ISSN: 2690-909X

Research Article

Advances in Neurology and Neuroscience

A Lightweight, Label-Free Deep Autoencoder for Real-Time Anomaly Detection in Resource-Constrained IoT Network Traffic

Samuel Yao Sebuabe[1*], Stephen Kofi Dotse[2], Harriet K. O. Lamptey[2], Ezekiel Okoe Annan[1], Martin Doe[1], Rebecca Amponsah[1], Ezra Tablaje Baabotin[2] and Kwame Assa-Agyei[3]

[1]*Department of Computer Science, Valley View University, Accra, Ghana*

[2]*Department of Information Technology, University of Professional Studies, Accra, Ghana*

[3]*School of Science and Technology, Wisconsin International University College, Accra, Ghana*

[4]*School of Science and Technology, Nottingham Trent University, Nottingham, United Kingdom*

***Corresponding Author**
Samuel Yao Sebuabe, Department of Computer Science, Valley View University, Accra, Ghana.

Submitted: 2026, Jun 12; **Accepted:** 2026, Jul 10; **Published:** 2026, Jul 22

Citation: Sebuabe, S. Y., Dotse, S. K., Lamptey, H. K. O., Annan, E. O., Doe, M., et. al. (2026). A Lightweight, Label-Free Deep Autoencoder for Real-Time Anomaly Detection in Resource-Constrained IoT Network Traffic. *Adv Neur Sci, 9*(3), 01-16.

Abstract

The proliferation of Internet of Things (IoT) devices has dramatically widened the cyber-attack surface of modern networks, while the limited computational resources and protocol heterogeneity of these devices render conventional signature-based and supervised intrusion detection systems impractical at scale. This paper presents a lightweight, label-free anomaly-detection framework built on a deep autoencoder and optimised explicitly for deployment at the IoT edge. The model is trained exclusively on benign network-flow records, so anomalies are identified through reconstruction error rather than through labelled attack signatures, removing the dependence on curated attack labels that constrain supervised detectors. Principal Component Analysis is embedded in the pipeline to compress the feature space and shrink the model footprint, and the anomaly decision boundary is derived analytically from the 95th percentile of the benign reconstruction-error distribution rather than tuned by trial and error. The framework is evaluated across two complementary public benchmarks, IoT-23 and ToN_IoT, and is benchmarked against Random Forest and Support Vector Machine classifiers. The autoencoder attains 98.3% and 97.1% precision with false-positive rates of 4.8% and 5.1% on the two datasets, respectively, together with ROC-AUC values above 0.90, while occupying only 0.06 MB and sustaining an inference latency of 0.156 ms per flow, a footprint roughly 46 times smaller than the Random Forest baseline. Although supervised models achieve higher recall when labelled data are available, the proposed approach detects previously unseen and zero-day patterns without retraining and with a negligible memory budget, making it a practical building block for real-time, on-device intrusion detection in constrained IoT environments. The study also characterises the recall-false-alarm trade-off governed by the threshold, providing actionable guidance for operators who must balance detection sensitivity against alert fatigue.

Keywords: Anomaly Detection, Internet Of Things, Deep Autoencoder, Intrusion Detection, Unsupervised Learning, Edge Computing

1. Introduction

1.1. Background and Motivation

The Internet of Things (IoT) has grown into one of the largest technological ecosystems in the world, with billions of interconnected devices embedded across healthcare, transportation, agriculture, manufacturing, smart cities, and national-security infrastructure [1,2]. This expansion has introduced complex security challenges stemming from weak

authentication mechanisms, constrained processing resources, outdated firmware, and inconsistent implementation of security standards across device manufacturers [2,3]. Because many IoT endpoints are deployed with minimal computational capability, they are highly susceptible to botnet recruitment, distributed denial-of-service (DDoS) campaigns, spoofing, ransomware, and unauthorised access to critical systems [1,4,5]. These weaknesses pose persistent threats to both personal and national infrastructure. Traditional intrusion detection systems (IDS) face fundamental limitations when applied to IoT environments. Signature-based detectors match traffic against a database of known attack patterns, which restricts their ability to recognise novel, modified, or zero-day threats that do not conform to previously catalogued signatures [1,5,6]. Even anomaly-based detectors struggle with the dynamic, heterogeneous, and high-dimensional nature of IoT traffic, which is characterised by diverse communication protocols, inconsistent feature distributions, and complex temporal dependencies [4,7,8]. Supervised machine-learning detectors, meanwhile, depend on large labelled datasets, yet IoT traffic corpora are typically scarce, imbalanced, noisy, and unable to capture the full range of attack variants observed in real deployments [1,5,9]. Collectively, these issues limit the practicality of conventional detection methods and motivate more adaptive, label-efficient solutions.

Recent advances in deep learning offer a promising alternative. Deep autoencoders have emerged as an effective unsupervised technique because they learn compact representations of normal network behaviour and flag deviations through reconstruction error, without requiring labelled attack samples [4,7,9]. By modelling the distribution of benign traffic rather than memorising attack fingerprints, autoencoders can capture subtle non-linear patterns that statistical or rule-based methods overlook, and they generalise to emerging attacks more gracefully than classical classifiers [1,5,8]. These properties make them strong candidates for large-scale intrusion detection in highly dynamic IoT settings. Despite this promise, several barriers still limit the adoption of autoencoder-based detectors in operational IoT systems. Achieving consistent generalisation across heterogeneous device types and network protocols is difficult because traffic patterns vary widely between devices and application domains [4,8,9]. The pronounced class imbalance of IoT datasets, in which benign traffic typically dwarfs attack traffic, inflates false-positive rates and erodes operator trust [1,5,7]. Crucially, deep models are often computationally heavy, whereas many IoT devices operate under strict energy and memory budgets that preclude complex security algorithms at the edge [3,4,9]. Most prior studies also report only offline evaluations and rarely quantify the model footprint or inference latency that would determine real-world deployability.

1.2. Problem Statement and Research Gap

The central problem addressed in this work is therefore twofold. First, IoT operators need a detector that can identify novel and zero-day attacks without depending on labelled attack data that may be unavailable or quickly outdated. Second, that detector must be small and fast enough to run on gateways or edge nodes whose memory and compute are severely limited. Existing autoencoder studies tend to optimise a single dataset for detection accuracy while leaving generalisation across datasets, principled threshold selection, and on-device efficiency under-examined [7–9]. The research gap, then, is the absence of a deep-autoencoder pipeline that is simultaneously (i) label-free, (ii) validated across more than one benchmark, (iii) equipped with an analytically grounded decision threshold, and (iv) explicitly profiled for edge deployment.

