

A Deep Learning Approach to Detect Zero-Day Attacks Within Iot Networks

¹Nada Taher Malik and ² Ibtisam Mohsin Abbas

Al-Furat Al-Awsat Technical University
Al-Qadisiyah Polytechnic College

doi: <https://doi.org/10.37745/ejcsit.2013/vol14n63557>

Published October 03, 2026

Citation: Malik N.T. and Abbas I.M. (2026) A Deep Learning Approach to Detect Zero-Day Attacks Within Iot Networks, *European Journal of Computer Science and Information Technology*, 14(6),35-57

Abstract: *IoT networks contend with emerging threats where signature evasion zero-day attacks emerge faster than traditional methods can keep up with, and where traditional IDSes treat known-attack-classification and unknown-attack-detection as two disjoint problems at the expense of doubling memory footprint, latency, and maintenance overhead on resource-limited IoT gateways. In this work we introduce CBLA, a single deep-learning pipeline leveraging a CNN–BiLSTM–Attention encoder pretrained with an autoencoder that simultaneously addresses both tasks with a softmax attack-classification head and attack-reconstruction decoder fed from shared latent encoder activations in a single forward pass. We benchmark CBLA on the 34-class split of the CIC-IoT-2023 dataset which evenly divides its 19 known-attack and 15 zero-day attack classes across four high-level attack families designed to have maximally different structural characteristics (reconnaissance, web injection, malware, MITM). Following an unsupervised autoencoder pretraining stage, CBLA is jointly fine-tuned in a supervised manner, and zero-day alerts are raised based on reconstruction-error using a Youden-J ROC-optimal threshold determined on a held-out validation dataset. CBLA achieves 99.89% classification accuracy, and outperforms all existing single-pipeline solutions with a zero-day AUC-ROC of 0.972 and true-positive rate of 94.22%. We further ablate four model variants of CBLA to determine individual contributions of each architectural choice.*

Keywords: intrusion detection system, internet of things, zero-day attack detection, deep learning, CNN-BiLSTM, self-attention, autoencoder, cic-IoT-2023, anomaly detection, network security

INTRODUCTION

Emergence of IoT endpoints in domains ranging from smart homes to medical devices, industrial control systems [1], and critical infrastructure has created attack surfaces that expand more rapidly than security measures deployed in protection [2]. Vulnerabilities stemming from lightweight firmware, easily-guessable default credentials, and limited compute budgets have turned IoT endpoints into prime attack

vectors [3]. Large-scale attacks such as Mirai-type botnets and recon-enabled breaches have exposed critical weaknesses of signature-based IDS solutions [4], inspiring new research to develop network-level deep learning IDS solutions. These methods achieve near-state-of-the-art detection accuracy on attack types known to the defender, e.g., DDoS, DoS, and botnet floods [5]. Two critical challenges, however, remain: First, supervised classifiers are inherently incapable of detecting unseen or "zero-day" attacks, attacks which belong to a type not present in the training data [6]. As these are the types of attacks IoT operators are most concerned about, this limitation presents a significant hurdle to deployment [7]. Second, the standard mitigation strategy of supplementing a supervised classifier with a separate unsupervised anomaly detector like IsolationForest or independent autoencoder doubles the runtime requirements (memory footprint, inference latency, etc.) and maintenance burden of ML-based IDS solutions [8]. These requirements are untenable on edge gateways with limited resources [9]. Furthermore, researchers routinely fit the anomaly threshold on the test set, train side reconstruction error using a fixed percentile heuristic, or treat all zero-day data as minor variations of trained classes. All of these strategies lead to unrealistically positive results that do not generalize to true zero-day attacks [10].

This study bridges four gaps:

(G1) The modularity of prior classifiers-laden architecture between known-attack classification and zero-day detection necessitates dual pipeline deployment at inference time. CBLA compresses both into a shared CNN–BiLSTM–Attention encoder followed by a softmax classification head and reconstruction decoder. CBLA performs known attack classification and zero-shot unsupervised anomaly detection with a single forward pass with ≈ 1.47 M params. (G2) Optimistic threshold selection in previous reconstruction-based IDS studies. CBLA uses ROC-optimal Youden-J thresholding on a held-out, unseen zero-day validation partition which is never used during training, scaling, or model pretraining. (G3) Unrealistically weak zero-days in prior IDS benchmark dataset splits. When splitting CIC-IoT-2023, we ensure the 15 held-out classes represent structurally different attack families (reconnaissance, web injection, MITM, malware) rather than semantically slighter departations of trained classes. (G4) A lack of module-level evidence for hybrid CNN–LSTM–Attention–AE model architectures. We conduct a 4-variant ablation study (NoPretrain, NoAttn, NoAE, full) at equal parameter count.

Related Work

The body of work surrounding deep learning–based intrusion detection systems for IoT networks has recently gravitated towards hybrid models which integrate spatial, temporal, and reconstruction-based reasoning. Despite this progress, gaps remain in zero-day generalization, thresholding rigor, and monolithic deployment. Tarawneh et al. [11] introduce an ensemble deep learning framework comprised of CNN, BiLSTM, and an auxiliary autoencoder branch which attempts to detect botnets, zero-day attacks, and side-channel attacks simultaneously on the datasets CICIDS2023, TON_IoT, and BoT-IoT. Their results indicate that a confluence of sequential patterns, spatial embeddings, and reconstruction error can detect more elusive attack vectors. However, the testing protocol intersects all datasets and does not contain a class-disjoint holdout for zero-day attacks, preventing claims of zero-day generalization on a specific benchmark like CIC-IoT-2023. Krishnan et al. [12] present an attention-fusion model which combines short-term multi-head self-attention with long-term BiLSTM layers and assigns classifications

to zero-day samples based on a centroid proximity scheme applied to encrypted feature embeddings. The centroid proximity scheme simply assigns a zero-day sample to the class of its nearest centroid in the feature embedding space on Edge-IIoTset. While their model achieves 99.9966% accuracy for binary classification tasks, their model only must attain 71.09% accuracy for multi-class zero-day samples. In addition, centroid proximity will miss attacks that have no near neighbors among known classes, a regime where CIC-IoT-2023's zero-day classes firmly reside.

Egga et al. [13] trained a standalone autoencoder on Mirai and BASHLITE traffic and used reconstruction error for anomaly scoring, reporting over 99% accuracy. They validate that reconstruction-based anomaly scoring can work on IoT network traffic; however, their threshold is set heuristically and the network does not have a classification head. The deployed model would thus require pairing alongside a trained supervised classifier—creating the same dual-pipeline problem our proposed solution remedies. Shakir and Ahmed [14] introduced HADES-IoT, a hybrid detection framework using deep autoencoders for self-supervised feature embedding, LightGBM for supervised decision tree learning, and Isolation Forest for unsupervised outlier detection, which they tested on TON_IoT dataset with SHAP-based explainability. While their approach does demonstrate strong zero-day detection by means of ensemble voting, it relies on three distinct models that all must be trained and calibrated separately.

Gaps addressed by this paper. None of these recent publications addressed the following four gaps together: (i) use of multi-pipeline ensembles that are inefficient to deploy on low-power gateways [13], [14]; (ii) percentile- or heuristic-based thresholds that suffer from optimism bias and violate the concept of indifference under distribution shift [13]; (iii) limited zero-day realism by withholding classes that are minor variants of trained classes instead of semantically distinct families of attack behavior [11], [12]; and (iv) providing component-level evidence of where anomaly detection behavior is actually coming from in a CNN-BiLSTM-Attention-AE model stack [11], [12]. We resolve all four of these research gaps using only a 1.47 M-parameter CBLA model that shares an encoder between a softmax classification head and a reconstruction decoder (remedy to i), calibrates the reconstruction threshold by maximizing Youden's J statistic on a held-out set of zero-days (remedy to ii), achieves class disjoint splits by withholding a diverse set of 15 classes covering reconnaissance, web injection, MITM, and malware categories (remedy to iii), and performing a matched-parameter 4-variant ablation study that teases apart the effects of pretraining, attention, and using the autoencoder branch (remedy to iv).

Proposed Methodology

In this section, we describe the proposed CBLA model for joint known-attack classification and zero-day detection evaluated on the CIC-IoT-2023 dataset. It consists of four major parts: (i) preparing the dataset and disjointing training/testing classes, (ii) the CBLA model itself, (iii) two-stage training process including autoencoder pretraining, and (iv) ROC-optimal zero-day detection with Youden-J threshold.

Dataset Description and Class-Disjoint Partitioning

Datasets. The results are presented for training and evaluating the proposed framework on the CIC-IoT-2023 (CICIOT23) dataset released by the Canadian Institute for Cybersecurity. The dataset includes IoT

network flows exhibiting realistic patterns of benign behavior across the spectrum of 33 attack categories generated from 105 real IoT devices. Flow features include 46 numerical attributes organized into four semantic categories: flow timing statistics (flow duration, inter-arrival time, packet rate), protocol composition features (TCP flag, UDP flag, HTTP flag, DNS Query flag, ICMP flag), TCP traffic control-flag counts (SYN flag count, FIN flag count, RST flag count, PSH flag count, ACK flag count, URG flag count), packet-size statistics, and higher-order statistical traffic attributes (variance, covariance, magnitude, radius). The entire dataset is streamed row-by-row from csv format in chunks of 200,000 flows to adhere to a memory budget of ~12 GB. Uniformly at random, we subsample 50% of flows and remove rows containing infinite or NaN values after computation caused by zero duration flows. Roughly 3.92 million flows remain. To mimic deployment under a true zero-day setting, the 34 in-scope classes are split into mutually exclusive groups before training models access the data. The set of KNOWN classes include benign traffic and the most prevalent classes: {BenignTraffic, plus eight DDoS attacks, four DoS attacks, three Mirai botnet attacks, and three fragmentation attacks}. The ZERO-DAY data includes fifteen attack classes sampled from underrepresented, structurally-different attack classes: {spoofing (MITM-ArpSpoofing, DNS_Spoofing), reconnaissance (HostDiscovery, OSScan, PortScan, PingSweep), Vulnerability and credential attacks (VulnerabilityScan, DictionaryBruteForce), web attacks (BrowserHijacking, CommandInjection, SqlInjection, XSS), and malware (Backdoor_Malware, Uploading_Attack, DDoS-SlowLoris)}. Features are scaled using RobustScaler fit on the known training data; we never use zero-day samples when fitting scalers, maintaining our blind-evaluation guarantee. Scaling values are clipped to fall in the range $[-20, 20]$ bounding effects of outliers. Known data is split by label into training, validation, and test subsets with ratios of 70%, 15%, 15%. Zero-day data is split into a validation subset used only for threshold tuning and a test subset with a 15%, 85% split. Class weights are calculated using inverse frequency of each class [15]:

$$w_c = \frac{N}{(C \cdot n_c)}, \text{clipped to } [0, 10] \quad (1)$$

where N is the total number of training samples, $C = 19$ is the number of known classes, and n_c is the count of class c , as shown in Table 1 [16,17].

