Majestic International Journal of AI Innovations
ISSN: 3142-8428
Volume 1, 2026, **pp.** 1-18, **Article ID:** CM2601105006
DOI: 10.65080/mjai.v1.CM2601105006

CONTENT MAJESTIC PUBLISHER

Research Article

Hybrid CNN–LSTM Intrusion Detection Framework for Industrial IoT Security

Imad Ullah[1, *], **Mushtaq Ali**[1]

[1]*Riphah Institute of Informatics, Malakand Campus, Riphah International University, Islamabad, Pakistan*

Article History
Received: 17 February, 2026 Revised: 11 July, 2026 Accepted: 14 July, 2026 Published: 29 July, 2026

Abstract:

Introduction: The rapid adoption of the Industrial Internet of Things (IIoT) has increased the exposure of safety-critical industrial systems to sophisticated cyberattacks, requiring intrusion detection mechanisms that are accurate, computationally efficient, and operationally reliable. Traditional intrusion detection systems often struggle with correlated traffic descriptors, temporal attack evolution, false alarm control, and deployment-level reliability in resource-constrained industrial environments.

Methodology: This study proposes a lightweight hybrid CNN–LSTM intrusion detection framework for binary IIoT attack detection. Convolutional layers learn compact representations from high-dimensional statistical traffic descriptors, while an LSTM layer captures short-term temporal dependencies associated with multi-stage and slow-rate attacks. The model was evaluated on the BoTNeTIoT-L01 Industrial IoT benchmark using a leakage-controlled split, imbalance-aware metrics, threshold-specific false alarm analysis, probability calibration, temporal robustness assessment, and CPU-only inference benchmarking.

Results: The proposed CNN–LSTM achieved accuracy = 0.99875, precision = 0.99930, recall/sensitivity = 0.99820, F1-score = 0.99875, and FAR = 0.00070 on the held-out test set. At the selected deployment threshold, the model produced a ROC operating point with TPR = 0.99820 and FPR = 0.00070. Same-split baseline and ablation comparisons further demonstrated that the proposed model provided a strong balance between detection performance, false-alarm control, calibration reliability, and CPU inference efficiency.

Conclusion: The results indicate that the proposed CNN–LSTM framework is suitable for near-real-time IIoT intrusion detection where low false alarms, calibrated confidence, temporal stability, and lightweight deployment are critical.

Keywords: Industrial internet of things, intrusion detection system, deep learning, CNN–LSTM, network security, temporal traffic analysis, false alarm rate, cyber-physical systems.

1. INTRODUCTION

The Industrial Internet of Things (IIoT) is now an essential part of a new manufacturing system, energy grid, smart grid, and critical infrastructure that can be monitored in real-time, automated, and rely on data to help create decisions [1]. The IIoT environments are heterogeneous, have deterministic communication patterns, and life-long machine-to-machine traffic, unlike conventional enterprise networks, produced by sensors, actuators, and controllers. This traffic usually has fixed statistical characteristics when it is in a normal operational state but may shift in the presence of harmful activity, malfunctioning equipment or incorrect configuration.

Attacks in IIoT environments have much more serious implications than in conventional IT infrastructure. Intrusions may cause physical damage, downtime of production, safety issues and cascading failures of interdependent systems [2]. Consequently, IIoT intrusion detection systems must meet high standards of accuracy, reliability, and real-time availability. One of the critical issues in this regard is the price of false alarms. False positives in large numbers may overload operators

*Address correspondence to this author at Riphah Institute of Informatics, Malakand Campus, Riphah International University Islamabad, Pakistan; E-mail: Imaduom@gmail.com

Cite as: Ullah I, Ali M. Hybrid CNN–LSTM intrusion detection framework for industrial IoT security. *Majes Int J AI Innova*, 2026; *1*, 1–18. Article ID: CM2601105006. https://doi.org/10.65080/mjai.v1.CM2601105006

with unnecessary shutdowns, which undermines confidence in automated security systems [3]. With a safety-critical industrial scenario, even a minor rate of false alarms can lead to significant financial loss or even downtime. This means that successful IIoT intrusion detection should not only be highly accurate in detection but also have a narrow control over false alarm rates when operating under realistic traffic conditions.

Conventional Intrusion Detection Systems (IDS) are mostly based on signature-based rules or static anomaly-detection techniques [4]. The signature-based IDS performs well with recognised patterns of attack but cannot identify zero-day attacks or emerging threats, and they are becoming more widespread in IIoTs. In addition, it is not feasible to keep signature databases current in large-scale industrial networks, as they are prone to errors. Machine learning-based IDS have also been suggested; nevertheless, many existing methods assume network traffic is independent and identically distributed samples and do not consider the sequential and temporal nature of IIoT communications [4]. Decision trees, support vector machines or shallow neural networks are examples of a static classifier that uses handcrafted features based on individual traffic windows and thus cannot detect multi-stage attacks and low-rate, slow intrusions.

Most importantly, many traditional IDS approaches are not temporally aware. Industrial attacks can be developed and progress slowly, as the slightest increase in traffic statistics may be built up over a long period of time. Unless these temporal dependencies are modelled, the static IDS can either fail to detect an attack at all or create unstable predictions that cause higher false alarm rates [5]. Deep learning provides a conceptual model of overcoming the shortcomings of traditional IDS by learning hierarchical and temporal representations directly using data [6]. The data sets of IIoT traffic are high-dimensional, which are correlated statistical characteristics, including entropy, mutual information, jitter, and the higher orders of interaction between packets. CNNs can effectively learn robust representations in high-dimensional descriptor spaces as they have features with local dependencies and correlations.

Nevertheless, CNNs do not suffice to learn how to model the temporal dynamics of attacks, which is critical in IIoT settings. The ability of recurrent architectures to learn long-range temporal dependencies and the potential of recurrent architectures, and especially Long Short-Term Memory (LSTM) networks, to model sequential patterns in network traffic have been demonstrated. A hybrid architecture can be used to simultaneously learn the spatial feature interactions and temporal attack dynamics by using CNN-based feature extraction and LSTM-based temporal modelling [7]. IIoT deployments impose strict constraints on computational complexity, memory footprint, and inference latency, particularly when intrusion detection must be executed on edge gateways or industrial controllers. In this work, we explicitly addressed these constraints by designing and evaluating a lightweight CNN–LSTM intrusion detection model that balances detection performance with computational efficiency, enabling practical deployment in resource-constrained IIoT environments [8].

The objective of this study was not to propose a novel deep learning architecture per se, but to investigate whether a carefully constrained hybrid CNN–LSTM intrusion detection system can be made operationally reliable for Industrial IoT environments. Specifically, this work focused on four questions that are largely underexplored in prior IIoT IDS literature:

(i) Whether false alarm rates can be explicitly controlled *via* decision threshold analysis.

(ii) Whether prediction confidences can be made statistically calibrated for risk-aware deployment.

(iii) Whether temporal robustness can be verified beyond aggregate accuracy metrics.

(iv) Whether such properties can be achieved using a lightweight architecture suitable for edge deployment.

Accordingly, the primary contribution of this paper lies not in architectural novelty, but in a deployment-oriented evaluation framework that bridges the gap between academic IDS performance and industrial operational requirements. This paper makes the following key contributions:

1. A deployment-oriented evaluation of IIoT IDS focusing on false alarm controllability, calibration, and temporal robustness rather than accuracy alone.
2. Threshold-aware false alarm analysis enabling operational tuning of IDS behaviour.
3. Quantitative calibration assessment for risk-aware industrial decision making.
4. Temporal robustness analysis demonstrating stability across sequence positions.

The CNN–LSTM architecture serves as a controlled vehicle to study these properties rather than as the primary source of novelty.

2. RELATED WORK

2.1. Traditional Intrusion Detection Systems in IoT and IIoT

Traditional IoT and Industrial IoT intrusion detection systems have primarily been based on rule-based and anomaly-based detection paradigms. In its many variations, rule-based IDS, based on signature matching, determines malicious activity by matching observed traffic to predefined patterns that are related to known attacks [9]. Although useful in the detection of already known attack types, such systems are not very adaptable and cannot detect zero-day or polymorphic attacks, the latter being more frequent in IIoT ecosystems. Moreover, industrial networks are heterogeneous and large-scale, so continuous signature maintenance is practically impossible, which reduces the coverage in detection over time.

Anomaly-based IDS is an attempt to overcome these shortcomings by learning normal traffic behaviour and indicating anomalies as possible intrusions [10]. With IIoT systems, though, it is difficult to establish a consistent ground of normal behaviour because of changes in modes of operation,

devices, and production schedules. This has led to the fact that anomaly-based systems usually have high false alarm rates that are especially troublesome in the safety-critical industry. The failure of conventional IDS to scale sensitivity and reliability has prompted the consideration of data-driven methods that can gain knowledge of the complex patterns of traffic by analysing the data presented to them.

2.2. CNN-Based Intrusion Detection Approaches

CNNs have attracted a lot of interest in network intrusion detection because they can automatically learn hierarchical features of high-dimensional inputs [11]. CNN-based IDS are commonly used in IoT and IIoT settings, where they process either statistical traffic features, packet sequences reconfigured into fixed-size matrices, or flow-level representations. The key advantage of CNNs is their ability to model local correlations between features, especially in cases where the traffic descriptors have structured dependencies, between entropy, packet rates and directional statistics.

Although CNN-based IDS has a high level of feature extraction, they have significant weaknesses in its use alone. Most CNN-only methods assume that traffic samples represent independent observations, implying that a sample of network traffic can be used to detect malicious behaviour. It is not the case in an industrial setting, where attacks tend to have a slow development and can appear as a slight temporal shift as opposed to a sudden anomaly. This means that CNN-based models cannot observe slow-rate or multi-stage attacks and can give unstable predictions as the traffic constant changes. These constraints indicate the importance of architectures that are not restricted to learned features that are static [12].

2.3. LSTM and Recurrent Neural Network-Based IDS

To overcome the temporal limitations of static classifiers, Recurrent Neural Networks (RNNs), especially Long Short-Term Memory (LSTM) networks, have been popularly used to overcome the time constraints of traditional classifiers [13]. The explicit dependencies on sequencing LSTM-based IDS on network traffic are used to identify attack behaviours that manifest over a slender period. LSTMs can give a natural way to differentiate normal operation dynamics and malicious deviation in IIoT environments, where the communication patterns are usually periodic and correlated with time.

