AI-Based Anomaly Detection for Enhanced Cybersecurity in IoT Networks

Mamdouh Muhammad
Computer Networks and Communication Systems (Informatik 7)
Department of Computer Science
Friedrich-Alexander-Universität Erlangen-Nürnberg (FAU)
Erlangen, Germany
mamdouh.muhammad@fau.de

Sushmetha Arumugam
Computer Networks and Communication Systems
Department of Computer Science
Friedrich-Alexander-Universität Erlangen-Nürnberg (FAU)
Erlangen, Germany
sushmetha.arumugam@fau.de

Loui Al Sardy
Computer Networks and Communication Systems (Informatik 7)
Department of Computer Science
Friedrich-Alexander-Universität Erlangen-Nürnberg (FAU)
Erlangen, Germany
loui.alsardy@fau.de

2025 9th Cyber Security in Networking Conference (CSNet) | 979-8-3315-7556-4/25/$31.00 ©2025 IEEE | DOI: 10.1109/CSNet67572.2025.11288241

***Abstract*—Intrusion Detection Systems (IDSs) play a crucial role in securing Internet of Things (IoT) networks, which are increasingly exposed to sophisticated cyber threats. This paper presents an adaptive, reconstruction-based IDS leveraging multiple class-specific Long Short-Term Memory Autoencoders (LSTM-AEs), each trained on a single traffic class (Normal, DDoS-HTTP, DDoS-TCP, DDoS-ICMP). Unlike conventional anomaly detection that models only normal traffic, our approach performs multi-class classification by comparing reconstruction errors across all class-specific models. Evaluated on the Edge-IIoTset dataset, the method achieved a macro-averaged F1-score of 0.9963, underscoring the suitability of LSTM-AE architectures for fine-grained, threat-specific detection in realistic IoT environments.**

***Index Terms*—IoT Security, Anomaly detection, Optuna, Deep Learning, LSTM Autoencoder, Edge-IIoTset.**

I. Introduction

The rapid expansion of internet-connected devices across households, industry, and smart infrastructure has significantly increased the cyber threat landscape. Securing Internet of Things (IoT) networks is critical to safeguard sensitive data, ensure operational continuity, and protect public safety [1]. While IoT offers tremendous benefits through automation and efficiency, it also introduces vulnerabilities due to the heterogeneity, limited computational capacity, and often inadequate security of connected devices [2]. Traditional perimeter-based defences are increasingly insufficient in such dynamic environments.

Intrusion Detection Systems (IDSs) have emerged as key components of modern cybersecurity architectures, monitoring network traffic and system behaviour to detect malicious activity. However, conventional IDSs, particularly those relying on signature-based methods or static machine learning models, often suffer from high false-positive rates, poor generalisation, and limited capacity to detect zero-day or behaviourally similar attacks [3], [4]. To address these limitations, recent studies have explored Deep Learning (DL) techniques that can learn complex attack patterns directly from network traffic data [5], [6].

Architectures such as Convolutional Neural Networks (CNNs), Long Short-Term Memory (LSTM) networks, and Gated Recurrent Units (GRUs) offer improved spatial and temporal dependencies in network traffic [7], [8]. LSTMs, in particular, are well-suited to sequential traffic analysis. While GRUs offer a computationally efficient alternative, and CNNs excel at detecting localised spatial features. Traditional machine learning models such as Random Forest (RF), Support Vector Machine (SVM), and K-Nearest Neighbors (KNN) remain useful for baseline comparisons due to their interpretability and robustness.

In this study, we propose a deep intrusion detection system using multiple LSTM autoencoders trained on separate traffic classes rather than a single normal-only model. This class-conditional training strategy enables fine-grained detection of high-impact volumetric threats, specifically DDoS-HTTP, DDoS-TCP, and DDoS-ICMP, as well as normal traffic. By classifying inputs based on the lowest reconstruction error among all trained autoencoders, the framework supports multi-class anomaly detection and demonstrates improved discrimination between similar attack patterns compared to traditional binary anomaly detection or static supervised classifiers. The LSTM autoencoder model is optimised through Bayesian hyperparameter tuning using the Optuna framework and is benchmarked against a Dense Neural Network (DNN) baseline.

The key contributions of this paper are as follows:

- **Preprocessing pipeline:** We design a reproducible pipeline that balances classes, extracts temporal and structural features (e.g., inter-arrival deltas, cyclic time), and standardizes inputs for sequential modeling.

- **Class-conditional LSTM-AEs:** We train separate LSTM autoencoders for each traffic class (Normal, DDoS-HTTP, DDoS-TCP, DDoS-ICMP), enabling multi-class detection via reconstruction error comparison rather than a single normal-only model.
- **Hyperparameter optimization:** Using Optuna, we tune latent size, window length, and thresholds per class to enhance detection accuracy and robustness.
- **Evaluation on Edge-IIoTset:** Our approach surpasses a DNN baseline and classical ML models, achieving a macro-averaged F1-score of 0.9963 with fine-grained DDoS detection.

The remainder of this paper is structured as follows. Section II provides an overview of related work on anomaly-based intrusion detection systems. Section III details the proposed system architecture, preprocessing pipeline, and model training strategy. Section IV presents the experimental setup, performance metrics, and comparative results. Finally, Section V concludes the paper and outlines directions for future research.