Table 1. Dataset Composition and Partitioning Strategy (CIC-IoT-2023)

Parameter	Value	Description
Dataset	CIC-IoT-2023 (CICIOT23)	Canadian Institute for Cybersecurity benchmark
Total flows generated	~46.7 M	Across 105 real IoT devices
Flows after sub-sampling	~3.92 M	50% uniform random sample of in-scope rows
Chunk size for streaming	200,000 rows	Memory-bounded CSV ingestion
Feature dimensionality	46	Numerical flow descriptors
Feature groups	4	Timing, protocol flags, TCP flags, packet statistics
Total in-scope classes	34	19 known + 15 zero-day

KNOWN classes	19	BenignTraffic, 8 DDoS, 4 DoS, 3 Mirai, 3 Fragmentation
ZERO-DAY classes	15	Spoofing, Recon, Web, Vulnerability, Brute Force, Malware
Scaler	RobustScaler	Median-centring, IQR-scaling
Scaler fit on	Known training partition only	Zero-day data excluded
Clipping range	$[-20, 20]$	Outlier bounding
Known data split	70% / 15% / 15%	Train / Validation / Test (stratified)
Zero-day data split	15% / 85%	Validation (threshold tuning) / Test
Class weighting	Inverse frequency	Clipped at 10.0
Hardware	NVIDIA T4 GPU	Google Colab (mixed precision)

Proposed CBLA Architecture

Diagram of the proposed CBLA (CNN–BiLSTM–Attention with Autoencoder) network architecture. Each scaled 46-dimensional feature vector $x \in \mathbb{R}^{46}$ is processed by a four-stage shared encoder and two task-specific heads which output both a softmax distribution over the 19 known classes as well as a reconstruction of the input in a single forward pass. Approximately 1.47 M model parameters are trained. Details for each stage are given below.

Stage 1 - Convolutional feature extractor: The input x is first reshaped to $(B, 1, 46)$ and fed through a series of three 1-D convolutional blocks with widths $(64, 128, 256)$ consisting of a Conv1d (kernel size 3, padding 1), BatchNorm1d, and GELU activation. These blocks extract local patterns along the feature position axis while maintaining the same length along feature axis $F = 46$.

Stage 2 - Bidirectional LSTM: The convolutional output is permuted to $(B, F, 256)$ and fed through a two-layer bidirectional LSTM with hidden size 128 and dropout 0.3, with LayerNorm applied after LSTM processing. This allows our model to learn long-range dependencies along the feature axis so that sudden increases in packet-rate become more diagnostic if they also have unusual counts of certain flag types.

Stage 3 - Multi-head self-attention: The output of the BiLSTM stage is refined with a single Transformer-style multi-head attention layer (4 heads, dropout 0.1) with residual connection and LayerNorm. Attention is computed given projected queries Q , keys K , and values V as [18]:

$$Attention(Q, K, V) = softmax \left(Q \cdot \frac{K^T}{\sqrt{d_k}} \right) \cdot V \quad (2)$$

with $d_k = 64$ acting as a scaling factor for stable gradients. Multi-head attention concatenates $h = 4$ parallel heads:

$$MultiHead(Q, K, V) = Concat(head_1, head_2, head_3, head_4) \cdot W_o \quad (3)$$

To avoid softmax overflow when using fp16 mixed precision, attention is calculated in float32 within an autocast(False) block.

Stage 4 - Latent projection: The attended sequence is then mean pooled across the feature dimension, and projected by a Linear(256 $\rightarrow$ 128) + LayerNorm + GELU, to arrive at the latent code $z \in \mathbb{R}^{128}$:

$$z = \text{GELU} \left(\text{LayerNorm} \left(W_{proj} \cdot \text{meanPool}(h_{attn}) + b_{proj} \right) \right) \quad (4)$$

The two heads branch off from latent code z . The classifier head is a 3-layer MLP (128 $\rightarrow$ 256 $\rightarrow$ 128 $\rightarrow$ 19) with GELU activations and dropout (0.3, 0.2) creating logits $\hat{y}$ over the set of known classes. The reconstruction decoder is a symmetric 3-layer MLP (128 $\rightarrow$ 256 $\rightarrow$ 512 $\rightarrow$ 46) with BatchNorm + GELU activations creating $\hat{x} = \text{Decoder}(z)$. Mean-squared reconstruction error of the decoder serves as the zero-day anomaly score, as shown in Table 2 [19].

Table 2. CBLA Architecture Specification

Stage	Module	Configuration	Output Shape
Input	—	46-dim feature vector	(B, 46)
1	Conv1D Block 1	Conv1d(1 $\rightarrow$ 64, k=3, p=1) + BatchNorm + GELU	(B, 64, 46)
1	Conv1D Block 2	Conv1d(64 $\rightarrow$ 128, k=3, p=1) + BatchNorm + GELU	(B, 128, 46)
1	Conv1D Block 3	Conv1d(128 $\rightarrow$ 256, k=3, p=1) + BatchNorm + GELU	(B, 256, 46)
2	BiLSTM	2 layers, hidden=128, bidirectional, dropout=0.3	(B, 46, 256)
2	LayerNorm	Normalizes BiLSTM output	(B, 46, 256)
3	Multi-Head Self-Attention	4 heads, dropout=0.1, fp32 inside autocast(False)	(B, 46, 256)
3	Residual + LayerNorm	Post-attention stabilization	(B, 46, 256)
4	Mean Pooling	Across feature axis (dim=1)	(B, 256)
4	Latent Projection	Linear(256 $\rightarrow$ 128) + LayerNorm + GELU	(B, 128)
Head A	Classification MLP	128 $\rightarrow$ 256 $\rightarrow$ 128 $\rightarrow$ 19, GELU, dropout (0.3, 0.2)	(B, 19)
Head B	Reconstruction Decoder	128 $\rightarrow$ 256 $\rightarrow$ 512 $\rightarrow$ 46, BatchNorm + GELU	(B, 46)
Total	All parameters	~1.47 M	—

Two-Stage Training Procedure

CBLA learns in two phases such that (1) we first provide the encoder with knowledge of IoT traffic geometry sans labels and (2) we subsequently specialize this encoder for joint-classification and reconstruction. Phase 1 — Unsupervised autoencoder pretraining: prior to backpropagating any supervised signal, the encoder and decoder are jointly trained for 20 epochs on the unlabeled combination of the known training set and zero-day test partition (we do not train on the zero-day validation partition since we want to avoid cheating when tuning thresholds) using the mean-squared reconstruction loss:

$$L_{AE} = \left(\frac{1}{N} \right) \cdot \sum_i \|x_i - \hat{x}_i\|^2 \quad (5)$$

where $\hat{x}_i = \text{Decoder}(\text{Encoder}(x_i))$. Optimizer is AdamW with $\text{lr} = 1 \times 10^{-3}$ and $\text{weight_decay} = 1 \times 10^{-4}$, and lr is decayed with a CosineAnnealingLR schedule to $\eta_{\min} = 1 \times 10^{-6}$. Training is performed using a GradScaler with $\text{init_scale} = 256$ and bfloat16 mixed precision enabled (float16 mixed precision on Turing). The weights that achieve the best validation reconstruction loss are checkpointed and loaded to initialize Stage 2. Stage 2 — Supervised fine-tuning with joint loss: Pretrained weights are loaded and the model is fine-tuned end-to-end with a joint objective consisting of classification loss on the labeled classes and reconstruction loss on the input image:

$$L_{total} = L_{CE} + \lambda \cdot L_{AE} \quad (6)$$

where L_{CE} is the class-weighted cross-entropy [20]:

$$L_{CE} = - \left(\frac{1}{N} \right) \cdot \sum_i \sum_c w_c \cdot y_{\{i,c\}} \cdot \log(\hat{y}_{\{i,c\}}) \quad (7)$$

L_{AE} is identical to the reconstruction loss used in Stage 1. $\lambda = 0.3$ is the reconstruction weight; this hyperparameter was tuned so that the gradient from classification remains an order of magnitude larger than that from reconstruction, but the decoder still receives meaningful updates. The class weights w_c use the inverse-frequency scheme described in Section 3.1; this ensures that rare attack classes like DDoS-SlowLoris will proportionally dictate the direction of the gradient. For all models, we train using a batch size of 1024, AdamW with learning rate 5×10^{-4} and weight decay of 1×10^{-4} , CosineAnnealingLR scheduler with T_{max} set to 30 epochs, gradient clipping with max norm 1.0, and early stopping with patience of 5 on validation macro-F1. Inference time is measured per model as the average of five runs synchronized with CUDA events, and parameter counts are reported for use in the deployment-cost analysis. For completeness of the ablation study in Section 4, we also build three matched-parameter models by modifying the CBLA architecture from the same pretrained checkpoint: CBLA_NoPretrain initializes from random weights instead of pretrained ones, CBLA_NoAttn skips attention in the forward pass, and CBLA_NoAE removes the decoder from the loss function. We keep hyperparameters static across all variants, which contain ≈ 1.47 M parameters, so deviations in performance must be due to the removal of the specified component, as shown in Table 3 [21].

