ingegneria sismica
International Journal of Earthquake Engineering
Anno XLIII - Num. 2 - 2026

Cybersecurity Solutions for Smart Home Networks Based on Embedded Artificial Intelligence

Jiahong Zhou[1,*]

[1] Information Engineering Department, Eastern Liaoning University, Dandong 118000, Liaoning, PR China. Safety Technology and Engineering
[2] No.632-1,1-24-1, Shenshui Road, Shenhe District, Shenyang City, Liaoning Province

SUMMARY: *Smart home networks consist of heterogeneous devices, fragmented protocols, and weakly consistent security policies, making household gateways vulnerable to device takeover, botnet expansion, credential abuse, and abnormal control behaviors. To address the joint constraints of detection accuracy, local deployment, and response timeliness, this paper proposes an embedded artificial intelligence-based cyber protection scheme for smart home networks, named EAI-SHGuard. The scheme unifies traffic statistics, device profiles, and state deviation signals into window-level security objects, and introduces a dual-branch lightweight detector that combines depthwise temporal convolution, linear attention, context-gated fusion, prototype regularization, and quantization-aware training. A risk grading module, furthermore, maps the results of detection into classified local processing measures, which include logging, speed restriction and isolation processing. Experiments were conducted on CICIoT2023, N-BaIoT, and ToN_IoT using standardized preprocessing and embedded deployment metrics. Under this protocol, EAI-SHGuard achieved Macro-F1 scores of 99.13%, 98.41%, and 97.84% on the three datasets, with an average of 98.46%. The model size is reduced to 3.6 MB, and the single-sample inference latency on a Raspberry Pi 4B is 6.1 ms. Case analysis further demonstrates that the proposed scheme can mitigate Mirai-like camera outbreaks and brute-force smart lock attacks through differentiated local responses. The results indicate that embedded AI can provide a practical security layer for smart home gateways when model compactness, contextual awareness, and policy linkage are considered together.*

KEYWORDS: *embedded artificial intelligence; smart home network; intrusion detection; edge security; cyber protection*

1 Introduction

Levisions, illumination controllers, and environment sensors keep the continuous connection to the network. The home route devices and edge control points need at the same time to handle the Wi-Fi, Bluetooth, Zigbee, and application layer control flow data. Inconsistencies coming from device brands, firmware versions, protocol stacks, and default security policies thus make a home network that is usually characterized by "multiple endpoints, diverse interfaces, and weak boundaries." After an attacker obtains entry to the home network through utilizing weak passwords, firmware loopholes, or exposed services, the attack flow does not keep being restricted to one single endpoint. On the contrary, it extends to other nodes through LAN broadcast programs, device searching, and controlling interfaces, hence further damaging

*13504074545@sohu.com
https://doi.org/10.65102/is2026752
Published: 30 April, 2026

privacy, access control, and remote working safety. Therefore, the core requirement for smart home network security protection is no longer merely detecting anomalous traffic, but rather promptly identifying, classifying, and mitigating threats on resource-constrained home devices.

Existing research has provided a relatively comprehensive analysis of the attack surface and security requirements for smart homes. System mapping and review studies indicate that risks in home networks are not limited to traditional network intrusions but also include device takeover, data theft, abuse of home automation rules, and lateral movement following the exploitation of edge control nodes [1-3]. Studies of real-world deployment environments further demonstrate that even though open-source or commercial home automation platforms possess certain authentication and access control capabilities, misconfigurations, exposed interfaces, and inadequate update mechanisms continue to create attack vectors [4, 5]. This implies that smart home security research cannot focus solely on cloud-based services or enterprise-level networks; instead, it must return to the most common observation points within the home LAN-namely, home gateways, edge controllers, and protocol adaptation nodes.

With regard to detection methods, the research on IoT security is currently moving from the simple matching based on rules to the anomaly detection and attack recognition which are driven by data. When comparing with early studies which used artificially set features and small-size samples, on one hand, recent work has, via summaries of security data sets, made clear the distinctions of data size, label arrangement, and scene reality in IoT intrusion detection; on the other hand, thus it has also pushed the inclusion of bigger-scale public data collections into main-stream assessment procedures [6]. From the angle of data foundation, N-BaIoT gives a device-level observation angle on botnet attacks, ToN_IoT further adds telemetry and mixed environment features, while CICIoT2023 brings multi-device, large-scale, multi-class attack traffic into a unified data frame [7-9]. Although these public resources have greatly promoted the promotion of model training and evaluation conditions, they also bring new challenges: protocol semantic meanings are different between different datasets, the device background in home terminals is often not given enough attention, and the "device type-control behavior-traffic anomaly" relationship, which has strong correlation with home scenarios, is still not fully used.

Along with the development of these data, light-weight detection models which are specially made for edge and embedded platforms have already started to come forth. Methods including TinyIDS, Stacking TinyML, and dynamic quantization ways have proven that putting intrusion detection functions upon resource-limited apparatuses is currently technically attainable, whereby model dimension, inference delay, and energy use are all decreased to the smallest degrees [10-12]. This method possesses direct connection to intelligent residences, because safety consciousness on the household level can not depend on the cloud forever. Overly frequent upload actions bring about enlargement of privacy exposure scope, while models that are too complicated surpass the computation holding ability of household gateway devices and low-power edge nodes. However, the majority of current lightweight researches all concentrate on one single public dataset, they put detection accuracy and parameter compression in the first place, there is not much discussion about how to bring the contextual differences between different household devices into the model, and also not about the integrating of detection results with local response mechanisms.

Another important tendency is the combination of federated learning, zero-trust systems, and explainability research into IoT intrusion checking. Federated learning lowers the requirement for uploading original data, zero-trust frameworks lay stress on uninterrupted verification, and interpretability approaches assist in alleviating the "black-box" problem within safety systems. Although these researches already discussed old problems in IoT situations-including private keeping, co-operation renewing, and believable degree checking-three

problems still exist when they are used in the smart home edge side. Firstly, numerous federated or cooperative detection projects suppose that clients own steady partial training situations, this is not completely consistent with the actual situation that has fluctuating equipment calculation ability, non-independent identical distribution data, and restricted online maintenance ability inside family environments [13-15]. Second, the research on explainability, it is often built upon server-side models or large-scale deep neural networks, these structures can promote the transparency of analysis but thus may not be fit for routine inference work on home gateways [16-18]. Third, most current researches regard "correct detection" as the final objective, with few discussions of the follow-up measures that are truly needed in smart home situations-that is, whether to limit bandwidth, whether to separate the device, and when to record events without blocking them.

Consequently, there is a significant research gap in smart home cybersecurity: on the one hand, protection systems need to leverage multi-source observations to improve their ability to identify complex home attacks; on the other hand, models must meet the constraints of embedded deployment regarding size, latency, and local execution; furthermore, detection results should be tradable into tiered response strategies that are practically meaningful for the home network.If these three elements cannot work together within a unified framework, home security systems will easily oscillate between being "accurate enough but impractical" and "deployable but ineffective."

