Computers, Materials & Continua

Tech Science Press

https://doi.org/10.32604/cmc.2026.082810

ARTICLE

Explainable Hierarchical Mamba for Edge-Based IoT Traffic Classification

Jiangyong Yu, Chuanping Hu* and Runnan Wang

School of Cyber Science and Engineering, Zhengzhou University, Zhengzhou, China

*Corresponding Author: Chuanping Hu. Email: cphu@zzu.edu.cn

Received: 23 March 2026; Accepted: 03 May 2026; Published: 15 June 2026

ABSTRACT: With the proliferation of Internet of Things (IoT) devices, accurate device fingerprinting of highly encrypted traffic has emerged as a critical challenge for ensuring network security. Existing deep learning models are either difficult to deploy in real-time due to excessive computational complexity (e.g., Transformers) or are limited in performance because their structure does not match the inherent hierarchy of traffic data (e.g., flattened state space models). Furthermore, a general lack of transparency in their decision-making processes restricts their trustworthiness in security-critical scenarios. To address these challenges, this paper proposes a Hierarchical Mamba with Gated Attribution Fingerprinting (HMX-GAF) framework. The framework explicitly models the intrinsic hierarchical structure of traffic data via a bespoke packet-flow dual-layer Mamba encoder, resolving the issue of architectural mismatch. Concurrently, it pioneers a zero-overhead Gated Attribution Fingerprinting (GAF) mechanism by leveraging the internal gating signals of the Mamba model, achieving high-fidelity intrinsic explainability. Comprehensive experiments on the CIC-IoT-2024 dataset demonstrate that HMX-GAF significantly outperforms current state-of-the-art models in both device identification and anomaly detection tasks, while maintaining the millisecond-level inference efficiency required for edge deployment.

KEYWORDS: Traffic classification; IoT security; device fingerprinting; state space model; Mamba; explainable AI; hierarchical modeling

1 Introduction

Smart speakers, security cameras, and network-enabled home appliances are rapidly transforming domestic spaces into highly networked micro-ecosystems. Market forecasts project that global smart-home revenue will leap from US $84.5 billion in 2024 to more than US $116 billion by 2029, reflecting both an extraordinary penetration rate and vast growth potential [1]. This trend not only drives an exponential increase in household network traffic but also lays a data foundation for subsequent intelligent services.

Beyond convenience, security and privacy threats proliferate in lockstep: malware-controlled cameras, continuously listening voice assistants, and vulnerability-prone smart hubs have all been exploited for large-scale attacks and data exfiltration. Zero Trust Architecture (ZTA) [2], therefore, has been widely adopted; its "never trust, always verify" maxim demands precise, real-time device fingerprints to support least-privilege access control and enable dynamic defence.

Within highly encrypted, heterogeneous home networks, obtaining accurate fingerprints faces two core challenges: (i) Transport Layer Security (TLS) 1.3 [3] and Encrypted Client Hello (ECH) [4] markedly diminish Deep Packet Inspection visibility; (ii) IoT traffic is highly dynamic and diverse, rendering rule-based methods largely ineffective. These macro-level obstacles push researchers toward deep-learning

solutions that rely solely on traffic metadata to compensate for diminished visibility and accommodate complex behaviour.

Among deep-learning frameworks for encrypted traffic, Transformers [5,6] achieve excellent performance but are hindered in real-time deployment by quadratic time complexity; State Space Models (SSMs) [7] such as Mamba [8] reduce complexity to linear time and, in NetMamba [9], set multiple new benchmarks, yet reveal fresh technical bottlenecks: first, flattening hierarchical traffic produces an architectural mismatch; second, the model remains a "black box," lacking intrinsic explainability [10]. Consequently, a lightweight model that both captures hierarchical structure and delivers real-time explanations is urgently needed.

In response, this paper proposes HMX-GAF, a hierarchical Mamba framework that models packet-flow dependencies and leverages selective gating for zero-cost explainability. Experiments on CIC-IoT-2024 confirm significant improvements in both accuracy and interpretability. The main contributions are: (1) an HMX hierarchical architecture capturing intra- and inter-packet dependencies; (2) a zero-overhead GAF explainability mechanism. The remainder of this paper is organised as follows: Section 2 surveys related work; Section 3 details the method; Section 4 reports experiments; Section 5 concludes. Fig. 1 contrasts flattening with our hierarchical approach.

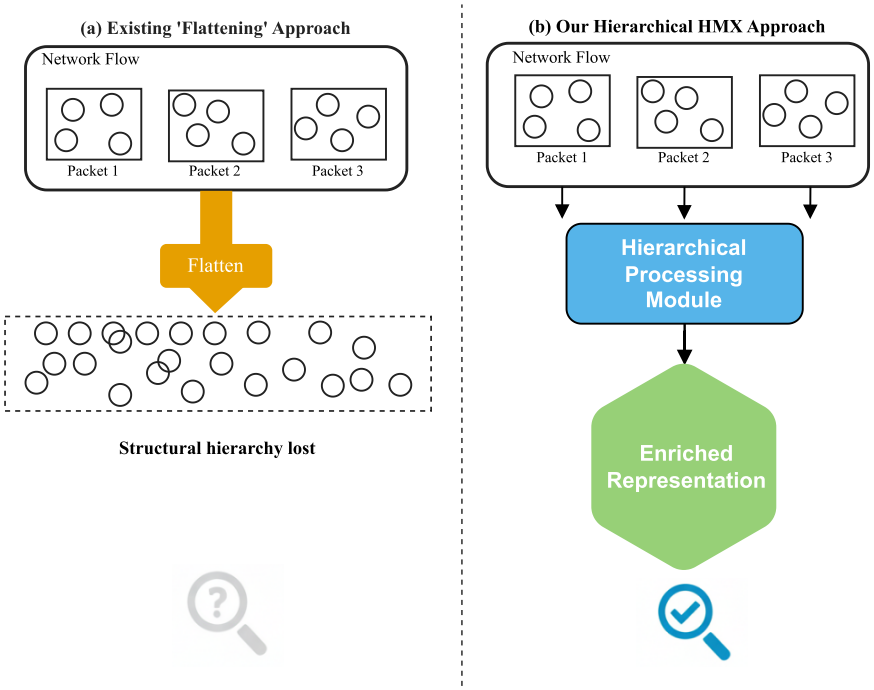

Figure 1: Conceptual comparison of traffic representation: (**a**) conventional flattening approach vs. (**b**) proposed hierarchical HMX approach.

