VT INTERNATIONAL PRESS
VTIPJMRR | Journal of Multidisciplinary Research & Review
www.journal.vtinterpress.com

(Research Article)

DT-FedLite: A Drift-Triggered Federated Lightweight Temporal Convolution–Attention Network with Warm-Start Hummingbird Search for Privacy-Preserving Threat Detection in Streaming IoT Wireless Networks

Kapil Dnyaneshwar Dere[1], Pushpalata G. Aher[2]

[1]*Department of Computer Engineering School of Engineering & Technology, Sandip University, Trimbak Road, Nashik, Maharashtra, India, 422002, Email: kapilddere@gmail.com, ORCID: 0000-0002-9432-9434*

[2]*Department of Computer Engineering School of Engineering & Technology, Sandip University, Trimbak Road, Nashik, Maharashtra, India, 422002, Email: pushpalata.aher@sandipuniversity.edu.in, ORCID ID: 0000-0001-9841-7215*

***Corresponding Author Email: kapilddere@gmail.com**

Received: 01/09/2026	Accepted: 26/09/2026	Published: 27/10/2026

ABSTRACT

Threat detectors deployed across Internet of Things wireless networks face three coupled difficulties: traffic and attack behaviour change continuously, raw telemetry cannot be centralised without exposing device and user information, and gateways offer only a few megabytes of memory and milliseconds of compute per decision. This paper presents DT-FedLite, a federated framework that addresses the three difficulties jointly. Each edge gateway runs a lightweight temporal convolution–attention network that combines depthwise-separable dilated causal convolutions with linear-complexity temporal attention, totalling 38.6 thousand parameters and 41.3 kilobytes after 8-bit quantisation. A dual-signal drift monitor, coupling a Page–Hinkley test on delayed-label loss with an unsupervised embedding-shift statistic, replaces fixed-period federation by event-driven rounds that fire only when the population-weighted drift score crosses a threshold. Client updates are trained with differentially private stochastic gradient descent, quantised, securely aggregated, and weighted by drift severity. Hyper-parameters are re-tuned after each drift episode by a warm-start artificial hummingbird algorithm that seeds its population and visit table from an archive of previously elite configurations. On streaming replays of CICIoT2023, Edge-IIoTset and AWID3 with injected sudden, gradual and recurring drift, DT-FedLite attains macro-F1 scores of 96.21%, 97.48% and 95.36% at a window-level privacy budget of $\varepsilon = 3.2$, recovers from drift within 8.0 windows compared with 25.7 for periodic federation, reduces uplink traffic by 90.0%, re-tunes hyper-parameters with 69% fewer fitness evaluations than a cold-start search, and classifies a window in 0.58 milliseconds on a Raspberry Pi 4 gateway.

Keywords: *federated learning; concept drift; temporal convolutional network; linear attention; artificial hummingbird algorithm; differential privacy; IoT intrusion detection; wireless network security*

I. INTRODUCTION

The Internet of Things (IoT) has turned wireless networks into the operational nervous system of homes, hospitals, factories and cities. Low-power devices communicate over IEEE 802.11, Bluetooth Low Energy, Zigbee and cellular links, and most of them have neither the compute budget nor the update discipline required to defend themselves. Consequently, network-level intrusion and threat detection at the gateway has become the practical line of defence [1], [2]. Deep sequence models have pushed detection accuracy on benchmark captures well above 95%, yet the conditions under which those numbers are obtained—a static, centrally pooled and fully labelled dataset—rarely hold in a deployed wireless IoT network.

Three properties of real deployments break that assumption. First, the data are *streaming* and *non-stationary*: firmware updates, new device types, seasonal usage and, above all, previously unseen attack campaigns continuously change the joint distribution of features and labels, a phenomenon known as concept drift [3], [4]. A detector trained once degrades silently, and the degradation is largest exactly when a novel attack appears. Second, the data are *private and distributed*: packet timing, device identifiers and radio fingerprints reveal occupancy, health and behavioural patterns, and regulatory regimes such as the GDPR and India's Digital Personal

 © 2026 VT International Press. All rights reserved. E-ISSN: 3139-7026

Data Protection Act discourage moving them to a central server. Federated learning (FL) [5], [6] keeps raw telemetry on the gateway and exchanges only model updates, and several studies have confirmed its viability for IoT intrusion detection [7], [8]. Third, the hardware is *resource constrained*: gateways are often single-board computers or access points with limited memory, power and backhaul bandwidth, so both the per-window inference cost and the per-round communication cost matter.

Existing federated intrusion detection systems typically address only a subset of these properties. Most synchronise on a fixed schedule, which wastes bandwidth when traffic is stable and reacts too slowly when it is not. Recurrent or transformer detectors commonly used for temporal modelling carry hundreds of thousands of parameters, inflating both on-device latency and the size of every update. Hyper-parameters such as learning rate, model width and drift sensitivity are usually fixed once by hand, even though the optimal setting shifts after a drift event, and federated hyper-parameter tuning is itself expensive because every candidate must be evaluated through communication rounds [9]. Finally, FL by itself does not provide a formal privacy guarantee, because gradients can leak membership information unless they are perturbed and aggregated securely.

This paper proposes **DT-FedLite**, a drift-triggered federated framework that treats detection accuracy, drift adaptation, privacy and edge efficiency as a single design problem. Its principal contributions are as follows.

(i) **A lightweight temporal convolution–attention network (LTCAN)** that stacks depthwise-separable, dilated causal convolution blocks [10], [11] with a linear-complexity temporal attention layer and attention pooling. LTCAN attains a 61-step receptive field with 38.6 k parameters, runs in 0.58 ms per window on a Raspberry Pi 4 after INT8 quantisation, and matches a standard-convolution variant with 4.1× more parameters.

(ii) **A dual-signal drift monitor and event-driven federation protocol**. A Page–Hinkley test on delayed-label loss is combined with an unsupervised Mahalanobis shift statistic on penultimate-layer embeddings, so that drift is detected even before labels arrive. Rounds are triggered when the population-weighted drift score crosses a threshold, with a heartbeat guaranteeing liveness, and client updates are weighted by drift severity.

(iii) **A privacy-preserving update path** combining per-example clipped differentially private SGD [12], 8-bit stochastic quantisation and secure aggregation [13], providing $(\varepsilon, \delta) = (3.2, 10^{-5})$ window-level guarantees (record-level $(6.4, 4.9 \times 10^{-4})$ by group privacy) while cutting uplink traffic by 90.0% relative to periodic FP32 federation.

(iv) **A warm-start artificial hummingbird algorithm (WS-AHA)** that extends the artificial hummingbird algorithm [14] with an elite archive, drift-scaled population seeding and visit-table priming, re-tuning eleven hyper-parameters after drift with 69% fewer fitness evaluations than its cold-start counterpart.

(v) **A streaming evaluation protocol** on three public datasets—CICIoT2023 [15], Edge-IIoTset [16] and AWID3 [17]—with non-IID client partitions and injected sudden, gradual and recurring drift, reporting detection, adaptation, communication, privacy and on-device cost together.

The remainder of the paper is organised as follows. Section II reviews related work. Section III describes the system model, datasets and every component of DT-FedLite. Section IV reports the experimental results, Section V discusses their implications and limitations, and Section VI concludes the paper.

II. LITERATURE REVIEW / RELATED WORK

Research relevant to DT-FedLite spans four threads: deep and lightweight detectors for IoT traffic, federated intrusion detection, concept-drift adaptation in centralised and federated settings, and metaheuristic hyper-parameter optimisation. Each is reviewed below, and Table 1 summarises how representative approach classes cover the requirements targeted in this work.

A. Deep and Lightweight Detectors for IoT Traffic

Comparative studies show that convolutional, recurrent and hybrid deep models consistently outperform shallow learners on modern IoT captures [1], and dedicated datasets now cover large device populations and diverse attack families [15], [16], [18]. For wireless links specifically, the AWID corpora expose MAC-layer and radio features of IEEE 802.11 attacks such as deauthentication floods, evil-twin and key-reinstallation exploits [2], [17]. Temporal convolutional networks (TCNs) use causal, dilated one-dimensional convolutions with residual connections and have matched or exceeded recurrent networks on many sequence tasks while being fully parallel [10], [19]. Attention mechanisms [20] let a model emphasise the few time steps that carry attack evidence, but standard self-attention scales quadratically with sequence length; kernel-based linear attention [21] removes that cost. On the efficiency side, depthwise-separable convolutions [11] and integer-only quantisation [22] are established techniques for embedded deployment. Kitsune [23] demonstrated online, on-device anomaly detection with an ensemble of small autoencoders, though it learns per device and does not share knowledge across sites. To the best of the authors' knowledge, the combination of depthwise-separable dilated TCN blocks with linear temporal attention has not been studied as a federated IoT detector under streaming drift.

