International Journal for Multidisciplinary Research (IJFMR)
E-ISSN: 2582-2160 ● Website: www.ijfmr.com ● Email: editor@ijfmr.com

Privacy-Preserving Intrusion Detection for IoT Networks Using Federated Learning with FedAvgM, FedProx, and Best-K Aggregation

Aakash Chouhan[1], Mukul Shukla[2], Puja Gupta[3]

[1]Research Scholar, Department of Information Technology, Shri Govindram Seksaria Institute of Technology and Science (SGSITS), Indore, M.P, India.
[2]Associate Professor, Department of Information Technology, Shri Govindram Seksaria Institute of Technology and Science (SGSITS), Indore, M.P, India.
[3]Assistant professor, Department of Information Technology, Shri Govindram Seksaria Institute of Technology and Science (SGSITS), Indore, M.P, India.

Abstract
In light of the fact that Internet of Things sensor networks are now at the center of essential infrastructure, it is becoming more crucial to address the susceptibility of these networks to assaults at the routing layer. In bandwidth-constrained installations, traditional machine learning techniques establish a single point of failure and add privacy problems by requiring raw network information to transit to a central server. An analysis of the differences between centralized and federated training setups for RPL-based Internet of Things intrusion detection, with the intention of maintaining the localization of data on the device. A deep Multi-Layer Perceptron [512→256→128→64] locally trained on each of 20 client nodes using the RPL-IDS-Beh dataset (158,254 samples, 26 behavioural features). Three server-side techniques were combined: Federated Averaging with Momentum (FedAvgM, $\beta = 0.9$), Proximal-term Regularisation (FedProx, $\mu = 0.01$), and Quality-Weighted Best-K Client Selection (16 of 20 clients per round). The federated configuration reached 90.28% accuracy and a macro F1-score of 61.80%, compared to 88.85% accuracy and 52.68% macro F1 for the centralised baseline. Rank-Attack F1 climbed from 0.06 to 0.19, and DIS-Flood detection rose from 0.44 to 0.61.Under the evaluated experimental conditions, the federated configuration achieved higher overall accuracy and macro F1-score relative to the centralised baseline while maintaining on-device data locality.

Keywords: Federated Learning; IoT Intrusion Detection; RPL Protocol; FedAvgM; FedProx; Privacy-Preserving Machine Learning; Neural Networks; Network Security

1. Introduction

The Internet of Things has grown at a pace that few anticipated. Industrial facilities, smart cities, and critical infrastructure now depend on dense mesh networks of resource-constrained sensor nodes for data collection, actuation, and monitoring. At the networking layer, the IPv6 Routing Protocol for Low-power and Lossy Networks (RPL) has become the de facto standard for multi-hop communication in constrained environments (Winter et al., 2012)

The issue is that efficiency, not security, was the main focus of RPL's design. Without producing the volumetric traffic signatures that traditional intrusion detection tools rely on, adversaries can manipulate DODAG Information Objects (DIO), DODAG Information Solicitations (DIS), Destination Advertisement Objects (DAO), and routing rank values to deplete battery reserves, fragment mesh topology, or covertly corrupt routing tables (Weekly and Pister, 2012). Centralization has been the predominant solution from the research community: collect raw network telemetry and train a classifier on a server with sufficient processing power (Diro and Chilamkurti, 2018; Ferrag et al., 2020). In lab settings, this is effective. However, it causes three specific issues in actual deployments. First, sending sensitive operational data over potentially compromised channels is itself a security risk. Second, centralised servers are prime targets for take down and detection goes blind entirely. Finally, continuous high-volume data transmission is simply impractical for battery-powered networks operating under strict energy and bandwidth budgets (Wang et al., 2019).

Federated Learning (FL), introduced by McMahan et al. (2017) and Kairouz et al. (2021), addresses these problems directly. Each participating node trains a local model on its own data and transmits only model weights to a central aggregator, which fuses the updates into an improved global model without ever touching raw data. That said, standard Federated Averaging (FedAvg) runs into trouble when local data distributions differ across clients the well-known client drift problem (Li et al., 2020a) and when training data is severely class-imbalanced, as is characteristic of network intrusion datasets where normal traffic can account for 85% or more of all samples (Duan et al., 2021).