2 Related Work

2.1 The Evolution of IoT Device Fingerprinting

IoT device fingerprinting has evolved through three stages [11,12]. Early work leveraged physical-layer characteristics such as clock skew [13] and Radio Frequency (RF) imbalances [14], but these methods are susceptible to environmental conditions and require specialized hardware. The second stage focused on plaintext protocol tokens (e.g., Dynamic Host Configuration Protocol (DHCP), HyperText Transfer Protocol

(HTTP) User-Agent), exemplified by IoT Sentinel [15]. However, with TLS now encrypting over 90% of home-network traffic, plaintext-based methods have become largely ineffective [16], driving a shift toward encryption-agnostic behavioral analysis.

2.2 Deep Learning Methods for Encrypted Traffic

Current approaches analyze sequences of traffic metadata (packet size, direction, inter-arrival times) using deep learning [17]. Convolutional Neural Networks (CNNs) [18] and Long Short-Term Memory networks (LSTMs) [19] capture local temporal patterns [20,21] but struggle with long-range dependencies. Transformers improved accuracy via global self-attention, yet their $O(n^2)$ complexity precludes millisecond-level gateway deployment. Pre-training paradigms such as ET-BERT [22] and YaTC [23] further advanced the state of the art through contextualized representation learning.

To overcome the complexity bottleneck, Structured State Space models (S4 [7], H3 [24]) reduce complexity to $O(n)$. Mamba [8] adds input-dependent selective gating for context-aware state evolution, matching Transformer quality at linear cost. Mamba-2 [25] further establishes a theoretical SSM–attention duality with 2–8× speedups. In the traffic domain, NetMamba [9] first applied Mamba to network traffic classification, raising Macro-F1 from 96.1% to 97.8% while cutting Graphics Processing Unit (GPU) memory by 40%, and ET-Mamba [26] extends the paradigm with ultra-low parameter counts. SSMs have also shown promise for traffic generation [27]. Despite these successes, existing methods flatten traffic into one-dimensional sequences, disregarding the inherent "packet-flow" hierarchy and lacking intrinsic interpretability [9]. In parallel, lightweight classification architectures have been explored for resource-constrained IoT environments, including Multi-Layer Perceptron (MLP)-based feature extractors with linear classifiers [28] and hybrid ensemble frameworks for industrial IoT intrusion detection [29]. However, these approaches do not address the hierarchical structure of traffic data or provide intrinsic explainability.

2.3 Interpretability in Network Security

Interpretability is essential for deploying deep learning in safety-critical security scenarios [30]: Security Operations Center (SOC) analysts need to understand alert root causes [31], and regulations such as the General Data Protection Regulation (GDPR) mandate auditability. Post-hoc methods like LIME [32] and SHAP [33] rely on perturbation-based sampling whose fidelity has been questioned [10,34], while attention-based visualizations correlate poorly with model gradients [35]. Our proposed GAF overcomes these limitations by leveraging the Mamba architecture's internal gating signals to generate fine-grained, input-aligned attributions at zero additional cost [8].

2.4 Summary and Research Motivation

In summary, physical-layer and protocol-token methods are ineffective under encryption [3]; Transformers face quadratic complexity bottlenecks [5]; and Mamba-based models lack hierarchical modeling and interpretability [9]. Meanwhile, edge deployment demands lightweight solutions [36]. This paper addresses these gaps with a hierarchical Mamba framework and gated attribution mechanism, targeting two objectives at linear complexity: (1) modeling the "packet-flow" hierarchy, and (2) providing real-time explanations.

3 Proposed Method

Formally, given a network flow $F = (P_1, \ldots, P_L)$ of L packets, the objectives are to learn a classifier $f : F \to \{1, \ldots, C\}$ that maps flows to one of C device classes, and simultaneously produce a per-token attribution map $\mathcal{A}$ that explains each decision, all within a single forward pass. This section details

the HMX-GAF framework (Fig. 2), which addresses these objectives via a hierarchical Mamba encoder complemented by an intrinsic attribution mechanism [7,8].

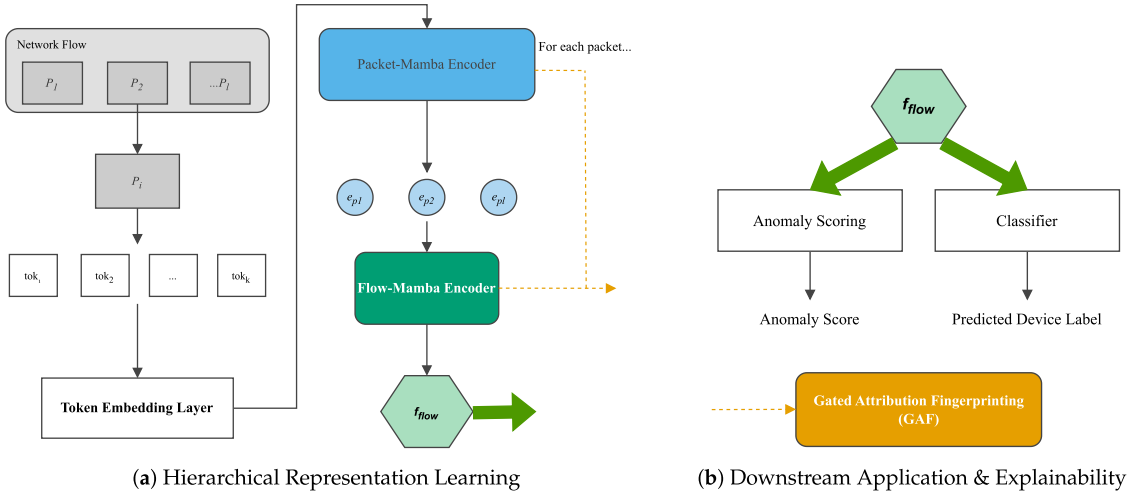

Figure 2: The architecture of the HMX-GAF framework. (**a**) Hierarchical representation learning; (**b**) downstream application & explainability.

3.1 Hierarchical Traffic Representation

Conventional models flatten all packet features into a single sequence, discarding the inherent two-level structure where packets are nested within flows [9,37], inflating the sequence length to $L \times K$ and conflating intra-packet and inter-packet patterns. We instead propose a multi-level representation that explicitly preserves the packet-flow hierarchy.

