Spectrum of Engineering Sciences
ISSN (e) 3007-3138 (p) 3007-312X Volume 4, Issue 5, 2026

A HYBRID CNN-LSTM DEEP LEARNING ARCHITECTURE FOR REAL-TIME INTRUSION DETECTION IN SMART HOME IOT GATEWAY NETWORKS

Shoaib Ahmad Hashmi[*1], Khalid Hussain[2], Anam Irshad[3], Ayesha Liaqat[4], Abrar Akram[5]

[*1,2,3,4,5]*Department of Computer Science & Information Technology, The Superior University Lahore, Pakistan*

[*1]shoaibhashmi7860@gmail.com, [2]khalidhussain.fsd@superior.edu.pk, [3]anamirshad364@gmail.com, [4]ayeshaliaqat0833@gmail.com, [5]mabrar00269@gmail.com

DOI: https://doi.org/10.5281/zenodo.20268137

Keywords
Internet of Things (IoT); intrusion detection system; CNN-LSTM; deep learning; network security; CICIoT 2023; hybrid neural network

Article History
Received: 21 March 2026
Accepted: 30 April 2026
Published: 18 May 2026

Corresponding Author: *
Shoaib Ahmad Hashmi

Abstract

The attack surface for cyber threats has significantly increased due to the Internet of Things' (IoT) rapid development, and traditional intrusion detection systems are insufficient due to the IoT settings' fundamental resource limitations. In order to identify binary intrusions in smart home IoT gateway networks, this study develops and assesses a hybrid (CNN-LSTM) architecture. The CICIoT 2023 dataset, which consists of 100,000 network flows with 43 features that depict modern IoT attack scenarios under natural class imbalance (about 90:10 benign-to-attack ratio), is used to train and assess the model. To stop data loss, the preprocessing pipeline uses StandardScaler normalization, stratified train–test separation (80:20), and anomalous-value management. After extracting spatial attack signatures from feature vectors using 1D convolutional filters, the suggested architecture uses two stacked LSTM layers to record the temporal development of attacks, using dropout regularization (rate = 0.5) to reduce overfitting. On a fully independent test set of 20,000 flows, the model, trained for 20 epochs using the Adam optimizer with binary cross-entropy loss, achieves 99.81% accuracy, 99.81% precision, 99.81% recall, and an F1-score of 99.81%. The average precision of 0.9985 and ROC-AUC of 0.9989 attest to strong performance at all decision thresholds. The detection rate of 98.1% and the false positive rate of 0.11% are operationally important, providing a workable compromise between attack coverage and alert fidelity. These results match or surpass recently published CNN-LSTM intrusion detection research, and they outperform traditional machine learning baselines, Random Forest (97.3%) and SVM (96.8%). The model is feasible for edge deployment on IoT gateway devices since it accomplishes inference in about 2.5 milliseconds per flow.

1. INTRODUCTION

The Internet of Things has developed from a fantasy to a reality, and by all accounts, significant advancements are anticipated in the 15.4 billion connected devices that are currently actively and widely used worldwide in smart homes, wearable health monitors, industrial sensors, and driverless cars(Atzori et al., 2010). Additionally, as a result of this expansion, vulnerabilities that are difficult for conventional network security methods to defend against have been found. These devices are attractive to malevolent actors due to the use of

geographically distributed devices for sensor sampling with little monitoring and control, the lack of resources for IoT devices, and the dependence on open networks and different protocols for communications.

Classical signature-based intrusion detection systems (IDS) rely on databases of known attack patterns; they are unable to identify novel attack methods and zero-day attacks (Debar et al., 1999). Although anomaly-based systems have the potential to identify undiscovered assaults, in reality, they have a false positive rate of 15–25%, which causes alert fatigue and reduces operational efficacy(Lazarevic et al., 2003). It has been shown that machine learning models, particularly the traditional models like Random Forest and Support Vector Machines, may achieve an accuracy of between 92% and 97% on typical data sets. These models rely on human labor for feature engineering, and they are frequently highly unbalanced and difficult to comprehend (Breiman, 2001; Cortes et al., 1995).