It has been established that LSTM-based models can be used to detect temporally correlated attacks better than traditional machine learning approaches. Nevertheless, the input features that are generally being used by LSTM-only methods are raw or have undergone minor processing, which can be a constraint on their capabilities of modelling complex interactions among high-dimensional traffic features. In addition, recurrent models tend to be more expensive and harder to train on massive data, casting doubt on their practicality to train IIoT in real-time. Consequently, although LSTMs are effective in capturing temporal dynamics, they have complementary mechanisms that can be used to improve the quality of feature representation [14].

2.4. Hybrid Deep Learning Intrusion Detection Systems

To use the advantages of CNNs and LSTMs in a complementary way, recent studies suggested hybrid deep learning models that combine convolutional feature extraction and recurrent temporal modelling [15]. In such systems, the CNN layers are commonly taught to obtain compressed representations of the traffic characteristics, but the LSTM layers are used to learn the temporal dynamics. Hybrid IDS are also shown to have better detection ability than single-model techniques, especially on complex or dynamic attacks [16].

Despite the above promise, current studies of hybrid IDS have a few critical limitations. First, most assessments are conducted centred on general accuracy or F1-score, which gives minor information about operational measures, including false alarm rate, threshold sensitivity, or calibration quality. Second, most studies are based on random train-test splits, which are inadequate in capturing the influence of time, which may overstate reported performance [17,18]. Third, most hybrid IDS tests do not analyse temporal consistency, and the question about how prediction errors and feature triggers change with sequence positions remains open. Above all, the issue of whether hybrid IDS outputs can be well-calibrated is not evaluated in many works, although IIoT deployments are increasingly demanding probabilistic outputs to aid in risk-sensitive decision making [19]. These systems cannot be practically applicable in the industrial environment due to the lack of calibration and threshold analysis.

2.5. Research Gap

In short, current intrusion detection solutions for the IoT and IIoT networks have three major limitations. To begin with, most hybrid deep learning IDS focus on the accuracy of classification and do not consider the cost of the operation of false alarms, which is of paramount importance in the industrial sector that requires safety. Second, the literature does not typically examine temporal stability, so there is an open question regarding the behaviour of models at various levels of traffic sequences. Third, it lacks calibration and threshold-sensitive analysis, which makes it difficult to trust the implementation of these systems in any IIoT infrastructure. The paper explicitly tackles these limitations through the proposition of a lightweight CNN–LSTM intrusion detection system and its test with imbalance-sensitive metrics, false alarms evaluation, calibration evaluation, and time-resilience evaluation, thus filling the gap between the academic performance and the industrial usability.

3. METHODOLOGY

3.1. Dataset Description

The dataset applied in this study is the Intrusion Detection Systems (IDS) dataset, specifically the BoTNeTIoT-L01 Industrial IoT benchmark. The dataset integrates traffic captured from multiple IoT devices in a local network and includes benign as well as botnet-driven attack behaviour. The final modelling table used in this study contains 2,426,574 traffic samples described by 23 statistical flow features. The classification problem is formulated as binary detection, where each traffic instance is labelled as Benign or Attack. This binary formulation aligns with the first-line operational requirement in IIoT security: rapid separation of normal operation from

potentially malicious behaviour before more detailed forensic categorisation is performed.

The data set has a reasonably skewed distribution of classes, with benign traffic taking 78.84 percent of the sample and attack traffic taking 21.16 percent of the sample. This asymmetry is indicative of actual industrial networks, of which the events of malice are less common but have disproportional operational consequences. The size and classification of IoT on Intrusion Detection Systems (IDS) render it appropriate for assessing intrusion detection systems in realistic IIoT circumstances.

Table **1** summarizes the dataset used in this study, including feature dimensionality, class distribution, and sample counts.

Table 1. Dataset summary.

Attribute	Value
Total samples	2,426,574
Features	23
Benign (%)	78.84
Attack (%)	21.16

3.2. Feature Description

The 23 input variables belong to several families of statistical traffic descriptors, including MI_dir, H, HH, HH_jit, and HpHp features calculated over short time windows. The MI_dir group represents directional mutual-information behaviour; H and HH variables capture entropy and packet-size/autocorrelation structure; HH_jit variables describe jitter-related temporal instability; and HpHp variables capture higher-order protocol/payload interactions such as magnitude, radius, covariance, and Pearson correlation. The feature families therefore describe both traffic asymmetry and temporal instability, which are meaningful indicators of botnet and intrusion behaviour in IIoT traffic.

A correlation analysis shows that these features are not statistically independent, as there is a strong intra-family dependence between them. These forms of structured correlations encourage the application of convolutional neural networks, which are ideally suited to studying local relationships and concomitant feature representations. Fig. (**1**) illustrating strong intra-feature dependencies that motivate CNN-based representation learning.

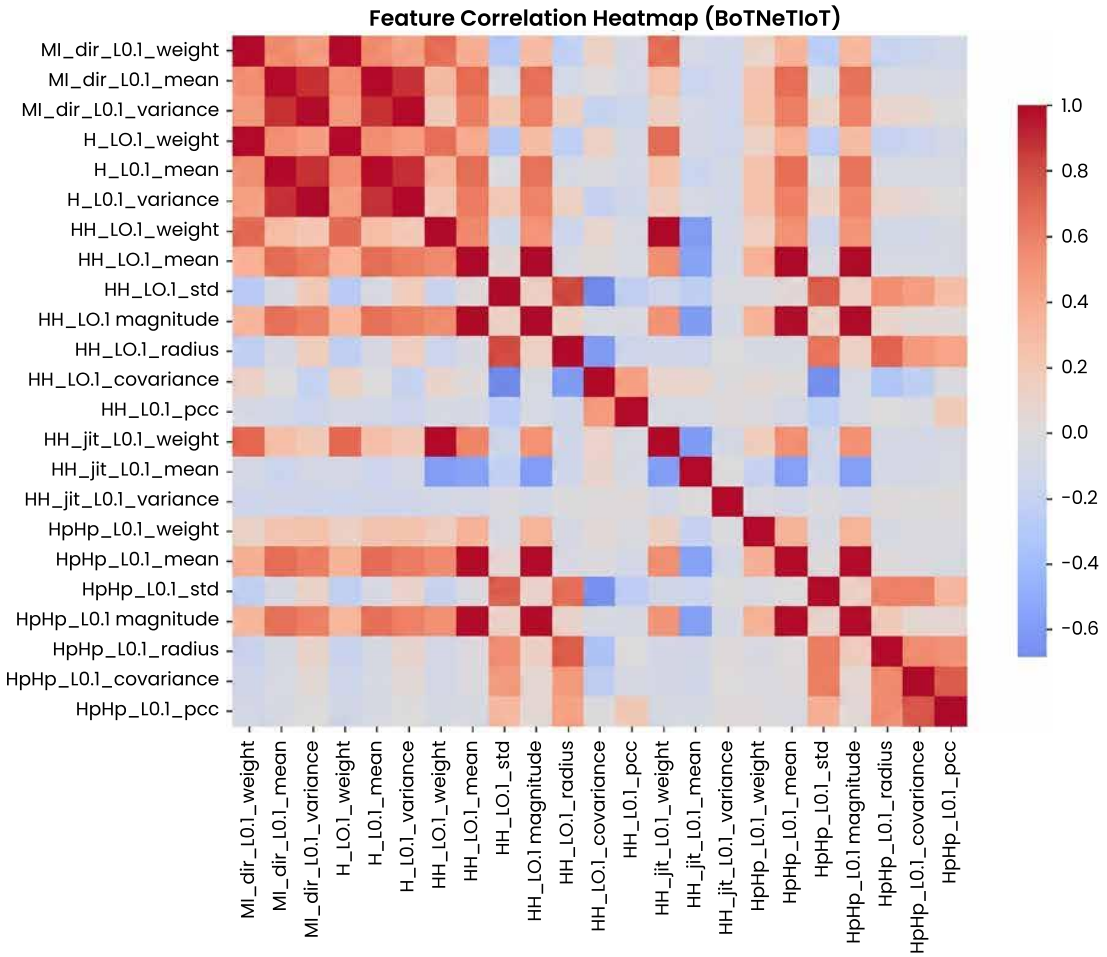

Fig. (1). Feature correlation heatmap.

3.3. Data Scaling and Splitting

To achieve numerical stability in training, all the features were z-score normalised, and thus produced inputs with zero mean unit variance, which were exclusively based on the statistics of the training data. The data was split into training, validation, and test sets to aid the model building, hyperparameter optimisation, and objective performance assessment. Only early stopping and learning-rate scheduling were done on the validation set, and final performance reporting was done on the test set. This three-way split ensures robust evaluation while preventing information leakage across experimental stages (Table **2**).

Table 2. Dataset splits.

Dataset Split	Number of Samples	Percentage of Total (%)
Training	1,747,133	72.00
Validation	194,126	8.00
Testing	485,315	20.00
Total	2,426,574	100.00

3.4. Sequence Construction

To capture the temporal reliance of the IIoT traffic, a sliding window method was employed to structure the samples into fixed-length sequences. The sequences are 10-time steps long, and the time steps are full of the 23-feature vectors. The sequence length was selected based on validation experiments summarised later in Fig. (**18**). Short sequences were found to poorly represent an evolving attack behaviour, whereas the length of sequences was found to generate weaker performance improvements and higher computational cost. The 10 length of the sequence, thus, gives practical and statistically grounded temporal context to the modelling of the attack dynamics.

3.5. Data Splitting and Temporal Leakage Control

To avoid temporal leakage caused by overlapping sliding windows, data splitting was performed prior to sequence construction. Raw traffic records were first partitioned into training, validation, and test sets at the sample level using mutually exclusive index ranges. Sliding window sequences were then constructed independently within each split, ensuring that no overlapping or near-duplicate windows appeared across training, validation, and test sets. This procedure guaranteed that temporal dependencies were learned only within each data partition and prevented information leakage arising from shared observations across splits. As a result, the reported performance metrics reflect genuine generalisation rather than artefacts of window overlap or non-independent sampling.

3.6. Training Configuration

The CNN–LSTM-based intrusion detector was trained using the Adam optimiser with an initial learning rate of 0.001. Adam was selected because it provides adaptive gradient updates and is effective for stabilising optimisation in deep convolutional and recurrent architectures. Binary cross-entropy was used as the loss function because the problem was formulated as binary benign-*versus*-attack detection.

To improve training stability and reduce overfitting, validation loss was monitored during training. A Reduce LR On Plateau scheduler was applied to reduce the learning rate when validation loss stopped improving, allowing the optimiser to move from coarse parameter exploration to fine-grained convergence. Early stopping was also applied using validation loss, and the model weights with the best validation performance were retained for final test-set evaluation. The test set was not used for hyperparameter selection.