A network flow is defined as a sequence of L packets:

$$\text{Flow} = (P_1, P_2, \ldots, P_L) \tag{1}$$

Each packet P_i is represented as a sequence of K intra-packet feature tokens. Specifically, we extract a fixed set of metadata fields from each packet header—including source port, destination port, protocol type, Transmission Control Protocol (TCP) flags, payload length, and inter-arrival time—and discretize continuous values into categorical bins. Notably, the inclusion of *protocol type* as an explicit input token allows the shared Packet-Mamba encoder to naturally differentiate TCP from User Datagram Protocol (UDP) semantics: because Mamba's gating parameters ($\Delta_t, \mathbf{B}_t, \mathbf{C}_t$) are all input-dependent (Section 3.2), the encoder implicitly conditions its behavior on the protocol context without requiring separate per-protocol branches. The tokenized representation is:

$$P_i \to S_{\text{pkt}}^{(i)} = (\text{tok}_1, \text{tok}_2, \ldots, \text{tok}_K) \tag{2}$$

Each token tok_j is mapped to a dense vector via a learnable embedding layer E_f:

$$S_{\text{emb}}^{(i)} = \big(E_f(\text{tok}_1), E_f(\text{tok}_2), \ldots, E_f(\text{tok}_K)\big) \tag{3}$$

To preserve the ordering semantics among tokens within a packet, a learnable positional encoding $\mathbf{PE} \in \mathbb{R}^{K\times d}$ is added element-wise to the embedded sequence:

$$\tilde{S}^{(i)}_{\text{emb}} = S^{(i)}_{\text{emb}} + \mathbf{PE} \tag{4}$$

The position-augmented sequence $\tilde{S}^{(i)}_{\text{emb}}$ serves as the input to the hierarchical encoder.

3.2 Hierarchical Mamba (HMX) Encoder

The HMX encoder employs a dual-layer stacked Mamba architecture to explicitly model the hierarchical nature of network traffic. Before describing each layer, we first review the core Selective State Space mechanism that underpins both.

Selective State Space Mechanism. Each Mamba layer is built upon a state space model that maps input x_t to output y_t through a latent state $h_t \in \mathbb{R}^N$. The continuous system $h'(t) = \mathbf{A}\,h(t) + \mathbf{B}\,x(t)$, $y(t) = \mathbf{C}\,h(t)$ is discretized via zero-order hold with step size Δ:

$$\bar{\mathbf{A}} = \exp(\Delta\,\mathbf{A}), \quad \bar{\mathbf{B}} = (\Delta\,\mathbf{A})^{-1}(\bar{\mathbf{A}} - \mathbf{I}) \cdot \Delta\,\mathbf{B} \tag{5}$$

yielding the recurrence $h_t = \bar{\mathbf{A}}\,h_{t-1} + \bar{\mathbf{B}}\,x_t$, $y_t = \mathbf{C}\,h_t$. Crucially, Mamba makes $\mathbf{B}$, $\mathbf{C}$, and Δ input-dependent via learned projections from x_t [8]: the gate Δ_t modulates how strongly each token updates the hidden state, enabling the model to emphasize informative tokens while suppressing noise. This selectivity is the theoretical cornerstone upon which our GAF mechanism (Section 3.3) is built.

Packet-Mamba (Packet-Level Encoder). The first layer processes $\tilde{S}^{(i)}_{\text{emb}}$ to capture intra-packet token dependencies. The selective scan is applied over the K tokens, and the final hidden state serves as the context-aware packet embedding:

$$ep_i = \text{PacketMamba}\left(\tilde{S}^{(i)}_{\text{emb}}\right) = h^{(p)}_K \tag{6}$$

The resultant packet embedding $ep_i \in \mathbb{R}^d$ encodes rich intra-packet contextual information, such as the semantic association between TCP flags and port numbers. All L packets in a flow are processed by a shared Packet-Mamba with tied parameters, ensuring parameter efficiency. We adopt last-state extraction rather than alternative pooling (mean-pooling, attention-pooling) because Mamba's selective scan progressively accumulates all token information into each hidden state via input-dependent gating. Unlike vanilla Recurrent Neural Networks (RNNs), Mamba's Δ_t-controlled update ensures $h^{(p)}_K$ retains full intra-packet context, making additional pooling redundant. This design choice is empirically validated in Section 4.4.

Flow-Mamba (Flow-Level Encoder). The second layer, termed Flow-Mamba, takes the sequence of packet embeddings S_f as its input to model inter-packet temporal dependencies:

$$S_f = \left(ep_1, ep_2, \ldots, ep_L\right) \tag{7}$$

Flow-Mamba computes the final hidden state $h^{(f)}_L$, which serves as the comprehensive embedding for the entire network flow:

$$f_{\text{flow}} = \text{FlowMamba}\left(S_f\right) = h^{(f)}_L \tag{8}$$

Although both layers share the same mathematical formulation, they operate at different semantic granularities: the former captures field-level correlations within a packet header, while the latter captures

behavioral patterns across the temporal evolution of a flow. This hierarchical design mirrors the physical generation process of network traffic, enabling more robust and semantically meaningful representations.

3.3 Gated Attribution Fingerprinting (GAF)

To achieve intrinsic explainability without post-hoc computation, GAF repurposes the selective gating signals already computed during the Mamba forward pass.

Theoretical Foundation. As shown in Eq. (5), the step size Δ_t controls how strongly input x_t updates the hidden state. Under a first-order approximation ($\bar{\mathbf{B}}_t \approx \Delta_t \cdot \mathbf{B}_t$), the state innovation scales linearly with Δ_t, making it a natural, gradient-aligned [35] measure of each token's contribution. For token z_t, its attribution score is:

$$\mathcal{A}(z_t) = \Delta_t \tag{9}$$

This hypothesis—that Δ serves as a high-fidelity proxy for feature importance—is empirically validated in Section 4.5 through comparison with gradient-based ground truth.

Hierarchical Attribution Pipeline. Leveraging the dual-layer HMX design, GAF enables a structured top-down attribution process across two semantic levels:

- **Flow-level attribution:** The $\Delta_i^{(f)}$ values computed by Flow-Mamba quantify the relative importance of each packet P_i to the final flow embedding f_{flow}. These scores are normalized across the flow to produce a packet-importance distribution:

$$\hat{\mathcal{A}}_{\text{flow}}(P_i) = \frac{\Delta_i^{(f)}}{\sum_{j=1}^{L} \Delta_j^{(f)}} \tag{10}$$

- **Packet-level attribution:** For each identified key packet, $\Delta_j^{(p)}$ values from Packet-Mamba localize the influential intra-packet tokens (e.g., which specific header field drove the decision):

$$\hat{\mathcal{A}}_{\text{pkt}}(\text{tok}_j \mid P_i) = \frac{\Delta_j^{(p,i)}}{\sum_{k=1}^{K} \Delta_k^{(p,i)}} \tag{11}$$

Composite Attribution Score. The two levels are composed multiplicatively to trace a decision down to a specific token:

$$\mathcal{A}_{\text{global}}(\text{tok}_j, P_i) = \hat{\mathcal{A}}_{\text{flow}}(P_i) \times \hat{\mathcal{A}}_{\text{pkt}}(\text{tok}_j \mid P_i) \tag{12}$$

This composite score traces decisions from flow level to individual packet features at *zero* additional inference cost, since all Δ values are already computed during the forward pass.