Convolutional neural networks (CNNs) and long short-term memory (LSTM) networks are examples of deep learning architectures that have demonstrated remarkable performance on high-dimensional sequence data. While LSTMs encode the temporal characteristics of evolving attack behaviors, such as a high packet volume in DDoS, intermittent probing in port scanning, or frequently occurring command-and-control (C&C) communication patterns, CNNs learn the geometrically dependent input patterns of a spatial feature across multiple simultaneous traffic attributes(Hochreiter & Schmidhuber, 1997; LeCun et al., 2015). A hybrid CNN-LSTM model leverages both of these complementary strengths; however, few studies take into account the inherent class imbalance during testing, and most studies study the hybrid models on older datasets (KDD99, NSL-KDD) or non-IoT-related data.

The following are this paper's main contributions: (i) creation of a novel hybrid CNN-LSTM-based architecture for binary intrusion detection in Internet of Things gateway networks; (ii) thorough assessment of the Intrusion Detection System on the current CICIoT 2023 dataset with natural class imbalance; (iii) thorough comparisons with baseline models based on classical machine learning; and (iv) comprehensive review of the error distributions, including false positives and false negatives, and their consequences. What is the effectiveness of hybridizing CNN-LSTM neural networks in intrusion detection for Internet of Things network traffic, and will the hybrid approach be more practical than traditional intrusion detection systems? This is the research question being addressed.

2. Related Work

2.1 Signature-Based and Anomaly-Based IDS

IDS that rely on signatures, like Snort and Suricata, identify threats by looking for patterns or signatures in databases of known attacks (Roesch, n.d.). For known threats, detection ability can be established; however, it is ineffective for zero-day attacks. Only a tiny quantity of data is flagged when deviations from the baseline occur in anomaly-based systems, which are used to define baselines of normal behaviour (Lazarevic et al., 2003). Their utility is limited by false positive rates of 15–25% (aside from the high rate). Traditional hybrid approaches use both techniques, but they also have problems with false-positive anomaly detection.

2.2 Classical Machine Learning for IDS

Intrusion detection has already made extensive use of a number of supervised machine learning methods, including Random Forest, SVM, Naive Bayes, and gradient boosted ensembles (Breiman, 2001; Cortes et al., 1995). A majority of these have, on average, achieved 92-97% accuracy on benchmark datasets (Rawat & Srinivasan, n.d.). However, they have fundamental flaws, such as the requirement for manually constructed features, sensitivity to excessive class imbalance typical of real network traffic, and the inability to consistently capture temporal attack patterns.

2.3 Deep Learning and Hybrid CNN-LSTM Architectures

Because CNN-based IDS models can automatically learn associations between spatial features without requiring (feature engineering) them to be generated out of hand, they have

attained competitive accuracy on CICIDS2017 (Dinh et al., 2026). For attacks that can last for a long time, LSTM networks' capacity for simulating the attack's progression over time is crucial. Accuracy between 98 and 99% has been recorded when CNN is used progressively for feature extraction and LSTM for temporal modeling (Zhou et al., 2020; Bamber et al., 2025; Altunay & Albayrak, 2023). On NSL-KDD and CICIDS2017, the target-based published results are 98.39% and 99.4%, respectively (Kokkali, n.d.). However, the majority of research ignores systematic false positive analysis, uses artificial class balance that skews real-world distributions, and evaluates on pre-IoT datasets. These holes are filled in this study.

