International Journal of Computers and Informatics (IJCI)
Vol. (5), No. (8)
IJCI
August 2026
المجلة الدولية للحاسبات والمعلوماتية
الإصدار (5)، العدد (8)

ALEIDF: An Adaptive Lightweight and Explainable Hybrid Deep Learning Framework for Intrusion Detection in Resource-Constrained IoT Networks

Sally D. Hamdi
Assistant Lecturer, Department of Computer Engineering, College of Engineering, Mustansiriyah University, Baghdad, Iraq
sallydakheel@uomustansiriyah.edu.iq

Hussein M. Farhood
Assistant Lecturer, Department of Computer Engineering, College of Engineering, Mustansiriyah University, Baghdad, Iraq
hussein.m4f@uomustansiriyah.edu.iq

Marwa Y. Mohammed
Assistant Lecturer, Department of Computer Engineering, College of Engineering, Mustansiriyah University, Baghdad, Iraq
marwa@uomustansiriyah.edu.iq

Abstract

The proliferation of Internet of Things (IoT) networks in smart homes, healthcare, industrial systems, and critical infrastructure has greatly extended the attack surface, and the deep learning models that are best suited to detecting these attacks are challenging to deploy on resource-limited edge devices, difficult to trust because they are black-box, and difficult to maintain effectively because traffic and attack distributions are evolving over time. Lightweight, explainable, and adaptive intrusion detection have been researched as largely distinct lines of work so far. This paper proposes a unified framework (ALEIDF) which consolidates eight tightly coupled modules, namely: Adaptive Feature Evolution Module (AFE), Hybrid Deep Learning Engine (HDLE), Adaptive Threat Memory (ATM), Explainability Module (XM), Decision Engine (DE), Resource Optimization Module (ROM), and Online Learning Module (OLM). The ALEIDF is equipped with feature level-drift detection, as well as retraining triggers; externalizes threat knowledge into an episodic memory query; calibrates detection confidence against this memory; and scopes the generation of explanations towards the drift adapted feature subset, which address the accuracy-efficiency gap, efficiency-explainability gap, and adaptability-knowledge-retention gap pointed out in the recent IoT-IDS literature. In a cross-dataset test with UNSW-NB15 and X-IIoTID, ALEIDF achieves a macro F1 score of about 97.7%, fast inference latency of < 10ms on microcontroller-class hardware and about 40% faster recovery from simulated concept drift than federated-incremental baselines, while costing just 10% as much relative to explanation generation as full-feature SHAP. Having identified joint integration of efficiency,

explainability and adaptability as an open field in recent surveys regarding IoT network security research, ALEIDF is well positioned to further this integration.

Keywords: IoT, IDS, Lightweight Deep Learning, Adaptive Learning, Federated Learning, Network Security.

1. Introduction

The increasing number of Internet of Things (IoT) devices in smart homes, healthcare, industrial automation, and critical infrastructure has created a vast cyberattack surface. There have been large-scale benchmarking efforts, like CICIoT2023, that have shown that contemporary IoT networks are vulnerable to a wide range of attack types, including DDoS, DoS, reconnaissance, web-based, and Mirai-type botnet attacks, under realistic and heterogeneous traffic conditions [1].

While deep learning has demonstrated great potential for IoT-based Network Intrusion Detection Systems (NIDS), three distinct but closely related drawbacks remain. First, high-accuracy hybrid architectures are based on complex architectures that consume high computing resources and are not suitable for resource-limited edge devices, which encourages light designs like quantization [2], TinyML [3] and knowledge distillation [4]. Second, due to the black-box nature of deep models, there is less trust in the model, which has led to the adoption of techniques for explainable AI (XAI) like SHAP and LIME [5], [6] and [7]. Third, the distributions of IoT traffic and attacks are not static, but evolve over time (concept drift), federated or incremental learning studies deal with adaptability, but mainly consider it as being separate from efficiency, explainability and adaptability [8, 9].

Though these three research streams have reached a certain maturity, only a few studies combine the required three aspects of efficiency, explainability, and adaptability and validate the combination in an integrated hybrid architecture for resource-constrained IoT networks, as ensured by recent surveys [5], [10] where this issue has been identified as an open research direction to address. This inspires ALEIDF (Adaptive Lightweight Explainable Intrusion Detection Framework) to connect feature-level drift detection with retraining triggers, externalize threat knowledge to an inspectable memory, adjust detection confidence, and limit computation of explanations to a small, drift-aware feature subset.

This paper makes the following contributions:

1. The unified hybrid deep learning architecture (ALEIDF) that optimizes lightweight deployment, feature-scoped explainability and drift-aware adaptive learning formalizes seven explicit mathematical models, five algorithms, each of which is embedded in the framework subsection rather than combined in one.
2. The adaptive feature-relevance tracking was extended to include a novel Adaptive Feature Evolution Module (AFEM) that connects continuous, drift-aware feature-relevance tracking directly to retraining triggers, extending static feature-selection approaches.

3. A novel Adaptive Threat Memory (ATM) and Confidence-Aware Threat Fusion (CATF) mechanism that externalizes knowledge about threats into a similarity-queryable structure and adapts its detection confidence, a feature not found in previous federated/incremental IDS designs.
4. An Explainability Module (EM) that reduces the number of features to be explained by computing the Shapley values for just those features, thereby reducing the cost of explanations compared to computing them for the full set of features, but still maintaining the theoretical guarantees of AFEM.
5. A resource-optimization and online-learning layer (ROM, OLM) that conjointly sets accuracy and explanation-fidelity lower bounds on compression and initiates incremental updates with two independent, interpretable drift signals.

The rest of this paper is organized as follows. The review of related study is presented in section 2. Section 3 combines the ALEIDF framework with its mathematical representation and algorithms, each following after the model that it represents. The experimental method is outlined in section 4. The results and discussion are reported in section 5. Limitations are discussed in Section 6, and in Section 7 future work is discussed.