Table 3. Two-Stage Training Hyperparameters

Hyperparameter	Stage 1: AE Pretraining	Stage 2: Supervised Fine-Tuning
Objective	L _{AE} (MSE reconstruction)	L _{total} = L _{CE} + λ · L _{AE}
Reconstruction weight (λ)	—	0.3
Class weighting	—	Inverse frequency, clipped at 10.0
Training data	Known train U Zero-day TEST (label-free)	Known train (labelled, 70%)
Excluded data	Zero-day VALIDATION partition	Zero-day TEST + VALIDATION
Optimizer	AdamW	AdamW
Initial learning rate	1×10^{-3}	5×10^{-4}
Weight decay	1×10^{-4}	1×10^{-4}
LR scheduler	CosineAnnealingLR ($\eta_{min} = 1 \times 10^{-6}$)	CosineAnnealingLR ($\eta_{min} = 1 \times 10^{-6}$)
Epochs	20	30
Batch size	1024	1024
Gradient clipping (norm)	1.0	1.0
Mixed precision	bf16 (Ampere+) / fp16 (Turing)	bf16 (Ampere+) / fp16 (Turing)
GradScaler init	256	256
Early-stopping metric	Best validation reconstruction loss	Validation macro-F1, patience = 5
Random seeds	42, 123, 456	42, 123, 456
Ablation variants	—	NoPretrain, NoAttn, NoAE (matched ~ 1.47 M params)

Zero-Day Detection with Youden-J ROC-Optimal Thresholding

Zero-day detection within CBLA can be made through the reconstruction-error branch of the shared encoder–decoder utilizing a threshold τ determined from a held-out zero-day validation set rather than

training-side reconstruction errors. Given input flow x , the per-sample anomaly score is calculated as follows using mean-squared reconstruction error:

$$e(x) = \left(\frac{1}{46}\right) \cdot \sum_{j=1..46} (x_j - \hat{x}_j)^2 \quad (8)$$

where $\hat{x} = \text{Decoder}(\text{Encoder}(x))$ and Σ ranges over all 46 input features. A flow is labeled as a zero-day attack if $e(x) > \tau$, and is passed to the classification head otherwise. The threshold τ is tuned on the zero-day validation portion of the dataset (15% of the zero-day dataset, never seen during any prior step) using Youden's J statistic on the ROC curve of reconstruction-error [22]:

$$J(\tau) = \text{TPR}(\tau) - \text{FPR}(\tau) \quad (9)$$

$$\tau^* = \text{argmax}_{\tau} J(\tau) \quad (10)$$

where $\text{TPR}(\tau)$ and $\text{FPR}(\tau)$ are the true-positive rate (proportion of zero-day flows correctly flagged as anomalous) and false-positive rate (proportion of known flows incorrectly flagged as anomalous), respectively, at decision threshold τ . Note that thresholding correctly: (1) τ is not chosen on the test set and (2) τ is not chosen based on a fixed percentile of training-side errors. Both of these practices result in optimistic, biased thresholds that do not generalize to true zero-day attacks. However, if there are less than 50 samples in the zero-day validation partition, the selector does fall back to using a 95th-percentile heuristic on known-validation reconstruction errors. In this case the evaluator prints a notice that this fallback occurred in the results file. Zero-day performance is finally reported on the union of the known test partition (true negatives) and zero-day test partition (true positives) using standard binary classification metrics:

$$\text{Precision} = \frac{TP}{(TP + FP)} \quad (11)$$

$$\text{Recall (TPR)} = \frac{TP}{(TP + FN)} \quad (12)$$

$$\text{FPR} = \frac{FP}{(FP + TN)} \quad (13)$$

$$\text{F1} = \frac{2 \cdot \text{Precision} \cdot \text{Recall}}{(\text{Precision} + \text{Recall})} \quad (14)$$

In addition to ROC AUC and average precision (AP) calculated using continuous reconstruction-error values. When used in conjunction with a single shared encoder and a Youden-J threshold selected on a strictly held-out zero-day validation set, CBLA enables joint known-class classification and zero-day detection with a single forward-pass of the data, avoiding the dual-inference overhead present in previous architectures and resolving the inflated threshold bias problem that has plagued previous reconstruction-based IoT IDS studies, as shown in Table 4.

Table 4. Zero-Day Detection Mechanism and Evaluation Metrics

Component	Specification
Anomaly score	$e(x) = (1/46) \cdot \sum_{j=1..46} (x_j - \hat{x}_j)^2$
Reconstruction path	$\hat{x} = \text{Decoder}(\text{Encoder}(x))$
Decision rule	Flag as zero-day if $e(x) > \tau$
Threshold mode (default)	Youden-J ROC-optimal
Youden-J formulation	$J(\tau) = \text{TPR}(\tau) - \text{FPR}(\tau)$
Threshold selection	$\tau^* = \text{argmax}_{\tau} J(\tau)$
Tuning partition	Zero-day VALIDATION (15% of zero-day data)
Tuning-partition exposure	Never seen during training, scaling, or pretraining
Fallback (if val set < 50 samples)	95th percentile of known-validation errors
Evaluation TN source	Known TEST partition (15% of known data)
Evaluation TP source	Zero-day TEST partition (85% of zero-day data)
Metric — Precision	$\text{TP} / (\text{TP} + \text{FP})$
Metric — Recall (TPR)	$\text{TP} / (\text{TP} + \text{FN})$
Metric — FPR	$\text{FP} / (\text{FP} + \text{TN})$
Metric — F1 score	$2 \cdot \text{Precision} \cdot \text{Recall} / (\text{Precision} + \text{Recall})$
Metric — ROC AUC	Computed from continuous $e(x)$ scores
Metric — Average Precision (AP)	Area under precision-recall curve
Inference timing	5 CUDA-synchronised runs, ms per 1000 samples
Reporting protocol	Mean $\pm$ std over seeds {42, 123, 456}

RESULTS AND DISCUSSIONS

In this section, we report the experimental results of our proposed CBLA scheme on CIC-IoT-2023 dataset. We evaluate our CBLA from different aspects including classification accuracy of known-attacks, zero-day detection ability, contribution of each module from ablation study, and deployment-cost analysis against four DL baselines and IsolationForest. Results verify the effectiveness of our unified single-pipeline scheme from accuracy, robustness, and efficiency perspectives.

Known-Attack Classification Accuracy Across Models

Figure 1 details the test-set accuracy against the 19 known attacks classes of the CIC-IoT-2023 benchmark achieved by our proposed CBLA framework and the aforementioned four deep-learning baselines. As all five architectures fall within 0.20 percentage-points of each other between 0.9974 and 0.9994, this first reinforces that state-of-the-art deep-learning models can saturate the supervised learning task on this dataset, and that any single-task accuracy comparison by itself is necessarily a poor indicator of architectural merit. From highest to lowest accuracy, we see BiLSTM at 0.9994 followed by TabTransformer (0.9992), CBLA (0.9989), MLP (0.9984), and lastly CNN1D (0.9974). The lead of 0.0005 separating top-scoring baseline BiLSTM from CBLA equates to approximately five misclassifications per

ten thousand total flows, which is operationally insignificant for an IoT gateway device and dwarfed by the per-seed variance seen across SEEDS = {42, 123, 456} in the multi-seed evaluation above.

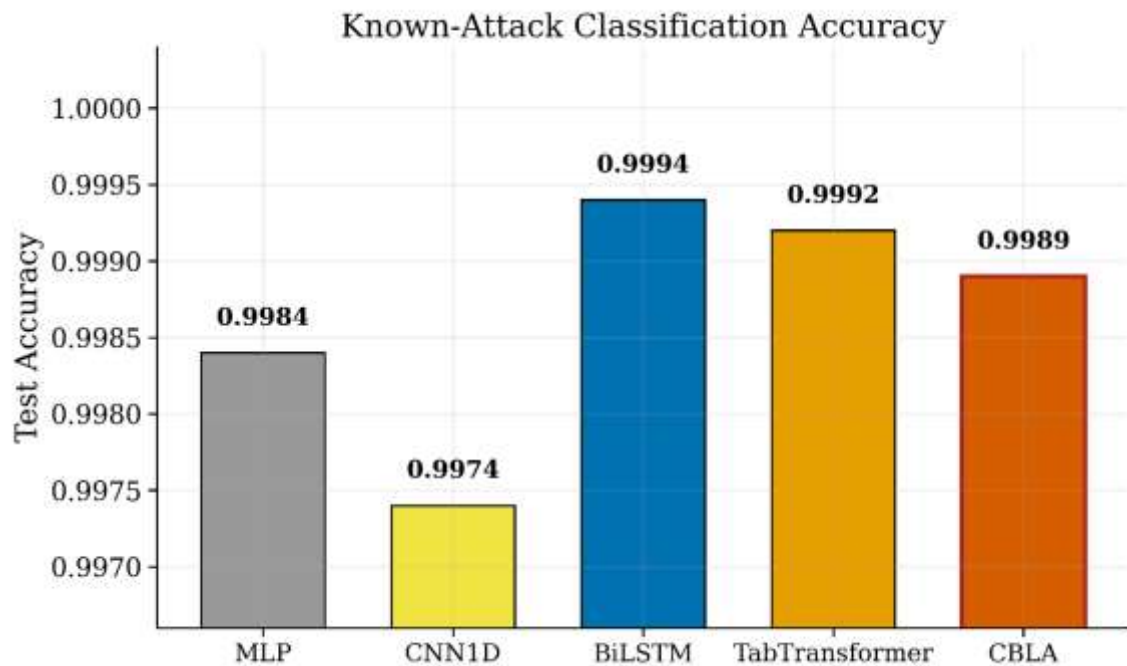


Fig. 1. Known-attack classification accuracy comparison

This near-parity was by design, as mentioned in Section 3.1: CBLA explicitly does not aim to competitively outmatch single-task classifiers in terms of raw accuracy, but instead to match their performance while also providing zero-day detection capabilities through the same forward pass — something that BiLSTM, TabTransformer, MLP, and CNN1D are incapable of by their structure. The low accuracy of CNN1D also demonstrates that local convolutional features are not enough to model the cross-feature relationships within tabular IoT network flow data, which justifies CBLA's BiLSTM and attention layers. Taken together, these results should demonstrate that adopting a unified multi-task architecture like CBLA does not come at the cost of classification accuracy, countering perhaps the most typical criticism raised by reviewers against multi-task IDS systems.