II. Related Work

As IoT deployments scale across critical infrastructure, anomaly-based Intrusion Detection Systems (IDSs) are gaining increased attention for their ability to detect novel and evolving threats. Traditional IDSs, particularly signature-based or rule-driven models, often struggle with generalisation and adaptability in dynamic IoT environments [3], [5]. To overcome these limitations, Deep Learning (DL) methods, especially recurrent architectures such as Long Short-Term Memory (LSTM) networks, have been explored due to their strengths in temporal pattern recognition within network traffic [6]. This architecture was originally introduced to address these shortcomings due to its strength in modelling temporal patterns in sequential traffic data [9].

LSTM networks are particularly suitable for detecting IoT-based attacks that exhibit time-dependent behaviour. For example, Thant et al. [7] proposed an LSTM-based IDS for IoT networks, demonstrating improved accuracy in detecting temporal attacks such as DDoS and slow-rate floods. Similarly, Vinayakumar et al. [10] showed that LSTM and Deep Neural Networks (DNNs) outperform traditional classifiers in modelling sequential traffic behaviour, particularly in multi-class intrusion detection scenarios. Popoola et al. [11] proposed a multi-stage hybrid deep learning framework, which achieved improved performance on multi-class intrusion detection in IoT networks.

In addition to single-architecture models, hybrid deep learning approaches have gained increasing attention. Wang et al. [12] proposed a CNN–GRU model with XGBoost-based feature selection, achieving strong results on BoT-IoT and UNSW-NB15 datasets. Kilichev et al. [13] applied a CNN–LSTM–GRU ensemble to detect anomalies in Electric Vehicle Charging Stations using the Edge-IIoTset dataset. However, they did not isolate high-impact attack classes, or optimise class-wise detection. Their methodology highlights the utility of Edge-IIoTset in realistic IIoT use cases. Qazi et al. [14] proposed a one-dimensional CNN-based intrusion detection system, achieving high accuracy on the CICIDS2017 dataset with reduced computational complexity and minimal preprocessing. Altangerel et al. [15] developed a one-dimensional CNN model for anomaly detection in IoT networks, demonstrating strong performance in constrained environments using custom network traffic features.

Shone et al. [16] introduced a deep autoencoder-based IDS capable of learning hierarchical representations from raw network data, reducing the need for manual feature engineering. Javaid et al. [17] introduced an autoencoder-based intrusion detection system trained on normal traffic patterns, highlighting the effectiveness of deep unsupervised learning in distinguishing anomalies from legitimate behaviour. Mohamed et al. [18] provided a comprehensive review of deep learning techniques for intrusion detection in IoT and IIoT systems, covering hybrid model of CNN and LSTM autoencoders, and discussing their strengths in real-world operational deployments.

However, while many of these works demonstrate strong results in multi-class intrusion detection, they often overlook evaluation across high-impact IoT attack types. Additionally, aspects such as per-class performance analysis, targeted feature selection, and real-time feasibility remain underexplored. Our work differs in its targeted feature engineering, use of balanced class distributions, and evaluation on high-impact volumetric DDoS attacks using class-specific autoencoders. This design addresses key gaps in the literature regarding class-level granularity, adaptive anomaly thresholds, and model interpretability for realistic IoT security applications.

III. Methodology

This section outlines the methodology adopted to design, implement, and evaluate an LSTM autoencoder-based anomaly detection system for IoT networks. The proposed pipeline includes dataset selection, feature engineering, class rebalancing, model architecture design, and hyperparameter optimisation using the Optuna framework.

A. Attacks covered

This paper focuses on four traffic classes drawn from the Edge-IIoTset dataset: DDoS_HTTP, DDoS_TCP, DDoS_ICMP, and Normal Traffic. This selection is informed by a recent report of ENISA Threat Landscape 2024 report [19], which identifies DDoS attacks as significant threats to interconnected systems and operational technologies. Each selected attack type is described below to highlight its operational characteristics and relevance to IoT threat detection.

DDoS_HTTP (Layer 7): These application-layer attacks involve large volumes of HTTP GET or POST requests that appear normal but aim to exhaust web server resources such as CPU, memory, and thread pools. Tools such as Slowloris exploit this vector by sending incomplete or slow headers