Based on this, this paper proposes EAI-SHGuard, an embedded AI-based smart home network security protection solution, focusing on smart home gateways and edge control nodes. Our work concentrates on the following three aspects. First, we develop a multi-source security data organization method tailored for home networks, unifying traffic statistics, device profiles, and state anomalies into window-level detection objects suitable for embedded processing.Second, we design a lightweight, two-branch detection model and integrate context-based gating, prototype constraints, and quantization-aware training into a single inference chain, enabling the model to meet home edge deployment requirements while maintaining its recognition capabilities. Third, we establish an evaluation protocol covering multiple public datasets, cross-device generalization, and local collaborative response, validating the solution’s effectiveness across four dimensions: detection accuracy, false positive control, inference overhead, and response timeliness.

2 Methods

2.1 Smart Home Threat Modeling and Sample Construction

This article takes the home gateway as the main observation node, and regards cameras, smart door locks, televisions, speakers, power outlets, lighting controllers, and protocol bridges as objects that need protection. The gateway does the aggregation of traffic summaries, device working states, and control behavior logs from both IP devices and non-IP home protocol adapters. The attack attacking surface mainly mainly comprises three big categories: first, brute-force force attempts and device device takeovers resulting from weak weak authentications and default default passwords; second, botnet behaviors, scanning, fake address disguising, and reflection striking started by broken terminal devices; and third, the abnormal control sequences that are triggered by the misuse of home automation interaction interfaces. Because in smart home environments data packets are frequently short, sessions are broken into pieces, and control behaviors have periodic features, thus relying only on original packets or individual traffic indexes often cannot at the same time describe both device identity and behavioral abnormal conditions. Hence, this paper uses a multi-source object arrangement method that

combines "network flow + device feature description + state offset," which is displayed in Equation (1).

$$\mathbf{z}_t = \left[\phi_n(\mathbf{x}_t^{net}) \,\|\, \phi_d\left(\mathbf{x}_t^{dev}\right) \,\|\, \phi_s(\mathbf{x}_t^{state})\right] \tag{1}$$

In the equation, x_t^{net} represents the network-side feature vector at time t, comprising 28 flow-level statistics such as packet count, byte count, uplink-to-downlink ratio, port distribution, session duration, TCP flag statistics, and arrival intervals; x_t^{dev} represents device-side contextual features, comprising 12 profiling metrics such as device type, expected port set, common communication directions, active time periods, and protocol usage preferences; x_t^{state} represents state deviation features, including 8 state metrics such as CPU utilization, abnormal connection counts, changes in control command frequency, and firmware update flags. The functions ϕ_n, ϕ_d, and ϕ_s represent normalization, discrete embedding, and scale-aligned mapping, respectively, while the symbol $\|$ denotes vector concatenation. After processing, a 48-dimensional unified feature vector is obtained for each time window.

To balance real-time performance with behavioral continuity, this paper employs a sliding window of 5 seconds in length with a 1-second step size, and concatenates $L = 12$ consecutive windows to form a detection sequence.This approach serves two purposes: first, to preserve short-term burst characteristics common in smart home attacks, such as directional spikes during Mirai scans or password brute-force attacks; second, to preserve the temporal dependencies of control behaviors, such as retry patterns following failed door lock authentication, or the differences between normal camera cloud synchronization and abnormal external connections. To illustrate how network-side, device-side, and state-side information are aggregated into a unified detection target, see Figure 1.

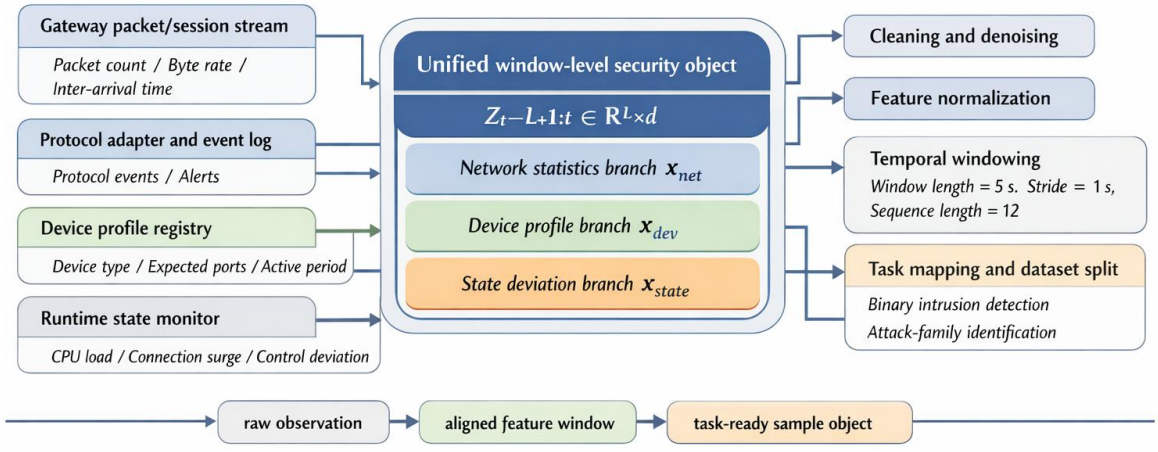

Figure 1: Organization of multi-source smart home security data and sample construction mechanism.

This paper selects three types of public datasets as the primary evaluation sources and uses two additional data resources for threshold calibration and scenario replay. Specifically, CICIoT2023 provides various types of IoT attack traffic, N-BaIoT offers a device-level botnet perspective, and ToN_IoT telemetry data is used to supplement heterogeneous sensor and gateway behavior;Edge-IIoTset is used for open-scenario threshold calibration, and IoT-23 is used for typical attack process replay [19, 20]. To adapt to the home gateway scenario, this paper divides the core tasks into two levels: the first is a binary classification intrusion detection task shared by the three datasets; the second is a six-class attack family identification task based

on CICIoT2023, used to drive subsequent tiered response.The mapping of public datasets to smart home scenarios is shown in Table 1.

Table 1: Mapping of Public Datasets to Smart Home Scenarios

Dataset	Smart Home Scenario Mapping	Unified Task	Balanced Sample Size	Data Split	Role in This Paper
CICIoT2023 [7]	Home gateway traffic from mixed appliances, cameras, sockets, and voice terminals	Binary classification + six types of attack family recognition	1,200,000	70%/10%/20%	Main training set and main test set
N-BaIoT [8]	Devices like cameras, routers, and doorbells taken over by botnets	Binary classification	180,000	Leave-one-device extrapolation + 20% holdout	Cross-device generalization validation
ToN_IoT Telemetry [9]	Anomalies in mixed environments of sensing devices, edge nodes, and gateways	Binary classification	240,000	70%/10%/20%	Cross-protocol transfer validation
Edge-IIoTset [19]	More complex external edge environments	Open scenario threshold calibration	120,000	Calibration set	OOD threshold and false alarm control
IoT-23 [20]	Replay of attack processes after home devices are controlled	Scenario replay	Scene slice subset	Case replay	Temporal disposition analysis

