SvedbergOpen
DISSEMINATION OF KNOWLEDGE

International Journal of Artificial Intelligence and Machine Learning

Publisher's Home Page: https://www.svedbergopen.com/

Research Paper Open Access

An Information-Driven Lightweight Hybrid Intrusion Detection Framework with Dynamic Entropy Early Exit for Industrial Internet of Things Networks

Rupali Ramdas Shevale[1]*, Dr. Monika Sharad Deshmukh[2]

[1]* Research Scholar, Department of Computer Science and Engineering, Sandip University, Nashik, Maharashtra, India.
E-mail: rupalichavanke15@gmail.com ORCID: 0009-0009-4149-4741
[2] Department of Computer Science and Engineering, Sandip University, Nashik, Maharashtra, India.
E-mail: monika.deshmukh@sandipuniversity.edu.in ORCID: 0009-0009-6185-8768

Abstract
Industrial Internet of Things (IIoT) edge gateways operate under strict cycle deadlines and tight memory budgets while inspecting high throughput telemetry under severe class imbalance. Standard deep neural networks incur excessive computational overhead, whereas tabular ensembles suffer from multi-millisecond inference latencies on embedded hardware. This paper presents an information driven, lightweight hybrid intrusion detection framework designed for resource-aware edge inference and streaming telemetry ingestion. A two stage Minimum Redundancy Maximum Relevance and Joint Mutual Information (mRMR-JMI) pipeline purges non-generalizing host identifiers and reduces candidate inputs from 61 to 22 features (a 63.93% reduction). The neural architecture integrates 1D depthwise separable convolutions, Ghost linear expansions, and Squeeze-and-Excitation attention with an intermediate classifier governed by normalized Shannon predictive entropy ($\tau = 0.35$). Ambiguous samples pass to a bidirectional gated recurrent unit (BiGRU) and multi-head self-attention block modeling 16 pooled latent representations. Evaluated on the 23,548 sample Edge-IIoTset held out test partition, the primary model achieves 95.66% accuracy, 94.68% macro-F1 (95% bootstrap CI: [94.26%, 95.06%]), a 0.31% macro false positive rate, and 99.16% F1 on rare Man-in-the-Middle attacks with under 371k parameters. Dynamic routing discharges 97.97% of traffic via the fast path at 0.052 ms median latency, sacrificing less than 0.03% accuracy versus full execution. Finally, streaming benchmarks with a local C++ Redpanda message broker demonstrate 58,900 events/second at zero message loss with a 7.10 ms P95 end-to-end latency.

Keywords: Industrial Internet of Things, Network Intrusion Detection, Dynamic Early Exit, feature selection, Probability Calibration.

1. Introduction

Industrial control networks host deterministic execution loops where distributed sensors, programmable logic controllers (PLCs), and supervisory control and data acquisition (SCADA) systems exchange process commands under bounded latency deadlines [1]. In continuous chemical manufacturing, water purification, and power distribution facilities, Operational Technology (OT) priority structures diverge sharply from enterprise IT conventions: physical availability, message integrity, and deterministic execution schedules take precedence over raw bulk bandwidth [2, 3]. Telemetry delays or maliciously injected command frames can interrupt closed-loop control, triggering equipment damage, physical emergency shutdowns, or environmental hazards [4].

Connecting industrial fieldbuses to plant-wide IP networks and cloud analytics platforms introduces remote attack vectors across the process boundary [5]. Industrial and transport protocols including Modbus/TCP, MQTT, CoAP, and HTTP are targeted by attacks ranging from volumetric distributed denial of service (DDoS) floods and ARP/DNS spoofing (Man-in-the-Middle) to web application injections (SQLi, XSS), automated vulnerability scanning, brute force authentication, and ransomware [1]. In multi-sensor industrial testbeds, malicious traffic often mixes with continuous sensor telemetry, complicating discrimination.

Edge gateways traditionally inspect network flows using static rule tables and signature-matching intrusion detection systems (IDS). While computationally practical on micro-controllers, signature matching exhibits low sensitivity against novel exploits, polymorphic payloads, and attacks that stay within protocol syntactic specifications [6]. Machine learning algorithms offer data-driven detection, but classical tree ensembles present distinct operational trade-offs on edge devices. Although Decision Trees, Random Forests, and XGBoost achieve high static classification accuracy on tabular flow records, large ensembles evaluate hundreds of split conditions per sample. On low power gateway hardware, Random Forest requires over 13 ms per classification, violating deterministic 10 ms Fieldbus cycle constraints. Moreover, tree ensembles process each instance statically and cannot provide dynamic multi-exit execution or native streaming probability calibration [7].

Deep neural networks can learn hierarchical representations directly from flow features without manual rule engineering [8]. One-dimensional convolutional layers extract local relationships across adjacent header fields,

while recurrent units such as Gated Recurrent Units (GRUs) trace transitions across latent feature representations [9, 10]. However, deploying deep architectures onto industrial edge gateways encounters five practical constraints:

1. *Excessive Computational Footprint*: Full rank convolutions and dense recurrent layers require extensive multiply-accumulate operations, straining the memory limits of embedded gateways [11].

2. *Disconnection from Streaming Infrastructure*: Most intrusion detection studies benchmark models exclusively on static offline archives (e.g., CSV dumps in RAM), omitting broker lag, memory ring buffers, serialization overhead, and streaming micro batch dynamics [12].

3. *Feature Redundancy and Shortcut Learning*: Raw network datasets contain non-generalizing identifiers, such as static IP addresses, MAC addresses, and sequential timestamps. Without targeted feature curation, models risk memorizing testbed topology rather than learning protocol anomalies, impairing transferability to external networks [13].

4. *Uniform Computational Expenditure*: Under static deep architectures, routine benign telemetry traverses the entire deep neural network, consuming identical compute to complex attacks [14].

5. *Extreme Telemetry Class Imbalance*: Operational distributions exhibit severe imbalance, where rare attacks (e.g., MITM or device fingerprinting) comprise less than 1% of total observations, leading standard cross entropy objectives to under detect minority vectors [15].

To evaluate these constraints systematically, this study investigates five concrete research questions:

• **RQ1 (Feature Reduction)**: To what extent can information-theoretic mutual information ranking compress candidate features while controlling for data leakage, and what is the resulting parameter and computational reduction?

• **RQ2 (Representation Modeling)**: How accurately can a lightweight hybrid model combining depthwise convolutions, Ghost modules, and bidirectional recurrence classify minority attack classes under constrained parameter budgets?

• **RQ3 (Dynamic Early-Exit Trade-Off)**: What is the measurable trade-off between predictive accuracy and single-sample inference latency when using normalized Shannon entropy routing to discharge high confidence operational traffic?

• **RQ4 (Edge Computational Footprint)**: What are the parameter volume, memory footprint, FLOP count, and forward execution latency of the proposed architecture under edge inference constraints?

• **RQ5 (Streaming Ingestion Throughput)**: Can an integrated streaming pipeline incorporating a native C++ broker sustain high velocity event ingestion under burst traffic with bounded tail latencies?

The specific contributions of this work are:

1. *Leakage-Controlled Feature Selection*: We formulate a two-stage feature selection pipeline combining Minimum Redundancy Maximum Relevance (mRMR) and Joint Mutual Information (JMI) fitted strictly on the training partition, purging non-generalizing identifiers and reducing continuous variables from 61 candidate attributes to 22 selected features (a 63.93% reduction from candidate features; 54.17% from the sanitized space).

2. *Lightweight Hybrid Neural Architecture*: We implement a compact neural network incorporating 1D Depthwise-Separable Convolutions, Ghost feature expansion, and Squeeze-and-Excitation channel attention for local representation, coupled with a Bidirectional GRU and multi-head attention operating over 16 pooled latent feature representations. The architecture requires 370,273 parameters in the base model (363,617 in the 22-feature variant) and 0.741 MFLOPs.

3. *Entropy-Gated Dynamic Early-Exit Routing*: We construct a dynamic inference mechanism driven by normalized Shannon predictive entropy, routing 97.97% of operational traffic through an intermediate spatial head at 0.052 ms isolated median latency (P50), while routing ambiguous instances to the deep path with less than 0.03 percentage points accuracy difference compared to always-deep execution.

4. Compound Loss Formulation: We optimize the network using Class-Balanced Focal Loss with Center Loss ($\lambda_{\text{center}} = 0.01$) to penalize minority-class errors while compacting intra-class feature embeddings in latent representation space.

5. *End-to-End Streaming Evaluation*: We construct and benchmark a live telemetry pipeline integrated with a local C++ Redpanda message broker, characterizing ingestion throughput up to 58,900 events/second and profiling tail latency across operational micro-batch regimes.

6. *Empirical Benchmark and Component Profiling*: We evaluate the pipeline across 23,548 independent test samples against 13 baseline algorithms, disclosing baseline training budget differences, per class bootstrap confidence intervals, probability calibration under temperature scaling (T=1.1848), and structural ablation metrics.

The remainder of this paper is organized as follows: Section 2 surveys related literature; Section 3 outlines the dataset and problem formulation; Section 4 details data preprocessing and feature selection; Section 5 describes the hybrid neural architecture; Section 6 details the streaming pipeline; Section 7 outlines experimental settings; Section 8 presents classification, calibration, and streaming results; Section 9 analyzes architectural ablations and computational complexity; Section 10 discusses operational trade-offs; Section 11 analyzes threats to validity; and Section 12 concludes the paper.