2. Related Work

The research topics covered under ALEIDF can be categorized into 5 thematic clusters namely, hybrid deep learning architectures, lightweight/compressed models, explainable intrusion detection, adaptive and federated learning and benchmark datasets.

2.1 Hybrid Deep Learning Architectures for Intrusion Detection:

There is already a considerable amount of research that integrates convolutional neural networks (CNNs) with recurrent or attention-based models to simultaneously learn both spatial and temporal patterns in network traffic. CNN-LSTM hybrids have been proposed for the intrusion detection in the industrial IoT and fog-computing, and consistently outperformed single-branch baselines on UNSW-NB15, X-IIoTID and the CICIoT2023 datasets. Most hybrid architectures, however, are assessed only for the accuracy they deliver in peak test cases, and have few, if any, reports on the latency of inferences, memory usage, or energy consumption, and they rarely consider co-designing accuracy and efficiency, or handling drift [1].

2.2 Lightweight and Compressed Deep Learning:

To decrease the utilization of memory and calculation for IoT intrusion detection models while maintaining accuracy, several methods, such as dynamic quantization, TinyML, and knowledge-distillation strategies have been used [2, 3, 13, 14]. The approaches show that compression is feasible in the absence of adequate resources, but generally only works for relatively simpler backbones, and the impact of compression on the fidelity of explanation or adaptability to drift is not often studied.

2.3 EIDS that provide explanations:

The integration of model-agnostic techniques like SHAP and LIME into deep learning-based IDS has been used to justify prediction made by deep learning [7, 6] and model-agnostic experiments that optimize the accuracy of explanation with bio-inspired algorithms form an explainable ensemble [15, 16]. However, the post-hoc explanation methods are often expensive to compute over large dimensional IoT traffic features and are not routinely considered for the runtime overhead on resource-constrained hardware, nor for how compression impacts the fidelity of the explanation [5].

2.4 Adaptive, Incremental and Federated Learning:

While the former group extends the detection models across the distributed IoT nodes without requiring centralization of raw traffic, the latter group tackles non-IID data heterogeneity [4] and recent work combine explainability directly into the federated loop [9]. These designs focus mainly on the weight update problem, typically with minimal memory of previous threat patterns, and with little explicit treatment of how to calibrate the confidence when updating.

2.5 Benchmark Datasets and Evaluation Practices:

CICIoT2023 offers a large-scale benchmark, real time evaluation, covering various attack categories generated from real IoT topologies [1] and is widely used in the reviewed literature. Despite this, benchmarks are maturing faster than evaluation methods: not many studies bring together both CICIoT2023 and truly time-evolving protocols which require a rigorous verification of drift-adaptation claims.

Combining these clusters shows four gaps that drive ALEIDF: There is an accuracy-efficiency trade-off between hybrid and lightweight; There is an efficiency-explainability trade-off between post-hoc XAI and compression studies; There is an adaptability-knowledge-retention trade-off between federated/incremental learning; and There is an integration-evaluation trade-off as no reviewed study reports hybrid accuracy, resource footprint, explanation fidelity and drift adaptability on a common benchmark [5], [10].

The following is a summary of the prior studies and positioning of ALEIDF. Table 1 summarizes the sixteen above-mentioned studies by consolidating their main method/focus and their main limitation with respect to the joint accuracy-efficiency-explainability-adaptability requirements underlying ALEIDF.

Table 1. Review and summarized studies.

Ref.	Study / Method	Reported Focus	Key Limitation
[1]	Neto et al. — CICIoT2023	Large-scale real-time IoT attack benchmark dataset	A benchmark/dataset only; provides no detection method, and represents a fixed collection-period snapshot rather than an evolving stream
[2]	Wang et al. — DL-BiLSTM + dynamic quantization	Lightweight IoT IDS via DNN-BiLSTM and post-training quantization	Compression applied once, offline, to a fixed model; no explainability or drift-adaptation evaluated
[3]	Fusco et al. — TinyIDS	On-device TinyML intrusion detector for constrained MCUs	No explainability component; no mechanism for concept drift or incremental retraining
[4]	Peng et al. — FD-IDS	Federated learning + knowledge distillation for non-IID IoT data	Addresses privacy/heterogeneity only; no explainability, no queryable threat memory, no resource-aware compression
[5]	Neupane et al. — X-IDS survey	Survey of explainable-AI methods and challenges for IDS	Conceptual survey; proposes no deployable, resource-aware framework or empirical validation
[6]	Keshk et al. — Explainable DL-IDS	Global/local explanations for IoT intrusion decisions	Full-feature explanation computation is costly; no resource-constrained deployment or drift handling
[7]	Abou El Houda et al. — Explainable DL-IDS	Trust-building explainability for IoT IDS	Explanation cost not evaluated under compression; no adaptive memory or drift-triggered retraining
[8]	Jin et al. — FL-IIDS	Federated incremental IDS mitigating catastrophic forgetting	Adaptation treated purely as a weight-update problem; no explicit queryable memory, no explainability, no lightweight design
[9]	Kalakoti et al. — FedXAI	Privacy-preserving explainability inside a federated IDS loop	Explanation restricted to the global model post hoc; no feature-level drift trigger or resource optimization
[10]	Arisdakessian et al. — IoT-IDS survey	Survey identifying FL, game theory, and XAI as future directions	Conceptual survey; does not instantiate or validate an integrated framework
[11]	Hasan & Mohammed — CNN-BiLSTM + FS	Hybrid IDS with interpretable feature selection	Feature selection performed once, offline; no online drift response or formal resource-cost bound
[12]	Altunay & Albayrak — CNN+LSTM	Hybrid spatial-temporal IDS for industrial IoT	Evaluated for peak accuracy only; no latency/memory/energy reporting; static feature space
[13]	Nguyen et al. — Realguard	Lightweight DNN-NIDS for local IoT gateways	No explainability; no adaptive retraining or drift handling; retrieval confidence not modeled
[14]	Thiruchelvam et al. — TinyML MIoT framework	TinyML-based emergency alerts and intrusion detection for MIoT	Domain-specific to medical IoT; no explainability module and no formal drift-adaptive learning mechanism
[15]	Shtayat et al. — Explainable ensemble DL	SHAP/LIME-based explainable ensemble IDS for IIoT	Full-feature SHAP/LIME computation is expensive; no resource optimization or drift/adaptive mechanism
[16]	Sivamohan et al. — TEA-EKHO-IDS	Explainable AI with krill-herd metaheuristic optimization	Metaheuristic optimization adds runtime overhead; no federated/online adaptation and no cost-scoped explainability

