of LoRA are inherently transferable and can be readily extended to general DNN architectures.

Specifically, in Ref. [20], aiming to enable the rapid fine-tuning of deep neural networks on low-cost embedded devices (e.g., the Raspberry Pi Zero 2W), the researchers proposed the Skip2-LoRA architecture. This approach involves the insertion of LoRA modules between layers and the incorporation of skip cache mechanism. This mechanism is designed to temporarily store intermediate forward pass results to avoid redundant computations. The subsequent experimental results indicated that Skip2-LoRA achieves accuracy comparable to full fine-tuning by updating only the LoRA parameters. These findings validates that LoRA is capable maintaining effective model performance and parameter compression capabilities even in non-transformer architectures, thereby affirming its potential as an efficient fine-tuning method for DNNs.

However, the methodology in Ref. [20] is limited by its inability to concurrently address the practical challenges of implementing LoRA within general DNN architectures in a Federated Learning.

Furthermore, prior studies [18, 19] do not concurrently address the reduction of communication overhead and training latency while sustaining model stability under complex Non-IID data distributions. Minimizing communication and training overheads while preserving model performance is essential for the effective deployment of federated learning within resource-constrained and heterogeneous environments.

III. The Proposed Adaptive LoRA Fine-Tuning Federated Learning

This section details the experimental design of the proposed Adaptive LoRA FL framework. This framework seamlessly integrates the LoRA technique into conventional DNN architectures within a FL environment. Our primary objective is to rigorously evaluate the impact of LoRA-based fine-tuning on model convergence and communication efficiency under Non-IID data distributions.

To address the applicability of the proposed approach in heterogeneous environments, three publicly available intrusion detection datasets were utilized to construct a three-client FL simulation platform. This platform was designed to emulate a realistic distributed training scenario across AIoT edge nodes. The three datasets employed are CIC-IDS2017, Edge-IIoTset, and TON-IoT.

The subsequent subsections provide a detailed account of the overall system architecture, data standardization process, deep neural network model design, layer freezing strategy, Adaptive LoRA FL training workflow, parameter optimization, and the early stopping mechanism. This collective presentation furnishes a comprehensive overview of the research methodology.

A. *System Architecture*

A federated learning environment comprising three clients was established, where each client was assigned one of the intrusion detection datasets. Specifically, Client 1 utilized CIC-IDS2017, Client 2 utilized Edge-IIoTset, and Client 3 utilized TON-IoT. Crucially, these datasets manifest notable disparities in terms of feature dimensions, feature types, and class distributions. This heterogeneity enables the experimental setup to robustly simulate a heterogeneous federated learning environment, thus closely reflecting the challenges of collaborative training with multi-source data in realistic edge computing scenarios.

As illustrated in Fig. 2, each client performs local model training using its respective dataset and subsequently uploads the updated model parameters to the central server. The server then aggregates these parameters and distributes the updated global model to the clients for the subsequent training round. This process facilitates collaborative learning without the need to share raw data, thereby ensuring data privacy while integrating knowledge across clients to enhance model generalization and robustness.

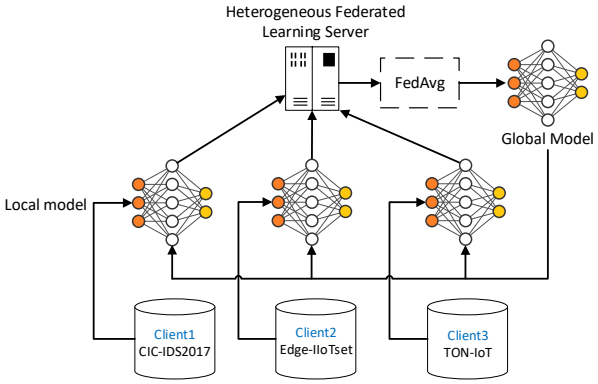

Fig. 2. The architecture of heterogeneous federated learning environment.

To further clarify the the data heterogeneity, Table I presents a summary of the basic characteristics of the three datasets, including their original feature dimensions and the number of classes. It is observable that CIC-IDS2017 contains the highest number of features, whereas TON-IoT includes the fewest, which indicates significant structural

differences among the datasets. These structural disparities consequently lead to inconsistent input and output dimensions across clients, rendering direct aggregation of local models either challenging or impractical.

TABLE I. ORIGINAL FEATURE DIMENSIONS AND CLASS COUNTS OF DATASETS

Dataset	Original Features	Classes
CIC-IDS2017	84	15
Edge-IIoTset	61	15
TON-IoT	42	10

However, due to the significant heterogeneity observed in both feature dimensions and category structures, directly performing weight aggregation on these datasets is fundamentally incompatible with the FL architecture, consequently leading to the failure of the weight aggregation process. To address this issue, a multi-stage data standardization process was developed, which comprises the following three steps:

(1) Principal Component Analysis (PCA): This seip extracts the most significant feature components to determine the minimal effective set of features.
(2) Heterogeneous Feature Mapping: This step maps the feature vectors of each dataset into a unified feature space, thereby enabling all clients to share the same input layer dimension.
(3) Class Union Method: This method unifies the output layer dimensions across clients to enable consistent global classification tasks.

Collectively, these three steps successfully mitigate the structural inconsistencies among heterogeneous datasets, thus enabling collaborative training across three distinct intrusion detection datasets. As a result, the proposed framework is able to establish a unified global model that can recognize diverse cyberattack behaviors across multiple domains.

B. Principal Component Analysis (PCA)

To reduce dimensional disparity between the three datasets, enhance training efficiency, and improve model generalization, P(PCA was applied to each dataset in an independent manner. PCA identifies the most representative principal components by performing eigenvalue decomposition on the feature covariance matrix. This process retains the main sources of variance while simultaneously eliminating redundant or correlated features.

Let the data matrix be $X \in \mathbb{R}^{n\times d}$, where n represents the number of samples and d is the original feature dimension. The data is first mean-centered such that its mean is zero, and the covariance matrix Σ is computed as in Eq. (2):

$$\Sigma = \frac{1}{n} X^T X \tag{2}$$

Eigenvalue decomposition is then performed on Σ to obtain a set of eigenvalues $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_d$ and their corresponding eigenvectors.

The Explained Variance Ratio (EVR) of the i^{th} principal component is defined as in Eq. (3):

$$\mathrm{EVR}_i = \frac{\lambda_i}{\sum_{j=1}^{d} \lambda_j} \tag{3}$$

The Cumulative Explained Variance Ratio (CEVR) for the first *k* principal components is determined by summing the individual EVR from the first to the *k*-th component, as expressed in Eq. (4):

$$\mathrm{CEVR}_k = \sum_{i=1}^{k} \mathrm{EVR}_i \tag{4}$$

This approach effectively reduces the feature dimensionality while preserving the majority of the data variance. This dimensionality reduction consequently improves the model's convergence stability and training efficiency.

To validate the efficiency of the proposed framework, a PCA was conducted to examine the intrinsic dimensionality of the feature sets. Fig. 3 illustrates the individual and cumulative explained variance ratios for each dataset, while the key parameters are summarized in Tables II and III.

Comparative analysis of the CEVR reveals significant redundancy within the original feature spaces. As summarized in Table II, achieving 95% of the CEVR necessitates the retention of 29, 32, and 25 Principal Components (PCs) for the CIC-IDS2017, Edge-IIoTset, and ToN-IoT datasets, respectively.

Conversely, the experimental results presented in Table III demonstrate that a substantial 100% of CEVR is preserved by retaining only 69, 52, and 42 PCs for the same respective datasets.

TABLE II. NUMBER OF PRINCIPAL COMPONENTS REQUIRED TO RETAIN 95% CUMULATIVE EXPLAINED VARIANCE

Dataset	Original Features	Selected PCs (*k*)
CIC-IDS2017	84	29
Edge-IIoTset	61	32
TON-IoT	42	25

TABLE III. NUMBER OF PRINCIPAL COMPONENTS REQUIRED TO RETAIN 100% CUMULATIVE EXPLAINED VARIANCE

Dataset	Original Features	Selected PCs (*k*)
CIC-IDS2017	84	69
Edge-IIoTset	61	52
TON-IoT	42	42

As illustrated in Fig. 3, a comparative analysis between the 95% and 100% CEVR highlights the inherent redundancy in full-parameter updates.

Fig. 3(a)–(c) illustrate the CEVR for the CIC-IDS2017, Edge-IIoTset, and ToN-IoT datasets, respectively. In all three cases, the CEVR curves exhibit a rapid initial ascent, indicating that the intrinsic information of each dataset is concentrated within a compact subset of highly explanatory principal components. Specifically, for the CIC-IDS2017 dataset shown in Fig. 3(a), achieving 100% PCA necessitates 69 PCs, whereas a substantial 95% threshold is reached with only 29 PCs. Similarly, Fig. 3(b)

demonstrates that the Edge-IIoTset dataset requires 52 PCs for 100% PCA, but preserves 95% of the information content using only 32 PCs. For the ToN-IoT dataset in Fig. 3(c), the 95% variance threshold is maintained with 25 PCs, a significant decrease from the 42 PCs needed for 100% PCA.

