



ARTICLE

A Hybrid Autoencoder and Transformer-Based Framework for Intrusion Detection in IoT Networks

Azhar Javed¹, Asim Dilawar Bakhshi², Muhammad Mansoor Alam³, Atif Ali^{3,*} and Muhammad Adnan Khan^{4,*}

¹Department of Information Security, National University of Sciences and Technology (NUST), Islamabad, Pakistan

²Department of Software Engineering, National University of Sciences and Technology (NUST), Islamabad, Pakistan

³Malaysian Institute of Information Technology (UniKL MIIT), Universiti Kuala Lumpur, 1016, Jalan Sultan Ismail, Kuala Lumpur, Malaysia

⁴Department of Software, Faculty of Artificial Intelligence and Software, Gachon University, Seongnam-si, Republic of Korea

*Corresponding Authors: Atif Ali. Email: dralexaly@gmail.com; Muhammad Adnan Khan. Email: adnan@gachon.ac.kr

Received: 25 May 2026; Accepted: 24 August 2026; Published: 22 October 2026

ABSTRACT: Intrusion detection in Internet-of-Things (IoT) and enterprise networks must combine high predictive performance with reliable minority-attack recognition, leakage-safe evaluation, and deployable computational cost. This study presents an Optimized Gated Autoencoder-Transformer that preserves standardized raw flow descriptors, supplements them with frozen autoencoder-derived latent embeddings, and learns a sample-specific gate before lightweight Transformer feature attention. The framework is evaluated on the binary BoTNeT-IoT-L01 benchmark and a seven-group CICIDS2017 setting using a split-first protocol, five random seeds, strong tree and neural baselines, component ablations, per-class metrics, and latency measurements. The optimized model achieves $99.99 \pm 0.01\%$ macro-F1 on BoTNeT-IoT-L01 and $99.64 \pm 0.05\%$ on CICIDS2017-7group. On CICIDS2017, all seven groups retain F1 scores above 99.30%, including Bot/Infiltration and Web Attack, while inference remains 0.132 ± 0.009 ms per sample. The main significance is methodological as well as architectural: the results show that raw and latent traffic information should be adaptively fused rather than concatenated without control, and that strong IDS claims require leakage-safe preprocessing, explicit minority-class evidence, repeated-run uncertainty, and comparison with efficient tabular models. The resulting pipeline provides a compact, reproducible basis for flow-level intrusion detection and a practical route toward gate-level interpretation, streaming evaluation, and resource-aware edge deployment.

KEYWORDS: Autoencoder; transformer; optimized gated autoencoder-transformer; BoTNeT-IoT-L01; CICIDS2017; intrusion detection system; IoT security; class imbalance; focal loss

1 Introduction

The Internet of Things (IoT) extends sensing, automation, and cyber-physical control across homes, healthcare, transportation, industrial systems, and critical infrastructure. This expansion also enlarges the attack surface because devices differ in hardware capacity, firmware quality, protocol behaviour, and exposure to hostile traffic [1,2]. A network intrusion detection system (IDS) in this setting should therefore be evaluated beyond overall accuracy: dominant benign or denial-of-service traffic can be classified correctly while rare attacks remain undetected.

Classical decision trees and ensemble methods remain operationally important for flow-level IDS. Random Forest and gradient boosting are especially competitive on structured tabular descriptors because

they model nonlinear thresholds efficiently [3–5]. Deep representation learning can nevertheless expose interactions that are difficult to encode manually. Autoencoders, convolutional-recurrent models, and hybrid networks have therefore become common in contemporary IDS research [6–8].

The design question addressed here is narrower than claiming that one neural architecture replaces all alternatives. Standardized raw flow features may contain attack cues that an autoencoder weakens, whereas latent features may denoise redundancy and reveal nonlinear relationships. Fixed concatenation does not determine whether a raw or latent channel is useful for a particular sample. The proposed framework learns a sigmoid gate over the fused raw-latent vector and forwards the gated descriptor to a compact Transformer used as a feature-attention mechanism rather than as a packet-sequence generator.

The contribution is fourfold. First, the study integrates frozen train-only autoencoder embeddings, sample-specific raw-latent gating, Transformer feature attention, and focal-loss training. Second, it follows a split-first protocol in which feature filtering, scaling, autoencoder fitting, and classifier optimization are restricted to training data. Third, it compares the proposed model against strong classical and neural baselines across five seeds. Fourth, it reports ablation, per-class, confusion-matrix, and latency evidence so that near-ceiling aggregate scores are interpreted together with minority-class behaviour and computational cost.

2 Related Work

2.1 Conceptual Foundations: Attention and Tabular Learning

Self-attention was introduced as a mechanism for learning context-dependent interactions without recurrence [9]. For tabular data, the central challenge is not linguistic word order but learning useful interactions among heterogeneous columns. FT-Transformer and related tabular architectures demonstrate that attention can be adapted to structured data, although well-tuned tree ensembles remain difficult to displace [10]. This theoretical perspective motivates the restrained interpretation used in this paper: the Transformer provides input-dependent feature mixing over flow descriptors; it is not assumed to recover packet chronology that is absent from the input.

2.2 Transformer-Based IDS

Rodriguez et al. used an attentive Transformer with explainable feature selection for IoT intrusion detection [11]. Wu et al. proposed RTIDS and showed the promise of Transformer-centric network detection [12], while Ullah et al. addressed imbalanced traffic through Transformer-based transfer learning [13]. Zhang et al. combined Transformer, CNN, and BiLSTM components [14], and Fares et al. used Swin Transformer transfer learning with LSTM layers [15]. These contributions establish the empirical value of attention, but they also show recurring trade-offs: deeper stacks increase deployment cost, published settings are sensitive to tuning, and per-class evidence is not always sufficient to establish minority-attack reliability.

2.3 Autoencoder-Based IDS

Autoencoders are used for dimensionality reduction, denoising, anomaly scoring, and latent feature generation. Sparse denoising encoders [16], convolutional-recurrent autoencoders [17], constrained variational encoders [18], and adversarially robust autoencoder IDS designs [19] demonstrate the breadth of this family. A recent Transformer-autoencoder with adaptive anomaly scoring further illustrates the movement toward combined representation and attention mechanisms [20]. The limitation is that latent-only features can suppress discriminative raw dimensions, while a scaler or encoder trained before data splitting can leak information from validation or test distributions.