2.2 Embedded AI Detection and Coordinated Response Model

After we have finished building a one-piece data collection, the detecting device must meet two groups of demands: one comes from smart home attack things themselves, therefore it needs the capability to recognize both short-time sudden flow and long-lasting behavior differences; the other one comes from embedded deployment, this means the model cannot depend on large-scale global attention mechanisms or structures with high dimension and wide weight. According to this foundation, EAI-SHGuard uses a two-branch structure that is composed of "partial abrupt change modeling + long-distance connection modeling + context-door combination." The part of local uses deep separable one-dimensional convolution operations to extract abnormality at packet level and direction changes inside short time windows, while the part of relation uses linear attention method to code dependencies between different windows, therefore making it possible to recognize low-frequency but continuous reconnaissance,

detection and gradual occupation behaviors. The device profile summation vectors take part in the gated fusion, thus they ensure that risk explanation for the identical traffic pattern keep different among the different home devices. The connection between the outputs of two branches and the context-gated mechanism is defined like what is shown in Equation (2).

$$\mathbf{h}_t^c = \mathrm{DSConv}(\mathbf{Z}_{t-L+1:t}) \tag{2a}$$

$$\mathbf{h}_t^a = \mathrm{LinAttn}(\mathbf{Z}_{t-L+1:t}) \tag{2b}$$

where $\mathbf{Z}_{t-L+1:t}$ denotes an input sequence consisting of L consecutive windows; $\mathbf{h}_t^c$ and $\mathbf{h}_t^a$ represent the output representations of the convolutional branch and the linear attention branch, respectively, as shown in Equation (3).

$$\alpha_t = \mathrm{softmax}\left(W_g \mathbf{c}_t + b_g\right) \tag{3a}$$

$$\mathbf{h}_t = \alpha_{t,1}\mathbf{h}_t^c + \alpha_{t,2}\mathbf{h}_t^a \tag{3b}$$

Here, $\mathbf{c}_t$ denotes the device profile summary vector; W_g and b_g are gating parameters; and α_t are branch weights. This design enables the model to automatically adjust the attention ratio between "sudden anomalies" and "persistent anomalies" based on context across devices such as cameras, door locks, and power outlets.

Given the significant class imbalance in smart home datasets and the requirement for stable classification performance after INT8 quantization in embedded deployments, this paper introduces class prototype compactness constraints and quantization consistency constraints in addition to the standard cross-entropy. The overall loss function is shown in Equation (4).

$$\mathcal{L} = \mathcal{L}_{ce} + \lambda \sum_{i=1}^{N} \left\| \mathbf{h}_i - \mu_{y_i} \right\|_2^2 + \beta \|\theta - Q(\theta)\|_2^2 \tag{4}$$

In the equation, $\mathcal{L}_{\mathrm{ce}}$ denotes the cross-entropy loss; $\mathrm{h_i}$ represents the fused representation of the i th sample; μ_{y_i} is the class prototype center corresponding to label y_i; θ denotes the full-precision parameters; $Q(\cdot)$ represents the INT8 quantization operator; and λ and β are balancing coefficients.Prototype constraints are used to reduce the dispersion of samples within the same class in the low-dimensional embedding space, thereby enhancing the stability of low-frequency attack classes; quantization consistency terms are used to minimize distribution shifts between full-precision training and quantized deployment.

The detection output is not directly equivalent to a protective action. To avoid widespread disconnections in home networks caused by false positives, this paper further introduces a risk scoring and tiered response mechanism, as shown in Equation (5).

$$r_t = \sum_{c \in \mathcal{A}} p_t(c)\omega_c + \eta \delta_t \tag{5a}$$

$$u_t = \begin{cases} \text{allow}, & r_t < \tau_1, \\ \text{rate\mbox} - \text{limit}, & \tau_1 \le r_t < \tau_2, \\ \text{quarantine}, & r_t \ge \tau_2. \end{cases} \tag{5b}$$

In the equation, $\mathcal{A}$ denotes the set of attack labels; $p_t(c)$ is the predicted probability that at time t, the device belongs to attack class c; ω_c is the attack severity weight; δ_t is the

device status deviation score; η is the deviation score weight; τ_1 and τ_2 represent the throttling threshold and isolation threshold, respectively; u_t is the gateway output action. This mechanism ensures that external network scans by cameras, consecutive authentication failures of door locks, and abnormal broadcasts from power outlets are not treated uniformly, but are routed into one of three handling paths-logging, throttling, or isolation-based on risk severity.

From the perspective of deployment and execution, the parameter quantity of EAI-SHGuard is maintained within 1.12 million. The convolution-type branch channels employ depth-split convolution operations in place of regular convolution operations, whereas the relation-type branch channels adopt linear attention mechanisms instead of attention mechanisms with quadratic complexity. On the inference aspect, the INT8 model and a streaming type window buffering are utilized by it. The direct aim of this construction is to decrease model dimension and per-sample time delay, thus enabling the model which to reside upon the home gateway and work continuously without depending upon outside servers. For illustrating the functional relation between the detection model and the local cooperative proposal, please look at Figure 2.

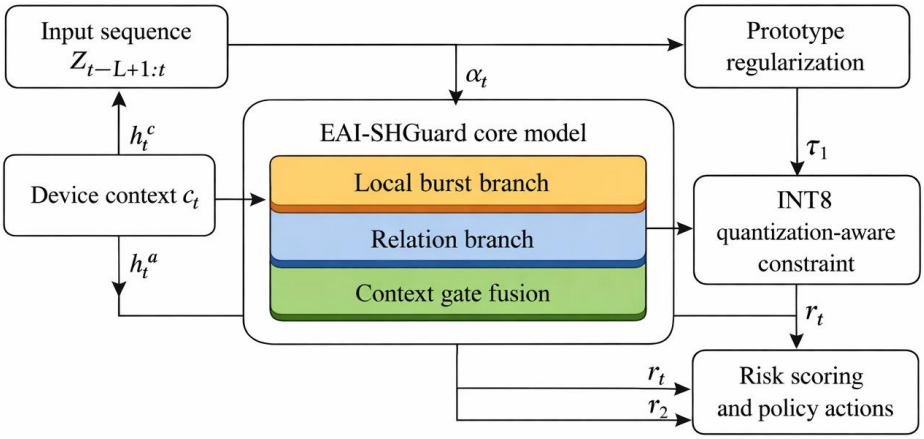

Figure 2: Diagram of the EAI-SHGuard Embedded Detection and Collaborative Response Mechanism.

2.3 Experimental Setup and Evaluation Protocol

To ensure that evaluation results reflect both detection capabilities and deployment value, this paper adopts a unified preprocessing, training protocol, and embedded measurement environment. The main experiment uses a 70%/10%/20% split for training, validation, and testing, respectively. Additionally, N-BaIoT employs a "leave-one-device-out" extrapolation setting to verify the model's transfer stability on new devices.The training phase employs the AdamW optimizer, with an initial learning rate of 1×10^{-3}, a batch size of 256, and a maximum of 50 training epochs. Training is terminated early if there is no improvement in validation set performance for 7 consecutive epochs. Quantization-aware training is introduced during the final 15 epochs to prevent the model from being overly affected by quantization noise during the early feature learning phase.