© 2026 VT International Press. All rights reserved.

B. Federated Intrusion Detection

FedAvg [5] established the client–server protocol in which local models are trained on private data and averaged by a server. Statistical heterogeneity across clients slows and destabilises FedAvg; FedProx [24] adds a proximal term to local objectives and SCAFFOLD [25] corrects client drift with control variates. In security, DÏoT [26] learned device-type-specific GRU models federatedly to flag compromised IoT devices, and Mothukuri et al. [27] combined federated GRU models with ensembling for IoT attack detection. Rahman et al. [7] compared centralised, on-device and federated detectors and found federated training close to centralised accuracy while preserving privacy, and Campos et al. [8] reviewed open challenges, notably non-IID data and evaluation realism. Privacy in these systems is usually implicit—raw data stay local—whereas formal protection requires differential privacy [12] and secure aggregation [13], both of which add cost that a lightweight design must absorb.

C. Concept Drift in Streaming and Federated Settings

Drift-adaptation surveys [3], [4] distinguish sudden, gradual, incremental and recurring drift and classify detectors by the statistic they monitor. Error-rate detectors such as DDM [28] and the Page–Hinkley test [29] are simple and sensitive, but they require labels, which in security operations arrive with delay and only for a fraction of traffic. Window-based detectors such as ADWIN [30] adapt their memory automatically. In federated learning, Canonaco et al. [31] adapted learning rates when drift was detected, and Jothimurugesan et al. [32] studied drift that is distributed unevenly across clients and proposed clustering-based remedies. These works establish that drift must be handled explicitly in FL, but they keep a periodic communication schedule and do not target constrained intrusion detectors; using drift itself as the trigger for communication is the gap DT-FedLite fills. More recent work confirms the need for drift handling in federated IoT settings: FLARE [44] uses separate training and inference schedulers to reduce edge data transfer and detect drift quickly, and Rehman et al. [45] compare incremental-learning strategies for federated intrusion detection under evolving attack patterns. Neither, however, couples communication to a privatised drift signal or re-tunes hyper-parameters after drift.

D. Metaheuristic and Federated Hyper-Parameter Optimisation

Population-based metaheuristics such as particle swarm optimisation (PSO) [33], the grey wolf optimiser (GWO) [34] and the whale optimisation algorithm [35] are widely used to tune intrusion detectors because they handle mixed, non-differentiable search spaces. The artificial hummingbird algorithm (AHA) [14] models guided, territorial and migratory foraging with a visit table that records how long each food source has gone unvisited, which gives it an explicit memory-driven balance of exploration and exploitation. Bayesian methods such as the tree-structured Parzen estimator (TPE) [36] are sample-efficient, and meta-learning warm starts [37] show that reusing past optima accelerates new searches. In FL, hyper-parameter tuning is costly because each evaluation consumes communication rounds [9]. No single optimiser dominates across problems [38]; the design question is therefore how to exploit the structure of the present problem, in which consecutive drift episodes produce related but not identical optima.

E. Research Gap

Table 1 shows that no prior approach class simultaneously offers event-driven federation, a lightweight temporal detector, formal privacy and drift-aware hyper-parameter re-tuning. DT-FedLite is designed to fill this gap.

Table 1. Coverage of requirements by representative approach classes (✓ addressed, ✗ not addressed).

Approach class	Federated	Temporal model	Drift-aware	Edge-light	HPO	Formal privacy
Centralised deep IDS [1]	✗	✓	✗	Partial	Partial	✗
On-device online detector [23]	✗	Partial	Partial	✓	✗	✗
Periodic FL-IDS [7], [26], [27]	✓	✓	✗	Partial	✗	✗
Drift-adaptive FL [31], [32]	✓	✗	✓	✗	✗	✗
Metaheuristic-tuned IDS [14], [34]	✗	Partial	✗	Partial	✓	✗
DT-FedLite (this work)	✓	✓	✓	✓	✓	✓

III. METHODOLOGY

This section formalises the problem and describes the datasets, the LTCAN detector, the drift-triggered federation protocol, the privacy-preserving update path and the WS-AHA tuner. Figure 1 gives an overview of the three-tier architecture.

© 2026 VT International Press. All rights reserved.

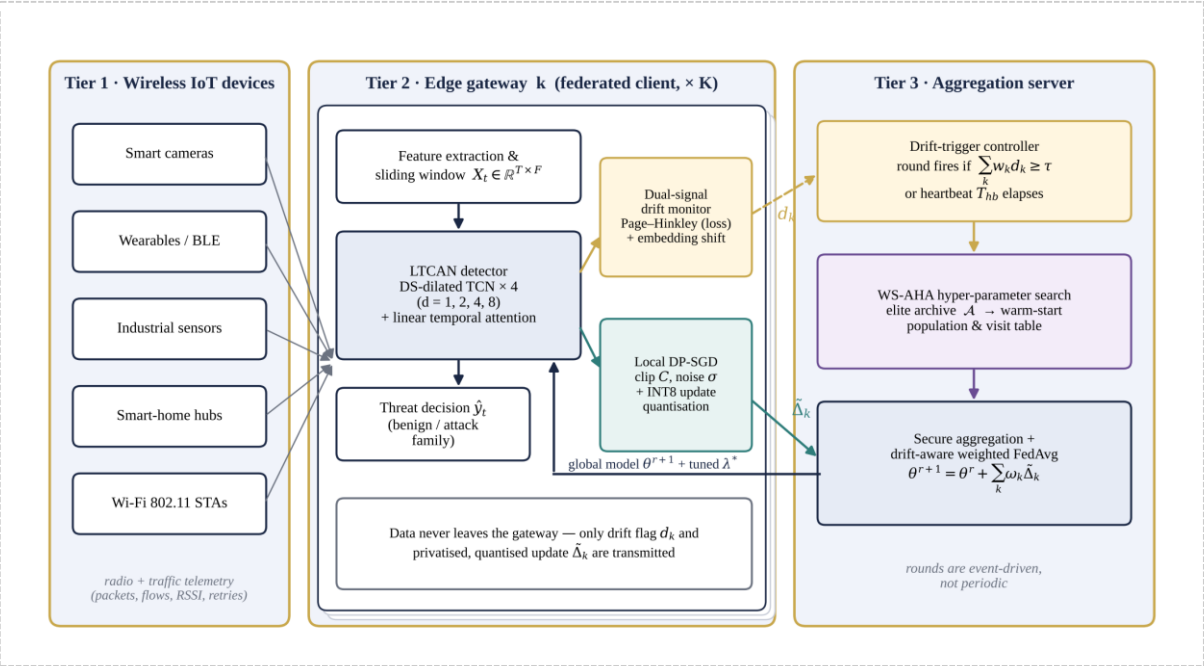


Figure 1. Overall architecture of DT-FedLite. Wireless IoT devices feed telemetry to edge gateways, which detect threats locally with LTCAN, monitor drift and transmit only drift flags and privatised, quantised updates; the server triggers rounds, re-tunes hyper-parameters with WS-AHA and performs drift-aware secure aggregation.

A. Research Design

The study follows a quantitative, experimental design: a new framework is specified, implemented and compared against established baselines on public benchmarks under a controlled streaming protocol, with ablations isolating each component.

System model. A network contains K edge gateways (clients). Gateway k observes a stream of traffic records from the wireless devices it serves; each record is a feature vector $x \in \mathbb{R}^F$ combining flow statistics (packet counts, inter-arrival times, flag counts, byte rates) and, where available, radio and MAC fields (received signal strength, frame subtype, retry flag). Records are grouped per source device and segmented into overlapping windows of length T with stride T/2:

$$X_t = [x_{t-T+1}, x_{t-T+2}, \ldots, x_t] \in \mathbb{R}^{T\times F} \qquad \ldots\ldots(1)$$

Each window carries a label $y_t \in$ □ (benign or an attack family) that becomes available only for a fraction p_ℓ of windows and after a delay of Δ_ℓ windows, reflecting analyst triage in a security operations centre. The joint distribution □$_k^t(X, y)$ may change over time and differs across clients. The goal is to learn global parameters θ that solve, at every time t,

$$min_\theta \Sigma_{k=1}^{K} (n_k/n) \, □_{(X,y)\sim□_{\{k}^t\}} [\, \ell(f_\theta(X), y) \,] \qquad \ldots\ldots(2)$$

subject to three constraints: raw records never leave their gateway and every released update satisfies (ε, δ)-differential privacy; per-window inference fits a latency budget of 2 ms and a memory budget of 8 MB on gateway hardware; and cumulative uplink traffic is minimised. Here n_k is the number of labelled windows in client k's replay buffer and $n = \Sigma_k n_k$.