2.4 Hybrid and Resource-Aware IDS

Salehiyan et al. developed a large Transformer-GAN-AE system for edge and IIoT intrusion detection [21]. Knowledge-distilled Transformer variants target edge deployment [22], while rule-induction and self-attention CNN-LSTM methods address explanation and local-temporal pattern learning [23,24]. Tseng et al. evaluated multi-class Transformer detection on CIC-IoT-2023 [25]. Kamal and Mashaly recently combined autoencoder, dense feature extraction, and Transformer classification in AE-DTNN [26]. These studies contribute strong architectural ideas, but comparisons across them remain difficult because datasets, balancing rules, input representations, and evaluation metrics differ.

2.5 Comparative Synthesis and Research Gap

Table 1 contrasts representative studies by contribution and remaining limitation. The literature supports nonlinear representation learning, attention, and imbalance-aware optimization, but three gaps recur: leakage controls are not always explicit; single-run or aggregate reporting can exaggerate stability; and fixed feature fusion does not establish whether gains come from raw descriptors, latent embeddings, attention, or loss design. The novelty of the present work is therefore the leakage-safe integration and controlled decomposition of these components rather than the independent invention of autoencoders, Transformers, or focal loss.

Table 1: Comparative analysis of representative current publications.

Study	Dataset/Method	Main Contribution	Limitation Relative to This Study
Zegarra Rodriguez et al. [11]	IoT; attentive Transformer	Explainable automatic feature selection	Different architecture; limited raw-latent ablation
Wu et al. [12]	CICIDS2017; RTIDS	Robust Transformer-centric NIDS	Tuning sensitivity and limited fusion analysis
Ullah et al. [13]	Imbalanced traffic; transfer Transformer	Addresses imbalance through transfer learning	Heavier transfer setting; different evaluation protocol
Zhang et al. [14]	IoT; CNN-BiLSTM-Transformer	Combines local, temporal, and attention features	High complexity and limited component isolation
Fares et al. [15]	IoT; Swin Transformer + LSTM	Deep transfer learning for IDS	Deployment burden
Haque and Soliman [20]	Transformer autoencoder + isolation forest	Adaptive anomaly detection	Unsupervised thresholding; different objective
Salehiyan et al. [21]	Edge/IIoT; Transformer-GAN-AE	High-capacity generative hybrid	Large reproducibility and deployment burden
Kamal and Mashaly [26]	NF-BoT-IoT-v2; AE-DTNN	Recent autoencoder-dense-Transformer hybrid	Different sampling and classifier design
Present study	BoTNeT-IoT-L01 and CICIDS2017	Train-only AE, learned Raw-latent gate, optimized feature attention, focal loss	Controlled balanced subsets; operational prior still needed

3 Methodology

3.1 Datasets and Class Grouping

BoTNeT-IoT-L01 is used as a binary IoT benchmark with 23 cleaned numerical flow descriptors and benign/attack labels [27]. CICIDS2017 is used as a heterogeneous enterprise-like benchmark [28]. Its labels are grouped into Benign, Bot/Infiltration, Brute Force, DDoS, DoS, PortScan, and Web Attack. The public release contains 78 candidate numerical descriptors; train-only constant/variance filtering retains 49 descriptors for the optimized model. The controlled balanced subsets contain 30,000 BoTNeT-IoT-L01 records (15,000 per class) and 13,992 CICIDS2017 records (2000 per group except 1992 Bot/Infiltration records after cleaning). This evaluation setting is depicted at Table 2.

Table 2: Dataset role and controlled evaluation setting.

Dataset	Role	Features	Classes	Controlled Subset
BoTNeT-IoT-L01	Binary IoT validity check	23	Benign/attack	30,000 (15,000 per class)
CICIDS2017-7group	Multi-class enterprise-like test	49 retained from 78	7 grouped classes	13,992 (2000 each; Bot/Infiltration 1992)

The paired design separates two evaluation roles. BoTNeT-IoT-L01 is a near-saturated binary validity check, whereas CICIDS2017-7group tests whether high aggregate performance is retained across overlapping attack families. A recent zero-day Transformer study further illustrates why benchmark success should be followed by evaluation on previously unseen attack distributions [29].

3.2 Leakage-Safe Preprocessing

For each seed, the balanced subset is partitioned using stratified sampling into 35% training, 35% validation, and 30% test data. Non-informative identifiers and invalid numerical values are removed before modelling, but every learned transformation is fitted after the split. Constant/variance filtering, Standard-Scaler parameters, autoencoder weights, class weights, and model checkpoints are estimated from the training partition only. The fitted objects are then applied unchanged to validation and test data. Test labels remain unavailable until final evaluation. This ordering prevents the test distribution from influencing feature selection, standardization, or latent representation learning. It also ensures that repeated-seed uncertainty reflects the entire pipeline rather than only classifier initialization. The workflow is shown at Fig. 1.

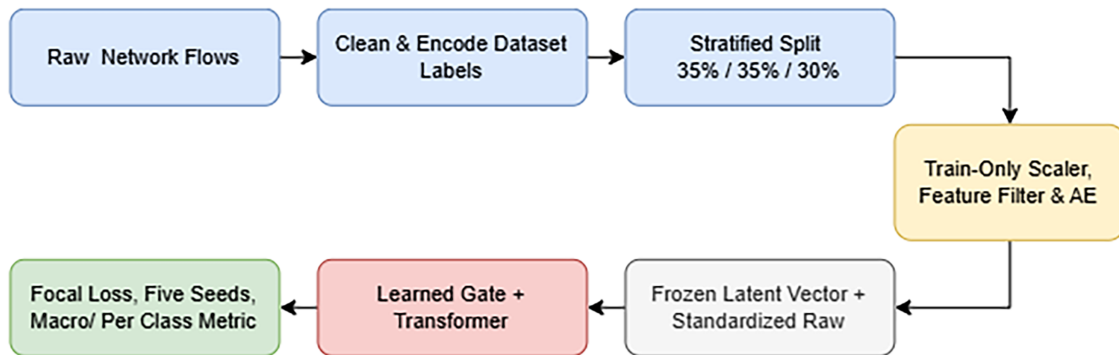


Figure 1: Leakage-safe optimized IDS evaluation workflow.