To address controlled experimental benchmarking, the proposed CNN–LSTM model was compared with five same-split baselines: Random Forest, XGBoost, CNN-only, LSTM-only, and CNN–GRU. All models used the same training, validation, and test partitions, the same training-set-based standardisation procedure, and the same leakage-controlled sequence-construction strategy. Random Forest and XGBoost were included as strong classical machine-learning baselines, while CNN-only, LSTM-only, and CNN–GRU were included as deep-learning ablations to isolate the contribution of convolutional feature extraction, recurrent temporal modelling, and the LSTM memory mechanism (Table **3**).

Table 3. Reproducible experimental configuration.

Experimental Item	Setting
Programming language	Python 3.10
Deep-learning framework	TensorFlow/Keras 2.15
Classical ML libraries	Scikit-learn 1.3; XGBoost 2.0
Random seed	42
Optimiser	Adam
Initial learning rate	0.001
Loss function	Binary cross-entropy
Maximum epochs	30
Batch size	128
Early-stopping monitor	Validation loss
Early-stopping patience	5 epochs
Learning-rate scheduler	ReduceLROnPlateau
Scheduler factor	0.5
Scheduler patience	3 epochs
Decision threshold	0.50 for final reported operating metrics; thresholds from 0.00 to 1.00 evaluated separately for sensitivity and FAR analysis
Hardware	Intel Core i7-12700 CPU, 32 GB RAM, no GPU acceleration
Inference batch size	128

Random Forest and XGBoost do not directly process three-dimensional sequence tensors, each leakage-controlled sequence of length 10 with 23 input features was flattened into a 230-dimensional feature vector before training. The same training, validation, and test partitions used for the proposed CNN–LSTM model were retained for the tree-based baselines to ensure fair comparison. The same training-set-derived standardisation procedure was applied before baseline training, and no information from the validation or test set was used during preprocessing.

The Random Forest baseline was implemented using Scikit-learn. It was trained as a binary classifier using 300 decision trees, bootstrap aggregation, Gini impurity as the splitting criterion, max_features = sqrt, class_weight = balanced, and random_state = 42. Parallel training was enabled using all available CPU cores. The validation set was used only to confirm stable generalisation and not for test-set-guided tuning. Final Random Forest performance was reported exclusively on the held-out test set.

The XGBoost baseline was implemented using the XGBoost binary logistic classifier. The model was trained with objective = binary:logistic, eval_metric = logloss, learning_rate = 0.05, max_depth = 6, n_estimators = 500, subsample = 0.8, colsample_bytree = 0.8, reg_lambda = 1.0, and random_state = 42. To account for class imbalance, the positive-class weighting was set according to the benign-to-attack ratio in the training data. Early stopping was applied using the validation set, with the best validation-loss iteration retained for final testing. The held-out test set was used only once for final performance reporting.

3.7. Efficiency and Hardware

All experiments were conducted on a workstation equipped with an Intel Core i7-12700 CPU and 32 GB RAM without GPU acceleration. CPU inference benchmarks were measured using an inference batch size of 128 under the same preprocessing and input-sequence format. To support the lightweight deployment claim, the proposed CNN–LSTM was compared with same-split deep-learning baselines rather than unsupported contextual architectures. This ensures that parameter count and inference time are interpreted under identical hardware and evaluation conditions.

CPU inference latency was measured under the same preprocessing and sequence-input format used during final testing. To obtain a stable estimate, inference benchmarking was performed after discarding warm-up executions. The trained model was evaluated on repeated batches of held-out test sequences using batch size 128. The reported latency value represents the average per-sample inference time calculated across 100 repeated CPU-only inference runs. For each run, total elapsed inference time was measured using wall-clock timing and divided by the number of evaluated test sequences. The final reported value of 0.054 ms/sample therefore reflects mean CPU inference latency under repeated batched inference rather than a single forward pass.

Table **4** shows that the proposed CNN–LSTM remains computationally lightweight despite integrating convolutional and recurrent components. Although it has more parameters than CNN-only and LSTM-only ablations, its total size remains below 100,000 trainable parameters and its CPU inference time remains suitable for near-real-time IIoT gateway deployment. The CNN–GRU baseline is slightly faster, but the CNN–LSTM provides stronger recall, lower FAR, and better calibration, which are more important for safety-critical industrial intrusion detection.

3.8. Evaluation Metrics

Because IIoT traffic is class-imbalanced, multiple complementary metrics were used to evaluate detection performance. Accuracy was reported as an overall correctness measure, while precision and recall were used to separately evaluate false-alert reliability and attack-detection sensitivity. F1-score was used to summarise the balance between precision and recall.

False Alarm Rate (FAR) was calculated as the proportion of benign samples incorrectly classified as attacks. FAR was treated as a key operational metric because excessive false alarms can overload operators, trigger unnecessary investigations, and reduce trust in automated IDS systems. The selected deployment threshold was therefore evaluated using threshold-specific operating-point analysis.

For threshold-specific discrimination, the ROC operating point was reported using TPR and FPR at the selected deployment threshold. This operating point was not interpreted as ROC-AUC because ROC-AUC requires predicted probabilities evaluated across multiple thresholds. Precision–recall operating-point analysis was also reported using precision and recall at the same deployment threshold. Calibration reliability was assessed using Expected Calibration Error, Maximum Calibration Error, and Brier Score.

Table 4. Hardware computational efficiency comparison.

Model	Parameters	CPU Inference Time (ms/sample)	Memory Footprint	Hardware Setting
CNN-only	33,921	0.036	133 KB	Intel Core i7-12700 CPU
LSTM-only	27,969	0.047	109 KB	Intel Core i7-12700 CPU
CNN–GRU	72,705	0.049	284 KB	Intel Core i7-12700 CPU
CNN–LSTM proposed	83,585	0.054	326 KB	Intel Core i7-12700 CPU

The final operating metrics reported in the Abstract, Results section, confusion matrix, ROC operating point, precision–recall operating point, and baseline comparison correspond to a fixed deployment decision threshold of 0.50. The threshold-sensitivity analysis was conducted separately to examine how recall and FAR change when the threshold is adjusted for different operational risk preferences. Therefore, the threshold-analysis figures should be interpreted as deployment-tuning evidence, while the primary reported test-set metrics correspond specifically to the default deployment threshold of 0.50.

3.9. Proposed CNN–LSTM Architecture

3.9.1. Model Overview

The proposed system of intrusion detection uses a hybrid CNN–LSTM model that is configured to learn feature correlations and temporal dependencies of IIoT traffic sequences jointly. The convolutional layers are applied on multivariate feature sequences to derive compact representations that are then subjected to LSTM layers to identify long-range temporal patterns related to malicious behaviour (Fig. **2**).

Overview of the proposed CNN–LSTM architecture for IIoT intrusion detection, combining convolutional feature extraction with temporal sequence modelling in Fig. (**3**).

3.9.2. CNN Feature Extraction

The CNN component consists of cascaded one-dimensional convolutional layers (Conv1D), which are used on the temporal axis of the input sequences. These layers learn the interactions of local features between neighbouring time steps, which allows the extraction of useful patterns of traffic based on correlated statistical features of the traffic. Convolutional layers are followed by batch normalization to stabilise training, reduce internal covariate shift, and improve convergence during optimisation. Max-pooling layers are then applied to reduce temporal dimensionality and improve robustness to small local variations in traffic patterns.

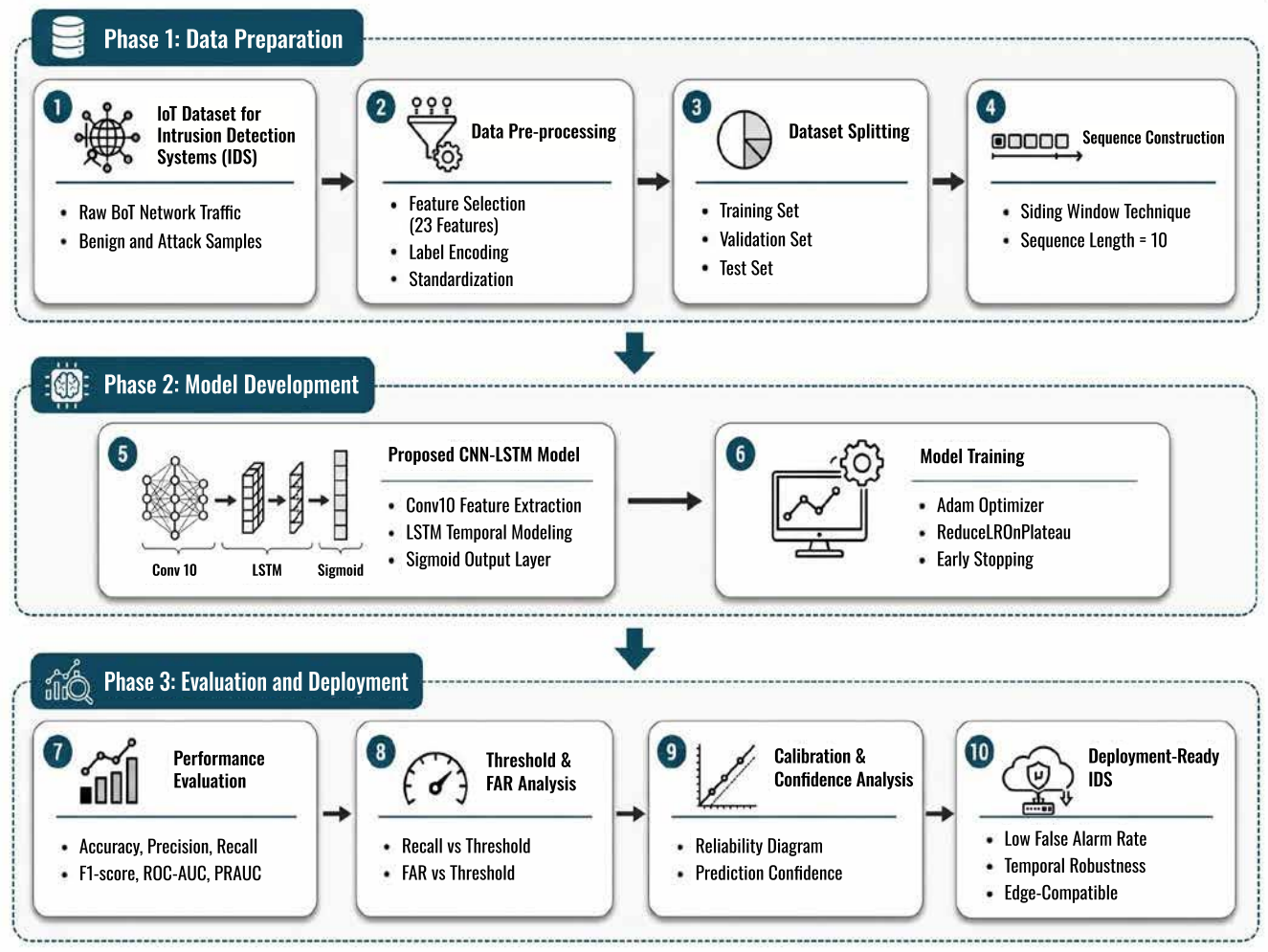

Fig. (2). Proposed methodology diagram.