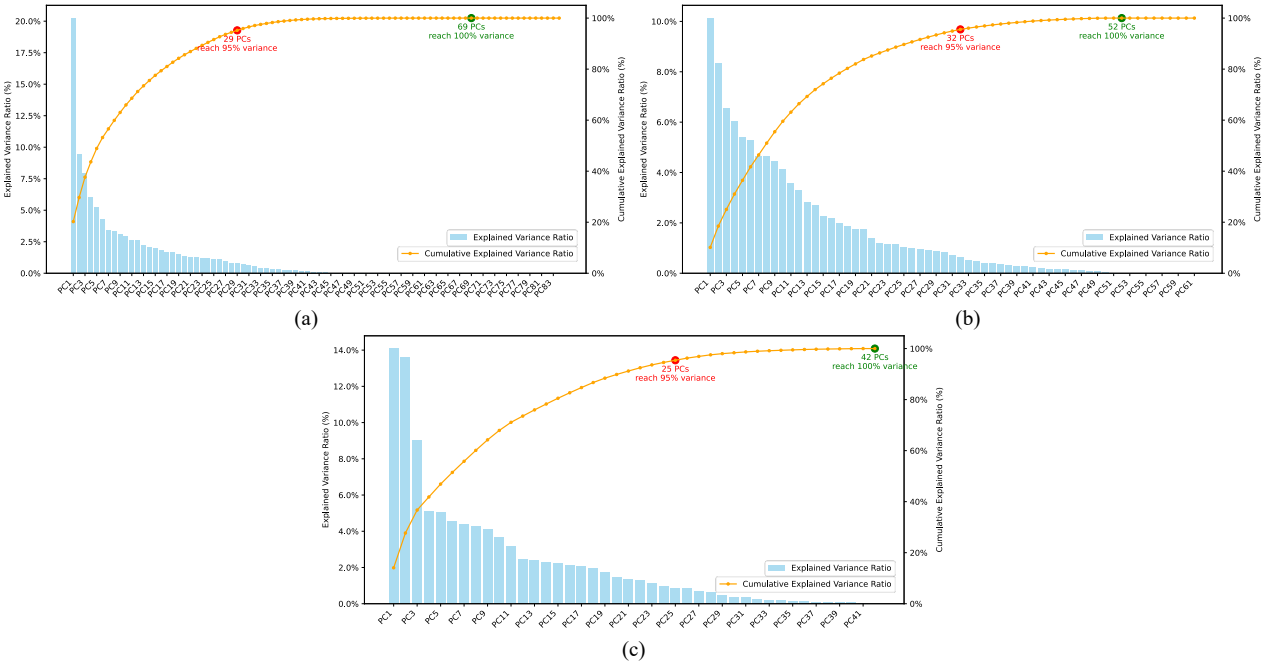

Fig. 3. The cumulative explained variance ratio for three datasets. (a) CIC-IDS2017 dataset; (b) Edge-IIoTset dataset; (c) TON-IoT dataset.

C. *Heterogeneous Feature Mapping and Zero Padding Strategy*

In addressing heterogeneous feature mapping, common input layer standardization involves aligning feature dimensions based on the maximum or minimum dimension among clients using PCA. However, this approach often results in inconsistent CEVR across clients, leading to significant information loss and adversely affecting convergence stability. Furthermore, even with unified input dimensions, the principal components extracted from disparate datasets exhibit substantial semantic inconsistency. By neglecting the physical significance of features, such discrepancies hinder effective parameter aggregation, thereby compromising the stability and generalization of the global model.

An alternative involves training a dedicated feature extractor [21] to map diverse features into a unified dimensional space. However, this method necessitates the training of additional neural network components, which inherently increases system complexity and computational overhead.

In the case of 100% PCA, the input dimensions for the three datasets vary significantly: CIC-IDS2017 retains 69 PCs, Edge-IIoTset retains 52 PCs, and TON-IoT retains 42 PCs. To facilitate parameter aggregation within the FL framework, a standardized global input layer with a dimension of 163 (the sum of all individual PCs) is established for all clients.

In this paper, a heterogeneous feature mapping approach is employed to unify the dimensions of the input layer. The global input space is defined as a 163-dimensional vector, corresponding to the sum of all PCs across the three distinct datasets. Each client is assigned an index range corresponding to its retained principal components, with the remaining positions being zero-padded. Table IV shows the client feature mapping in detail.

TABLE IV. CLIENT FEATURE MAPPING (163-DIMENSIONAL GLOBAL INPUT SPACE)

Client	Dataset	Principal Component Range	Zero-Padded Dimensions
1	CIC-IDS2017	1–69	70–163
2	Edge-IIoTset	70–121	1–69, 122–163
3	TON-IoT	122–163	1–121

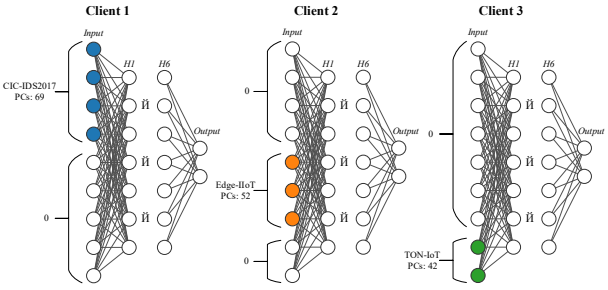

Fig. 4. Feature Mapping between 3 heterogeneous datasets.

Fig. 4 illustrates the feature mapping mechanism designed to harmonize disparate data structures, thereby elucidating the intricate interrelationships among the three heterogeneous datasets: CIC-IDS2017, Edge-IIoTset, and TON-IoT. By executing this cross-dataset alignment, the FL framework effectively mitigates discrepancies in feature dimensionality and semantic

interpretation—factors that traditionally impede collaborative training within Non-IID data environments.

This approach effectively guarantees consistent model aggregation during FL by providing all clients with a unified input layer dimension. Furthermore, the local characteristics of each dataset are meticulously preserved, thereby allowing the global model to capitalize on the integration of heterogeneous knowledge without introducing structural inconsistency.

D. Class Union Method

In heterogeneous federated learning environments, different datasets often possess distinct output class definitions. In the absence of adequate unification, such inconsistencies can significantly impede effective global model aggregation. To address this challenge, a Class Union Method was proposed to align the output layer dimensions across all clients and the global model.

The process involved the integration of all unique class labels from the three datasets into a unified class set. Subsequently, the output layer of each client was mapped to this union set. For classes that are not represented in each client's dataset, the corresponding output neurons are initialized and fixed at zero and explicitly excluded from gradient updates during training.

The implementation of class unification was preceded by the consolidation of multiple fine-grained attack labels, a process that was undertaken with the objective of optimizing the efficiency of class structures. For instance, in CIC-IDS2017, multiple DoS-related attacks (GoldenEye, Hulk, Slowhttptest, Slowloris) were consolidated into a single "DoS" category [4, 22, 23]. Consequently, CIC-IDS2017 contains 12 classes, Edge-IIoTset contains 11, and TON-IoT contains 10. The final unified output layer comprises 18 unique categories.

TABLE V. MAPPING OF ORIGINAL LABELS TO UNIFIED CLASSES

Dataset	Merged Label	Original Label
CIC-IDS2017	DoS	DoS GoldenEye DoS Hulk DoS Slowhttptest DoS slowloris
Edge-IIoTset	DDoS	DDoS_HTTP DDoS_ICMP DDoS_TCP DDoS_UDP
	Scanning	Port_Scanning Vulnerability_scanner

Table V details the specific categories within the CIC-IDS2017 and Edge-IIoTset datasets that require consolidation to facilitate subsequent union processing. This experimental analysis evaluates class recognition performance by comparing results from both before and after the consolidation process. According to the experimental data presented in Fig. 5, it is observed that class consolidation does not adversely affect the overall recognition performance.

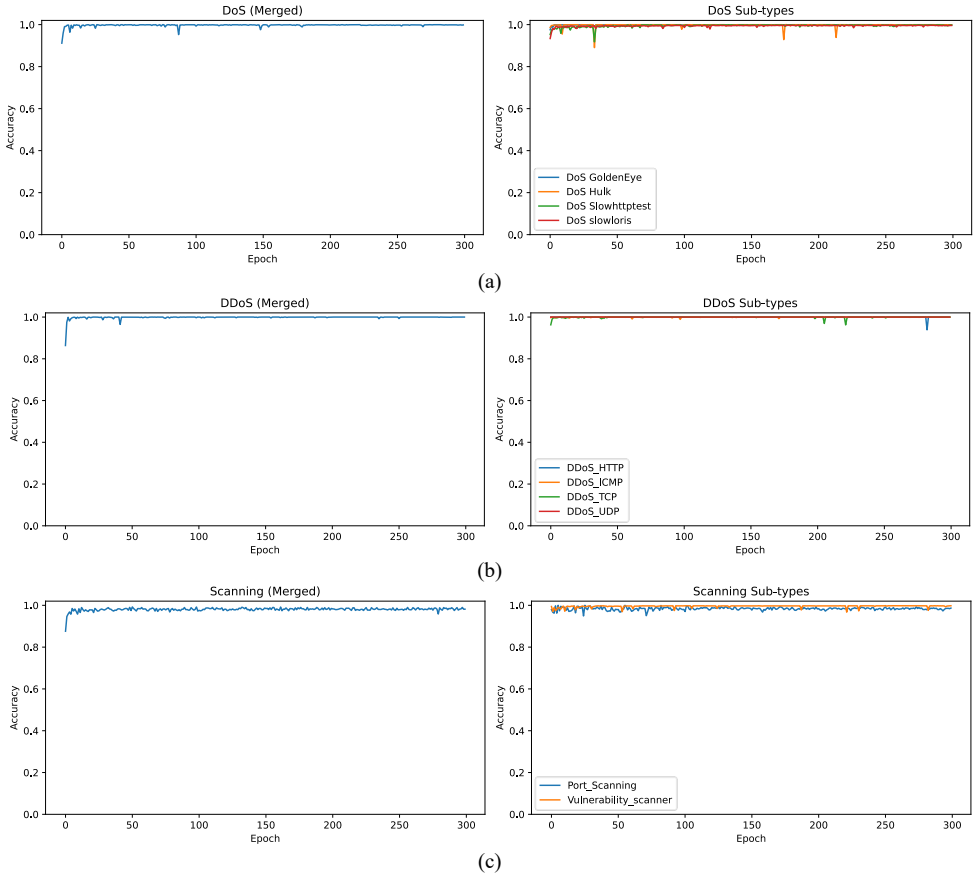

Fig. 5. Performance comparison between merged classes and sub-types. (a) DoS classes in the CIC-IDS2017 dataset; (b) DDoS classes in the Edge-IIoTset dataset; (c) Scanning classes in the Edge-IIoTset dataset.