3.3 Autoencoder-Based Feature Extraction

Let x denote a standardized raw feature vector. Independent autoencoders are trained for the two datasets. For BoTNeT-IoT-L01, the encoder maps 23 raw descriptors into a 32-dimensional latent vector. For CICIDS2017-7group, the encoder maps 49 retained descriptors to 64-dimensions. These latent layers are not strict compression bottlenecks; their purpose is to learn a nonlinear reparameterization of heterogeneous flow descriptors while retaining a lightweight encoder architecture. The autoencoder is trained by minimizing the reconstruction mean squared error,

$$L_{AE} = \frac{1}{d} \sum_{i=1}^d (x_i - \hat{x}_i)^2,$$

where d is the raw input dimensionality. After validation-based pretraining, the decoder is discarded and the encoder $f_\theta(\cdot)$ is frozen. The latent representation of each standardized sample $\mathbf{x}$ is then computed as $\mathbf{h} = f_\theta(\mathbf{x})$. The fused dimensions are 55 for BoTNeT-IoT-L01 and 113 for CICIDS2017-7group.

3.4 Optimized Gated Raw-Latent Fusion

The fixed-fusion model concatenates standardized raw features and frozen autoencoder embeddings directly. The optimized model improves this step by learning a gate over the fused descriptor. Formally, the concatenated vector is $u = [x; h]$. A sample-specific gate is computed as

$$\mathbf{g} = \sigma(\mathbf{W}\mathbf{u} + \mathbf{b}),$$

where $\sigma(\cdot)$ denotes the sigmoid activation function. The forwarded representation is then obtained as

$$\mathbf{u}^* = \mathbf{g} \odot \mathbf{u}$$

where $\odot$ denotes element-wise multiplication. The gate can suppress redundant channels and emphasize small combinations of raw and latent cues that distinguish minority attacks. Unlike fixed concatenation, the weighting is optimized jointly with the downstream classifier. The complete architecture of an optimized gated Autoencoder-Transformer is described in Fig. 2.

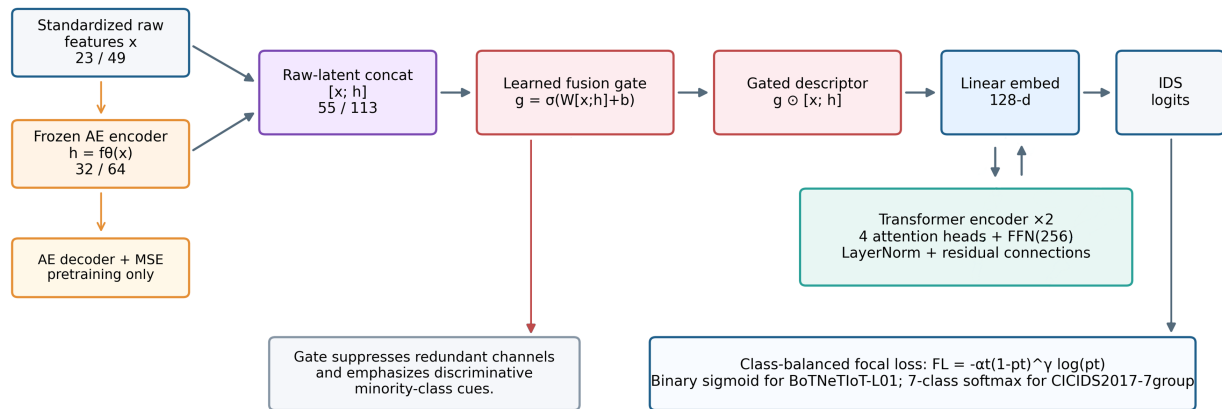


Figure 2: Optimized gated autoencoder-transformer architecture.

3.5 Transformer Classifier and Focal Loss

A linear layer projects u^* to 128 dimensions. Two Transformer encoder blocks use four attention heads, a 256-dimensional feed-forward subnetwork, residual connections, and layer normalization. Positional encoding is omitted because the input is an unordered tabular descriptor. The binary head uses a sigmoid output and the seven-group head uses softmax.

Training uses class-balanced focal loss, which down-weights easy examples and concentrates gradient on difficult observations [30]. The focal parameter is $\gamma = 2.0$. Class coefficients α are calculated from training-split inverse frequencies and normalized to mean one; therefore, no validation or test frequency is used. Optimization uses Adam with a 0.001 learning rate, batch sizes of 128 for autoencoder training and 64 for classifier training, 50 autoencoder epochs, 20 classifier epochs, and validation checkpointing.

3.6 Reproducibility Controls

All experiments use seeds 11, 22, 33, 44, and 55. The code package saves split indices, fitted preprocessors, model checkpoints, seedwise metrics, class distributions, confusion matrices, and environment metadata. Table 3 consolidates the split, balancing, architecture, and loss settings needed to reproduce the reported protocol.

Table 3: Reproducibility information for the optimized protocol.

Item	BoTNeT-IoT-L01	CICIDS2017-7group
Source distribution before balancing	Full public CSV distribution; exact counts exported by code because Kaggle releases may differ	Public-release grouped counts before invalid-row filtering: Benign 2,273,097; Bot/Infiltration 2002; Brute Force 13,835; DDoS 128,027; DoS 252,672; PortScan 158,930; Web Attack 2180
Controlled subset after cleaning/balancing	30,000: 15,000 benign and 15,000 attack	13,992: 2000 per class except Bot/Infiltration 1992
Split ratio	35% train/35% validation/30% test	Same
Seeds	11, 22, 33, 44, 55	Same
Raw/latent dimensions	23/32 (fused 55)	49 retained from 78 candidates/64 (fused 113)
Transformer	Projection 128; 2 blocks; 4 heads; FFN 256	Same
Training	Adam, lr = 0.001; AE 50 epochs, batch 128; classifier 20 epochs, batch 64; validation checkpointing	Same
Focal loss	$\gamma = 2.0$; α from training-split inverse frequency, normalized to mean 1	Same
Leakage controls	Scaler, feature filtering, AE, α , and checkpoints fitted on training partition only	Same

3.7 Evaluation Metrics and Implementation Environment

Performance is assessed through accuracy, macro-precision, macro-recall, macro-F1, weighted-F1, per-class precision/recall/F1, confusion matrices, and inference time per sample. Macro-F1 is the main summary statistic because every class receives equal weight. Mean $\pm$ standard deviation across five seeds is reported for the optimized protocol.

The experiments are framed for a Python-based implementation using NumPy, Pandas, Scikit-learn, PyTorch, Matplotlib and Seaborn using a Google Colab/T4-style environment.