3.4 Joint Training and Anomaly Scoring

The model is trained with a composite loss to jointly optimize classification accuracy and embedding space geometry (Fig. 3):

$$\mathcal{L} = \mathcal{L}_{\text{cls}} + \lambda \mathcal{L}_{\text{triplet}} \tag{13}$$

where $\mathcal{L}_{\text{cls}}$ denotes the standard cross-entropy loss over the device label space.

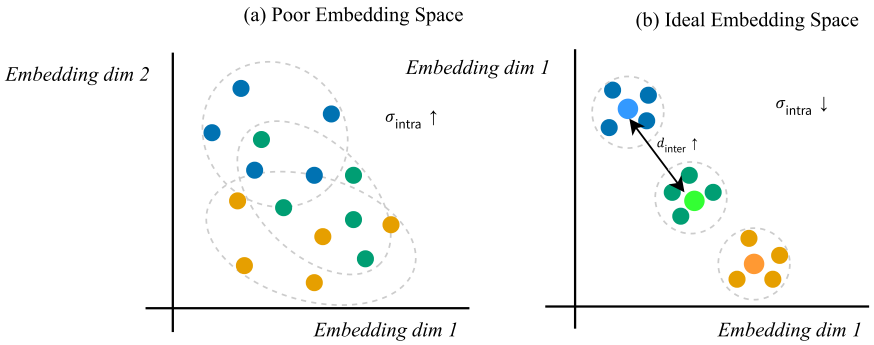

Figure 3: Impact of triplet loss on embedding space quality. (**a**) A poor embedding space with overlapping clusters vs. (**b**) an ideal embedding space with compact and well-separated clusters.

Triplet Loss and Hard Mining. The triplet loss encourages compact, well-separated clusters for each device class:

$$\mathcal{L}_{\text{triplet}} = \max\big(0,\ d(f_a, f_p) - d(f_a, f_n) + m\big) \tag{14}$$

where $d(\cdot,\cdot)$ is the Euclidean distance and $m = 0.2$ is the margin. We employ online semi-hard negative mining to select negatives satisfying $d(f_a, f_p) < d(f_a, f_n) < d(f_a, f_p) + m$, balancing gradient strength with training stability. Concretely, within each mini-batch of 256 flows (sampled to contain at least 4 flows per device class), we form all valid anchor-positive pairs from same-class flows and search the batch for semi-hard negatives from different classes. When no semi-hard negative exists for a given pair, we fall back to the hardest negative (smallest $d(f_a, f_n)$); pairs lacking any valid negative are skipped. This yields approximately 1200 triplets per batch on average. The coefficient $\lambda = 0.1$ is determined via grid search.

Anomaly Scoring. After training, a class centroid $\mu_c = \frac{1}{|\mathcal{D}_c|} \sum_{f \in \mathcal{D}_c} f_{\text{flow}}$ is computed for each known device class. At inference, a new flow f_{new} is scored by its distance to the nearest centroid:

$$\text{AnomalyScore} = \left\| f_{\text{new}} - \mu_{c^*} \right\|^2 \tag{15}$$

Flows exceeding a threshold τ (calibrated on the validation set) are flagged as anomalous; GAF (Section 3.3) then identifies the responsible features. We choose Euclidean distance for its simplicity and edge efficiency; a comparison with Mahalanobis and kNN scoring is provided in Section 4.4. Note that $f_{\text{flow}} = h_L^{(f)}$ is the final hidden state of the Flow-Mamba recurrence, which is inherently order-sensitive: two flows with identical per-packet statistics but different temporal orderings produce distinct embeddings, ensuring that temporal dynamics are captured in the anomaly score.

4 Experimental Results and Analysis

Our evaluation addresses three research questions: **RQ1** (Effectiveness)—does HMX–GAF significantly outperform flat-sequence baselines? **RQ2** (Architectural Contribution)—what do the hierarchy and triplet loss each contribute? **RQ3** (Explainability)—does GAF provide high-fidelity, analyst-meaningful insights?

4.1 Experimental Setup

4.1.1 Datasets and Splitting Strategy

Our study is primarily based on the CIC-IoT-2024 dataset [19], a large-scale and realistic collection of traffic from 97 distinct smart home devices. This dataset provides a challenging benchmark due to its

diversity and scale. For the anomaly detection task, we augment the test set with malicious traffic samples from the Mirai and BashLite portions of the IoT-23 dataset [38].

To ensure the model learns generalizable device fingerprints, we strictly adhere to a device-disjoint splitting strategy: the 97 device identities are randomly shuffled and partitioned into training/validation/test sets with a 70%/15%/15% ratio, guaranteeing mutually exclusive device sets across partitions. The final distribution is summarized in Table 1.

Table 1: Dataset split statistics based on the device-disjoint principle.

Set	Purpose	# of Devices	Device ID Examples	# of Flows	Percentage
Training	Model Training	68	D01–D68	284,521	~70%
Validation	Hyperparameter Tuning	15	D69–D83	61,156	~15%
Test (Benign)	Final Performance Eval.	14	D84–D97	59,782	~15%
Test (Malicious)	Anomaly Detection Eval.	External dataset	IoT-23 Mirai/BashLite	5210	External AD test only

4.1.2 Data Preprocessing and Feature Extraction

The raw network traffic data (in `.pcap` format) was processed through a multi-stage pipeline to generate feature sequences suitable for the HMX–GAF model.

1. *Flow Generation:* Raw packets were first assembled into bidirectional traffic flows using a 5-tuple identifier (source Internet Protocol (IP), destination IP, source port, destination port, protocol) with a 60-s inactivity timeout.
2. *Flow Truncation/Padding:* To create fixed-length sequences for batch processing, each flow was truncated to the first $L = 50$ packets. Flows with fewer than 50 packets were padded with zero vectors to ensure uniform length.
3. *Feature Extraction:* For each packet within a flow, we extracted a sequence of $K = 10$ intra-packet features (tokens), including packet length, TCP window size, IP flags, and protocol type. These categorical and numerical features were then mapped to a unified integer vocabulary.
4. *Normalization:* All numerical features, such as packet length, were normalized to a $[0, 1]$ range using min–max scaling based on statistics computed *only* from the training set to prevent data leakage.

This pipeline results in each network flow being represented as a tensor of shape $(L \times K)$, which serves as the input to the model's embedding layer. The same fixed input shape ($L = 50$, $K = 10$) is used during inference timing to avoid kernel selection variance.

4.1.3 Evaluation Metrics and Baselines

Evaluation Metrics: We employ Accuracy and Macro-F1 for device classification, Area Under the Receiver Operating Characteristic Curve (AUC-ROC) for anomaly detection, parameter count and inference latency on NVIDIA Jetson Xavier NX for efficiency, and intra-class variance σ_{in}/inter-class distance d_{inter} for embedding space geometry.