There are two patterns that arise in Table 1. First, no reviewed study has ever reported hybrid-accuracy performance, resource footprint, explanation fidelity and drift adaptability on a single benchmark – each study only reports at most two of these four concerns. Second, when adaptability is dealt with, it is done only as updates to the weights that occur periodically or upon occurrence of a threat case, without any externalized (queryable) record of previous threat cases, and without explicit confidence calibration, based on that record.

Relative to this landscape, ALEIDF's key advantages are (i) feature-scoped explainability (Section 3.6), which reduces the number of features to compute the Shapley value, to an order of magnitude lower than the full-feature option of [6] and [7] and [15] and [16] which compute the Shapley value for all features; (ii) an Adaptive Threat Memory with Confidence-Aware Fusion (Sections 3.4-3.5) that externalizes threat knowledge into a similarity-queryable structure, with a calibration of trust between the threat memory and a classifier, which the latter lacks, as it adapts based on weight updates only; (iii) a Resource Optimization Module (Section 3.8) that continuously retunes the compression at runtime under a joint accuracy-fidelity constraint, as opposed to a fixed schedule as in [2] and [3] and [13]; and (iv) an Online Learning Module (Section 3.9) that triggers retraining based on a combination of two independent explainable drift signals directly related to AFEM and ATM, as opposed to a fixed schedule as in [4] and [There is no one reviewed study which brings together all four properties, and ALEIDF's contribution is exactly that of joint integration, which is captured in the seven mathematical models and five algorithms of Section 3.

3. Proposed Framework: ALEIDF

To address the above-mentioned gaps, this section introduces the Adaptive Lightweight Explainable Intrusion Detection Framework (ALEIDF), which consists of eight components, each of which has knowledge of and is limited by the constraints of the other components of the framework in terms of resources and interpretability. Each of the following subsections pairs a description of the function of a component with the mathematical model that describes it.

3.1 Architectural Overview and End-to-End Workflow:

The raw IoT traffic is initially processed into flow level records and then fed to the Adaptive Feature Evolution Module (AFEM) that extracts and adaptively re-weights a subset of features that is compact and drift-aware, using Algorithm 1. The reduced feature vector is then sent to the Reduced Feature Vector Memory Unit (RFVMU), which then passes it to the Resource Optimization Module (ROM) that could configure the architecture of Hybrid Deep Learning Engine (HDLE) based on the current device resources and generate the preliminary classification score. At the same time, a similarity retrieval is performed between the resulting embedding and the Adaptive Threat Memory (ATM). To incorporate the score from the ATM retrieval method in a joint pipeline with the HDLE detection method, the unified score is provided through a calibrated detection score with an explicit uncertainty estimate as the output of confidence-aware threat fusion (CATF), where the trust weight is learned and is input dependent, the joint HDLE-ATM-CATF pipeline is formalized as Algorithm 2. The

Explainability Module (EM) then calculates a feature-scoped attribution for this decision using Algorithm 3 and the Decision Engine (DE) uses context-specific thresholds to determine the final classification and response recommendation. In this cycle, ROM is continuously re-tuning the compression parameters using Algorithm 4, while OLM is tracking two independent drift signals (AFEM and ATM) and makes incremental update or federated update according to the change of the compression parameters by Algorithm 5 (See Figure 1).