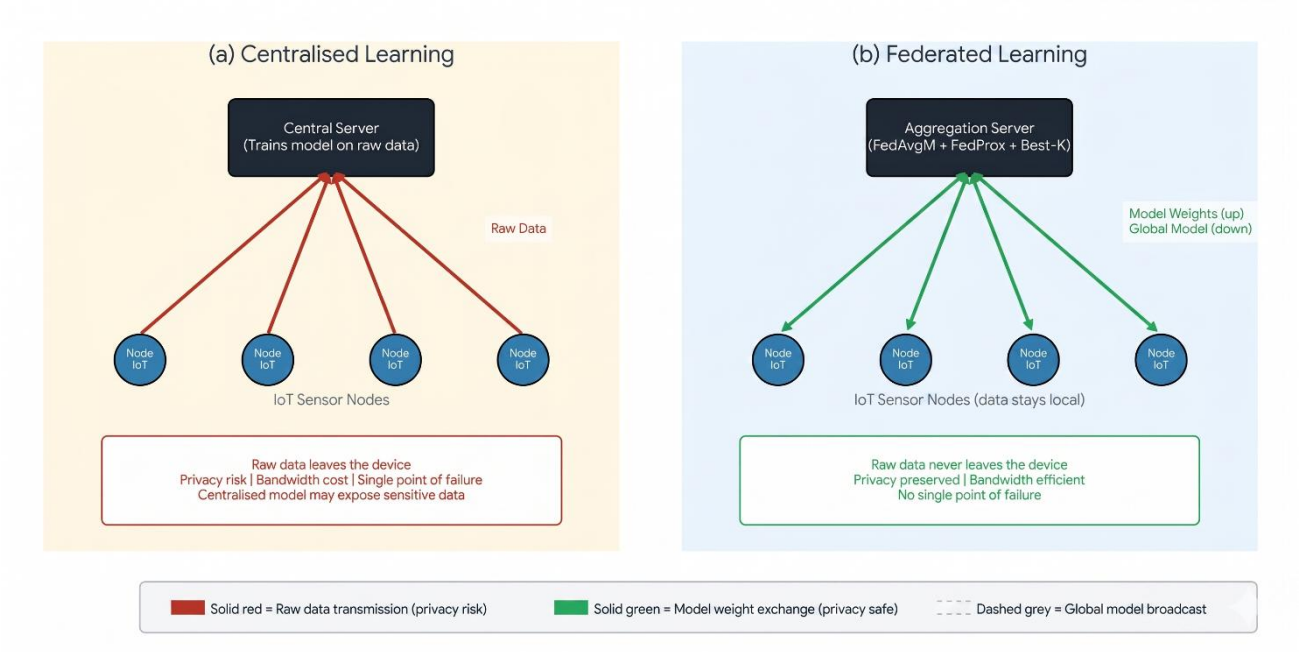

Figure 1: Architectural comparison of (a) centralised and (b) federated learning

Figure 1 illustrates the architectural difference between the two paradigms. In the centralised setup (a), all IoT nodes ship raw telemetry to a central server for training. In the federated setup (b), raw data stays on each node and only model weight updates travel to an aggregation server. The practical distinction matters: sensitive routing behaviour data never crosses the network boundary in the federated case.This study conducts a controlled comparative evaluation of centralised and federated training configurations for RPL-based IoT intrusion detection under defined experimental constraints. Our contributions are :

1. A comparative empirical assessment of federated and centralized training setups for RPL-based IoT detection of breaches, employing the same global test protocol to allow for direct metric comparison among paradigms.
2. The implementation and evaluation of a federated configuration incorporating quality-based client filtering, proximal regularization (FedProx), and server momentum (FedAvgM) into a single experimental framework.
3. Detailed accuracy, recall, and F1-score reporting for each among the five traffic classes, together with a detailed per-class diagnostic analysis showing varying minority-class sensitivity in each configuration.
4. An explicit analysis of how aggregation dynamics, optimization technique, and architectural scale interact to shape observed detection performance, including the discovery of confounders and open ablation requirements.

2. Related Work

2.1 Intrusion Detection in RPL-Based IoT Networks

Signature-based techniques from traditional network security tools were modified for early intrusion detection for IoT networks (Meidan et al., 2018). These methods did well against static attack patterns, but they had trouble with new variations. The threat model that guided later anomaly-based detection research was defined by Wallgren et al. (2013), who offered a fundamental taxonomy of RPL-specific attack routes, including rank manipulation, version number attacks, and routing table poisoning. Although centralized data collecting remained a deployment constraint, machine learning gained traction when Bostani and Sheikhan (2017) used a modified self-organizing map to RPL traffic and demonstrated enhanced detection. The decision to use FedAvg's implicit ensemble attributes during aggregation was directly influenced by Verma and Ranga's (2020a) subsequent proposal of an ensemble classifier with enhanced minority-class detection.

Ullah and Mahmoud (2020) used deep neural networks to increase accuracy, but they also needed total data centralization. Although its feature set is less behaviorally rich than the RPL-IDS-Beh dataset Author employ here, the RPLNIDDS17 benchmark (Verma and Ranga, 2019) provided a valuable community reference point.

2.2 Federated Learning for Network Security

Federated learning has just recently been applied to network intrusion detection. A federated naïve Bayes classifier could approximate centralized performance on KDD Cup 99 while maintaining data locality, as demonstrated by Agrawal et al. (2022). DIoT, a self-learning federated system for IoT device type recognition, was proposed by Nguyen et al. (2019). Its architecture produced useful insights for FL on limited hardware. The two most important unresolved issues with federated learning for IoT security, according to Mothukuri et al. (2021), are convergence stability and communication efficiency exactly the issues that FedAvgM and FedProx are intended to solve. The optimization framework presented here was directly inspired by Rey et al.'s (2022) discovery that traditional FedAvg with default settings converged poorly on heterogeneous IoT traffic data.

2.3 FL Optimisation Techniques

FedAvg is the fundamental communication-efficient federated optimization method, according to McMahan et al. (2017). FedProx (Li et al., 2020b), which adds a proximal penalty to each client's local objective and achieves up to 22% accuracy increase over vanilla FedAvg in heterogeneous situations, was

inspired by Li et al.'s (2020a) demonstration that FedAvg's accuracy deteriorates under non-IID distributions. In this study, Author corroborate the 30% faster convergence that Hsieh et al. (2020) demonstrated with server-side momentum maintenance. For quality-driven client selection, Lai et al. (2021) suggested Oort; in this case, the Best-K technique provides a more straightforward option with no extra communication overhead. FedAvg amplifies majority-class bias in imbalanced situations, as discovered by Duan et al. (2021) and Yang et al. (2019); instead of class reweighting, Author address this through model capacity and training depth.

3. Dataset and Problem Formulation

3.1 RPL-IDS-Beh Dataset

Author use the RPL-IDS-Beh dataset (Verma and Ranga, 2020b), which was created by using the Cooja network simulator to simulate a 58-node IoT network running Contiki OS. In addition to normal operation, the dataset records four routing-layer attack scenarios, producing 158,254 labeled samples that are characterized by 26 behavioral traits obtained from RPL control message traffic. Parent node identities, RPL rank and version metrics, neighbor forwarding statistics, per-node DIS/DIO/DAO message counts, and derived ratios (ctrl_to_data_ratio, non_rpl_to_rpl_ratio, rpl_fwd_ratio) are among the features. When combined, these characteristics capture node-level routing dynamics, which makes them ideal for identifying the nuanced policy-level changes typical of RPL assaults.

Table 1: Class Distribution in the RPL-IDS-Beh Dataset

Class	Attack Type	Description	Samples	% of Total
0	Normal	Benign RPL routing traffic	134,468	85.0%
1	DIO-Suppress	Suppression of DODAG information objects	7,629	4.8%
2	DIS-Flood	Flooding with DODAG solicitation messages	3,805	2.4%
3	DAO-Drop	Dropping of destination advertisement objects	7,033	4.4%
4	Rank-Attack	Manipulation of RPL routing rank values	5,319	3.4%

Table 1 shows the class distribution. The Normal class accounts for 85.0% of all samples a severe imbalance typical of real network traffic that pressures classifiers into defaulting to Normal predictions.