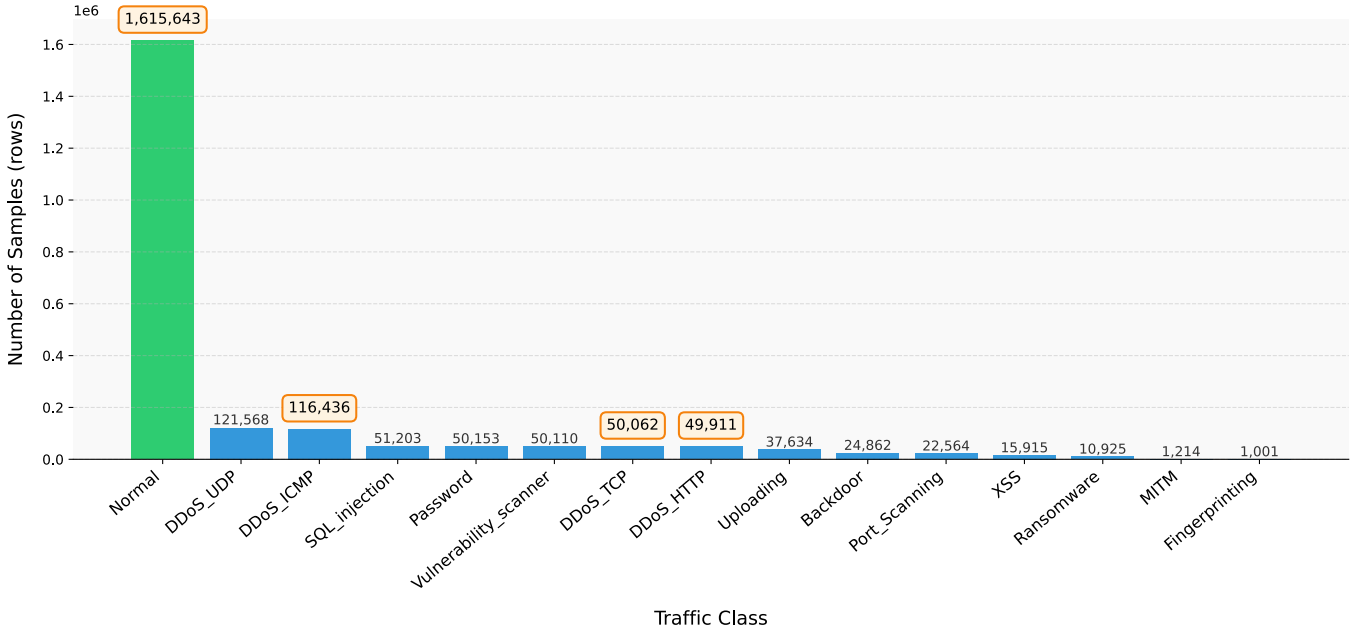

Fig. 1. Class distribution in the Edge-IIoT dataset (DNN File).

to keep connections open indefinitely, often avoiding basic filtering due to their low-rate, fragmented traffic profiles.

DDoS_TCP (Layer 4): This attack floods servers with spoofed SYN packets during the TCP handshake, causing the server to allocate resources for connections that never complete. The resulting backlog of half-open connections leads to resource exhaustion. Stateful TCP processing makes such attacks especially difficult to mitigate

DDoS_ICMP (Layer 3): These network-layer attacks, including ping floods and Smurf attacks, overwhelm network infrastructure with Echo Requests. Smurf attacks use spoofed IP addresses and broadcast amplification to magnify their impact. The stateless nature of ICMP complicates mitigation, particularly on legacy devices lacking modern traffic controls.

Normal Traffic: Benign traffic collected under operational conditions is included to represent baseline behaviour across typical IIoT protocols.

B. Dataset Description

This study employs the Edge-IIoTset dataset [20], a publicly available benchmark specifically designed for intrusion detection in Industrial IoT (IIoT) environments. The dataset comprises over 2 million labeled network records, collected from simulated IIoT scenarios with three main directories: normal traffic, attack traffic, and a preselected subset for machine learning and deep learning tasks. It includes 15 attack types and multiple IIoT-specific communication protocols such as Modbus/TCP (MBTCP) and MQTT, which are widely used in manufacturing, energy, and smart infrastructure. Table I lists its attributes and metadata.

Our experiments utilise the preprocessed file `DNN-EdgeIIoT-dataset.csv`, which is part of the dataset's machine learning-ready subset. This file contains flow-level features derived from diverse traffic sessions. Protocol-specific features such as `mbtcp.len`, `mqtt.msg`, and `tcp.flags` are included alongside statistical indicators and temporal metadata, enabling both tabular and sequential learning.

For this study, we focus on a four-class setting involving Normal traffic and three volumetric attack types: DDoS HTTP, DDoS TCP, and DDoS ICMP. The DDoS UDP class was excluded due to missing timestamp values in the `frame.time` and `udp.time_delta` fields, which resulted in schema misalignment, as summarised in Table I. This subset was selected to reflect high-impact threats and enable the application of temporal sequence models.

C. Preprocessing Pipeline

The preprocessing pipeline transforms raw traffic data into structured sequences suitable for temporal modelling with LSTM autoencoders. It consists of several key stages, including data cleaning, stratified data splitting, temporal and structural feature engineering, categorical encoding, and feature scaling. These steps collectively ensure that the input data is normalised, balanced, and temporally consistent, which is essential for effective anomaly detection in sequential traffic patterns.