1.3. Contributions and Novelty

The novelty of this work lies not in any single component but in their principled integration into one deployable pipeline whose every design choice is justified against edge constraints, and in the joint satisfaction of four properties that prior studies address only in isolation. Concretely, this paper makes the following contributions:

(a) A label-free deep-autoencoder pipeline that couples PCA-based feature compression with a compact 32-168-16-32 architecture, engineered specifically to minimise model size and inference latency for edge deployment while learning only from benign traffic.

(b) An analytically calibrated, percentile-based anomaly threshold (the 95th percentile of the benign reconstructionerror distribution) that prioritises low-false-alarm operation, accompanied by an explicit characterisation of the recallfalse-positive trade-off, so that the operating point is reproducible rather than hand-tuned.

(c) A cross-dataset evaluation on IoT-23 and ToN_IoT that, alongside Random Forest and Support Vector Machine baselines, quantifies the accuracy-versus-deployability trade-off between supervised and unsupervised detectors under identical preprocessing.

(d) A computational-efficiency analysis, reporting model size and per-flow inference latency, that demonstrates feasibility for gateway and edge deployment — an aspect frequently omitted in prior offline-only studies.

1.4. Organization of the Paper

The remainder of the paper is organised as follows. Section 2 reviews IoT security threats, intrusion detection paradigms, and autoencoder-based anomaly detection, and positions the present study against prior work. Section 3 describes the proposed methodology, including the threat model, datasets, preprocessing, feature engineering, autoencoder design, threshold calibration, baselines, and evaluation protocol. Section 4 reports the experimental results, and Section 5 discusses their implications and limitations. Section 6 concludes and outlines directions for future work.

2. Related Work

The rapid proliferation of the IoT has reshaped modern information systems by connecting billions of devices across healthcare, transportation, agriculture, smart cities, and industrial automation [2,1]. These systems exchange diverse traffic, each protocol with its own rules, data rates, and communication patterns. While the IoT has improved efficiency and automation, it has

also intensified cybersecurity risk through weak authentication, insufficient processing power, irregular firmware updates, and a lack of standardised security practice among manufacturers [3–5]. Conventional detection approaches are inadequate for these dynamic and emerging conditions, which motivates more sophisticated and adaptive anomaly detection. This section reviews the relevant literature on IoT architecture, threats, intrusion detection, machine-learning and deep-learning methods, and autoencoder-based detection, and consolidates the gaps that justify the proposed framework.

2.1. IoT Architecture and Traffic Characteristics

IoT systems are commonly organised into three layers: perception, network, and application [2,1]. The perception layer comprises the sensors and embedded devices that collect data; the network layer carries traffic over lightweight protocols such as MQTT, CoAP, HTTP, Zigbee, and 6LoWPAN; and the application layer delivers services and analytics. In contrast to conventional enterprise networks, IoT networks operate under tight constraints defined by limited bandwidth, low power consumption, and intermittent connectivity [3,7]. The breadth of device types and communication modes makes IoT traffic highly heterogeneous, producing disparate feature distributions and irregular packet intervals [4,7]. IoT traffic also exhibits non-linear temporal dependencies and high-dimensional feature spaces, which reduce the effectiveness of legacy statistical detectors [6,7]. This variability complicates the construction of reliable baselines for anomaly detection [9].

2.2. Cybersecurity Threats in IoT Environments

The growth of IoT devices has greatly expanded the attack surface of contemporary networks. IoT systems are frequently targeted by DDoS attacks, botnet malware, spoofing, ransomware, and unauthorised access [1,5,4]. Botnets such as Mirai demonstrated the destructive impact of compromised devices marshalled to launch large-scale DDoS attacks against critical infrastructure [1]. Such attacks typically manifest as anomalous traffic spikes, unusual connection attempts, and irregular packet distributions. DDoS in particular remains one of the most prevalent IoT threats, flooding gateways and cloud servers with illegitimate traffic [4]. Signature-based methods perform poorly against evolving DDoS variants because they depend on previously recognised patterns [5,6], while spoofing and unauthorised access exploit weak verification and misconfigured devices, leading to data breaches and system compromise [3]. The continuous evolution and diversity of these attacks call for adaptive detectors capable of recognising both established and emerging behaviour.

2.3. Intrusion Detection Systems in IoT

Intrusion detection systems are broadly divided into signature-based and anomaly-based categories [1]. Signature-based systems rely on known attack patterns; although effective against catalogued threats, they cannot detect zero-day or novel variants [5,6]. Given the heterogeneity of IoT traffic, exclusive reliance on signatures is a significant limitation. Anomaly-based systems instead model normal behaviour and flag departures from it [7]. They are more adaptable but tend to produce higher falsepositive rates because IoT traffic fluctuates [4,8]. Hybrid schemes that combine both paradigms have been proposed, yet balancing detection accuracy against computational efficiency remains an open challenge [1].

2.4. Machine-Learning Approaches for IoT Anomaly Detection

Machine learning has been widely applied to IoT intrusion detection to improve adaptability and accuracy [1,10,11]. Supervised algorithms such as Support Vector Machines, Random Forest, k-Nearest Neighbours, and Decision Trees outperform rule-based systems in classification [5,6,12], but they require large, well-labelled datasets that are often scarce and imbalanced in IoT contexts [13,8] and may generalise poorly to attack patterns absent from training data [4]. Unsupervised methods have therefore attracted growing interest because they operate without labelled attack data [7]. Clustering and dimensionality-reduction techniques can isolate anomalies within normal traffic, making them effective for zero-day detection [8], although they can struggle with the high-dimensional, non-linear interactions typical of IoT traffic [1].

2.5. Deep Learning and Autoencoders

Deep-learning models handle high-dimensional, non-linear network traffic well [4,7]. Architectures including Deep Neural Networks, Convolutional Neural Networks, Recurrent Neural Networks, and Long Short-Term Memory networks have been used to capture temporal and spatial relationships in IoT data [1,10]. These models provide superior feature extraction and stronger generalisation to complex attacks [9], but their computational cost complicates deployment in resource-limited settings [3], which has driven interest in lightweight architectures that balance accuracy and efficiency [8]. Autoencoders are unsupervised neural networks that learn compressed latent representations of their input [9]. An encoder maps input features into a lower-dimensional latent space and a decoder reconstructs the original input; the reconstruction error, the discrepancy between input and output, serves as the anomaly score [7]. Deep autoencoders with several hidden layers have proven effective at identifying abnormal IoT traffic by learning intricate non-linear representations [4,8]. Sparse autoencoders use regularisation to heighten anomaly sensitivity, denoising autoencoders improve robustness by reconstructing corrupted inputs [9, 13], and variational autoencoders model probabilistic distributions of traffic to enhance detection [4]. Empirically, autoencoders surpass conventional classifiers in zero-day settings because they model normal behaviour rather than memorising attack fingerprints [1,8]. Threshold selection nonetheless remains a key difficulty, since poor calibration produces excessive false positives or false negatives [9,7].