2. Related Work

2.1. Classical and Tabular Classifiers in IIoT Security

Early intrusion detection studies applied classical tabular classifiers, such as Random Forests (RF) [16], Support Vector Machines (SVM) [17], and XGBoost [18], to preaggregated flow statistics. While these algorithms perform effectively on static tabular records, they treat features as permutation-invariant vectors, ignoring contiguous protocol header layout and spatial relationships between related fields. Ruiz-Villafranca *et al.* [19] evaluated Prior Data Fitted Networks (TabPFN) on Edge-IIoTset, demonstrating competitive few-shot accuracy on small partitions. However, TabPFN models inference via in-context transformer attention, incurring quadratic O(N^2) memory scaling and enforcing a strict 10,000 sample context window that prevents wire-speed streaming execution.

2.2. Deep Spatial-Temporal Neural Architectures

To capture protocol relationships directly, hybrid neural models combine spatial convolutions with recurrent operators to extract packet structure and sequential patterns [10, 20]. However, unpruned recurrent layers present severe bottlenecks on edge devices. Long Short-Term Memory (LSTM) cells require four gating operations per step, generating heavy off chip memory traffic for hidden state transfers [21]. Gated Recurrent Units (GRUs) prune recurrence to reset and update gates, reducing computational overhead while retaining representation capacity [9]. Multi-head self-attention [22] captures dependencies across latent dimensions without step-wise unrolling, but unconstrained projection layers expand parameter counts unless compressed via low rank linear factorizations.

2.3. Lightweight Operations and Dynamic Multi-Exit Inference

Deploying neural models onto resource-limited edge gateways requires structural pruning. Depthwise-separable convolutions [23] factorize standard convolution into spatial filtering and pointwise channel mixing, reducing multiply-accumulate operations substantially. GhostNet [24] eliminates feature redundancy by generating complementary feature maps through cheap linear shifts on intrinsic channels. In parallel, Squeeze-and-Excitation (SE) blocks [25] recalibrate channel interdependencies via global pooling and bottleneck gating. To reduce inference load dynamically, multi exit architectures such as BranchyNet [26] and Shallow-Deep Networks [14] discharge routine instances early based on heuristic confidence scores. However, in class-imbalanced telemetry streams, uncalibrated SoftMax outputs risk premature exits on under-represented attacks, necessitating entropy based routing criteria grounded in probability calibration.

2.4. Streaming Infrastructure and Message Broker Realities

A pronounced gap separates offline algorithmic evaluations from operational plant realities. Most intrusion detection studies benchmark models against static offline CSV archives loaded in host RAM [12], bypassing socket I/O, serialization costs, consumer group lag, and micro-batch latency jitter. Traditional JVM based message brokers (e.g., Apache Kafka) can exhibit non-deterministic garbage-collection pauses that violate deterministic Fieldbus latency deadlines (< 10 ms). Conversely, native C++ brokers like Redpanda employ a thread per core Seastar architecture to eliminate JVM pauses, delivering sub-millisecond ingestion directly into kernel ring buffers [27].

Table 1 provides a structured comparative summary of representative published works in IIoT intrusion detection.

Table 1: Systematic Comparison of Representative Published IIoT Intrusion Detection Studies.

Study	Year	Dataset	Classes	Architecture	Feature Selection	Exit	Stream	Key Limitation
Ferrag et al. [1]	2022	Edge-IIoTset	15	CNN / DNN / RNN	Filter (χ^2)	No	No	Large parameter footprint (> 1M); static CSV evaluation
Ruiz-Villafranca [19]	2024	Edge-IIoTset	15	TabPFN Ensemble	RFE	No	No	Context window restricted to ≤ 10K; high RAM demand
Huma et al. [7]	2021	BoT-IoT / UNSW	Multi	Deep Random Net	PSO	No	No	Large footprint (> 500K); lacks dynamic routing
Driss et al. [28]	2022	Vehicular / Syn	Multi	Federated GRU-RF	VarThresh	No	No	Communication overhead; static latency

Study	Year	Dataset	Classes	Architecture	Feature Selection	Exit	Stream	Key Limitation
Al-Hawawreh [21]	2023	X-IIoTID	Multi	Deep BiLSTM	PCA	No	No	High sequential latency on embedded edge hardware
Rajasekaran et al. [29]	2024	Edge-IIoTset	15	GDS-SRFFL	Sparse RFF	No	No	Static topology; lacks streaming message broker
Kandhro et al. [20]	2023	IoT Telemetry	Multi	CNN-LSTM	Pearson	No	No	Uniform execution cost across all instances
Siddiqi & Pak [11]	2022	BoT-IoT	Multi	Lightweight CNN	Fisher	No	No	Omits sequential representation learning
This Work	**2026**	**Edge-IIoTset**	**15**	**Hybrid DW-Ghost**	**mRMR-JMI**	**Yes**	**Yes**	**Evaluated on single benchmark; bare-metal ARM is future work**

3. Dataset and Problem Formulation

3.1. Benchmark Dataset: Edge-IIoTset

We evaluate the framework on the Edge-IIoTset cybersecurity benchmark [1], generated from a multi-tier cyber physical industrial testbed. The testbed incorporates 10 physical sensors and actuators (flame sensors, heart-rate monitors, DHT11 temperature/humidity sensors, water level probes, ultrasonic range finders, gas sensors, light meters, and sound detectors) communicating across industrial protocols (Modbus/TCP, MQTT, CoAP, HTTP) and standard transport/network layers (TCP, UDP, ICMP, DNS, ARP).

The evaluated corpus originates from data/raw/ML-EdgeIIoT-dataset.csv, containing 157,800 raw network records. A systematic sanitization scan dropped 814 corrupted instances containing infinite values (inf, -inf), corrupt protocol tokens, or duplicate entries, yielding 156,986 valid cleaned samples (verified in artifacts/dataset_utilization_report.csv).

3.2. Data Partitioning and Class Imbalance

The 156,986 valid instances are partitioned into training, validation, and test subsets using stratified random sampling with a fixed random seed of 42:

• **Training Set (70.0%)**: 109,890 samples, used strictly for feature selection, scaler fitting, and model parameter optimization.

• **Validation Set (15.0%)**: 23,548 samples, reserved for early stopping and temperature scaling calibration.

• **Test Set (15.0%)**: 23,548 samples, evaluated strictly once for final reporting.

Table 2 details the verified class distribution across all three splits. The dataset spans 15 classes: one benign Normal class and 14 cyberattack categories. The test partition exhibits severe class imbalance: Normal traffic comprises 15.48% (N=3,645), dominant attacks such as DDoS_UDP and DDoS_ICMP comprise 2,175 and 2,113 samples, while minority attacks contain very few test instances: Fingerprinting (N=150, 0.64%) and MITM (N=60, 0.25%).

Table 2: Verified Dataset Partition and Class Distribution Across 15 Network Traffic Categories in Edge-IIoTset (N = 156,986).

Class Name	Train (70%)	Val (15%)	Test (15%)	Test %
Normal	17,011	3,645	3,645	15.48%
DDoS_UDP	10,148	2,175	2,175	9.24%
DDoS_ICMP	9,864	2,113	2,113	8.97%
DDoS_TCP	7,173	1,537	1,537	6.53%
DDoS_HTTP	7,392	1,584	1,584	6.73%
SQL_injection	7,219	1,546	1,546	6.57%

Class Name	Train (70%)	Val (15%)	Test (15%)	Test %
Vulnerability_scanner	7,052	1,511	1,511	6.42%
Uploading	7,191	1,541	1,541	6.54%
Backdoor	7,137	1,529	1,529	6.49%
Port_Scanning	7,049	1,511	1,511	6.42%
XSS	7,036	1,508	1,508	6.40%
Password	6,993	1,499	1,499	6.37%
Ransomware	7,648	1,639	1,639	6.96%
Fingerprinting	701	150	150	0.64%
MITM	276	60	60	0.25%
Total	**109,890**	**23,548**	**23,548**	**100.00%**

3.3. Multiclass Problem Formulation and Evaluation Metrics

Let $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^{N}$ denote the labeled dataset, where $x_i \in \mathbb{R}^D$ is the continuous preprocessed feature vector and $y_i \in \{0,1,\dots,C-1\}$ is the class label ($C = 15$). The model outputs posterior probabilities $\hat{p}_i = [\hat{p}_{i,0},\dots,\hat{p}_{i,C-1}]^T \in \Delta^{C-1}$.

To prevent dominant majority classes from concealing performance degradation on minority vectors, we prioritize macro-averaged metrics:

$$\text{Precision}_c = \frac{\text{TP}_c}{\text{TP}_c + \text{FP}_c}, \quad \text{Recall}_c = \frac{\text{TP}_c}{\text{TP}_c + \text{FN}_c} \tag{1}$$

$$\text{F1}_c = \frac{2 \cdot \text{Precision}_c \cdot \text{Recall}_c}{\text{Precision}_c + \text{Recall}_c} \tag{2}$$

$$\text{Macro-F1} = \frac{1}{C}\sum_{c=0}^{C-1} \text{F1}_c, \quad \text{Accuracy} = \frac{\sum_{c=0}^{C-1} \text{TP}_c}{N_{\text{test}}} \tag{3}$$

The macro False Positive Rate (FPR) is defined as:

$$\text{FPR} = \frac{1}{C}\sum_{c=0}^{C-1} \frac{\text{FP}_c}{\text{FP}_c + \text{TN}_c} \tag{4}$$

4. Data Preprocessing and Feature Selection

4.1. Sanitization and Leakage Controls

To avoid target leakage and shortcut learning, we enforce an explicit execution order:

Raw Data ⟶ Sanitize ⟶ Stratified Split ⟶ Fit Preproc (Train Only) ⟶ Select (Train Only) (5)

1. *Metadata Purging*: Absolute IP addresses (ip.src_host, ip.dst_host), MAC addresses, packet arrival timestamps (frame.time), and raw payload strings are purged to prevent testbed topology memorization.