The comparison models fall into two categories. The first category consists of lightweight or embeddable detection models, such as TinyIDS, Stacking TinyML, and dynamic quantization-based IDS, to assess the limits of resource-efficient solutions in home environments.The second category comprises representative IoT deep detection models, including federated/zero-trust-related approaches and high-performance deep architecture reproduction baselines. The main table presents Stacking TinyML, DeepAK-IoT, and TFKAN, while the remaining models-ABCNN-IDS, Deep Transfer Detection, and CAEAID-are used for

low-false-alarm ROC curves and supplementary analysis [21-25].All models were retrained and evaluated using the same feature set to minimize performance biases caused by differences in data organization.

Evaluation metrics are divided into two groups: detection metrics and deployment metrics. Detection metrics include Accuracy, Precision, Recall, Macro-F1, AUC, and FPR. Deployment metrics include model size, per-sample inference latency, average CPU utilization, inference power consumption, and the response latency from the first malicious window to the triggering of a policy action.In this paper, the latter is referred to as the average response delay. Its practical significance aligns more closely with the security requirements of smart home environments than simple classification delay, as the primary concerns in a home setting are when to throttle or isolate a device, and whether policy actions will cause significant interference with normal control chains.

Embedded deployment testing was conducted on a Raspberry Pi 4B (4 GB RAM, Cortex-A72 1.5 GHz), with supplementary measurements taken on an RK3568 home control board and a Jetson Nano.All models utilized the same streaming input buffer and single-sample online inference approach; power consumption was measured as the steady-state average over 500 inference windows using an external power meter. To mitigate sporadic interference, all latency results are reported as medians, and a three-variable coupling relationship is presented in conjunction with model size and accuracy. To illustrate the unified relationship between dataset partitioning, model comparison, and embedded measurements, see Figure 3.

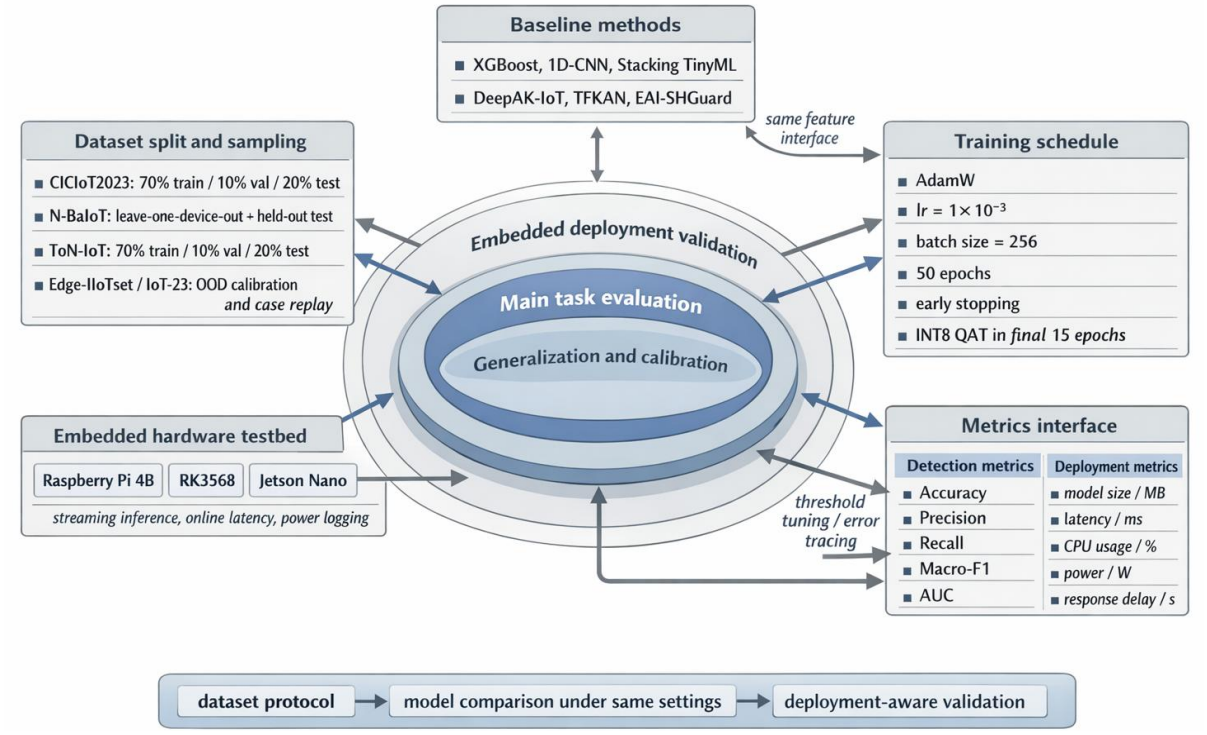

Figure 3: Unified evaluation protocol for dataset organization, baseline comparison, and embedded deployment measurement.

As shown in Figure 3, the experimental design in this paper does not treat data partitioning, model training, and hardware deployment as separate processes, but rather integrates main task evaluation, generalization calibration, and embedded validation into a single protocol framework.CICIoT2023 and ToN-IoT are used for main training and standard testing; N-BaIoT is used to evaluate cross-device generalization capabilities through leave-one-out extrapolation;

and Edge-IIoTset and IoT-23 are responsible for open-scenario threshold calibration and attack process replay, respectively.The baseline model and EAI-SHGuard are compared under identical input interfaces, training schedules, and evaluation metrics to avoid superficial advantages resulting from differences in data organization.Figure 3 further illustrates that detection metrics such as Accuracy, Macro-F1, and AUC are not the only outputs; model size, inference latency, CPU utilization, power consumption, and response latency are also transmitted back to the evaluation system via a unified measurement interface on the Raspberry Pi 4B, RK3568, and Jetson Nano.

3 Results and Discussion

3.1 Overall Detection and Protection Performance

This section first examines the main results of EAI-SHGuard under a unified experimental protocol. To avoid drawing localized conclusions based on a single dataset, Table 2 reports the model's performance across three datasets: CICIoT2023, N-BaIoT, and ToN_IoT.

Table 2: Comparison of Key Results for Different Models Across Three Datasets

Model	CICIoT2023 Acc/%	CICIoT2023 Macro-F1/%	N-BaIoT Acc/%	N-BaIoT Macro-F1/%	ToN_IoT Acc/%	ToN_IoT Macro-F1/%	Avg. Macro-F1/%
XGBoost	97.86	97.14	97.31	96.55	95.74	94.89	96.19
1D-CNN	98.43	97.96	98.22	97.41	96.81	95.93	97.10
Stacking TinyML [11]	98.94	98.58	98.67	97.95	97.42	96.88	97.80
DeepAK-IoT [22]	99.08	98.81	98.91	98.22	97.89	97.11	98.05
TFKAN [24]	99.19	98.94	99.04	98.37	98.11	97.34	98.22
EAI-SHGuard	99.42	99.13	99.21	98.41	98.67	97.84	98.46

In Table 2, EAI-SHGuard achieves Macro-F1 scores of 99.13%, 98.41%, and 97.84% on the three datasets, respectively, with a cross-dataset average of 98.46%, outperforming the other comparison models in the table.Compared to XGBoost and 1D-CNN, which have lower resource overhead but limited expressive power, the proposed approach demonstrates a more significant improvement on ToN_IoT, indicating that relying solely on single traffic statistics makes it difficult to maintain stable classification when protocol types and operational states become more complex.Compared to Stacking TinyML, EAI-SHGuard achieved an average Macro-F1 score improvement of 0.66 percentage points, indicating that introducing relational branches and context-gated mechanisms within a lightweight model framework is meaningful.Compared to high-performance deep models such as DeepAK-IoT and TFKAN, EAI-SHGuard still maintains an average advantage of 0.41 and 0.24 percentage points, respectively, indicating that embedded constraints do not significantly weaken the model's ability to represent complex attack patterns. To compare the classification stability of different models within the low false-positive range, see Figure 4.