Baseline Models: We compare against three categories under the *same pipeline*: (1) **Random Forest (RF)** using statistical features; (2) **Transformer-Tiny**, **Transformer (Vanilla)**, **Transformer-XL** [6], and **TabTransformer** [39]; (3) **Mamba (Generic)** [8] and **NetMamba-U** [9].

Implementation Details. All models use the Adam optimizer. For HMX–GAF, $\lambda = 0.1$ and $m = 0.2$; early stopping is applied on validation Macro-F1. We report mean ± std over five runs. Efficiency metrics are measured on Jetson Xavier NX (FP32, batch = 1, 200 runs after 50 warm-ups).

4.2 Main Performance Evaluation (RQ1)

Table 2 summarizes the performance comparison on the CIC-IoT-2024 test set (Fig. 4). HMX–GAF achieved a Macro-F1 of 0.992 (+0.3% over NetMamba-U) with low standard deviation (±0.002), and the highest AUC-ROC of 0.987. Under the throughput setting (*FP32*, batch = 32) on Jetson Xavier NX, it reaches **20.1 ms** per sample, maintaining competitive efficiency. The edge setting (batch = 1) is analyzed in Section 4.3 (Table 3).

Table 2: Main performance comparison on the CIC-IoT-2024 dataset.

Model	Parameters (M)	Latency (ms)	Accuracy (%)	Macro-F1	AUC-ROC
RF (Baseline)	0.02	11.5	82.3 ± 0.6	0.785 ± 0.008	0.815 ± 0.007
TabTransformer	2.6	26.3	97.2 ± 0.5	0.972 ± 0.005	0.958 ± 0.006
Transformer-Tiny	2.1	25.8	98.50 ± 0.4	0.985 ± 0.004	0.965 ± 0.005
Transformer (Vanilla)	3.0	27.8	98.3 ± 0.3	0.983 ± 0.003	0.962 ± 0.004
Transformer-XL	3.4	28.6	98.6 ± 0.3	0.986 ± 0.003	0.969 ± 0.004
Mamba (Generic)	1.7	18.4	98.9 ± 0.3	0.988 ± 0.003	0.973 ± 0.004
NetMamba-U	1.8	18.9	98.95 ± 0.3	0.989 ± 0.003	0.974 ± 0.004
HMX-GAF (Ours)	2.0	20.1	**99.18** ± 0.2	**0.992** (+0.3%)*	**0.987** (+1.3%)*

Note: Results are the mean ± standard deviation of 5 runs. The best results are shown in bold. *Values in parentheses denote the relative improvement over the NetMamba-U baseline.

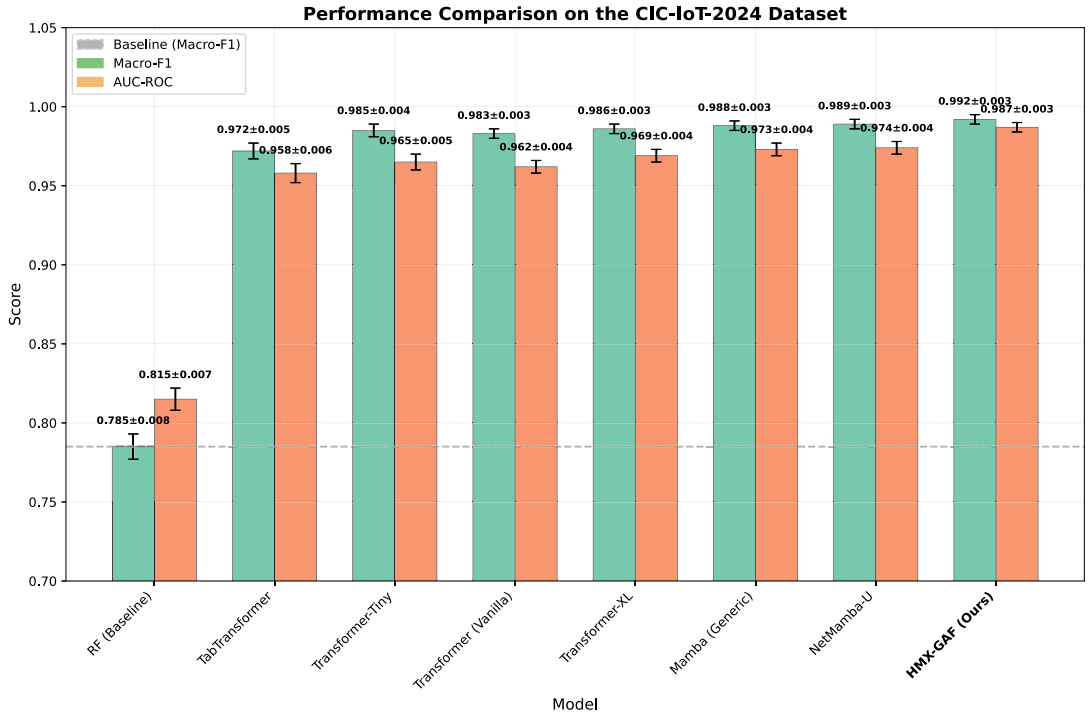

Figure 4: Performance comparison of HMX-GAF and baseline models on the CIC-IoT-2024 dataset.

Table 3: Edge inference efficiency on NVIDIA Jetson Xavier NX (B = 1, FP32).

Model	Latency (ms, mean ± std)	P95 (ms)	Peak Mem (MB)	Energy (mJ)
RF (Baseline, CPU)	11.5 ± 0.69	12.6	CPU only, not profiled	58
TabTransformer	36.8 ± 2.58	41.0	394	442
Transformer-Tiny	36.1 ± 2.53	40.2	359	433
Transformer (Vanilla)	38.9 ± 2.72	43.4	423	467
Transformer-XL	40.0 ± 2.80	44.6	497	480
Mamba (Generic)	25.8 ± 1.81	28.8	280	310
NetMamba-U	26.5 ± 1.86	29.6	286	318
HMX–GAF (Ours)	**28.1 ± 1.97**	**31.3**	**352**	**337**

Note: Measured on Jetson Xavier NX (FP32, batch = 1, fixed input shapes). Latency: mean ± std over 200 runs after 50 warm-ups; P95: 95th-percentile.

4.3 Inference Efficiency on Edge (B = 1)

Table 3 reports single-sample inference on NVIDIA Jetson Xavier NX (FP32, B = 1, 200 runs after 50 warm-ups). SSM models are consistently more efficient than Transformers. HMX–GAF lies on the Pareto frontier: while slightly slower than NetMamba-U (28.1 vs. 26.5 ms), it achieves higher Macro-F1 (0.992 vs. 0.989), offering a superior accuracy–efficiency trade-off. Tail latency (P95) is well controlled at +2–4 ms above the mean.