Data Cleaning. The dataset was filtered to include four relevant traffic classes: DDoS_HTTP, DDoS_TCP, DDoS_ICMP, and Normal. These were selected to align with the study's focus on prevalent volumetric attacks and representative normal traffic. No missing values were detected in the filtered data. Duplicate entries were removed to reduce redundancy and prevent biased learning. Continuous fields

TABLE I
SUMMARY OF EDGE-IIOTSET DATASET CHARACTERISTICS

Attribute	Description
Granularity	Flow-level data with timestamped records capturing detailed network activity.
Size (samples/flows)	∼2.2 million labeled records.
Features	63 features: protocol-level (TCP, UDP, MQTT, HTTP, DNS, ICMP), statistical indicators, temporal metrics.
Volume (Time Span)	Multi-day traffic emulation reflecting realistic IIoT operations.
Attack Taxonomy	14 attack types across 5 categories: DDoS, Injection, Scanning, MITM, Malware.
Representation	Suitable for classical ML and deep learning; supports tabular and sequence models.
Class Distribution	Imbalanced; includes rare attacks (e.g., MITM attack, Fingerprinting attack).
License	Open academic license via IEEE DataPort.
Data Collection Method	Simulated IIoT network with protocol/device-level emulation.
Temporal Splits	Timestamped data supports sequence modelling (Except `DDoS_UDP` attack).
Relevance & Currency	Released in 2023 and aligned with modern IIoT threats and system design.
Labeled	Fully labeled with ground truth annotations; includes two columns: `Attack_label` and `Attack_type`, enabling both binary and multi-class classification.
Issues Identified	`DDoS_UDP` attack entries lack values in the `frame.time` (a.k.a. timestamp) field with 0 values all over the `udp.time_delta` feature, resulting in column misalignment or schema shift during parsing. According to our analysis, each subsequent field is offset by one column, displacing the true values.

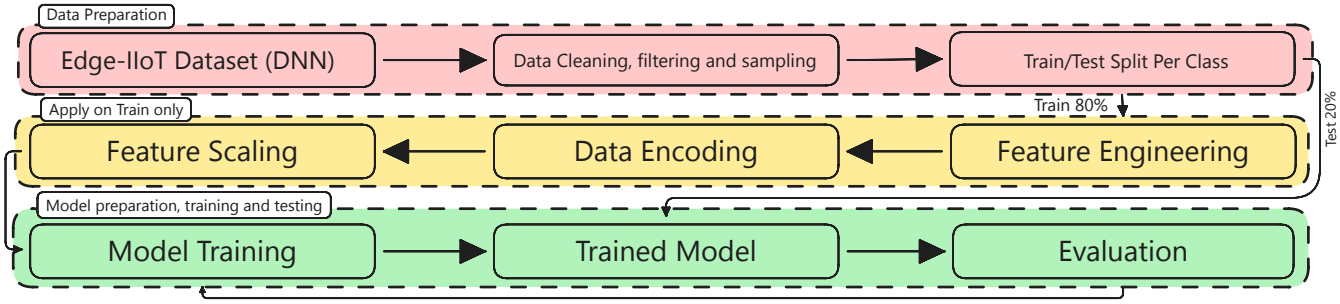

Fig. 2. Data preprocessing, training, and evaluation pipeline.

TABLE II
HYPERPARAMETERS USED IN THE LSTM AUTOENCODER EXPERIMENTS: OPTUNA-OPTIMIZED (UPPER PART) AND FIXED EXPERIMENTAL PARAMETERS (LOWER PART)

Parameter	Value
Optuna-Optimized Parameters	
Window length (W)	5–20 timesteps *(chosen: 10)*
Latent dimension	8–64 *(chosen: 20)*
Batch size	64, 128, 256, 512 *(chosen: 256)*
Epochs	10–50 *(chosen: 30)*
Learning rate	10^{-4} – 10^{-2} *(chosen:* 10^{-3}*)*
Threshold percentile	95–99.9% *(chosen: 99%)*
Fixed Experimental Parameters	
Normal samples (N_{normal})	200,000
Attack samples (N_{attack})	49,000 per class
Train/test split	80% / 20%
Random seed	42

TABLE III
CLASS SIZES BEFORE AND AFTER BALANCING

Class	Before Balancing	After Balancing
Normal	1,615,643	200,000
DDoS_HTTP	49,911	49,000
DDoS_TCP	50,062	49,000
DDoS_ICMP	116,436	49,000

were retained in numeric format, while non-numeric fields were identified for encoding in subsequent stages.

Data Splitting. An 80/20 stratified split was applied to divide the dataset into training and test subsets, maintaining class balance across both partitions. Importantly, the split was performed prior to the generation of temporal windows to prevent information leakage between training and evaluation data. A fixed random seed (42) was used to ensure experimental reproducibility.

Feature Engineering. In contrast to the dataset creators' default recommendations [21], this study introduces additional temporal and structural features to enhance model performance. Arrival timestamps were used to calculate inter-arrival time deltas, which help capture changes in packet rates. Timestamps were also encoded as cyclical variables using sine and cosine transformations of time-of-day to retain periodic behavioural patterns while avoiding discontinuities.

IP addresses were transformed into 32-bit integers by concatenating binary-encoded octets, producing compact, model-compatible numerical representations that preserve uniqueness without implying ordinal relationships. These engineered features were combined with raw packet attributes to form the complete feature set.