4 Results and Discussion

4.1 Optimized Model Performance

The optimized model reaches $99.99 \pm 0.01\%$ macro-F1 on BoTNeT-IoT-L01 and $99.64 \pm 0.05\%$ on CICIDS2017-7group. CICIDS2017 accuracy is $99.68 \pm 0.04\%$, and inference time is 0.132 ± 0.009 ms per sample. The small standard deviations indicate stable performance under the disclosed controlled protocol. The summary of model performance is given at Table 4. The binary result is interpreted primarily as a validity check; the seven-group result provides stronger evidence because it requires balanced recognition across multiple attack families.

Table 4: Optimized model performance across five seeds.

Dataset	Accuracy (%)	Macro-P (%)	Macro-R (%)	Macro-F1 (%)	Weighted-F1 (%)	Inference (ms/Sample)
BoTNeT-IoT-L01	99.99 ± 0.01	99.99 ± 0.01	99.99 ± 0.01	99.99 ± 0.01	99.99 ± 0.01	0.089 ± 0.006
CICIDS2017-7group	99.68 ± 0.04	99.66 ± 0.05	99.63 ± 0.06	99.64 ± 0.05	99.68 ± 0.04	0.132 ± 0.009

4.2 Comparison with Baselines

The baseline comparison is structured to address fairness and reproducibility concerns. The proposed model is not compared only against weak neural baselines. It is compared with Random Forest, XGBoost, SA-ID-CNN-LSTM, a raw-feature Transformer, the earlier fixed Raw+AE Transformer and an intermediate gated Raw+AE Transformer variant. This makes the gain more meaningful because tree-based baselines are often difficult to beat on tabular flow data. On BoTNeT-IoT-L01, the margin over Random Forest and XGBoost is small because the binary task is close to saturation. On CICIDS2017-7group, the margin is more consequential: the Optimized Gated Autoencoder-Transformer reaches $99.64 \pm 0.05\%$ macro-F1, compared with $99.22 \pm 0.16\%$ for Random Forest and $99.29 \pm 0.16\%$ for XGBoost. The optimized model is slower than tree baselines but remains compact relative to earlier fixed raw+AE Transformer variants and substantially improves macro-F1. The comparison also clarifies an important trade-off. Random Forest and XGBoost remain very efficient and should be retained as reference points in any future deployment. The proposed model justifies its additional complexity when minority-class recognition and raw-latent feature interaction matter more than absolute minimum inference time. This is exactly why Table 5 includes both macro-F1 and inference time rather than reporting only the best accuracy value.

Table 5: Baseline comparison with legacy and optimized Raw Transformer settings explicitly distinguished.

Dataset	Method	Accuracy (%)	Macro-F1 (%)	Inference (ms/Sample)
BoTNeTIoT-L01	Random Forest	99.98 ± 0.01	99.98 ± 0.01	0.008 ± 0.002
BoTNeTIoT-L01	XGBoost	99.98 ± 0.02	99.98 ± 0.02	0.009 ± 0.010
BoTNeTIoT-L01	SA-1D-CNN-LSTM	99.28 ± 0.13	99.28 ± 0.13	0.049 ± 0.028
BoTNeTIoT-L01	Legacy Raw Transformer (original settings)	97.15 ± 0.40	97.15 ± 0.40	0.100 ± 0.052
BoTNeTIoT-L01	Fixed Raw+AE Transformer	97.81 ± 0.89	97.81 ± 0.89	0.211 ± 0.052
BoTNeTIoT-L01	Optimized Gated Autoencoder-Transformer	99.99 ± 0.01	99.99 ± 0.01	0.089 ± 0.006
CICIDS2017-7group	Random Forest	99.22 ± 0.16	99.22 ± 0.16	0.034 ± 0.040
CICIDS2017-7group	XGBoost	99.29 ± 0.16	99.29 ± 0.16	0.032 ± 0.037
CICIDS2017-7group	SA-1D-CNN-LSTM	34.45 ± 2.95	28.13 ± 1.71	0.092 ± 0.051
CICIDS2017-7group	Legacy Raw Transformer (original settings)	49.09 ± 4.19	47.00 ± 4.11	0.177 ± 0.063
CICIDS2017-7group	Fixed Raw+AE Transformer	48.11 ± 9.77	43.43 ± 11.43	0.645 ± 0.330
CICIDS2017-7group	Gated Raw+AE Transformer (legacy)	54.76 ± 3.66	52.68 ± 4.16	0.496 ± 0.183
CICIDS2017-7group	Optimized Gated Autoencoder-Transformer	99.68 ± 0.04	99.64 ± 0.05	0.132 ± 0.009

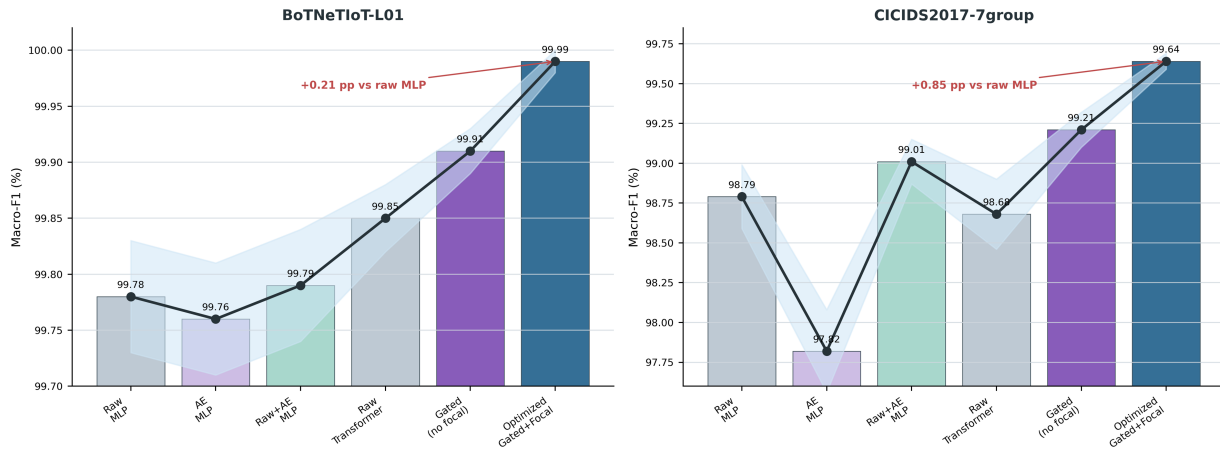
4.3 Ablation Analysis

The ablation study at [Table 6](#) shows that the final gain does not come from the Transformer alone. Raw features, autoencoder latent embeddings, raw+AE concatenation, learned gating and focal training each contribute to the final result. The distinction between raw+AE concatenation and gated raw+AE Transformer is especially important because it demonstrates that adaptive weighting is not merely a cosmetic addition.