Threat model. Adversaries launch network attacks against IoT devices (flooding, reconnaissance, spoofing, web and brute-force attacks, Mirai-style botnet activity and 802.11-specific attacks). The aggregation server is honest-but-curious: it follows the protocol but may try to infer client data from updates. Gateways are trusted to run the protocol correctly; Byzantine or poisoning clients are outside the scope of this paper and are discussed as a limitation in Section V.

B. Data Collection

Three public datasets were selected to cover consumer IoT, industrial IoT and the IEEE 802.11 radio layer. **CICIoT2023** [15] was captured from a topology of 105 IoT devices and contains 33 attacks grouped into seven families (DDoS, DoS, reconnaissance, web-based, brute force, spoofing and Mirai) described by 46 flow features; the family-level task with a benign class gives eight classes. **Edge-IIoTset** [16] is an IoT/IIoT testbed dataset with fourteen attack types across five threat categories plus normal traffic; the 15-class task on the authors' recommended feature subset is used. **AWID3** [17] records attacks against an IEEE 802.11 enterprise network; MAC-layer and radiotap fields were filtered to 48 non-identifying features and the traffic was grouped into normal

© 2026 VT International Press. All rights reserved.

plus seven attack categories. Identifier fields (IP and MAC addresses, SSIDs, timestamps) were used only to group records into per-device windows and were then removed from all feature vectors to prevent shortcut learning; no identifier is presented to the model.

Pre-processing. Numeric features were log-transformed where heavily skewed and standardised with statistics computed on each client's warm-up segment only, so that no information crossed client boundaries. Categorical fields were one-hot encoded and constant columns dropped. The resulting dimensionality F is listed in Table 2.

Streaming and federated partitioning. Records were replayed in capture order. Source devices were assigned to K clients using a Dirichlet distribution over class proportions with concentration $\alpha = 0.3$ [39], which produces strongly non-IID clients in which several attack families are present at only a few gateways. The first 10% of each stream served as a labelled warm-up; thereafter labels were revealed for $p_\ell = 10\%$ of windows after a delay of $\Delta_\ell = 20$ windows.

Drift injection. To obtain controlled ground truth, six drift events were injected into each stream of 3,000 global windows: four *sudden* events in which a held-out attack family appears simultaneously at a subset of clients; one *gradual* event in which the benign traffic mix shifts linearly over 200 windows towards a different device profile; and one *recurring* event in which a previously seen attack family returns after a long absence. Natural drift present in the captures was left untouched.

Table 2. Datasets and streaming configuration.

Property	CICIoT2023	Edge-IIoTset	AWID3
Network type	Consumer IoT (105 devices)	IoT / IIoT testbed	IEEE 802.11 enterprise
Classes used	8 (benign + 7 families)	15 (normal + 14 attacks)	8 (normal + 7 categories)
Input features F (after encoding)	46	61	48
Clients K / Dirichlet α	20 / 0.3	16 / 0.3	12 / 0.3
Window length T / stride	32 / 16	32 / 16	32 / 16
Global stream length (windows)	3,000	3,000	3,000
Injected drift events	4 sudden, 1 gradual, 1 recurring	4 sudden, 1 gradual, 1 recurring	4 sudden, 1 gradual, 1 recurring
Label rate p_ℓ / delay Δ_ℓ	10% / 20 windows	10% / 20 windows	10% / 20 windows

C. Lightweight Temporal Convolution–Attention Network (LTCAN)

LTCAN (Figure 2) maps a window X_t to class probabilities through four stages. A point-wise convolution first projects the F input features to C = 48 channels. Four residual blocks then extract multi-scale temporal patterns. Each block contains two depthwise-separable causal convolutions with kernel size k = 3 and dilation $d_\ell \in \{1, 2, 4, 8\}$. For channel c at time τ, the depthwise stage computes

$$\tilde{h}_{\tau,c}^{(\ell)} = \Sigma_{i=0}^{k-1} w_{c,i}^{(\ell)} \cdot h_{\tau - d_\ell \cdot i,\, c}^{(\ell)} \qquad \ldots\ldots(3)$$

and a point-wise 1 × 1 convolution $W_{pw}^{(\ell)}$ mixes channels. Factorising a standard convolution in this way reduces its parameter count from $k \cdot C^2$ to $k \cdot C + C^2$ [11]. Each block applies layer normalisation, GELU activation, dropout and a residual connection:

$$H^{(\ell+1)} = H^{(\ell)} + Dropout(\, GELU(\, LN(\, W_{pw}^{(\ell)}\, \tilde{H}^{(\ell)}\,)\,)\,) \qquad \ldots\ldots(4)$$

With two convolutions per block, the receptive field of the stack is

$$R = 1 + 2(k-1)\, \Sigma_{\ell=1}^{L} d_\ell = 1 + 2 \cdot 2 \cdot (1 + 2 + 4 + 8) = 61 \qquad \ldots\ldots(5)$$

time steps, which exceeds the window length T = 32, so every output position sees the full window while causality keeps the model usable for incremental, record-by-record scoring.

Convolutions capture local motifs such as bursts and periodic beacons, whereas many attacks are characterised by relations between distant records—for example, a scan followed by an exploit. LTCAN therefore adds a two-head *linear temporal attention* layer [21]. With queries Q, keys K and values V obtained by linear projection of the block output Z, and the feature map $\varphi(u) = \mathrm{elu}(u) + 1$, the attention output at position τ is

$$A_\tau = \varphi(Q_\tau)^T\, \Sigma_{s=1}^{T}\, \varphi(K_s)\, V_s^T \;/\; \varphi(Q_\tau)^T\, \Sigma_{s=1}^{T}\, \varphi(K_s) \qquad \ldots\ldots(6)$$

which costs $O(T \cdot d^2)$ rather than the $O(T^2 \cdot d)$ of softmax attention [20]. An attention-pooling layer then forms a window embedding as a weighted sum of time steps,

$$\alpha_\tau = softmax_\tau(\, v^T\, tanh(\, W_a\, z_\tau\,)\,), \quad \bar{z} = \Sigma_{\tau=1}^{T}\, \alpha_\tau\, z_\tau \qquad \ldots\ldots(7)$$

and a two-layer classifier (48 → 64 → |□|) produces logits. The pooling weights α_τ can serve as an indicative explanation signal, highlighting which records in a window most influenced an alert; their faithfulness as explanations is not evaluated in this study. Because attack families are highly imbalanced, LTCAN is trained with a class-balanced focal loss [40], [41],

$$\mathscr{L} = -\, \Sigma_{(X,y)}\, [\, (1 - \beta_{cb}) \,/\, (1 - \beta_{cb}^{\,n_y})\,]\, (1 - p_y)^{\gamma}\, \log p_y \qquad \ldots\ldots(8)$$

© 2026 VT International Press. All rights reserved.

where p_y is the predicted probability of the true class, n_y is the class count in the client's buffer, $\beta_{cb} = 0.999$ and $\gamma = 2$. The complete model has 38.6 k parameters and requires 2.14 MFLOPs per window. For deployment it is quantised to INT8 with per-channel weight scales [22].

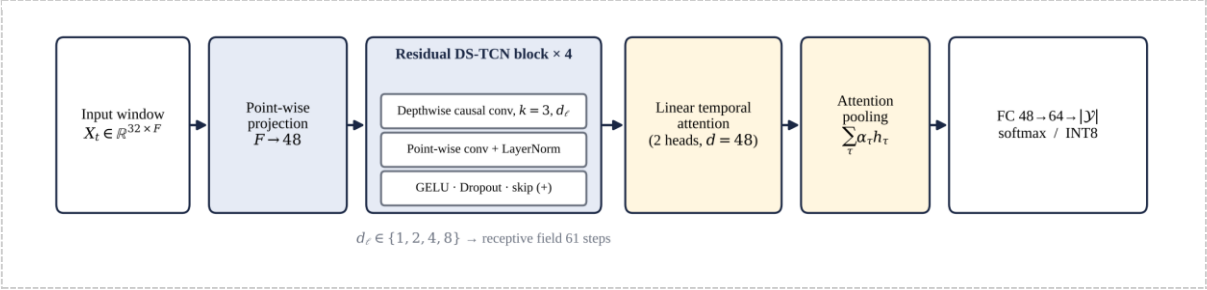


Figure 2. LTCAN detector. Four residual depthwise-separable dilated causal TCN blocks are followed by linear temporal attention, attention pooling and a compact classifier.