Tables VI and VII present a quantitative evaluation of the classification performance following the implementation of the category unification mechanism for the CIC-IDS2017 and Edge-IIoTset datasets, respectively. The objective of this analysis is to determine whether the consolidation of heterogeneous labels adversely affects the model's discriminative capabilities.

For the CIC-IDS2017 dataset (Table VI), the experimental metrics indicate that consolidating various DoS-related attack labels into a single, unified category resulted in negligible fluctuations in system performance. All consolidated categories maintained high recall, suggesting that the model successfully captured the shared underlying characteristics of these attack types.

Similarly, the results for the Edge-IIoTset dataset (Table VII) demonstrate robust recognition stability after merging DDoS-related and Scanning-related labels. The consistency across performance metrics confirms that the integration strategy successfully facilitates dataset consolidation without compromising the integrity of feature representations or model generalization.

TABLE VI. PERFORMANCE COMPARISON BETWEEN MERGED CLASSES AND SUB-TYPES IN THE CIC-IDS2017 DATASET

CIC-IDS2017	Recall (%)	Merged Label	Recall (%)
DoS GoldenEye	98.84	DoS	99.84
DoS Hulk	99.92		
DoS Slowhttptest	99.78		
DoS slowloris	99.58		

TABLE VII. PERFORMANCE COMPARISON BETWEEN MERGED CLASSES AND SUB-TYPES IN THE EDGE-IIOTSET DATASET

Edge-IIoTset	Recall (%)	Merged Label	Recall (%)
DDoS_HTTP	100	DDoS	100
DDoS_ICMP	99.92		
DDoS_TCP	100		
DDoS_UDP	100		
Port_Scanning	98.68	Scanning	98.16
Vulnerability_scanner	99.80		

As outlined in Table VIII, this class union method design effectively resolves output-layer inconsistency across heterogeneous datasets and substantially enhances the global model's ability to recognize diverse intrusion patterns in federated learning environments.

TABLE VIII. MAPPING OF UNIFIED CLASS LABELS ACROSS HETEROGENEOUS INTRUSION DETECTION DATASETS

Label ID	Union Label	CIC-IDS2017	Edge-IIoTset	TON-IoT
0	Backdoor	-	Backdoor	Backdoor
1	Bot	Bot	-	-
2	Brute Force	Web Attack Brute Force	-	-
3	DDoS	DDoS	DDoS_HTTP/ICMP/TCP/UDP	DDoS
4	DoS	DoS GoldenEye/Hulk/Slowhttptest/slowloris	-	DoS
5	Fingerprinting	-	Fingerprinting	-
6	FTP	FTP-Patator	-	-
7	Heartbleed	Heartbleed	-	-
8	Infiltration	Infiltration	-	-
9	Injection	Web Attack Sql Injection	SQL_injection	Injection
10	MITM	-	MITM	MITM
11	Normal	BENIGN	Normal	Normal
12	Password	-	Password	Password
13	Ransomware	-	Ransomware	Ransomware
14	Scanning	PortScan	Port_Scanning/Vulnerability_scanner	Scanning
15	SSH	SSH-Patator	-	-
16	Uploading	-	Uploading	-
17	XSS	Web Attack XSS	XSS	XSS

The unified framework employed in this paper ensures consistent interpretation of class labels across clients. Crucially, this enables the global model to seamlessly integrate heterogeneous datasets and achieve optimal training efficiency.

E. Model Architecture Design

All three clients in this study employed the same DNN architecture to ensure model consistency and the feasibility of aggregation across heterogeneous data environments. As shown in Fig. 6, this DNN model comprises an eight-layer structure, including one input layer, six hidden layers, and one output layer. The input layer dimension is set to 163, corresponding to the unified feature size generated during the heterogeneous feature mapping process. Each of the six hidden layers contains 128 neurons, while the output layer dimension corresponds to 18 unified output classes.

In order to ensure the optimal balance between model performance and communication efficiency, this DNN architecture will serve as the baseline model for the proposed Adaptive LoRA FL experiments. In Adaptive LoRA FL, the LoRA matrix is injected into each weight layer to validate its feasibility and convergence characteristics within the FL framework.

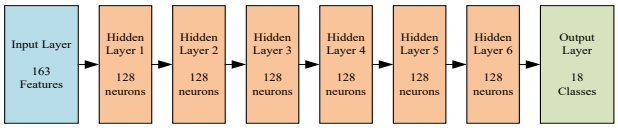

Fig. 6. The overall architecture of the DNN model.

F. Data Sampling Strategy and Sample Cap Validation

To validate the empirical reasonableness of setting a sampling upper limit of 10,000 records, this study conducted a comparative experiment benchmarking the restricted sampling approach against the unrestricted datasets. The evaluation utilized a DNN model trained locally using 300 epochs. Note that the ToN-IoT dataset inherently imposes sample size constraints within its original architecture, this comparative analysis was focused exclusively on the CIC-IDS2017 and Edge-IIoTset datasets.

As illustrated in Figs. 7 and 8, the experimental results reveal a critical performance divergence during the training process. In the absence of sample restrictions, the model exhibits a progressive bias toward majority categories as the number of epochs increases. This phenomenon leads to a significant degradation in the recognition capability of minority attack categories. The overall model performance declines sharply when the full dataset is utilized, characterized by a severe collapse in recall rates for minority classes.

These findings objectively demonstrate that the limit of 10,000 records cap serves as an essential regularization mechanism. By mitigating the dominance of majority classes, the sampling limit preserves the integrity of feature learning across all attack categories.

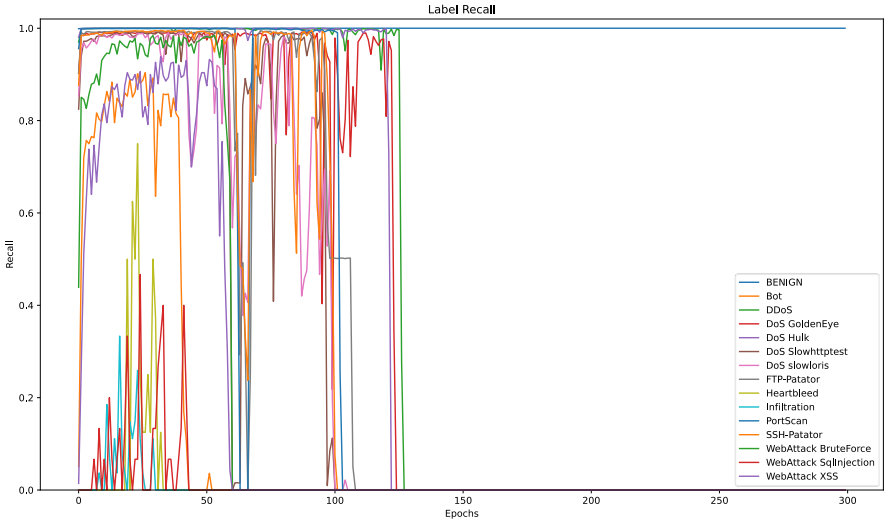

Fig. 7. Class-wise recall on CIC-IDS2017 without sample cap (300 epochs).

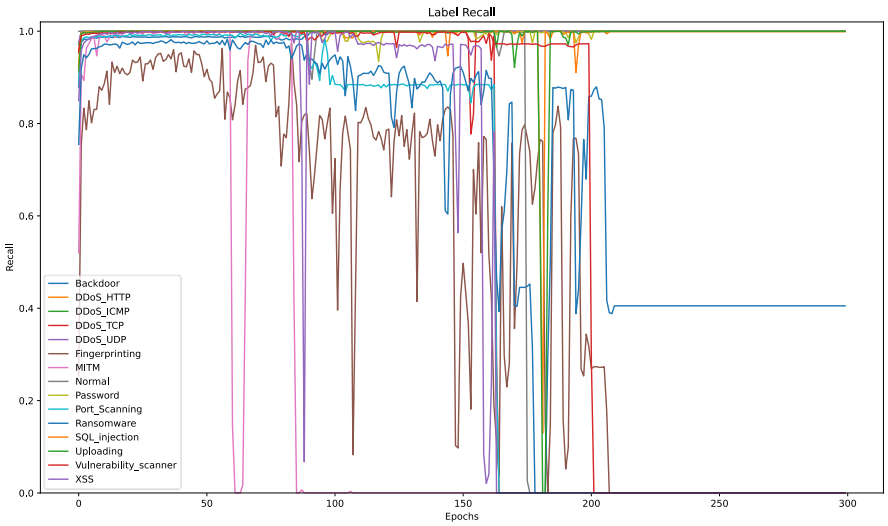

Fig. 8. Class-wise recall on Edge-IIoTset without sample cap (300 epochs).

In order to mitigate the impact of class imbalance during model training, a class-based sampling strategy was adopted during the data preprocessing stage. A maximum sampling threshold of 10,000 samples per class was set.

The sampling rules were defined as follows:

- Oversized Classes (Sample Count >10,000): In instances where the number of samples exceeded the specified threshold, a random subset of samples was selected.
- Undersized Classes (Sample Count ≤ 10,000): In instances where the number of samples fell below the designated threshold, the entire set of samples was retained without modification.

After the sampling stage, each dataset was divided into a training dataset constituting 75% of the total and a testing dataset accounting for the remaining 25%. This process was undertaken to ensure consistency and reproducibility during the training and evaluation of the model.