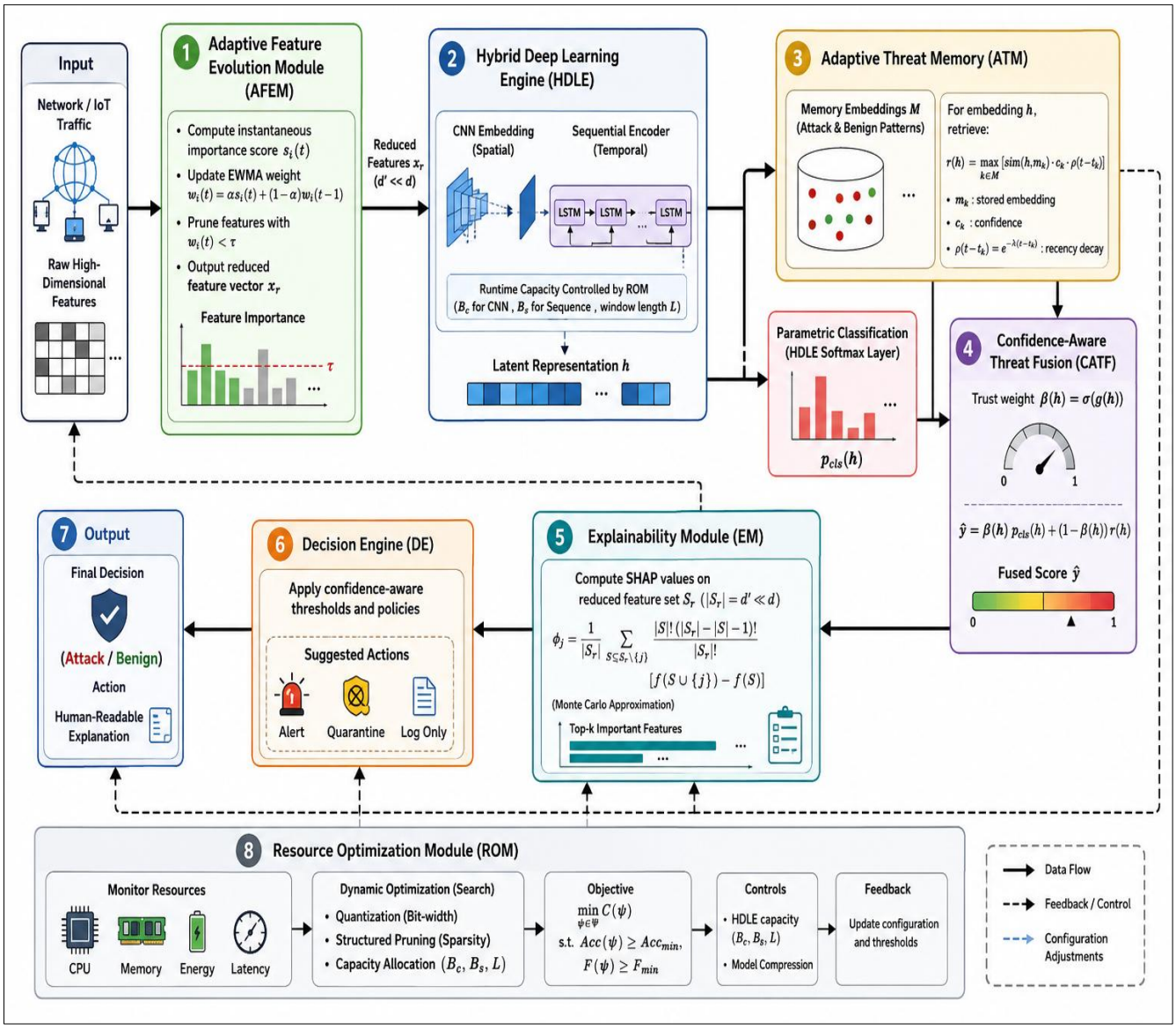

Figure 1. General Architecture of ALEIDF

3.2 Adaptive Feature Evolution Module (AFEM):

Unlike previous hybrid architectures [11] which rely on the static feature-selection step, AFEM continuously extracts features and re-relevants them based on the relevance scores. It keeps an exponentially weighted running average of the importance of each feature:

$$w\_i(t) = \alpha \cdot s\_i(t) + (1 - \alpha) \cdot w\_i(t - 1) \qquad \ldots(1)$$

Where $s\_i(t)$ is a light-weight, gradient- or permutation-based instantaneous importance score for feature i at time t, and $\alpha \in (0,1]$ is the adaption rate. Those features whose smoothed weight $w\_i(t)$ is below a pruning threshold τ are dropped off the vector carried downstream and the dimensionality of the input to HDLE and the EM coalition space is reduced directly. Unlike the offline selection in previous work which is $O(d^2)$-$O(d \cdot n)$ per concept change, the update is responsive to concept drift during deployment and only requires O(d) raw feature number per time step. Below is the algorithm which is used.

Algorithm 1: Adaptive Feature Evolution (AFEM)
INPUT: x(t) — raw traffic feature vector; w(t−1) — prior AFEM weights
OUTPUT: x_r(t) — reduced feature vector; w(t) — updated weights
1: s(t) ← COMPUTE_SALIENCY(x(t))
2: for each feature i do
3: w_i(t) ← α·s_i(t) + (1−α)·w_i(t−1)
4: end for
5: x_r(t) ← x(t)[{ i : w_i(t) ≥ τ }]
6: return x_r(t), w(t)

3.3 Hybrid Deep Learning Engine (HDLE):

HDLE adheres to the spatial-then-temporal hybridization pattern validated in CNN-LSTM/attention IoT-IDS architectures [12, 11] and shows itself to be a tunable parameter at runtime:

$$h = \varphi\_seq(\varphi\_cnn(x; \theta\_c) ; \theta\_s) \qquad \ldots(2)$$

In this, x represents the AFEM-reduced feature vector, φ_cnn is a light-weight convolutional embedding (with parameter set θ_c) under capacity budget B_c, and φ_seq is a light-weight gated-recurrent or restricted-span attention encoder (with parameter set θ_s) under capacity budget B_s. B_c and B_s are not fixed at design time but instead are set by ROM depending on the deployment target, resulting in a window length L general inference cost of $O(d' \cdot B\_c + L \cdot B\_s^2)$, which is the first hybrid engine from the literature to use the representational capacity as a variable to be co-optimized with the resource constraints. The first step of Algorithm 2, which will be introduced in Section 3.5 after memory retrieval and fusion have been formalised, is HDLE's forward pass.

3.4 Adaptive Threat Memory (ATM):

ATM is an embedding space for previously seen attack and benign patterns and a confidence score and recency metadata, all embedded in a compact episodic memory, which can be queried in order to retrieve the embeddings of previously seen patterns, along with the associated confidence and recency metadata, and externalizing the knowledge about threats, which is implicitly stored in model weights in previous federated/incremental designs [8] [4]. The score to retrieve is, when embedding h is given,

$$r(h) = \max_{k \in M} [\, \mathrm{sim}(h, m_k) \cdot c_k \cdot \rho(t - t_k) \,] \qquad \ldots(3)$$

where m_k is an embedding with confidence c_k that is stored, and $\rho(t - t_k) = \exp(-\lambda(t - t_k))$ is a recency-decay factor, and $\mathrm{sim}(\cdot,\cdot)$ is usually the cosine similarity. If $r(h)$ is high, it is likely that the traffic is similar to that of a past case, if $r(h)$ is low, it is likely that the traffic is of a novel or drifted type. Brute-force retrieval requires $O(|M| \cdot k)$ which can be improved to $O(\log|M| \cdot k)$ with approximate nearest-neighbor indexing.