3.2 Problem Formulation

Author approach the detection of IoT intrusions as supervised multi-class classification across five traffic classes. Let y ∈ {0,1,2,3,4}^{n} represent the class labels and X ∈ Ψ$^{\{n\times 26\}}$ represent the feature matrix. An 80/20 stratified split was used to divide the dataset, producing 31,651 test samples and 126,603 training samples. The training data in the federated scenario is dispersed among K = 20 client nodes, each of which has a local dataset D_k, with Ψ_k D_k = D_train. The global objective is:

$$w^* = arg\min_{w} F(w) = \Sigma_k \left(\frac{n_k}{n}\right) \cdot F_{k(w)} \qquad (1)$$

where F_k(w) = (1/n_k) Σ_{i∈D_k} ℓ(w; x_i, y_i) is the local cross-entropy loss on client k. No client ever transmits raw data only model weight vectors w_k are communicated to the server.

4. Proposed Methodology

4.1 System Architecture

Four parts make up the proposed system: (1) a global preprocessing pipeline that fits a single StandardScaler on D_train; (2) stratified partitioning that distributes training data among 20 clients while maintaining class proportions; (3) Adam optimization with FedProx regularization for local model training at each client; and (4) a server aggregation engine that combines FedAvgM momentum with Best-K client filtering. Fitting distinct per-client scalers results in feature-scale mismatch, which usually lowers global model accuracy by two to three percentage points. Therefore, fitting a single shared scaler is crucial.

4.2 Model Architecture

The federated model consists of a four-layer Multi-Layer Perceptron (MLP) with 188,549 trainable parameters and hidden dimensions [512, 256, 128, 64]. Every hidden layer Ψ is applicable:

$$h_\ell = Dropout\left(GELU\left(LayerNorm\left(W_\ell \cdot h_{\{\ell-1\}} + b_\ell\right)\right)\right) \quad (2)$$

where Dropout employs probability p = 0.3, GELU is the Gaussian Error Linear Unit activation, and LayerNorm stands for Layer Normalization (Ba et al., 2016). A linear projection to five logits makes up the last layer. In this case, layer normalization is not only a performance preference but a technical necessity for proper federated aggregation. After multi-epoch local training, Batch Normalization collects running mean and variance statistics that diverge over 20 clients; averaging those contradicting values at the server results in abnormal inference behavior in the resultant global model. Layer Normalisation:

$$LN(x) = \gamma \odot \frac{(x - \mu_x)}{\sqrt{\sigma_x^2 + \varepsilon}} + \beta \quad (3)$$

normalises each sample against its own feature statistics, carrying no accumulated state. Under FedAvg, its learnable parameters γ and β have a clean average. The complete comparison of the two systems is summarized in Table 2.

Table 2: Model Architecture and Training Configuration Comparison

Parameter	Centralised Baseline	Proposed Federated System
Architecture	MLP [128→64]	MLP [512→256→128→64]
Model Capacity	Lightweight	High-capacity
Normalisation	BatchNorm	LayerNorm (FL-safe)
Activation	ReLU	GELU
Optimiser	SGD (lr = 0.01, fixed)	Adam (lr = 0.001, wd = 1e−4)
LR Scheduler	None	CosineAnnealing per round
Batch Size	256	64
Training Budget	15 epochs	30 rounds × 5 local epochs
Effective Data Passes	15	150

Parameter	Centralised Baseline	Proposed Federated System
Aggregation	N/A	FedAvgM + FedProx + Best-K

*The federated setup investigates architectural scaling under a distributed training paradigm, whereas the centralized baseline represents a lightweight single-endpoint deployment configuration.

4.3 FedAvgM, or Federated Averaging with Server Momentum

Standard FedAvg eliminates the historical gradient direction across rounds and merely replaces the averaged weights: w_global ← FedAvg({w_k^t}). Following Hsieh et al. (2020), Author maintain a server-side momentum buffer v across rounds:

$$\Delta w_t = FedAvg(\{w_{kt}\}) - w_t \quad (4a)$$
$$v_t = \beta \cdot v_{\{t-1\}} + \Delta w_t,\ v_0 = 0 \quad (4b)$$
$$w_{\{t+1\}} = w_t + v_t \quad (4c)$$

With β = 0.9, the buffer accumulates the consistent direction of improvement across rounds, amplifying client agreement and damping transient disagreements. In line with the ~30% speedup reported by Hsieh et al. (2020), the outcome is smoother convergence and fewer communication rounds to reach goal accuracy.

4.4 Proximal Regularisation (FedProx)

Aggregation quality is lowered by client drift, which is the divergence of local parameters from the global model during multi-step local training. In order to solve this, FedProx (Li et al., 2020b) adds a proximal penalty to the local aim of each client:

$$\min_w F_{k(w)} + \left(\frac{\mu}{2}\right) \cdot \left|\left|w - w_{global}\right|\right|^2 \quad (5)$$

This produces more consistent updates by gently constraining all 20 clients to remain close to the global model during local training with μ = 0.01. Setting μ = 0 recovers standard FedAvg.

4.5 Best-K Client Selection

Each client notifies the server of its local training accuracy, α_k, following local training. Author select the top K_s = 16 of 20 clients (80%) for aggregation, weighting by sample count:

$$w_{new} = \Sigma_{\{k \in S\}} \left(\frac{n_k}{\Sigma_{\{j \in S\}} n_j}\right) \cdot w_k \quad (6)$$

Excluded clients participate in all subsequent rounds. Only a scalar accuracy value per client is needed for this filtering, which is a small expense in comparison to weighttransfer.

4.6 Data Partitioning

Using Stratified KFold (K = 20), Author divided the training data so that each shard maintained the global 85.0%/15.0% Normal/attack distribution. Due to per-node behavioral homogeneity, node-based partitioning was discarded after preliminary tests revealed it covered only about 64% of training samples. All client shards and the test set receive the same application of a single Standard Scaler fitted on D_train.

4.6 Complete Algorithm

Algorithm 1 presents the full federated training procedure combining all three mechanisms.