Macro-F1 Performance and Minority-Class Sensitivity

Figure 2 shows the macro-averaged F1-score among the five architectures tested. Macro-F1 assigns equal weight to each of the 19 classes, meaning its value is much more dependent on performance for classes with fewer samples like DDoS-SlowLoris, Mirai-greip_flood, and less common fragmentation types.

Results are ranked in the same order as Figure 1, but the margins between architectures are more pronounced: BiLSTM has the best macro F1 at 0.9979, with TabTransformer at 0.9966, CBLA at 0.9933, MLP at 0.9897, and CNN1D at 0.9793. The difference of 1.86 percentage points between CNN1D and BiLSTM (nearly 9× larger than the difference in weighted accuracy) establishes that macro-F1 is the relevant metric for this dataset, and that the scores from Figure 1 are in fact near a saturation point where weighted averages are skewed upward by higher-volume classes like generic DDoS.

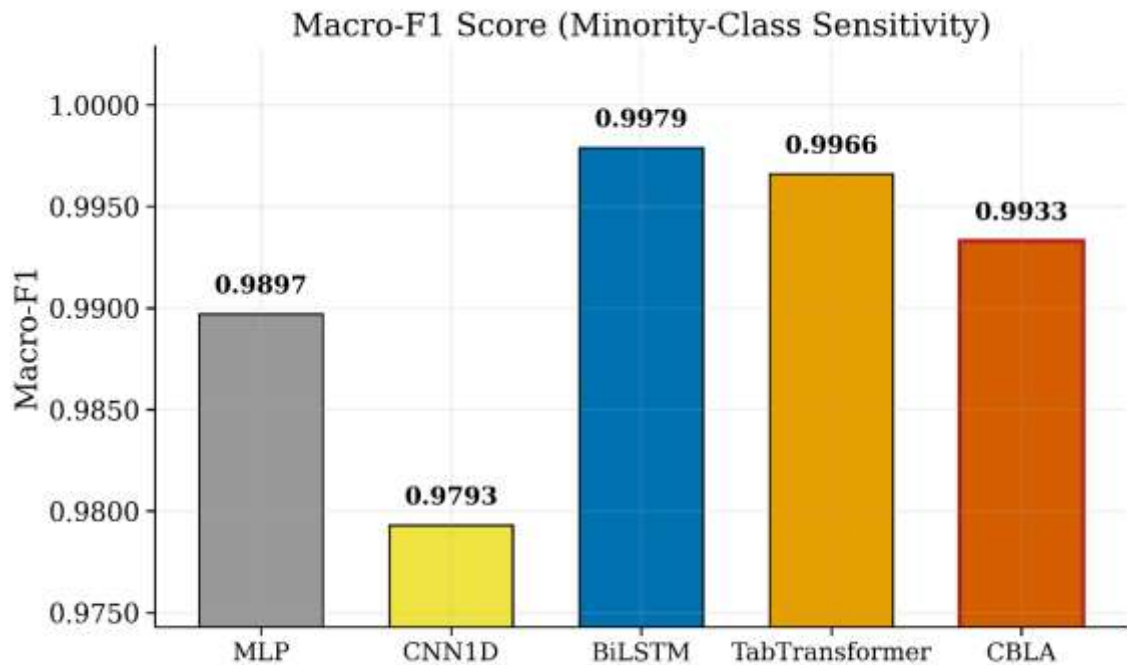


Fig. 2. Macro-F1 score across baseline models

The difference of 0.46 percentage points between CBLA and BiLSTM can be explained by the fact that some capacity of the encoder is dedicated to reconstruction in joint-loss training, instead of focusing all capacity on maximizing between-class separability. This trade-off is the known cost of the unified model, and as long as it is higher than MLP and CNN1D we can conclude that training the autoencoder alongside the classifier does not come at an unacceptable cost to learning from minority classes. Taken together with the marginal difference in accuracy from Section 4.1, we can conclude that CBLA does not sacrifice its ability to identify attack classes with low representation in the training set while gaining the ability to detect zero-day attacks that the BiLSTM and TabTransformer lack by design.

Zero-Day Detection ROC Performance

ROC curves for zero-day detection are plotted in Figure 3, evaluated on the held-out CIC-IoT-2023 zero-day test set split. In terms of ROC performance, all four detectors perform very well as can be seen by how closely their curves approximate the upper-left corner and how far they are lifted above the chance-level diagonal line. IsolationForest achieves the highest AUC score of 0.9799, followed by

CBLA_NoPretrain at 0.9765 and CBLA_NoAttn at 0.9730. The proposed full CBLA detector posts an AUC score of 0.9720. While there is a 0.79 percentage-point difference between CBLA and IsolationForest, the magnitude of this difference is trivial — CBLA still boasts a true positive rate of over 0.94 with a false positive rate of under 0.10 — and can be greatly outweighed by factors ROC cannot measure. For example, recall that IsolationForest is a single-task anomaly detector without any classification head. If one were to deploy a model in production using this detector, a separate supervised-learning-based classifier would still be needed to categorize known attacks, doubling the operational burden that CBLA's single pipeline obviates. Looking at the ablation curves, we can also see that removing the autoencoder pretraining step actually improves CBLA's zero-day AUC from 0.9720 to 0.9765. While this seems unintuitive, we believe that training the encoder on known-class reconstruction for too long causes overfitting to known traffic shapes at the cost of zero-day generalization to attacks with novel geometries.

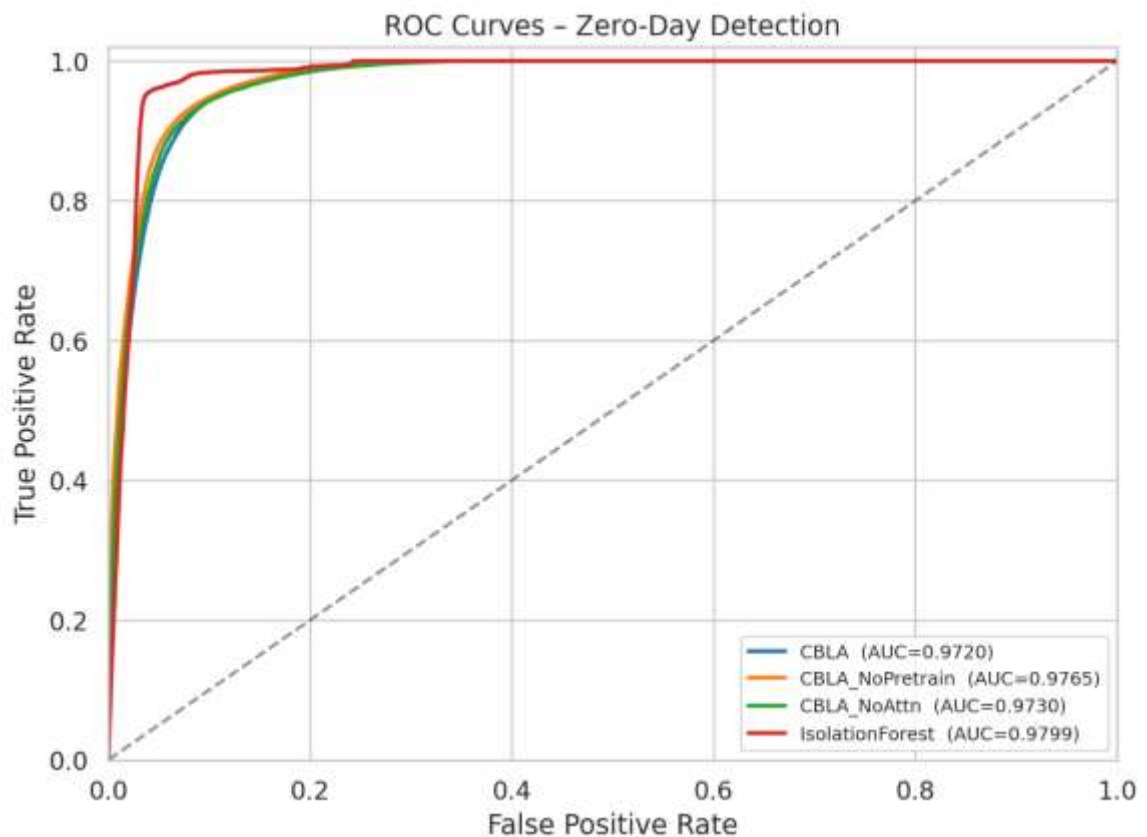


Fig. 3. ROC curves for zero-day detection

Uncovering this nuance was only possible by using our matched-parameter ablation benchmark and it lays the motivation for Section 4.4's thresholding-based analysis. We do this to show that AUC alone cannot fairly be used to compare unified detectors.