- **Timeframe Engineering.** Timestamps were parsed, converted to nanoseconds, and sorted chronologically. Inter-arrival time deltas were computed, and daily periodicity was captured using sine and cosine encodings of time-of-day (see Eqs. 1–5). All temporal features were standardised using statistics from the training set only.

$$t_i = \text{timestamp of frame } i \text{ in seconds} \quad (1)$$

$$\Delta t_i = t_i - t_{i-1}, \quad \Delta t_0 = 0 \quad (2)$$

$$\text{secs}_i = t_i \bmod 86400 \quad (3)$$

$$\text{tod\_sin}_i = \sin\left(\frac{2\pi \cdot \text{secs}_i}{86400}\right) \quad (4)$$

$$\text{tod\_cos}_i = \cos\left(\frac{2\pi \cdot \text{secs}_i}{86400}\right) \quad (5)$$

$$x' = \frac{x - \mu_x}{\sigma_x} \quad \text{(z-score normalisation)} \quad (6)$$

- **IP Address Engineering.** IPv4 addresses were transformed from dotted-decimal format into 32-bit integers by concatenating binary-encoded octets. For example, 192.168.0.128 becomes:

  $$11000000\ 10101000\ 00000000\ 10000000 \rightarrow 3232235648$$

  This representation reduces dimensionality and ensures compatibility with numerical models, without implying ordinal relationships.

Data Encoding. Categorical features such as `ip.src_host`, `ip.dst_host`, `http.request.method`, and `tcp.flags` were encoded using label encoding. Each unique value was assigned an integer label, thereby avoiding the feature space explosion associated with one-hot encoding, particularly problematic in high-cardinality fields such as IP addresses and protocol flags. Encoded categorical features were merged with numerical attributes (e.g., `tcp.len`, `icmp.seq_le`) into a single feature matrix for scaling and modelling.

Feature Scaling. All numerical features were standardised using z-score normalisation with the `StandardScaler` from Scikit-learn. Scaling parameters (mean and standard deviation) were computed exclusively on the training set and subsequently applied to the test set to avoid data leakage. Standardisation to zero mean and unit variance ensured numerical stability during training and helped balance feature influence in the LSTM model, which is sensitive to input magnitudes.

D. Model Architecture

The proposed intrusion detection framework employs a class-conditional LSTM autoencoder to model the temporal dynamics of multivariate IoT traffic. The model is designed to learn a compact representation of normal and attack-class traffic patterns and to detect anomalies based on reconstruction errors.

Each input instance is represented as a sequence of W time steps, where each step contains a standardised feature vector. The architecture follows a symmetric encoder–decoder structure implemented using the Keras deep learning library.

The **encoder** contains a single LSTM layer with d hidden units, which processes the input sequence and encodes it into a fixed-length latent vector, as expressed in Eq. 7:

$$h_t = \text{LSTM}_{enc}(x_t, h_{t-1}), \quad z = h_T \quad (7)$$

where x_t is the input feature vector at time step t, h_t is the hidden state produced by the encoder LSTM at that step, and $z = h_T$ represents the final latent vector summarizing the entire input sequence.

This representation is then passed to a `RepeatVector` layer, which replicates the latent vector across W time steps to initialise the decoder input.

The **decoder** includes a second LSTM layer that attempts to reconstruct the original input sequence from the latent representation, following Eq. 8:

$$\hat{x}_t = \text{LSTM}_{dec}(z, h_{t-1}) \quad (8)$$

where $\hat{x}_t$ is the reconstructed feature vector at time step t, generated by the decoder LSTM using the latent vector z and the previous hidden state h_{t-1}.

The output of the decoder is passed through a Time Distributed dense layer to produce a sequence of reconstructed feature vectors with the same dimensionality as the input.

The model is trained using the Adam optimiser and the Mean Squared Error (MSE) loss function, defined as:

$$\text{MSE} = \frac{1}{N}\sum_{i=1}^{N}(x_i - \hat{x}_i)^2 \quad (9)$$

where x_i and $\hat{x}_i$ denote the original and reconstructed feature vectors at time step i, and N is the number of time steps in the sequence.

During inference, each class-specific autoencoder computes a reconstruction error for a given input sequence. Let $\hat{x}_{i,c}$ denote the reconstructed vector produced by the autoencoder trained on class c. The per-class reconstruction error is computed as:

$$E_c = \frac{1}{N}\sum_{i=1}^{N}(x_i - \hat{x}_{i,c})^2 \quad (10)$$

where $\hat{x}_{i,c}$ is the reconstruction from the autoencoder of class c. The predicted class is the one associated with the autoencoder yielding the lowest reconstruction error,

provided that the error is below a class-specific threshold. These thresholds are determined using percentile-based tuning via the Optuna framework. This strategy enables effective discrimination among multiple traffic classes based on reconstruction confidence.

E. Training Strategy

Each LSTM autoencoder was trained independently on a single traffic class to learn its temporal behaviour. The objective was to minimise the reconstruction error of class-specific input sequences using the Adam optimiser and the Mean Squared Error (MSE) loss function defined in Eq. 9. This class-conditional training scheme enables each autoencoder to specialise in reconstructing patterns unique to its corresponding class.

To optimise model performance, key hyperparameters, including latent dimension size, window length, learning rate, batch size, and reconstruction error threshold percentile, were tuned using the Optuna framework. The macro-averaged F1-score on a validation split of the training data served as the objective metric during tuning. Optuna's pruning mechanism was used to terminate underperforming trials early, thereby accelerating the optimisation process.

Training was conducted for up to 30 epochs with early stopping, using a fixed random seed (42) to ensure reproducibility. Because class balancing was applied during preprocessing, no additional class weighting was needed during training.