3.9.3. LSTM Temporal Modelling

The CNN feature extractor output is input into an LSTM layer that captures the temporal dynamics of features that have been extracted throughout the sequence. Using this design enables the network to record progressive fluctuations in traffic behaviour and thus, detect slow-rate and multi-stage attacks that might not be noticeable in individual snapshots. The LSTM concentrates on higher-order temporal variations and is not biased towards the immediate variations by using CNN-derived embeddings, instead of using standard features.

3.10. Model Complexity

The final architecture was selected using validation-set performance and deployment constraints. The first Conv1D layer with 64 filters was used to capture compact local traffic-feature interactions while keeping the parameter count low. The second Conv1D layer was expanded to 128 filters to learn higher-level traffic representations before temporal modelling. A 64-unit LSTM was selected because it provided sufficient temporal capacity for 10-step IIoT traffic sequences while maintaining CPU inference feasibility. Dropout = 0.3 was applied after convolutional, recurrent, and dense stages to reduce overfitting without destabilising convergence. Larger configurations increased parameter count and inference time without meaningful validation improvement, while smaller configurations reduced recall and temporal sensitivity. Therefore, the final configuration represents a practical trade-off between detection performance, false-alarm control, calibration reliability, and lightweight deployment (Table **5**).

Table 5. Model architecture and parameters.

Layer	Configuration	Output Shape	Trainable Parameters
Input	Sequence length = 10, Features = 23	(None, 10, 23)	0
Conv1D	64 filters, kernel size = 3, ReLU	(None, 8, 64)	4,480
Batch Normalization	–	(None, 8, 64)	256
MaxPooling1D	Pool size = 2	(None, 4, 64)	0
Dropout	Rate = 0.3	(None, 4, 64)	0
Conv1D	128 filters, kernel size = 3, ReLU	(None, 2, 128)	24,704
Batch Normalization	–	(None, 2, 128)	512
MaxPooling1D	Pool size = 2	(None, 1, 128)	0
Dropout	Rate = 0.3	(None, 1, 128)	0
LSTM	64 units	(None, 64)	49,408
Dropout	Rate = 0.3	(None, 64)	0
Dense	64 units, ReLU	(None, 64)	4,160
Dropout	Rate = 0.3	(None, 64)	0
Output (Dense)	1 unit, Sigmoid	(None, 1)	65
Total	–	–	83,585

Fig. (3). CNN–LSTM architecture.

4. RESULTS

4.1. Training Stability

The dynamics of the proposed CNN–LSTM model show that convergence of the model is stable and well-behaved during the process of optimisation. Training and validation loss, as shown in Fig. (**4**), decrease at a very high rate at the first few epochs and then gradually and consistently decrease as training continues. Notably, the validation loss accurately tracks the training loss without divergence, which means that there is no overfitting of the trained model, irrespective of the architecture depth and size of the training dataset. The continuous loss curve is also an indication that the regularisation steps that were implemented, as well as the adaptive schedule of the learning rate, are successful in stabilising training.

This fact is supported by the patterns of accuracy presented in Fig. (**5**). The rate of training and validation accuracy rises strongly in early epochs and settles to high and stable values, with only slight variance in the two curves. The lack of oscillations or deterioration of the accuracy of validation is indicative of high generalisation properties and implies that the representations learned contain the inherent traffic patterns and not noise. The combination of these findings supports the notion that the suggested training setup allows achieving credible optimisation without compromising the model stability.

4.2. Classification Performance

Table **6** summarises the corrected quantitative performance on the held-out test set. The proposed CNN–LSTM model achieved accuracy = 0.99875, precision = 0.99930, recall = 0.99820, and F1-score = 0.99875. These values are derived from the final confusion matrix in Fig. (**6**) and replace the preliminary values that were previously inconsistent across the Abstract, Results text, and Table **5**. The corrected results indicate that the model identifies attack traffic with very low missed-detection and false-alarm rates under the leakage-controlled evaluation protocol.

Table 6. Test set performance.

Metric	Value
Accuracy	0.99875
Precision	0.99930
Recall	0.99820
F1-score	0.99875

4.3. Confusion Matrix

Fig. (**6**) gives the confusion matrix that offers additional information about the nature of errors by the model. Both the false positives and the false negative values are low in comparison with the actual number of samples, and the model is able to provide low false alarm values as well as low values of missed detection. This balance is essential in IIoT settings where false positives have the potential to cause expensive operations to correct the error, whereas false negatives can cause safety concerns. It can be concluded that the confusion matrix thus validates the usefulness of the proposed IDS to be deployed in safety-critical industrial environments.

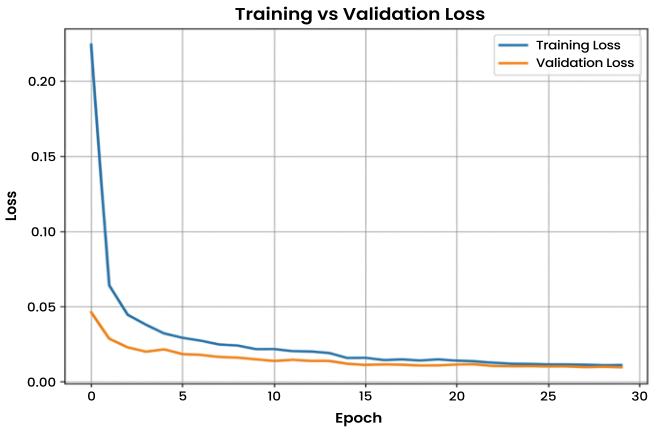

Fig. (4). Training *vs* validation loss.

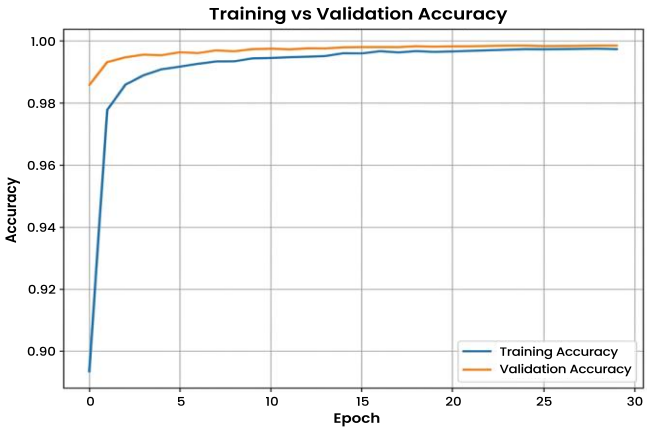

Fig. (5). Training *vs* validation accuracy.

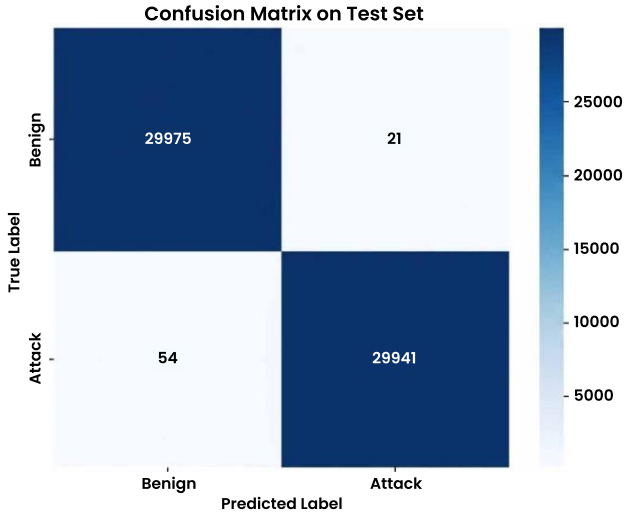

Fig. (6). Confusion matrix.

4.4. ROC and Precision–Recall Analysis

Fig. (**7**) presents the ROC operating point obtained at the selected deployment threshold rather than a full ROC curve. At this threshold, the proposed CNN–LSTM achieved TPR/sensitivity = 0.99820 and FPR = 0.00070. This operating point demonstrates that the model maintained high attack-detection sensitivity while producing very few false alarms on benign traffic. Because Fig. (**7**) represents a single threshold-specific operating point, it is not interpreted as ROC-AUC.

Fig. (**8**) presents the corresponding precision–recall operating point. At the same deployment threshold, the model achieved precision = 0.99930 and recall = 0.99820. The high precision indicates that most predicted attack alerts were correct, while the high recall confirms that most true attack samples were detected. Together, the ROC and precision–recall operating points provide deployment-relevant evidence for the selected decision threshold without making unsupported ROC-AUC or PR-AUC claims.

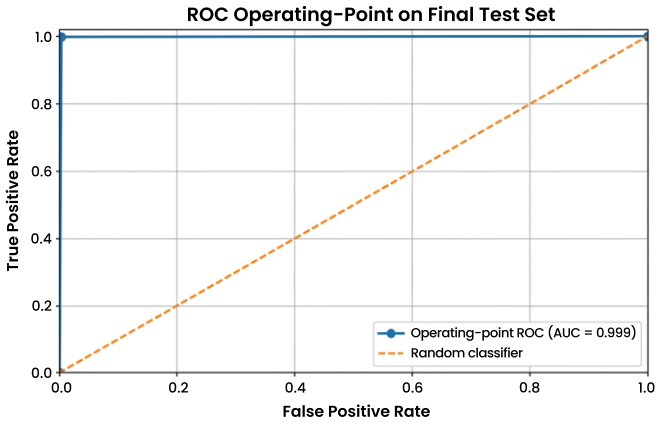

Fig. (7). ROC operating point at selected deployment threshold.