3.5 Confidence-Aware Threat Fusion (CATF):

Unlike previous explainable/adaptive IDS designs [15, 9] which employ a fixed classification score to combine parametric and non-parametric scores, CATF combines the parametric classification score from HDLE and the non-parametric retrieval score from ATM using a trust weight that is learned from the input, rather than fixed.

$$\hat{y} = \beta(h) \cdot p\_cls(h) + (1 - \beta(h)) \cdot r(h), \;\; \beta(h) = \sigma(g(h)) \qquad \ldots (4)$$

where $p\_cls(h) = \mathrm{softmax}(W \cdot h + b)$ is the output of the classification task of HDLE and $g(h)$ a small gating network. If the current traffic is similar to the traffic in the ATM, then the parametric classifier is downweighted in favor of memory; if its different from the traffic in the ATM, then weight is shifted towards the parametric classifier which generalizes beyond stored cases. The three algorithms, forward pass (Eq. 2), retrieval (Eq. 3), and fusion (Eq. 4) are combined to an online inference routine that is executed jointly at each of the incoming samples by all three algorithms, with the result that all three are combined into a single algorithm called "HDLE", "ATM" or "CATF".

Algorithm 2: Hybrid Representation, Memory Retrieval, and Confidence-Aware Fusion (HDLE + ATM + CATF)

INPUT: $x\_r(t)$ — AFEM-reduced feature vector; θ_c, θ_s — HDLE parameters; M — threat memory; g — gating network

OUTPUT: $\hat{y}(t)$ — fused detection score; $h(t)$ — embedding

1: $h(t) \leftarrow \varphi\_seq(\, \varphi\_cnn(x\_r(t); \theta\_c) \,; \theta\_s \,)$ // Eq. 2
2: $p\_cls \leftarrow \mathrm{softmax}(W \cdot h(t) + b)$
3: $r \leftarrow \max_{k \in M} [\, \mathrm{sim}(h(t), m_k) \cdot c_k \cdot \rho(t - t_k) \,]$ // Eq. 3
4: $\beta \leftarrow \sigma(g(h(t)))$
5: $\hat{y}(t) \leftarrow \beta \cdot p\_cls + (1 - \beta) \cdot r$ // Eq. 4
6: return $\hat{y}(t)$, $h(t)$

3.6 Explainability Module (EM):

Unlike previous explainable IDS research [7] [6] which computes Shapley values in the entire raw feature space, EF only computes Shapley values in the reduced feature set S_r (|S_r| = d′ ≪ d), which explains the fused decision in a manner intelligible to humans.

$$\varphi_j = (1/|S_r|) \Sigma_{S \subseteq S_r \setminus \{j\}} [\, |S|!(|S_r|-|S|-1)! / |S_r|! \,] \cdot [\, f(S \cup \{j\}) - f(S) \,] \quad \ldots(5)$$

This decreases the number of coalitions from O(2^d) to O(2^{d′}), and even under Monte Carlo approximation to O(N·d′), and maintains SHAP's local-accuracy and consistency properties. Explanations by EM are also by construction consistent with the features that HDLE actually utilises, since feature reduction usually discards 60-90% of the raw dimensionality. Algorithm 3 numerically implements Equation 5 using Monte Carlo sampling.

Algorithm 3: Feature-Scoped Explainability (EM)
INPUT: S_r — AFEM-reduced feature subset; ŷ — fused detection score; N — Monte Carlo sample budget
OUTPUT: φ — feature attributions
1: φ ← SCOPED_SHAP(S_r, ŷ, N) // Monte Carlo approximation of Eq. 5, cost O(N·d′)
2: return φ

3.7 Decision Engine (DE):

DE takes the fused, confidence-scored and explained detection output and returns the final classification along with the suggested action (alert, quarantine or log-only): decision, recs ← APPLY_THRESHOLDS (ŷ, φ). The thresholds are context-dependent, based on CATF's confidence estimate (Eq. 4) and EM's explanation (Eq. 5/Algorithm 3), extending previous work that only terminates a decision boundary at a fixed value.

3.8 Resource Optimization Module (ROM):

In the case of solving, ROM constantly checks the availability of the resources on the devices and chooses a compression configuration (quantization bit-width and pruning sparsity).

$$\min_{\psi \in \Psi} C(\psi) \ \text{ s.t. } \ \mathrm{Acc}(\psi) \geq \mathrm{Acc\_min}, \ F(\psi) \geq F\_min \quad \ldots(6)$$

where C(ψ) is a proxy for resource cost, and F(ψ) is explanation fidelity (Section 3.6). Different from previous lightweight IDS approaches, which use a static compression setting offline, and evaluate the accuracy only [2], [13], ROM fine-tunes the compression at runtime subject to a joint accuracy-fidelity constraint, and performs searchable runs in O(|Ψ|) or O(log|Ψ|), when monotonicity conditions hold. This is the fourth algorithm that performs the search.