Each of the four autoencoders is trained exclusively on data from a single class. At inference, all four models compute reconstruction errors as described in Eq. 10, and classification is based on the lowest error among them rather than a binary anomaly vs. normal decision. The class corresponding to the model with the lowest error is assigned as the predicted label, provided the error is below a threshold determined via Optuna tuning, as expressed in Eq. 11:

$$\hat{c} = \arg\min_{c}\{S_c(x)\}, \quad S_c(x) < \tau_c \tag{11}$$

Sequences that exceed all class-specific thresholds can optionally be flagged as anomalous or rejected, depending on the deployment context.

F. Threat Model Assumptions

The proposed system assumes a passive, external attacker model in which adversaries can observe and inject malicious network traffic into the IoT environment but do not possess control over internal system components or endpoint devices. This reflects common threat scenarios in which industrial IoT networks are exposed to volumetric flooding attacks or stealthy intrusion attempts via unsecured network interfaces.

Specifically, the study targets high-impact distributed denial-of-service (DDoS) attack types, such as HTTP floods, TCP SYN floods, and ICMP amplification attacks, which aim to exhaust network or application-layer resources through abnormal traffic patterns. These are assumed to be mounted by remote adversaries leveraging spoofed IP addresses, low-and-slow tactics, or protocol misuse.

The threat model excludes scenarios involving compromised IoT nodes, insider threats, or adversaries with privileged access to the infrastructure. Additionally, adversarial machine learning (e.g., evasion or poisoning attacks) is considered out of scope for this work. These limitations reflect a practical focus on early-stage intrusion detection within constrained industrial and edge environments.

This threat model supports the design of an IDS that operates on flow-level features and sequence data without relying on encrypted payload inspection or host-level instrumentation.

G. Optuna Framework

Optuna [22], a Bayesian optimisation framework for automated hyperparameter tuning and early stopping, was employed to optimise the performance of the LSTM autoencoder. The search space included encoder latent dimensions (8–64), batch sizes (64, 128, 256, 512), training window lengths (5–20 timesteps), training epochs (10–50), Adam optimiser learning rates (10^{-4}–10^{-2}), and reconstruction-error percentile thresholds (95–99.9%).

The optimisation objective was to maximise the macro-averaged F1-score over the hyperparameter search space, formulated in Eq. 12:

$$\max_{\theta \in \Theta} \; F1_{\text{macro}}(\theta) \tag{12}$$

where θ represents hyperparameters (latent dimension, learning rate, window size, threshold percentile). The objective function was evaluated on a stratified validation split of the training set, and Optuna's pruning mechanism dynamically terminated underperforming trials based on intermediate evaluation results, thereby accelerating convergence.

This tuning procedure yielded class-specific hyperparameter configurations that effectively balanced detection accuracy, computational efficiency, and generalisability across heterogeneous traffic patterns.

IV. Evaluation and Experiments

The evaluation focuses on four representative classes: Normal, DDoS-HTTP, DDoS-TCP, and DDoS-ICMP, which were selected based on their prevalence and severity in current IoT threat landscapes as discussed in III-A.

Performance was assessed using standard multi-class classification metrics: Accuracy, Precision (Eq. 14), Recall (Eq. 15), F1-score (Eq. 16), computed using macro-averaging (Eq. 13) to account for class imbalance.

$$F1_{\text{macro}} = \frac{1}{C}\sum_{c=1}^{C}\frac{2 \cdot P_c \cdot R_c}{P_c + R_c} \tag{13}$$

$$\text{Precision}_c = \frac{TP_c}{TP_c + FP_c} \tag{14}$$

$$\text{Recall}_c = \frac{TP_c}{TP_c + FN_c} \tag{15}$$

TABLE IV
PER-CLASS PERFORMANCE METRICS OF THE LSTM AUTOENCODER MODEL: BEST VALUES IN EACH COLUMN ARE HIGHLIGHTED

Class	Precision	Recall	F1-Score
Normal	0.9993	**0.9999**	**0.9996**
DDoS_ICMP	0.9996	0.9908	0.9952
DDoS_TCP	**0.9999**	0.9868	0.9933
DDoS_HTTP	0.9945	**0.9997**	0.9971
Macro Avg	0.9983	0.9943	0.9963
Weighted Avg	0.9967	0.9967	0.9967

$$F1_c = \frac{2 \cdot \text{Precision}_c \cdot \text{Recall}_c}{\text{Precision}_c + \text{Recall}_c} \tag{16}$$

A. Comparison with Edge-IIoTset Benchmarks

Ferrag et al. [20] reported baseline results on the Edge-IIoT dataset using a variety of classical and deep learning models, including decision trees, random forests, support vector machines, k-nearest neighbours, and deep neural networks. Among these, the deep neural network achieved the best performance, reaching an F1-score of approximately 0.95 in their six-class classification setting.

In contrast, the proposed LSTM autoencoder, evaluated in a four-class setting, achieved an overall accuracy of 99.67% and a macro-averaged F1-score of 0.9963. As shown in Table IV, class-wise F1-scores exceeded 0.993 for all attack types and reached 0.999 for Normal traffic. These results represent a relative improvement of 4-5 percentage points over the strongest DNN baseline, underscoring the advantages of sequence-aware reconstruction-based modelling for high-volume attack detection in IoT networks.