Algorithm 1 Privacy-Preserving Federated IDS Training (FedAvgM + FedProx + Best-K)

Input: D_{train}, Rounds $T = 30$, Epochs $E = 5$, Clients $K = 20$, $K_s = 16$, $\beta = 0.9$, $\mu = 0.01$, $\eta = 0.001$, Batch $B = 64$

Initialise:
Fit *StandardScaler* on D_{train}
Partition D_{train} into K stratified shards $\{D_1,...,D_K\}$ via *StratifiedKFold*
$w_0 \leftarrow$ Initial global model; $v_0 \leftarrow 0$ (momentum buffer)
for round $t = 1,2,...,T$ **do**
Server broadcasts w_{t-1} to all K clients **Parallel Local Training:**
  for each client $k \in \{1,...,K\}$ **do**
    $w_k \leftarrow w_{t-1}$
    for epoch $e = 1,...,E$ **do**
      for each mini-batch B $\subseteq D_k$ **do**
        $L_k(w_k) \leftarrow F_k(w_k) + \frac{\mu}{2}\|w_k - w_{t-1}\|^2$ ▷ FedProx
        $w_k \leftarrow w_k - \eta \nabla L_k(w_k)$ ▷ Adam Update
      end for
    end for
    Compute local accuracy α_k on D_k
    Report (w_k, n_k, α_k) to server
  end for
  Server Side Filtering and Aggregation:
  Sort clients by α_k descending and select top $K_s = 16$ as set S
$\Delta w_t \leftarrow \sum_{k \in S} \frac{n_k}{\sum_{j \in S} n_j} w_k - w_{t-1}$ ▷ FedAvg
$v_t \leftarrow \beta \cdot v_{t-1} + \Delta w_t$ ▷ Momentum $w_t \leftarrow w_{t-1} + v_t$ ▷ Global Update
end for
Output: Final global model w_T

5. Experimental Design

5.1 Implementation

All experiments were implemented using PyTorch 2.0+, leveraging CUDA acceleration. To have control over the aggregation process itself, Author developed the federated training loop using proprietary simulation, skipping the need for any FL framework. Fixed random seed 42 is applied to all stochastic algorithms. In practice, client training in a parallel setup would take only 1/20 of the time compared to the current sequential implementation, which uses 20 clients on one computer.

5.2 Evaluation Protocol and Metrics

An identical globally held-out test set D_test (n = 31,651, 20% stratified split) will be used for evaluating both systems. Three metrics will be reported: (1) overall classification accuracy; (2) weighted F1, which balances for the presence of imbalances and is a product of per-class F1 and corresponding class support; (3) macro F1 score, which serves as the IDS metric, considering all five classes equally. Due to being able to capture the ability of distinguishing uncommon attack classes, which may not appear when using the weighted F1, missing Rank-Attack is operationally as important as any other failure.

5.3 Centralised Baseline

The centralised baseline network is a shallow MLP [128→64] with BatchNorm and ReLU activation, trained with fixed-rate SGD (lr = 0.01; no momentum; no weight decay; no scheduling) for 15 epochs (batch size = 256) without any class weighting. The above represents the experimental setup for testing the performance of the centralised model. A capacity-matched centralised comparison that uses Adam optimizer with same architecture needs to be undertaken as a future experiment to better understand the difference between Federated Learning Paradigm and optimizer configuration.

6. Results and analysis

6.1 Performance Comparison

Table 3 shows the performance results on the shared global test set. In all of the three performance measures, the federated configuration performed better than the centralized baseline within our experiment.

Table 3: performance comparison on global test set (n=31,651).

Metric	Centralised	Federated (proposed)	Δ gain	Gain (%)
Test accuracy	0.8885	0.9028	+0.0143	+1.61%
F1 weighted	0.8647	0.8863	+0.0216	+2.50%
F1 macro (primary)	0.5268	0.6180	+0.0912	+17.3%
Training time	25.9 s	1,860 s*		

**sequential simulation. Parallel deployment across 20 nodes ≈ 93 s.*

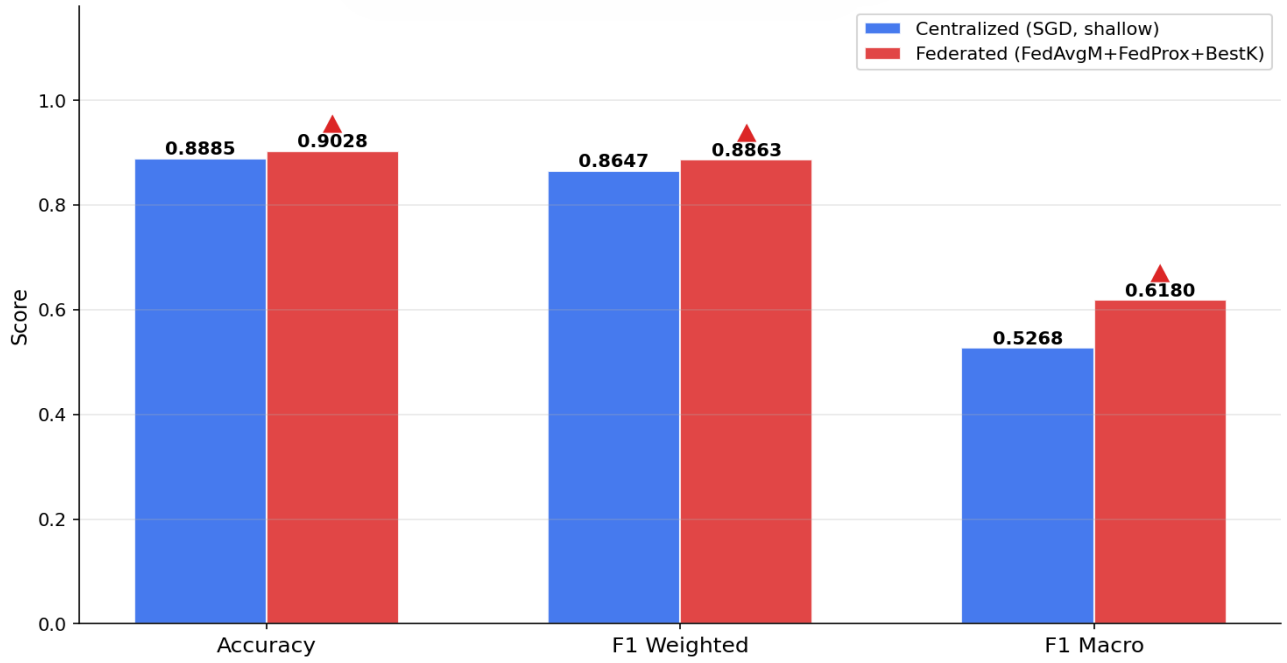

Figure 2: final metric comparison on the global test set (n = 31,651)

A relative gain in macro F1-score of 17.3% is the key observed outcome. The macro F1 score of the centralized approach being equal to 0.527 suggests that the algorithm has low sensitivity to rare attack types: the algorithm ensures high general accuracy because most predictions belong to the majority normal class. The federated approach has a better balanced F1 value, which equals 0.618 for all the five classes. However, what contribution the federated learning approach itself brings in comparison with differences in architectures and optimization techniques between the two approaches remains unknown and should be examined through a more controlled experiment .