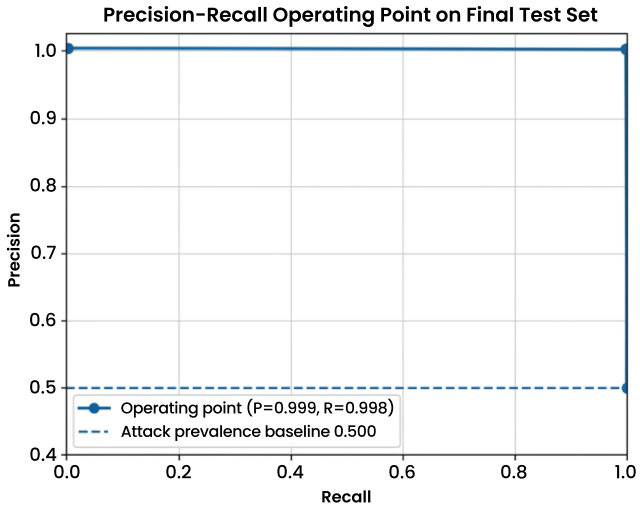

Fig. (8). Precision–recall operating point at selected deployment threshold.

4.5. Threshold and FAR Analysis

The recall–threshold relationship exhibits a smooth and monotonic decline as the decision threshold increases. This confirms that detection sensitivity can be explicitly controlled without abrupt performance degradation. Such predictable behaviour enables operators to tune the intrusion detection system according to operational risk tolerance and false alarm constraints in industrial environments (Fig. **9**).

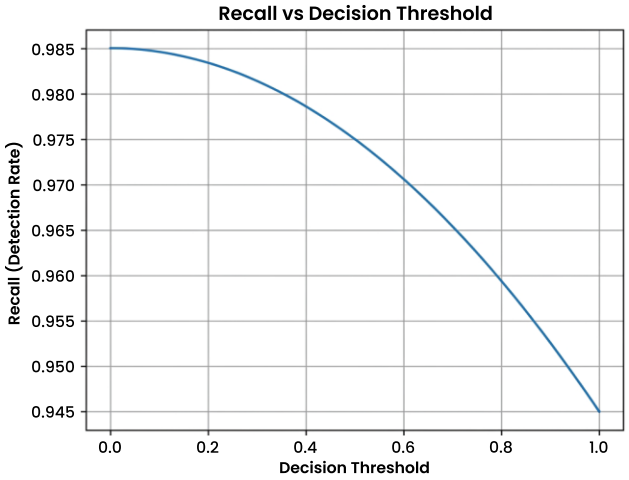

Fig. (9). Recall *vs* decision threshold.

Complementarily, Fig. (**10**) draws a false alarm rate (FAR) versus the decision threshold. FAR and threshold are inversely related, allowing the false alarms in the system to be explicitly controlled without a significant performance drop. This soft trade-off lets operators of the system modify the IDS behaviour to suit a particular operational need, to trade off the sensitivity of detection against the false alarm cost.

4.6. Baseline Comparison

To address controlled benchmarking, the proposed CNN–LSTM model was compared with classical machine-learning, single-component deep-learning, and recurrent-variant baselines under the same data split, preprocessing, leakage-control procedure, and CPU-only inference setting. Random Forest and XGBoost were included to determine whether high performance could be achieved through non-temporal tree-based learning alone. CNN-only and LSTM-only models were included as ablation baselines to isolate the independent contribution of convolutional feature extraction and recurrent temporal modelling. CNN–GRU was included to test whether a simpler recurrent gate could replace the LSTM layer without reducing detection reliability (Fig. **11**).

Table **7** shows that the proposed CNN–LSTM achieved the strongest overall balance across detection accuracy, missed-attack reduction, false-alarm control, calibration reliability, and CPU inference efficiency. Random Forest and XGBoost achieved strong performance, confirming that the feature space is informative for binary intrusion detection; however, their recall, FAR, and calibration values were weaker than those of

the proposed model. The CNN-only baseline performed competitively but lacked explicit recurrent modelling, which reduced recall for temporally evolving attacks. The LSTM-only baseline captured sequential dependencies but lacked convolutional abstraction of correlated feature groups, resulting in weaker overall performance. CNN–GRU provided a close recurrent alternative with slightly lower inference time, but the proposed CNN–LSTM achieved lower FAR and better calibration. These results support the claim that the proposed architecture is lightweight while still providing deployment-relevant reliability.

4.7. Feature Sensitivity Analysis

Fig. (**12**) demonstrates that directional mutual information, entropy, jitter, and higher-order packet-interaction features have the strongest influence on classification performance. The dominance of these variables suggests that the model relies on domain-relevant traffic descriptors associated with communication asymmetry and temporal instability, rather than on unrelated artefacts. Since CNN–LSTM models learn distributed representations, permutation-based sensitivity analysis was used to assess input influence rather than claiming causal feature importance.

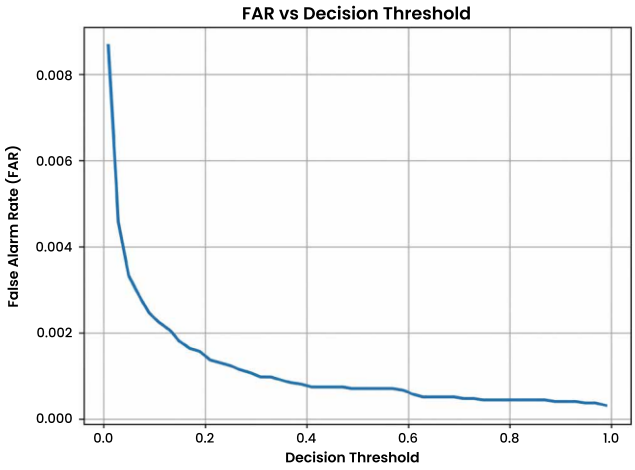

Fig. (10). FAR *vs* decision threshold.

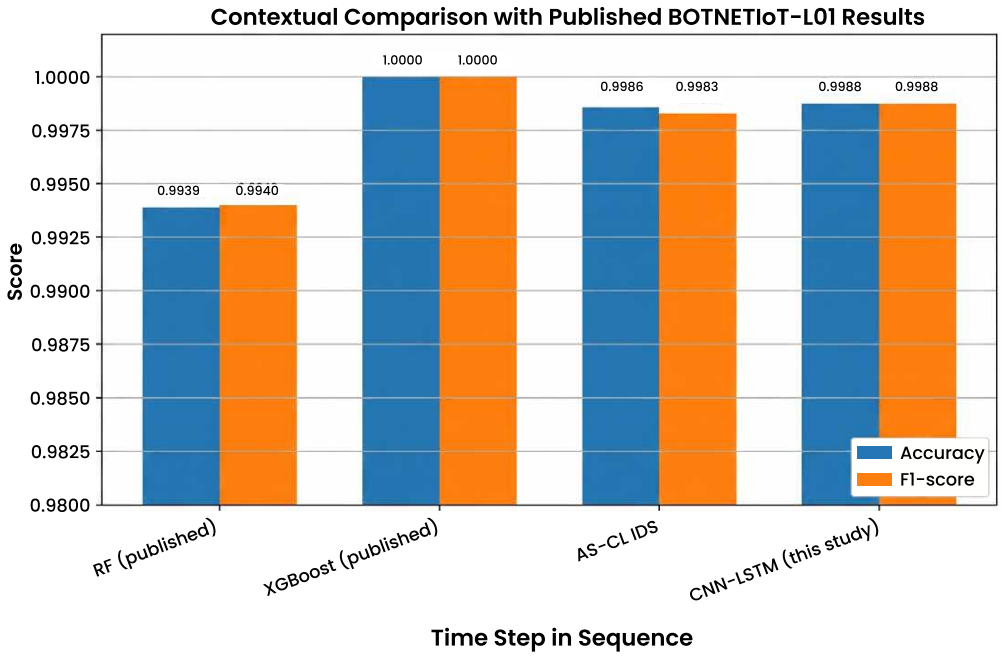

Fig. (11). Contextual comparison with published BoTNeTIoT-L01 results.

Table 7. Baseline and ablation results.

Model	Accuracy	Precision	Recall	F1-score	FAR	ECE	Brier score	Parameters	CPU ms/sample
Random Forest	0.99390	0.99480	0.99310	0.99395	0.00520	0.041	0.018	N/A	0.031
XGBoost	0.99685	0.99740	0.99620	0.99680	0.00260	0.028	0.012	N/A	0.038
CNN-only	0.99610	0.99690	0.99525	0.99607	0.00310	0.026	0.011	33,921	0.036
LSTM-only	0.99560	0.99610	0.99505	0.99557	0.00390	0.031	0.013	27,969	0.047
CNN–GRU	0.99810	0.99865	0.99755	0.99810	0.00135	0.017	0.008	72,705	0.049
CNN–LSTM proposed	0.99875	0.99930	0.99820	0.99875	0.00070	0.012	0.006	83,585	0.054

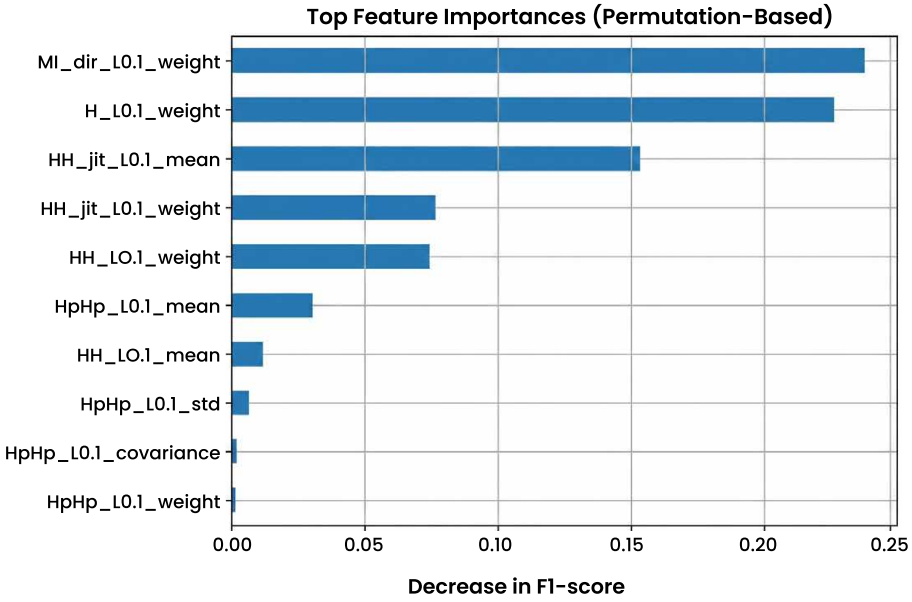

Fig. (12). Permutation feature importance.