B. Confusion Matrix Analysis

Fig. 3 shows the confusion matrix for the LSTM autoencoder model. Misclassifications were minimal across all classes. The model classified both Normal and DDoS_HTTP traffic with near-perfect accuracy, while DDoS_TCP and DDoS_ICMP yielded false negative rates below 0.5%.

This low level of inter-confusion demonstrates that combining class-conditional reconstruction with temporal feature encoding enables the model to distinguish subtle behavioural differences DDoS variants. In contrast, baseline classifiers reported by Ferrag et al. [20] struggled to achieve such fine-grained discrimination. The consistently high precision and recall across all classes confirm the robustness and reliability of the proposed framework for volumetric intrusion detection in IoT environments.

V. CONCLUSION

This study introduces a multi-class intrusion detection system for Internet of Things networks. The method uses class-specific LSTM autoencoders trained on segmented traffic sequences to learn the behavior of each traffic class. It targets three volumetric attack types: DDoS-HTTP, DDoS-TCP, and DDoS-ICMP, along with normal traffic. These cases were

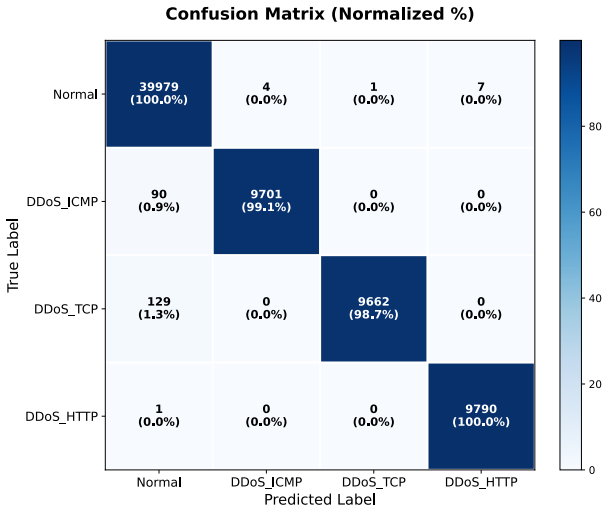

Fig. 3. Confusion matrix for the evaluated model.

selected from the Edge-IIoTset dataset to reflect common and severe IoT threats.

Preprocessing included class balancing, timestamp-based features such as inter-arrival intervals and cyclic time encoding, IP address transformation, categorical encoding, and z-score normalization. Traffic data was divided into windows to keep temporal dependencies intact. Hyperparameters including latent size, window length, learning rate, and reconstruction thresholds were tuned with the Optuna framework, which pruned unpromising trials to reduce training time.

The LSTM autoencoder reached near-perfect Precision, Recall, and F1-score, surpassing benchmarks from decision trees, random forests, SVM, KNN, and deep neural networks tested on Edge-IIoTset.

Future work will aim to deploy the model on low-power edge devices by reducing complexity and inference delay. The study also plans to expand the threat model to include insider and adversarial attacks such as poisoning and evasion, exploring architectures like transformers and adaptive sequence models to improve resilience.

TABLE V
PERFORMANCE COMPARISON OF PROPOSED LSTM AUTOENCODER AND EDGE-IIOTSET BASELINES FOR NORMAL AND DDOS TRAFFIC DETECTION (PR: PRECISION, RC: RECALL, F1: F1-SCORE)

2*Model	2*Traffic Type	Metrics (Pr / Rc / F1)			F1 Gain vs. Baselines		
		Normal	DDoS	Macro Avg	DT	RF	DNN
Proposed (LSTM-AE)	Combined	**0.9993 / 0.9999 / 0.9996**	**0.9945 / 0.9997 / 0.9971**	**0.9983**	+0.1871	+0.0971	+0.0471
DT	Combined	1.00 / 1.00 / 1.00	0.73 / 0.92 / 0.81	0.905	–	–	–
RF	Combined	1.00 / 1.00 / 1.00	0.98 / 0.83 / 0.90	0.950	–	–	–
SVM	Combined	1.00 / 1.00 / 1.00	0.96 / 0.83 / 0.89	0.945	–	–	–
KNN	Combined	1.00 / 1.00 / 1.00	0.88 / 0.90 / 0.89	0.945	–	–	–
DNN	Combined	1.00 / 1.00 / 1.00	0.92 / 0.98 / 0.95	0.975	–	–	–

REFERENCES

[1] M. A. Khan and K. Salah, "Iot security: Review, blockchain solutions, and open challenges," *Future Generation Computer Systems*, vol. 82, pp. 395–411, 2018, doi:10.1016/j.future.2017.11.022.

[2] B. C. Mahmoud Ammar, Giovanni Russello, "Internet of things: A survey on the security of iot frameworks," *Journal of Information Security and Applications*, vol. 38, pp. 8–27, 2018, doi:10.1016/j.jisa.2017.11.002.

[3] M. Labonne, "Anomaly-based network intrusion detection using machine learning," Ph.D. dissertation, Institut Polytechnique de Paris, 2020, available at: https://theses.hal.science/tel-02988296.