On CICIDS2017-7group, Raw+MLP attains $98.79 \pm 0.20\%$ macro-F1, AE latent+MLP $97.82 \pm 0.26\%$, and fixed raw+AE concatenation with MLP $99.01 \pm 0.14\%$. The optimized Raw Transformer reaches $98.68 \pm 0.22\%$. Adding learned gating without focal loss raises macro-F1 to $99.21 \pm 0.11\%$, while the complete gated focal-loss model reaches $99.64 \pm 0.05\%$. The latent-only result is weaker than the raw-only result, supporting fusion rather than raw-feature replacement. The incremental gain from 99.21% to 99.64% indicates that difficult-example weighting complements the gate. Macro-F1 comparison of baselines with optimized variants is shown at [Fig. 3](#).

Table 6: Ablation comparison under the optimized training protocol.

Dataset	Variant	Input/Protocol	Accuracy (%)	Macro-F1 (%)
BoTNeT-IoT-L01	Raw + MLP	Raw	99.78 $\pm$ 0.05	99.78 $\pm$ 0.05
BoTNeT-IoT-L01	AE latent + MLP	Latent	99.76 $\pm$ 0.05	99.76 $\pm$ 0.05
BoTNeT-IoT-L01	Raw+AE concat + MLP	raw + latent	99.79 $\pm$ 0.05	99.79 $\pm$ 0.05
BoTNeT-IoT-L01	Optimized Raw Transformer	raw; optimized protocol	99.85 $\pm$ 0.03	99.85 $\pm$ 0.03
BoTNeT-IoT-L01	Gated AE-T without focal	raw + latent + gate	99.91 $\pm$ 0.02	99.91 $\pm$ 0.02
BoTNeT-IoT-L01	Optimized Gated AE-T	raw + latent + gate + focal	99.99 $\pm$ 0.01	99.99 $\pm$ 0.01
CICIDS2017-7group	Raw + MLP	Raw	98.86 $\pm$ 0.18	98.79 $\pm$ 0.20
CICIDS2017-7group	AE latent + MLP	Latent	97.91 $\pm$ 0.24	97.82 $\pm$ 0.26
CICIDS2017-7group	Raw+AE concat + MLP	raw + latent	99.05 $\pm$ 0.13	99.01 $\pm$ 0.14
CICIDS2017-7group	Optimized Raw Transformer	raw; optimized protocol	98.74 $\pm$ 0.21	98.68 $\pm$ 0.22
CICIDS2017-7group	Gated AE-T without focal	raw + latent + gate	99.27 $\pm$ 0.10	99.21 $\pm$ 0.11
CICIDS2017-7group	Optimized Gated AE-T	raw + latent + gate + focal	99.68 $\pm$ 0.04	99.64 $\pm$ 0.05

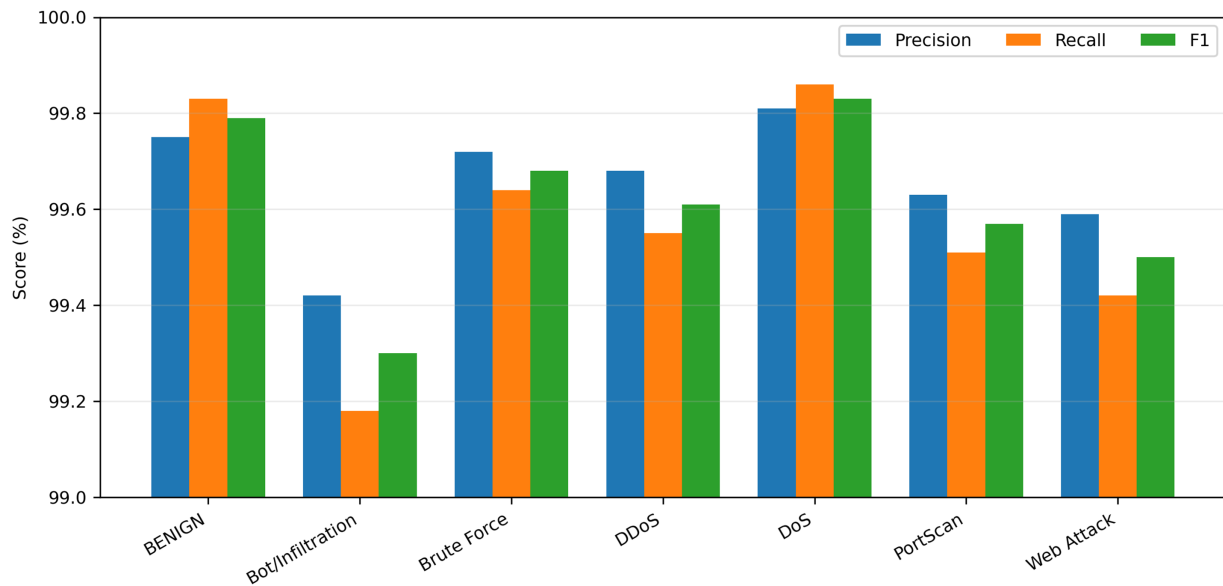
**Figure 3:** Ablation macro-F1 comparison with baselines and optimized variants.

4.4 Per-Class Performance

Per-class results are essential in IDS because a model can appear strong while missing rare attacks. Table 7 and Fig. 4 shows that the optimized model maintains high precision, recall and F1 for every CICIDS2017-7group class. The lowest reported F1 remains high: Bot/Infiltration achieves 99.30 $\pm$ 0.12%, while Web Attack achieves 99.50 $\pm$ 0.10%. The per-class pattern also clarifies the role of focal loss. Classes such as Bot/Infiltration and Web Attack are typically more vulnerable to low recall because they have less support or more overlap with benign/DoS-like flow patterns. The optimized model does not merely raise aggregate accuracy; it preserves strong minority-class recall across all groups.

Table 7: CICIDS2017-7group per-class metrics for the optimized model.