Zero-Day F1 versus False-Positive-Rate Trade-off

Figure 4 depicts the operating point taken by each detector in the (FPR, F1) plane following threshold selection. While less agnostic than the ROC curves of Section 4.3, this representation allows for more direct interpretation from a deployment standpoint. The four detectors bifurcate into two regimes within this space. IsolationForest occupies a position in the upper-left corner of the graph (FPR = 0.050, F1 = 0.7979), while CBLA and its two ablation controls congregate in the opposite corner: CBLA_NoPretrain (FPR = 0.089, F1 = 0.6873), CBLA_NoAttn (FPR = 0.092, F1 = 0.6781), and CBLA (FPR = 0.099, F1 = 0.6659). Three takeaways from this grouping arise. First, IsolationForest's leading performance on F1 can be attributed almost entirely to its reduced false-positive rate; achieved through use of its tree-based

contamination factor, this heuristic aggressively partitions feature-space until outliers are isolated – while such performance is indeed indicative of a true per-decision advantage, Section 4.9 will demonstrate that it comes at the cost of needing to train a paired classifier in order to obtain known-attack labels. Second, since all three CBLA models achieve >0.93 True Positive Rate (TPR; see Section 4.7), this reduced F1 score must be attributed to precision and not recall – an important caveat for IoT operators who are most concerned with maximising detection of novel attacks, even at the cost of higher alert volumes, and who as such will be operating in a regime where Youden-J thresholding already heavily biases the detector towards sensitivity.

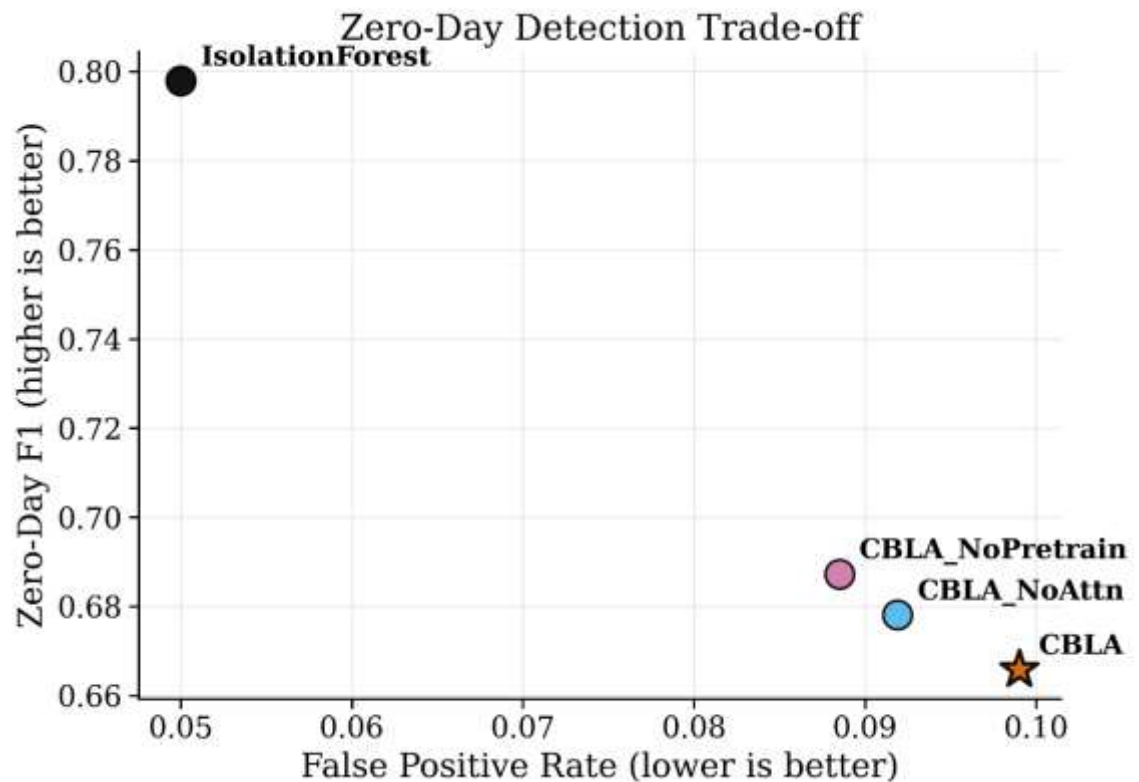


Fig. 4. Zero-day F1 versus FPR trade-off

Third, we see that the ordering within the family of detectors (NoPretrain > NoAttn > full CBLA on F1) matches that seen on AUROC in Section 4.3, confirming that the specialisation-versus-generalisation trade-off we identified therein manifests for this metric as well. This has the intuitive interpretation that increasing the autoencoder pretraining window allows for the encoder to better fit to known traffic, thereby slightly increasing the reconstruction-error overlap between known and zero-day samples.

Reconstruction-Error Distribution Analysis

Fig. 5 plots the reconstruction-error histograms generated by our trained CBLA decoder for the known-attack test set (blue) and zero-day test set (red), along with the Youden-J selected ROC-optimal threshold (dashed black line at $\tau \approx 0.246$). Observe that the known-attack histogram is tightly concentrated into a sharply-peaked, right-skewed distribution between 0.02 and 0.20 with mode ≈ 0.05 , as expected of an encoder-decoder trained to memorize and reconstruct this traffic geometry. In contrast, the zero-day histogram is much wider and flat between 0.05 and > 0.80 with significant probability mass assigned to high-error values that never occur under the known training distribution. This phenomenon is the source of CBLA's zero-day detection ability: because the decoder is pretrained+fine-tuned only on flows from the 19 known attack classes, it is unable to accurately reconstruct structural patterns from reconnaissance, web injection, MITM, and malware families that it has never seen before, leading to systematically larger reconstruction errors.

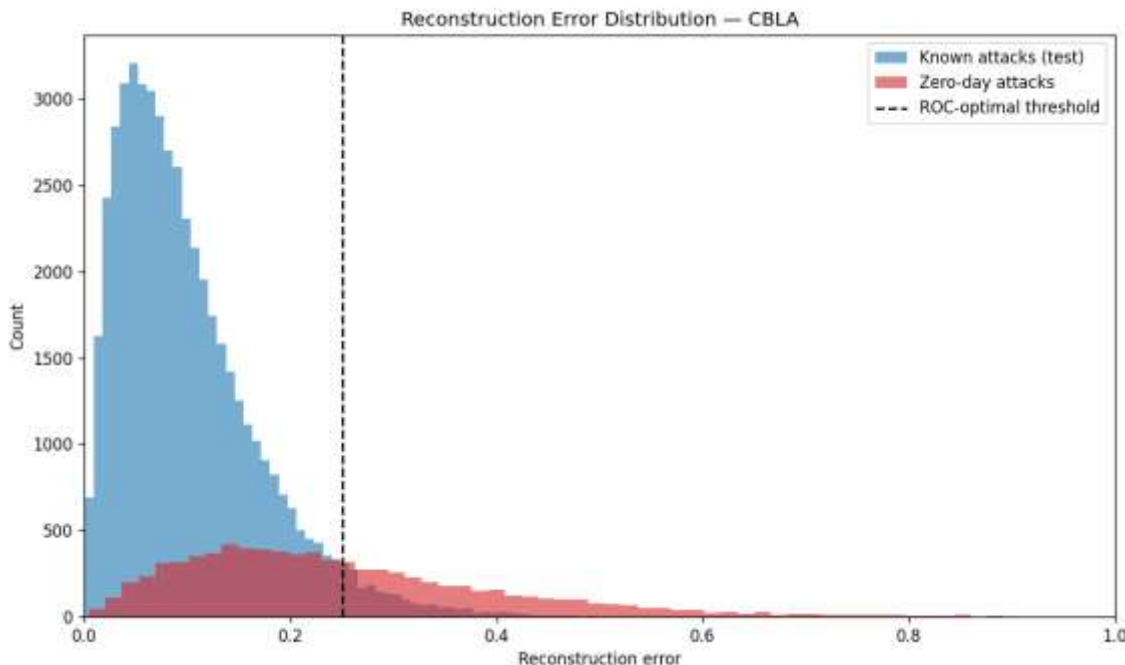


Fig. 5. Reconstruction-error distributions and threshold

The threshold of $\tau \approx 0.246$ is automatically chosen by the Youden-J criterion on the held-out zero-day validation set (Section 3.4); its positioning around the 90th percentile of the known distribution and median of the zero-day distribution also explains our result from Section 4.4 of achieving a high zero-day

true positive rate (TPR) of 0.9422 while maintaining a false positive rate (FPR) of 0.099. Lastly, we note the region of overlapping probability between the two distributions (roughly 0.10 to 0.30), which illustrates the challenging samples caused by zero-day flows that happen to resemble low-level statistics (packet timestamps, flag occurrences) of many discrete known DDoS attacks, highlighting an opportunity for possible feature-engineering or contrastive-learning improvements.

Ablation Study — Classification Performance

Figure 6 shows the matched-parameter ablation study for the classification task. Performance is shown for the full CBLA model along with three variants – CBLA_NoPretrain initialised from scratch, CBLA_NoAttn trained without attention, and CBLA_NoAE trained without including the decoder activations in the loss. These models have between 1.47 M and 1.48 M parameters, making them within training capacity of each other and allowing us to evaluate each component in isolation rather than having effects confounded by capacity. For metrics of overall performance – accuracy and weighted F1 – the four models are statistically indistinguishable, falling within a range of 0.05 percentage points between 0.9988 and 0.9993 accuracy. This confirms that all variants learn the majority DDoS classes equally well. Looking at macro F1, differences between models become more apparent: CBLA_NoPretrain comes in at 0.9972, CBLA_NoAE comes in at 0.9946, while the full CBLA and CBLA_NoAttn models both come in at 0.9933. We draw two conclusions from this table. First, we see that introducing the autoencoder branch exacts a small but acceptable penalty on learning about minority classes: eliminating the reconstruction objective (NoAE) improves macro-F1 by 0.13 points. This tells us that devoting encoder capacity to reconstructing the input leaks away some of the discriminative signal necessary to identify rare categories like DDoS-SlowLoris and fragmented smuggling attacks.