2.6. Benchmark Datasets and Evaluation Metrics

Several benchmark datasets are widely used in IoT anomaly detection, including IoT-23, ToN_IoT, Bot-IoT, and UNSWNB15 [1,14]. These corpora contain a range of attack scenarios and benign samples, but class imbalance and limited protocol diversity remain significant concerns [5,8]. Performance is commonly assessed with accuracy, precision, recall, F1-score, and ROC-AUC [7,1]. Because IoT traffic distributions are skewed, F1-score and recall are considered more reliable indicators of detection performance

than overall accuracy [4].

2.7. Research Gap and Positioning

Despite considerable progress, the literature exhibits persistent gaps. Many autoencoder-based models generalise poorly across IoT protocols such as CoAP and MQTT, and across device categories [15,4]. Dataset imbalance frequently inflates false-positive rates and undermines reliability [5,1]. Most studies report offline evaluations rather than real-time deployment assessments, which limits understanding of practical scalability [7], and computational constraints continue to impede deep architectures at the network edge [3]. Table 1 positions the present study against representative prior work along the dimensions that matter most for operational IoT IDS. To our knowledge, few studies jointly satisfy label-free operation, cross-dataset validation, principled threshold calibration, and an explicit edge-efficiency profile; this combination defines the contribution of the present work.

3. Methodology

This section presents a quantitative, experimental methodology for developing and evaluating the proposed deepautoencoder detector. Experimental designs are standard in intrusion detection research because they support systematic training, testing, validation, and comparison of computational models under controlled conditions [1,10, 7]. The workflow proceeds from dataset acquisition to preprocessing, feature engineering, model construction, and training on benign traffic, reconstruction-error thresholding, comparative evaluation against supervised baselines, and finally a scalability assessment, ensuring scientific rigour and reproducibility [10,1].

3.1. System Overview

Figure 1 illustrates the end-to-end pipeline. Raw IoT traffic is collected and preprocessed through cleaning, categorical encoding, and normalisation. Flow-based features are extracted and compressed with PCA, then passed to the deep autoencoder. The encoder projects the input into a low-dimensional latent space, and the decoder reconstructs it; the per-sample reconstruction error is compared against a calibrated threshold, and any sample exceeding the threshold is labelled anomalous.

3.2. Threat Model

The detector operates at the network layer, inspecting packet- and flow-level metadata (for example, packet counts, byte rates, flow duration, inter-arrival times, and TCP flags) observed at routers, gateways, or network monitors. It does not inspect encrypted payloads or devicespecific sensor readings. The adversary is assumed capable of launching volumetric and behavioural attacks, DDoS, botnet command-and-control, port scanning, injection, and backdoor activity that perturb flow-level statistics away from benign baselines. The defender observes only benign traffic during training and must detect deviations at inference without prior knowledge of specific attack signatures, reflecting realistic conditions in which labelled attack data are scarce or unavailable.

Study	Label-free	Multi-dataset	Principled threshold	Edge-efficiency profile
Ferrag et al.	No	Yes	n/a	No
Alaghbari et al.	Yes	No	Partial	No
Liu et al.	Yes	No	Partial	Partial
Chen et al.	No	No	n/a	No
Alrayes et al.	Yes	No	No	No
This work	Yes	Yes (IoT-23 + ToN_IoT)	Yes (95th percentile)	Yes (size + latency)

Table 1: Positioning of the Proposed Framework Against Representative Prior Work [1,7-9,13].

3.3. Datasets and Justification

The credibility of anomaly detection research depends heavily on the realism, diversity, and quality of its datasets [1,14]. Many early IoT datasets were criticised for limited protocol diversity, unrealistic traffic distributions, and severe class imbalance [5,8]. To mitigate these concerns and improve external validity, this study uses two widely adopted public benchmarks, IoT-23 and ToN_IoT, which are extensively used in recent empirical IoT anomaly detection research [1,4,7]. Both contain realistic networkflow records captured from IoT devices under benign and malicious conditions, including botnet activity, DDoS, injection, and port scanning [1,5]. Using two datasets with different benign/malicious ratios additionally allows the label-free generalisation of the detector to be tested rather than assumed.

3.4. Data Preprocessing

IoT traffic data are typically high-dimensional, noisy, incomplete, and heterogeneous in format, so preprocessing is essential for stable neural-network training and reliable convergence [4,6,8,13]. The pipeline begins with data cleaning: duplicate records are removed to eliminate sampling bias, and missing values are handled through median imputation for numeric fields and mode imputation for categorical fields, consistent with established practice [1,10]. Categorical features such as protocol identifiers and connection states are converted to numeric form through label encoding. Because neural networks are sensitive to feature magnitude, Min-Max normalisation scales every feature to the range, which improves gradient-descent stability and accelerates convergence [8,7]. Finally, all labels are mapped to a binary scheme, with benign as 0 and any attack class as 1 to suit the anomaly detection formulation. Class imbalance, a persistent challenge in IoT IDS, is addressed structurally by training the autoencoder only on benign samples so that the model learns normal behaviour and treats departures from it as anomalies; mixed benign and malicious samples are reserved for evaluation [1,4,5,9].

3.5. Feature Engineering and Dimensionality Reduction

Effective feature engineering improves detection accuracy by emphasising discriminative traffic characteristics [6,1]. This study derives flow-based statistical features, packet count, byte rate, flow duration, inter-arrival-time statistics, connection frequency, and TCP-flag distributions that are widely recognised as robust descriptors of IoT network behaviour [10,4]. Principal Component Analysis (PCA) is then applied, retaining 95% of the variance, to reduce redundancy and computational overhead [8,7]. Compressing the feature space lowers memory consumption and accelerates inference, directly addressing deployment on resource-constrained devices [3,1]. A feature-contribution analysis derived from the PCA component loadings identifies the attributes that most influence reconstruction error, answering the research question concerning which IoT traffic features are most relevant to anomaly detection.

3.6. Deep Autoencoder Design

The detector is a fully connected deep autoencoder comprising symmetric encoder and decoder branches. The encoder maps the PCA-reduced input through hidden layers of 32 and 16 units to an 8-unit latent bottleneck; the decoder mirrors this with 16- and 32-unit layers before an output layer that matches the input dimensionality. Rectified Linear Unit (ReLU) activations are used in the hidden layers for their efficiency and resistance to the vanishing-gradient problem, while the output layer uses a sigmoid activation consistent with the feature scaling. The compact bottleneck forces the network to learn a parsimonious representation of benign behaviour, which both improves anomaly contrast and keeps the parameter count and therefore the model footprint small [1,4].