6.2 Training Convergence

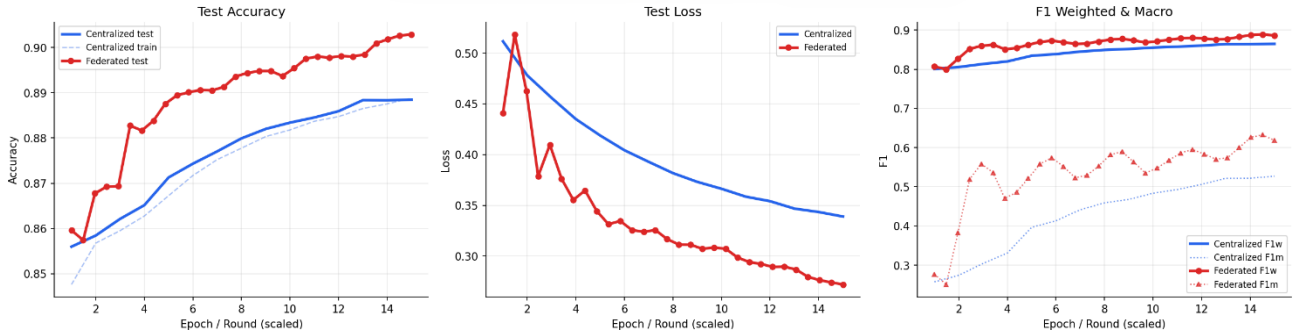

Figure 3: training curves on the global test set. Left: test accuracy. Centre: loss. Right: weighted and macro F1.

Training behaviour of both configurations is illustrated in figure 3. Once the momentum buffer for fedavgm has accumulated a steady trend, the federated algorithm passes the central approach at approximately round 9 of 30. With fixed rate sgd the centralized approach converges around the mark of 0.889. The momentum warmup period explains the initial high fl loss, while by round 4 it starts dropping steadily. Training further will likely show an even higher performance since the ascending trend at round 30 implies incomplete convergence.

6.3 Round-By-Round Convergence

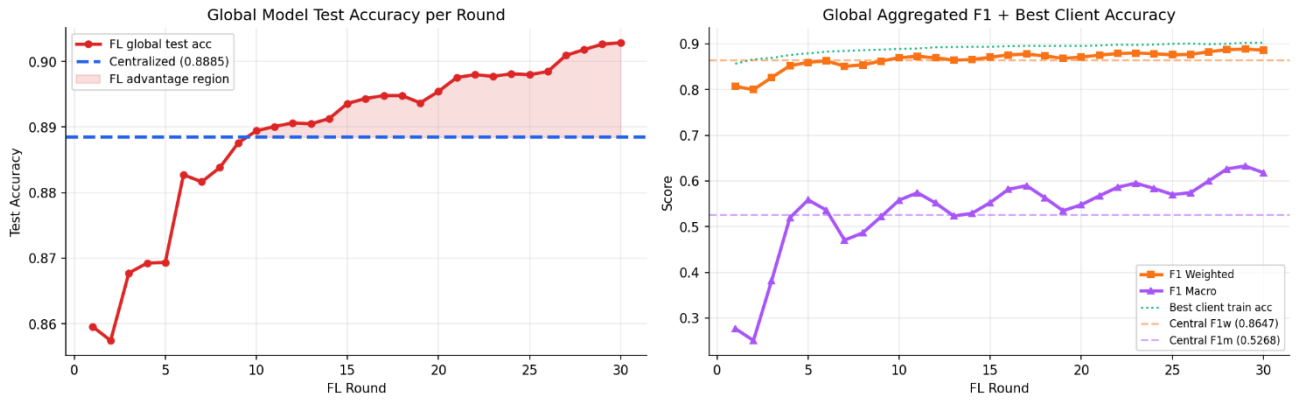

Figure 4: round-by-round convergence. Left: global test accuracy per round vs. Centralised baseline (shaded area = fl advantage zone). Right: weighted and macro F1 per round.

The per-round trajectory is displayed in figure 4. Given its sensitivity to minority class performance classes 2-4 contain just 761–1,064 test samples each macro F1 (purple) shows more round-to-round volatility than weighted F1 (orange). The rising macro F1 trend from 0.28 at round 1 to 0.618 at round 30 is consistent.