The final sample distributions for the three publicly available intrusion detection datasets are summarized in Tables IX–XI. The total processed sample counts are: 58,015 for CIC-IDS2017, 91,401 for Edge-IIoTset, and 91,043 for TON-IoT. These processed datasets will serve as the primary input for subsequent experiments.

TABLE IX. CIC-IDS2017 DATASET SAMPLING AND PARTITION RESULTS

Class	Total Samples	Training Set	Testing Set
BENIGN	10000	7500	2500
Bot	1956	1467	489
DDoS	10000	7500	2500
FTP-Patator	7935	5951	1984
Heartbleed	11	8	3
Infiltration	36	27	9
PortScan	10000	7500	2500
SSH-Patator	5897	4422	1475
Web Attack Brute Force	1507	1130	377
Web Attack Sql Injection	21	15	6
Web Attack XSS	652	489	163
DoS	10000	7500	2500
Total	58015	43509	14506

TABLE X. EDGE-IIOTSET DATASET SAMPLING AND PARTITION RESULTS

Class	Total Samples	Training Set	Testing Set
Backdoor	10000	7500	2500
Fingerprinting	1001	750	251
MITM	400	300	100
Normal	10000	7500	2500
Password	10000	7500	2500
Ransomware	10000	7500	2500
SQL_injection	10000	7500	2500
Uploading	10000	7500	2500
XSS	10000	7500	2500
DDoS	10000	7500	2500
Scan	10000	7500	2500
Total	91401	68550	22851

TABLE XI. TON-IOT DATASET SAMPLING AND PARTITION RESULTS

Class	Total Samples	Training Set	Testing Set
Backdoor	10000	7500	2500
DDoS	10000	7500	2500
DoS	10000	7500	2500
Injection	10000	7500	2500
MITM	1043	782	261
Normal	10000	7500	2500
Password	10000	7500	2500
Ransomware	10000	7500	2500
Scanning	10000	7500	2500
XSS	10000	7500	2500
Total	91043	68282	22761

G. Layer-Freezing Strategies

In the heterogeneous federated learning framework, the present study investigates how varying freeze depths impact the convergence speed and training stability of the global model based on a hierarchical freezing mechanism. Within the DNN architecture, seven distinct freezing configurations are defined, each corresponds to a distinct number of frozen hidden layers. It is imperative to note that all experiments adhere to a uniform training procedure; during the initial five global training epochs, all layers are maintained in a fully trainable state to ensure comprehensive parameter initialization. Beginning from the sixth global training epoch, specific freezing strategies are applied to freeze partial layers, while the remaining layers continue to update. The definition of the progressive freezing strategy is shown in Fig. 9.

- Freeze0: This configuration serves as the baseline, where no layer is frozen. Consequently, all eight layers (input, six hidden, and output) remain fully trainable.
- Freeze1 to Freeze5: These configurations indicate that the freezing process begins at the first hidden layer and proceeds sequentially, with the number of frozen layers increasing progressively up to five (Freeze5).
- Freeze6: In this most restrictive configuration, all six hidden layers are frozen, leaving only the input and output layers available for training.

The investigation utilized the progressive layer-freezing strategies to systematically analyze the impact of each strategy on training efficiency, final accuracy and convergence stability. The optimal freezing depth obtained from these preliminary experiments was subsequently adopted as the baseline configuration for the proposed Adaptive LoRA FL framework.

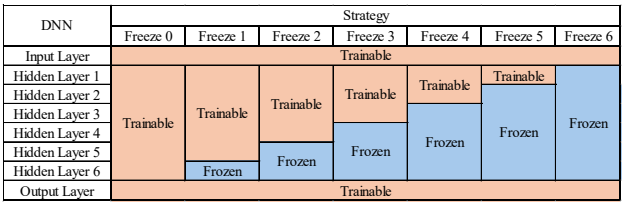

Fig. 9. Illustration of the progressive freeze strategy.

H. Weight-Based Federated Learning (Weight FL)

In order to establish a baseline for comparison, we implemented the Weight Federated Learning (Weight FL) model based on the standard weight-based federated learning framework. The training process comprised 300 rounds of global training, with each round consisting of the following three phases:

(1) Local Training and Dynamic Epoch Adjustment: Each client independently trains its local model. A dynamic epoch strategy was employed:

- During the initial five global rounds, 300 epochs were used to accelerate initial convergence.
- From the sixth round onwards, the number of local epochs was reduced to 50 to maintain model stability and optimized communication overhead.

(2) Aggregation: Clients upload their updated full weight parameters to the central server. The server performs parameter aggregation using the FedAvg algorithm to update the global model weights.

(3) Evaluation and Synchronization: The updated global model is distributed back to all participating clients for evaluation using their local test sets. The resulting accuracy metric is then reported to the server for logging and comparative analysis.

This standard Weight FL process serves as the critical baseline for evaluating the performance improvements introduced by the proposed Adaptive LoRA FL method.

I. Adaptive LoRA FL

The proposed Adaptive LoRA FL framework integrates LoRA into federated learning to significantly reduce full weight communication overhead and improve training efficiency. Adaptive LoRA FL differs fundamentally from standard weight FL in its training methodology, employing the following two-stage training process:

(1) Initial Full Weight Training Phase: All clients initially train using the full weight model, uploading weights to the central server for FedAvg aggregation after each global round. This phase is threshold-driven: when the aggregated global model achieves an 80% accuracy threshold across all clients, model training advances to the next phase. This stage ensures all clients possess a robust, pre-trained model, which serves as the critical foundation for LoRA fine-tuning and is essential for effective adaptation in the subsequent phase.

(2) LoRA Fine-Tuning and Low-Rank Parameter Transmission Phase: In the second phase, all clients' full weights parameters are frozen. Trainable LoRA matrices (A/B matrices) will be injected into each weight layer, specificatlly the fully connected layers. Subsequently, each round of client training updates only the LoRA parameters, and only these low-rank parameters are transmitted to the server for aggregation. The aggregated LoRA parameters are then be returned by the server to all clients. This mechanism significantly reduces communication overhead and accelerates model convergence.

J. Early Stopping Mechanism

In order to avoid unnecessary training after model convergence, the central server will employ an early stopping mechanism. Algorithm 1 presents the pseudocode for this mechanism. For each global epoch *r*, the global accuracy G_r is computed and recorded to one decimal place. The stopping condition is defined as flows: if the accuracy remains within a ±0.5% tolerance of the historical best value for five consecutive epochs ($P = 5$), the server will trigger an early stopping signal and notify all clients to terminate training. This mechanism simultaneously reduces both training time and communication overhead.

Algorithm 1: Early Stopping Mechanism at the Server Side

Input: *Global accuracy G_r evaluated at each global round*
Output: *Trigger signal for early stopping*
Parameters: *Patience P = 5, tolerance δ = 0.5%*

Initialize: best = 0, counter = 0
for each global round r = 1 to R **do**
  Compute global accuracy G_r
  Round to one decimal place: $\hat{G}_r = \text{round}(G_r, 1)$
  if $\hat{G}_r >$ best **then**
    best ← $\hat{G}_r$
    counter ← 0
  else if $|\hat{G}_r - \text{best}| \leq \delta$ **then**
    counter ← counter + 1
  else
    counter ← 0
  end if

  if counter = P **then**
    Trigger early stopping and terminate training
    break
  end if
end for

IV. Experimental Results

This section details the experimental design, environment setup, evaluation metrics, and results analysis for three distinct FL frameworks: Weight FL, LoRA FL, and Adaptive LoRA FL.

Three clients utilize the CIC-IDS2017, Edge-IIoTset, and TON-IoT datasets, respectively, to analyze experimental results under Non-IID data distribution conditions. These results specifically compare the three FL frameworks in terms of three key performance indicators: global accuracy, communication cost, and training time.

A. Experimental Environment

This subsection details the hardware and software configurations, as well as the data distribution across clients, to evaluate the proposed federated learning architecture and training methodology. The system comprises one server and three clients, all of which run on separate physical machines in order to simulate a realistic distributed federated learning setup. The specific configurations are detailed as follows:

- Hardware Configurations

(1) FL Server: Equipped with an Intel Core i5-9400 CPU and 16 GB RAM, lacking a Graphics Processing Unit (GPU).
(2) Client Nodes (Client 1–3): Each client node is equipped with an Intel Core i5-14400 CPU, an NVIDIA GeForce RTX 3060 GPU, and 16 GB RAM.

- Software Configurations

(1) Operating System: Windows 10.
(2) Training Environment: Configured using Python 3.9.
(3) Federated Learning Framework: Flower 1.9.0.
(4) Model Training Library: PyTorch 2.5.1.

B. Evaluation Metrics

The performance evaluation of the three FL frameworks Weight FL, LoRA FL, and Adaptive LoRA FL will employ three core metrics: accuracy, communication cost, and total training time. The definition and purpose of each metric are as follows:

- Accuracy: Accuracy quantifies the global model's classification capability across the combined test sets of the CIC-IDS2017, Edge-IIoTset, and TON-IoT datasets.
- Communication Cost: Communication cost measures the average data transfer between the server and clients during the federated learning process.
- Total Training Time: The total training time presents the cumulative duration from the start to the completion of the federated learning process, reflecting the overall training efficiency of the system.

1) Accuracy

Accuracy is a fundamental metric used to assess the classification performance of a model. The proportion of correctly predicted samples to the total number of samples is defined as in Eq. (5):

$$Accuracy = \frac{TP+TN}{TP+TN+FP+FN} \quad (5)$$

where TP(True Positive) and TN(True Negative) denote the number of correctly predicted samples, while FP(False Positive) and FN (False Negative) represent the misclassified ones.

2) Communication cost

Communication cost serves as the primary metric for measuring transmission efficiency within the federated learning framework. As shown in Eq. (6), communication cost calculates the average total data exchanged between clients and servers throughout the entire training process, measured in Megabytes (MB).