D. Dual-Signal Drift Detection and Event-Driven Triggering

Each client monitors two complementary statistics. The *supervised* signal applies the Page–Hinkley test [29] to the loss ℓ_s on windows whose delayed labels have arrived. With running mean $\bar{\ell}_s$ and tolerance δ_{PH},

$$m_t = \Sigma_{s \le t} (\ell_s - \bar{\ell}_s - \delta_{PH}), \quad M_t = min_{s \le t} m_s, \quad PH_t = m_t - M_t \qquad \ldots\ldots(9)$$

and an alarm is raised when $PH_t > \lambda_{PH}$. Because labels are sparse and late, this signal alone reacts slowly. The *unsupervised* signal therefore tracks the penultimate embedding $\bar{z}$ of *every* window. Let μ_{ref} and the diagonal covariance Σ_{ref} be estimated on the embeddings of the last stable period, and $\hat{\mu_t}$ the mean over the most recent M = 16 windows. The shift statistic

$$S_t = M \cdot (\hat{\mu_t} - \mu_{ref})^T \Sigma_{ref}^{-1} (\hat{\mu_t} - \mu_{ref}) \qquad \ldots\ldots(10)$$

is approximately χ^2-distributed with dimension equal to the embedding size when no drift occurs, so its threshold κ is set as a high quantile of that distribution. Because consecutive windows overlap (stride T/2), the averaged embeddings are not independent and the χ^2 reference is only nominal; the quantile is therefore treated as a tunable hyper-parameter (Table 3) and the realised false-alarm rate is reported empirically in Section IV-B. Client k summarises both signals in a normalised drift score

$$d_k(t) = min\{ 1, max(PH_t / \lambda_{PH}, S_t / \kappa) \} \qquad \ldots\ldots(11)$$

and transmits it to the server as a single byte whenever it changes by more than 0.05. The server fires a federated round when the population-weighted score $D(t) = \Sigma_k (n_k/n)\, d_k(t) \ge \tau$, or when at least a fraction ρ of clients report $d_k = 1$, which lets localised drift at a few gateways still trigger adaptation. A heartbeat round is also forced if $T_{hb} = 100$ windows elapse without one guaranteeing that slowly accumulating change is eventually absorbed. After a trigger, rounds continue in a short burst (up to eight rounds) until D(t) falls below $\tau/2$. Reference statistics μ_{ref} and Σ_{ref} are re-estimated on each client once its drift score has remained below 0.2 for 50 windows.

E. Privacy-Preserving, Drift-Aware Aggregation

In a triggered round r, each participating client receives θ^r and trains for E local epochs on its labelled replay buffer (a reservoir of 4,096 windows that preserves rare classes). Training uses DP-SGD [12]: per-example gradients g_i are clipped to norm C_{clip} and Gaussian noise is added, together with a FedProx proximal term [24] that limits divergence from the global model under non-IID data:

$$\theta_k \leftarrow \theta_k - \eta \, [\, (1/B)(\Sigma_{i \in \mathcal{B}} \, clip(g_i, C_{clip}) + \mathcal{N}(0, \sigma^2 C_{clip}^2 I)) + \mu_{prox}(\theta_k - \theta^r) \,] \qquad \ldots\ldots(12)$$

The privacy loss across all rounds is tracked with a Rényi differential privacy accountant [42]; with noise multiplier $\sigma = 0.9$ and sampling rate $B/n_k = 1/64$, the full 3,000-window deployment consumes $\varepsilon = 3.2$ at $\delta = 10^{-5}$ per client. The protected unit is a single window. Because consecutive windows overlap with stride T/2, each record contributes to at most two windows, so by group privacy [47] the corresponding record-level guarantee is $(2\varepsilon, 2e^{\wedge}\varepsilon\ \delta) \approx (6.4, 4.9 \times 10^{-4})$. The accountant treats the replay buffer at each step as the client's dataset; since reservoir sampling changes the buffer over time, these figures should be read as the standard accounting under that assumption rather than a worst-case bound. The model delta $\Delta_k = \theta_k - \theta^r$ is quantised to 8 bits with an unbiased stochastic rounding operator,

$$\tilde{\Delta_k} = s_k \cdot round_{stoch}(\Delta_k / s_k), \quad s_k = \|\Delta_k\|_\infty / 127 \qquad \ldots\ldots(13)$$

and uploaded through a secure-aggregation protocol [13], so that the server observes only the sum. Since quantisation is applied to an already privatised quantity, it does not weaken the guarantee. Clients that experienced stronger drift hold the most up-to-date view of the new distribution, so their contributions are up-weighted:

$$\omega_k = n_k(1 + \beta\, d_k) / \Sigma_j n_j(1 + \beta\, d_j), \quad \theta^{r+1} = \theta^r + \eta_s \Sigma_k \omega_k \tilde{\Delta_k} \qquad \ldots\ldots(14)$$

© 2026 VT International Press. All rights reserved.

The weights ω_k depend only on public scalars (n_k and d_k), so they can be applied inside secure aggregation by having each client pre-scale its masked update. With $\beta = 0$ the rule reduces to FedAvg [5].

F. Warm-Start Artificial Hummingbird Search (WS-AHA)

DT-FedLite exposes eleven hyper-parameters, collected in a vector λ (Table 3): learning rate, local epochs, channel width, number of blocks, attention heads, dropout, proximal coefficient, drift-weight β, Page–Hinkley threshold, embedding-shift quantile and trigger threshold τ. Integer and categorical dimensions are encoded as continuous coordinates in [0, 1] and rounded on decoding. A configuration is scored by

$$J(\lambda) = F1_{val}(\lambda) - \gamma_1 \cdot Lat(\lambda)/Lat_{max} - \gamma_2 \cdot Size(\lambda)/Size_{max} \qquad \ldots\ldots(15)$$

where $F1_{val}$ is the federated validation macro-F1 after a three-round proxy run on eight randomly chosen clients, Lat and Size are the measured INT8 latency and model size, $Lat_{max} = 2$ ms and $Size_{max} = 256$ KB, and $\gamma_1 = \gamma_2 = 0.05$. Validation F1 is computed on each client and only the scalar is shared. These scalars, like the one-byte drift scores d_k, are not privatised and are therefore outside the ε accounting above; hyper-parameter selection on private data can itself leak information [46], so a fully private variant would add noise to them or account for the tuning procedure explicitly.

Base search. AHA [14] maintains N hummingbirds, each associated with a food source (candidate solution) x_i, and a visit table VT whose entry $VT_{i,j}$ counts how long hummingbird i has not visited source j. In each iteration, with equal probability, a hummingbird performs *guided foraging* towards the target source $x_{i,tar}$ with the highest visit level,

$$v_i(t+1) = x_{i,tar}(t) + a \cdot D \cdot (x_i(t) - x_{i,tar}(t)), \quad a \sim \square(0, 1) \qquad \ldots\ldots(16)$$

or *territorial foraging* in its own neighbourhood,

$$v_i(t+1) = x_i(t) + b \cdot D \cdot x_i(t), \quad b \sim \square(0, 1) \qquad \ldots\ldots(17)$$

where D is a random axial, diagonal or omnidirectional flight-direction vector. A candidate v_i replaces x_i only if it improves fitness, and the visit table is updated accordingly. Every 2N iterations the worst source is replaced through *migration foraging*,

$$x_{wor}(t+1) = Lb + r \cdot (Ub - Lb), \quad r \sim U(0, 1) \qquad \ldots\ldots(18)$$

where Lb and Ub are the search bounds. AHA's visit table gives it a memory of *where* the swarm has recently been, a property that WS-AHA exploits.

Warm start. Consecutive drift episodes produce related optima: a new attack family rarely changes the best model width but often shifts the best learning rate or trigger sensitivity. WS-AHA therefore keeps an archive $\square$ of the top-q configurations (q = 5) from every previous search, and makes three changes. (1) *Population seeding:* a fraction $\varphi = 0.6$ of the N = 12 hummingbirds are initialised around archived elites, with a perturbation that grows with the drift severity $\bar{D}$ observed at the trigger,

$$x_i(0) = clip_{[Lb,Ub]}(a_{\pi(i)} + \zeta_0(1 + \bar{D}) (Ub - Lb) \odot \varepsilon_i), \quad \varepsilon_i \sim \square(0, I) \qquad \ldots\ldots(19)$$

where $a_{\pi(i)}$ is the archived elite assigned to hummingbird i and $\zeta_0 = 0.05$; the remaining 40% are drawn uniformly to retain exploration. (2) *Visit-table priming:* entries $VT_{i,j}(0)$ are set to $q_v = 2$ for seeded sources j and 0 otherwise, so that guided foraging initially visits the archived region before migrating outward. (3) *Fitness caching:* archived elites are re-evaluated once on the post-drift proxy, and their stale scores are discarded. The search stops when the best fitness improves by less than 0.001 over 10 evaluations or after 120 evaluations. The very first tuning, at deployment, runs as a cold-start AHA.

Table 3. WS-AHA search space and configuration selected on CICIoT2023.