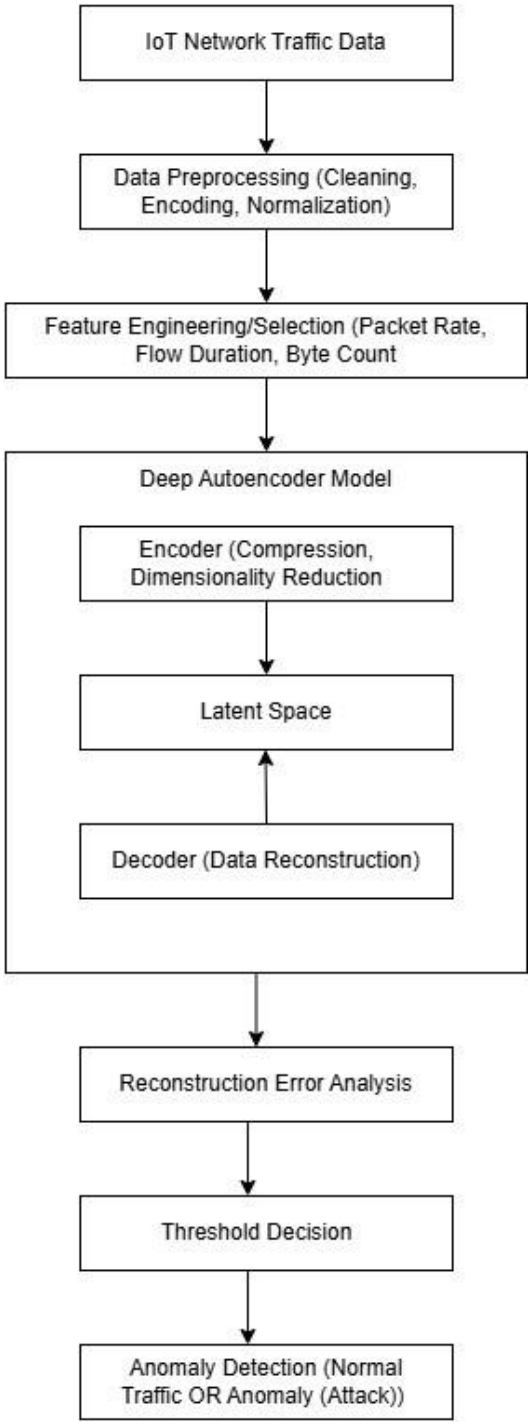

Figure 1: Conceptual framework for IoT anomaly detection using a deep autoencoder.

Benign traffic is used to learn a compact latent representation; reconstruction error above the calibrated threshold marks an anomaly.

Training minimises the Mean Squared Error (MSE) between input and reconstruction, which quantifies reconstruction deviation precisely [15]. Weights are updated with the Adam optimiser (learning rate 0.001) for its adaptive step sizes and computational efficiency [9]. Dropout and L2 weight decay regularise the network to curb overfitting and improve generalisation across traffic distributions, and early stopping (patience of five epochs, monitoring validation loss) restores the best weights and prevents over-training [1,4]. For an input vector x of dimension d and its reconstruction ˆx, the per-sample reconstruction error is

$$(x) = -\sum (x_i - \hat{x}_i)^2, \quad i = 1,\ldots,d \quad 1\ d$$

$$e. \qquad (1)^{\,d}\ i{=}1$$

3.7. Anomaly threshold calibration

After training on benign traffic, the reconstruction error is computed for each test instance. Rather than tuning the decision boundary heuristically, the anomaly threshold τ is set analytically to the 95th percentile of the benign trainingerror distribution:

$$\tau = P95\ e(x) : x \in X\text{benign,}^{\text{train}}). \quad (2)$$

Any test instance with e(x) > τ is classified as anomalous. This data-driven rule ensures that the great majority of benign traffic is classified correctly while keeping false alarms low, and it makes the threshold reproducible across datasets. Because raising τ trades recall for fewer false positives, Section 4.4 reports the full trade-off curve so that operators can adjust τ to their tolerance for alert fatigue [9,7].

3.8. Baseline Models

The autoencoder is compared against two supervised classifiers that are standard baselines in IoT IDS research: a Support Vector Machine (RBF kernel) and a Random Forest [6,10,1]. Unlike the autoencoder, the baselines are trained on a labelled mixture of benign and malicious samples. This comparison tests the hypothesis that an unsupervised autoencoder offers superior detection of zero-day and previously unseen attacks relative to supervised classifiers, while exposing the accuracy cost of dispensing with labels.

3.9. Evaluation Metrics

Because IoT datasets are skewed, accuracy alone can be misleading [8,4]. Evaluation, therefore, reports precision, recall, F1-score, ROC-AUC, and the false-positive rate (FPR). Recall measures the ability to catch malicious traffic, while F1-score balances sensitivity and precision [7,1]. The metrics are defined from the confusion-matrix counts of true positives (TP), true negatives (TN), false positives (FP), and false negatives (FN):

Accuracy =_____. (3)

TP+TN +FP+FN

TP

Precision =,

TP+FP

(4)

TP

Recall =.

TP+FN

Precision·Recall

$F_1 = 2\cdot$ _____,

Precision+Recall

(5)

FP

FPR = .

FP+TN

3.10. Implementation and Complexity

The model is implemented in Python with TensorFlow/Keras, and the baselines use Scikit-learn. Random seeds are fixed to support reproducibility, consistent with best practice in empirical cybersecurity research [10,1]. Inference complexity for the autoencoder is linear in the number of network parameters: for a fully connected network with layer widths n0,n1,...,nℓ, a single forward pass costs $O(\sum_k n_k n_{k-1})$, which for the compact 32-16-8-16-32 topology is small enough to support per-flow scoring at the edge. Section 4.7 reports the measured model size and inference latency that follow from this design.

4. Results and Analysis

This section reports the experimental results obtained with the methodology of Section 3. Evaluation covers preprocessing outcomes, feature contribution, training convergence, reconstruction-error and threshold behaviour, detection performance with confusion matrices, comparison against supervised baselines, and an efficiency profile for edge feasibility. The IoT-23 and ToN_IoT benchmarks provide realistic network-flow records under benign and malicious conditions, and performance is reported with the standard metrics defined above [1,7,4].