The calculation of the communication cost is based on the data exchange procedure in each global round *r* of the federated learning process:

- Server to Clients (Download): The server transmits the aggregated global parameters (weights or low-rank matrices) to the $N = 3$ participating clients.
- Clients to Server (Upload): After all clients complete their local training, each client uploads its updated trained parameters to the server for subsequent aggregation.

During each training round *r*, each client *i* records two transmission components: the uploaded data amount ($S_{i,r}^{upload}$) and the downloaded data amount ($S_{i,r}^{download}$).

The average communication cost ($C_{average}$) is calculated by aggregating all transmission amounts across all clients and rounds, and then computing the average data exchange per client, defined as follows in Eq. (6):

$$C_{average} = \frac{1}{N}\sum_{i=1}^{N}\sum_{r=1}^{R}(S_{i,r}^{upload} + S_{i,r}^{download}) \quad (6)$$

where R denotes the total number of global rounds, and N represents the number of clients ($N = 3$).

3) Total training time

The total Training time (T_{total}) serves as the primary metric for evaluating training efficiency within the federated learning framework. As shown in Eq. (7), T_{total} calculates the cumulative duration from the initiation to the completion of the entire federated learning process, measured in seconds (s).

The total training time is measured based on the latency encountered in each global round. During each training round *r*, each client records two components:

- Local training time (T_{local}).
- Communication time (T_{comm}).

In synchronous FL, the server must wait for the slowest client to complete its local task. Therefore, the total time for each round is dominated by the maximum client time. The total training time T_{total} is calculated by summing the maximum time taken across all clients in each round, defined as follows in Eq. (7):

$$T_{total} = \sum_{r=1}^{R}\left(\max_{N\in\{1,\ldots,N\}}(T_{local} + T_{comm})\right) \quad (7)$$

where R denotes the total number of global rounds, and N represents the number of clients ($N = 3$).

C. Experimental Results of Weight FL

This subsection presents the experimental results of the Weight-based Federated Learning (Weight FL) to evaluate the effect of different layer-freezing strategies on global model convergence speed and stability. All three clients adopted identical hyperparameters, which are summarized in Table XII.

TABLE XII. FEDERATED LEARNING HYPERPARAMETERS UNDER HETEROGENEOUS ENVIRONMENTS

Hyperparameter	Value/Configuration
Batch Size	512
Learning Rate	0.0005
Optimizer	Adam
Activation Function	ReLU
Epochs	50
Global Round	300
Aggregation Strategy	FedAvg

1) Comparison of seven layer-freezing strategies

To further analyze the effect of different layer-freezing depths on model convergence behavior, seven freezing strategies, denoted as Freeze0 through Freeze6, were systematically evaluated.

As illustrated in Fig. 10, the evolution of model accuracy was tracked over 300 global rounds. Each figure provides a comparative view of the local accuracy achieved by the three participating clients (trained on the CIC-IDS2017, Edge-IIoTset, and ToN-IoT datasets) and the resulting global accuracy aggregated at the server.

To provide a quantitative assessment of convergence efficiency, a threshold of 99% global accuracy (0.99) was established. The convergence speed is defined as the first global round in which the aggregated model reaches or exceeds this threshold, indicated by a vertical dashed line in the respective plots. This criterion allows for an objective comparison across varying freezing depths.

The experimental results indicate that shallow freezing configurations (Freeze0 and Freeze1) fail to trigger the convergence condition within the 300-round limit. This observation suggests that insufficient freezing leads to slower convergence and higher performance fluctuations, likely due to the high dimensionality of the update space in a heterogeneous environment.

Conversely, Fig. 10(d) demonstrates that the Freeze3 strategy achieves the 0.99 threshold at the earliest global round, representing the optimal balance between parameter plasticity and training stability. While Freeze3 exhibits the fastest convergence, deeper freezing strategies (Freeze4–Freeze6) show a trend toward slower convergence or reduced adaptability. This indicates that excessive freezing may overly constrain the model's capacity to learn from the diverse local data distributions in heterogeneous environments.

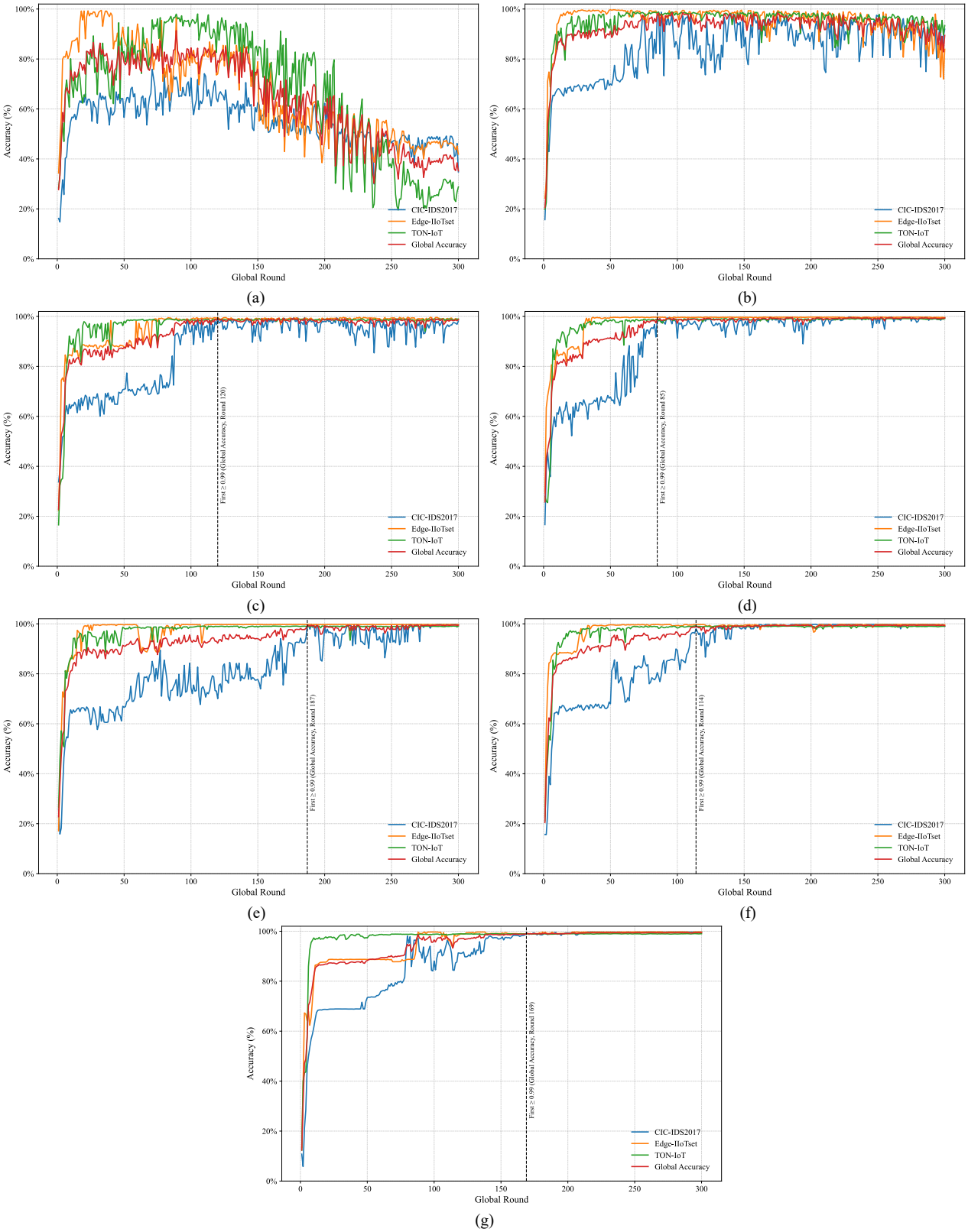

Fig. 10. Global accuracy of weight FL under different layer-freezing strategies (convergence threshold = 0.99). (a) Freeze0: Not triggered; (b) Freeze1: Not triggered; (c) Freeze2: Triggered at round 120; (d) Freeze3: Triggered at round 85; (e) Freeze4: Triggered at round 187; (f) Freeze5: Triggered at round 114; (g) Freeze6: Triggered at round 169.

In summary, the Freeze3 freezing strategy demonstrated the most balanced performance across all datasets, exhibiting significantly faster convergence and enhanced stability in both local and global accuracy.

Consequently, the Freeze3 freezing strategy was selected as the baseline configuration for all subsequent optimization experiments.

2) Parameter optimization and early stopping mechanism

Following the confirmation of the Freeze3 freezing strategy as the optimal configuration, further training parameter optimization and early stopping mechanisms were implemented to substantially enhance convergence efficiency and stability. Specifically, experimental results from Fig. 10(d) indicated that Client 1 (CIC-IDS2017) exhibited slower convergence speed compared to other clients. To address this disparity, the learning rate for Client 1 was incrementally increased from 0.0005 to 0.001. Conversely, Clients 2 (Edge-IIoTset) and 3 (TON-IoT) retained the default learning rate of 0.0005. The final, optimized learning rate configuration is presented in Table XIII.

TABLE XIII. LEARNING RATE CONFIGURATION FOR WEIGHT FL OPTIMIZATION

Client	Dataset	Learning Rate
Client1	CIC-IDS2017	0.001
Client2	Edge-IIoTset	0.0005
Client3	TON-IoT	0.0005

Following the incorporating the early stopping mechanism and learning rate tuning, the enhanced experimental results are presented in Fig. 11 and Table XIV. The optimized Freeze3 freezing strategy configuration successfully triggered the early stopping mechanism during the 135th global training round, achieving a peak global accuracy of 98.91%. Compared to the baseline results presented in Fig. 10(d), the optimized parameters significantly reduced the number of required global training rounds (from 300 to 135). This experiment conclusively validates the efficacy of both the tailored learning rate tuning and the strategic implementation of the early stopping mechanism in improving training efficiency and accelerating model convergence.