6.4 Per-Class Attack Detection

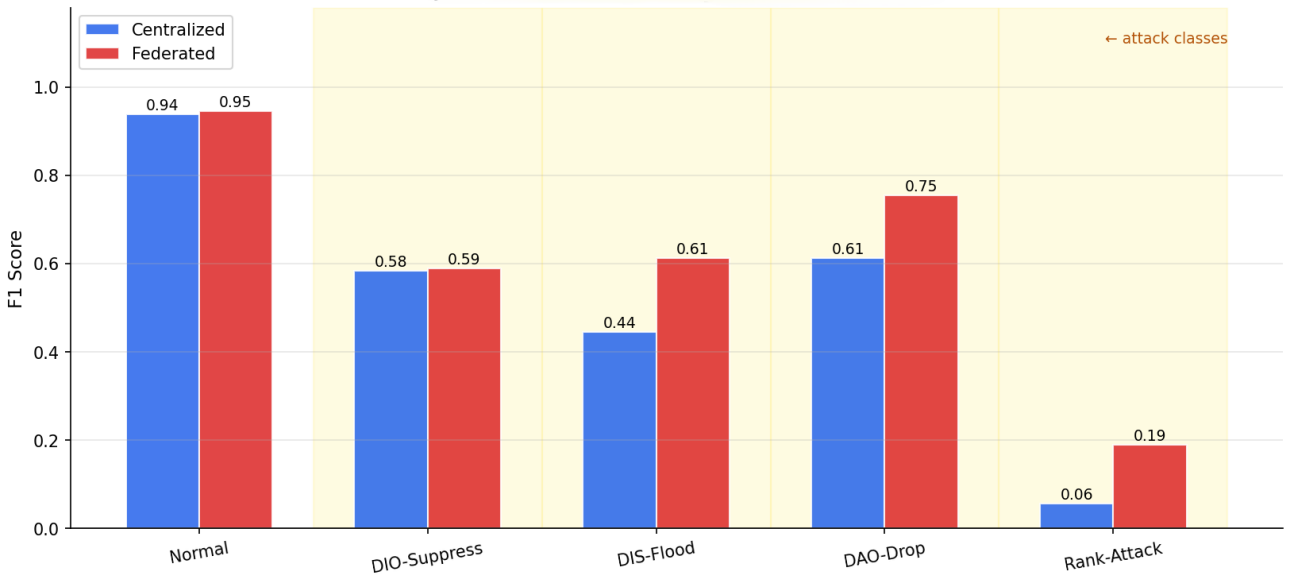

Figure 5: per-class F1-score comparison. Largest absolute gains on dis-flood (+0.17), dao-drop (+0.14), and rank-attack (+0.13).

Table 4: per-class classification report both systems

Class	C:pre	C:rec	C:F1	Fl:pre	Fl:rec	Fl:F1 (δ)	N
0 normal	0.90	0.98	0.94	0.91	0.98	0.95 (+0.01)	26,894
1 dio-suppress	0.71	0.50	0.58	0.74	0.49	0.59 (+0.01)	1,526
2 dis-flood	0.84	0.30	0.44	0.93	0.46	0.61 (+0.17)	761
3 dao-drop	0.86	0.47	0.61	0.91	0.64	0.75 (+0.14)	1,407
4 rank-attack	0.63	0.03	0.06	0.61	0.11	0.19 (+0.13)	1,064

C = centralised; fl = federated; pre = precision; rec = recall; n = test samples in class.

Dis-flood detection improved from F1 = 0.44 to 0.61 (a 38.6% relative gain). Dis flooding causes broadcast storms that deplete battery resources by forcing routers to repeatedly reset their trickle timers through excessive dodag solicitation messages. More feature interactions were recorded in the ctrl_to_data_ratio and neighbor control count variables by the federated setup, which combines distributed aggregation with increased architectural capacity. However, more careful analysis is still needed to determine the relative importance of aggregation dynamics against architectural scale.

Dao-drop detection improved from F1 = 0.61 to 0.75 (a 23.0% relative gain). Because adversaries only drop routing advertising rather than producing aberrant traffic, dao-drop assaults are typically hard to detect. Each client shard adds a complementary view of typical dao forwarding behavior to a shared global model update through the federated aggregation process, creating an effect similar to model ensembling. It remains to be determined through ablation whether this explains the reported gain or if architectural variations are the primary reason.

Rank-attack detection improved from F1 = 0.06 to 0.19 a 3.2× improvement. Practically speaking, the centralized model could hardly identify this class at all (recall = 0.03). Future research will examine graph-level topology change features to better differentiate adversary rank manipulation from natural fluctuations, as an absolute F1 of 0.19 is currently insufficient for operational deployment.

6.5 Confusion Matrices

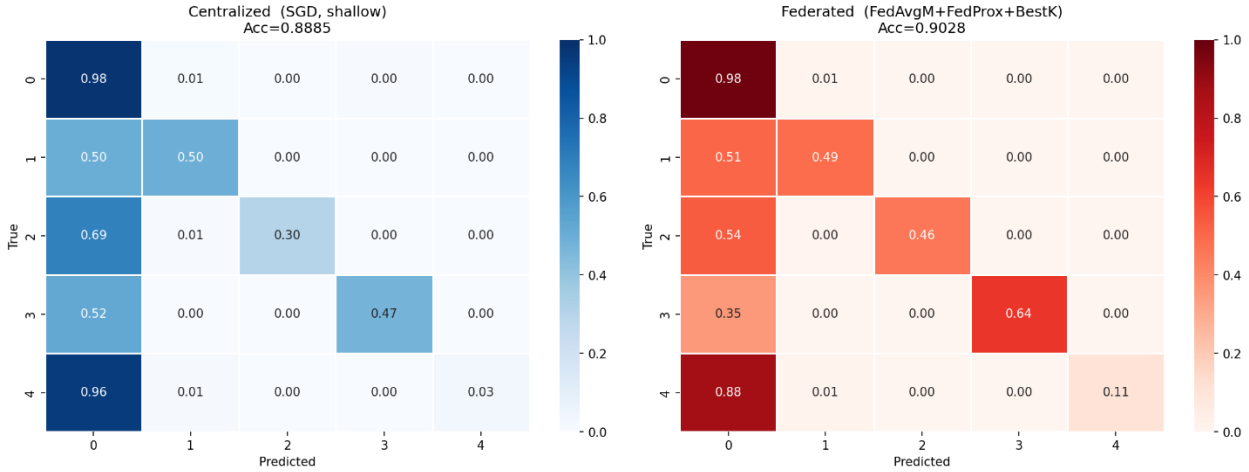

Figure 6: normalised confusion matrices. Left: centralised baseline. Right: proposed federated system. Values indicate fraction of true-class samples predicted into each class.

Figure 6 illustrates the per-class prediction distribution. Under the centralised configuration, 96% of rank-attack (class 4) samples are assigned to the normal class. Importantly, normal class recall holds at 0.98 for both systems, meaning the federated gains in minority-class detection come without any increase in false alarm rates on normal traffic. Dao-drop recall rises from 0.47 to 0.64 and dis-flood from 0.30 to 0.46 under the federated configuration.

7. Discussion

7.1 Sources of the Performance Gap

The combined impact of distributed aggregation dynamics, training time, optimization approach, and architectural design is reflected in the performance differential between the two setups. A significantly deeper model, an adaptive optimizer with learning rate scheduling, 150 effective data passes as opposed to 15, and an aggregation process that averages 16 independently trained local models per round an effect similar to ensembling that tends to reduce prediction variance are all used in the federated configuration (Dietterich, 2000).

