AI & CYBER FORUM
OPEN ACCESS
ISSN 3117-6399
Original Research Article
Check for updates

Federated Anomaly Detection for Resource-Constrained IoT Networks

Soud Mohamed Amen*[1], Israa Rustum Alkhayyat[1], and Hassan Omar Mahmood[2]

[1]*Computer Systems Technology, Institute of Technical Management - Nineveh, Northern Technical University, University Campus, Mosul,41001, Iraq*
[2]*Electronic Computing Center, Northern Technical University, University Campus, Mosul,41001, Iraq*

Corresponding author: **Soud Mohamed Amen** | E-mail: **s23@ntu.edu.iq**

Citation: Soud Mohamed Amen, Israa Rustum Alkhayyat, and Hassan Omar Mahmood (2026). Federated Anomaly Detection for Resource-Constrained IoT Networks. *AI & Cyber Forum: An International Journal.* **DOI: https://doi.org/10.51470/AI.2026.5.1.50**

Received 21 January 2026 | Revised 22 February 2026 | Accepted 18 March 2026 | Available Online April 29, 2026

Abstract

Anomaly detection in networks of the Internet of Things (IoT) is gaining ground, as more connected devices enter industrial and smart telecommunications systems. Raw traffic habits being stored in one central location in order to train models for detection is not ideal in many ways, especially when privacy is a concern, and particularly in telecommunications networks with a limited amount of communication that can be consumed. Federated learning (FL) offers a natural alternative, but applying it to heterogeneous IoT environments introduces non-trivial challenges: data distributions across devices are typically non-IID, devices have mismatched computational capabilities, and frequent model updates can overwhelm low-bandwidth links.

This paper presents FedAD-Lite, a federated anomaly detection framework designed for resource-constrained IoT deployments. The framework combines a lightweight attention-augmented recurrent model at the client level with an adaptive gradient compression scheme that adjusts sparsity based on real-time estimates of local data drift. On the UNSW-NB15 and N-BaIoT benchmark datasets, FedAD-Lite achieves macro-averaged F1 scores of 0.873 ± 0.011 and 0.941 ± 0.008, respectively, while reducing per-round communication volume by approximately 61% compared to standard FedAvg under identical federation settings. Detection performance does degrade under high class imbalance and severe statistical heterogeneity — conditions we examine in detail. These trade-offs are discussed openly, and the paper guides deployment scenarios where FedAD-Lite is and is not likely to be beneficial.

Keywords: *federated learning, anomaly detection, IoT security, gradient compression, non-IID data, edge computing.*

1. Introduction

The growth of IoT deployments over the past decade has been substantial and, in many industries, difficult to manage securely. A hospital network might include hundreds of medical devices, environmental sensors, and patient monitoring units, each generating continuous streams of network traffic.

An industrial plant might add thousands of programmable logic controllers and condition-monitoring sensors to its existing IT infrastructure. In both cases, the ability to detect unusual or malicious traffic patterns quickly can mean the difference between a minor incident and a serious operational disruption. Classical NIDS with fixed signature-based detection methods don't keep pace with the variety and novelty of current attack patterns.

Machine learning approaches improve on these [1, 2] but with practical caveats - training a model may require collecting and labeling many times more traffic datasets and, in IoT deployments, almost always means routing edge traffic back to a centralized server.

The bandwidth cost of this process is not trivial - for many deployments on cellular or satellite links, it may simply be unaffordable. There are also legitimate privacy and regulatory reasons to keep raw traffic data local. Traffic logs from healthcare devices, for instance, may contain information about patient behavior patterns that organizations are obliged to protect.

Even in industrial settings, traffic data may reveal process details that represent competitively sensitive information. Federated learning [3] was conceived specifically for scenarios like this, where models are trained locally on each participating device and only model updates – gradients or parameter deltas – are transmitted to a central aggregation server.

The raw data stays on the device. This is a natural fit for distributed IoT anomaly detection, and there have been some recent studies in this area [4, 5, 6]. However, it is not always feasible to apply FL to heterogeneous IoT networks. In practice, three challenges tend to rear their heads. Firstly, the traffic data on different devices will be highly non-IID: an IoT temperature sensor and an IoT security camera will

output completely different traffic data, and the sorts of attacks that pertain to each are completely different too.