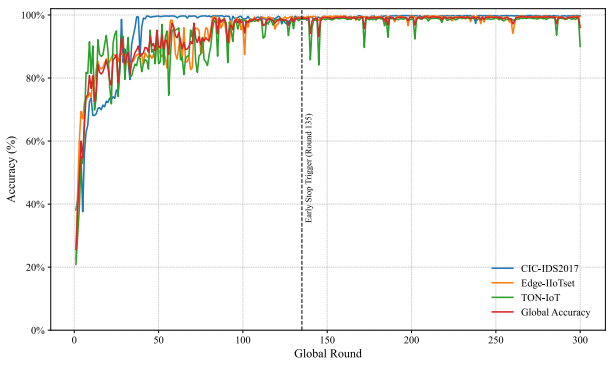

Fig. 11. Global accuracy of weight FL Using the optimized Freeze3 freezing strategy.

TABLE XIV. EARLY STOPPING PERFORMANCE OF THE OPTIMIZED FREEZE3 STRATEGY

Client	Dataset	Accuracy (%)
Client1	CIC-IDS2017	99.24
Client2	Edge-IIoTset	98.96
Client3	TON-IoT	98.65
-	Global	98.91

D. Experimental Results of LoRA FL

This section analyzes the experimental results of the LoRA-based federated learning (LoRA FL) framework. The LoRA matrix (rank = 8) is injected into each linear layer without utilizing any pre-trained weights. Under this configuration, each client is responsible for updating and uploading only the LoRA parameters during the training process, while the server utilizes the FedAVG algorithm to aggregate these LoRA parameters across all participating clients.

The experimental results, detailed in Fig. 12, clearly demonstrate that the standard LoRA FL framework exhibits significant instability throughout the training process, with its overall performance being substantially lower than that of the Weight FL baseline. This instability is primarily attributed to the lack of an initial weight pretraining phase, which subsequently hinders the model's ability to achieve stable convergence. Specifically, the aggregated global accuracy fluctuates markedly between 70% and 80% and consequently fails to trigger the early stopping mechanism. These findings indicate that relying solely on LoRA parameter updates severely limits the model's generalization capability within heterogeneous data environments, which in turn significantly reduces the stability of global aggregation process. This observation underscores the critical importance of the pretraining stage in maintaining the stability and performance of the global model.

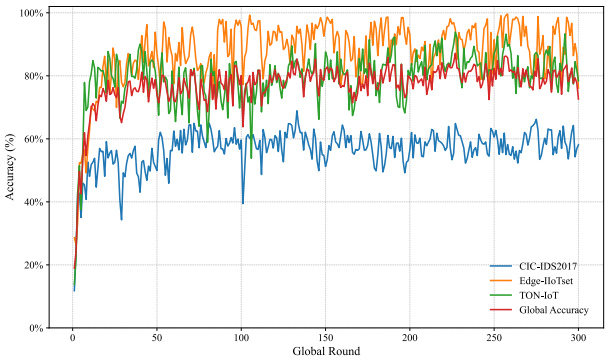

Fig. 12. Global accuracy of LoRA FL.

E. Experimental Results of Adaptive LoRA FL

This subsection analyzes the experimental results of the Adaptive LoRA FL framework proposed in this paper. To ensure experimental fairness and consistency, the optimal freezing strategy (Freeze3) identified from the Weight FL experiments was adopted as the baseline configuration. The framework operates in two distinct phases:

1) Weight training phase

During the Weight Training Phase, all clients train the model using the full weight parameters. Following the conclusion of each global epoch, the updated model parameters are uploaded to the server for aggregation. Since LoRA technology is not yet enabled during this phase, the parameter transmission cost remains identical to that of the full-weight learning (Weight FL). The server is designed to trigger the next phase only when the

aggregated global model achieves an accuracy of 80% on the test dataset across all clients. This design critically ensures that all participating clients establish a stable and adequate pre-trained model foundation before the transition into the LoRA fine-tuning phase.

2) LoRA fine-tuning phase

Upon reaching the LoRA fine-tuning activation threshold (i.e., 80% accuracy across all clients), the server instructs all clients to freeze their current weight parameters and inject the LoRA matrix (set at rank r = 8). These LoRA parameters are specifically injected into the deep neural network layers for subsequent fine-tuning, as visually detailed in Fig. 13.

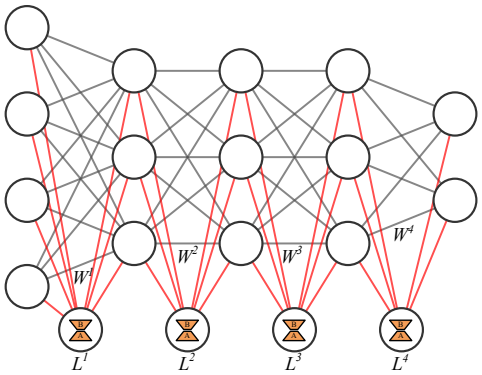

Fig. 13. Illustration of LoRA initialization injected into DNN layers.

At this fine-tuning stage, the client uploads only the compressed LoRA parameters to the server each round, while the server merely aggregates and updates these matrices. This approach significantly reduces communication costs compared to full-weight updates. Furthermore, since this phase involves LoRa fine-tuning rather than full weight training, the learning rate is subsequently lowered to ensure stable convergence and effectively prevent overfitting. All relevant training settings and final configurations for this stage are detailed in Table XV.

TABLE XV. LEARNING RATE CONFIGURATION FOR THE LORA FINE-TUNING PHASE IN ADAPTIVE LORA FL

Client	Dataset	Learning Rate
Client1	CIC-IDS2017	0.001
Client2	Edge-IIoTset	0.0005
Client3	TON-IoT	0.0005

3) Experimental results and analysis

To further evaluate the impact of dimensionality reduction on the training stability of the Adaptive LoRA FL framework, a comparative analysis was conducted between the 95% and 100% PCA configurations. Figs. 14 and 15 illustrate the global accuracy evolution for both configurations.

A significant divergence in convergence behavior is observed during the later stages of training. Specifically, the PCA 95% configuration (Fig. 14) exhibits pronounced fluctuations in global accuracy. In contrast, the PCA 100% setting (Fig. 15) demonstrates a more stable convergence trend following the activation of the LoRA mechanism. Consequently, the PCA 100% configuration was selected for the final system parameters to ensure superior training stability within the federated learning environment.

Fig. 15 shows that under the 100% PCA configuration, all clients performed full parameter training during the initial 36 rounds of the recommended Adaptive LoRA FL global training. When the local accuracy of all clients reached the 80% threshold, the activation condition for LoRA fine-tuning was triggered. Upon meeting this condition, the system entered the LoRA fine-tuning phase, during which the original DNN model weights were frozen. Global accuracy then steadily increased until the early stopping mechanism was triggered at the 73rd round. The average global accuracy reached 99.27%. Relevant performance metrics are detailed in Table XVI.

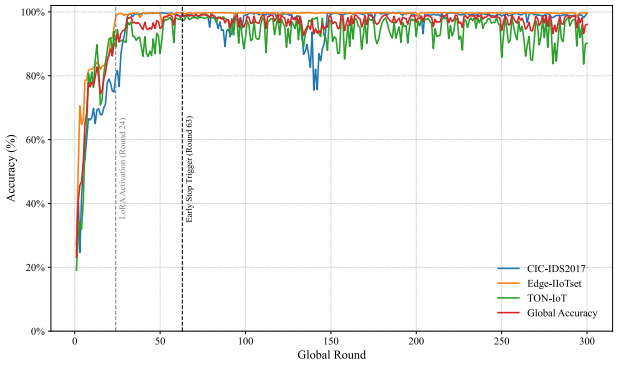

Fig. 14. Global accuracy curve of Adaptive LoRA FL based on the Freeze3 strategy (using 95% PCA).

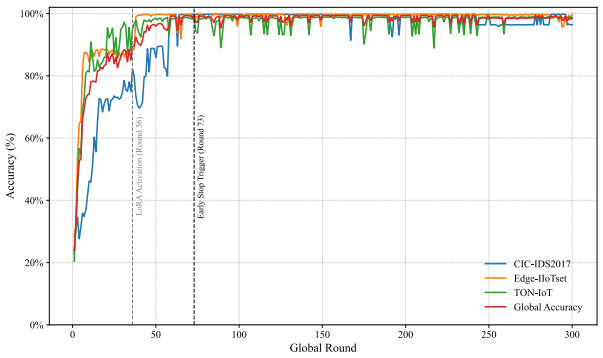

Fig. 15. Global accuracy curve of Adaptive LoRA FL based on the Freeze3 strategy (using 100% PCA).

TABLE XVI. EXPERIMENTAL RESULTS OF ADAPTIVE LORA FL (USING 100% PCA)

Client	Dataset	Accuracy (%)
Client1	CIC-IDS2017	99.71
Client2	Edge-IIoTset	99.70
Client3	TON-IoT	98.56
-	Global	99.27

F. Comparison of Experimental Results

This subsection systematically analyzes and compares the experimental results obtained from three distinct federated learning architectures: Weight FL, LoRA FL, and Adaptive LoRA FL framework proposed in this paper. All experiments were conducted five times under identical experimental settings, and the reported results correspond

to the average performance across these runs. The primary objective of this comparative study is to quantitatively evaluate the practical impact of the proposed Adaptive LoRA FL framework on convergence stability and communication efficiency when deployed within heterogeneous federated learning environments.

1) Comparison of global accuracy

The comparison results for global accuracy across the three frameworks are presented in Fig. 16. After completing 300 rounds of global training, the weighted FL baseline achieved an average global accuracy of 98.61%. Upon enabling the early stopping mechanism, the global accuracy of Weighted FL improved significantly to 99.09%, which confirms that the early stopping mechanism effectively halts training at a relatively stable state of the model.