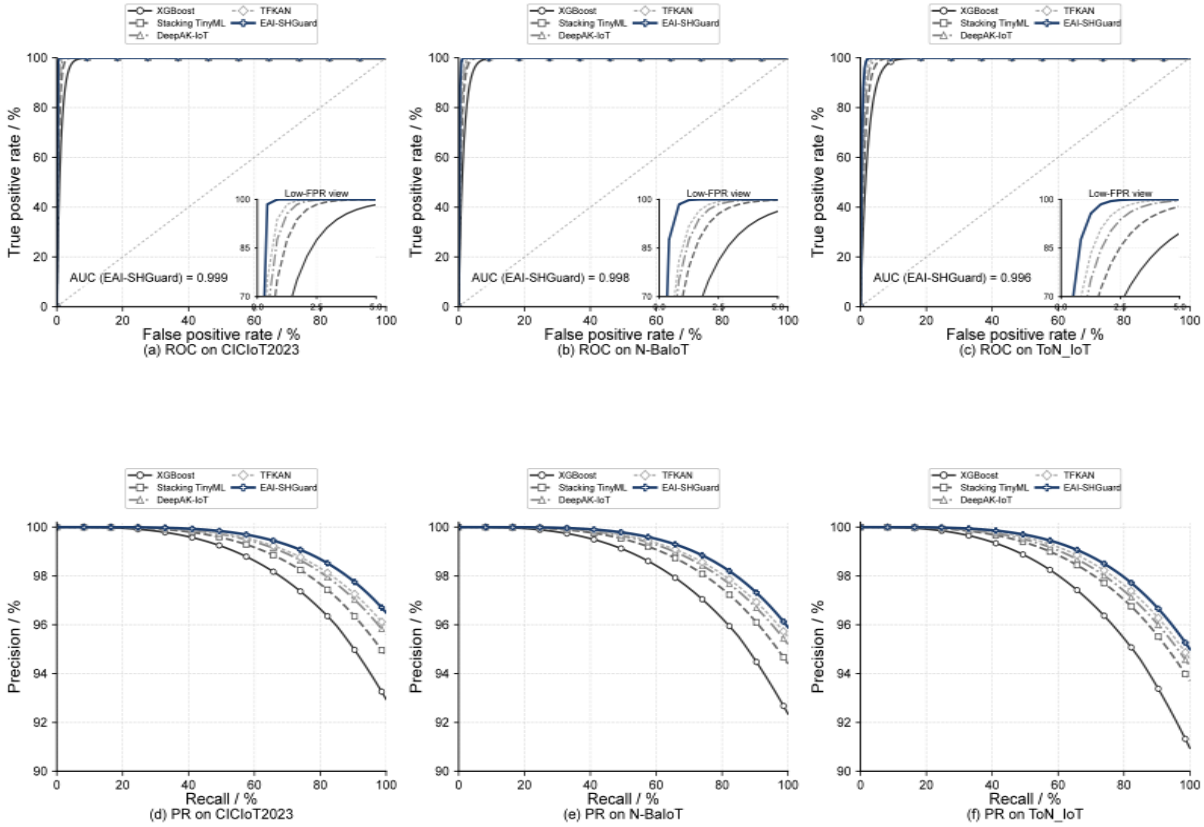

Figure 4: ROC-PR combination results for the three datasets.

In Figure 4, the ROC curves of EAI-SHGuard for all three datasets lie within the outer envelope region, with AUC values reaching 0.999, 0.998, and 0.996 on CICIoT2023, N-BaIoT, and ToN-IoT, respectively, corresponding to FPRs of 0.43%, 0.56%, and 0.71%. Correspondingly, the PR curves maintain high accuracy even in the high-recall range, indicating that the model does not rely on aggressive thresholds to achieve superficial performance but instead maintains stable recognition capabilities under low-FPR conditions.When examining individual datasets, the overall performance gap among models on CICIoT2023 is small, but EAI-SHGuard still maintains the outermost curve; on N-BaIoT, its advantage is primarily reflected in the retention of recall in the low false positive rate range; on ToN-IoT, the gap between models widens further, indicating that under conditions of more complex protocols and stronger telemetry perturbations, the proposed method demonstrates more stable adaptability to heterogeneous scenarios.Combining the ROC and PR results, Figure 4 demonstrates that the advantages of our method are reflected in both false positive control and high recall retention, rather than excelling in a single evaluation metric. To further illustrate the model's discrimination performance across different attack families, see Figure 5.

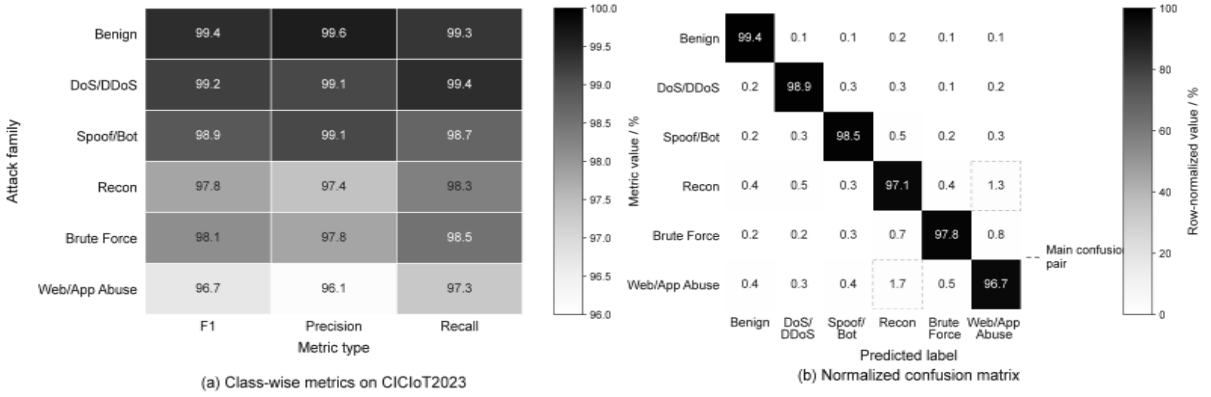

Figure 5: CICIoT2023 attack family identification heatmap and misclassification migration map.