Hyper-parameter	Search range	Selected
Learning rate η (log scale)	$10^{-4} – 10^{-2}$	1.8×10^{-3}
Local epochs E	{1, …, 5}	2
Channel width C	{16, 24, 32, 48, 64}	48
Residual blocks L	{2, 3, 4, 5}	4
Attention heads	{1, 2, 4}	2
Dropout	0.0 – 0.3	0.12
Proximal coefficient μ_{prox}	0.0 – 0.1	0.01
Drift weight β	0.0 – 2.0	0.8
Page–Hinkley threshold λ_{PH}	5 – 50	18
Embedding-shift quantile for κ	0.95 – 0.999	0.995
Trigger threshold τ	0.10 – 0.50	0.25

G. Overall Procedure and Complexity

Algorithm 1 summarises one deployment of DT-FedLite.

© 2026 VT International Press. All rights reserved.

Algorithm 1. DT-FedLite: drift-triggered federated training and detection

Input: K clients with streams, initial θ^0, archive $\mathcal{A} = \emptyset$, thresholds τ, ρ, T_{hb}
Server tunes λ^* with cold-start AHA; broadcasts θ^0, λ^*
for each window t **do**
  each client k: classify X_t with INT8 LTCAN; update PH_t (Eq. 9) with any newly labelled windows
  compute S_t (Eq. 10) and $d_k(t)$ (Eq. 11); send d_k if it changed by > 0.05
  server: $D(t) \leftarrow \Sigma_k (n_k/n)\, d_k(t)$
  if $D(t) \geq \tau$ **or** $|\{k : d_k = 1\}| \geq \rho K$ **or** $t - t_{last} \geq T_{hb}$ **then**
    if $\bar{D} \geq \tau$ (true drift) **then** $\lambda^* \leftarrow$ WS-AHA($\mathcal{A}$, $\bar{D}$) (Eqs. 15–19); update $\mathcal{A}$
    repeat (burst, ≤ 8 rounds): broadcast θ^r, λ^*
      each client: DP-SGD + proximal local training (Eq. 12); quantise $\tilde{\Delta}_k$ (Eq. 13)
      secure aggregation with drift-aware weights (Eq. 14) $\rightarrow \theta^{r+1}$
    until $D(t) < \tau/2$; $t_{last} \leftarrow t$
end for

Per window, LTCAN costs $O(T \cdot L \cdot (k \cdot C + C^2) + T \cdot d^2)$ operations, linear in window length. The drift monitor adds O(C) per window for the running embedding mean. Per triggered round, each client uploads P bytes for an INT8 update of a P-parameter model, so total uplink is $K \cdot P \cdot R_{trig}$, where R_{trig} is the number of triggered rounds; by contrast, periodic FP32 federation costs $4 \cdot K \cdot P \cdot R_{per}$. Between drift events, the local objective is the FedProx objective, for which convergence under bounded heterogeneity has been established [24]; the trigger only determines *when* that optimisation is resumed.

H. Experimental Settings, Baselines and Metrics

Implementation. Federated training was simulated in PyTorch with one process per client; DP-SGD and RDP accounting followed [12], [42]. On-device latency and memory were measured on a Raspberry Pi 4 Model B (quad-core Cortex-A72, 4 GB RAM) running an ONNX Runtime INT8 build, single thread, batch size 1, averaged over 10,000 windows after 500 warm-up runs. All results are means over five random seeds.

Baselines. Six baselines were implemented with the same pre-processing, partitions and streaming protocol: (1) a centralised non-private TCN trained on pooled data (an upper reference that violates the privacy constraint); (2) FedAvg with a two-layer LSTM (128 units); (3) FedProx with a 1D-CNN; (4) FedAvg with a two-layer transformer encoder (d = 96); (5) a DÏoT-style federated GRU [26]; and (6) Periodic-Fed-LTCAN, which uses the LTCAN detector with FP32 FedAvg every 10 windows, no drift trigger and hand-set hyper-parameters. Baselines (2)–(5) also federate every 10 windows (300 rounds). For drift analysis, a static LTCAN that is never updated after warm-up and a local-only LTCAN retrained by ADWIN [30] without federation were added. All baselines were tuned with TPE [36] under the same 120-evaluation budget

Metrics. Detection is reported as accuracy and macro-F1 on a held-out, chronologically final 15% of each client's data. Streaming behaviour is measured by prequential (test-then-train) macro-F1 over a sliding window of 50 windows; mean and minimum prequential F1 in the 200 windows following each drift event; and *recovery time*, the number of windows until prequential F1 returns to within two points of its pre-drift level. Cost is measured by the number of federated rounds, cumulative uplink megabytes, parameter count, model size, MFLOPs, latency and peak memory. Privacy is reported as ε at $\delta = 10^{-5}$ and as the AUC of a loss-threshold membership-inference attack [43].

IV. RESULTS

This section reports detection accuracy, adaptation to drift, communication and on-device cost, hyper-parameter search efficiency, privacy and ablations. Unless stated otherwise, DT-FedLite operates at $\varepsilon = 3.2$.

A. Detection Performance

Table 4 compares final detection performance. DT-FedLite achieves macro-F1 of 96.21%, 97.48% and 95.36% on CICIoT2023, Edge-IIoTset and AWID3, exceeding the strongest federated baseline (Periodic-Fed-LTCAN) by 2.36, 2.37 and 2.72 points and the best conventional federated model (FedAvg-Transformer-lite) by 4.45, 3.54 and 4.88 points. It remains within 0.67, 0.44 and 0.74 points of the non-private centralised TCN, despite differential privacy and non-IID partitioning. The gap between accuracy and macro-F1 is largest for the recurrent baselines, whose errors concentrate on rare families such as web attacks and brute force on CICIoT2023 and key-reinstallation attacks on AWID3. Across the five seeds, the standard deviation of DT-FedLite's macro-F1 did not exceed 0.21 points, and its advantage over every federated baseline was significant under a Wilcoxon signed-rank test over seed–dataset pairs ($p < 0.01$).

Table 4. Detection performance on the held-out final segment (%; mean of five seeds).

Method	CICIoT2023		Edge-IIoTset		AWID3	
	Acc.	Macro-F1	Acc.	Macro-F1	Acc.	Macro-F1

© 2026 VT International Press. All rights reserved.

Centralised TCN (non-private ref.)	99.21	96.88	99.47	97.92	98.85	96.10
FedAvg-LSTM	96.12	88.47	97.30	91.06	95.44	87.93
FedProx-1D-CNN	96.85	90.12	97.92	92.85	96.07	89.20
FedAvg-Transformer-lite	97.40	91.76	98.21	93.94	96.52	90.48
DÏoT-style FedGRU	96.03	88.10	97.05	90.42	95.19	87.31
Periodic-Fed-LTCAN	98.02	93.85	98.66	95.11	97.31	92.64
DT-FedLite (proposed, ε = 3.2)	**98.94**	**96.21**	**99.32**	**97.48**	**98.47**	**95.36**

B. Drift Adaptation and Communication Cost

Figure 3(a) traces prequential macro-F1 on the CICIoT2023 stream, and Table 5 summarises the six drift events. The static detector loses 5 to 17 points at every sudden event and never recovers, confirming that a one-off model is inadequate. Periodic federation eventually restores accuracy, but its fixed 10-window cadence and equal weighting of clients dilute the few gateways that first observe a new attack, giving recovery times of 25.7 windows for Periodic-Fed-LTCAN and 42.0 for FedAvg-LSTM. DT-FedLite detects drift with a mean delay of 2.6 windows, fires a burst of rounds immediately and up-weights the affected clients, recovering within 8.0 windows on average and keeping the post-drift minimum at 89.98% versus 78.64% for periodic federation. The local-only ADWIN baseline adapts reasonably at clients that see the new attack but cannot transfer that knowledge, so other clients remain vulnerable when the campaign spreads.

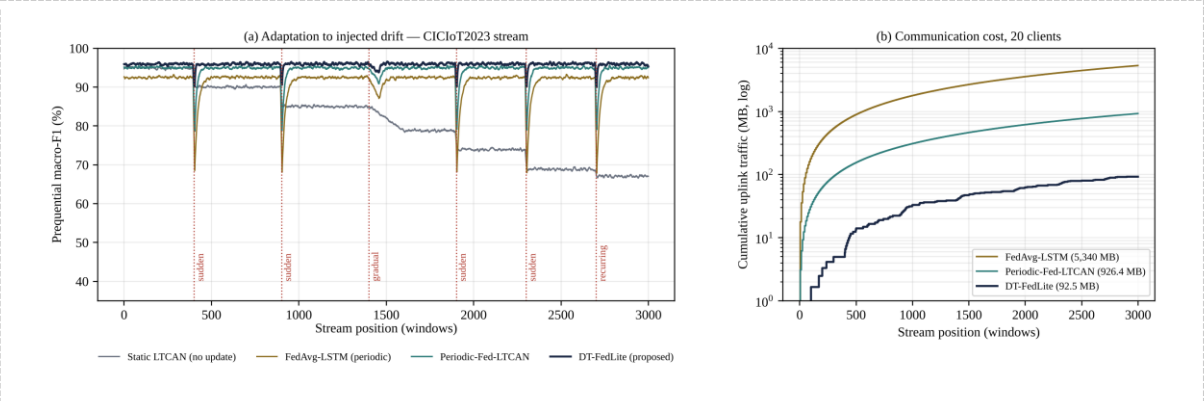


Figure 3. (a) Prequential macro-F1 on the CICIoT2023 stream with six injected drift events (dotted lines). (b) Cumulative uplink traffic for 20 clients on a logarithmic scale.