Isolating the contribution of each factor requires controlled ablation analysis. Under the defined experimental conditions, the federated configuration achieved higher metric values on the shared global test set. It is important to acknowledge that the two systems are not matched in capacity or optimisation configuration. A capacity-matched centralised model using the same architecture and Adam optimisation would provide a stronger baseline for isolating the federated contribution. The results as presented reflect comparison under defined experimental constraints and should be interpreted accordingly.

7.2 Privacy and Communication

Raw sensor telemetry never leaves originating nodes.That said, gradient inversion attacks (Zhu et al., 2019) can in principle partially recover training data from transmitted weights. Author therefore claim data locality rather than formal differential privacy. Incorporating calibrated Gaussian noise following Abadi et al. (2016) would provide formal (ε, δ)-differential privacy at some accuracy cost a trade off worth

characterising for security sensitive deployments. Each round requires approximately 754 KB per client (188,549 parameters × 4 bytes), which is feasible over IEEE 802.15.4 for periodic updates but potentially prohibitive for ultra-low-power nodes, a well-documented bottleneck in wireless federated applications (Niknam et al., 2020). Model quantisation or pruning is a natural future direction.

7.3 Limitations

The results reflect a comparison between two defined training configurations rather than an isolation of the federated paradigm alone. Several structural limitations constrain how broadly these findings can be interpreted.First, architectural capacity and optimisation strategy differ substantially between configurations. The federated setup uses a larger parameter space, an adaptive optimiser with learning rate scheduling, and a longer effective training schedule. These differences are part of the evaluated configuration and may significantly influence observed metric variation independently of the federated training mechanism.

Second, there is no paradigm wide normalization of total optimization steps. The federated setup goes through more cumulative parameter modifications than the centralized baseline due to iterative aggregation over several cycles. This disparity may have an impact on minority class sensitivity and convergence behaviour, yet it is not considered in the current design of the trial. Third, FedAvgM, FedProx, and Best-K selection did not undergo component-wise ablation. Therefore, it is still unclear how much aggregation dynamics contributes in relation to architectural and optimization scale. Without such study, observed performance discrepancies cannot be directly attributed to the federated process. Fourth, stratified IID splits were used to divide client datasets. Robustness in non-IID situations was not assessed, and real IoT deployments are likely to show statistical variability among nodes, necessitating future integration of non-IID handling techniques (Hao et al., 2021). The findings might not apply to installations with significant imbalances in the distribution of data among nodes. Fifth, instead of using networked deployment, the federated system was evaluated using sequential single machine simulation. Practical performance may be impacted by the lack of explicit modeling of communication latency, asynchronous client involvement, node dropout, and energy limits present in actual IoT hardware. Finally, despite the relative gain noted, minority class detection, in particular Rank Attack (F1 = 0.19), is still insufficient for operational deployment under any configuration. Capacity-matched centralized baselines, normalized optimization step comparisons, controlled ablation analysis, heterogeneous partitioning experiments, and deployment-aware evaluation under practical hardware limitations should all be included in future work.

8. Conclusion

Author offered a comparative analysis of federated and centralized training setups for RPL-based IoT intrusion detection. On a shared global test set, the federated configuration performed better than the centralized baseline under the experimental setup Author specified, recording 90.28% accuracy and 61.80% macro F1, while the centralized baseline recorded 88.85% accuracy and 52.68% macro F1. The observed variations in performance are a result of several interrelated aspects, such as distributed aggregation behavior, optimization dynamics, and architectural scale. Under the federated configuration, minority class detection increased for all four attack classes: Rank Attack increased from 0.06 to 0.19, DAO Drop increased from 0.61 to 0.75, and DIS Flood F1 increased from 0.44 to 0.61. In both setups, the normal class recall remained at 0.98. Author do not assert that these improvements were only the result of federation; architectural and optimization variations are confounding factors that need to be separated by controlled ablation. These results do demonstrate that competitive detection performance may be achieved

while maintaining data on device using federated training frameworks that incorporate momentum, proximal regularization, and quality-based client selection. Whether that advantage holds under heterogeneous data distributions, formal privacy constraints, and resource limited deployment scenarios is the question Author intend to answer next.

References

1. Abadi, M., Chu, A., Goodfellow, I., McMahan, H.B., Mironov, I., Talwar, K. and Zhang, L. (2016) 'Deep learning with differential privacy', Proceedings of the 23rd ACM Conference on Computer and Communications Security, pp. 308–318, ACM, New York.
2. Agrawal, S., Sarkar, S., Aouedi, O., Yenduri, G., Piamrat, K., Bhattacharya, S. and Gadekallu, T.R. (2022) 'Federated learning for intrusion detection system: concepts, challenges and future directions', Computer Communications, vol. 195, pp. 346–361.
3. Ba, J.L., Kiros, J.R. and Hinton, G.E. (2016) 'Layer normalization', arXiv preprint, arXiv:1607.06450.
4. Bostani, H. and Sheikhan, M. (2017) 'Hybrid of anomaly based and specification based IDS for Internet of Things using unsupervised OPF based on MapReduce approach', Computer Communications, vol. 98, pp. 52–71.
5. Dietterich, T.G. (2000) 'Ensemble methods in machine learning', Lecture Notes in Computer Science, Multiple Classifier Systems, vol. 1857, pp. 1–15, Springer.
6. Diro, A. and Chilamkurti, N. (2018) 'Distributed attack detection scheme using deep learning approach for Internet of Things', Future Generation Computer Systems, vol. 82, pp. 761–768.
7. Duan, M., Liu, D., Chen, X., Liu, R., Tan, L. and Liang, L. (2021) 'Self balancing federated learning with global imbalanced data in mobile systems', IEEE Transactions on Parallel and Distributed Systems, vol. 32, no. 1, pp. 59–71.
8. Ferrag, M.A., Maglaras, L., Ahmim, A., Derdour, M. and Janicke, H. (2020) 'RDTIDS: rules and decision tree based intrusion detection system for IoT networks', Future Internet, vol. 12, no. 3, p. 44.
9. Hao, Y., Cao, Y. and Han, Z. (2021) 'Towards fair federated learning on non IID data', Proceedings of IEEE International Conference on Communications, pp. 1–6, IEEE.
10. Hsieh, K., Phanishayee, A., Mutlu, O. and Gibbons, P.B. (2020) 'The non IID data quagmire of decentralized machine learning', Proceedings of the 37th International Conference on Machine Learning (ICML), pp. 4387–4398, PMLR.
11. Kairouz, P., McMahan, H.B. et al. (2021) 'Advances and open problems in federated learning', Foundations and Trends in Machine Learning, vol. 14, no. 1–2, pp. 1–210.
12. Lai, F., Zhu, X., Madhyastha, H.V. and Chowdhury, M. (2021) 'Oort: efficient federated learning via guided participant selection', Proceedings of the 15th USENIX OSDI, pp. 19–35.
13. Li, X., Huang, K., Yang, W., Wang, S. and Zhang, Z. (2020a) 'On the convergence of FedAvg on non IID data', Proceedings of ICLR 2020.
14. Li, T., Sahu, A.K., Zaheer, M., Sanjabi, M., Smola, A. and Smith, V. (2020b) 'Federated optimization in heterogeneous networks (FedProx)', Proceedings of Machine Learning and Systems (MLSys), vol. 2, pp. 429–450.
15. McMahan, H.B., Moore, E., Ramage, D., Hampson, S. and y Arcas, B.A. (2017) 'Communication efficient learning of deep networks from decentralized data', Proceedings of AISTATS 2017, pp. 1273–1282, PMLR.