4.1. Experimental Setup and Preprocessing

Preprocessing followed the steps in Section 3.4: duplicates were removed to eliminate sampling bias, missing values were imputed statistically, categorical fields were label-encoded, and Min-Max normalisation scaled every feature to, which markedly improved gradient-descent stability. Labels were mapped to a binary scheme, with benign traffic encoded as 0 and all attack types, including botnet and DDoS, encoded as 1, in line with the anomaly detection formulation [1,16,8]. Both datasets exhibit pronounced class imbalance, with malicious traffic dominating. In IoT-23, benign traffic constitutes roughly 11.39% of samples against 88.61% malicious; in ToN_IoT, the split is 23.69% benign and 76.31% malicious. Despite the imbalance, the absolute number of benign samples is large enough to train the autoencoder effectively. Training exclusively on benign traffic lets the model learn normal behaviour and flag anomalies by reconstruction error during evaluation on mixed data. The higher benign proportion of ToN_IoT provides a useful second operating point for assessing generalisation, and the overall protocol mirrors realistic deployments where labelled attack data are limited.

Parameter	Value
Datasets	IoT-23, ToN_IoT
Training data	Benign traffic only (label = 0)
Test data	Mixed (benign and attack samples)
Model	Deep autoencoder (unsupervised)
Architecture	32-16-8-16-32 (ReLU; sigmoid output)
Optimiser/loss	Adam (lr = 0.001) / Mean Squared Error
Batch size/epochs	256 / 30 with early stopping (patience 5)
Evaluation metrics	Accuracy, Precision, Recall, F1, ROC-AUC, FPR

Table 2: Experimental Configuration.

4.2. Feature Contribution

Feature engineering focused on flow-based statistics, packet count, byte rate, flow duration, TCP flags, and inter-arrival time, which together provide a robust description of network behaviour. PCA reduced dimensionality and memory consumption while preserving 95% of the variance. The contribution analysis (Table 4) shows that all retained features influence reconstruction error, with packet count, byte rate, and flow duration the most influential, and inter-arrival time and TCP flags providing additional discriminative value; packet count also acts as an indirect proxy for connection frequency. These results corroborate findings that flow-based features are highly discriminative for IoT anomaly detection [10,4]. The prominence of packet count (0.89), byte rate (0.85), and flow duration (0.82) is consistent with prior reports identifying these attributes as strong indicators of anomalous behaviour, and supports the design of lightweight, efficient detectors that rely on a small, well-chosen feature set.

4.3. Training Convergence

The autoencoder was trained on benign traffic with the configuration in Table 2. Table 5 and Figure 2 summarise the training and validation loss over epochs. Both curves converge rapidly within the first few epochs and stabilise at approximately 0.004, with a negligible gap between training and validation loss, evidence that the model does not overfit and that the chosen features are highly representative of normal traffic. The early plateau indicates that additional epochs add little, so early stopping can be applied to save computation. This stable convergence is consistent with prior deep-autoencoder studies, and the reduction from 0.067 to roughly 0.004 demonstrates efficient learning of representative traffic patterns [7].

4.4. Reconstruction Error and Threshold Selection

Figure 3 shows that benign samples cluster at very low reconstruction error, confirming that the autoencoder reconstructs normal traffic accurately, whereas attack samples exhibit substantially higher error, reflecting their departure from learned behaviour. The clear separation validates the discriminative power of reconstruction error. A threshold of 0.00567, the 95th percentile of the benign training-error distribution, was selected; for readability, it is rounded to 0.006 in Figure 3 and Table 7. The trade-off this threshold governs is quantified in Table 7. A lower threshold (0.004) maximises recall but raises the false-positive rate, misclassifying more benign traffic as attacks; a higher threshold (0.008) suppresses false positives at the cost of recall, letting more attacks pass undetected. The percentile-based choice provides a balanced operating point that favours low false alarms as a priority in operational IDS, where alert fatigue is costly while retaining strong detection. The large gap between benign (0.0035) and attack (0.0650) mean errors confirms that the model has learned normal behaviour well, in agreement with prior reports that reconstruction error is an effective anomaly indicator [9].

4.5. Detection Performance

Anomaly detection was performed by thresholding the reconstruction error; instances above τ were labelled anomalous. Table 8 reports the detection metrics on both datasets. Precision is high on both datasets (98.3% and 97.1%), indicating that the overwhelming majority of flagged anomalies are genuinely malicious, while the low falsepositive rates (4.8% and 5.1%) confirm that the detector raises few spurious alarms, a critical property for real-world IoT IDS. Recall is comparatively lower (80.4% and 82.5%), meaning that some attacks evade detection; this is the expected behaviour of an anomaly-based detector calibrated for low false alarms, and it is directly controllable through τ as shown in Table 7. The modest variation between datasets reflects their differing distributions and attack mixes. The precision values compare favourably with prior

Dataset	Total samples	Benign (%)	Malicious (%)	Attack types
IoT-23	3,581,028	11.39	88.61	Port scanning, Okiru botnet, DDoS, C&C
ToN_IoT	211,043	23.69	76.31	DDoS, injection, backdoor

Table 3: Dataset Distribution After Preprocessing.

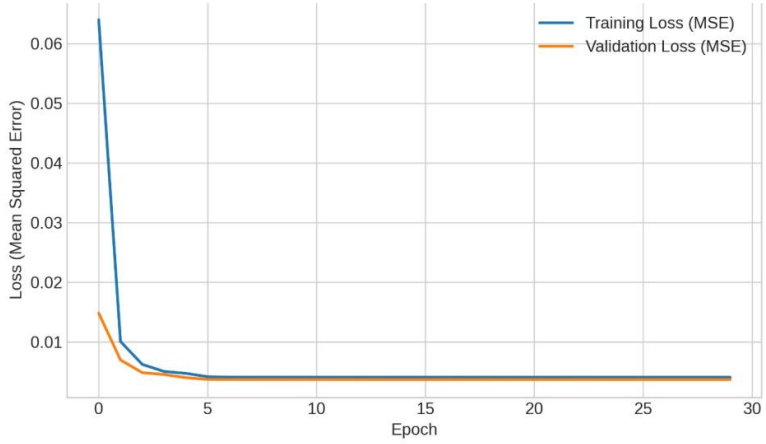

Figure 2: Training versus validation loss. Loss falls sharply within the first three epochs and stabilises near 0.004; the close agreement between the two curves indicates effective generalisation without overfitting.