Table 5. Drift adaptation and communication on the CICIoT2023 stream (20 clients, 3,000 windows).

Method	Mean F1 after drift (%)	Min. F1 after drift (%)	Recovery (windows)	Fed. rounds	Uplink (MB)
Static LTCAN (no update)	77.85	66.61	not recovered	0	0.0
Local-only LTCAN + ADWIN	93.08	77.12	21.5	0	0.0
FedAvg-LSTM (periodic)	90.08	67.85	42.0	300	5,340.0
FedAvg-Transformer-lite (periodic)	91.77	72.20	33.8	300	5,457.6
Periodic-Fed-LTCAN	93.79	78.64	25.7	300	926.4
DT-FedLite (proposed)	**95.71**	**89.98**	**8.0**	**112**	**92.5**

Event-driven rounds also reduce communication sharply (Figure 3(b)). DT-FedLite used 112 rounds instead of 300: 47 were triggered by the injected drift events, 30 were heartbeats and 35 responded to alarms not aligned with an injected event, most of which coincided with natural shifts in benign traffic. Together with INT8 updates, this reduced uplink traffic to 92.5 MB, 90.0% below Periodic-Fed-LTCAN and 98.3% below FedAvg-LSTM. The dual-signal monitor was decisive: the Page–Hinkley test alone had a mean detection delay of 9.8 windows, because only 10% of labels arrive and they arrive 20 windows late, while the embedding statistic alone detected drift in 3.1 windows but raised 2.7 false alarms per 1,000 windows. Their combination achieved a 2.6-window delay with 0.9 false alarms per 1,000 windows. Similar trends were observed on Edge-IIoTset (recovery 7.1 vs. 23.9 windows; 104 vs. 300 rounds) and AWID3 (recovery 9.4 vs. 28.2 windows; 121 vs. 300 rounds). Part of the uplink saving comes from quantisation rather than triggering: an INT8 variant of periodic federation would transmit 41.3 KB × 20 clients × 300 rounds = 247.8 MB (computed, not measured), so event-driven triggering alone accounts for a 62.7% reduction relative to that reference.

C. On-Device Efficiency

Table 6 reports model cost on the Raspberry Pi 4 gateway. LTCAN has 5.8× fewer parameters and 6.7× fewer FLOPs than the LSTM and transformer baselines. After INT8 quantisation it occupies 41.3 KB, classifies a window in 0.58 ms and peaks at 3.9 MB of memory, comfortably inside the 2 ms and 8 MB budgets; quantisation reduced macro-F1 by only 0.06 points. At the observed peak rate of about 450 windows per second per gateway

© 2026 VT International Press. All rights reserved.

on CICIoT2023, LTCAN uses roughly 26% of a single core, leaving headroom for packet capture and feature extraction.

Table 6. Model cost on a Raspberry Pi 4 Model B gateway (single thread, batch size 1, T = 32).

Model	Params (k)	Size (KB)	MFLOPs / window	Latency (ms)	Peak RAM (MB)
LSTM (2 × 128)	222.5	890.0	14.26	4.31	22.7
1D-CNN	118.7	474.8	7.58	1.94	14.2
Transformer-lite (2 layers, d = 96)	227.4	909.6	14.93	5.12	26.5
LTCAN, FP32	38.6	154.4	2.14	0.97	6.3
LTCAN, INT8 (deployed)	**38.6**	**41.3**	**2.14**	**0.58**	**3.9**

D. Hyper-Parameter Search Efficiency

Table 7 and Figure 4(a) compare optimisers for the re-tuning that follows each drift event, averaged over the six events of the CICIoT2023 stream. Under the same 120-evaluation budget, cold-start AHA reached a higher fitness (0.926) than TPE, GWO, PSO and random search, consistent with the exploration–exploitation balance provided by its visit table. WS-AHA started from a fitness of 0.906—already close to the cold-start optimum—and reached 99% of its final value after 19 evaluations instead of 62, a 69% reduction that shortened re-tuning from 95 to 29 minutes of simulated federated time. It also converged to a better final configuration (J = 0.934), because the archive concentrates the search in a region already known to suit the deployment. The proxy runs also consume communication that is not included in the 112 rounds and 92.5 MB of Table 5. Each fitness evaluation is a three-round run on eight clients, so 19 evaluations per drift event correspond to 57 proxy rounds and, at the deployed INT8 size of 41.3 KB, about 18.8 MB of uplink; over the six CICIoT2023 events this is roughly 342 proxy rounds and 113 MB. Including it, DT-FedLite's total uplink of about 205.5 MB remains 77.8% below Periodic-Fed-LTCAN, although the baselines' one-off TPE tuning is likewise excluded from their totals.

Table 7. Hyper-parameter re-tuning after drift on CICIoT2023 (mean over six drift events; budget 120 evaluations).

Optimiser	Best fitness J	Evaluations to 99% of best	Time to 99% (min)	Final macro-F1 (%)
Random search	0.902	104	161	94.12
TPE (Bayesian)	0.921	71	109	95.08
PSO	0.917	83	128	94.86
GWO	0.919	78	120	94.97
AHA (cold start)	0.926	62	95	95.62
WS-AHA (proposed)	**0.934**	**19**	**29**	**96.21**

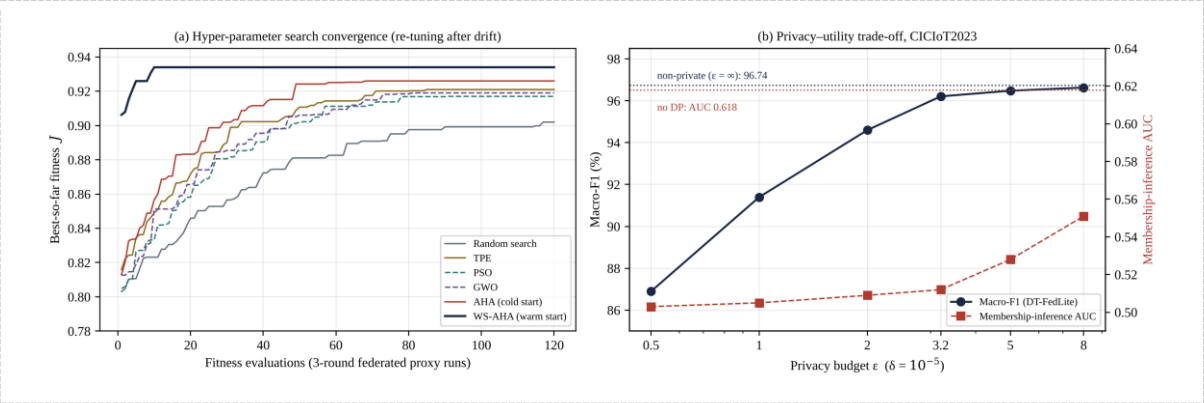


Figure 4. (a) Best-so-far fitness during post-drift re-tuning. (b) Macro-F1 and membership-inference AUC of DT-FedLite as a function of the privacy budget ε.

E. Privacy–Utility Trade-Off

Figure 4(b) varies the privacy budget. Macro-F1 is 96.74% without differential privacy, falls by only 0.53 points at ε = 3.2 and by 2.14 points at ε = 2, and declines more steeply below ε = 1. A loss-threshold membership-inference attack [43] achieved an AUC of 0.618 against the non-private model but only 0.512 at ε = 3.2—close to random guessing—confirming that the chosen operating point provides meaningful empirical protection at a small cost in accuracy.

F. Ablation Study

Table 8 removes or replaces one component at a time on CICIoT2023. Removing linear attention costs 1.34 points of final macro-F1, showing that long-range relations matter beyond what the dilated convolutions capture.

An ISO 9001:2015 Certified Journal © 2026 VT International Press. All rights reserved.

Replacing depthwise-separable with standard convolutions leaves accuracy essentially unchanged (+0.09) while multiplying parameters by 4.1, validating the factorisation. Replacing the drift trigger with periodic federation preserves final accuracy but reduces post-drift F1 by 1.40 points while using 2.7× more rounds. Using only the supervised Page–Hinkley signal saves rounds but detects drift late, lowering post-drift F1 by 1.65 points; using only the embedding signal causes extra false-alarm rounds. Drift-aware weighting contributes 0.68 points of post-drift F1, WS-AHA contributes 1.51 points of final F1 over hand-set hyper-parameters, and the warm start itself adds 0.59 points over cold-start AHA under the same budget. Differential privacy at $\varepsilon = 3.2$ costs 0.53 points. The periodic variant in this table differs from Periodic-Fed-LTCAN in Table 4, which additionally uses hand-set hyper-parameters (Section III-H); WS-AHA alone accounts for 1.51 points of final macro-F1 here, which explains most of the gap between 96.12% and 93.85%.