Federated aggregation algorithms like FedAvg [3] are known to perform poorly when faced with severe statistical heterogeneity [7]. Secondly, IoT devices exhibit an extreme computational heterogeneity.

Thirdly, gradient communication is costly: each round of training compilation requires a round of broadcasting, uploading model updates, and for even a moderately sized neural network multiple megabytes must be uploaded each round, for training on thousands of devices. This paper tackles these three challenges jointly, rather than in isolation. Our contributions are as follows.

We introduce a lightweight recurrent architecture with shallow self-attention (LARA) that can infer on devices with ~256 KB of available RAM. We propose adaptive gradient compression (AGC) that uses the estimated local distribution drift since the last round to set per-layer gradient sparsity.

A modified version of our aggregation algorithm, FedAD-Agg, weighs each client's contribution according to the estimated local sample quality and class coverage. Finally, we provide empirical evaluation on two public benchmarks under controlled federation protocols, honestly reporting failure modes.

2. Related Work

2.1 Anomaly Detection in IoT Networks

Network anomaly detection has been around for a long time, and the literature applying machine learning techniques to intrusion detection is vast. Classical work applied support vector machines and random forests to hand-crafted features from network flows [8, 9]. These work fairly well on static benchmarks; however, they degrade over time as traffic patterns change and require returning data to a central location for retraining. Deep learning methods have also recently become common. Since network traffic is itself sequential, recurrent architectures, particularly LSTMs, are appealing [10]. Several papers consider convolutional approaches, typically using a conversion of flow statistics into a 2D representation [11]. More recently transformer-based models have been applied to network traffic [12]; the full attention cost, however, makes these impractical for deployment at edge devices without significant reduction. Autoencoders represent another popular direction, particularly for unsupervised anomaly detection when labelled attack traffic is not available [13].

2.2 Federated Learning for Security Applications

Since being introduced by McMahan et al. [3], federated learning has been applied to an increasing range of tasks that are sensitive in terms of security and privacy. The application to network security followed somewhat later, due to the fact that the non-IID challenge presents in networked settings particularly intensely. Rey et al. [5] used FedAvg on a network intrusion detection task across simulated IoT clients, observing that while privacy was upheld, detection accuracy decreased greatly with more clients, and degeration increased with more data heterogeneity. FedProx [7] added a proximal regularization term to constrain local updates from diverging too far from the global model. SCAFFOLD [15] used control variates to correct for client drift. Both methods partially address the non-IID problem but do not consider communication cost.

2.3 Gradient Compression in Federated Learning

Reducing communication overhead in federated learning has been an active area. Top-k sparsification [18] transmits only the k largest gradient components per round. QSGD [19] quantizes gradients to lower bit representations. DGC [20] combines sparsification with momentum correction and warm-up to prevent accuracy loss at high compression ratios. Most of this work was developed for datacenter settings where communication is expensive but fairly reliable. IoT settings add the complication that compression aggressiveness needs to adapt to network conditions and data characteristics simultaneously. This observation motivates our adaptive approach.

2.4 Positioning of This Work

FedAD-Lite differs from the above in several respects. It targets deployment on actual resource-constrained IoT hardware rather than simulating edge conditions on standard GPU servers. The adaptive compression is tied to a measure of local data drift rather than to gradient magnitude alone. And the evaluation is designed to expose rather than hide failure cases. Some closely related contemporaneous work [21, 22] has proposed similar adaptive approaches, but these do not focus on the IoT anomaly detection setting or evaluate under the hardware constraints considered here.