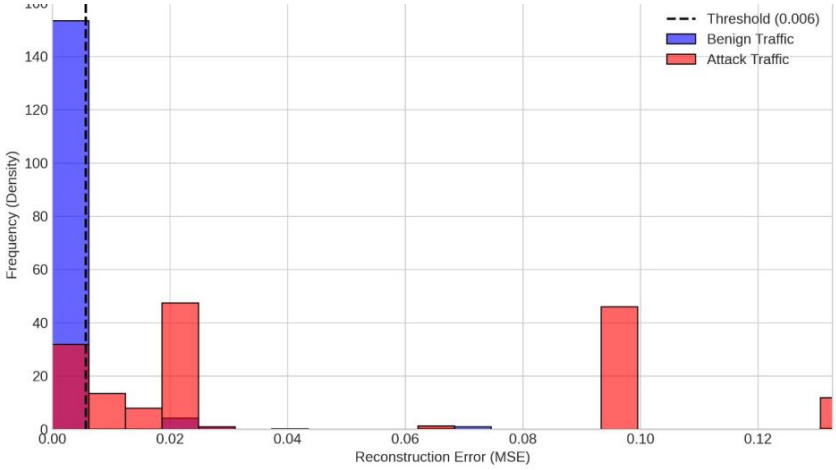

Figure 3: Reconstruction-error distributions for benign and attack traffic. Benign samples concentrate at low error while attack samples spread to much higher values; the dashed line marks the calibrated threshold. The minimal overlap explains the low false-positive rate.

deep-learning IDS reporting precision above 90% [1,4]. Tables 9 and 10 give the confusion matrices.

The matrices confirm high precision with moderate recall: most flagged anomalies are correct, but a portion of attacks is missed, consistent with an anomaly detector that prioritises minimising false positives over capturing every intrusion.

4.6. Comparison with Supervised Baselines

To contextualise the results, the autoencoder was benchmarked against Random Forest and SVM classifiers, which historically outperform rule-based systems but depend on labelled data. Figs. 4 and 5 and Tables 11 and 12 summarise the comparison.

Random Forest achieves the highest ROC-AUC (≈0.99) because its supervised training lets it learn explicit deci-sion boundaries from labelled data, while the SVM and the autoencoder both reach ≈0.92. The autoencoder's slightly lower score reflects its unsupervised nature: it relies solely on benign patterns and never observes attack labels. Crucially, however, Random Forest's dependence on labelled data limits its usefulness where new attack signatures are un-

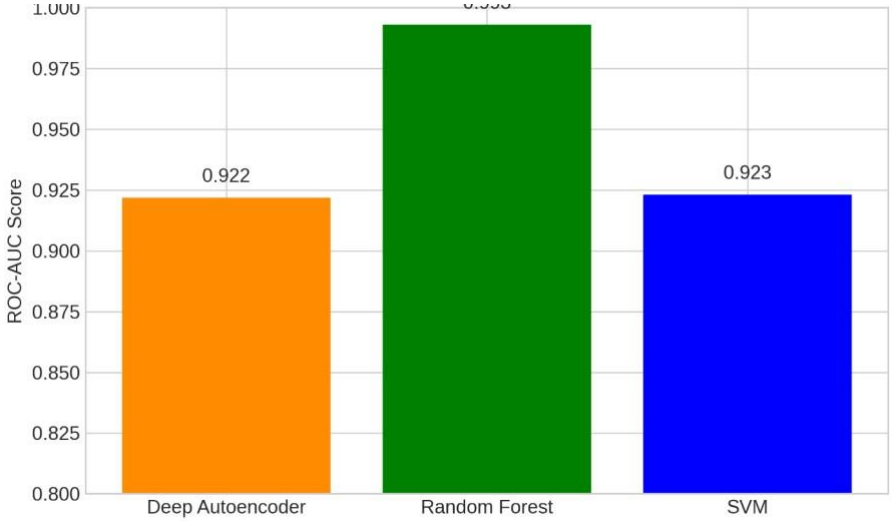

Figure 4: ROC-AUC comparison across models. Random Forest attains the highest score (≈0.99); the SVM and the proposed autoencoder follow closely (≈0.92), all well above random performance.

Feature	Importance score
Packet count (flow-based)	0.89
Byte rate (bytes per second)	0.85
Flow duration (time length)	0.82
Inter-arrival time (packet timing)	0.79
TCP flags (control indicators)	0.75

Table 4: Feature contribution derived from PCA component loadings (relative scores in [0, 1]; higher is more influential).

Epoch	Training loss (MSE)	Validation loss (MSE)	
1	0.067	0.015	
2	0.007	0.005	
3	0.0045	0.0042	
5	0.0042	0.0040	
10	0.0041	0.0040	
20	0.0040	0.0040	
30	0.0040	0.0040	

Table 5: Training and validation loss (MSE) by epoch

Traffic type	Mean reconstruction (MSE)	error
Benign	0.0035	
Attack	0.0650	

Table 6 Average reconstruction error by traffic type

Threshold	False-positive rate	Recall
0.004	0.08	0.99
0.006	0.05	0.97
0.008	0.02	0.82

Table 7: Threshold selection and the recall versus false-positive tradeoff

Metric	IoT-23	ToN_IoT
Accuracy (%)	83.6	85.2
Precision (%)	98.3	97.1
Recall (%)	80.4	82.5
F1-score (%)	88.4	89.2
ROC-AUC (%)	91.9	90.8
False-positive rate (%)	4.8	5.1

Table 8: Detection performance of the proposed autoencoder

	Predicted normal	Predicted attack
Actual normal	389,000	19,000
Actual attack	623,000	2,550,028

Table 9: Confusion matrix — IoT-23

	Predicted normal	Predicted attack
Actual normal	47,580	2,400
Actual attack	31,600	129,463

Table 10: Confusion matrix — ToN_IoT.

available, whereas the autoencoder detects departures from normal behaviour without any labelled attack data, making it better suited to emerging and zero-day threats in dynamic IoT environments.

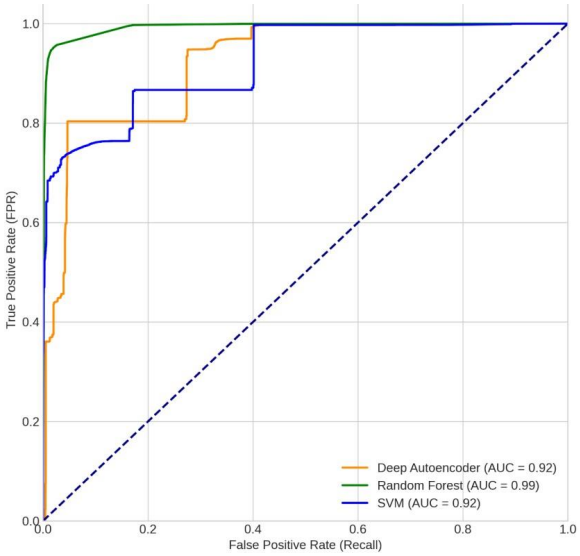

Figure 5: Receiver Operating Characteristic curves for the deep autoencoder, Random Forest, and SVM. The diagonal denotes random classification; every model lies well above it, with curves close to the top-left corner.