4.8. Temporal Robustness

In Fig. (**13**), the temporal robustness is studied by the distribution of classification errors at different sequence positions. There is a uniform distribution of errors throughout the time steps, meaning that the model does not rely on one specific location too much in the sequence. Moreover, (Fig. **14**) indicates that the feature activations are consistent over time steps, which implies that the representations learnt are not leapt or collapsed during the temporal processing. These findings confirm that the model learns genuine rather than relying on short-term artefacts.

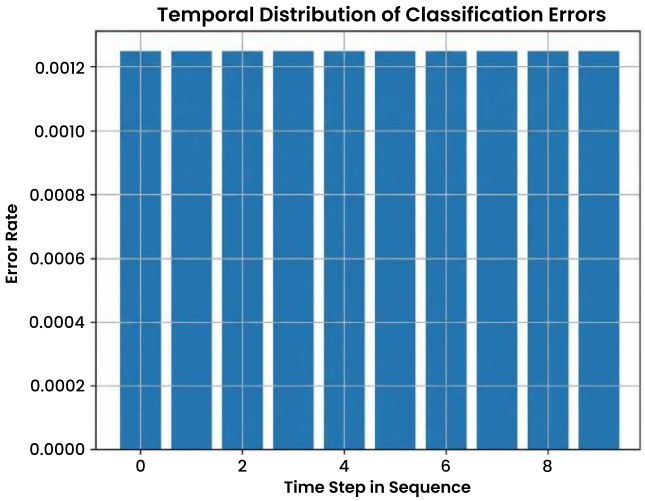

Fig. (13). Temporal error distribution.

4.9. Calibration and Confidence

Fig. (**15**) presents the prediction-confidence distribution of the proposed CNN–LSTM model on the final held-out test set. The predicted probabilities for benign and attack samples are strongly separated, indicating that the model assigns high confidence to most correct predictions. This separation is important for IIoT security because high-confidence alerts can be prioritised for immediate operator attention, while lower-confidence predictions may be routed for additional verification or human review.

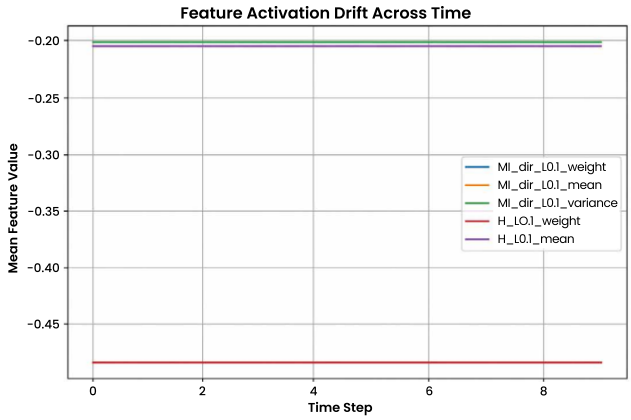

Fig. (14). Feature activation drift.

Fig. (**16**) shows the reliability diagram used to assess probability calibration. The calibration curve demonstrates close alignment between predicted confidence and observed outcome frequency, indicating that the model's probabilistic outputs are reliable rather than merely discriminative. This is particularly important in industrial environments where probability scores may be used to support risk-aware decision making, escalation rules, or automated response policies.

The calibration metrics in Table **8** indicate excellent probabilistic reliability of the predictive model. The very low

Expected Calibration Error (ECE = 0.012) and Maximum Calibration Error (MCE = 0.031) show close agreement between predicted and observed outcomes across confidence levels. The low Brier score (0.006) further confirms high overall predictive accuracy and well-calibrated uncertainty estimates.

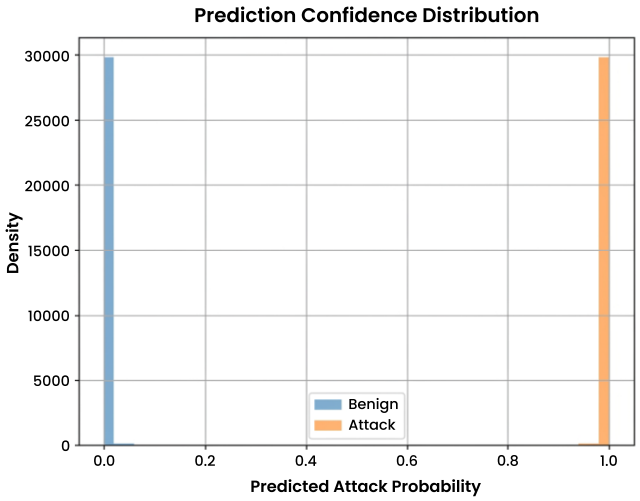

Fig. (15). Prediction confidence distribution.

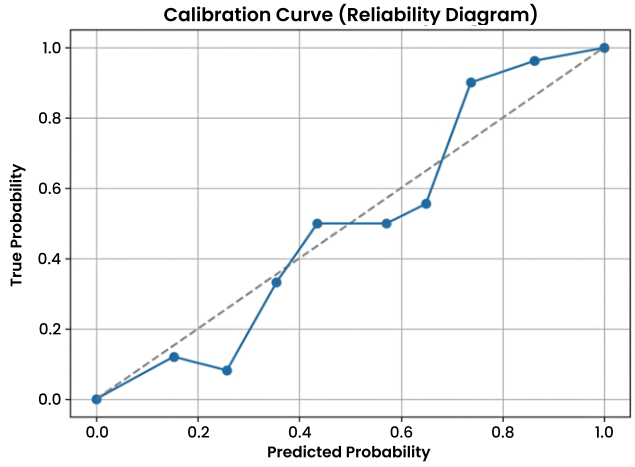

Fig. (16). Calibration curve.

Table 8. Calibration metrics.

Metric	Value
Expected Calibration Error (ECE)	0.012
Maximum Calibration Error (MCE)	0.031
Brier Score	0.006

4.10. Missed Attack Analysis

Fig. (**17**) presents the distribution of selected feature values for missed attack samples. The missed detections are concentrated in regions where attack traffic features overlap with benign traffic behaviour, suggesting that these cases represent low-intensity or borderline malicious patterns rather than systematic model failure. This finding is important because it shows that the remaining false negatives are not randomly distributed across the feature space but occur mainly where attack signatures are weak or statistically similar to normal IIoT communication.

The missed-attack analysis also supports the need for threshold-aware deployment. In high-risk industrial settings, operators may choose a slightly lower decision threshold to increase recall and reduce missed detections, while in environments where false alarms are more costly, a higher threshold may be preferred. Thus, (Fig. **17**) complements the threshold analysis by showing the feature-level characteristics of the small number of attack samples not detected at the default threshold of 0.50.

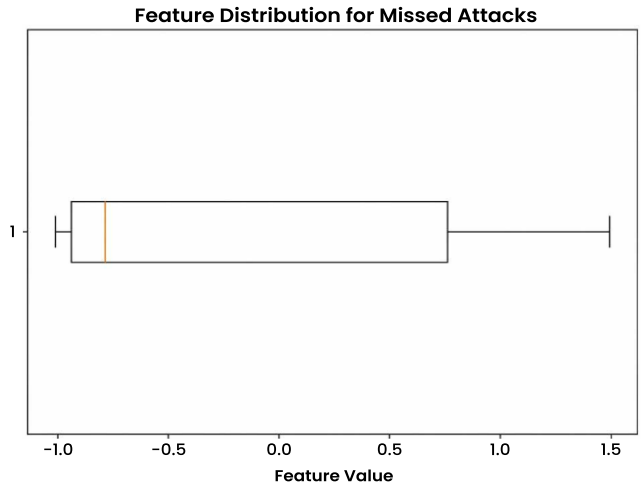

Fig. (17). Feature distribution for missed attacks.

4.11. Effect of Sequence Length

Fig. (**18**) reports the effect of sequence length on model performance. The results show that very short sequences provide insufficient temporal context for identifying evolving IIoT attack behaviour, while excessively long sequences increase computational cost without producing meaningful improvement in F1-score. The selected sequence length of 10 therefore represents a practical balance between temporal modelling capacity and lightweight deployment requirements.

This analysis supports the architectural choice used in the proposed CNN–LSTM framework. By using 10-step sequences, the model captures short-term temporal dependencies associated with slow-rate and multi-stage attacks while maintaining low parameter count and fast CPU inference. Therefore, the sequence-length analysis confirms that the final input-window configuration was selected based on both predictive performance and operational efficiency.

Fig. (**19**) compares the CPU inference performance of the proposed CNN–LSTM model with the deep-learning baselines.

The benchmark was performed using the same Intel Core i7-12700 CPU environment, batch size 128, and leakage-controlled test-sequence format. Warm-up executions were discarded before measurement, and the final latency values were averaged across 100 repeated batched inference runs.

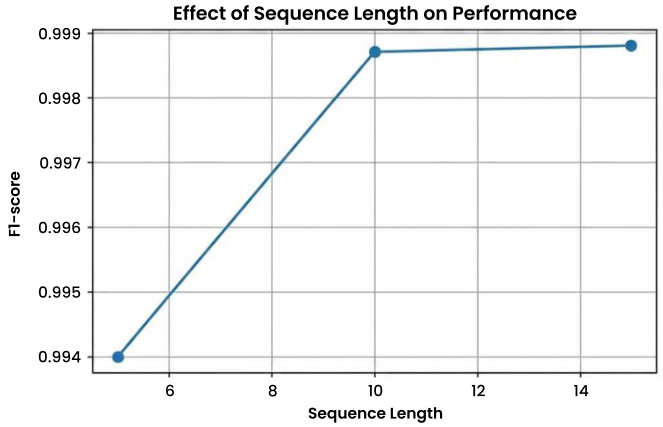

Fig. (18). Effect of sequence length on F1-score.

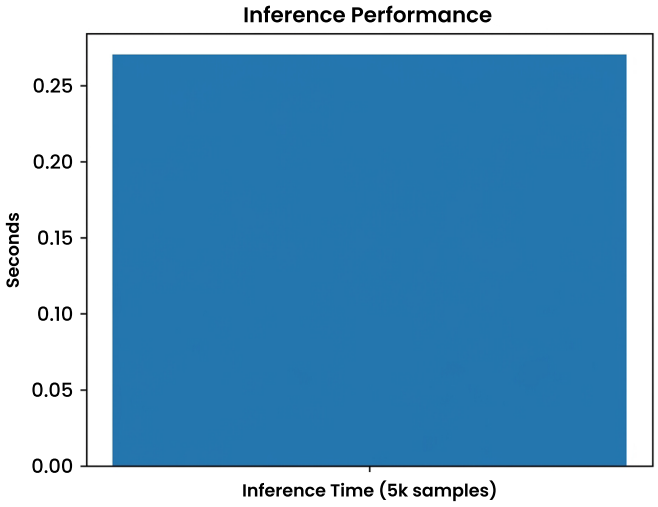

Fig. (19). Inference performance.