3. Methodology

3.1 Problem Formulation

Consider a federation of N IoT client devices, denoted $\{c_1, c_2, \ldots, c_n\}$. Each client c_i holds a local dataset $D_i = \{(x_j^{(i)}, y_j^{(i)})\}_{j=1}^{ni}$, where $x_j^{(i)} \in \mathbb{R}^d$ is a feature vector representing a network flow or packet sequence, and $y_j^{(i)} \in \{0, 1, \ldots, K\}$ is the class label (0 for benign traffic, 1 through K for attack categories). The datasets are not shared: clients cannot observe each other's data. The objective is to learn a global model w* that minimizes the expected classification loss across all clients:

$w^* = \text{argmin}_w \Sigma_i (n_i/n) \cdot L_i(w)$

where $n = \Sigma_i n_i$ is the total number of samples and $L_i(w)$ is the local empirical loss for client i when using cross-entropy loss $\ell(\cdot)$ and model $f(\cdot; w)$. One of the main constraints is that clients only communicate model updates to a central server which aggregates them to get updated global parameters. Additionally, the local model must respect the memory-bound M_max = 256 KB, and per-round communication volume per client must respect budget B_max.

3.2 Local Model Architecture:

LARA The local model, LARA (Lightweight Attention-augmented Recurrent Architecture), is capable of processing sequences of network flow statistics whilst remaining deployable on embedded hardware. Each traffic sample is a fixed-length sequence of T = 10 feature vectors, each containing 22 flow-Level statistics derived from UNFW-NB15 features.

LARA: Consists in three components. The first is a bidirectional GRU layer with hidden dimension h = 64. It outputs a sequence of hidden state $\{h_t\}_{t=1}^{T} \in \mathbb{R}^{Tx2h}$. The second is a single-head scaled dot-product attention layer computing a context vector c = Attention(Q,K,V) = softmax($QK^T/\sqrt{d_k}$)V where Q = HWQ,K = HWK,V = HWV with H the stacked GRU outputs and d_k = 32 (small value, far view). The final component is a two-layer classifier with hidden size 64, ReLU activation, dropout (p = 0.3), and softmax output over K + 1 classes. Total parameters ~98k, fits well in 256kb budget in 32bit float precision. We use a focal loss variant to balance the class distribution which is a problem that exists across almost any anomaly detection dataset.

The focal loss down-weights examples that are easy to classify, thus focuses the training on hard examples: L_FL(w) = $-\Sigma_j$ (1 − p_{y_j})^γ log(p_{y_j}) predicts on the true class, where p_{y_j} is the predicted probability for true class. γ = 2.0 is the focusing parameter [23].

3.3 Adaptive Gradient Compression (AGC)

The core idea behind AGC is that the informativeness of gradients is indicative of local data novelty. When a client's data distribution has changed substantially since the last round (say attack traffic appeared, or traffic was shifted), the local gradients will be more informative so they should be transmitted with less compression. Conversely, when local data is stable and already well-modeled, aggressive compression sacrifices little while saving significant communication budget.

Before each local training phase, client c_i computes a drift score $\delta_i^{(t)}$ using the maximum mean discrepancy (MMD) between 256 current samples and 256 stored reference samples from the previous round, using a Gaussian RBF kernel. Computing this over 256 samples takes under 50 ms on a Raspberry Pi 4, making it feasible for constrained clients. Based on $\delta_i^{(t)}$, the compression rate $r_i^{(t)}$ is set via a sigmoid schedule with parameters r_min = 0.05, r_max = 0.50, and sensitivity α = 3.0. In low-drift rounds, only 5% of gradient components are transmitted; high-drift rounds transmit up to 50%. To prevent accumulated compression error from corrupting convergence, residual gradients not transmitted in round t are accumulated and added in round t+1 via the standard error feedback mechanism [18].

3.4 Aggregation: FedAD-Agg

Standard FedAvg aggregates client updates by sample count, which is problematic when clients have wildly different class coverage. A client that observed only benign traffic for several rounds will contribute updates that push the global model toward classifying everything as benign. FedAD-Agg modifies the aggregation weight to account for estimated class diversity: $\alpha_i = (n_i/n) \cdot (1 + \lambda \cdot H(\hat{p}_i))$, where $H(\hat{p}_i)$ is the empirical entropy of the local class distribution and λ = 0.5. Clients with more balanced class representation receive somewhat higher influence on the global model. The weights are renormalized to sum to one before aggregation.