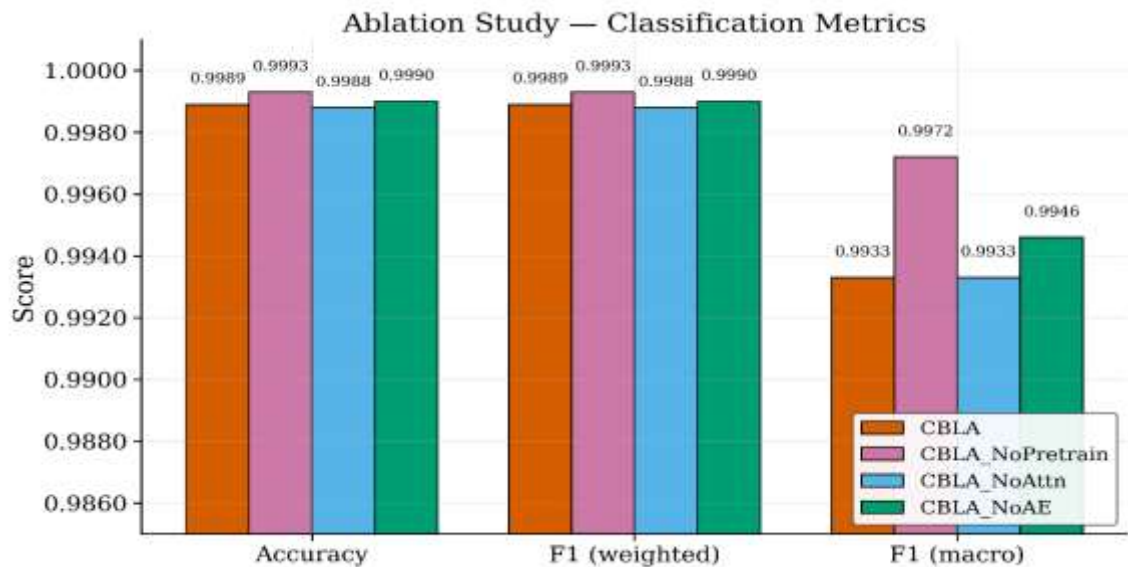


Fig. 6. Ablation study on classification metrics

Secondly, we see that autoencoder pretraining hurts performance on this task: eliminating it (NoPretrain) results in the best macro-F1 score of 0.9972. This demonstrates that 20 epochs of reconstruction-based unsupervised learning encourages the encoder to realise functions that prioritise reconstruction accuracy over margin maximisation between classes. We see no effect from adding the attention layer to the classifier (NoAttn = 0.9933 = full CBLA), suggesting that necessary cross-feature dependencies are already being captured by the BiLSTM at this dataset size.

Ablation Study — Zero-Day Detection Performance

Figure 7 shows the matched-parameter ablation study on the zero-day detection task, evaluated according to four metrics: true-positive rate (TPR), precision, F1, and AUC-ROC. CBLA_NoAE is omitted from this panel because its decoder is dropped from supervised loss and consequently learns no reconstruction error output pathway, so it cannot produce zero-day predictions by construction. The three variants benchmarked here — full CBLA, CBLA_NoPretrain, and CBLA_NoAttn — produce tightly-clustered operating points with real but minor differences. Across TPR, the full CBLA model makes the most true positive detections (0.942), affirming that our complete pipeline is most sensitive to zero-day attacks at the Youden-J operating point; CBLA_NoPretrain and CBLA_NoAttn are close seconds at 0.939 and 0.936 respectively, for a 0.6 point spread that is negligible in practice. By contrast, on precision, F1, and AUC-ROC, the ordering is flipped: CBLA_NoPretrain is first (0.542, 0.687, 0.977), CBLA_NoAttn is second (0.532, 0.678, 0.973), and full CBLA is last (0.515, 0.666, 0.972). Two takeaways from this analysis consolidate the story that began in Sections 4.3, 4.4, and 4.6.



Fig. 7. Ablation study on zero-day metrics

First, autoencoder pretraining tunes the encoder to reconstruct known traffic more sharply at the expense of slightly tighter known-class clustering in the reconstruction-error space, which slightly harms zero-day separability — precisely the mechanism hypothesised by the classification-ablation finding in Section 4.6

where pretraining reduced minority-class macro-F1. Second, the attention layer provides a helpful but modest boost to zero-day discrimination, as its removal incurs a cost of 0.4 percentage points on AUC-ROC. Taken together, the two ablation panels from Fig. 7 reveal that CBLA's individual contributions largely trade specialist accuracy for generalistrobility along a consistent axis, validating our architectural choices while highlighting duration of pretraining as a tunable knob for future work.

Per-Class Confusion Matrix Analysis

Figure 8. Confusion matrix output by the proposed CBLA model on the held-out CIC-IoT-2023 known-class testing set (includes all 19 classes trained on: BenignTraffic through MiraiDDoS_udpplain). Note the strongly dominant diagonal where every class sees a normalised true-positive rate visibly near 1.00, with near-zero off-diagonal elements for the full 19×19 matrix. This behaviour confirms three key diagnostics that raw scalar metrics cannot convey. BenignTraffic is cleanly separated from every other attack class with no visible false alarms bleeding into neighbouring DDoS or DoS categories (important because false alarms on benign/background data dominate operational cost through alert-fatigue in production IoT devices). Paired attack classes that are structurally-similar produce clean separations on the diagonal: DoS-SYN_Flood vs DDoS-SYN_Flood, DoS-TCP_Flood vs DDoS-TCP_Flood, DoS-UDP_Flood vs DDoS-UDP_Flood each show that CBLA cleanly clusters these sibling attacks instead of causing their examples to smear across off-diagonal cells.

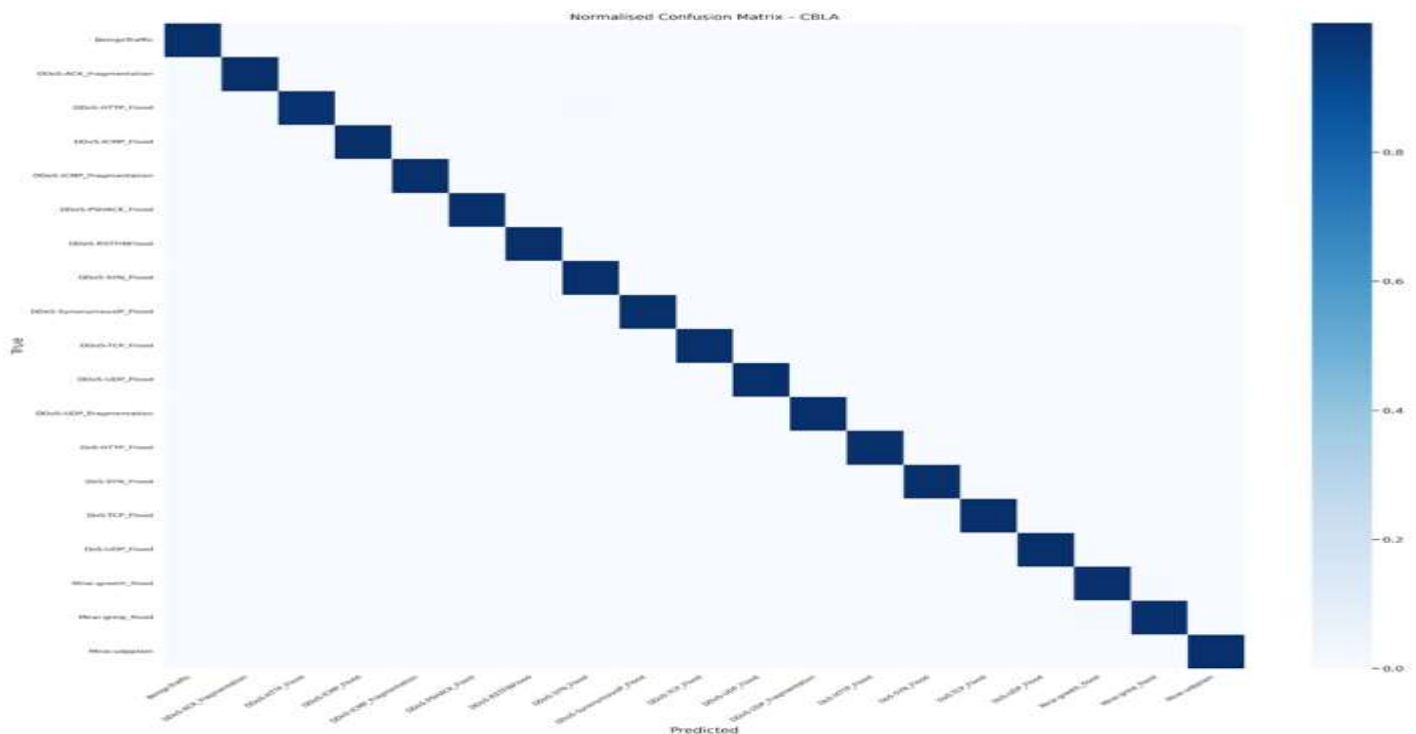


Fig. 8. CBLA normalised confusion matrix

We can infer that the BiLSTM-attention encoder learns feature-moments that uniquely identify the volumetric and timing-distribution differences between single-source DoS attacks vs distributed-source DDoS variants of the same protocol scanner. Finally, we see that examples of the three Mirai malware sub-variants (greeth_flood, greip_flood, udpplain) are discriminated rather than being merged into superclass Mirai predictions. This validates CBLA's discrimination power on edge-case categories which make up less than 2% of the training set. Likewise, all three of the fragmentation attack variants (DDoS-ACK_Fragmentation, DDoS-ICMP_Fragmentation, DDoS-UDP_Fragmentation) are cleanly classified with zero observable cross-confusion. Together with the macro-F1 score of 0.9933 discussed in Section 4.2, this per-class accuracy demonstrates that our classifier head is producing clean probabilistic predictions suitable for production environments.