4.4 Ablation Study (RQ2)

We conduct ablations under the same pipeline (device-disjoint split, identical preprocessing/training schedule); Table 4 reports mean ± std over five seeds. Three findings stand out: (i) *Hierarchy is the dominant contributor*—removing it lowers Macro-F1 by 1.4 pp and AUC by 1.2 pp; (ii) *Triplet loss acts as a geometry regularizer*—it brings a modest Macro-F1 gain but a larger AUC gain; (iii) *GAF provides small yet consistent improvements* without extra training overhead. Replacing the Flow-Mamba backbone with Transformer or Gated Recurrent Unit (GRU) further degrades both metrics, confirming the advantage of SSM-style backbones.

We further evaluate two additional design choices. First, we compare three packet aggregation strategies: Mean-Pool $\left(ep_i = \frac{1}{K}\sum_j h_j^{(p)}\right)$, Attention-Pool (learned weighted sum), and our default Last-State $\left(h_K^{(p)}\right)$. As shown in the middle portion of Table 4, Last-State achieves the best Macro-F1 (0.992) with no additional parameters, confirming that the Mamba recurrence effectively accumulates full token context into its final state. Second, we compare anomaly scoring functions: Euclidean distance to the nearest centroid (default), Mahalanobis distance, and k-Nearest Neighbor (k = 5) in embedding space. Mahalanobis yields a marginal AUC-ROC improvement (+0.001) at 1.12× overhead, while kNN slightly underperforms (−0.001) at 3.5–6× overhead. The Euclidean baseline thus offers the best trade-off for edge deployment.

4.5 GAF Explainability Analysis (RQ3)

We validate GAF through three complementary analyses.

(1) Qualitative Case Study. As illustrated in Fig. 5, GAF correctly attributed the classification of a "Google Home Mini" to Domain Name System (DNS) queries for `*.google.com` and subsequent QUIC bursts—a known behavioral signature. For a Mirai-infected camera, GAF identified repeated TCP

connection attempts to port 23 (Telnet) as the root cause of the high anomaly score, aligning with expert domain knowledge.

Table 4: Ablation results on the CIC-IoT-2024 dataset (RQ2).

Variant	Accuracy (%)	Macro-F1	Δ vs. Full (F1)	AUC-ROC	Δ vs. Full (AUC)
HMX-GAF (Full)	**99.18** ± 0.20	**0.992** ± 0.002	—	**0.987** ± 0.003	—
w/o Hierarchy (Removal of Hierarchy)	98.70 ± 0.30	0.978 ± 0.003	−0.014	0.975 ± 0.004	−0.012
w/o Triplet Loss (Removal of Triplet)	98.90 ± 0.30	0.987 ± 0.003	−0.005	0.981 ± 0.003	−0.006
w/o GAF (Disable Gating/Attribution)	99.00 ± 0.20	0.989 ± 0.003	−0.003	0.983 ± 0.003	−0.004
Backbone → Transformer	98.70 ± 0.30	0.986 ± 0.003	−0.006	0.979 ± 0.004	−0.008
Backbone → GRU	98.50 ± 0.30	0.981 ± 0.003	−0.011	0.976 ± 0.004	−0.011
Packet Aggregation Strategy					
Aggregation → Mean-Pool	98.95 ± 0.24	0.989 ± 0.003	−0.003	0.983 ± 0.003	−0.004
Aggregation → Attn-Pool	99.04 ± 0.22	0.991 ± 0.002	−0.001	0.985 ± 0.003	−0.002
Anomaly Scoring Function (classification metrics unchanged)					
Scoring → Mahalanobis	—	—	—	0.988 ± 0.003	+0.001
Scoring → kNN (k = 5)	—	—	—	0.986 ± 0.004	−0.001

Note: Results are reported as mean ± standard deviation over 5 runs with different seeds. The best results are shown in bold. Δ denotes the absolute change w.r.t. the Full model.

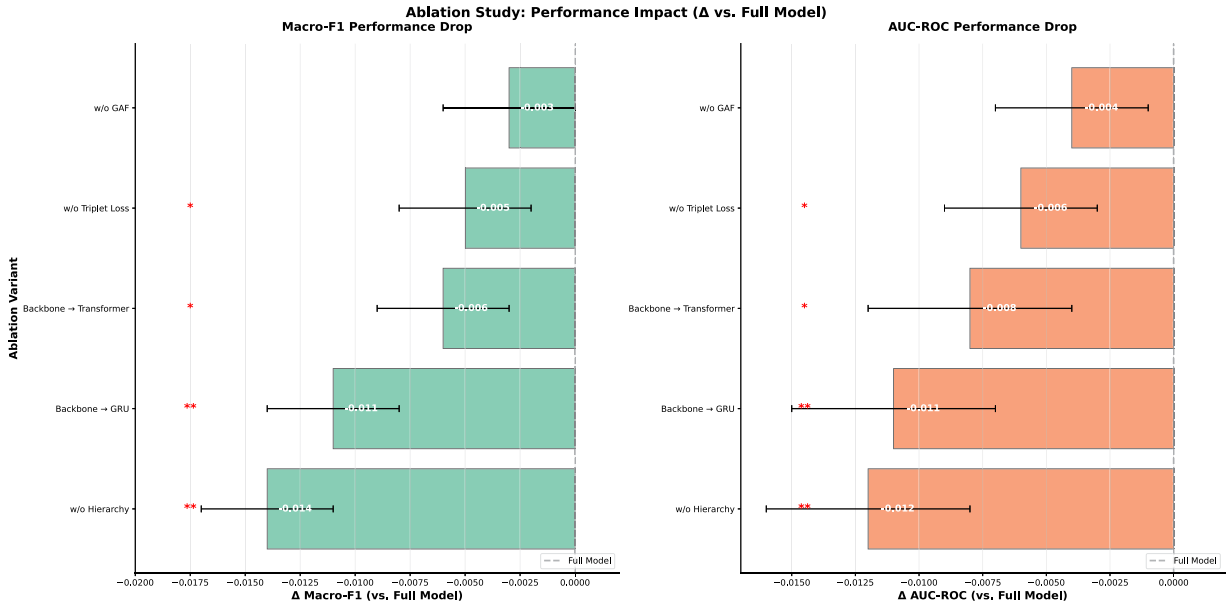

Figure 5: Ablation study: Performance impact of removing key components from the HMX-GAF model. Note: Statistical significance was evaluated against the full model over five independent runs using a two-sided paired t-test; *indicates $p < 0.05$, and **indicates $p < 0.01$. Error bars represent standard deviations.