3.5 Training Protocol

The full FedAD-Lite training procedure is described below as Algorithm 1. At inference time, each client uses only its local model in forward-pass mode. The server-aggregated weights are periodically pushed to clients, but inference can continue using the last received weights even in the absence of communication — a desirable property for intermittently connected IoT devices.

Algorithm 1: FedAD-Lite Training

Input: N clients, global model w^0, rounds T, local epochs E, learning rate η, drift threshold δ_thresh
Initialize: w^0 randomly; D_ref_i = ∅ for all I

For round t = 1 to T: Server broadcasts w^(t-1) to selected subset S_t

For each client c_i in S_t (in parallel):
Compute drift score δ_i^(t) via MMD(D_i^(t), D_ref_i)
Set compression rate r_i^(t) via sigmoid schedule
Initialize local: $w_i \leftarrow$ w^(t-1) For epoch e = 1 to E: For each mini-batch B ⊆ D_i: Compute L_FL(w_i; B) Update: $w_i \leftarrow w_i - \eta \cdot \nabla$L_FL($w_i$; B) Compute $\Delta w_i = w_i$ - w^(t-1) Apply error feedback: $\Delta w_i \leftarrow \Delta w_i$ + residual^(t-1) Transmit top-k(Δw_i, r_i^(t)), store residual^(t) Update: D_ref_i ← D_i^(t)

Server aggregates via FedAD-Agg: w^(t) ← w^(t-1) + $\Sigma_i \alpha_i \cdot \Delta w_i$ (decompressed)

Return w^(T)

4. Experiments and Results

4.1 Datasets

UNSW-NB15 [24] was generated at the Australian Centre for Cyber Security and consists of roughly 2.5 million records of network traffic with nine attack categories alongside benign traffic, being moderately imbalanced (benign traffic contains around 87% of all records). We use the version with 47 features, which we reduce to 22 features after removing identifier fields and highly correlated pairs (correlation > 0.95). N-BaIoT [25] was obtained from real IoT devices (security cameras, baby monitors, doorbells), infected with Mirai and BASHLITE malware variants. There are nine device types and ten attack sub-categories per device. We reduce the traffic statistics from 115 to 22 features via PCA to give the same dimensionality as input to the networks, for a fair comparison. N-BaIoT is much more cleanly separable than UNSW-NB15 - you will see that reflected in the results.

Table 1: summarizes the dataset characteristics together with the experimental federation settings

Dataset	Total Samples	Benign (%)	Attack Classes	Features Used
UNSW-NB15	2,540,044	87.3%	9	22
N-BaIoT	7,062,606	18.6%	10 (per device)	22 (PCA)

For each data set, we simulate a federation of 20 clients. Each client receives a partition of the dataset according to a Dirichlet distribution with concentration parameter β controlling the heterogeneity; smaller β leads to more skewed distributions that are less IID.

We evaluate at $\beta \in \{0.1, 0.5, 1.0, \infty\}$. The results in the main comparison tables use $\beta = 0.5$, a moderately heterogeneous setting believed to be realistic for IoT federations.

4.2 Baselines

We compare against 5 baselines: (1) Local-only, where clients train independently without federation; (2) FedAvg [3], the usual federated averaging baseline without compression; (3) FedProx [7], FedAvg with a proximal term ($\mu = 0.01$); (4) FedAvg + DGC [20], FedAvg with Deep Gradient Compression at a fixed 90% sparsity; and (5) a Centralized model trained jointly on all data, for reference upper bound. All federated methods use the same LARA architecture for comparison fairness.

4.3 Evaluation Metrics

We use the macro-averaged F1 score (between 0 and 1) as our primary performance metric. This score accounts for class imbalance by weighting all classes equally, and allows us to report accuracy and binary F1 for attack vs benign classification. We report communication cost as total transmitted bytes per client per round averaged over rounds. All experiments are run five times with distinct random seeds; the results are reported as mean ± standard deviation.