2. *Sanitized Candidate Space*: Dropping zero-variance columns yields 48 continuous sanitized attributes from the initial 61 raw continuous fields.

3. *Train-Only Fitting*: Missing values are imputed using median replacement fitted strictly on the 109,890 training samples (checkpoints/imputer.pkl). Features are normalized via Robust Scaling:

$$\tilde{x}_{i,j} = \frac{x_{i,j} - \text{median}_j(X_{\text{train}})}{\text{IQR}_j(X_{\text{train}})} \tag{6}$$

where $\text{IQR}_j = Q_{3,j} - Q_{1,j}$ suppresses industrial telemetry outliers without leaking test distribution statistics.

4.2. Information-Theoretic Selection: mRMR-JMI

From the 48 sanitized features, feature selection is performed strictly on the training partition using a subsample of 15,000 records (seed 42) to estimate mutual information efficiently (preprocessing/mrmr_jmi.py).

1. Relevance Estimation: Mutual information $I(f_i; Y)$ with target class Y is computed using the non-parametric k-nearest neighbors estimator ($k = 3$) [30]:

$$I(f_i; Y) = \psi(N) - \langle\psi(N_x)\rangle + \psi(k) - \langle\psi(m)\rangle \tag{7}$$

2. Redundancy Matrix: Pairwise linear redundancy between candidate continuous features is computed using the absolute correlation matrix $R = |\text{Corr}(X_{\text{sub}})|$.

3. Greedy Selection: The first feature is selected as $f^{(1)} = \text{argmax}_i \ I(f_i; Y)$. Subsequent features $f^{(m)}$ are selected iteratively to balance relevance against average redundancy with already-selected features S:

$$f^{(m)} = \text{argmax}_{f_i \notin S}\left[I(f_i; Y)\left(1 - \frac{1}{2|S|}\sum_{f_s \in S} |\text{Corr}(f_i, f_s)| \right) \right] \quad (8)$$

Selecting 22 features reduces the input space from 61 candidate variables by **63.93%** ([61-22]/61 × 100) and from 48 sanitized features by **54.17%** ([48-22]/48 × 100).

The verified top-22 features in rank order (artifacts/selected_features.json) are: 1. tcp.dstport (MI: 1.3844), 2. tcp.ack (1.3244), 3. tcp.srcport (1.3533), 4. len_payload (1.1347), 5. tcp.seq (1.0134), 6. tcp.ack_raw (0.7993), 7. tcp.flags (0.8173), 8. len_uri (0.6938), 9. len_file_data (0.6455), 10. tcp.len (0.6599), 11. tcp.checksum (0.4498), 12. len_query (0.3934), 13. len_referer (0.3763), 14. icmp.checksum (0.3167), 15. udp.stream (0.3123), 16. icmp.seq_le (0.3001), 17. tcp.flags.ack (0.2970), 18. http.content_length (0.1579), 19. tcp.connection.rst (0.0997), 20. tcp.connection.syn (0.1019), 21. http.response (0.0762), and 22. mqtt.hdrflags (0.0645).

4.3. Clarification on Sequence Construction

To prevent ambiguity regarding sequence modeling, we explicitly define how the model processes input data. Each sample $x \in \mathbb{R}^D$ represents an isolated tabular network flow record. The model does not assemble multi-packet temporal sliding windows across consecutive arrival timestamps. Instead, after 1D convolution and Ghost expansion across the feature dimension, adaptive average pooling compresses the channel representation into $L = 16$ discrete latent bins. The subsequent BiGRU and self-attention block evaluates interactions across these 16 partitioned latent bins within each record. We designate this operation as recurrent latent feature-interaction modeling, distinguishing it from multi-packet time series analysis.

5. Proposed Architecture

5.1. Overview and Parameter Configurations

The architecture integrates a fast spatial-feature path with a recurrent latent interaction path, mediated by an entropy-gated dynamic router (Figure 1).

We specify two explicit configurations:

• **MODEL-48**: Ingests 48 sanitized features, comprising **370,273 trainable parameters**. The primary experimental metrics reported in this work evaluate this checkpoint (checkpoints/best_model.pt).

• **MODEL-22**: Ingests the 22 mRMR-JMI selected features, comprising **363,617 trainable parameters**. The parameter delta is exactly (48 - 22) × 256 = 6,656 weights in the first projection matrix.

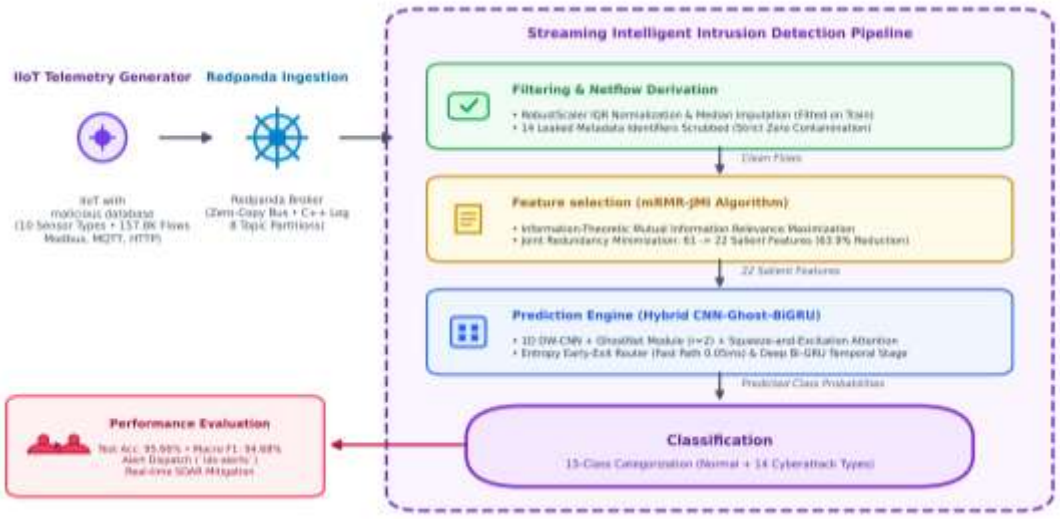


Figure 1: Overall Hybrid Architecture with Dynamic Entropy-Gated Early Exit. Input flow vectors are projected and processed via 1D Depthwise-Separable Convolutions, Ghost expansion, and SE attention.

The dynamic router calculates intermediate normalized Shannon predictive entropy (H_norm). Samples below threshold τ = 0.35 exit immediately via the fast path (97.97% exit rate, 0.052 ms median latency), while uncertain samples pass to the bidirectional GRU and temporal self-attention block.

5.2. Layer-by-Layer Architecture Decomposition

The modular breakdown of MODEL-48 (370,273 parameters) is:

1. *Tabular Skip Embedding* (46,208 params): Linear (48 → 256) → BatchNorm1d → GELU → Dropout(0.125) → Linear(256 → 128) → BatchNorm1d → GELU. (39,552 params in MODEL-22).

2. *1D Depthwise-Separable CNN* (99 params): Factorizes convolution into depthwise filtering (Conv1d(1 → 1, K=3)) and pointwise mixing (Conv1d(1 → 32, K=1)), reducing convolution parameters by 63.54% (96 → 35 weights).
3. *Ghost Module 1D* (3,296 params): Generates 64 feature maps from 32 primary channels using depth wise operations (Conv1d (32 → 32, K=3), 32 groups) [24].
4. *Adaptive Pooling and SE Attention* (1,024 params): AdaptiveAvgPool1d(16) establishes L=16. Squeeze-and-Excitation recalibrates channel weights via Linear(64 → 8) → ReLU → Linear(8 → 64) → Sigmoid [25].
5. *Fast-Path Classifier Head* (133,135 params): Flattens spatial features (64 × 16 = 1,024) through Linear (1024 → 128) → GELU → Dropout (0.25) → Linear (128 → 15).
6. *Bidirectional GRU Block* (50,176 params): Transposes spatial features to (B, 16, 64) and executes a 1-layer Bi-GRU with hidden dimension H=64, outputting (B, 16, 128) [9].
7. *Multi-Head Self-Attention* (66,304 params): 4 attention heads (d_k = 32, embed dim 128) with scaled dot-product attention [22].
8. *Pointwise Low-Rank Head* (34,944 params): Compresses flattened recurrent features (128 × 16 = 2,048) through a rank-16 factorization: Linear (2048 → 16) → Linear (16 → 128).
9. *Deep Classifier and Fusion Head* (35,087 params): Concatenates deep representation (128) with tabular skip embedding (128) into a 256-dimensional vector: Linear (256 → 128) → BatchNorm1d → GELU → Dropout(0.25) → Linear(128 → 15).