In contrast, the LoRA FL framework without a weight pre-training phase, achieved only 80.22% global accuracy and exhibited pronounced instability throughout training. This instability strongly emphasizes the necessity of weight pre-training when applying LoRA to heterogeneous federated environments.

Crucially, the proposed Adaptive LoRA FL framework achieved a global accuracy of 97.82% after 300 global rounds. When combined with an early stopping mechanism, the global accuracy further improved to 98.58%.

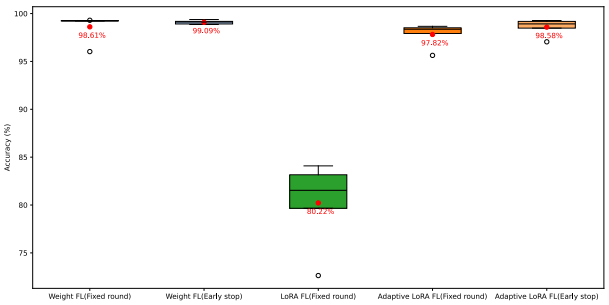

Fig. 16. Comparison of global accuracy among different FL methods.

2) Comparison of communication cost

Communication cost is formally defined as the average data transmission amount from the client to the server throughout the entire training process. The comparative results for this metric are presented in Fig. 17.

The Weighted FL baseline incurred an average communication cost of 278.49 MB upon completing 300 global rounds. When the early stopping mechanism was enabled in Weight FL, the total average communication cost decreased substantially to 146.89 MB.

In contrast, the LoRA FL further reduces the communication cost to 37.40 MB. Although this approach achieves the lowest transmission overhead, the lack of a pre-training phase resulted in unstable model convergence.

Crucially, the proposed Adaptive LoRA FL framework attains a competitive communication cost of 64.81 MB across 300 global rounds. When integrated with the early stopping mechanism, the average communication cost is further optimized to 39.26 MB. This results in a substantial 85.9% reduction in transmission overhead compared to the Weight FL baseline. These findings underscore the framework's ability to achieve high-efficiency data exchange, making it particularly well-suited for bandwidth-constrained environments.

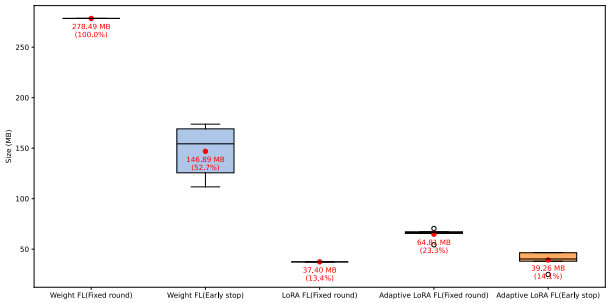

Fig. 17. Comparison of total communication cost among different FL methods.

3) Comparison of total training time

Fig. 18 illustrates the total training time required for the different federated learning architectures.

The Weight FL baseline required 10,248.8 s to complete 300 rounds of global training. When the early stopping mechanism was enabled, training was terminated at an average of 158 global rounds across five independent runs, reducing the total time to 5765.6 s, representing a significant time saving of approximately 43.74%.

In contrast, although the LoRA FL framework incurred lower communication costs, the necessity for additional updates to LoRA parameters and gradient computations at each fully connected layer led to a longer training duration than Weight FL. Furthermore, the LoRA FL requires longer training times than Weight FL. Furthermore, the absence of a weight pre-training phase meant the model was unable to trigger the early stopping mechanism. Consequently, completing 300 rounds of global training with LoRA FL required 12,150.2 s.

Crucially, the proposed Adaptive LoRA FL required 11,809.4 s to complete 300 global rounds. However, the effective weight pre-training accelerated convergence, allowing the training to successfully trigger the early stopping mechanism at an average of 89 global rounds across five independent runs. This resulted in the total training time being dramatically reduced to 3927.8 s, which is 61.68% significantly lower than the optimized Weight FL approach.

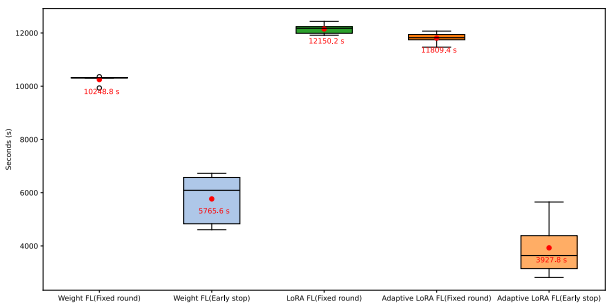

Fig. 18. Comparison of total training time among different FL methods.

Although the LoRA FL framework achieves the lowest average communication cost (37.40 MB), it suffers from insufficient global stability and relatively low accuracy (80.22%). In contrast, the Adaptive LoRA FL framework effectively balances communication efficiency and convergence performance through its innovative two-stage design. The initial full-weight training phase establishes a stable foundation, while the subsequent LoRA fine-tuning phase substantially reduces transmission overhead. This framework demonstrates strong practicality in heterogeneous federated environments.

The overall experimental results indicate that the proposed Adaptive LoRA FL simultaneously achieves highly efficient communication (39.26 MB), rapid convergence (terminating at an average of 89 global rounds with a training time of 3927.8 s across five independent runs), and superior stability and accuracy (98.58%).

G. *Comparative Analysis of Adaptive LoRA FL against FedAvg, FedProx, and FedNova*

To further validate the empirical performance of the proposed Adaptive LoRA FL framework within a heterogeneous environment, a comparative analysis was conducted against three widely adopted aggregation methods: FedAvg, FedProx, and FedNova. These benchmarks represent the prevailing standards in federated learning and are extensively utilized to address challenges associated with Non-IID data distributions.

In this experimental setup, FedAvg, FedProx, and FedNova were implemented using a weight-based federated learning architecture. To ensure a rigorous and consistent comparison, all three baseline methods were integrated with an early stopping mechanism to prevent redundant training rounds.

The experimental results for the global accuracy comparison are illustrated in Fig. 19. All three FL baseline methods demonstrate stable and high accuracy performance within the heterogeneous data environment. Based on the average global accuracy derived from five experimental trials, FedAvg Weight FL achieved approximately 99.09%, FedProx Weight FL reached 98.94%, and FedNova Weight FL recorded 98.90%. These results indicate that weight-based FL architectures maintain robust classification efficacy even under Non-IID conditions.

In comparison, the proposed Adaptive LoRA FL achieved an average global accuracy of approximately 98.58% under identical early stopping conditions. Although this performance is marginally lower than that of the three FL baseline methods, the overall accuracy remains at a high level, nearly reaching the 99% threshold.

As the primary design objective of Adaptive LoRA FL is not solely the maximization of absolute accuracy, the framework focuses on achieving a critical balance between high predictive performance and the substantial reduction of communication and overall training overheads. Unlike FedAvg, FedProx, and FedNova, which necessitate the transmission and aggregation of full model weights in every global round, the proposed Adaptive LoRA FL framework optimizes resource utilization through its Adaptive LoRA mechanism.

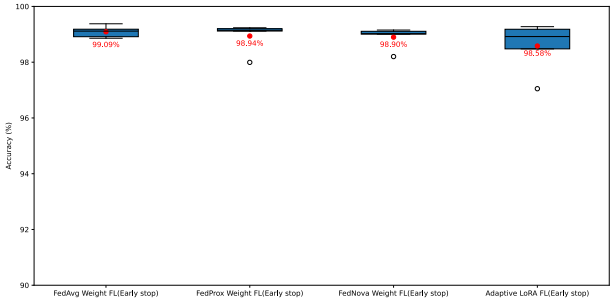

Fig. 19. Global accuracy comparison under early stopping.

Figs. 20 and 21 present a comparative analysis of total training time and communication costs between the three baseline FL methods and the proposed Adaptive LoRA FL, all operating under identical early stopping conditions.

As illustrated in Fig. 20, the three baseline FL methods exhibit comparable performance in terms of resource consumption. Specifically, the average training times for FedAvg, FedProx, and FedNova across five experimental trials were approximately 5765.6 s, 5553.2 s, and 5226.8 s, respectively. These results indicate that weight-based FL architectures incur substantial training time in heterogeneous data environments, even when integrated with early stopping mechanisms.

In contrast, the proposed Adaptive LoRA FL framework demonstrates a superior convergence efficiency. The average total training time for the proposed framework was recorded at approximately 3927.8 s, representing a significant reduction compared to the three baseline FL methods. These findings suggest that the integration of LoRA fine-tuning and layer-freezing strategies effectively accelerates model convergence and optimizes overall training duration within the federated learning environment.

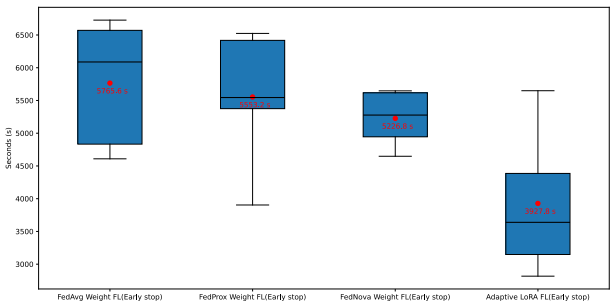

Fig. 20. Total training time comparison under early stopping.

Regarding communication overhead, the experimental results presented in Fig. 21 demonstrate a significant reduction in data transmission for the proposed framework. The average communication costs for FedAvg, FedProx, and FedNova were approximately 146.89 MB, 138.55 MB, and 127.80 MB, respectively. In contrast, the average communication cost for Adaptive LoRA FL was recorded at only 39.26 MB.