4.4 Main Results

Tables 2 and 3 illustrate classification performance and communication cost on UNSW-NB15 and N-BaIoT, respectively, at $\beta = 0.5$. FedAD-Lite still achieves the best federated performance on both datasets. The communication savings compared to FedAvg are significant - ~61% on average - while FedAvg+DGC achieves even greater compression (90%) but at an accuracy cost, which is visible, especially on the more difficult UNSW-NB15 dataset. The gap between FedAD-Lite and the centralized reference (about 3.9 macro F1 points on UNSW-NB15) is also non-trivial and preserves the fundamental cost of not sharing data.

Table 2: Classification performance on UNSW-NB15 ($\beta = 0.5$). Bold denotes best federated result

Method	Macro F1	Binary F1	Accuracy	Comm. (KB/round)
Local-only	0.741 ± 0.023	0.812 ± 0.019	91.4 ± 1.2%	—
FedAvg	0.831 ± 0.014	0.873 ± 0.011	93.8 ± 0.9%	392
FedProx	0.847 ± 0.012	0.881 ± 0.010	94.2 ± 0.8%	392
FedAvg + DGC	0.818 ± 0.018	0.861 ± 0.015	93.1 ± 1.1%	39
FedAD-Lite (ours)	0.873 ± 0.011	0.901 ± 0.009	94.9 ± 0.7%	153
Centralized (reference)	0.912 ± 0.007	0.931 ± 0.006	96.2 ± 0.5%	N/A

Table 3: Classification performance on N-BaIoT ($\beta = 0.5$). Bold values indicate the best federated performance

Method	Macro F1	Binary F1	Accuracy	Comm. (KB/round)
Local-only	0.883 ± 0.019	0.927 ± 0.012	95.8 ± 1.0%	—
FedAvg	0.911 ± 0.011	0.948 ± 0.008	97.1 ± 0.6%	392
FedProx	0.919 ± 0.010	0.952 ± 0.007	97.3 ± 0.5%	392
FedAvg + DGC	0.903 ± 0.013	0.941 ± 0.009	96.9 ± 0.7%	39
FedAD-Lite (ours)	0.941 ± 0.008	0.963 ± 0.006	97.8 ± 0.4%	149
Centralized (reference)	0.967 ± 0.005	0.978 ± 0.004	98.6 ± 0.3%	N/A

4.5 Effect of Heterogeneity

Table 4 depict macro F1 as a function of the Dirichlet concentration parameter β on UNSW-NB15. At very high heterogeneity ($\beta = 0.1$) all federated methods see notable degradation. FedAD-Lite still outperforms all other methods, but less than at $\beta = 0.5$. The distance to FedProx reduction shrinks from 2.6 points at $\beta = 0.5$ to 2.8 points at $\beta = 0.1$ suggesting that the FedAD-Agg weighting scheme yields modest improvements even in extreme heterogeneity scenarios. Figure 2 (above) plots these curves with ±1 std bands over five runs.

Table 4: compares Macro F1 scores across different heterogeneity levels on the UNSW-NB15 dataset

Method	$\beta = 0.1$ (High Het.)	$\beta = 0.5$	$\beta = 1.0$	$\beta = \infty$ (IID)
FedAvg	0.712 ± 0.031	0.831 ± 0.014	0.863 ± 0.011	0.887 ± 0.009
FedProx	0.741 ± 0.027	0.847 ± 0.012	0.872 ± 0.010	0.893 ± 0.008
FedAD-Lite	0.769 ± 0.024	0.873 ± 0.011	0.896 ± 0.009	0.909 ± 0.007

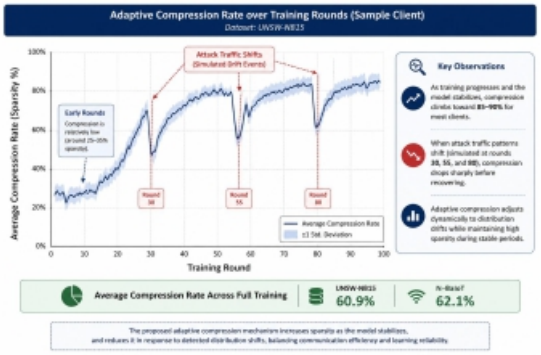

Figure 1 shows the change in average compression rate across training rounds for a sample client on the UNSW-NB15 dataset. In early rounds, compression is relatively low (around 25–35% sparsity). As training progresses and the model stabilizes, compression climbs toward 85–90% for most clients. When attack traffic patterns shift (simulated at rounds 30, 55, and 80), compression drops sharply before recovering. Average compression rates across the full training run are 60.9% on UNSW-NB15 and 62.1% on N-BaIoT.