Table 8. Ablation study on CICIoT2023.

Variant	Final macro-F1 (%)	Mean F1 after drift (%)	Fed. rounds	Params (k)
DT-FedLite (full)	**96.21**	**95.71**	**112**	**38.6**
– linear attention (TCN + average pooling)	94.87	94.52	118	29.4
Standard instead of DS convolutions	96.30	95.66	112	158.9
– drift trigger (periodic, every 10 windows)	96.12	94.31	300	38.6
Page–Hinkley signal only	96.18	94.06	89	38.6
Embedding-shift signal only	96.15	95.12	147	38.6
– drift-aware weighting ($\beta = 0$)	96.03	95.03	126	38.6
– WS-AHA (hand-set hyper-parameters)	94.70	94.47	131	38.6
WS-AHA → cold-start AHA (same budget)	95.62	95.18	117	38.6
– differential privacy ($\varepsilon = \infty$)	96.74	96.02	112	38.6

V. DISCUSSION

The results support the central premise of DT-FedLite: in a streaming IoT network, *when* clients communicate matters as much as *what* they communicate. Periodic federation spends most of its rounds while traffic is stationary and too few of them in the critical period after a new attack appears. Tying rounds to a drift signal concentrates communication where it changes the model, which is why DT-FedLite simultaneously recovers 3.2× faster and transmits 10× less than Periodic-Fed-LTCAN. The drift-aware weighting is a lightweight complement to that idea: under strongly non-IID partitions, a new attack family is first seen by a handful of gateways, and plain FedAvg averages their signal away. Up-weighting high-drift clients acts as a short-lived, data-driven form of personalisation without maintaining per-client models.

The detector results show that temporal modelling need not be expensive. Dilated causal convolutions supply a receptive field covering the whole window, and linear attention supplies global context at linear cost; together they outperform recurrent and transformer baselines that are an order of magnitude larger. This contrasts with the common assumption in federated intrusion detection that recurrent units are necessary for sequence awareness [26], [27]. The small model has a second benefit specific to federated settings: every update is small, so the cost of each additional round—and therefore the cost of reacting quickly—is low.

The hyper-parameter study illustrates why warm starting suits drift. Consecutive optima are correlated, as also observed in meta-learned Bayesian optimisation [37], and AHA's visit table provides a natural place to encode that prior: priming it makes the swarm revisit known-good regions first, while the uniformly initialised 40% and the migration step retain the ability to escape when drift has genuinely moved the optimum. The fact that cold-start AHA outperformed PSO, GWO and TPE under the same budget is consistent with the original evaluation of AHA [14], but it should not be over-generalised given the no-free-lunch result [38]; the more robust finding is the large gain from warm starting itself.

From a deployment perspective, the framework is practical for gateway-class hardware: the INT8 model fits in tens of kilobytes, inference uses about a quarter of one core at peak load, drift flags cost one byte, and the privacy budget for a full deployment stays at $\varepsilon = 3.2$. Attention-pooling weights can be surfaced in alerts as a hint to analysts about which records influenced a detection (subject to validating their faithfulness), which supports triage and, in turn, the supply of the delayed labels on which the supervised drift signal depends.

Limitations. Several limitations qualify these conclusions. First, although natural drift was retained, the six principal drift events were injected by re-ordering public captures; live deployments may exhibit slower, overlapping or adversarially timed drift. Second, the labelling model—10% of windows labelled after 20 windows—is a simplification of analyst behaviour, and the supervised signal will weaken further if labels are scarcer. Third, the threat model excludes malicious clients; drift-aware weighting could be exploited by a poisoning client that fakes high drift, so robust aggregation and drift-score attestation are needed before adversarial deployment. Fourth, differential privacy is accounted per client at the window level (record level via group privacy, Section III-E), hyper-parameter-tuning scalars and drift scores are not privatised, and device-level or client-level guarantees would require more noise. Fifth, the drift-adaptive federated methods closest to this

© 2026 VT International Press. All rights reserved.

work [31], [32], [44] were not re-implemented as baselines, and sensitivity to the label rate, label delay and trigger threshold was not evaluated; both are needed to establish the generality of the gains. Finally, latency was measured on a single hardware platform, and the federation was simulated rather than run over real wireless backhaul with stragglers and dropouts.

VI. CONCLUSION

This paper presented DT-FedLite, a drift-triggered federated framework for privacy-preserving threat detection in streaming IoT wireless networks. It combines a 38.6 k-parameter temporal convolution–attention detector, a dual-signal drift monitor that converts drift into event-driven federated rounds, a differentially private, quantised and securely aggregated update path with drift-aware weighting, and a warm-start artificial hummingbird algorithm that re-tunes hyper-parameters after each drift episode. Across CICIoT2023, Edge-IIoTset and AWID3 streams with injected drift, DT-FedLite achieved the highest macro-F1 among federated methods at $\varepsilon = 3.2$, recovered from drift about three times faster than periodic federation, reduced uplink traffic by 90%, cut re-tuning effort by 69% and classified each window in 0.58 ms on a Raspberry Pi 4.

Future work will extend the framework with Byzantine-robust, drift-attested aggregation; combine the trigger with clustered or personalised federation for persistently heterogeneous sites; incorporate semi-supervised and active-learning signals to reduce dependence on delayed labels; and validate the system in a live testbed with real wireless backhaul, heterogeneous gateway hardware and red-team-generated drift.

REFERENCES

[1] Ferrag, M. A., Maglaras, L., Moschoyiannis, S., & Janicke, H. (2020). Deep learning for cyber security intrusion detection: Approaches, datasets, and comparative study. *Journal of Information Security and Applications, 50*, 102419.

[2] Kolias, C., Kambourakis, G., Stavrou, A., & Gritzalis, S. (2016). Intrusion detection in 802.11 networks: Empirical evaluation of threats and a public dataset. *IEEE Communications Surveys & Tutorials, 18*(1), 184–208. https://doi.org/10.1109/COMST.2015.2402161

[3] Gama, J., Žliobaitė, I., Bifet, A., Pechenizkiy, M., & Bouchachia, A. (2014). A survey on concept drift adaptation. *ACM Computing Surveys, 46*(4), Article 44. https://doi.org/10.1145/2523813

[4] Lu, J., Liu, A., Dong, F., Gu, F., Gama, J., & Zhang, G. (2019). Learning under concept drift: A review. *IEEE Transactions on Knowledge and Data Engineering, 31*(12), 2346–2363. https://doi.org/10.1109/TKDE.2018.2876857

[5] McMahan, B., Moore, E., Ramage, D., Hampson, S., & Agüera y Arcas, B. (2017). Communication-efficient learning of deep networks from decentralized data. In *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics (AISTATS)*, PMLR 54 (pp. 1273–1282).

[6] Kairouz, P., McMahan, H. B., Avent, B., Bellet, A., Bennis, M., Bhagoji, A. N., et al. (2021). Advances and open problems in federated learning. *Foundations and Trends in Machine Learning, 14*(1–2), 1–210. https://doi.org/10.1561/2200000083

[7] Rahman, S. A., Tout, H., Talhi, C., & Mourad, A. (2020). Internet of Things intrusion detection: Centralized, on-device, or federated learning? *IEEE Network, 34*(6), 310–317.

[8] Campos, E. M., Saura, P. F., González-Vidal, A., Hernández-Ramos, J. L., Bernabé, J. B., Baldini, G., & Skarmeta, A. (2022). Evaluating federated learning for intrusion detection in Internet of Things: Review and challenges. *Computer Networks, 203*, 108661.

[9] Khodak, M., Tu, R., Li, T., Li, L., Balcan, M.-F., Smith, V., & Talwalkar, A. (2021). Federated hyperparameter tuning: Challenges, baselines, and connections to weight-sharing. In *Advances in Neural Information Processing Systems 34 (NeurIPS 2021)*.