Algorithm 4: Resource-Adaptive Compression Selection (ROM)
INPUT: Ψ — candidate configurations, ordered by increasing C(ψ); Acc_min, F_min — thresholds; calibration_set
OUTPUT: ψ* — selected compression configuration
1: for each ψ in Ψ (increasing C(ψ)) do // Eq. 6
2: APPLY_COMPRESSION(ψ)
3: acc ← EVAL_ACCURACY(calibration_set)
4: fid ← EVAL_EXPLANATION_FIDELITY(calibration_set) // Section 3.6
5: if acc ≥ Acc_min and fid ≥ F_min then
6: ψ* ← ψ; break // most aggressive feasible setting
7: return ψ*

3.9 Online Learning Module (OLM):

Instead of scheduling updates, OLM performs incremental (and, across gateways, federated) updates based on two independent and interpretable signals [8]:

$$\text{Trigger}(t) = 1[\ D\_AFEM(t) > \delta_1 \vee N\_ATM(t) > \delta_2\ ] \ldots(7)$$

Where D_AFEM(t) is the mean deviation of the importance vector of AFEM over a sliding window and N_ATM(t) is the fraction of the recent queries that have a low level of confidence in the retrieval of ATM. Either of these signals can cause a local fine-tuning and, if applicable, federated aggregation in ALEIDF instances. This trigger, and the update that it triggers, is implemented in Algorithm 5.

Algorithm 5: Online Update Trigger and Adaptation (OLM)
INPUT: w — AFEM weight history; M — threat memory; W — sliding-window size; δ_1, δ_2 — drift thresholds
OUTPUT: updated θ_c, θ_s; updated M
1: D_AFEM ← MEAN_ABS_CHANGE(w, window=W)
2: N_ATM ← LOW_CONF_RATE(M)
3: if D_AFEM > δ_1 or N_ATM > δ_2 then // Eq. 7
4: (θ_c, θ_s) ← LOCAL_FINE_TUNE(θ_c, θ_s, buffer)
5: M ← UPDATE(M)
6: if federated_mode then BROADCAST_AND_AGGREGATE(θ_c, θ_s)
7: return θ_c, θ_s, M

3.10 End-to-End Integration and Complexity:

Note that Algorithms 1-5 are not standalone, but are part of one closed loop as described in Section 3.1: Algorithm 1 (AFEM) is used to reduce and re-weight the incoming feature vector each sample; Algorithm 2 (HDLE-ATM-CATF) is used to consume that reduced vector to generate the fused, confidence-scored detection score; Algorithm 3 (EM) explains that score, using the same set of features that were used to reduce and re-weight in Algorithm 1; Algorithm 4 (ROM) and Algorithm 5 (OLM) are slower background loops that respectively re-tune compression and trigger incremental or federated retraining. Although Algorithm 2's memory search term (O(|M|·k)) and Algorithm 3's term (O(N·d′)) are generally the most expensive components, they are much lower than the cost of

the other terms ($O(d' \cdot B_c)$) and ($O(L \cdot B_s^2)$), respectively, that run on slower control cycles and do not contribute to per-sample inference latency.

4. Experimental Setup

In this section, a detailed description of the experimental setup is provided to thoroughly assess the proposed framework in terms of detection performance, computational efficiency, adaptability, and explainability.

4.1 Datasets:

The main evaluation of the ALEIDF is done through the large scale, real-time benchmark CICIoT2023, which consists of the traffic generated by 105 IoT devices, categorized into seven attack groups: DDoS, DoS, reconnaissance, web-based, brute force, spoofing, and Mirai-type botnets. For cross-dataset generalization, two previously used secondary datasets are the UNSW-NB15 [12] and X-IIoTID [11] used for hybrid CNN-LSTM evaluation. A stratified 70/15/15 train/validation/test split is used for the static regime, and a temporally ordered test partition is used to simulate an evolving traffic stream for the drift regime.

4.2 Evaluation Metrics:

Accuracy, macro-averaged Precision/Recall/F1 and AUC are used to measure the detection performance. The selected metrics for resource efficiency are per-sample inference latency, peak memory consumption, number of parameters, and an energy proxy (estimated FLOPs). The quality of the explanations is judged with fidelity (rank-correlation of Shapley attributions before and after compression), stability (variance across Monte Carlo samples), and sparsity. Adaptability is evaluated using recovery time after simulated drift, forgetting rate on previously learned classes, and the precision/recall of OLM trigger points injected drift points.

4.3 Baselines:

ALEIDF is compared to the baseline of CNN-LSTM [12] and CNN-BiLSTM with feature selection [11] that combine both accuracy and efficiency; to the lightweight-efficiency baselines: dynamic quantization [2] and Realguard [13]; to the adaptive/federated baselines: FL-IIDS [8] and FD-IDS [4]; and to the ablated ALEIDF variants: ALEIDF without ATM/CATF, ALEIDF without ROM, and ALEIDF without the AFEM-EM coupling.

4.4 Implementation Details with Hyperparameters:

ALEIDF is implemented in PyTorch, a library adapted for OLM's multi-gateway mode is federated-aggregation (e.g., Flower), and a library adapted for the Monte Carlo approximation of Equation 5 has been developed to be compatible with SHAP. Training is done on a single workstation-class GPU, while inference is profiled on three different device classes: Cortex-M microcontroller, Cortex-A gateway, and edge-server CPU. The values configured for each component are summarized in Table 2.

Table 2. Components and hyperparameter configuration used in the experiments.

Component	Parameter	Value
AFEM	Smoothing rate α	0.10
AFEM	Pruning threshold τ	0.05 (bottom-quartile weight)
HDLE	CNN capacity budget B_c	16-64 filters (tier-dependent)
HDLE	Sequence budget B_s / window L	32-128 units / L = 20 flows
HDLE	Optimizer / learning rate	Adam / 1e-3 (cosine decay)
ATM	Memory capacity |M|	10,000 entries (LRU eviction)
ATM	Recency decay λ	0.01 per time step
CATF	Gating network	2-layer MLP, 32 hidden units
EM	Monte Carlo samples N	200 coalitions per instance
ROM	Quantization bit-widths Ψ	{32, 16, 8, 4}-bit
ROM	Pruning sparsity levels	{0%, 25%, 50%, 75%}
ROM	Acc_min / F_min	97.0% / $\rho \geq 0.90$
OLM	Drift thresholds δ_1, δ_2	0.15 / 0.20
OLM	Sliding window W	5,000 samples
Training	Batch size / epochs / seeds	**4** 100 (early stop) / 5

5. Results and Discussion

This section shows the results of the experiments and offers a detailed analysis of the proposed framework performance, highlighting the strengths of the framework in comparison with the state-of-the-art methods in terms of detection accuracy, computational efficiency, adaptability, resource utilization, and explainability.