In Figure 5, the F1 scores for Benign, DoS/DDoS, and Spoof/Bot are 99.4%, 99.2%, and 98.9%, respectively,with Precision at 99.6%, 99.1%, and 99.1%, respectively, and Recall at 99.3%, 99.4%, and 98.7%, respectively. This indicates that the model exhibits strong stability against high-frequency perturbations and medium-to-high-intensity attack patterns.For Recon, Brute Force, and Web/App Abuse, the F1 scores were 97.8%, 98.1%, and 96.7%, respectively. Among these, the Precision and Recall for Web/App Abuse dropped to 96.1% and 97.3%, respectively, making it the lowest-performing group among all categories.The normalized confusion matrix further indicates that the misclassification rate from Recon to Web/App Abuse is 1.3%, while the misclassification rate from Web/App Abuse to Recon is 1.7%, constituting the primary misclassification migration pair; in contrast, most other off-diagonal elements remain within the 0.1%-0.8% range.These results indicate that the model's errors do not propagate widely across categories but are primarily concentrated between semantically similar categories-lightweight reconnaissance and application-layer abuse-which aligns with the actual evolution of low-intensity anomalous behavior in smart home scenarios.

3.2 Module Ablation, Cross-Scene Generalization, and Sources of Error

After confirming the overall performance, we need to address a more critical question: Does the performance improvement of EAI-SHGuard stem from the model's overall architecture, or from a specific module? To this end, Table 3 presents the results of module ablation and embedded overhead comparisons.

Table 3: Comparison of Module Ablation and Embedded Overhead

Variant	CICIoT2023 Macro-F1/%	Low-Frequency Attack Recall/%	FPR/%	Model Size/MB	Single Sample Latency/ms
Complete Model	99.13	96.02	0.43	3.6	6.1
Remove Device Context Gating	97.41	93.18	0.88	3.4	5.8
Remove Relation Branch	97.89	94.22	0.79	2.9	4.9
Remove Prototype Constraints	98.24	93.71	0.72	3.6	6.0
Retain FP32, Remove Quantization Consistency Constraints	99.24	96.11	0.41	11.4	9.7

Table 3 shows that after removing device context-based gatekeeping, Macro-F1 decreased by 1.72 percentage points, recall for low-frequency attacks decreased by 2.84 percentage points, and FPR rose from 0.43% to 0.88%.This result indicates that in a smart home environment, merely observing whether "traffic resembles an attack" is insufficient; the risk semantics of the same pattern differ across cameras, door locks, and televisions. Only when device profiles are incorporated into the gating mechanism can the model form more stable, scenario-specific judgments. After removing the relationship branch, the recall for low-frequency attacks decreased by 1.80 percentage points, suggesting that continuous reconnaissance and incremental brute-force attacks rely heavily on cross-window relationship modeling.After removing prototype constraints, the model's recall for sparse categories dropped most significantly, indicating that the compactness of the embedding space directly impacts attacks with few samples.

Another noteworthy phenomenon is the role of quantization constraints. If FP32 is retained and the quantization consistency term is removed, the full-precision model's F1 score increases by only 0.11 percentage points, but the model size rises from 3.6 MB to 11.4 MB, and the latency per sample increases from 6.1 ms to 9.7 ms.For home gateways, this accuracy gain is insufficient to offset the storage and response costs. These results demonstrate that quantization-aware training is not merely a post-deployment step, but rather an essential consideration that must be incorporated into the design of embedded security models from the outset. To explain the source of the model's advantages and the basis for threshold selection, see Figure 6.

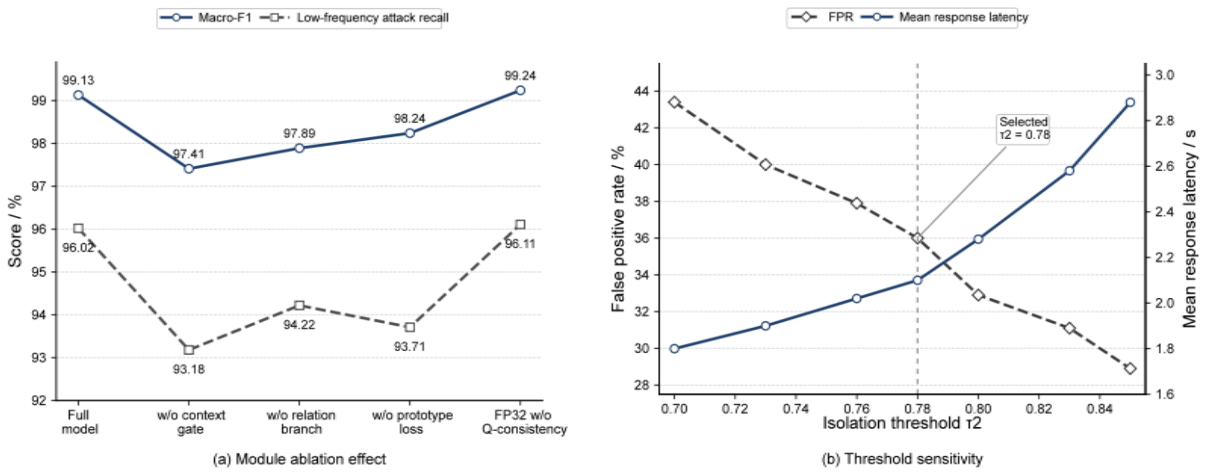

Figure 6: Module ablation and threshold sensitivity coupling curve.

Figure 6 explains the sources of model performance from the perspectives of structural ablation and threshold sensitivity. The full model achieves a Macro-F1 of 99.13%, a recall for low-frequency attacks of 96.02%, and an FPR of 0.43%.When the device context gating is removed, the Macro-F1 drops to 97.41%, the recall for low-frequency attacks drops to 93.18%, and the FPR rises to 0.88%, indicating that the contextual boundaries provided by the device profile play a direct role in suppressing false positives.After removing relationship branches, the Macro-F1 score was 97.89%, the recall for low-frequency attacks was 94.22%, and the FPR was 0.79%, indicating that modeling cross-window dependencies is particularly critical for persistent reconnaissance and incremental brute-force attacks.After removing prototype constraints, Macro-F1 dropped to 98.24%, and recall for low-frequency attacks decreased to 93.71%, indicating that the compactness of the embedding space primarily affects the stability of sparse categories.Threshold sensitivity results show that as $\tau_2$increases from a lower range to 0.85, the FPR can be further reduced from 0.43% to 0.29%, but the average response latency

increases from 1.8 s to 2.9 s;when τ_2= 0.78, the FPR is 0.36% and the average response latency is 2.1 s, achieving a more balanced trade-off between false positive control and response timeliness. Therefore, Figure 6 demonstrates that the performance improvement of our method stems from the synergistic interaction of multiple modules, rather than from a single threshold tuning.

Sources of error are primarily concentrated in two types of windows. The first type involves encrypted burst traffic caused by device updates, video transmission, or large file synchronization. In short-term statistics, this traffic resembles the precursors of scanning or denial-of-service attacks, making it prone to inflating risk scores.The second category consists of application-layer abuse and low-frequency reconnaissance, which exhibit limited fluctuations within a single window and only reveal anomalous trends over longer time scales. The former suggests that the system needs to integrate whitelists with update schedules, while the latter highlights that relationship graphs and device context are indispensable in home security.