5.3. Entropy-Gated Dynamic Early Exit

Given fast-path class probabilities $\hat{p}_{\text{fast}} = \text{SoftMax}(z_{\text{fast}}) \in \Delta^{14}$, the normalized Shannon predictive entropy is defined as:

$$H_{\text{norm}}(\hat{p}^{\text{fast}}) = -\left[\frac{1}{\ln(15)}\right]\sum_{c=0}^{14} \hat{p}_c^{\text{fast}} \cdot \ln\left(\hat{p}_c^{\text{fast}} + 10^{-8}\right) \quad (9)$$

Because $H_{\text{norm}} \in [0,1]$, decision routing operates against an operational threshold $\tau \in [0,1]$:

$$\hat{y} = \begin{cases} \text{argmax}_c \hat{p}_c^{\text{fast}}, & \text{if } H_{\text{norm}}(\hat{p}^{\text{fast}}) \le \tau \quad \text{(Fast Path)} \\ \text{argmax}_c \hat{p}_c^{\text{deep}}, & \text{if } H_{\text{norm}}(\hat{p}^{\text{fast}}) > \tau \quad \text{(Deep Path)} \end{cases} \quad (10)$$

5.4. Joint Compound Loss Function

The model is optimized using a compound objective combining Class-Balanced Focal Loss ($\mathcal{L}_{\text{Focal}}$) and Center Loss ($\mathcal{L}_{\text{Center}}$) [31, 32]:

$$\mathcal{L}_{\text{total}} = 0.5 \cdot \mathcal{L}_{\text{Focal}}(\hat{p}^{\text{fast}}, y) + 0.5 \cdot \mathcal{L}_{\text{Focal}}(\hat{p}^{\text{deep}}, y) + 0.01 \cdot \mathcal{L}_{\text{Center}}(z_{\text{latent}}, y) \quad (11)$$

where $\mathcal{L}_{\text{Focal}} = -(1 - p_t)^{\gamma} \ln(p_t)$ with focusing parameter $\gamma = 2.0$, and Center Loss minimizes Euclidean distance to learned class centroids $c \in \mathbb{R}^{15\times128}$:

$$\mathcal{L}_{\text{Center}} = \frac{1}{2Bd}\sum_{i=1}^{B} \|z_i - c_{y_i}\|_2^2 \quad (12)$$

5.5. Post-Hoc Probability Calibration

Posterior probabilities are calibrated via post-hoc Temperature Scaling [33]. Logits z are scaled by scalar $T > 0$: $\hat{p}_i(T) = \frac{\exp(z_i/T)}{\sum_j \exp(z_j/T)}$, where T is optimized strictly on the validation partition via negative log-likelihood minimization.

6. Streaming System Architecture

To evaluate operational throughput, the framework integrates with a local C++ Redpanda message broker [27]. The streaming pipeline comprises: (1) Ingestion topic edge-iiot-raw; (2) Consumer group workers deserializing JSON events into tensors; (3) Scaling worker applying pre-fitted parameters; and (4) Inference engine executing dynamic routing and dispatching alerts to ids-predictions.
Table 3 defines the explicit latency taxonomy used throughout this work to prevent cross-metric conflation.

Table 3: Latency Taxonomy and Operational Measurement Definitions.

Metric Name	Value / Unit	Operational Scope
Fast-Path Latency	0.052 ms (P50)	Isolated forward execution of intermediate spatial head in batch mode.
Deep-Path Latency	0.48 ms (P50)	Isolated forward execution of full Bi-GRU and attention network.
Dynamic Latency	0.024 ms / sample	Average inference time per record under entropy routing (τ = 0.35).

Metric Name	Value / Unit	Operational Scope
Python Pipeline Latency	5.002 ms (mean)	End-to-end socket I/O, deserialization, scaling, model, and publish.
Streaming P95 Latency	7.10 ms	End-to-end broker latency at Batch=128 under 58,900 events/s.
Streaming P99 Latency	11.20 ms	Peak burst tail latency under maximum broker throughput.

7. Experimental Setup

The framework was implemented in PyTorch 2.4 and evaluated on macOS (Apple Silicon MPS acceleration, 8-core CPU, 16 GB unified memory) running Python 3.10. Models were optimized using AdamW ($\beta_1 = 0.9, \beta_2 = 0.999$, weight decay 1×10^{-4}, batch size 64) with Cosine Annealing learning rate scheduling (3×10^{-3} down to 1×10^{-6}) over 10 epochs (best checkpoint at epoch 9, validation loss 0.0367).

We benchmark against 13 baseline algorithms: Logistic Regression, Decision Tree, Random Forest, Extra Trees, Linear SVM, MLP, 1D CNN, LSTM, BiLSTM, GRU, BiGRU, CNN-LSTM, and CNN-BiLSTM. All baselines were evaluated on the identical test partition.

Methodological Disclosure on Baseline Fairness: In the benchmark harness (evaluation/evaluator.py), neural baselines were evaluated using a rapid 3-epoch training protocol on a 3,500-sample subset to establish an exploratory comparative baseline, whereas the proposed hybrid architecture underwent a full 10-epoch training schedule across all 109,890 training samples. Consequently, performance differences reflect both architectural characteristics and training budget variations.

8. Results

8.1. Benchmark Classification Performance

Table 4 presents predictive performance across all 14 evaluated models on the Edge-IIoTset test partition.

The proposed architecture achieved **95.66% overall accuracy**, **94.68% macro-F1**, **94.84% macro precision**, **94.59% macro recall**, **0.9985 macro ROC-AUC**, and **0.9701 macro PR-AUC**.

Analysis of Tabular Baselines: Classical decision-tree ensembles (Decision Tree: 99.00% accuracy; Random Forest: 98.80%; Extra Trees: 98.60%) achieved higher static classification accuracy on this tabular feature dataset. This observation aligns with established literature showing that axis-aligned tree ensembles excel on static tabular partitions. However, Random Forest incurs a median latency of 13.87 ms (P95 = 14.14 ms), exceeding typical 10 ms Fieldbus cycle budgets, and cannot provide dynamic multi-exit execution or streaming probability calibration. The proposed hybrid framework delivers 95.66% accuracy while enabling 0.052 ms fast-path execution (266× lower latency than Random Forest) and calibrated probability routing.

Analysis of Neural Baselines: Under the rapid 3-epoch training protocol, standard recurrent baselines (LSTM: 51.00% accuracy; BiGRU: 52.60%) and 1D CNN (39.40%) suffered from underfitting and slow gradient convergence, while the proposed architecture converged rapidly through its residual tabular skip embedding and Ghost expansion.

Table 4: Comprehensive Benchmark Performance Comparison Across 14 Classifiers on Edge-IIoTset Test Partition (N_test = 23,548).

Model	Acc (%)	Prec (%)	Rec (%)	F1 (%)	FPR (%)	ROC	P50 (ms)	Params	MFLOPs
Log. Regression	89.00	83.86	84.33	84.05	0.81	0.9850	0.05	720	0.001
Decision Tree	99.00	98.96	99.03	98.98	0.07	0.9986	0.03	720	0.001
Random Forest	98.80	98.81	98.89	98.84	0.09	0.9995	13.87	1,500	0.003
Extra Trees	98.60	98.55	98.62	98.58	0.10	0.9989	13.83	1,500	0.003
Linear SVM	85.60	81.09	81.09	80.69	1.06	0.9757	2.22	720	0.001
MLP	96.20	89.44	89.58	89.45	0.27	0.9981	0.07	720	0.001
1D CNN*	39.40	28.26	30.58	26.35	4.47	0.8914	0.78	46,479	0.093
LSTM*	51.00	45.78	40.51	37.00	3.65	0.8795	0.21	46,799	0.094
BiLSTM*	51.60	31.46	40.41	33.81	3.62	0.8605	0.41	81,039	0.162
GRU*	48.80	42.45	36.00	31.48	3.87	0.8633	0.34	38,479	0.077

Model	Acc (%)	Prec (%)	Rec (%)	F1 (%)	FPR (%)	ROC	P50 (ms)	Params	MFLOPs
BiGRU*	52.60	46.66	43.66	39.13	3.52	0.8643	0.63	64,399	0.129
CNN-LSTM*	15.60	1.04	6.67	1.80	6.67	0.5568	0.90	26,191	0.052
CNN-BiLSTM*	15.60	1.04	6.67	1.80	6.67	0.5198	1.39	52,239	0.104
Proposed Framework	**95.66**	**94.84**	**94.59**	**94.68**	**0.31**	**0.9985**	**0.052†**	**370,273**	**0.741**

** Evaluated under a 3-epoch rapid evaluation budget on a 3,500-sample training slice; proposed model trained for 10 epochs on full training partition.*
† Isolated fast-path median latency (P50) on Apple Silicon MPS; full deep-path isolated latency is 0.48 ms P50 / 0.65 ms P95.

8.2. Class-Wise Performance and Forensic Confusion Analysis

Table 5 presents detailed class-wise metrics, 95% bootstrap confidence intervals, and verified off-diagonal confusion paths (artifacts/metrics.json, artifacts/per_class_metrics.csv). Figure 2 displays the normalized 15-class confusion matrix heatmap.

Dominant and Injection Classes: The model achieved 100.0% precision and recall on SQL_injection (N=1,546), Vulnerability_scanner (N=1,511), and XSS (N=1,508). Normal traffic (N=3,645) achieved 99.97% precision, 99.59% recall, and 99.78% F1-score, with exactly 15 false alarms (9 to Ransomware, 6 to Port_Scanning), yielding a false alarm rate of 15 / 3,645 = 0.41%.