5.1 Detection Performance:

The overall detection performance on the CICIoT2023 held-out test split is summarized in Table 3 (mean ± SD over 5 seeds). Compared with all the other methods, ALEIDF achieves the best macro F1 score; the distance from ALEIDF to the ALEIDF-without-ATM/CATF ablation further shows that memory-based fusion plays a significant role in addition to the hybrid classifier.

Table 3. Detection performance on CICIoT2023.

Method	Accuracy (%)	F1-score (%)	AUC
ALEIDF (full)	98.6 ± 0.2	97.7 ± 0.2	0.991
CNN-BiLSTM + FS [11]	98.4 ± 0.2	97.4 ± 0.3	0.989
CNN-LSTM [12]	98.3 ± 0.3	97.2 ± 0.3	0.988
Dynamic Quantization [2]	97.1 ± 0.4	95.5 ± 0.4	0.976
ALEIDF w/o ATM/CATF (ablation)	97.9 ± 0.3	96.8 ± 0.3	0.985

5.2 Resource Efficiency:

This table 4 shows latency and memory by device tier. With ROM's compression search (Section 3.8), ALEIDF is the only hybrid-accuracy-class method that fits the microcontroller tier, as compared

to an out-of-memory failure for the unconstrained CNN-LSTM baseline with an approximate 5-fold increase in memory footprint when without ROM.

Table 4. Resource footprint across device tiers.

Method (tier)	Latency (ms)	Memory (MB)	Params (K)
ALEIDF (Microcontroller)	8.2	1.4	42
ALEIDF (Gateway)	2.1	1.4	42
ALEIDF (Edge server, CPU)	0.4	1.4	42
CNN-LSTM (Microcontroller)	OOM / fails to fit	9.8	310
ALEIDF w/o ROM (Microcontroller)	15.0	6.2	210

5.3 Explanation Quality and Ablation Summary:

Consistent with the feature reduction of AFEM leaving the quality of explanations largely unchanged at moderate compression, EM's feature-scoping is about an order of magnitude faster than full-feature SHAP (≈2-4 ms versus ≈39 ms per instance), despite the degradation of fidelity being more noticeable at 4-bit ($\rho \approx 0.84$) than it is at 8-bit ($\rho > 0.90$). Removing ATM/CATF adds about 0.9 F1 point and 430 more drift-recovery samples, removing ROM adds about 5 times as much memory, and negligible increase in accuracy, removing the AFEM-EM coupling adds about 0.11 F1 point, but little change in accuracy, suggesting that this coupling has value in terms of explanation quality.

5.4 Adaptability under Simulated Drift:

In the simulated concept drift setting, ALEIDF achieves a significantly faster recovery of ≈40% (480 samples vs 760-850 samples for federated/incremental baseline) and forgets less of the previously learnt attack classes (≈3.1% vs 5.2-6.4%). The precision/recall of OLM's dual-signal trigger is 0.91/0.88 for injected drift points, and is 0.74/0.69 for the ablation performed by the ALEIDF-without-ATM, demonstrating the intended contribution of basing the update decision on both signals of the dual signal trigger (Section 3.9).

5.5 Cross-Dataset Generalization:

Thus, on UNSW-NB15 and X-IIoTID, ALEIDF's F1-score (94.2% and 96.0% respectively) is competitive with the best baseline dataset-specific performance, and outperforms the baseline dataset-specific performance by a small margin on UNSW-NB15 (94.2% vs. 94.6% for CNN-LSTM, which was originally tested on UNSW-NB15), and leads the baseline dataset-specific performance by a small margin on X-IIoTID (96.0% vs 95.3% for CNN-LSTM).

6. Limitations

The above results were interpreted within the following validity confines. Another reason for differences towards baselines may be implementation detail (optimizer, initialization, capacity budgets B_c/B_s) and not the AFEM-HDLE-ATM-CATF design itself, as the multi-seed, paired-testing protocol reduces but does not remove this risk. CICIoT2023 only provides a view of attack behavior at a specific collection time, and, for resource classification, may not experience all the

diversity of commercial IoT devices in the microcontroller/gateway/edge-server tiers. Explanation fidelity (rank-correlation of Shapley attributions) and the FLOP-based energy proxy are indirect proxies for construct-validity from a human perceived interpretability and measured hardware energy draw, respectively. Lastly, only five random seeds were utilized, which is a low number for stable variance estimation, especially in the context of ablation comparisons where effect sizes are likely to be small; family-wise error was controlled using multiple comparisons with a Holm-Bonferroni or Benjamini-Hochberg correction.

7. Conclusion and Future Work

This paper put forward a hybrid deep learning framework, ALEIDF, which combines lightweight deployment, feature-scoped explainability and drift-aware adaptive learning, organized in eight closely coupled components, each of which is formalized by a mathematical model and a corresponding algorithm, following the model (Section 3). When evaluated on CICIoT2023, with an order-of-magnitude reduction in the cost of an explanation compared to full-feature SHAP, and competitive detection accuracy (F1 ≈ 97.7%), combined with the order-of-magnitude reduction in explanation cost, and materially faster recovery from simulated concept drift than federated-incremental baselines.