Model	Precision	Recall	F1-score	ROC-AUC
SVM	0.90	0.89	0.89	0.92
Random Forest	0.98	0.99	0.99	0.99
Autoencoder (proposed)	0.98	0.82	0.89	0.92

Table 11: Comparison with baseline models (per-metric).

The autoencoder matches Random Forest on precision (0.98) and ties the SVM on F1-score and ROC-AUC, but trails on recall, the expected signature of an unsupervised detector that models normal traffic rather than attack behaviour. Despite the lower recall, the autoencoder offers a more flexible and scalable solution because it can flag previously unseen patterns without labelled data, supporting prior findings on the suitability of autoencoders for zero-day detection [8,9].

In aggregate, Random Forest leads in accuracy and F1score, the SVM is balanced but has the highest false-positive rate, and the autoencoder is competitive with a low falsepositive rate despite reduced recall. These results lay out the central trade-off: accuracy-driven supervised models versus adaptability- and efficiency-driven unsupervised detection. The next subsection shows that the autoencoder decisively wins the dimension that matters most for edge deployment.

4.7. Scalability and Edge Feasibility

Many IoT devices run under tight resource budgets, so a lightweight model is essential. Table 13 compares model size and per-flow inference latency. The autoencoder is by far the smallest model at 0.06 MB, roughly 46 times smaller than Random Forest and over five times smaller than the SVM, while sustaining a low inference latency of 0.156 ms per flow. Random Forest achieves the lowest latency but at a far larger memory footprint, and the SVM is an order of magnitude slower, which limits its scalability in real time. The autoencoder's compactness follows directly from PCA-based feature compression and the lightweight 32-16-8-16-32 topology, which together minimise the trainable parameter count. Although training is computationally intensive, it is performed offline; at deployment, the model scores flows with very low latency, making

Model	Accuracy (%)	F1-score (%)	False-positive rate (%)
Deep autoencoder	83.6	88.4	4.8
Random Forest	97.8	97.5	2.1
Support Vector Machine	91.0	89.0	6.2

Table 12: Aggregate comparison (averaged across IoT-23 and ToN_IoT).

Model	Model size (MB)	Inference latency (ms/flow)
Deepautoencoder (PCA-optimised)	0.06	0.156
Random Forest	2.78	0.051
Support Vector Machine	0.33	2.830

Table 13: Model efficiency comparison

it well-suited to real-time anomaly detection on gateways and edge nodes. This efficiency profile, with high precision and low false alarms at a fraction of the footprint of supervised baselines, is the principal practical advantage of the proposed approach.

5. Discussion

5.1. Principal Findings

The results show that the proposed deep autoencoder detects IoT anomalies effectively while learning only from benign traffic. It achieves high precision (98.3% and 97.1% on IoT-23 and ToN_

IoT) and low false-positive rates (4.8% and 5.1%), confirming that the reconstruction-error mechanism distinguishes abnormal from normal traffic reliably, a result consistent with prior deep-learning IDS work that reported strong false-alarm suppression [1,4]. Recall is lower, indicating that some attacks are missed; this follows from calibrating the threshold for low false alarms and is directly tunable through τ. The wide separation between benign and attack reconstruction errors (Figure 3) provides further evidence that reconstruction error is a dependable anomaly signal.

5.2. Comparison with Existing Studies

The autoencoder's ROC-AUC values (91.9% and 90.8%) are in line with prior deep-autoencoder studies reporting values above 90% [7,9], and its high precision supports earlier observations that autoencoders separate normal and malicious IoT traffic well. Against supervised baselines, the autoencoder records slightly lower overall accuracy than expected, since supervised models exploit labels to learn explicit boundaries, but those same labels constrain supervised models in dynamic environments where new attacks continually emerge. By learning normal behaviour rather than memorising attack signatures, the proposed model is better positioned for zero-day and evolving threats, reinforcing the value of anomaly-based detection in modern IoT security.

5.3. Robustness and Threats to Validity

Because the operating point is fixed analytically rather than tuned, the reported metrics are reproducible and not the product of threshold over-fitting; the recall-false-alarm curve in Table 7 makes the sensitivity of the detector to τ explicit rather than hidden. Evaluating on two datasets with markedly different benign/malicious ratios (11.39% versus 23.69% benign) tests label-free generalisation directly: the consistency of precision and false-positive rate across both corpora indicates that the learned notion of normality transfers rather than over-fitting to one distribution. The principal residual threat to external validity is reliance on public benchmarks, which is addressed in the limitations and future-work directions below.

5.4. Implications

Two implications stand out. First, operating without labelled data is a decisive advantage where comprehensive labelled datasets are difficult or impossible to obtain, as is common in production IoT networks. Second, the model's small footprint (0.06 MB) and low inference latency (0.156 ms/flow) enable deployment on edge devices, supporting timely, ondevice detection and mitigation rather than reliance on centralised analysis. The work also underscores the practical importance of the precision-false-alarm balance: minimising spurious alerts is essential to avoid disruption and operator fatigue, and the percentile-based threshold offers a principled, reproducible way to manage that balance.

5.5. Limitations

Several limitations qualify these findings. The evaluation relies on public datasets that, while realistic, may not capture the full diversity of operational IoT environments. The detector's reliance on a fixed reconstruction-error threshold introduces sensitivity to threshold choice under shifting traffic conditions. Recall is lower than that of supervised baselines, so some attacks go undetected at the chosen operating point. Finally, although PCA improves efficiency, dimensionality reduction may discard information useful for recognising subtle or complex attack patterns. These limitations motivate the future work outlined below.

6. Conclusion and Future Work

This paper presented a lightweight, label-free deepautoencoder framework for anomaly detection in IoT network traffic, designed for deployment on resourceconstrained edge devices. Trained only on benign traffic and equipped with PCA-based feature compression and a percentile-calibrated threshold, the model achieved high precision (98.3% and 97.1%), low false-positive rates (4.8% and 5.1%), and ROC-AUC above 0.90 across the IoT-23 and ToN_IoT benchmarks, while occupying just 0.06 MB and scoring flows in 0.156 ms. Although supervised baselines such as Random Forest achieve higher recall and accuracy when labelled data are available, they cannot match the proposed model's combination of label-free operation, cross-dataset generalisation, and minimal footprint, the properties that determine real-world deployability in dynamic IoT settings. The study's contribution to knowledge is the demonstration that a carefully compressed autoencoder can deliver dependable, low-false-alarm, zero-day-capable detection at a fraction of the resource cost of supervised alternatives, and the identification of packet count, byte rate, and flow duration as the most influential flow features. Future work will pursue five directions. First, hybrid designs that pair the unsupervised autoencoder with a lightweight supervised stage could raise recall while preserving low false alarms. Second, the model should be deployed and evaluated in live IoT testbeds under dynamic conditions to validate the offline efficiency results. Third, advanced architectures, variational and denoising autoencoders, may sharpen anomaly discrimination. Fourth, adaptive thresholding that tracks evolving traffic could reduce sensitivity to a fixed τ. Finally, broadening the evaluation to additional datasets with diverse protocols and attack types would further test generalisation and robustness.