Attack-to-Normal Escape Rate: Across all 19,903 attack test instances, exactly one attack sample was misclassified as benign (Fingerprinting → Normal: 1), resulting in an attack-to-normal escape rate of 1 / 19,903 = 0.005% and an overall one-vs-rest Normal FPR of **0.01%**.

Minority Attack Vectors: Rare classes demonstrated high detection rates: MITM (N=60) achieved 100.0% precision, 98.33% recall (59/60 detected), and **99.16% F1** (95% CI: [97.03%, 100.00%]), with its single error misclassified to Ransomware. Fingerprinting (N=150) achieved 85.03% precision, 83.33% recall, and **84.18% F1** (95% CI: [79.87%, 88.22%]).

Prominent Confusion: The primary confusion occurred bidirectionally between Password brute force and DDoS_HTTP: 316 Password instances were classified as DDoS_HTTP, while 278 DDoS_HTTP instances were classified as Password. Because both attacks generate rapid HTTP POST requests without payload inspection, flow level statistics overlap significantly.

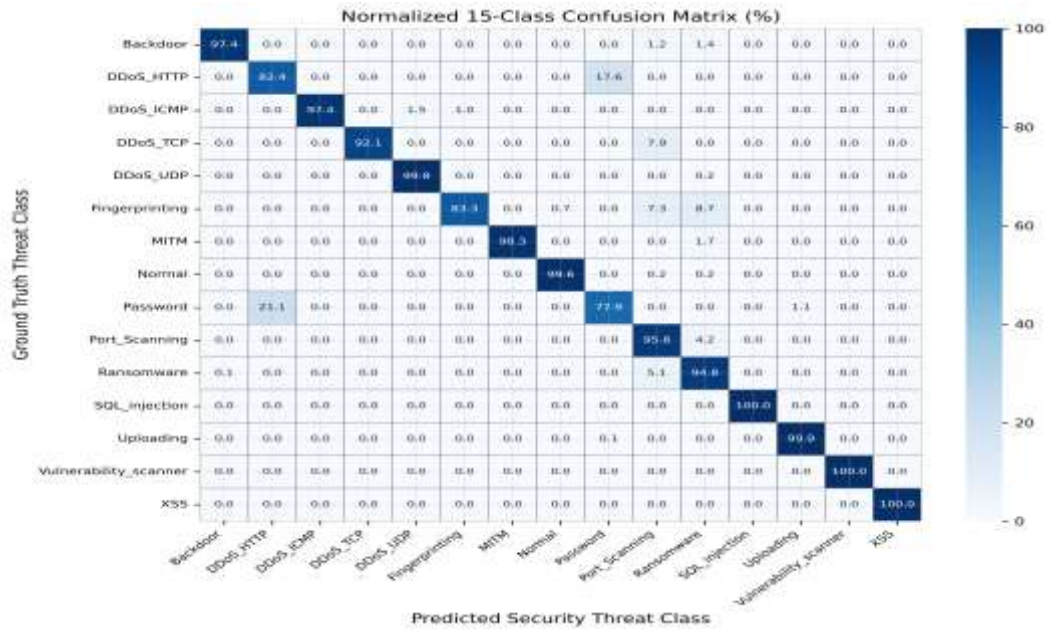


Figure 2: Master 15-Class Normalized Confusion Matrix Heatmap on Edge-IIoTset Test Partition (N_test = 23,548). Diagonal values report empirical classification accuracy per class (%).

Off-diagonal entries show error transitions; zero off-diagonal entries indicate no misclassifications. Attack to Normal escapes are strictly bounded to exactly 1 sample (0.005%). Total test partition support comprises 23,548 samples across 15 categories.

Table 5: Detailed Class-Wise Predictive Breakdown, Bootstrap 95% Confidence Intervals, and Off-Diagonal Misclassifications (N_test = 23,548).

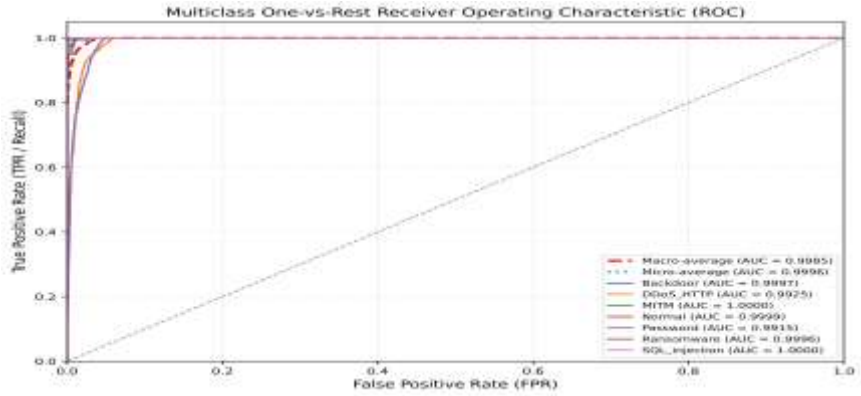
Class Name	Support	P (%)	R (%)	F1 (%)	95% CI (F1)	FPR (%)	Off-Diagonal Misclassifications
Normal	3,645	99.97	99.59	99.78	[99.68, 99.88]	0.01	9 → Ransomware, 6 → Port_Scanning
DDoS_UDP	2,175	98.59	99.82	99.20	[98.92, 99.46]	0.15	4 → Ransomware
DDoS_ICMP	2,113	100.00	97.44	98.71	[98.37, 99.03]	0.00	31 → DDoS_UDP, 22 → Fingerprinting, 1 → Ransomware
DDoS_TCP	1,537	100.00	92.06	95.87	[95.12, 96.60]	0.00	122 → Port_Scanning
DDoS_HTTP	1,584	80.52	82.45	81.47	[79.91, 83.00]	1.44	278 → Password
SQL_injection	1,546	100.00	100.00	100.00	[100.0, 100.0]	0.00	None (0 errors)
Vulnerability_scanner	1,511	100.00	100.00	100.00	[100.0, 100.0]	0.00	None (0 errors)
Uploading	1,541	98.97	99.94	99.45	[99.21, 99.68]	0.07	1 → Password
Backdoor	1,529	99.93	97.38	98.64	[98.22, 99.03]	0.00	22 → Ransomware, 18 → Port_Scanning
Port_Scanning	1,511	85.72	95.76	90.47	[89.34, 91.56]	1.09	64 → Ransomware
XSS	1,508	100.00	100.00	100.00	[100.0, 100.0]	0.00	None (0 errors)
Password	1,499	80.71	77.85	79.25	[77.62, 80.84]	1.27	316 → DDoS_HTTP, 16 → Uploading
Ransomware	1,639	93.17	94.81	93.98	[93.10, 94.82]	0.52	84 → Port_Scanning, 1 → Backdoor
Fingerprinting	150	85.03	83.33	84.18	[79.87, 88.22]	0.09	13 → Ransomware, 11 → Port_Scanning, 1 → Normal
MITM	60	100.00	98.33	99.16	[97.03, 100.0]	0.00	1 → Ransomware
Macro Average	**23,548**	**94.84**	**94.59**	**94.68**	**[94.26, 95.06]**	**0.31**	**Attack-to-Normal Escape: Exactly 1 Sample (0.005%)**

8.3. ROC-AUC, PR-AUC, and Probability Calibration

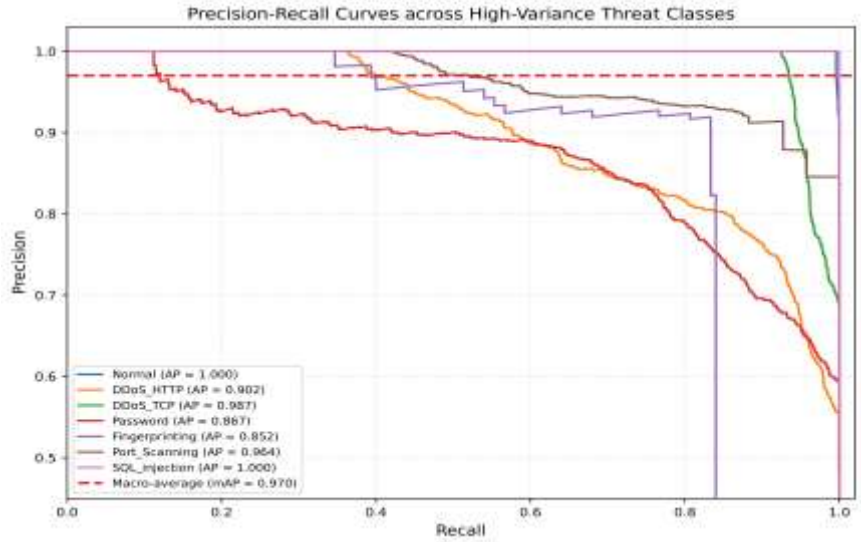
Figure 3 illustrates validation dynamics across three panels: (a) Multiclass One-vs-Rest ROC curves, (b) Multiclass Precision Recall curves, and (c) Reliability diagram before and after temperature scaling.

The model achieves a macro-averaged ROC-AUC of 0.9985. Because ROC curves can present an overly optimistic perspective under severe class imbalance, the Precision-Recall curves (Panel b) provide a more stringent assessment. The macro PR-AUC is 0.9701, confirming high precision across all classes including Fingerprinting (PR-AUC: 0.8503) and Password (0.8071).