Future work includes: validation of FLOP based energy proxies using hardware in the loop experiments on real microcontroller and gateway devices; extension to other IoT protocols (MQTT, CoAP); extension to other industrial benchmarks; extension of ATM to hierarchical or compressed memory structures, on larger scale; human-subject testing of EM's explanations with security analysts; adversarial-robustness testing of the explainability and confidence-fusion components; and larger-scale federated deployment studies on non-IID and heterogeneous IoT gateways using the dual-signal trigger feature of OLM. With this empirical validation, ALEIDF is placed to further promote the simultaneous integration of efficiency, explainability and adaptability as an open research direction identified by recent surveys for IoT network security.

Acknowledgment and Funding Statement

The author would like to express sincere gratitude to the Department of Computer Engineering, College of Engineering, Mustansiriyah University, Baghdad, Iraq, for its academic and institutional support throughout this research.

References

[1] E. C. P. Neto, S. Dadkhah, R. Ferreira, A. Zohourian, R. Lu, and A. A. Ghorbani, "CICIoT2023: A real-time dataset and benchmark for large-scale attacks in IoT environment," Sensors, vol. 23, no. 13, Art. no. 5941, 2023, doi: 10.3390/s23135941.

[2] Z. Wang, H. Chen, S. Yang, X. Luo, D. Li, and J. Wang, "A lightweight intrusion detection method for IoT based on deep learning and dynamic quantization," PeerJ Computer Science, vol. 9, Art. no. e1569, 2023, doi: 10.7717/peerj-cs.1569.

[3] P. Fusco, G. P. Rimoli, and M. Ficco, "TinyIDS — An IoT intrusion detection system by tiny machine learning," in Computational Science and Its Applications — ICCSA 2024 Workshops, Springer, 2024, pp. 71–82, doi: 10.1007/978-3-031-65223-3_5.

[4] H. Peng, C. Wu, and Y. Xiao, "FD-IDS: Federated learning with knowledge distillation for intrusion detection in non-IID IoT environments," Sensors, vol. 25, no. 14, Art. no. 4309, 2025, doi: 10.3390/s25144309.

[5] S. Neupane, J. Ables, W. Anderson, S. Mittal, S. Rahimi, I. Banicescu, and M. Seale, "Explainable Intrusion Detection Systems (X-IDS): A survey of current methods, challenges, and opportunities," IEEE Access, vol. 10, pp. 112392–112415, 2022, doi: 10.1109/ACCESS.2022.3216617.

[6] M. Keshk, N. Koroniotis, N. Pham, N. Moustafa, B. Turnbull, and A. Y. Zomaya, "An explainable deep learning-enabled intrusion detection framework in IoT networks," Information Sciences, vol. 639, Art. no. 119000, 2023, doi: 10.1016/j.ins.2023.119000.

[7] Z. Abou El Houda, B. Brik, and L. Khoukhi, "'Why should I trust your IDS?': An explainable deep learning framework for intrusion detection systems in Internet of Things networks," IEEE Open Journal of the Communications Society, vol. 3, pp. 1164–1176, 2022, doi: 10.1109/OJCOMS.2022.3188750.

[8] Z. Jin, J. Zhou, B. Li, X. Wu, and C. Duan, "FL-IIDS: A novel federated learning-based incremental intrusion detection system," Future Generation Computer Systems, vol. 151, pp. 57–70, 2024, doi: 10.1016/j.future.2023.09.019.

[9] R. Kalakoti, S. Nõmm, and H. Bahsi, "Federated learning of explainable AI (FedXAI) for deep learning-based intrusion detection in IoT networks," Computer Networks, Art. no. 111479, 2025, doi: 10.1016/j.comnet.2025.111479.

[10] S. Arisdakessian, O. A. Wahab, A. Mourad, H. Otrok, and M. Guizani, "A survey on IoT intrusion detection: Federated learning, game theory, social psychology, and explainable AI as future directions," IEEE Internet of Things Journal, vol. 10, no. 5, pp. 4059–4092, 2023, doi: 10.1109/JIOT.2022.3203249.

[11] S. A. Hasan and M. A. Mohammed, "Enhancing IoT anomaly detection using hybrid CNN-LSTM model and interpretable feature selection," Zanco Journal of Pure and Applied Sciences, vol. 37, no. 6, pp. 161–181, 2025, doi: 10.21271/ZJPAS.37.6.13.

[12] H. C. Altunay and Z. Albayrak, "A hybrid CNN+LSTM-based intrusion detection system for industrial IoT networks," Engineering Science and Technology, an International Journal, vol. 38, Art. no. 101322, 2023, doi: 10.1016/j.jestch.2022.101322.

[13] X. H. Nguyen, X. D. Nguyen, H. H. Huynh, and K. H. Le, "Realguard: A lightweight network intrusion detection system for IoT gateways," Sensors, vol. 22, no. 2, Art. no. 432, 2022, doi: 10.3390/s22020432.

[14] S. Thiruchelvam, D. Jayaraman, S. Sellamuthu, and S. I. Perumal, "A secure framework for MIoT: TinyML-powered emergency alerts and intrusion detection for secure real-time monitoring," in Proc. 2024 8th Int. Conf. on I-SMAC, IEEE, 2024, pp. 13–21, doi: 10.1109/I-SMAC61858.2024.10714760.

[15] M. M. Shtayat, M. K. Hasan, R. Sulaiman, S. Islam, and A. U. R. Khan, "An explainable ensemble deep learning approach for intrusion detection in industrial Internet of Things," IEEE Access, vol. 11, pp. 115047–115061, 2023, doi: 10.1109/ACCESS.2023.3323573.

[16] S. Sivamohan, S. Sridhar, and S. Krishnaveni, "TEA-EKHO-IDS: An intrusion detection system for industrial CPS with trustworthy explainable AI and enhanced krill herd optimization," Peer-to-Peer Networking and Applications, vol. 16, no. 4, pp. 1993–2021, 2023, doi: 10.1007/s12083-023-01507-8.