(2) Perturbation Test. We injected a known irrelevant token into 10,000 flows; GAF achieved 96.2% top-1 precision in identifying it as least important (Table 5).

(3) Gradient Cross-Validation. We computed Spearman's rank correlation between GAF scores and gradient-based importance (L2-norm of output gradients w.r.t. input embeddings) over 10,000 test samples, obtaining a mean correlation of 0.88 (σ = 0.06). This confirms that GAF provides a faithful, zero-overhead alternative to expensive post-hoc gradient methods. For reference, generating SHAP explanations for a single flow requires ~500 forward passes (~14 s on Jetson Xavier NX), while LIME requires ~1000 perturbation samples (~28 s); GAF produces attributions within the same single forward pass used for classification.

Table 5: Quantitative validation results for GAF's attribution precision.

Evaluation Metric	Attribution Precision (%)
Top-1 Precision	96.2
Top-3 Precision	99.5

5 Discussion and Conclusion

5.1 Discussion

Generalization. Our evaluation uses CIC-IoT-2024 (97 device types). Generalizability to enterprise or industrial IoT environments remains to be verified through cross-dataset evaluation.

Scalability. The hierarchical architecture introduces a moderate memory increase over flat models (352 vs. 286 MB on Jetson Xavier NX). Scaling to ultra-high-throughput (≥10 Gbps) core networks may require FP16/INT8 quantization. Additionally, the default truncation to $L = 50$ packets covers 87% of flows in the CIC-IoT-2024 dataset; preliminary experiments at $L \in \{20, 50, 100, 200\}$ show diminishing returns beyond $L = 50$ (Macro-F1: 0.984, 0.992, 0.993, 0.993), and adaptive-length processing remains a direction for future work.

Explainability Scope. GAF attributions are tied to the model's internal Δ dynamics (cf. Section 3.2) [40]. Integrating external domain knowledge could further enhance actionability for SOC analysts.

5.2 Conclusion

This paper proposed HMX-GAF, a Hierarchical Mamba framework with Gated Attribution Fingerprinting for IoT traffic classification. The HMX hierarchical encoder achieves a Macro-F1 of 0.992 on CIC-IoT-2024 via dual-layer Mamba architecture; the GAF mechanism provides zero-cost attributions with 96.2% top-1 precision and 0.88 Spearman correlation with gradient-based importance; and edge evaluations on Jetson Xavier NX confirm 28.1 ms inference latency at batch size 1.

Future work will focus on: (1) cross-domain validation on heterogeneous datasets; (2) model compression via knowledge distillation and quantization-aware training; and (3) integrating GAF attributions with automated threat response systems.

Acknowledgement: The authors express their gratitude to Professor Chuanping Hu for his guidance and support throughout the study. His advice on academic exploration and manuscript writing has been a source of inspiration.

Funding Statement: The authors received no specific funding for this study.

Author Contributions: The authors confirm contribution to the paper as follows: study conception and design: Jiangyong Yu; data collection: Jiangyong Yu; analysis and interpretation of results: Jiangyong Yu, Runnan Wang; draft

manuscript preparation: Jiangyong Yu; supervision: Jiangyong Yu, Chuanping Hu. All authors reviewed and approved the final version of the manuscript.

Availability of Data and Materials: Data available on request from the authors.

Ethics Approval: Not applicable.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. MarketsandMarkets. Smart home market size, share & trends report, 2029 [Internet]. 2024 [cited 2026 Mar 15]. Available from: https://www.marketsandmarkets.com/Market-Reports/smart-homes-and-assisted-living-advanced-technologies-and-global-market-121.html.
2. Rose S, Borchert O, Mitchell S, Connelly S. Zero trust architecture (NIST special publication 800-207). Gaithersburg, MD, USA: National Institute of Standards and Technology; 2020. doi:10.6028/NIST.SP.800-207.
3. Rescorla E. The transport layer security (TLS) protocol version 1.3 (RFC 8446). Fremont, CA, USA: Internet Engineering Task Force; 2018. doi:10.17487/RFC8446.
4. Rescorla E, Oku K, Sullivan N, Wood CA. TLS encrypted client hello (RFC 9849). Fremont, CA, USA: Internet Engineering Task Force; 2026. doi:10.17487/RFC9849.
5. Vaswani A, Shazeer N, Parmar N, Uszkoreit J, Jones L, Gomez AN, et al. Attention is all you need. In: Proceedings of the 31st International Conference on Neural Information Processing Systems; 2017 Dec 4–9; Long Beach, CA, USA. p. 5998–6008.
6. Dai Z, Yang Z, Yang Y, Carbonell J, Le QV, Salakhutdinov R. Transformer-XL: attentive language models beyond a fixed-length context. In: Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics; 2019 Jul 28–Aug 2; Florence, Italy. p. 2978–88. doi:10.18653/v1/P19-1285.
7. Gu A, Goel K, Ré C. Efficiently modeling long sequences with structured state spacesl. In: Proceedings of the Tenth International Conference on Learning Representations; 2022 Apr 25–29; Virtual. p. 14323.
8. Gu A, Dao T. Mamba: linear-time sequence modeling with selective state spaces. arXiv:2312.00752. 2024.
9. Wang T, Xie X, Wang W, Wang C, Zhao Y, Cui Y. NetMamba: efficient network traffic classification via pre-training unidirectional Mamba. In: Proceedings of the IEEE 32nd International Conference on Network Protocols (ICNP); 2024 Oct 28–31; Charleroi, Belgium. p. 1–11. doi:10.1109/ICNP61940.2024.10858569.
10. Das A, Rad P. Opportunities and challenges in explainable artificial intelligence (XAI): a survey. arXiv:2006.11371. 2020.
11. Safi M, Dadkhah S, Shoeleh F, Mahdikhani H, Molyneaux H, Ghorbani AA. A survey on IoT profiling, fingerprinting, and identification. ACM Trans Internet Things. 2022;3(4):1–39. doi:10.1145/3539736.
12. Aouedi O, Piamrat K, Viho C. A survey of machine learning methods for IoT and smart environments device identification. ACM Comput Surv. 2023;55(10):1–38. doi:10.1145/3578935.
13. Kohno T, Broido A, Claffy KC. Remote physical device fingerprinting. IEEE Trans Dependable Secur Comput. 2005;2(2):93–108. doi:10.1109/TDSC.2005.26.
14. Xie L, Peng L, Zhang J, Hu A. Radio frequency fingerprint identification for Internet of Things: a survey. Secur Saf. 2024;3:2023022. doi:10.1051/sands/2023022.
15. Miettinen M, Marchal S, Hafeez I, Asokan N, Sadeghi AR, Tarkoma S. IoT SENTINEL: automated device-type identification for security enforcement in IoT. In: Proceedings of the IEEE 37th International Conference on Distributed Computing Systems (ICDCS); 2017 Jun 5–8; Atlanta, GA, USA. p. 2177–84. doi:10.1109/ICDCS.2017.283.
16. Okonkwo Z, Foo E, Li Q, Hou Z. A CNN-based encrypted network traffic classifier. In: Proceedings of the 2022 Australasian Computer Science Week; 2022 Feb 14–18; Brisbane, Australia. p. 74–83. doi:10.1145/3511616.3513107.
17. Rezaei S, Liu X. Deep learning for encrypted traffic classification: an overview. IEEE Commun Mag. 2019;57(5):76–81. doi:10.1109/MCOM.2019.1800819.