16. Meidan, Y., Bohadana, M., Mathov, Y., Mirsky, Y., Shabtai, A., Breitenbacher, D. and Elovici, Y. (2018) 'N BaIoT: network based detection of IoT botnet attacks using deep autoencoders', IEEE Pervasive Computing, vol. 17, no. 3, pp. 12–22.
17. Mothukuri, V., Parizi, R.M., Pouriyeh, S., Huang, Y., Dehghantanha, A. and Srivastava, G. (2021) 'A survey on security and privacy of federated learning', Future Generation Computer Systems, vol. 115, pp. 619–640.
18. Nguyen, T.D., Marchal, S., Miettinen, M., Fereidooni, H., Asokan, N. and Sadeghi, A.R. (2019) 'DIoT: a federated self learning anomaly detection system for IoT', Proceedings of IEEE ICDCS 2019, pp. 756–767.
19. Niknam, S., Dhillon, H.S. and Reed, J.H. (2020) 'Federated learning for wireless communications: motivation, opportunities, and challenges', IEEE Communications Magazine, vol. 58, no. 6, pp. 46–51.
20. Rey, V., Sanchez, P.M.S., Celdrán, A.H. and Bovet, G. (2022) 'Federated learning for malware detection in IoT devices', Computer Networks, vol. 204, p. 108693.
21. Ullah, I. and Mahmoud, Q.H. (2020) 'A two level flow based anomalous activity detection system for IoT networks', Electronics, vol. 9, no. 3, p. 530.
22. Verma, A. and Ranga, V. (2019) 'ELNIDS: ensemble learning based network intrusion detection system for RPL based IoT', Proceedings of IoT SIU 2019, pp. 1–6, IEEE.
23. Verma, A. and Ranga, V. (2020a) 'Machine learning based intrusion detection systems for IoT applications', Wireless Personal Communications, vol. 111, pp. 2287–2310.
24. Verma, A. and Ranga, V. (2020b) 'RPL NIDDS17: network intrusion detection dataset for RPL based IoT networks', Data in Brief, vol. 30, p. 105257.
25. Wallgren, L., Raza, S. and Voigt, T. (2013) 'Routing attacks and countermeasures in the RPL based Internet of Things', International Journal of Distributed Sensor Networks, vol. 9, no. 8, p. 794326.
26. Wang, S., Tuor, T., Salonidis, T., Leung, K.K., Makaya, C., He, T. and Chan, K. (2019) 'Adaptive federated learning in resource constrained edge computing systems', IEEE Journal on Selected Areas in Communications, vol. 37, no. 6, pp. 1205–1221.
27. Weekly, K. and Pister, K. (2012) 'Evaluating sinkhole routing attacks in RPL networks', Proceedings of IEEE ICIT 2012, pp. 1975–1980.
28. Winter, T., Thubert, P., Brandt, A., Hui, J. and Kelsey, R. et al. (2012) 'RPL: IPv6 routing protocol for low power and lossy networks', IETF RFC 6550.
29. Yang, Q., Liu, Y., Chen, T. and Tong, Y. (2019) 'Federated machine learning: concept and applications', ACM Transactions on Intelligent Systems and Technology, vol. 10, no. 2, pp. 1–19.
30. Zhu, L., Liu, Z. and Han, S. (2019) 'Deep leakage from gradients', Advances in Neural Information Processing Systems (NeurIPS), vol. 32, pp. 14774–14784.
31. Gupta, P., Shukla, M., Arya, N., Singh, U. and Mishra, K., 2022. Let the Blind See: An AIIoT-Based Device for Real-Time Object Recognition with the Voice Conversion. In Machine Learning for Critical Internet of Medical Things (pp. 177-198). Springer, Cham.
32. Kushwaha, U., Gupta, P., Airen, S. and Kuliha, M., 2022, December. Analysis of CNN Model with Traditional Approach and Cloud AI based AppGupta, P., Sharma, V. and Varma, S., 2022. A novel algorithm for mask detection and recognizing actions of human. Expert Systems with Applications, p.116823.roach. In 2022 International Conference on Automation, Computing and Renewable Systems (ICACRS) (pp. 835-842). IEEE.

33. Gupta, P., Shukla, M., Arya, N. and Singh, U., 2023. 5 Internet Smart and of Intelligent Things in Healthcare Systems. IoT in Healthcare Systems: Applications, Benefits, Challenges, and Case Studies, p.77.
34. Gupta, P., Arya, N., Singar, C.P., Chaudhari, A., Singh, U. and Gupta, S., 2025. Safety of Pedestrians in AI-Optimized VANETs for Autonomous Vehicles via Real-Time Vehicle-to-Vehicle Communication. In AI-Driven Transportation Systems: Real-Time Applications and Related Technologies (pp. 169-181). Cham: Springer Nature Switzerland.
35. Gupta, P. and Singh, U., 2025. Evaluation of several yolo architecture versions for person detection and counting. Multimedia Tools and Applications, pp.1-24.