[10] Bai, S., Kolter, J. Z., & Koltun, V. (2018). *An empirical evaluation of generic convolutional and recurrent networks for sequence modeling* (arXiv:1803.01271). arXiv. https://arxiv.org/abs/1803.01271

[11] Howard, A. G., Zhu, M., Chen, B., Kalenichenko, D., Wang, W., Weyand, T., Andreetto, M., & Adam, H. (2017). *MobileNets: Efficient convolutional neural networks for mobile vision applications* (arXiv:1704.04861). arXiv. https://arxiv.org/abs/1704.04861

[12] Abadi, M., Chu, A., Goodfellow, I., McMahan, H. B., Mironov, I., Talwar, K., & Zhang, L. (2016). Deep learning with differential privacy. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security* (pp. 308–318). https://doi.org/10.1145/2976749.2978318

[13] Bonawitz, K., Ivanov, V., Kreuter, B., Marcedone, A., McMahan, H. B., Patel, S., Ramage, D., Segal, A., & Seth, K. (2017). Practical secure aggregation for privacy-preserving machine learning. In *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security* (pp. 1175–1191). https://doi.org/10.1145/3133956.3133982

[14] Zhao, W., Wang, L., & Mirjalili, S. (2022). Artificial hummingbird algorithm: A new bio-inspired optimizer with its engineering applications. *Computer Methods in Applied Mechanics and Engineering, 388*, 114194. https://doi.org/10.1016/j.cma.2021.114194

[15] Neto, E. C. P., Dadkhah, S., Ferreira, R., Zohourian, A., Lu, R., & Ghorbani, A. A. (2023). CICIoT2023: A real-time dataset and benchmark for large-scale attacks in IoT environment. *Sensors, 23*(13), 5941. https://doi.org/10.3390/s23135941

© 2026 VT International Press. All rights reserved.

[16] Ferrag, M. A., Friha, O., Hamouda, D., Maglaras, L., & Janicke, H. (2022). Edge-IIoTset: A new comprehensive realistic cyber security dataset of IoT and IIoT applications for centralized and federated learning. *IEEE Access, 10*, 40281–40306. https://doi.org/10.1109/ACCESS.2022.3165809

[17] Chatzoglou, E., Kambourakis, G., & Kolias, C. (2021). Empirical evaluation of attacks against IEEE 802.11 enterprise networks: The AWID3 dataset. *IEEE Access, 9*, 34188–34205.

[18] Moustafa, N. (2021). A new distributed architecture for evaluating AI-based security systems at the edge: Network TON_IoT datasets. *Sustainable Cities and Society, 72*, 102994. https://doi.org/10.1016/j.scs.2021.102994

[19] Lea, C., Flynn, M. D., Vidal, R., Reiter, A., & Hager, G. D. (2017). Temporal convolutional networks for action segmentation and detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (pp. 156–165).

[20] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., & Polosukhin, I. (2017). Attention is all you need. In *Advances in Neural Information Processing Systems 30 (NIPS 2017)* (pp. 5998–6008).

[21] Katharopoulos, A., Vyas, A., Pappas, N., & Fleuret, F. (2020). Transformers are RNNs: Fast autoregressive transformers with linear attention. In *Proceedings of the 37th International Conference on Machine Learning (ICML)*, PMLR 119 (pp. 5156–5165).

[22] Jacob, B., Kligys, S., Chen, B., Zhu, M., Tang, M., Howard, A., Adam, H., & Kalenichenko, D. (2018). Quantization and training of neural networks for efficient integer-arithmetic-only inference. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (pp. 2704–2713).

[23] Mirsky, Y., Doitshman, T., Elovici, Y., & Shabtai, A. (2018). Kitsune: An ensemble of autoencoders for online network intrusion detection. In *Proceedings of the Network and Distributed System Security Symposium (NDSS 2018)*. Internet Society.

[24] Li, T., Sahu, A. K., Zaheer, M., Sanjabi, M., Talwalkar, A., & Smith, V. (2020). Federated optimization in heterogeneous networks. *Proceedings of Machine Learning and Systems, 2*, 429–450.

[25] Karimireddy, S. P., Kale, S., Mohri, M., Reddi, S., Stich, S., & Suresh, A. T. (2020). SCAFFOLD: Stochastic controlled averaging for federated learning. In *Proceedings of the 37th International Conference on Machine Learning (ICML)*, PMLR 119 (pp. 5132–5143).

[26] Nguyen, T. D., Marchal, S., Miettinen, M., Fereidooni, H., Asokan, N., & Sadeghi, A.-R. (2019). DÏoT: A federated self-learning anomaly detection system for IoT. In *Proceedings of the 39th IEEE International Conference on Distributed Computing Systems (ICDCS)* (pp. 756–767).

[27] Mothukuri, V., Khare, P., Parizi, R. M., Pouriyeh, S., Dehghantanha, A., & Srivastava, G. (2022). Federated-learning-based anomaly detection for IoT security attacks. *IEEE Internet of Things Journal, 9*(4), 2545–2554.

[28] Gama, J., Medas, P., Castillo, G., & Rodrigues, P. (2004). Learning with drift detection. In *Advances in Artificial Intelligence – SBIA 2004* (Lecture Notes in Computer Science, Vol. 3171, pp. 286–295). Springer.

[29] Page, E. S. (1954). Continuous inspection schemes. *Biometrika, 41*(1/2), 100–115.

[30] Bifet, A., & Gavaldà, R. (2007). Learning from time-changing data with adaptive windowing. In *Proceedings of the 2007 SIAM International Conference on Data Mining* (pp. 443–448).

[31] Canonaco, G., Bergamasco, A., Mongelluzzo, A., & Roveri, M. (2021). Adaptive federated learning in presence of concept drift. In *Proceedings of the International Joint Conference on Neural Networks (IJCNN)* (pp. 1–7). IEEE.

[32] Jothimurugesan, E., Hsieh, K., Wang, J., Joshi, G., & Gibbons, P. B. (2023). Federated learning under distributed concept drift. In *Proceedings of the 26th International Conference on Artificial Intelligence and Statistics (AISTATS)*, PMLR 206.

[33] Kennedy, J., & Eberhart, R. (1995). Particle swarm optimization. In *Proceedings of ICNN'95 – International Conference on Neural Networks* (Vol. 4, pp. 1942–1948). IEEE.

[34] Mirjalili, S., Mirjalili, S. M., & Lewis, A. (2014). Grey wolf optimizer. *Advances in Engineering Software, 69*, 46–61. https://doi.org/10.1016/j.advengsoft.2013.12.007

[35] Mirjalili, S., & Lewis, A. (2016). The whale optimization algorithm. *Advances in Engineering Software, 95*, 51–67. https://doi.org/10.1016/j.advengsoft.2016.01.008

[36] Bergstra, J., Bardenet, R., Bengio, Y., & Kégl, B. (2011). Algorithms for hyper-parameter optimization. In *Advances in Neural Information Processing Systems 24 (NIPS 2011)* (pp. 2546–2554).

[37] Feurer, M., Springenberg, J. T., & Hutter, F. (2015). Initializing Bayesian hyperparameter optimization via meta-learning. In *Proceedings of the AAAI Conference on Artificial Intelligence, 29*(1) (pp. 1128–1135).

[38] Wolpert, D. H., & Macready, W. G. (1997). No free lunch theorems for optimization. *IEEE Transactions on Evolutionary Computation, 1*(1), 67–82.

[39] Hsu, T.-M. H., Qi, H., & Brown, M. (2019). *Measuring the effects of non-identical data distribution for federated visual classification* (arXiv:1909.06335). arXiv. https://arxiv.org/abs/1909.06335

[40] Lin, T.-Y., Goyal, P., Girshick, R., He, K., & Dollár, P. (2017). Focal loss for dense object detection. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)* (pp. 2980–2988).

[41] Cui, Y., Jia, M., Lin, T.-Y., Song, Y., & Belongie, S. (2019). Class-balanced loss based on effective number of samples. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (pp. 9268–9277).

[42] Mironov, I. (2017). Rényi differential privacy. In *Proceedings of the IEEE 30th Computer Security Foundations Symposium (CSF)* (pp. 263–275).

[43] Shokri, R., Stronati, M., Song, C., & Shmatikov, V. (2017). Membership inference attacks against machine learning models. In *Proceedings of the 2017 IEEE Symposium on Security and Privacy (SP)* (pp. 3–18).

© 2026 VT International Press. All rights reserved.

[44] Chow, T., Raza, U., Mavromatis, I., & Khan, A. (2023). *FLARE: Detection and mitigation of concept drift for federated learning based IoT deployments* (arXiv:2305.08504). arXiv. https://arxiv.org/abs/2305.08504

[45] Rehman, M. U., Bahsi, H., & Kalakoti, R. (2026). *Incremental federated learning for intrusion detection in IoT networks under evolving threat landscape* (arXiv:2603.10776). arXiv. https://arxiv.org/abs/2603.10776

[46] Papernot, N., & Steinke, T. (2022). Hyperparameter tuning with Rényi differential privacy. In *Proceedings of the International Conference on Learning Representations (ICLR 2022)*. https://arxiv.org/abs/2110.03620

[47] Dwork, C., & Roth, A. (2014). The algorithmic foundations of differential privacy. *Foundations and Trends in Theoretical Computer Science*, *9*(3–4), 211–407. https://doi.org/10.1561/0400000042

© 2026 VT International Press. All rights reserved.