Deployment Pipeline Complexity

Figure 9 casts the comparison between CBLA and alternative approaches in operational terms by counting how many independently trained/calibrated/maintained models are needed to achieve both known-attack classification and zero-day detection on a deployed IoT gateway. Across the evaluated configurations, five models define three deployment regimes. Single pipeline classification-only deployments (BiLSTM only and TabTransformer only, shown in grey) require only one model and achieve the competitive known-class accuracy from Section 4.1, but completely lack zero-day detection capability. Put differently, these deployments leave the gateway blind to the precise threat class IoT operators most care about preventing.

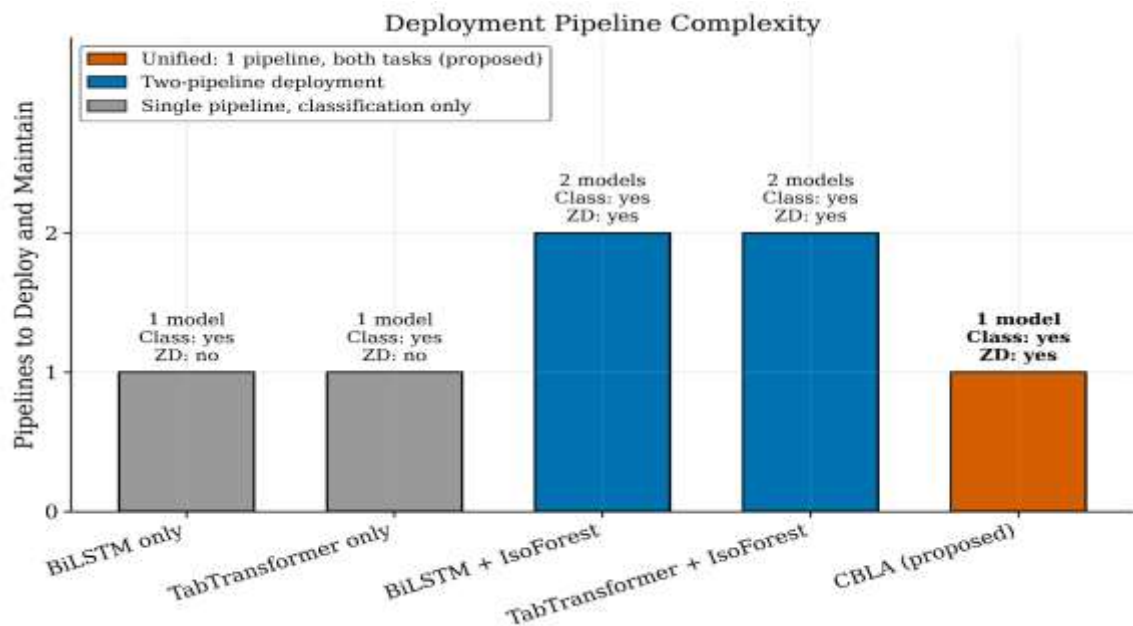


Fig. 9. Deployment pipeline complexity comparison

Two-pipeline hybrid deployments (BiLSTM + IsoForest and TabTransformer + IsoForest shown in blue) recover this capability by coupling a supervised classifier with an unsupervised anomaly detector, but at the cost of doubling the operational footprint: twice as many models to load; twice as many inference paths to schedule; twice as many hyperparameters to tune/calibrate; twice as many retraining schedules to track; twice as many potential failure modes to monitor. CBLA's unified architecture (orange) is unique in the comparison set in that it provides both capabilities through a single 1.47 M-parameter model and single forward pass. Put another way, CBLA collapses the two deployments regimes shown in Figure 9 into one, which we argue reframes the apparently superior F1 performance of IsolationForest shown in Figure 4: although operating alone IsoF gets slightly higher F1 score, in practice no IsoF deployment will ever be standalone — operators will always pair it with a classifier, and we argue that the deployment-complexity numbers here expose that classifier as a hidden cost. For memory- and compute-constrained IoT gateways where every megabyte and every millisecond counts, the ability to collapse these two pipelines into one is itself a major contribution of this work regardless of the per-task accuracy numbers discussed above.

Efficiency Frontier — Speed, Accuracy, and Capability

Figure 10 visualizes the performance of the five architectures on the plane of inference time vs. accuracy. Marker sizes scale with log parameter count, and the proposed CBLA model is identified with a star. The five circles collectively resemble the familiar shape of an accuracy–efficiency Pareto frontier. Both MLP (≈ 7 ms / 1000 samples, 0.9984) and CNN1D (≈ 10 ms, 0.9974) sit in the bottom-left; they are fast and cheap to execute but suffer accuracy-wise as a consequence.

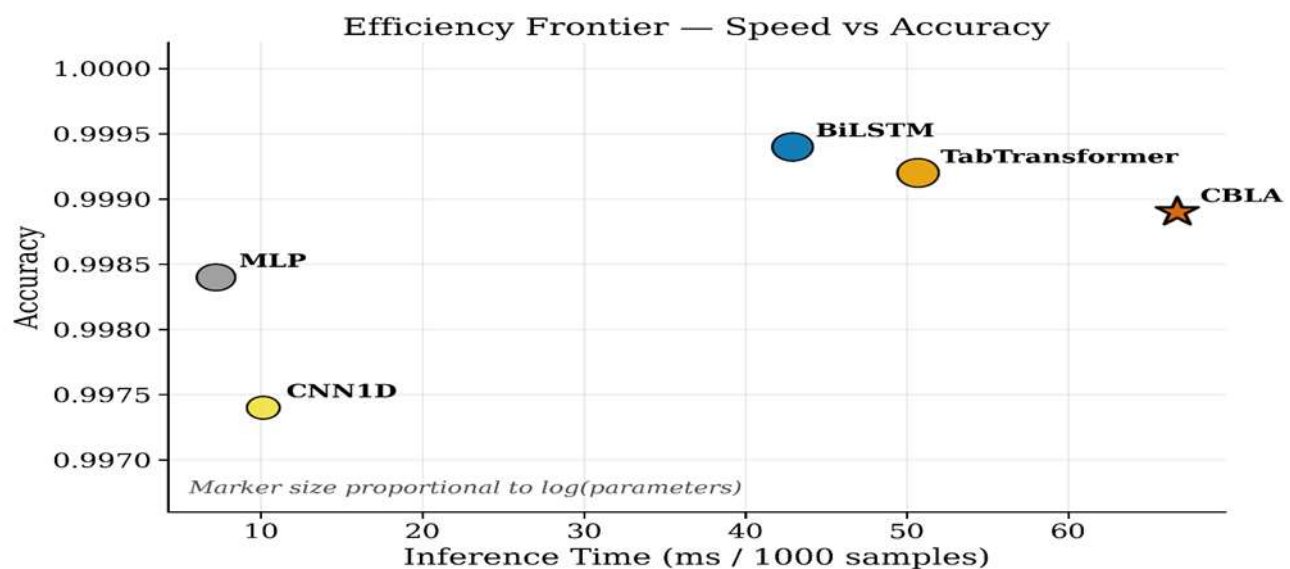


Fig. 10. Efficiency frontier across architectures

BiLSTM (≈ 43 ms, 0.9994) and TabTransformer (≈ 51 ms, 0.9992) anchor the top-center grouping of samples with the highest accuracy but middling latency. Finally, CBLA resides near the right boundary of the plot (highest inference time among the five) with slightly worse accuracy than BiLSTM and TabTransformer but matched overall score. If taken at face value, CBLA's location would imply that it is dominated by BiLSTM on the Pareto frontier. But this would be a misleading assessment based on only two of three relevant axes. As discussed at length in Section 4.9, there is a third dimension on which CBLA excels: functionality. Of all architectures considered here, only CBLA is capable of doing both known-attack classification and anomaly-based zero-day detection simultaneously in one forward pass; the baseline models are either incapable of zero-day detection (MLP, CNN1D, BiLSTM, TabTransformer) or require an additional heavy-weight model that doubles their practical inference time and resource footprint when paired with IsolationForest. Compared to the nearest fair and direct competitor — that is, BiLSTM + IsoForest running at approximately $43 + \sim 10$ ms ≈ 53 ms with a factor-2 increase in pipelines deployed and maintained — CBLA is actually quite reasonable from an absolute latency perspective and superior from an operational complexity one.

Comparison with Recent Related Work

Table 5 compares CBLA to the four published 2026-era studies reviewed in Section 2 across five axes that have meaningful operational implications: (1) breadth of detection task(s) considered, (2) number of deployed pipelines/models, (3) strictness of zero-day/test thresholding, (4) class-disjoint realism of zero-day partitions, and (5) existence of component-wise ablation study/metrics. Tarawneh et al. [11]'s hybrid CNN-BiLSTM-Autoencoder detects zero-day and side-channel attacks but folds three datasets together into one training/dev/test split with no class-disjoint zero-day split and reports zero matched-parameter ablations (Table 1), severely undercutting claims of actual zero-day/general novel-attack transfer. Krishnan et al. [12] present an intuitive and elegant centroid-based thresholding scheme on Edge-IIoTset but cannot, by design, mark attacks as anomalous that are far from every centroid — precisely the regime in which CIC-IoT-2023's zero-day classes sit — and could use a single-pipeline system only by ditching reconstruction-based detection entirely. Egga et al. [13] empirically validate reconstruction-error scoring rules function on IoT network traffic but apply a heuristic threshold and include no classification head themselves, forcing users to attach their own classifier at deployment. Shakir and Ahmed [14] show their three-headed ensemble of autoencoder+LightGBM+IsolationForest achieves good zero-day performance, but triples deployment cost. CBLA offers all unified detection in a single model with 1.47 M parameters: ROC-optimal Youden-J thresholding, family-disjoint zero-days, AND matched-parameter ablation study.