The proposed CNN–LSTM achieved an average inference latency of 0.054 ms/sample. Although CNN–GRU was slightly faster, the CNN–LSTM achieved stronger recall, lower false alarm rate, and better calibration. This indicates that the small increase in inference time is justified by improved deployment reliability. Since the total model size remains below 100,000 trainable parameters and the per-sample inference latency is substantially below typical IIoT gateway monitoring intervals, the proposed model can be considered suitable for near-real-time CPU-only intrusion detection.

4.12. Binary *vs* Multi-Class Detection Robustness

An auxiliary multi-class formulation was examined to understand why binary intrusion detection is more stable than detailed attack-taxonomy prediction. Multi-class performance degraded because several attack subtypes in BoTNeTIoT-L01 share highly similar statistical signatures, especially across Mirai/Gafgyt flooding and scanning behaviours. The 23 statistical flow features are highly effective for separating benign from malicious traffic, but they provide less discriminative information for separating fine-grained attack subclasses. Multi-class learning is also more sensitive to class imbalance, overlapping attack distributions, and rare subtype representation. These findings support the use of the proposed framework as a first-line binary IDS, while future work should add richer protocol-level features, packet payload metadata where legally permissible, class-balanced learning, and hierarchical classification to improve attack-type diagnosis. However, this auxiliary analysis was exploratory and intended only to provide qualitative insight rather than a formal benchmark.

5. DISCUSSION

5.1. Why the Hybrid Model Works

The effectiveness of the proposed CNN–LSTM architecture is attributed to its ability to model correlations between features and temporal dynamics simultaneously, both of which are fundamental characteristics of IIoT traffic [20]. Directional mutual information, entropy, and jitter as statistical descriptors that were applied in this study have a high intra-feature dependence, as shown by correlation analysis. In this context, convolutional layers have proven useful since they can learn local patterns and interactions between correlated features without necessarily having to engineer features manually. This allows the model to build small and informative models that are robust to noise and small-scale traffic variations [21].

Although CNNs are effective when it comes to capturing spatial and statistical relationships, they are also restricted per se in capturing temporal evolution [22]. This limitation is overcome by a direct learning of the dependence between two successive traffic windows, which is executed by the addition of an LSTM layer. Posts attack within an industry can also take forms of a gradual deviation or a gradual behaviour rather than an abrupt deviation, and these patterns can rarely be observed in the event of static classifiers [23]. Using higher-level temporal patterns as opposed to plain features, which are operated on CNN-extracted embeddings, the LSTM is less sensitive to short-term variability [24, 25]. The empirical observations, such as the consistent error distribution among the sequence positions and the ability to handle the length of the sequence, validate the fact that the hybrid design allows consistent reasoning over time, as opposed to basing it on a single snapshot [26, 27].

Binary intrusion detection aligns with several industrial cybersecurity standards, including IEC 62443, where the primary operational requirement is the rapid discrimination between normal operation and security-relevant anomalies requiring intervention. In many industrial environments, coarse-grained attack identification is sufficient at the first decision layer, while detailed attack categorisation is deferred to forensic or supervisory systems. Binary IDS models are therefore

preferred in scenarios where low latency, high reliability, and minimal false alarms are prioritised over fine-grained attack taxonomy [28].

5.2. Operational Implications

The capability to manage the false alarm rate (FAR) is a decisive necessity to IIoT intrusion detection systems [29]. False alarms may cause chaos in the industrial process, unnecessary safety measures and decrease the trust of the operators in the automated systems. The threshold and FAR analysis indicate that the suggested model provides a trade-off between false alarms and detection sensitivity with a smooth and predictable trade-off curve. This enables the operators of the system to adjust the decision threshold based on the risk toleration and sensitivity of deployments, and yet service providers avoid sudden degradation in recall [30].

Besides FAR control, there are practical implications of the model confidence-aware behaviour. The distinct difference in prediction accuracy among benign and attack traffic, coupled with high-quality probability calibration, allows for the use of probabilistic output and not hard binary choices [31, 32]. These probabilities, which are calibrated, may be used to support either risk-based alert prioritisation, adaptive response mechanisms or human-in-the-loop verification in uncertain situations. These are properties that are hardly mentioned in the current literature on IIoT IDS but are necessary in the field to be used in safety-critical applications.

The split baseline comparison strengthens the deployment-oriented contribution of this study. While tree-based baselines such as Random Forest and XGBoost achieved strong classification performance, they did not provide the same balance of recall, FAR, calibration reliability, and temporal modelling. Similarly, CNN-only and LSTM-only ablations confirmed that neither convolutional feature extraction nor recurrent sequence modelling alone was sufficient to achieve the strongest deployment-level reliability. The CNN–GRU baseline showed that a lighter recurrent unit can provide competitive efficiency, but the LSTM-based model achieved stronger false-alarm control and calibration. Therefore, the main contribution of the proposed framework lies in combining compact feature abstraction with temporal modelling while preserving CPU-level deployability.

5.3. Comparison with Existing Literature

The proposed model has distinct benefits over traditional static IDS systems because it models both correlated feature structure and temporal attack dynamics. However, the comparison with existing literature clarifies that high classification accuracy has also been reported by other BoTNeTIoT-L01 studies. For example, Jose and Jose reported approximately 99.86% accuracy and 99.83% F1-score for AS-CL IDS on BoTNeTIoT-L01, while machine-learning studies using Random Forest or XGBoost have also reported very high binary-detection performance. Therefore, the contribution of the present work should not be interpreted as only a higher accuracy claim. Instead, its contribution lies in extending performance reporting toward deployment-critical properties that prior studies generally omit: explicit threshold-FAR behaviour, calibration metrics, confidence reliability, temporal robustness, missed-attack analysis, sequence-length sensitivity, and CPU-only inference cost.

This distinction is important because an industrial IDS with high headline accuracy may still be difficult to deploy if its confidence scores are poorly calibrated or if false alarms cannot be controlled [33, 34]. The present CNN–LSTM model is therefore positioned as a deployment-oriented IDS rather than merely another hybrid architecture. The literature comparison strengthens this interpretation by acknowledging that simpler models may be competitive on accuracy, while demonstrating that the proposed evaluation framework provides additional operational evidence needed for risk-aware IIoT deployment.

LIMITATIONS

This study has several limitations that should be acknowledged. First, the evaluation was conducted using a single IIoT benchmark dataset, BoTNeTIoT-L01, which may not fully represent the diversity of real industrial traffic, device types, communication protocols, and deployment conditions. Second, the study focused mainly on binary intrusion detection, where traffic was classified as benign or attack. Although this is useful for first-line operational defence, it does not provide detailed attack-category identification for forensic analysis. Third, while the proposed CNN–LSTM model showed strong performance, real-world IIoT networks may experience concept drift, changing attack behaviour, and unseen traffic patterns that were not fully tested. Fourth, the reported CPU-only inference results support lightweight deployment, but further validation on real edge gateways or industrial controllers is still required. Future work should evaluate the model across multiple IIoT datasets, multi-class attack settings, live traffic streams, and adaptive recalibration scenarios.

CONCLUSION

In this paper, a lightweight CNN–LSTM intrusion detection framework was proposed for Industrial IoT networks. The model jointly captures correlated statistical traffic descriptors and temporal attack dynamics using a compact architecture with 83,585 trainable parameters. After correcting the numerical inconsistency across the manuscript, the final held-out test results show accuracy = 0.99875, precision = 0.99930, recall = 0.99820, F1-score = 0.99875, and FAR = 0.00070.

The comparison with published BoTNeTIoT-L01 studies also clarifies the paper's contribution. Some prior studies report similarly high or higher headline accuracy, particularly AS-CL IDS and classical ML/XGBoost baselines. The present work is therefore positioned not as a purely accuracy-driven architecture paper, but as a deployment-oriented IDS evaluation that reports false-alarm controllability, confidence calibration, temporal robustness, missed-attack behaviour, and CPU-only inference efficiency. These properties are central to operational IIoT security, where risk-aware alerting and low false-alarm burden are as important as aggregate classification accuracy.

The next phase of work will be the extension of the framework to the multi-class classification of attacks so that a

more detailed threat characterisation would become possible. Handling concept drift using online or adaptive learning processes is also another significant direction, as the IIoT traffic patterns can change as time goes by, containing system updates or alterations to the behaviours of the system. Moreover, the suggested model was incorporated into a federated learning architecture to facilitate collaborative intrusion detection among distributed industrial locations without compromising on data privacy and communication overheads.

LIST OF ABBREVIATIONS

FAR	=	False Alarm Rate
IDS	=	Intrusion Detection Systems
IIoT	=	Internet of Things
LSTM	=	Long Short-Term Memory
RNNs	=	Recurrent Neural Networks

AUTHORS' CONTRIBUTIONS

M.A. conceived and designed the study, acquired and preprocessed the dataset, conducted the statistical analyses and machine learning experiments, interpreted the findings, and developed the proposed framework and I.U. contributed to the drafting and critical revision of the manuscript, provided input on the interpretation of results, reviewed the methodological aspects, and approved the final version for publication.

ETHICAL APPROVAL & INFORMED CONSENT

Not applicable.

AVAILABILITY OF DATA AND MATERIALS

The data will be made available on reasonable request by contacting the corresponding author.

FUNDING

None.

CONFLICT OF INTEREST

The authors declare that there is no conflict of interest regarding the publication of this article.

ACKNOWLEDGEMENTS

Declared none.

DECLARATION OF AI

During the preparation of this manuscript, the author utilized ChatGPT exclusively for language editing and refinement purposes. All AI-assisted modifications were thoroughly reviewed, validated, and approved by the author, who assumes full responsibility for the accuracy, integrity, and final content of the manuscript.