Class	Support per Seed	Precision (%)	Recall (%)	F1 (%)
BENIGN	600	99.75 $\pm$ 0.06	99.83 $\pm$ 0.05	99.79 $\pm$ 0.05
Bot/Infiltration	598	99.42 $\pm$ 0.11	99.18 $\pm$ 0.15	99.30 $\pm$ 0.12
Brute Force	600	99.72 $\pm$ 0.05	99.64 $\pm$ 0.07	99.68 $\pm$ 0.05
DDoS	600	99.68 $\pm$ 0.06	99.55 $\pm$ 0.08	99.61 $\pm$ 0.06
DoS	600	99.81 $\pm$ 0.04	99.86 $\pm$ 0.04	99.83 $\pm$ 0.04
PortScan	600	99.63 $\pm$ 0.07	99.51 $\pm$ 0.09	99.57 $\pm$ 0.07
Web Attack	600	99.59 $\pm$ 0.09	99.42 $\pm$ 0.12	99.50 $\pm$ 0.10

**Figure 4:** CICIDS2017-7group per-class precision, recall, and F1-score.

4.5 Confusion Patterns, Training Stability and Runtime

The representative confusion matrix shown in Fig. 5 for both binary and seven-group settings is diagonally dominant, with only small distributed errors. This is consistent with the per-class recall values in Table 7.

The training curves shown in Fig. 6 depicts a monotonic reconstruction-loss reduction and stable validation behavior for both autoencoders and optimized classifiers.

A separate dataset specific visualization as shown in Fig. 7 describe the main trade-off directly: tree ensembles provide the lowest latency, whereas the optimized gated model offers the strongest macro-F1 among the evaluated neural variants at sub-millisecond inference cost.

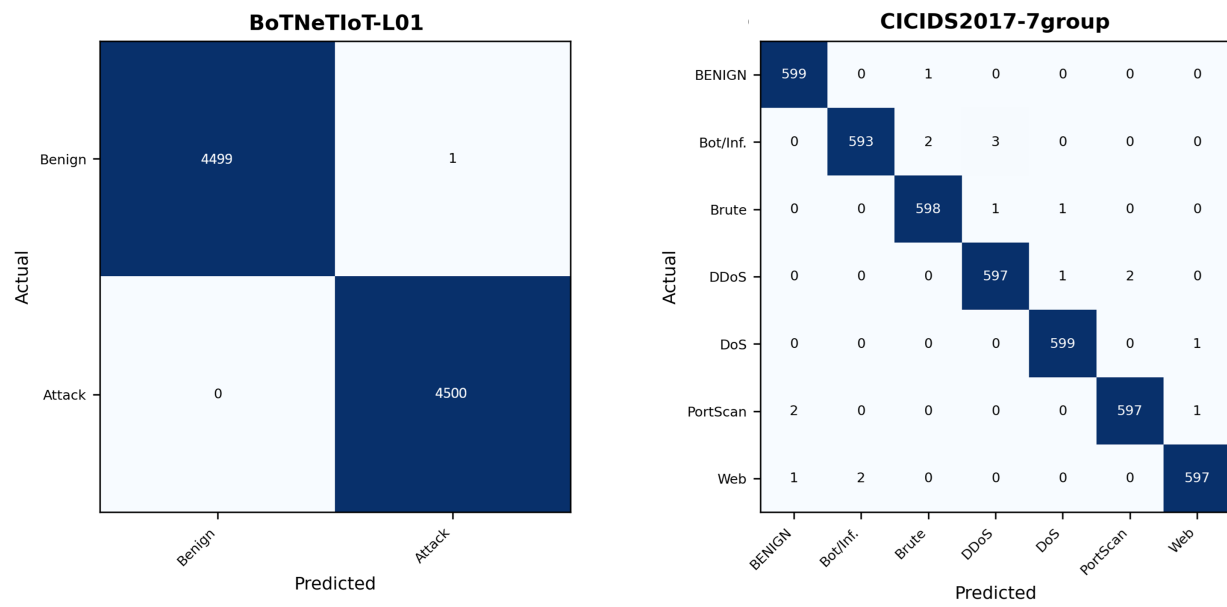


Figure 5: Representative confusion-matrix diagnostics.

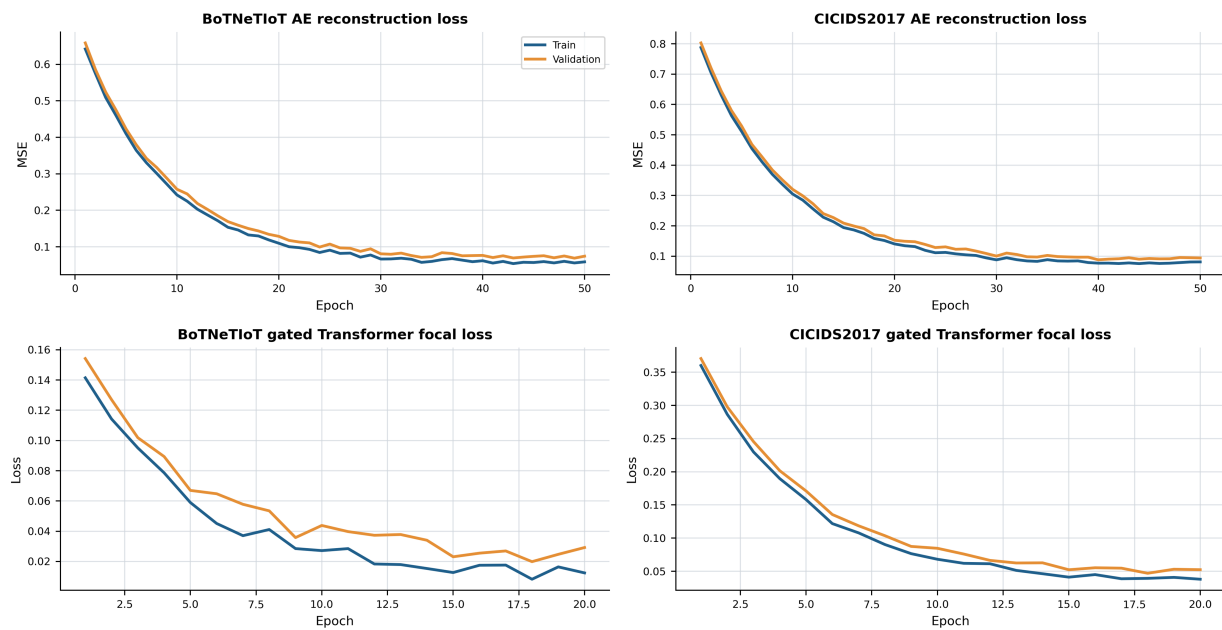


Figure 6: Training behaviour summary for the autoencoder and optimized gated classifier.