3.3 Embedded Deployment Efficiency and Case Studies

If a solution achieves high accuracy only in offline testing, its value to smart home network security remains limited. This section further evaluates the actual operational efficiency of EAI-SHGuard on embedded platforms and analyzes its coordinated response performance through typical home attack scenarios. To compare the overall cost of different models on embedded platforms, see Figure 7.

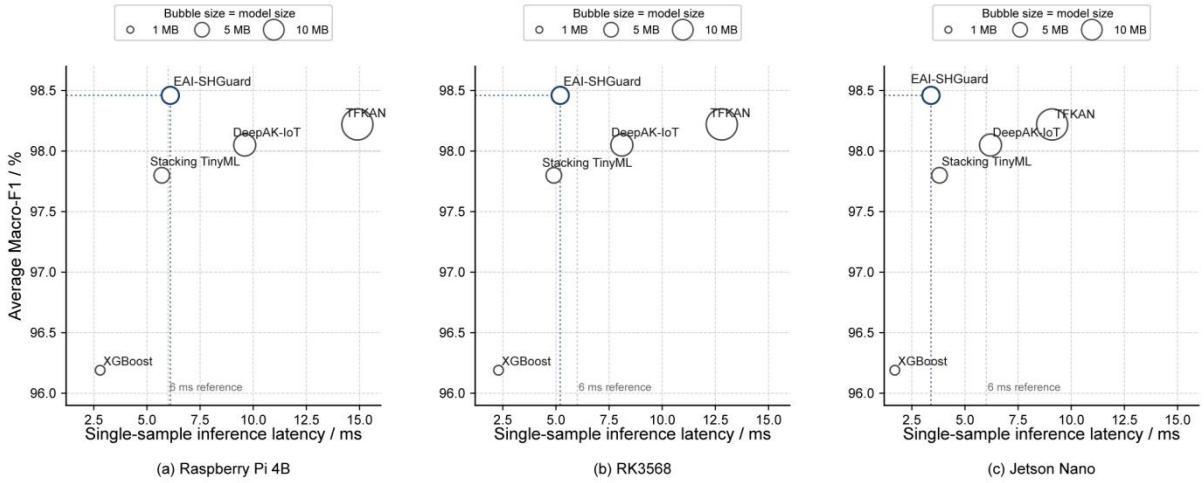

Figure 7: Coupled plot of accuracy, latency, and model size.

Figure 7 presents the accuracy-latency-model size coupling relationships for different models on the Raspberry Pi 4B, RK3568, and Jetson Nano in a three-quadrant plot, where the horizontal axis represents inference latency, the vertical axis represents the average Macro-F1 score, and the bubble area indicates model size.The EAI-SHGuard model has a size of 3.6 MB, with single-sample inference latencies of 6.1 ms, 5.2 ms, and 3.4 ms on the Raspberry Pi 4B, RK3568, and Jetson Nano, respectively, while maintaining an average Macro-F1 score of over 98%.In contrast, while TFKAN maintains high accuracy, its model size reaches 11.6 MB, and the latency on the Raspberry Pi 4B rises to 14.9 ms, shifting significantly toward the "large size-high latency" region; XGBoost and 1D-CNN are located further to the left, indicating lower latency, but their height on the vertical axis is insufficient, and their average Macro-F1 remains lower than that of the proposed method.Looking further at platform differences, the

latency of all models on the Jetson Nano shifts left overall, but the size differences remain; the relative positions on the Raspberry Pi 4B and RK3568 better reflect the actual deployment constraints of home gateways.Figure 7 thus demonstrates that the advantage of EAI-SHGuard lies not in achieving extreme optimization for a single metric, but in maintaining a relatively stable balance of accuracy, latency, and memory footprint across all three platform types. To illustrate the solution's dynamic response effectiveness during real-world home attack scenarios, see Figure 8.

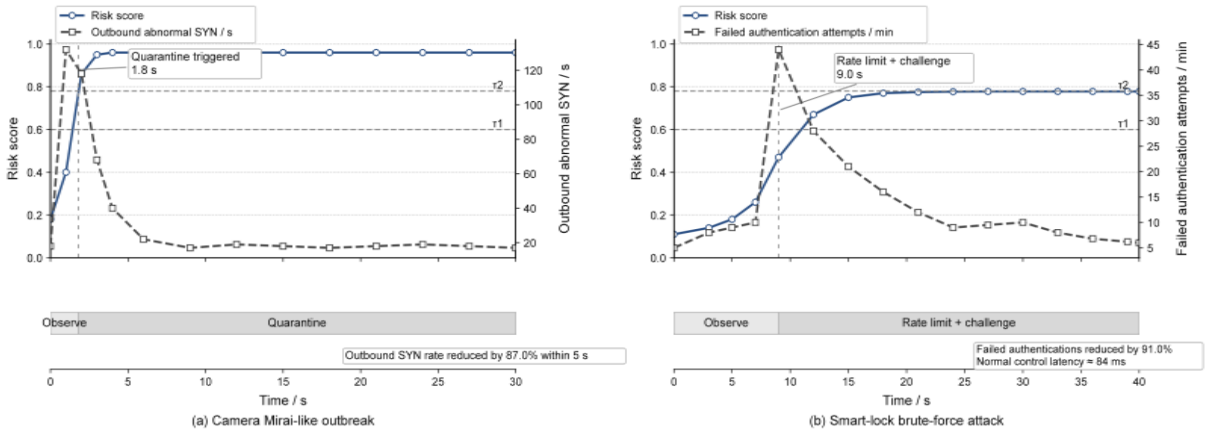

Figure 8: Timeline of risk evolution and coordinated response in typical attack scenarios.

Figure 8 has demonstrated the risk development and local cooperative reaction course in two typical household attack situations. In the Mirai-pattern outer connection situation which includes a camera, the risk value quickly climbed from an initial 0.14 to 0.93, passed the τ2 threshold at 1.8 seconds and caused an isolation action to be taken; Within 5 seconds after the isolation is completed, the quantity of abnormal outgoing SYN data packets has dropped by 87.0%, hence this shows that the quick blocking method is appropriate for this kind of high-strength sudden attack. In the scene of violent cracking of smart door lock password, the risk score ascends more slowly, it mainly stays between τ1 and τ2, the system gives first priority to rate-limiting and challenge tactics instead of isolating the device right away; In the course of this procedure, abnormal verification demands have a reduction of 91.0%, therefore the average extra time delay for ordinary control connections is merely 84 ms. Together, these results demonstrate that the proposed solution does not equate anomaly detection with immediate disconnection, but rather outputs differentiated response paths based on risk severity, device type, and behavioral phase. This advances home gateway protection from merely "being able to identify" to "being able to respond without excessive intervention."