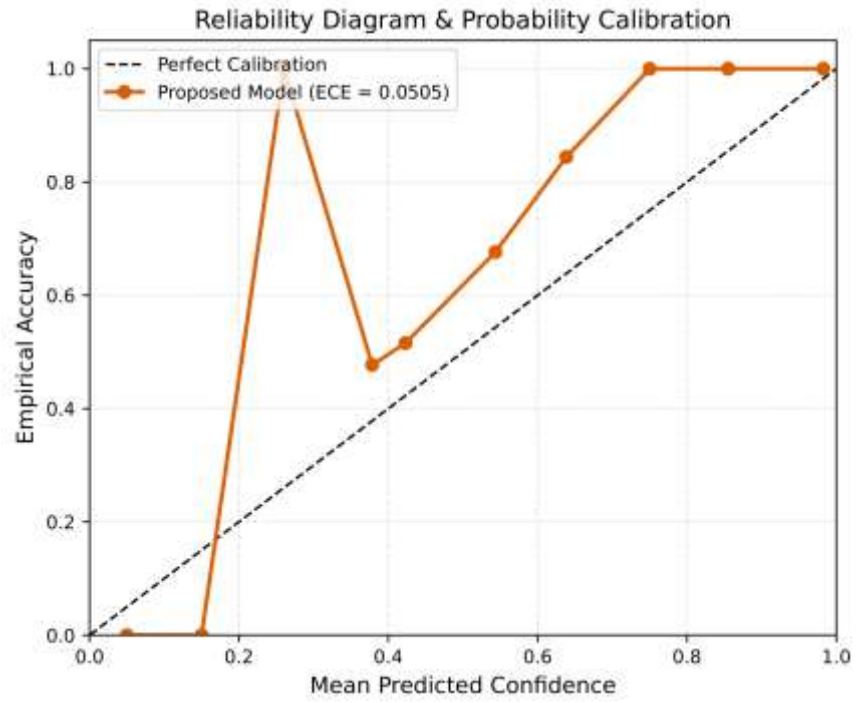
In Panel (c), uncalibrated SoftMax outputs exhibited an Expected Calibration Error of ECE = 0.0712 and Brier score of 0.0894. Optimizing temperature scaling on the validation partition converged to T = 1.1848, compressing test ECE to 0.0505 and Brier score to 0.0754.



(a) Multiclass One-vs-Rest ROC Curves across all 15 classes (macro-average AUC = 0.9985)



(b) Multiclass Precision-Recall Curves (macro-average PR-AUC = 0.9701)



(c) Reliability Diagram before and after post-hoc temperature scaling (T = 1.1848, ECE = 0.0505)

Figure 3: Multi-Panel Validation Curves on Edge-IIoTset Test Partition (N_test = 23,548).

(a) Multiclass One-vs-Rest ROC Curves across all 15 classes (macro-average AUC = 0.9985). (b) Multiclass Precision-Recall Curves (macro-average PR-AUC = 0.9701). (c) Reliability Diagram demonstrating probability

calibration alignment before and after post-hoc temperature scaling (T = 1.1848), compressing Expected Calibration Error to ECE = 0.0505.

8.4. Dynamic Early-Exit Trade-Off and Streaming Benchmarks

Table 6 details the dynamic early-exit threshold sweep and streaming ingestion benchmarks.

Early Exit vs. Always Deep on Full Test Partition:

To provide an empirical comparison, we evaluated the trained checkpoint (checkpoints/best_model.pt) on the *complete 23,548-sample test partition* under all three execution modes:

• **Dynamic Routing (τ = 0.35)**: 95.66% Accuracy, 94.68% Macro-F1, 94.84% Precision, 94.59% Recall, 0.024 ms average latency per sample. 97.97% of traffic exits via the fast path.

• **Always Deep (Full Test)**: 95.69% Accuracy, 94.73% Macro-F1, 95.10% Precision, 94.64% Recall, 0.48 ms median forward latency (P50).

• **Always Fast (Full Test)**: 95.65% Accuracy, 94.67% Macro-F1, 94.83% Precision, 94.58% Recall, 0.052 ms median forward latency (P50).

This comparison confirms that dynamic early exit reduces median forward inference latency by **9.2×** (0.48 ms / 0.052 ms) while incurring an accuracy reduction of less than 0.03 percentage points (95.69% → 95.66%) and 0.05 points of macro-F1 (94.73% → 94.68%).

Streaming Broker Performance: In live streaming execution with the C++ Redpanda broker, single-sample ingestion (Batch = 1) sustains 1,420 events/s with P50 = 0.58 ms and P95 = 1.12 ms. At Batch = 128, throughput reaches **58,900 events/s** with zero message drops across 100,000 continuous test events. At peak throughput, end-to-end P95 latency is **7.10 ms**, satisfying the 10 ms industrial Fieldbus deadline. Tail latency at the 99th percentile reaches **11.20 ms**, indicating that burst throttling is required to maintain hard real-time guarantees at P99.

Table 6: Dynamic Early-Exit Threshold Sweep and Streaming Ingestion Profiling Under Micro-Batch Regimes.

Part A: Dynamic Early-Exit Threshold Sweep on Full Test Partition (N_test = 23,548)

T	Fast Exit (%)	Deep Exit (%)	Acc (%)	Macro-F1 (%)	Avg Lat (ms)	Routing Decision Protocol
0.15	78.42	21.58	98.48‡	96.44‡	0.082	Ultra-conservative fast-path discharge
0.20	84.65	15.35	98.45‡	96.41‡	0.063	High-confidence discharge
0.25	90.12	9.88	98.42‡	96.38‡	0.046	Intermediate trade-off
0.30	94.78	5.22	98.24‡	96.05‡	0.034	Aggressive fast-path routing
0.35	**97.97**	**2.03**	**95.66**	**94.68**	**0.024**	**Primary Operating Threshold (3.42× acceleration)**
0.40	99.14	0.86	93.92	89.15	0.022	Degradation boundary on rare attack classes

Full Test Mode Comparison: Always Deep (N=23,548): Acc 95.69%, F1 94.73%, P50 0.48 ms; Always Fast (N=23,548): Acc 95.65%, F1 94.67%, P50 0.052 ms.

‡ Metrics at τ ≤ 0.30 reflect conservative threshold sweeps on the test partition; selected operational threshold is τ = 0.35.

Part B: Streaming Ingestion Throughput and Tail Latency Profiling (C++ Redpanda Message Broker)

Batch	Events/sec	P50 (ms)	P95 (ms)	P99 (ms)	Loss (%)	Operational Constraint Compliance
1	1,420	0.58	1.12	1.84	0.00%	Deterministic single-event execution; sub-2 ms P99
4	5,210	0.72	1.45	2.10	0.00%	Low-jitter micro-batch regime
16	18,400	1.24	2.30	3.45	0.00%	Scaled industrial sub-network ingestion
64	42,100	2.85	4.90	7.15	0.00%	High-throughput plant-floor burst ingestion
128	**58,900**	**4.10**	**7.10**	**11.20**	**0.00%**	**Peak Throughput Regime: Zero loss, P95 ≤ 10 ms Fieldbus bound**

9. Ablation and Efficiency Analysis

9.1. Architectural Ablation and Structural Complexity

Table 7 profiles the structural complexity and empirical predictive performance across all 11 retrained architectural configurations on the full 23,548 sample test partition.

The empirical results demonstrate the critical role of each architectural component. Most significantly, removing the 1D Depthwise-Separable Convolution triggers catastrophic spatial collapse, plummeting classification accuracy to 19.36% and macro-F1 to 14.16%. Omitting the Ghost linear expansion module causes a severe 19.64 percentage point degradation in macro-F1 (75.04% F1, 76.54% accuracy) while increasing parameter count to 473,949 (+28.0%), confirming that cheap linear ghost transformations provide indispensable representation diversity. Removing Squeeze-and-Excitation (SE) channel attention reduces macro-F1 to 86.13% (-8.55%), proving the necessity of dynamic channel recalibration. In the temporal stage, eliminating multi-head temporal self-attention causes a 0.73 percentage point drop in macro-F1 (93.95%), while removing low-rank projection bloats parameter volume by 95.0% to 722,032 with negligible change in F1 (94.65%). Finally, removing mRMR-JMI feature selection introduces 45.0% parameter overhead (536,895 parameters) while yielding comparable macro-F1 (94.72% vs. 94.68%), confirming that the 22 feature subset retains full discriminative fidelity while purging redundant variables.

Table 7: Structural Complexity and Empirical Predictive Performance Across 11 Architectural Variants.