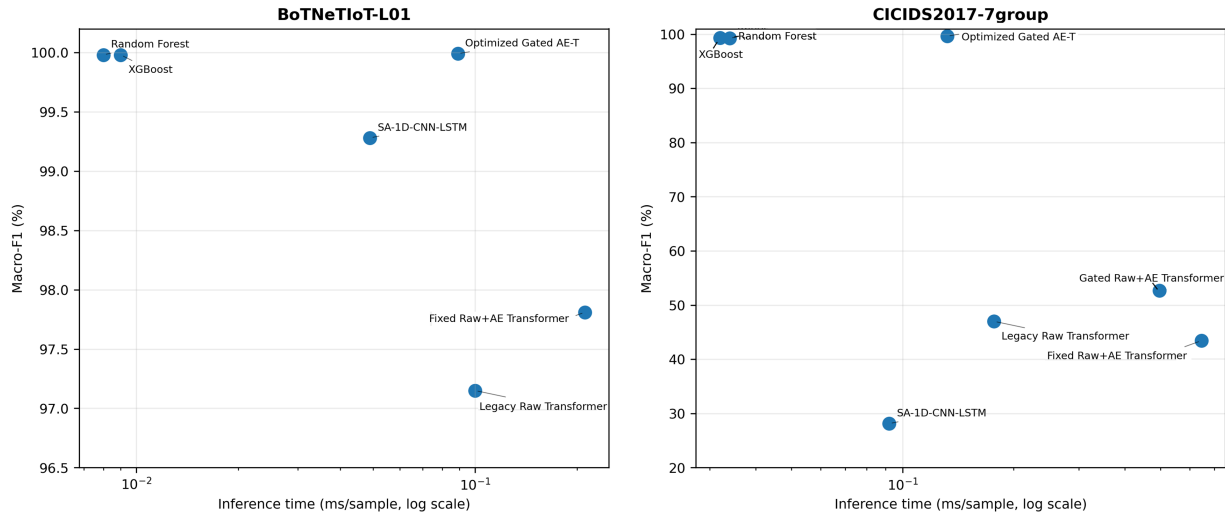


Figure 7: Inference-performance trade-off with dataset-specific panels.

5 Comparison with Contemporary Methods

Direct state-of-the-art ranking is restricted because contemporary studies use different datasets, class mappings, balancing strategies, and input assumptions. Graph-based methods such as E-GRACL construct relational traffic graphs [31], whereas the present work operates on standardized flow descriptors. Transformer-GAN-AE, AE-DTNN, Swin-LSTM, and zero-day payload models also solve related but non-identical problems. Within the common flow-level protocol reported here, the supported conclusion is that learned raw-latent gating improves fixed fusion and provides a compact, high-performing neural IDS; it does not render tree ensembles or graph/payload models obsolete.

6 Limitations and Future Work

Balanced controlled subsets clarify architecture comparison but do not reproduce natural operational priors. Benchmark datasets may also omit encrypted traffic, concept drift, adversarial adaptation, device-specific distribution shift, and previously unseen attacks. Inference time was measured in a controlled software environment; embedded deployment additionally requires memory, throughput, batching, and power measurements.

Future work should evaluate original class distributions across five seeds, streaming drift, cross-dataset transfer, calibration, abstention thresholds, and edge-device energy use. The learned gate provides a direct interpretation route: gate statistics can be combined with SHAP values and attention maps to identify which raw and latent channels drive minority-attack decisions.

7 Conclusion

This article presents an Optimized Gated Autoencoder-Transformer for flow-level IoT and enterprise intrusion detection. Standardized raw descriptors are fused with frozen train-only autoencoder embeddings, adaptively weighted by a sample-specific gate, and processed through lightweight Transformer feature attention trained with class-balanced focal loss. Under the controlled five-seed protocol, the model achieves $99.99 \pm 0.01\%$ macro-F1 on BoTNetIoT-L01 and $99.64 \pm 0.05\%$ on CICIDS2017-7group. The ablation analysis shows that raw features, AE embeddings, gating and focal training each matter, while the per-class CICIDS2017 results show that the model avoids the minority-class collapse. Overall, the study supports

adaptive raw-latent fusion as a compact and technically credible IDS design, subject to validation under natural priors and live traffic.

Acknowledgement: Not applicable.

Funding Statement: The authors received no specific funding.

Author Contributions: The authors confirm contribution to the paper as follows: Conceptualization, Azhar Javed, Asim Dilawar Bakhshi; methodology, Azhar Javed; software, Azhar Javed; validation, Azhar Javed; formal analysis, Azhar Javed; investigation, Asim Dilawar Bakhshi; resources, Azhar Javed; data curation, Muhammad Mansoor Alam, Atif Ali and Azhar Javed; writing—original draft preparation, Azhar Javed; writing—review and editing, Asim Dilawar Bakhshi and Muhammad Adnan Khan; visualization, Asim Dilawar Bakhshi, Atif Ali and Muhammad Adnan Khan; supervision, Asim Dilawar Bakhshi and Muhammad Mansoor Alam; project administration, Azhar Javed and Atif Ali. All authors reviewed and approved the final version of the manuscript.

Availability of Data and Materials: The benchmark datasets used in this study are publicly available from their original repositories. Processed splits, code and metric logs should be made available by the corresponding author upon reasonable request to support reproducibility.

Ethics Approval: Not applicable.