Declarations

Funding: The authors received no specific funding for this work.

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

Ethics approval Not applicable.

This study uses publicly available benchmark datasets and does not involve human participants or animals.

Data Availability: This study uses the publicly available IoT-23 and ToN_IoT benchmark datasets; no new data were generated. The source code underlying the implementation is summarised in the Appendix and is available from the corresponding author on reasonable request.

Author Contributions: Samuel Yao Sebuabe: Conceptualization,

Methodology, Supervision, Writing – review & editing. Stephen Kofi Dotse: Methodology, Validation, Writing – review & editing. Harriet K. O. Lamptey: Software, Data curation, Investigation. Martin Doe: Formal analysis, Visualization. Ezekiel Okoe Annan: Resources, Writing – original draft. Kwame Assa-Agyei: Validation, Writing – review & editing, Supervision. Rebecca Amponsah: Investigation. All authors read and approved the final manuscript.

References

1. Ferrag, M.A., Shu, L., Djallel, H., and Choo, K.K.R., "Deeplearning-based intrusion detection for distributed denial of service attack in Agriculture 4.0," Electronics, vol. 10, no. 11, pp. 1257, 2021.
2. Sikder, A.K., Petracca, G., Aksu, H., Jaeger, T., and Uluagac, A.S.,"A survey on sensor-based threats to Internet-of-Things (IoT) devices and applications," arXiv preprint arXiv:1802.02041, 2018.
3. Saleh, R.A.A., Al-Awami, L., Ghaleb, M., and Abudaqa, A.A.,"Lightweight intrusion detection for IoT systems using artificial neural networks," Applications of Computational Intelligence (Springer), 2024.
4. Liao, H., Murah, M.Z., Hasan, M.K., Aman, A.H.M., Fang, J., Hu,X., and Khan, A.U.R., "A survey of deep learning technologies for intrusion detection in the Internet of Things," IEEE Access, vol. 12, pp. 4745-4761, 2023.
5. Islam, N., Farhin, F., Sultana, I., Kaiser, M.S., Rahman, M.S.,Mahmud, M., Hosen, A.S.M.S., and Cho, G.H., "Towards machine learning-based intrusion detection in IoT networks," Computers, Materials & Continua, vol. 69, no. 2, pp. 1801-1821, 2021.
6. Thakkar, A., and Lohiya, R., "A review of the advancement in intrusion detection datasets," Procedia Computer Science, vol. 167, pp. 636-645, 2020.
7. Liu, Y., Wang, H., Zheng, X., and Tian, L., "An efficient framework for unsupervised anomaly detection over edge-assisted Internet of Things," ACM Transactions on Sensor Networks, 2023.
8. Chen, Z., Li, Z.W., Huang, J., Liu, S.Z., and Long, H.X., "An effective method for anomaly detection in the industrial Internet of Things using XGBoost and LSTM," Scientific Reports, vol. 14, no. 1, 2024.
9. Alaghbari, K.A., Lim, H.S., Saad, M.H.M., and Yong, Y.S., "Deepautoencoder-based integrated model for anomaly detection and efficient feature extraction in IoT networks," IoT, vol. 4, no. 3, pp. 345-365, 2023.
10. Sai Kiran, K.V.V.N.L., Devisetty, R.N.K., Kalyan, N.P., Mukundini, K., and Karthi, R., "Building an intrusion detection system for IoT environment using machine learning techniques," Procedia Computer Science, vol. 171, pp. 2372-2379, 2020.
11. Mahdavinejad, M.S., Rezvan, M., Barekatain, M., Adibi, P.,Barnaghi, P., and Sheth, A.P., "Machine learning for Internet of Things data analysis: a survey," Digital Communications and Networks, vol. 4, no. 3, pp. 161-175, 2018.
12. Tahir, U., Abid, M.K., Fuzail, M., and Aslam, N., "EnhancingIoT security through machine learning-driven anomaly detection," VFAST Transactions on Software Engineering, vol. 12, no. 2, pp. 1-13, 2024.
13. Alrayes, F.S., Zakariah, M., Amin, S.U., Khan, Z.I., and Helal,M., "Intrusion detection in IoT systems using denoising autoencoder," IEEE Access, vol. 12, pp. 122401-122425, 2024.
14. Khraisat, A., and Alazab, A., "A critical review of intrusion detection systems in the Internet of Things: techniques, deployment strategy, validation strategy, attacks, public datasets and challenges," Cybersecurity, vol. 4, no. 1, 2021.
15. Susilo, B., and Sari, R.F., "Intrusion detection in IoT networksusing deep learning algorithm," Information, vol. 11, no. 5, pp. 279, 2020.
16. Bhardwaj, A., Kaushik, K., Bharany, S., Elnaggar, M.F., Mossad,M.I., and Kamel, S., "Comparison of IoT communication protocols using anomaly detection with security assessments of smart devices," Processes, vol. 10, no. 10, pp. 1952, 2022.

Appendix A Implementation Summary

This appendix summarises the implementation to support reproducibility. The pipeline is implemented in Python with TensorFlow/Keras (autoencoder) and Scikit-learn (baselines, PCA, preprocessing). Random seeds are fixed at 42 for NumPy and TensorFlow.
Preprocessing. Records are de-duplicated; Zeek placeholder tokens are converted to missing values; numeric fields are median-imputed, and categorical fields modeimputed and label-encoded; all features are Min-Max scaled to [0, 1]; labels are mapped to benign = 0 and attack = 1.

Feature reduction. PCA retains 95% of the variance; feature importance is approximated from the absolute PCA component loadings, normalised to [0, 1].

Model. A Keras Sequential autoencoder with encoder layers Dense(32)-Dense(16)-Dense(8, bottleneck) and decoder layers Dense(16)-Dense(32)-Dense(input_dim, sigmoid), all hidden layers using ReLU. It is compiled with the Adam optimiser (lr = 0.001) and MSE loss, trained for up to 30 epochs (batch size 256) with EarlyStopping (patience 5, restore best weights) on benign data only.
Detection. The per-sample MSE between input and reconstruction is computed; the threshold is the 95th percentile of the benign training error; test flows above the threshold are flagged as anomalous.

Evaluation. Precision, recall, F1-score, accuracy, ROCAUC, and FPR are computed with Scikit-learn; Random Forest (100 trees) and SVM (RBF, probability=True) baselines are trained on a labelled subsample for comparison, and model size and per-flow latency are measured for the efficiency analysis.