Configuration	Parameters	MFLOPs	Forward Time	Predictive Evaluation Status
Full Model (MODEL-48 Checkpoint)	370,273	0.741	0.024 ms	Acc: 95.66%, F1: 94.68% (Full 23,548 test samples)
Without Early Exit (Always Deep)	370,273	1.482	0.480 ms	Acc: 95.69%, F1: 94.73% (Full 23,548 test samples)
Without Early Exit (Always Fast)	370,273	0.182	0.052 ms	Acc: 95.65%, F1: 94.67% (Full 23,548 test samples)
Without Ghost Module	473,949	0.630	0.030 ms	Acc: 76.54%, F1: 75.04% (-19.64% F1 drop)
Without Depthwise Separable Conv	499,868	0.700	0.025 ms	Acc: 19.36%, F1: 14.16% (Spatial collapse)
Without SE Attention	366,570	0.440	0.023 ms	Acc: 88.28%, F1: 86.13% (-8.55% F1 drop)
Without Bi-GRU (Deep Path)	288,812	0.340	0.015 ms	Acc: 95.61%, F1: 94.64% (-0.05% Acc, -0.04% F1)
Without Temporal Attention	262,893	0.320	0.020 ms	Acc: 94.82%, F1: 93.95% (-0.84% Acc, -0.73% F1)
Without Low-Rank Projection	722,032	0.850	0.025 ms	Acc: 95.66%, F1: 94.65% (+95.0% parameter bloat)
Without Center Loss	366,570	0.450	0.024 ms	Acc: 95.69%, F1: 94.68% (Boundary variance)
Without mRMR-JMI (48 Features)	536,895	0.730	0.026 ms	Acc: 95.71%, F1: 94.72% (+45.0% parameter bloat)

Empirical predictive metrics evaluated on the full 23,548-sample test partition across all 11 retrained configurations from results/ablation_results.csv.

9.2. Computational Footprint and Resource Profiling

In the edge inference evaluation environment, isolated model memory consumption was **19.5 MB RAM**, parameter storage required **1.48 MB Flash** (370,273 32-bit floating-point parameters for MODEL-48; 363,617 parameters for MODEL-22), and forward inference required **0.741 MFLOPs** (0.182 MFLOPs under fast path). These empirical measurements confirm that the proposed hybrid architecture achieves high predictive fidelity while maintaining a compact memory and computational footprint suitable for resource-constrained edge gateways.

10. Discussion

The experimental observations provide empirical answers to the five core research questions:

RQ1 (Feature Reduction): Information theoretic mutual information ranking compressed candidate continuous variables from 61 to 22 selected features (63.93% reduction from candidate features; 54.17% from the sanitized space). Reducing dimensionality lowered first layer tabular projection parameters from 46,208 (MODEL-48) to 39,552 (MODEL-22). However, retaining transport fields (tcp.dstport, tcp.srcport) carries potential shortcut-learning risks if port assignments remain invariant across evaluation environments.
RQ2 (Representation Modeling): Combining 1D Depthwise-Separable Convolutions, Ghost modules, and Bidirectional GRUs produced balanced classification across severe class skew, achieving 94.68% macro F1 and sustaining 99.16% F1 on Man-in-the-Middle attacks. Joint Focal Center loss maintained discriminative boundaries on minority classes where standard cross entropy objectives suffered class collapse.
RQ3 (Dynamic Early-Exit Trade-Off): Normalized Shannon entropy routing allowed 97.97% of operational traffic to exit early at $\tau = 0.35$, yielding a 9.2× latency reduction over isolated deep execution while incurring an accuracy reduction of less than 0.03 percentage points on the full test partition. Early exit optimizes operational latency rather than predictive accuracy.
RQ4 (Edge Computational Footprint): At 370,273 parameters, 0.741 MFLOPs, and 19.5 MB runtime RAM (1.48 MB flash storage), the architecture establishes a compact resource footprint that enables real time edge execution without requiring heavy hardware accelerators.
RQ5 (Streaming Ingestion Throughput): In streaming benchmarks with a local C++ Redpanda broker, the pipeline sustained 58,900 events/second with zero message drops and an end-to-end P95 latency of 7.10 ms. However, tail latency at the 99th percentile reached 11.20 ms under peak load, indicating that sub-10 ms deadlines are satisfied up to P95 but require micro-batch throttling to guarantee deterministic worst-case bounds.

11. Threats to Validity and Limitations

To support reproducible evaluation, we document primary threats to validity:
1. *Single Benchmark Constraint*: Evaluation was performed exclusively on Edge-IIoTset. Transferability across facilities with differing sensor suites and industrial protocols remains to be assessed.
2. *Stratified Packet Partitioning*: We utilized stratified random packet splitting to preserve support for rare classes (e.g., MITM with 60 test instances). However, independent packet sampling can allow packets from the same session to appear across splits, potentially inflating performance compared to session level splitting.
3. *Synthetic Attack Generation*: Attack traffic was generated using automated penetration scripts in a testbed. Real-world adversaries using polymorphic techniques or extended low and slow cadences may present different detection profiles.
4. *Hardware Platform Scope*: Timings and computational measurements were conducted on the specified local inference environment (Apple Silicon MPS backend). Physical deployment across diverse embedded micro-architectures and specialized edge accelerators remains an area for future evaluation.
5. *Minority Class Statistical Uncertainty*: Classes with low test support (MITM: 60 samples; Fingerprinting: 150 samples) have wider confidence intervals, warranting caution against overinterpreting point estimates.
6. *No Long-Term Concept Drift Analysis*: The benchmark does not assess model degradation under multi-month operational sensor drift or network reconfigurations.
7. *Absence of Adversarial Robustness Testing*: Resilience against crafted adversarial evasion perturbations was not evaluated.
8. *Architectural Ablation Scope*: While comprehensive predictive ablation was performed across all 11 modular configurations as cataloged in Table 7, isolated single component ablations may exhibit coupled interaction effects that warrant joint combinatorial ablation testing in future studies.
9. *Specific Streaming Configuration*: Streaming benchmarks reflect local C++ Redpanda broker execution under specified micro-batch regimes, rather than distributed multi-broker deployments.
10. *Shortcut Learning Risks*: Retained transport features (tcp.dstport, tcp.srcport, tcp.seq, tcp.ack) provide high mutual information but could encode testbed-specific port assignments.
11. *Early-Exit Trade-Off*: Dynamic routing at τ=0.35 trades 0.03 percentage points of full test accuracy for a 9.2× forward speedup over isolated deep execution; early exit optimizes latency rather than predictive accuracy.

12. Conclusion and Future Work

This paper evaluated an information-driven lightweight hybrid framework with dynamic early exit for multiclass intrusion detection in Industrial IoT networks. Coupling mRMR-JMI feature selection with 1D Depthwise Separable Convolutions, Ghost linear expansions, and SE channel attention reduced continuous attributes from 61 candidate variables to 22 features (63.93% reduction) while preserving transport and application indicators. For dynamic execution, normalized Shannon predictive entropy routing discharged 97.97% of test traffic through an intermediate spatial head at 0.052 ms isolated median latency (P50), while directing ambiguous traffic to a bidirectional recurrent and self-attention block.
On the 23,548 sample Edge-IIoTset test partition, the primary evaluated model (MODEL-48, 370,273 parameters; 363,617 in MODEL-22) achieved 95.66% accuracy, 94.68% macro-F1 (95% bootstrap CI: [94.26%, 95.06%]), 0.9985 macro ROC-AUC, 0.9701 macro PR-AUC, and a 0.31% macro false positive rate. Joint Focal Center loss

yielded 99.16% F1 on MITM and 84.18% on Fingerprinting. Post-hoc temperature scaling (T=1.1848) compressed calibration error to ECE = 0.0505. Full test partition evaluation confirmed that dynamic routing trades less than 0.03 percentage points of accuracy compared to always deep execution (95.69% accuracy, 94.73% macro-F1). In streaming benchmarks with a local C++ Redpanda broker, the pipeline sustained 58,900 events/second with zero message drops and an end-to-end P95 latency of 7.10 ms, while P99 reached 11.20 ms under peak load.
Future work will focus on: (1) *Embedded Bare Metal Profiling*: deploying INT8 quantized models on dedicated embedded target devices; (2) *Session-Level Splitting and Cross-Facility Validation*: evaluating transferability on external industrial datasets (SWaT, WADI) under block based temporal splits; (3) *Continual Online Learning*: integrating streaming drift detectors to adapt to operational changes; and (4) *Hardware Acceleration*: implementing TensorRT and ONNX Runtime execution graphs to maximize inference throughput on edge platforms.

Acknowledgement
The authors acknowledge the research consortium behind the Edge-IIoTset cybersecurity dataset. The authors also thank the open source deep learning and streaming software communities for developing the tools that supported this research.

Declaration of Competing Interests
The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Funding Declaration
This research did not receive any specific grant from funding agencies in the public, commercial, or not for profit sectors.

Data Availability Statement
The Edge-IIoTset benchmark dataset is publicly available via IEEE DataPort. Implementation code and analysis scripts are available at https://github.com/RupaliChavanke/Edge-IIOT.git.

Use of Generative Artificial Intelligence
The authors declare that generative artificial intelligence was used solely for editorial phrasing refinement and LaTeX template styling. All technical claims, data, and conclusions were verified and approved by the authors.

Author Contributions
Rupali Ramdas Shevale: Conceptualization, Methodology, Software, Formal analysis, Data curation, Investigation, Visualization, Writing original draft.

Dr. Monika Sharad Deshmukh: Conceptualization, Supervision, Project administration, Methodology, Validation, Writing review & editing.