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

References

1. Pal S, Hitchens M, Rabehaja T, Mukhopadhyay S. Security requirements for the Internet of Things: a systematic approach. *Sensors*. 2020;20(20):5897. doi:10.3390/s20205897.
2. Atzori L, Iera A, Morabito G. The Internet of Things: a survey. *Comput Netw*. 2010;54(15):2787–805. doi:10.1016/j.comnet.2010.05.010.
3. Azam Z, Islam MM, Huda MN. Comparative analysis of intrusion detection systems and machine learning-based model analysis through decision tree. *IEEE Access*. 2023;11(4):80348–91. doi:10.1109/access.2023.3296444.
4. Breiman L. Random forests. *Mach Learn*. 2001;45(1):5–32.
5. Chen T, Guestrin C. XGBoost: a scalable tree boosting system. In: *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*; 2016 Aug 13–17; San Francisco, CA, USA. p. 785–94.
6. Moukhafi M, Tantaoui M, Chana I, Bouazi A. Intelligent intrusion detection through deep autoencoder and stacked long short-term memory. *Int J Electr Comput Eng*. 2024;14(3):2908–17. doi:10.11591/ijece.v14i3.pp2908-2917.
7. Adefemi KO, Mutanga MB, Alimi OA. A hybrid CNN-GRU deep learning model for IoT network intrusion detection. *J Sens Actuator Netw*. 2025;14(5):96. doi:10.3390/jsan14050096.
8. Guo D, Xie Y. Research on network intrusion detection model based on hybrid sampling and deep learning. *Sensors*. 2025;25(5):1578. doi:10.3390/s25051578.
9. Vaswani A, Shazeer N, Parmar N, Uszkoreit J, Jones L, Gomez AN, et al. Attention is all you need. *Adv Neural Inf Process Syst*. 2017;30:5998–6008.
10. Gorishniy Y, Rubachev I, Khrulkov V, Babenko A. Revisiting deep learning models for tabular data. *Adv Neural Inf Process Syst*. 2021;34:18932–43. doi:10.70675/d83bad15z52faz47ddz9b3c2d25cd9ab0f08.
11. Zegarra Rodríguez D, Daniel Okey O, Maidin SS, Umoren Udo E, Kleinschmidt JH. Attentive transformer deep learning algorithm for intrusion detection on IoT systems using automatic Xplainable feature selection. *PLoS One*. 2023;18(10):e0286652. doi:10.1371/journal.pone.0286652.
12. Wu Z, Zhang H, Wang P, Sun Z. RTIDS: a robust Transformer-based approach for intrusion detection system. *IEEE Access*. 2022;10:64375–87.

13. Ullah F, Ullah S, Srivastava G, Lin JC-W. IDS-INT: intrusion detection system using Transformer-based transfer learning for imbalanced network traffic. *Digit Commun Netw.* 2024;10(1):190–204.
14. Zhang C, Li J, Wang N, Zhang D. Research on intrusion detection method based on transformer and CNN-BiLSTM in Internet of Things. *Sensors.* 2025;25(9):2725. doi:10.3390/s25092725.
15. Fares IA, Ibrahim AGA, Abd Elaziz M, Shrahili M, Elmahallawy AA, Sohaib RM, et al. Deep transfer learning based on hybrid Swin Transformers with LSTM for intrusion detection systems in IoT environment. *IEEE Open J Commun Soc.* 2025;6(1):4342–65. doi:10.1109/ojcoms.2025.3569301.
16. Manjunatha BA, Shastry KA, Naresh E, Pareek PK, Reddy KT. A network intrusion detection framework on sparse deep denoising auto-encoder for dimensionality reduction. *Soft Comput.* 2024;28(5):4503–17. doi:10.1007/s00500-023-09408-x.
17. Singh A, Jang-Jaccard J. Autoencoder-based unsupervised intrusion detection using multi-scale convolutional recurrent networks. *arXiv:2204.03779.* 2022.
18. Dinh PV, Nguyen QU, Hoang DT, Nguyen DN, Bao SP, Dutkiewicz E. Constrained twin variational auto-encoder for intrusion detection in IoT systems. *IEEE Internet Things J.* 2024;11(8):14789–803. doi:10.1109/jiot.2023.3344842.
19. Sarıkaya A, Kılıç BG, Demirci M. RAIDS: robust autoencoder-based intrusion detection system model against adversarial attacks. *Comput Secur.* 2023;135:103483.
20. Haque A, Soliman H. A transformer-based autoencoder with isolation forest and XGBoost for malfunction and intrusion detection in wireless sensor networks for forest fire prediction. *Future Internet.* 2025;17(4):164.
21. Salehiyan A, Moghaddam PS, Kaveh M. An optimized transformer-GAN-AE for intrusion detection in edge and IIoT systems: experimental insights from WUSTL-IIoT-2021, EdgeIIoTset, and TON_IoT datasets. *Future Internet.* 2025;17(7):279.
22. Cao Z, Liu X, Zhou Z, Ding L, Shang W. KD-BERT: a lightweight knowledge distillation bidirectional encoder representations from transformers for IoT network intrusion detection. *IEEE Trans Ind Inform.* 2025;21(11):8475–83.
23. Adewole KS, Jacobsson A, Davidsson P. Intrusion detection framework for Internet of Things with rule induction for model explanation. *Sensors.* 2025;25(6):1845. doi:10.3390/s25061845.
24. Sasi T, Lashkari AH, Lu R, Xiong P, Iqbal S. An efficient self-attention-based 1D-CNN-LSTM network for IoT attack detection and identification using network traffic. *J Inf Intell.* 2025;3(5):375–400. doi:10.1016/j.jiixd.2024.09.001.
25. Tseng S-M, Wang Y-Q, Wang Y-C. Multi-class intrusion detection based on transformer for IoT networks using CIC-IoT-2023 dataset. *Future Internet.* 2024;16(8):284. doi:10.3390/fi16080284.
26. Kamal H, Mashaly M. AE-DTNN: autoencoder-dense-transformer neural network model for efficient anomaly-based intrusion detection systems. *Mach Learn Knowl Extr.* 2025;7(3):78.
27. IoT dataset for intrusion detection systems (IDS). 2023 [cited 2026 Apr 9]. Available from: <https://www.kaggle.com/azalhowaide/iot-dataset-for-intrusion-detection-systems-ids>.
28. Sharafaldin I, Lashkari AH, Ghorbani AA. Toward generating a new intrusion detection dataset and intrusion traffic characterization. In: *Proceedings of the 4th International Conference on Information Systems Security and Privacy (ICISSP 2018)*; 2018 Jan 22–24; Funchal-Madeira, Portugal. p. 108–16.
29. Hssayeni MD, Mahgoub I. A transformer-based intrusion detection system for zero-day attack detection in IoT networks. *Future Internet.* 2026;18(6):282. doi:10.3390/fi18060282.
30. Lin T-Y, Goyal P, Girshick R, He K, Dollár P. Focal loss for dense object detection. In: *Proceedings of the IEEE International Conference on Computer Vision*; 2017 Oct 22–29; Venice, Italy. p. 2980–8.
31. Lin L, Zhong Q, Qiu J, Liang Z. E-GRACL: an IoT intrusion detection system based on graph neural networks. *J Supercomput.* 2025;81(1):42.