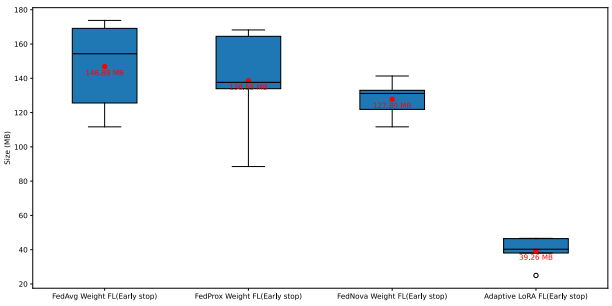

Fig. 21. Total communication cost comparison under early stopping.

These findings indicate that the transition to the LoRA fine-tuning phase, which restricts data exchange to low-rank parameters drastically reduces the per-round communication cost. Consequently, Adaptive LoRA FL maintains high global accuracy while substantially lowering communication overhead.

H. Evaluation of Generalization Capabilities in Cross-Domain Datasets

To evaluate the generalization capability and stability of the proposed Adaptive LoRA FL framework across diverse application domains, this study incorporated the USTC-TFC2016 dataset [24]. Compared to the previously utilized intrusion detection datasets, USTC-TFC2016 features distinct traffic behaviors and feature distributions. The inclusion of this dataset enables a rigorous simulation of real-world scenarios within a cross-domain, heterogeneous federated learning environment.

As illustrated in Fig. 22, the Adaptive LoRA FL initially performed full-parameter training across all clients. The transition to the LoRA fine-tuning mechanism was triggered at the 81st global round upon satisfying the activation criteria. Experimental results indicate that while the global accuracy exhibited a steady and consistent improvement following the activation of LoRA, the early stopping condition was not met within the specified threshold. Consequently, the training process proceeded until the completion of 300 global rounds, achieving a final global accuracy of 97.62%.

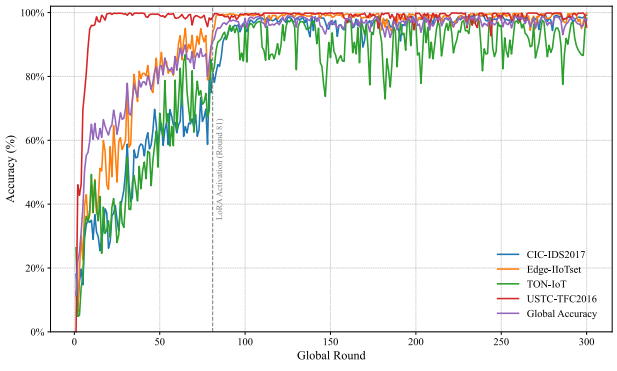

Fig. 22. Global accuracy of Adaptive LoRA FL on the cross-domain USTC-TFC2016 dataset.

V. Conclusion

In conclusion, this study proposes the Adaptive LoRA FL framework to mitigate communication overhead and training instability in heterogeneous federated learning. Evaluations on the CIC-IDS2017, Edge-IIoTset, and TON-IoT datasets demonstrate that the framework achieves an optimized trade-off between communication efficiency and convergence stability.

By integrating LoRA with layer freezing strategies, the proposed Adaptive LoRA FL framework significantly reduces data transmission costs. Specifically, by freezing appropriate hidden layers and employing LoRA to replace full weight updates, the framework minimizes the amount of parameters transmitted. Additionally, an early stopping mechanism avoids redundant iterations, substantially reducing training time while maintaining global model performance. Compared against FedAvg, FedProx, and FedNova benchmarks, the Adaptive LoRA FL framework demonstrates robustness even under highly Non-IID data environments. Overall, this research establishes an efficient and scalable paradigm for federated learning in complex IIoT scenarios.

Conflict of Interest

The authors declare no conflict of interest.

Author Contributions

LHC, THL, and YSH were responsible for the conception and design of this study; LHC and YSH performed the system setup, experimental execution, data preprocessing, and analysis, and drafted the initial manuscript; THL reviewed and edited the manuscript, managed the project timeline, and contributed to data interpretation; LHC and THL jointly supervised the research process and secured the funding; all authors had approved the final version.

Funding

This work was supported by the National Science and Technology Council, Taiwan under Grant No. MOST 114-2221-E-142-001-MY3 and 113-2221-E-142-005-MY2.

References

[1] H. B. McMahan, E. Moore, D. Ramage *et al.,* "Communication-efficient learning of deep networks from decentralized data," in *Proc. 20th Int. Conf. Artificial Intelligence and Statistics (AISTATS)*, Fort Lauderdale, 2017, pp. 1273–1282.

[2] E. J. Hu, Y. Shen, P. Wallis *et al.*, "LoRA: Low-rank adaptation of large language models," arXiv Preprint, arXiv: 2106.09685, 2022.

[3] I. Sharafaldin, A. H. Lashkari, and A. A. Ghorbani, "Toward generating a new intrusion detection dataset and intrusion traffic characterization," in *Proc. 4th Int. Conf. Inf. Syst. Security and Privacy (ICISSP)*, Funchal, 2018.

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

[5] T. M. Booij, I. Chiscop, E. Meeuwissen *et al.,* "ToN_IoT: The role of heterogeneity and the need for standardization of features and attack types in IoT networkintrusion datasets," *IEEE Internet Things J.*, vol. 9, no. 1, pp. 485–496, 2022.

[6] T. Li, A. K. Sahu, M. Zaheer *et al.,* "Federated optimization in heterogeneous networks," in *Proc. Mach. Learn. Syst. (MLSys)*, 2020, vol. 2, pp. 429–450.

[7] J. Wang, Q. Liu, H. Liang *et al.,* "Tackling the objective inconsistency problem in heterogeneous federated optimization,"

Adv. Neural Inf. Process. Syst. (NeurIPS), vol. 33, pp. 7611–7623, 2020.

[8] S. P. Karimireddy, S. Kale, M. Mohri *et al.,* "SCAFFOLD: Stochastic controlled averaging for federated learning," in *Proc. 37th Int. Conf. Mach. Learn. (ICML)*, 2020, vol. 119, pp. 5132–5143.

[9] Q. Yang, Y. Liu, T. Chen *et al.,* "Federated machine learning: Concept and applications," *ACM Trans. Intell. Syst. Technol.*, vol. 10, no. 2, pp. 1–19, 2019.

[10] P. Kairouz, H. B. McMahan, B. Avent *et al.*, "Advances and open problems in federated learning," *Found. Trends Mach. Learn.*, vol. 14, no. 1–2, pp. 1–210, 2021.

[11] F. Haddadpour, M. M. Kamani, A. Mokhtari *et al.,* "Federated learning with compression: Unified analysis and sharp guarantees," in *Proc. 24th Int. Conf. Artif. Intell. Stat. (AISTATS)*, San Diego, 2021, pp. 2350–2358.

[12] E. Malan, V. Peluso, A. Calimera *et al.,* "Automatic layer freezing for communication efficiency in cross-device federated learning," *IEEE Internet Things J.*, vol. 11, no. 4, pp. 6072–6083, 2024.

[13] T. Dettmers, A. Pagnoni, A. Holtzman *et al.,* "QLoRA: Efficient finetuning of quantized LLMs," in *Proc. 37th Int. Conf. Neural Inf. Process. Syst. (NeurIPS)*, Red Hook, 2023, pp. 10088–10115.

[14] Y. Xu, L. Xie, X. Gu *et al.,* "QA-LoRA: Quantization-aware low-rank adaptation of large language models," arXiv Preprint, arXiv:2309.14717, 2023.

[15] Z. Liu, J. Lyn, W. Zhu *et al.,* "ALoRA: Allocating low-rank adaptation for fine-tuning large language models," arXiv Preprint, arXiv:2403.16187, 2024.

[16] Z. Han, C. Gao, J. Liu *et al.,* "Parameter-efficient fine-tuning for large models: A comprehensive survey," arXiv Preprint, arXiv:2403.14608, 2024.

[17] Z. Hong, J. Xiong, H. Yang *et al.,* "Lightweight low-rank adaptation vision transformer framework for cervical cancer detection and cervix typclassification," *Bioengineering*, vol. 11, no. 5, 468, 2024. https://doi.org/10.3390/bioengineering11050468

[18] L. Yi, H. Yu, G. Wang *et al.,* "pFedLoRA: Model-heterogeneous personalized federated learning with LoRA tuning," arXiv Preprint, arXiv:2310.13283, 2024.

[19] Z. Wang, Z. Shen, Y. He *et al.,* "FLoRA: Federated fine-tuning large language models with heterogeneous low-rank adaptations," in *Proc. 38th Int. Conf. Neural Inf. Process. Syst. (NeurIPS)*, Red Hook, 2025, vol. 37, pp. 22513–22533.

[20] H. Matsutani, M. Kondo, K. Sunaga *et al.,* "Skip2-LoRA: A lightweight on-device DNN fine-tuning method for low-cost edge devices," in *Proc. 30th Asia South Pac. Design Autom. Conf. (ASPDAC)*, New York, 2025, pp. 51–57.

[21] B. Xu, S. Yan, S. Li *et al.,* "A federated transfer learning framework based on heterogeneous domain adaptation for students' grades classification," *Applied Sciences*, vol. 12, no. 21, 10711, 2022.

[22] M. Li, L. Luo, K. Xiao *et al.,* "Adaptive semi-supervised algorithm for intrusion detection and unknown attack identification," *Applied Sciences*, vol. 15, no. 4, 1709, 2025.

[23] X. Zhao, K. W. Fok, and V. L. L. Thing, "Enhancing network intrusion detection performance using generative adversarial networks," *Computers & Security*, vol. 145, 104005, 2024.

[24] W. Wang, M. Zhu, X. Zeng *et al.,* "Malware traffic classification using convolutional neural network for representation learning," in *Proc. 2017 International Conf. on Information Networking (ICOIN),* 2017, pp. 712–717.