[4] R. Priyadarshini and R. Barik, "A deep learning based intelligent framework to mitigate ddos attack in fog environment," *Journal of King Saud University - Computer and Information Sciences*, vol. 34, no. 3, 2019, doi:10.1016/j.jksuci.2019.04.010.

[5] Z. Yang, X. Liu, T. Li, D. Wu, J. Wang, Y. Zhao, and H. Han, "A systematic literature review of methods and datasets for anomaly-based network intrusion detection," *Computers & Security*, vol. 116, p. 102675, 2022, doi:10.1016/j.cose.2022.102675.

[6] B. Lindemann, B. Maschler, N. Sahlab, and M. Weyrich, "A survey on anomaly detection for technical systems using lstm networks," *Computers in Industry*, vol. 131, p. 103498, 2021, doi:10.1016/j.compind.2021.103498.

[7] Y. M. Thant, M. M. Su Thwin, and C. S. Htwe, "Iot network intrusion detection using long short-term memory recurrent neural network," *2023 IEEE Conference on Computer Applications (ICCA)*, pp. 334–339, 2023, doi:10.1109/ICCA51723.2023.10182005.

[8] M. Parihar and C. Fung, "Ids with deep learning techniques," in *Proceedings of the 2023 7th Cyber Security in Networking Conference (CSNet)*, 2023, pp. 1–4, doi:10.1109/CSNet59123.2023.10339748.

[9] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 1997, doi:10.1162/neco.1997.9.8.1735.

[10] R. Vinayakumar, K. P. Soman, and P. Poornachandran, "Deep learning approach for intelligent intrusion detection system," *IEEE Access*, vol. 7, pp. 41 525–41 550, 2019, doi:10.1109/ACCESS.2019.2895334.

[11] S. Popoola, Y. Tsado, A. Ogunjinmi, E. Sanchez-Velazquez, Y. Peng, and D. Rawat, "Multi-stage deep learning for intrusion detection in industrial internet of things," *IEEE Access*, 2025, doi:10.1109/ACCESS.2025.3557959.

[12] Z. Wang, H. Huang, R. Du, X. Li, and G. Yuan, "Iot intrusion detection model based on cnn-gru," *Frontiers in Computing and Intelligent Systems*, vol. 4, no. 2, pp. 90–95, 2023, doi:10.54097/fcis.v4i2.10302.

[13] D. Kilichev, D. Turimov, and W. Kim, "Next-generation intrusion detection for iot evcs: integrating cnn, lstm, and gru models," *Mathematics*, vol. 12, no. 4, p. 571, 2024, doi:10.3390/math12040571.

[14] E. U. H. Qazi, A. Almorjan, and T. Zia, "A one-dimensional convolutional neural network (1d-cnn) based deep learning system for network intrusion detection," *Applied Sciences*, vol. 12, no. 16, p. 7986, 2022, doi:10.3390/app12167986.

[15] G. Altangerel, M. Tejfel, and E. Tsogbaatar, "Iot anomaly detection with 1d cnn using p4 capabilities," *Acta Electrotechnica et Informatica*, vol. 23, pp. 3–12, 2023, doi:10.2478/aei-2023-0006.

[16] N. Shone, T. N. Ngoc, V. D. Phai, and Q. Shi, "A deep learning approach to network intrusion detection," *IEEE Transactions on Emerging Topics in Computational Intelligence*, vol. 2, no. 1, pp. 41–50, 2018, doi:10.1109/TETCI.2017.2772792.

[17] A. Javaid, Q. Niyaz, W. Sun, and M. Alam, "A deep learning approach for network intrusion detection system," in *Proceedings of the 9th EAI International Conference on Bio-Inspired Information and Communications Technologies (BIONETICS)*, 2016, pp. 21–26, doi:10.4108/eai.3-12-2015.2262516.

[18] M. Ibrahim, S. Gharghory, and H. Kamal, "A hybrid model of cnn and lstm autoencoder-based short-term pv power generation forecasting," *Springer*, vol. 106, pp. 4239–4255, 2024, doi:10.1007/s00202-023-02220-8.

[19] European Union Agency for Cybersecurity (ENISA), "Enisa threat landscape 2024: Mapping the threat landscape for emerging and connected technologies," 2024, available at: https://www.enisa.europa.eu/publications/enisa-threat-landscape-2024.

[20] M. A. Ferrag, L. Maglaras, H. Janicke, J. Jiang, and P. Fragkou, "Edge-iiotset: A new comprehensive realistic cyber security dataset of iot and iiot applications for centralized and federated learning," *IEEE Access*, vol. 10, pp. 40 258–40 277, 2022, doi:10.1109/ACCESS.2022.3165809.

[21] M. A. F. et al., "Edge-iiotset: A new comprehensive realistic cyber security dataset of iot and iiot applications for centralized and federated learning," 2022, available at: https://www.kaggle.com/datasets/mohamedamineferrag/edgeiiotset-cyber-security-dataset-of-iot-iiot.

[22] T. Akiba, S. Sano, T. Yanase, T. Ohta, and M. Koyama, "Optuna: A next-generation hyperparameter optimization framework," in *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining (KDD '19)*, 2019, pp. 2623–2631, doi:10.1145/3292500.3330701, Available at: https://optuna.org/.