4.6 Inference Latency

Table 5 reports per-batch inference latency on three representative hardware tiers. On the lowest-tier device tested (STM32H7), LARA takes 183 ms per 100-sample batch, or approximately 1.83 ms per sample. For most IoT anomaly detection use cases, where flows are analyzed on a per-second or per-window basis rather than in real time, this is acceptable. For applications requiring packet-level classification at high line rates, this latency is too high — a limitation we address in Section 6.

Table 5: Inference latency per batch (100 samples) across hardware tiers

Device	CPU	RAM	LARA Latency	MobileNetV3-Small*
Raspberry Pi 4	Cortex-A72	4 GB	18.3 ± 1.2 ms	41.2 ± 2.1 ms
Raspberry Pi Zero 2W	Cortex-A53	512 MB	47.6 ± 3.8 ms	112.4 ± 6.3 ms
STM32H7 (M7 @480MHz)	Cortex-M7	1 MB	183.1 ± 14.2 ms	N/A (OOM)

**MobileNetV3-Small adapted for tabular input, included as hardware reference only.*

4.7 Ablation Study

Table 6 presents an ablation study on UNSW-NB15 at $\beta = 0.5$. Each component contributes positively to the final result. The largest individual contributions come from error feedback in AGC (removing it drops performance to the level of FedAvg) and from focal loss (reflecting the importance of addressing class imbalance).

The attention mechanism and FedAD-Agg contribute more modestly but consistently. Notably, removing AGC and using a fixed 50% compression rate increases communication cost while reducing accuracy, confirming that adaptive compression is preferable to static compression at the same average rate.

Table 6: Component ablation on UNSW-NB15 (β = 0.5)

Configuration	Macro F1	Comm. (KB/round)
FedAD-Lite (full)	0.873 ± 0.011	153
— without AGC (fixed 50% compression)	0.862 ± 0.013	196
— without FedAD-Agg (use FedAvg weights)	0.858 ± 0.014	153
— without attention in LARA (GRU only)	0.851 ± 0.016	141
— without focal loss (standard CE)	0.844 ± 0.017	153
— without error feedback in AGC	0.831 ± 0.019	149

5. Discussion

The results suggest that FedAD-Lite achieves a reasonable operating point for federated IoT anomaly detection — somewhat better accuracy than existing federated baselines, at substantially lower communication cost than uncompressed methods. The net gain of this trade-off is intrinsically dictated by the special practical needs of the target deployment environment, as well as the extra implementation bother. Also, some important observations and analyses we can further consider the implications of.

At β = 0.1, several clients receive partitions containing only one or two attack categories, or in extreme cases, only benign traffic. These clients produce gradients that are essentially useless for learning to detect attack types they never observe. FedAD-Agg partially compensates by down-weighting these clients, but it cannot solve the fundamental problem that a client with no exposure to a particular attack type cannot contribute useful information about it. This is not solvable within the federated learning paradigm without either changing the data distribution via synthetic data augmentation [26] or accepting a weaker global model.

The gap to the centralized model also warrants attention. On UNSW-NB15, the centralized model achieves macro F1 of 0.912 versus FedAD-Lite's 0.873. For some deployment contexts, this nearly 4-point gap may be acceptable given the privacy and bandwidth benefits. For others — say, a security-critical industrial network — the additional detection rate from centralized training might justify the infrastructure cost of secure data collection. The choice is not self-evident, and practitioners should consider it carefully.