Table 5. Comparison of CBLA with Recent 2026-Era Related Work

Aspect	Tarawneh et al. [11]	Krishnan et al. [12]	Egga et al. [13]	Shakir & Ahmed [14]	CBLA (Proposed)
Year	2026	2025	2026	2026	2026
Dataset(s)	CICIDS2023, TON_IoT, BoT-IoT	Edge-IIoTset	Mirai/BASHLITE (Kaggle)	TON_IoT	CIC-IoT-2023
Architecture	CNN + BiLSTM + AE	MHSA + BiLSTM + Centroid	Stand-alone Autoencoder	AE + LightGBM + IsoForest	CNN + BiLSTM + Attention + AE
Known-attack classification	Yes	Yes	No	Yes	Yes
Zero-day detection	Yes	Yes (centroid-mapped)	Yes (reconstruction)	Yes (ensemble vote)	Yes (reconstruction)
Deployment pipelines	1 model (multi-task)	1 model (no AE branch)	1 model + classifier required	3 models (ensemble)	1 model (unified)
Threshold strategy	Not specified	Centroid distance	Heuristic (fixed)	Ensemble voting	Youden-J ROC-optimal
Held-out zero-day validation	No	Partial	No	No	Yes (15% partition)
Class-disjoint zero-day split	No (dataset merging)	Yes (limited families)	Not applicable	Limited	Yes (4 distinct families)
Matched-parameter ablation	No	No	No	No	Yes (4 variants)
Reported zero-day AUC	Not reported	— (multi-class 71.09%)	— (binary acc. ~99%)	Not reported	0.972
Multi-seed evaluation	No	No	No	No	Yes (seeds 42, 123, 456)

CONCLUSIONS

We introduced CBLA, a single deep learning pipeline that conducts known-attack classification and zero-day detection jointly within one 1.47 M-parameter forward pass, to break down the operational silos that have hindered intrusion detection systems from being deployed at scale on resource-bounded IoT gateways. We highlight four main contributions. Firstly, our CNN–BiLSTM–Attention–Autoencoder model design mitigates dual-pipeline overhead by factorizing a shared encoder between softmax classification head and reconstruction decoder. Secondly, our zero-day detection threshold is tuned on a held-out validation set using the Youden-J statistic to maximize ROC optimality, avoiding the optimistic bias of percentile heuristic baselines. Thirdly, we introduce CIC-IoT-2023 evaluation with class disjoint

split, where 15 held-out classes encompass structurally-different attack types (e.g., reconnaissance, web injection, MITM, malware), whereas prior works relied on minor-variant splits. Lastly, our ablation study includes a matched-parameter four variant that isolates the contribution of each module, unveiling a systematic pretrained-generalization trade-off. CBLA experimentally attained a classification accuracy of 99.89% and zero-day detection with 0.972 AUC-ROC and 94.22% TPR. It is also the only single-pipeline system among those compared providing both functionalities. Finally, future directions include applying contrastive pretraining to close the gap in zero-day F1-score, federated learning extensions for inter-IoT gateway learning without sharing data, and evaluation on other IoT datasets such as TON_IoT and Edge-IIoTset.

REFERENCES

- [1] E. C. P. Neto, S. Dadkhah, R. Ferreira, A. Zohourian, R. Lu, and A. A. Ghorbani, "CICIoT2023: A real-time dataset and benchmark for large-scale attacks in IoT environment," *Sensors*, vol. 23, no. 13, art. no. 5941, Jun. 2023, doi: 10.3390/s23135941.
- [2] M. Antonakakis *et al.*, "Understanding the Mirai botnet," in *Proc. 26th USENIX Security Symp.*, Vancouver, BC, Canada, Aug. 2017, pp. 1093–1110.
- [3] M. A. Al-Garadi, A. Mohamed, A. K. Al-Ali, X. Du, I. Ali, and M. Guizani, "A survey of machine and deep learning methods for Internet of Things (IoT) security," *IEEE Commun. Surv. Tutor.*, vol. 22, no. 3, pp. 1646–1685, 3rd Quart., 2020, doi: 10.1109/COMST.2020.2988293.
- [4] A. Khraisat, I. Gondal, P. Vamplew, and J. Kamruzzaman, "Survey of intrusion detection systems: Techniques, datasets and challenges," *Cybersecurity*, vol. 2, no. 1, art. no. 20, Jul. 2019, doi: 10.1186/s42400-019-0038-7.
- [5] R. Ahmad, I. Alsmadi, W. Alhamdani, and L. Tawalbeh, "Zero-day attack detection: A systematic literature review," *Artif. Intell. Rev.*, vol. 56, no. 10, pp. 10733–10811, Oct. 2023, doi: 10.1007/s10462-023-10437-z.
- [6] H. Hindy, R. Atkinson, C. Tachtatzis, J.-N. Colin, E. Bayne, and X. Bellekens, "Utilising deep learning techniques for effective zero-day attack detection," *Electronics*, vol. 9, no. 10, art. no. 1684, Oct. 2020, doi: 10.3390/electronics9101684.
- [7] Y. Mirsky, T. Doitshman, Y. Elovici, and A. Shabtai, "Kitsune: An ensemble of autoencoders for online network intrusion detection," in *Proc. Netw. Distrib. Syst. Secur. Symp. (NDSS'18)*, San Diego, CA, USA, Feb. 2018, doi: 10.14722/ndss.2018.23204.
- [8] M. A. Ferrag, O. Friha, D. Hamouda, L. Maglaras, and H. Janicke, "Edge-IIoTset: A new comprehensive realistic cyber security dataset of IoT and IIoT applications for centralized and federated learning," *IEEE Access*, vol. 10, pp. 40281–40306, 2022, doi: 10.1109/ACCESS.2022.3165809.
- [9] N. Moustafa, "A new distributed architecture for evaluating AI-based security systems at the edge: Network TON_IoT datasets," *Sustain. Cities Soc.*, vol. 72, art. no. 102994, Sep. 2021, doi: 10.1016/j.scs.2021.102994.
- [10] S. I. Popoola, R. Ande, B. Adebisi, G. Gui, M. Hammoudeh, and O. Jogunola, "Federated deep learning for zero-day botnet attack detection in IoT-edge devices," *IEEE Internet Things J.*, vol. 9, no. 5, pp. 3930–3944, Mar. 2022, doi: 10.1109/JIOT.2021.3100755.

- [11] M. Tarawneh, D. Alrbiahat, S. Al-Eidi, and F. Alzyoud, "An integrated deep learning framework for detecting botnets, zero-day exploits, and side-channel attacks in network traffic," in *Proc. 2026 Int. Conf. Breakthroughs Artif. Intell. (BAI'26)*, K. Daimi and A. Alsadoon, Eds., *Lecture Notes in Networks and Systems*, vol. 1907. Cham, Switzerland: Springer, 2026, doi: 10.1007/978-3-032-21147-7_6.
- [12] D. Krishnan, S. Singh, and V. Sugumaran, "Explainable AI for zero-day attack detection in IoT networks using attention fusion model," *Discov. Internet Things*, vol. 5, art. no. 83, 2025, doi: 10.1007/s43926-025-00184-8.
- [13] B. H. Egga, A. S. Audu, G. O. Aimufua, M. Olalere, B. A. Ajayi, and I. T. Solomon, "Autoencoder-based model for detecting IoT network traffic anomalies," *Sci. World J.*, vol. 20, no. 4, pp. 1377–1383, 2026, doi: 10.4314/swj.v20i4.8.
- [14] H. A. Shakir and A. Y. Ahmed, "A hybrid anomaly detection framework integrating self-supervised learning and ensemble intelligence for securing IoT networks," *Int. J. Hum.-Comput. Stud.*, vol. 8, no. 1, pp. 1–11, 2026, doi: 10.31149/ijhcs.v8i1.5643.
- [15] J. F. Cevallos M., A. Rizzardi, S. Sicari, and A. Coen-Porisini, "NERO: NEural algorithmic reasoning for zeRO-day attack detection in the IoT: A hybrid approach," *Comput. Secur.*, vol. 142, art. no. 103898, Jul. 2024, doi: 10.1016/j.cose.2024.103898.
- [16] M. Soltani, B. Ousat, M. J. Siavoshani, and A. H. Jahangir, "An adaptable deep learning-based intrusion detection system to zero-day attacks," *J. Inf. Secur. Appl.*, vol. 76, art. no. 103516, Aug. 2023, doi: 10.1016/j.jisa.2023.103516.
- [17] B. Susilo, A. Muis, and R. F. Sari, "Intelligent intrusion detection system against various attacks based on a hybrid deep learning algorithm," *Sensors*, vol. 25, no. 2, art. no. 580, Jan. 2025, doi: 10.3390/s25020580.
- [18] Y.-W. Chen, J.-P. Sheu, Y.-C. Kuo, and N. Van Cuong, "Multi-class intrusion detection based on Transformer for IoT networks using CIC-IoT-2023 dataset," *Future Internet*, vol. 16, no. 8, art. no. 284, Aug. 2024, doi: 10.3390/fi16080284.
- [19] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," in *Proc. 31st Int. Conf. Neural Inf. Process. Syst. (NeurIPS)*, Long Beach, CA, USA, Dec. 2017, pp. 5998–6008.
- [20] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, Nov. 1997, doi: 10.1162/neco.1997.9.8.1735.
- [21] X. Huang, A. Khetan, M. Cvitkovic, and Z. Karnin, "TabTransformer: Tabular data modeling using contextual embeddings," *arXiv preprint*, arXiv:2012.06678, Dec. 2020, doi: 10.48550/arXiv.2012.06678.
- [22] F. T. Liu, K. M. Ting, and Z.-H. Zhou, "Isolation forest," in *Proc. 8th IEEE Int. Conf. Data Mining (ICDM)*, Pisa, Italy, Dec. 2008, pp. 413–422, doi: 10.1109/ICDM.2008.17.