18. Wang W, Zhu M, Wang J, Zeng X, Yang Z. End-to-end encrypted traffic classification with one-dimensional convolution neural networks. In: Proceedings of the 2017 IEEE International Conference on Intelligence and Security Informatics (ISI); 2017 Jul 22–24; Beijing, China. p. 43–8. doi:10.1109/ISI.2017.8004872.
19. Rabbani M, Gui J, Nejati F, Zhou Z, Kaniyamattam A, Mirani M, et al. Device identification and anomaly detection in IoT environments. IEEE Internet Things J. 2025;12(10):13625–43. doi:10.1109/JIOT.2024.3522863.
20. Aceto G, Ciuonzo D, Montieri A, Pescapé A. Mobile encrypted traffic classification using deep learning: experimental evaluation, lessons learned, and challenges. IEEE Trans Netw Serv Manag. 2019;16(2):445–58. doi:10.1109/TNSM.2019.2899085.
21. Lotfollahi M, Jafari Siavoshani M, Shirali Hossein Zade R, Saberian M. Deep packet: a novel approach for encrypted traffic classification using deep learning. Soft Comput. 2020;24(3):1999–2012. doi:10.1007/s00500-019-04030-2.
22. Lin X, Xiong G, Gou G, Li Z, Shi J, Yu J. ET-BERT: a contextualized datagram representation with pre-training transformers for encrypted traffic classification. In: Proceedings of the ACM Web Conference (WWW); 2022 Apr 25–29; Virtual. p. 633–42. doi:10.1145/3485447.3512217.
23. Zhao R, Zhan M, Deng X, Wang Y, Wang Y, Gui G, et al. Yet another traffic classifier: a masked autoencoder based traffic transformer with multi-level flow representation. Proc AAAI Conf Artif Intell. 2023;37(4):5420–7. doi:10.1609/aaai.v37i4.25674.
24. Fu DY, Dao T, Saab KK, Thomas AW, Rudra A, Ré C. Hungry hungry hippos: towards language modeling with state space models. In: Proceedings of the Eleventh International Conference on Learning Representations; 2023 May 1–3; Kigali, Rwanda.
25. Dao T, Gu A. Transformers are SSMs: generalized models and efficient algorithms through structured state space duality. arXiv:2405.21060. 2024.
26. Xu J, Chen L, Xu W, Dai L, Wang C, Hu L. ET-Mamba: a Mamba model for encrypted traffic classification. Information. 2025;16(4):314. doi:10.3390/info16040314.
27. Chu A, Jiang X, Liu S, Bhagoji AN, Bronzino F, Schmitt P, et al. Feasibility of state space models for network traffic generation. In: Proceedings of the 2024 SIGCOMM Workshop on Networks for AI Computing; 2024 Aug 4–8; Sydney, NSW, Australia. p. 9–17. doi:10.1145/3672198.3673792.
28. Chandroth J, Ali J. A lightweight MLP-based feature extraction with linear classifier for intrusion detection system in Internet of Things. Electronics. 2026;15(8):1604. doi:10.3390/electronics15081604.
29. Govindarajan V, Selvam L, Ravi V, Sowmya V, Soman KP. Aegis-5: a hybrid ensemble framework for intrusion detection in Industry 5.0 driven smart manufacturing environment. ACM Trans Auton Adapt Syst. 2026:3787224. doi:10.1145/3787224.
30. Rjoub G, Bentahar J, Wahab OA, Mizouni R, Cohen R, Otrok H, et al. A survey on explainable artificial intelligence for cybersecurity. IEEE Trans Netw Serv Manag. 2023;20(4):5115–40. doi:10.1109/TNSM.2023.3282740.
31. Khani P, Moeinaddini E, Dehghan Abnavi N, Shahraki A. Explainable artificial intelligence for feature selection in network traffic classification: a comparative study. Trans Emerg Telecomm Technol. 2024;35(4):e4970. doi:10.1002/ett.4970.
32. Ribeiro MT, Singh S, Guestrin C. "Why should I trust you?": explaining the predictions of any classifier. In: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining; 2016 Aug 13–17; San Francisco, CA, USA. p. 1135–44. doi:10.1145/2939672.2939778.
33. Lundberg SM, Lee SI. A unified approach to interpreting model predictions. In: Proceedings of the 31st International Conference on Neural Information Processing Systems; 2017 Dec 4–9; Long Beach, CA, USA. p. 4765–74.
34. Gaspar D, Silva P, Silva C. Explainable AI for intrusion detection systems: LIME and SHAP applicability on multilayer perceptron. IEEE Access. 2024;12(11):30164–75. doi:10.1109/ACCESS.2024.3368377.
35. Jain S, Wallace BC. Attention is not explanation. In: Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies; 2019 Jun 2–7; Minneapolis, MN, USA. p. 3543–56. doi:10.18653/v1/N19-1357.
36. Wang Z, Chen H, Yang S, Luo X, Li D, Wang J. A lightweight intrusion detection method for IoT based on deep learning and dynamic quantization. PeerJ Comput Sci. 2023;9(19):e1569. doi:10.7717/peerj-cs.1569.

37. Sivanathan A, Gharakheili HH, Loi F, Radford A, Wiber C, Vishwanath A, et al. Classifying IoT devices in smart environments using network traffic characteristics. IEEE Trans Mob Comput. 2019;18(8):1745–59. doi:10.1109/TMC.2018.2866249.
38. Garcia S, Parmisano A, Erquiaga MJ. IoT-23: a labeled dataset with malicious and benign IoT network traffic (Version 1.0.0) [Internet]. 2020 [cited 2026 Jan 1]. Available from: https://zenodo.org/records/4743746.
39. Huang X, Khetan A, Cvitkovic M, Karnin Z. TabTransformer: tabular data modeling using contextual embeddings. arXiv:2012.06678. 2020.
40. Manna S, Sett N. Reconciling privacy and explainability in high-stakes: a systematic inquiry. Trans Mach Learn Res. arXiv:2412.20798. 2025.