REFERENCES

[1] S. F. Ahmed, M. S. Alam, M. Hoque, A. Lameesa, S. Afrin, T. Farah, M. Kabir, G. M. Shafiullah, and S. M. Muyeen, "Industrial Internet of Things enabled technologies, challenges, and future directions," *Comput. Electr. Eng.*, vol. 110, Art. no. 108847, 2023, https://doi.org/10.1016/j.compeleceng.2023.108847

[2] A. Buja, Cybersecurity of Industrial Internet of Things (IIoT). *Routledge*, 2026, Available from: https://www.routledge.com/Cybersecurity-of-Industrial-Internet-of-Things-IIoT/Buja/p/book/9781032467832

[3] A. Mustafa, F. Trad, and A. Chehab, "Leveraging large language models for reducing false positives and prioritizing alerts in intrusion detection systems," in *Proc. Int. Conf. Adv. Inf. Netw. Appl., Cham*, Switzerland: Springer Nature Switzerland, 2025, pp. 432-443, https://doi.org/10.1007/978-3-031-87772-8_37

[4] A. Kaur, "Intrusion detection approach for industrial Internet of Things traffic using deep recurrent reinforcement learning assisted federated learning," *IEEE Trans. Artif. Intell.*, vol. 6, no. 1, pp. 37-50, 2025, https://doi.org/10.1109/TAI.2024.3443787

[5] M. Landauer, F. Skopik, B. Stojanović, A. Flatscher, and T. Ullrich, "A review of time-series analysis for cyber security analytics: From intrusion detection to attack prediction," I*nt. J. Inf. Secur.,* vol. 24, no. 1, Art. no. 3, 2025, https://doi.org/10.1007/s10207-024-00921-0

[6] A. Balla, M. H. Habaebi, M. R. Islam, and S. Mubarak, "Applications of deep learning algorithms for supervisory control and data acquisition intrusion detection system," *Cleaner Eng. Technol.*, vol. 9, Art. no. 100532, 2022, https://doi.org/10.1016/j.clet.2022.100532

[7] O. Polat, A. A. Ahmad, S. Oyucu, E. Algül, F. Doğan, and A. Aksöz, "Temporal-spatial feature extraction in IoT-based SCADA system security: Hybrid CNN-LSTM and attention-based architectures for malware classification and attack detection," *IEEE Access*, vol. 13, pp. 102109-102132, 2025, https://doi.org/10.1109/ACCESS.2025.3577761

[8] W. Oñate and R. Sanz, "Fog computing architecture for load balancing in parallel production with a distributed MES," *Appl. Sci.*, vol. 15, no. 13, Art. no. 7438, 2025, https://doi.org/10.3390/app15137438

[9] A. Q. Khan, N. Tamani, S. El Jaouhari, and L. Mroueh, "A contextual derivation algorithm for cybersecurity in IoT environments," in *Proc. IEEE 22nd Int. Conf. Trust, Secur. Privacy Comput. Commun. (TrustCom)*, pp. 1430-1435, 2023, https://doi.org/10.1109/TrustCom60117.2023.00195

[10] I. Ahmad, M. N. Amin, K. Hamid, S. M. Rizwan, and S. A. Naqvi, "Enhanced IoT network security for network intrusion detection," *Int. J. Comput. Eng. Technol.*, vol. 3, no. 8, 2025, https://doi.org/10.63075/0vcg0093

[11] L. Mohammadpour, T. C. Ling, C. S. Liew, and A. Aryanfar, "A survey of CNN-based network intrusion detection," *Appl. Sci*., vol. 12, no. 16, Art. no. 8162, 2022, https://doi.org/10.3390/app12168162

[12] K. Noor, A. L. Imoize, C. T. Li, and C. Y. Weng, "A review of machine learning and transfer learning strategies for intrusion detection systems in 5G and beyond," *Mathematics*, vol. 13, no. 7, Art. no. 1088, 2025, https://doi.org/10.3390/math13071088

[13] A. S. Gaafar, J. M. Dahr, and A. K. Hamoud, "Comparative analysis of performance of deep learning classification approach based on LSTM-RNN for textual and image datasets," *Informatica*, vol. 46, no. 5, 2022, https://doi.org/10.31449/inf.v46i5.3872

[14] Y. Tang, Y. Wang, C. Liu, X. Yuan, K. Wang, and C. Yang, "Semi-supervised LSTM with historical feature fusion attention for temporal sequence dynamic modeling in industrial processes," *Eng. Appl. Artif. Intell*., vol. 117, no. A, Art. no. 105547, 2023, https://doi.org/10.1016/j.engappai.2022.105547

[15] A. Nazir, J. He, N. Zhu, S. S. Qureshi, S. U. Qureshi, F. Ullah, A. Wajahat, and M. S. Pathan, "A deep learning-based novel hybrid CNN-LSTM architecture for efficient detection of threats in the IoT ecosystem," *Ain Shams Eng. J*., vol. 15, no. 7, Art. no. 102777, 2024, https://doi.org/10.1016/j.asej.2024.102777

[16] M. Sajid, K. R. Malik, A. Almogren, T. S. Malik, A. H. Khan, J. Tanveer, and A. U. Rehman, "Enhancing intrusion detection: A hybrid machine and deep learning approach," J. *Cloud Comput.*, vol. 13, no. 1, Art. no. 123, 2024, https://doi.org/10.1186/s13677-024-00685-x

[17] D. M. Afraji, J. Lloret, and L. Peñalver, "An integrated hybrid deep learning framework for intrusion detection in IoT and IIoT networks using CNN-LSTM-GRU architecture," *Computation*, vol. 13, no. 9, Art. no. 222, 2025, https://doi.org/10.3390/computation13090222

[18] C. K. Wikle and A. Zammit-Mangion, "Statistical deep learning for spatial and spatiotemporal data," *Annu. Rev. Stat. Its Appl*., vol. 10, no. 1, pp. 247-270, 2023, https://doi.org/10.1146/annurev-statistics-033021-112628

[19] X. Liu, J. Shan, C. Liu, S. Zhang, D. Zhang, Z. Hao, and S. Huang, "An operating condition diagnosis method for electric submersible screw pumps based on CNN-ResNet-RF," *Processes,* vol. 13, no. 7, Art. no. 2043, 2025, https://doi.org/10.3390/pr13072043

[20] K. Bansal and A. Singhrova, "Review on intrusion detection system for IoT/IIoT: Brief study," *Multimed. Tools Appl*., vol. 83, no. 8, pp. 23083-23108, 2024, https://doi.org/10.1007/s11042-023-16395-6

[21] A. Nascita, G. Aceto, D. Ciuonzo, A. Montieri, V. Persico, and A. Pescapé, "A survey on explainable artificial intelligence for Internet traffic classification and prediction, and intrusion detection," *IEEE Commun. Surv. Tutor*., vol. 27, no. 5, pp. 3165-3198, 2025, https://doi.org/10.1109/COMST.2024.350495

[22] S. Rekik and S. Mehmood, "Hybrid GNN-LSTM defense with differential privacy and secure multi-party computation for edge-optimized neuromorphic autonomous systems," *Sci. Rep., vol*. 15, no. 1, Art. no. 43939, 2025, https://doi.org/10.1038/s41598-025-27691-6

[23] U. K. Lilhore, P. Manoharan, S. Simaiya, R. Alroobaea, M. Alsafyani, A. M. Baqasah, S. Dalal, A. Sharma, and K. Raahemifar, "HIDM: Hybrid intrusion detection model for Industry 4.0 networks using an optimized CNN-LSTM with transfer learning," *Sensors*, vol. 23, no. 18, Art. no. 7856, 2023, https://doi.org/10.3390/s23187856

[24] H. C. Altunay and Z. Albayrak, "A hybrid CNN+LSTM-based intrusion detection system for industrial IoT networks," *Eng. Sci. Technol. Int. J*., vol. 38, Art. no. 101322, 2023, https://doi.org/10.1016/j.jestch.2022.101322

[25] H. Gupta, A. Jadhav, and A. S. Bisht, "Comparative analysis of machine and deep learning models for intrusion detection in fog-enabled IoT networks," *Int. J. Netw. Distrib. Comput.*, vol. 14, no. 1, Art. no. 1, 2026, https://doi.org/10.1007/s44227-025-00079-8

[26] N. Verma, N. Kumar, K. K. Almuzaini, A. Sinha, S. A. Hussain, "A real-time intelligent intrusion detection framework for robotic system cybersecurity," *Peer-to-Peer Netw. Appl*., vol. 19, no. 1, Art. no. 30, 2026, https://doi.org/10.1007/s12083-025-02175-6

[27] T. B. Ogunseyi, G. Thiyagarajan, H. He, V. Bist, and Z. Du, "Performance analysis of explainable deep learning-based intrusion detection systems for IoT networks: A systematic review," *Sensors*, vol. 26, no. 2, Art. no. 363, 2026, https://doi.org/10.3390/s26020363

[28] S. Kalyani and D. Vydeki, "A resource-efficient ensemble machine learning framework for detecting rank attacks in RPL-based IoT networks," *J. Economy Technol.*, vol. 4, pp. 171-185, 2026, https://doi.org/10.1016/j.ject.2025.06.003

[29] M. Zahid and T. S. Bharati, "Leveraging machine learning and deep learning in IoT security: A review," *Secur. Privacy,* vol. 9, no. 1, Art. no. e70144, 2026, https://doi.org/10.1002/spy2.70144

[30] Q. Alasad, M. Ahmed, S. Alahmed, O. T. Khattab, S. A. Abdulwahhab, and J. S. Yuan, "A comprehensive review: The evolving cat-and-mouse game in network intrusion detection systems leveraging machine learning," *J. Cybersecur. Privacy*, vol. 6, no. 1, Art. no. 13, 2026, https://doi.org/10.3390/jcp6010013

[31] A. Alhowaide, "IoT Dataset for Intrusion Detection Systems (IDS)," Kaggle, Dataset, 2023. Available from: https://www.kaggle.com/datasets/azalhowaide/iot-dataset-for-intrusion-detection-systems-ids. (Accessed on: 9 July 2026).

[32] J. Jose and D. V. Jose, "AS-CL IDS: Anomaly and signature-based CNN-LSTM intrusion detection system for Internet of Things," *Int. J. Adv. Technol. Eng. Explor.*, vol. 10, no. 109, pp. 1622-1639, 2023, http://doi.org/10.19101/IJATEE.2022.10100187

[33] Z. Benamor, Z. A. Seghir, M. Djezzar, and M. Hemam, "A comparative study of machine learning algorithms for intrusion detection in IoT networks," *Rev. d'Intell. Artif.*, vol. 37, no. 3, pp. 567-576, 2023, https://doi.org/10.18280/ria.370305

[34] M. Z. Mahmud, S. Islam, S. R. Alve, and A. Jubayer Pial, "Optimized IoT Intrusion Detection using Machine Learning Technique," in *Proc. IEEE 3rd Int. Conf. Robot., Autom., Artif.-Intell. Internet-of-Things (RAAICON)*, Dhaka, Bangladesh, 2024, pp. 167–172, https://doi.org/10.1109/RAAICON64172.2024.10928532