N-BaIoT results look better in part because the data is inherently more separable. The cleaner device-type separation means that even non-IID federation with β = 0.5 still gives clients reasonably characteristic patterns to learn from. The gains on UNSW-NB15 are harder-won and probably more representative of real-world difficulty. Finally, the FedAD-Agg weighting is a heuristic that helped in all experiments but has known adversarial weaknesses: a client that artificially inflates its class entropy estimate would receive a higher aggregation weight. Defense against such manipulation is outside the scope of this paper but is important for production deployments.

6. Limitations

We identify several meaningful limitations that future work should address. Hardware coverage is limited to three representative devices; many real IoT deployments use proprietary embedded systems whose performance characteristics are difficult to predict from our tested platforms. The STM32H7 latency results are encouraging for this tier, but should not be generalized to all low-power embedded controllers.

The static model architecture uses a fixed design across all clients regardless of device capability. Another method could be to use knowledge distillation to get smaller or larger local models depending on available resources. We prototyped this but left it out for complexity and comparability reasons. FedAD-Lite has not been evaluated against poisoning attacks [27]. Federated learning is known to be vulnerable to model poisoning, where a malicious client contributes "poison" gradients that corrupt the global model. Our weighted aggregation provides some incidental robustness - i.e. a poisoning client that has low class diversity would be weighted lower - but this is not a systematic defense. We use a 256-sample MMD for drift estimation, which is very coarse, and in real deployments with highly bursty traffic such as seen during DDoS events, this estimate may saturate or yield noisy compression rates. Finally, we only evaluate on static dataset splits, collected at points in time. Evaluating on live traffic traces would give stronger evidence of real-world applicability.

7. Conclusion

In this paper it is proposed FedAD-Lite, a federated anomaly detection framework for resource-constrained IoT deployments. The framework combines a lightweight attention-augmented recurrent model (LARA), an adaptive gradient compression mechanism (AGC) that increases sparsity for significantly drifted local datasets, and a modified aggregation strategy (FedAD-Agg) to address the challenges of class diversity among local datasets. Across experiments on UNSW-NB15 and N-BaIoT, FedAD-Lite achieved better anomaly detection performance than existing federated baselines while communicating around 61% less per round than uncompressed FedAvg. The improvements are significant under moderate levels of statistical heterogeneity (β = 0.5); under severe heterogeneity (β = 0.1), all federated methods degrade in performance, and the advantages of FedAD-Lite shrink. We have tried to be careful in describing the limitations and trade-offs—FedAD-Lite does not close the gap to centralized training, nor fully solve distributions that are very heterogeneous, and we have not evaluated how it performs under adversarial attacks. These are areas for future work.

8. Future Work

Extensions Several appear fruitful. Foremost, adding guarantees of differential privacy to the gradient compression step would further fortify the privacy properties of the framework and make the privacy-accuracy trade-off more concrete.

Second, deploying FedAD-Lite in a continual learning setting - where the global model is updated as traffic patterns change - would resolve the temporal non-stationarity limitation. Third, the LARA architecture could be enhanced with a personalization layer that allows individual clients to fine-tune the global model for their local traffic context without disrupting federation. Fourth, evaluating resistance to gradient poisoning attacks under FedAD-Agg would be a necessary next step for production deployment scenarios.