From a deployment perspective, EAI-SHGuard is better suited to serve as the baseline security layer for home gateways rather than replacing all cloud-based or upper-layer security services. Its advantages lie in its ability to perform rapid local assessments and immediate blocking, thereby reducing the transmission of private data and minimizing cloud-side decision-making delays; its limitations, however, include limited capabilities regarding deep protocol semantics, cross-home collaborative threat intelligence, and long-term zero-day attack modeling.Therefore, in practical smart home applications, a more reasonable deployment approach is to use EAI-SHGuard as a local rapid defense line, combined with a home cloud control platform or operator-side threat intelligence update mechanisms, to form a layered protection structure.

4 Conclusion

This paper focuses on the smart home gateway scenario and proposes a cybersecurity protection solution based on embedded artificial intelligence, named EAI-SHGuard. The overall design is structured across three layers: multi-source data organization, lightweight detection modeling, and local coordinated response.

(1) This paper unifies traffic statistics, device profiles, and state anomalies into window-level detection objects, establishing a multi-data-source sample organization method tailored for home networks. This enables the relationship between "device type-communication behavior-risk changes" in smart homes to be integrated into a unified detection framework.

(2) This treatise establishes a two-branch light-weight model and unites it with context-relying gating, prototype restrictions, and quantization-conscious training. Therefore, the solution obtains an average Macro-F1 score of 98.46% over three public datasets, while it maintains the model volume at 3.6 MB, and reduces the single-sample inference delay on a Raspberry Pi 4B to 6.1 ms, hence it proves the feasibility that this solution can be deployed at the home edge.

(3) This paper still has two limitations: first, there is room for improvement in the ability to distinguish between deep-level encrypted business semantics and misuse at the application layer with limited samples; second, the current coordination strategy is primarily based on observable information at the gateway side and has not yet introduced a long-term cross-household collaborative update mechanism. Future work could further focus on new home protocols such as Matter, continuous learning and updates, and more granular policy orchestration.

About the Author

Jia Hong Zhou holds a Ph.D. She is an associate professor at the School of Information Engineering, Liaodong University, Dandong, Liaoning, China. Her primary research interests include the design and development of artificial intelligence embedded systems, Internet of Things security technology engineering, and risk control and management.

References

[1] Ayavaca-Vallejo, L., & Avila-Pesantez, D. F. (2023). Smart home IoT cybersecurity survey: A systematic mapping. In 2023 Conference on Information Communications Technology and Society (ICTAS) (1-6).

[2] Vardakis, G., Hatzivasilis, G., Koutsaki, E., et al. (2024). Review of smart-home security using the Internet of Things. Electronics, 13(16), 3343.

[3] Magara, T., & Zhou, Y. (2024). Internet of Things (IoT) in smart homes: Privacy and security. Journal of Electrical and Computer Engineering, 2024, 7716956.

[4] Peterson, E., & Mujeye, S. (2025). Addressing IoT vulnerabilities in smart homes. In Proceedings of the 2025 8th International Conference on Software Engineering and Information Management (116-121).

[5] Tokarz, K., Mączka, P., Mrozek, D., et al. (2025). Security audit of Internet of Things home automation systems. Procedia Computer Science, 270, 955-964.

[6] Alex, C., & Creado, S. (2023). A comprehensive survey of IoT security datasets: taxonomy, classification, and machine learning mechanisms. Computers & Security, 132, 103283.

[7] Neto, E. C. P., Dadalto, L. V., Viegas, E., et al. (2023). CICIoT2023: A real-time dataset and benchmark for large-scale attacks in IoT environments. Sensors, 23(13), 5941.

[8] Meidan, Y., Bohadana, M., Mathov, Y., et al. (2018). N-BaIoT-Network-based detection of IoT botnet attacks using deep autoencoders. IEEE Pervasive Computing, 17(3), 12-22.

[9] Alsaedi, A., Moustafa, N., Tari, Z., et al. (2020). TON_IoT telemetry dataset: A new generation dataset of IoT and IIoT for data-driven intrusion detection systems. IEEE Access, 8, 165130-165150.

[10] Fusco, P., Rimoli, G. P., & Ficco, M. (2024). TinyIDS: An IoT intrusion detection system using Tiny Machine Learning. In Computational Science and Its Applications - ICCSA 2024 Workshops (71-82).

[11] Sharma, A., Rani, S., & Shabaz, M. (2025). An optimized stacking-based TinyML model for attack detection in IoT networks. PLoS One, 20(8), e0329227.

[12] Misrak, S. F., & Melaku, H. M. (2025). Lightweight intrusion detection system for IoT with improved feature engineering and advanced dynamic quantization. Discover Internet of Things, 5(1), 97.

[13] Olanrewaju-George, B., & Pranggono, B. (2025). Federated learning-based intrusion detection system for the Internet of Things using unsupervised and supervised deep learning models. Cyber Security and Applications, 3, 100068.

[14] Albanbay, N., Tursynbek, Y., Graffi, K., et al. (2025). Federated learning-based intrusion detection in IoT networks: Performance evaluation and data scaling study. Journal of Sensor and Actuator Networks, 14(4), 78.

[15] Javeed, D., Saeed, M. S., Adil, M., et al. (2024). A federated learning-based zero-trust intrusion detection system for the Internet of Things. Ad Hoc Networks, 162, 103540.

[16] Ahakonye, L. A. C., Nwakanma, C. I., Lee, J. M., et al. (2024). Machine learning explainability for intrusion detection in the Industrial Internet of Things. IEEE Internet of Things Magazine, 7(3), 68-74.

[17] Alabbadi, A., & Bajaber, F. (2025). An intrusion detection system over IoT data streams using explainable artificial intelligence (XAI). Sensors, 25(3), 847.

[18] Binhulayyil, S., Li, S., & Saxena, N. (2025). Explainable AI-based intrusion detection in IoT systems. Internet of Things, 31, 101589.

[19] Ferrag, M. A., Friha, O., Hamouda, D., et al. (2022). Edge-IIoTset: A new comprehensive realistic cybersecurity dataset of IoT and IIoT applications for centralized and federated learning. IEEE Access, 10, 40281-40306.

[20] Garcia, S., Parmisano, A., & Erquiaga, M. J. (2020). IoT-23: A labeled dataset with

malicious and benign IoT network traffic (Version 1.0.0) [Data set]. Zenodo.

[21] Momand, A., Jan, S. U., & Ramzan, N. (2024). ABCNN-IDS: Attention-based convolutional neural network for intrusion detection in IoT networks. Wireless Personal Communications, 136, 1981-2003.

[22] Ding, W., Abdel-Basset, M., & Mohamed, R. (2023). DeepAK-IoT: An effective deep learning model for cyberattack detection in IoT networks. Information Sciences, 634, 157-171.

[23] Ahmad, B., Wu, Z., Huang, Y., et al. (2024). Enhancing the security in IoT and IIoT networks: An intrusion detection scheme leveraging deep transfer learning. Knowledge-Based Systems, 305, 112614.

[24] Fares, I. A., Abd Elaziz, M., Aseeri, A. O., et al. (2025). TFKAN: Transformer based on Kolmogorov-Arnold Networks for intrusion detection in IoT environments. Egyptian Informatics Journal, 30, 100666.

[25] Yin, Z., Chen, H., Ma, H., et al. (2025). CAEAID: An incremental contrast learning-based intrusion detection framework for IoT networks. Computer Networks, 262, 111161.