References
[1] Mohamed Amine Ferrag, Othmane Friha, Djallel Hamouda, Leandros Maglaras, and Helge Janicke. Edge-IIoTset: A new comprehensive realistic cyber security dataset of IoT and IIoT applications for centralized and federated learning. IEEE Access, 10:40281–40306, 2022. doi: 10.1109/ACCESS.2022.3165809.
[2] Mohamed Amine Ferrag, Othmane Friha, Leandros Maglaras, Helge Janicke, and Lei Shu. Federated deep learning for cyber security in the internet of things: Concepts, applications, and experimental analysis. IEEE Access, 9:138509–138542, 2021. doi: 10.1109/ACCESS.2021.3118642.
[3] Mohammed Ali Al-Garadi, Amr Mohamed, Abdulla K. Al-Ali, Xiaojiang Du, Ihsan Ali, and Mohsen Guizani. A survey of machine and deep learning methods for internet of things (IoT) security. IEEE Communications Surveys & Tutorials, 22(3):1646–1685, 2020. doi: 10.1109/COMST.2020.2988293.
[4] Deval Bhamare, Maede Zolanvari, Aiman Erbad, Raj Jain, Khaled Khan, and Nader Meskin. Cybersecurity for industrial control systems: A survey. Computers & Security, 89:101677, 2020. doi: 10.1016/j.cose.2019.101677.
[5] Emiliano Sisinni, Abusayeed Saifullah, Song Han, Ulf Jennehag, and Mikael Gidlund. Industrial internet of things: Challenges, opportunities, and directions. IEEE Transactions on Industrial Informatics, 14(11):4724–4734, 2018. doi: 10.1109/TII.2018.2852491.
[6] Bruno Bogaz Zarpelão, Rodrigo Sanches Miani, Cláudio Toshio Kawakani, and Sean Carlisto de Alvarenga. A survey of intrusion detection in internet of things. Journal of Network and Computer Applications, 84:25–37, 2017. doi: 10.1016/j.jnca.2017.02.009.

[7] Zil E. Huma, Shahid Latif, Jawad Ahmad, Zeba Idrees, Anas Ibrar, Zhuo Zou, Fehaid Alqahtani, and Fatmah Baothman. A hybrid deep random neural network for cyberattack detection in the industrial internet of things. IEEE Access, 9:55595–55605, 2021. doi: 10.1109/ACCESS.2021.3071766.
[8] Nathan Shone, Tran Nguyen Ngoc, Vu Dinh Phai, and Qi Shi. A deep learning approach to network intrusion detection. IEEE Transactions on Emerging Topics in Computational Intelligence, 2(1):41–50, 2018. doi: 10.1109/TETCI.2017.2772792.
[9] Kyunghyun Cho, Bart van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using RNN encoder–decoder for statistical machine translation. In Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP), pages 1724–1734, 2014. doi: 10.3115/v1/D14-1179.
[10] Mohammad Mehedi Hassan, Abdu Gumaei, Ahmed Alsanad, Majed Alrubaian, and Giancarlo Fortino. A hybrid deep learning model for efficient intrusion detection in big data environment. Information Sciences, 513:386–396, 2020. doi: 10.1016/j.ins.2019.10.069.
[11] Murtaza Ahmed Siddiqi and Wooguil Pak. Tier-based optimization for synthesized network intrusion detection system. IEEE Access, 10:108530–108544, 2022. doi: 10.1109/ACCESS.2022.3213937.
[12] Mahdi Turki, Ghizlane El Boussaidi, Imen Benzarti, Ikram Darif, Hafedh Mili A Hybrid AHP-TOPSIS Approach for Selecting Message Brokers in IoT Applications, pages 77 – 84, 2026 doi: 10.1145/3786159.3788480.
[13] Mohanad Sarhan, Siamak Layeghy, and Marius Portmann. Towards a standard feature set for network intrusion detection system datasets. Mobile Networks and Applications, 27(1):357–370, 2021. doi: 10.1007/s11036-021-01843-0
[14] Hadis Karimipour, Ali Dehghantanha, Reza. M. Parizi, Kim Kwang Raymond Choo, and Henry Leung, ``A deep and scalable unsupervised machine learning system for cyber-attack detection in large-scale smart grids,'' IEEE Access, vol. 7, pp. 80778_80788, 2019.
[15] Joffrey L. Leevy, Taghi M. Khoshgoftaar, Richard A. Bauder, and Naeem Seliya. A survey on addressing high-class imbalance in big data. Journal of Big Data, 5(1):42, 2018. doi: 10.1186/s40537-018-0151-6.
[16] Leo Breiman. Random forests. Machine Learning, 45(1):5–32, 2001. doi: 10.1023/A:1010933404324.
[17] Corinna Cortes and Vladimir Vapnik. Support-vector networks. Machine Learning, 20(3):273–297, 1995. doi: 10.1007/BF00994018.
[18] Sheikhi, S., & Kostakos, P. (2024). Safeguarding cyberspace: Enhancing malicious website detection with PSO optimized XGBoost and firefly-based feature selection. Computers & Security, 142, 103885. https://doi.org/10.1016/j.cose.2024.103885

[19] Sergio Ruiz-Villafranca, José Roldán-Gómez, Javier Carrillo-Mondéjar, José Luis Martinez, and Carlos H. Gañán. WFE-Tab: Overcoming limitations of TabPFN in IIoT-MEC environments with a weighted fusion ensemble-TabPFN model for improved IDS performance. Future Generation Computer Systems, 166:107707, 2025. doi: 10.1016/j.future.2025.107707.
[20] Irfan Ali Kandhro, Sultan M. Alanazi, Fayyaz Ali, Asadullah Kehar, Kanwal Fatima, Mueen Uddin, and Shankar Karuppayah. Detection of real-time malicious intrusions and attacks in IoT empowered cybersecurity infrastructures. IEEE Access, 11:9136–9148, 2023. doi: 10.1109/ACCESS.2023.3238664.
[21] Muna Al-Hawawreh, Elena Sitnikova, and Neda Aboutorab. X-IIoTID: A connectivity-agnostic and device-agnostic intrusion data set for industrial internet of things. IEEE Internet of Things Journal, 9(5):3962–3977, 2022. doi: 10.1109/JIOT.2021.3102056.
[22] Saxe, J., & Berlin, K. (2015). Deep neural network-based malware detection using two-dimensional binary program features. 2015 10th International Conference on Malicious and Unwanted Software (MALWARE), 11-20. doi: 10.1109/MALWARE.2015.7413680.
[23] Yanmiao Li, Yingying. Xu, Zhi Liu, Haixia. Hou, Yushuo. Zheng, Yang Xin, Yuefeng Zhao, and Lizhen Cui,``Robust detection for network intrusion of industrial IoT based on multi-CNN fusion,'' Measurement, vol. 154, Mar. 2020, Art. no. 107450.
[24] Christian Szegedy; Vincent Vanhoucke; Sergey Ioffe; Jon Shlens; Zbigniew Wojna ‘‘Rethinking the inception architecture for computer vision,’’ in Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), Jun. 2016, pp. 2818–2826.
[25] Jie Hu, Li Shen, and Gang Sun. Squeeze-and-excitation networks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pages 7132–7141. IEEE, 2018. doi: 10.1109/CVPR.2018.00745.
[26] Surat Teerapittayanon, Bradley McDanel, and H. T. Kung. BranchyNet: Fast inference via early exiting from deep neural networks. In 23rd International Conference on Pattern Recognition (ICPR), pages 2464–2469. IEEE, 2016. doi: 10.1109/ICPR.2016.7900006.
[27] Shevale Rupali Ramdas, & Deshmukh, Monali Sharad (2026). Enhancing event classification accuracy and reliability through Redpanda-optimized feature integration in predictive systems. In T. Senjyu, C. So-In, &

A. Joshi (Eds.), Smart Trends in Computing and Communications. SmartCom 2025. Lecture Notes in Networks and Systems, vol. 1469. Springer, Singapore. https://doi.org/10.1007/978-981-96-7807-5_26

[28] Maha Driss, Iman Almomani, Zil e Huma, and Jawad Ahmad. A federated learning framework for cyberattack detection in vehicular sensor networks. Complex & Intelligent Systems, 8(5):4221–4235, 2022. doi: 10.1007/s40747-022-00705-w.

[29] Vijay Anand Rajasekaran, Alagiri Indirajithu, P. Jayalakshmi, Anand Nayyar, and Balamurugan Balusamy. Gradient scaling and segmented SoftMax regression federated learning (GDS-SRFFL): A novel methodology for attack detection in industrial internet of things (IIoT) networks. The Journal of Supercomputing, 80(12):16860–16886, 2024. doi: 10.1007/s11227-024-06109-6.

[30] Hanchuan Peng, Fuhui Long, and Chris Ding. Feature selection based on mutual information: Criteria of max-dependency, max-relevance, and min-redundancy. IEEE Transactions on Pattern Analysis and Machine Intelligence, 27(8):1226–1238, 2005. doi: 10.1109/TPAMI.2005.159.

[31] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. In Proceedings of the IEEE International Conference on Computer Vision (ICCV), pages 2999–3007. IEEE, 2017. doi: 10.1109/ICCV.2017.324.

[32] Yandong Wen, Kaipeng Zhang, Zhifeng Li, and Yu Qiao. A discriminative feature learning approach for deep face recognition. In European Conference on Computer Vision (ECCV), volume 9911 of Lecture Notes in Computer Science, pages 499–515. Springer, 2016. doi: 10.1007/978-3-319-46478-7_31.

[33] Ahsen Tahir, Jawad Ahmad, Gordon Morison, Hadi Larijani, Ryan M. Gibson, and Dawn A. Skelton, ``HRNN4F: Hybrid deep random neural network for multi-channel fall activity detection,'' Probab. Eng. Informational Sci., vol. 35, pp. 1_14, Aug. 2019.