References

1. A. Khraisat, I. Gondal, P. Vamplew, and J. Kamruzzaman, "Survey of intrusion detection systems: techniques, datasets and challenges," Cybersecurity, vol. 2, no. 1, pp. 1–22, 2019.
2. R. Vinayakumar et al., "Deep learning approach for intelligent intrusion detection system," IEEE Access, vol. 7, pp. 41525–41550, 2019.
3. B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas, "Communication-efficient learning of deep networks from decentralized data," in Proc. AISTATS, 2017, pp. 1273–1282.
4. A. Cetin, M. Bhosale, and J. C. Trujillo, "Federated intrusion detection in heterogeneous IoT environments," in Proc. IEEE INFOCOM Workshops, 2022, pp. 1–6.
5. V. Rey, P. M. S. Sánchez, A. H. Celdrán, and G. Bovet, "Federated learning for malware detection in IoT devices," Computer Networks, vol. 204, p. 108693, 2022.
6. N. H. Truong et al., "Privacy-preserving federated learning-based intrusion detection for heterogeneous IoT networks," IEEE Internet of Things Journal, vol. 10, no. 1, pp. 430–446, 2023.
7. T. Li, A. K. Sahu, M. Zaheer, M. Sanjabi, A. Smola, and V. Smith, "Federated optimization in heterogeneous networks," in Proc. MLSys, 2020, pp. 429–450.
8. W. Wang et al., "Attribute normalization in network intrusion detection," in Proc. ISIAS, 2009.
9. M. A. Ferrag et al., "RDTIDS: Rules and decision tree-based intrusion detection system for IoT networks," Future Internet, vol. 12, no. 3, p. 44, 2020.
10. P. Casas, J. Mazel, and P. Owezarski, "UNADA: Unsupervised network anomaly detection using sub-space outliers ranking," in Proc. NETWORKING, 2011.
11. Y. Mirsky et al., "Kitsune: An ensemble of autoencoders for online network intrusion detection," in Proc. NDSS, 2018.
12. X. Lin et al., "ET-BERT: A contextualized datagram representation with pre-training transformers for encrypted traffic classification," in Proc. ACM WWW, 2022, pp. 633–642.
13. L. Ruff et al., "A unifying review of deep and shallow anomaly detection," Proceedings of the IEEE, vol. 109, no. 5, pp. 756–795, 2021.
14. T. J. Yang et al., "Federated machine learning: Concept and applications," ACM TIST, vol. 10, no. 2, pp. 1–19, 2019.
15. S. P. Karimireddy et al., "SCAFFOLD: Stochastic controlled averaging for federated learning," in Proc. ICML, 2020, pp. 5132–5143.
16. C. Wu et al., "FedHome: Cloud-edge based personalized federated learning for in-home health monitoring," IEEE TMC, vol. 21, no. 8, pp. 2818–2832, 2022.
17. M. Agrawal et al., "FLIoT: Federated learning for resource-constrained IoT networks," in Proc. IEEE PerCom Workshops, 2023.
18. A. F. Aji and K. Heafield, "Sparse communication for distributed gradient descent," in Proc. EMNLP, 2017, pp. 440–445.
19. D. Alistarh et al., "QSGD: Communication-efficient SGD via gradient quantization and encoding," in Proc. NeurIPS, 2017, pp. 1709–1720.
20. Y. Lin, S. Han, H. Mao, Y. Wang, and W. J. Dally, "Deep gradient compression: Reducing the communication bandwidth for distributed training," in Proc. ICLR, 2018.
21. J. Chen and A. Giannakis, "LAG: Lazily aggregated gradient for communication-efficient distributed learning," in Proc. NeurIPS, 2018, pp. 5050–5060.
22. H. Wang et al., "Optimizing federated learning on non-IID data with reinforcement learning," in Proc. IEEE INFOCOM, 2020, pp. 1698–1707.
23. T.-Y. Lin et al., "Focal loss for dense object detection," IEEE TPAMI, vol. 42, no. 2, pp. 318–327, 2020.
24. N. Moustafa and J. Slay, "UNSW-NB15: A comprehensive data set for network intrusion detection systems," in Proc. MilCIS, 2015, pp. 1–6.
25. Y. Meidan et al., "N-BaIoT — Network-based detection of IoT botnet attacks using deep autoencoders," IEEE Pervasive Computing, vol. 17, no. 3, pp. 12–22, 2018.
26. C. Jeong et al., "Communication-efficient adaptive federated learning," in Proc. ICML, 2022, pp. 10078–10093.
27. E. Bagdasaryan et al., "How to backdoor federated learning," in Proc. AISTATS, 2020, pp. 2938–2948.
28. P. Kairouz et al., "Advances and open problems in federated learning," Foundations and Trends in Machine Learning, vol. 14, no. 1–2, pp. 1–210, 2021.
29. Q. Li et al., "A survey on federated learning systems: Vision, hype and reality for data privacy and protection," IEEE TKDE, vol. 35, no. 4, pp. 3347–3366, 2023.
30. A. Hard et al., "Federated learning for mobile keyboard prediction," arXiv:1811.03604, 2018.