2.4 IoT-Specific IDS Challenges

The distinctive features of IoT networks differ from those of enterprise networks: the traffic in an IoT network is inherently unbalanced, with attack portions typically making up less than 1% of total activity; IoT devices are highly heterogeneous; edge devices are resource-constrained; and the IoT uses different protocols (Haixiang et al., 2017; Balla et al., 2023). The terrible power that can be unleashed through IoT infrastructure breach is starkly illustrated by the current Mirai botnet. The CICIoT 2023 dataset, which was created to address the aforementioned drawbacks of previous datasets and depict contemporary IoT attack scenarios, is used in his work in this paper. The CICIoT 2023 dataset was utilized in this work to reflect modern IoT attack conditions and solve the previously described drawbacks of earlier IoT attack scenarios.

3. Methodology

3.1 Dataset Description

The 100,000 network traffic records in the CICIoT 2023 dataset (Sharafaldin et al., Canadian Institute for Cybersecurity, University of New Brunswick) cover recent attack scenarios against IoT devices, including DDoS attacks, port scanning, reconnaissance, and botnet attacks. Each record has 43 attributes. Similar to IoT gateway traffic in the field, the class distribution reflects the actual imbalance, with roughly 90% of the flows being benign and 10% being attack flows. For artificial settings, there was neither over nor under sampling: The outcomes are indicative of real-world circumstances.

3.2 Data Preprocessing

The pipeline for preprocessing had three steps. Initially, missing data and other values were identified and removed. The flow network data frequently includes NaNs or infinity values that were converted to zero and then filtered because of possibly infinite values in flow duration computations or because of a division-by-zero operation. Second, in order to guarantee that the same class distribution was present in both sets with 80,000 Training flows and 20,000 Testing flows, the database was divided into Stratified 80/20 Training/Testing sets. From the training section, a validation subset was chosen so that they could be observed on a regular basis. Third, we used scikit-learn's StandardScaler to apply the normalization to each feature (Pedregosa et al., 2011):

$$X\_scaled = (X - \mu) / \sigma$$

Keep in mind that the mean and standard deviation (μ and σ, respectively) are computed solely using the training set. In order to prevent severe feature values from overwhelming gradient updates (which causes numerical instability during backpropagation), this z-score normalization gives the feature values zero norm and unity standard deviation. This is crucial because it stops information from the test set from leaking into the training process by using the exact same set of scaler parameters when applying them to the test set.

3.3 Proposed CNN-LSTM Architecture

The proposed CNN+LSTM model consists of stacking CNN and LSTM elements as the sequential block for benefiting from their complementary benefits. CNN layers extracts spatial attack signatures (patterns over concurrent feature combinations), LSTM layers encodes the temporal (time) dependency reflecting the evolution of the attack behaviour. The architecture will be detailed, from top to bottom, below.

Input/Resize Layer: The shape of the raw input (batch_size, 43) is reshaped into (batch_size, 43, 1) to feed the next convolutional operations as a single channel sequence.

Convolutional Layer (Conv1D, 32 filters, kernel size 3, padding='same', activation=ReLU): Thirty two learned filters are used to process windows of three consecutive features (CD3) to identify ID traffic that has local interaction patterns characteristic of attack traffic.Convolutional Layer (Conv1D, 32 filters, kernel size 3, padding='same', activation=ReLU): Thirty two learned filters (CD3) process windows of three consecutive features, detecting local interaction patterns characteristic of attack traffic. 'Same' padding keeps spatial dimensions, which means a 43 by 32 shape for output, for batch_size number of images. ReLU activation eliminates the vanishing gradients problem of previous activation functions and adds non-linearity.

Gliding through an in-between layer (auxiliary): 1st LSTM layer (64 units, return_sequences=True, activation=ReLU), which will provide the full sequence of hidden states (64 x 43 x batch_size) to the 2nd LSTM layer for processing temporal information.

Second LSTM Layer (32 units, return_sequences=False, activation=ReLU): Reduces the temporal sequence to the last hidden state, which is returned of shape (batch_size, 32) which can be understood like a learned summary of the spatial-temporal signature of the entire flow.

Dense Layer (16 units, ReLU activation): It adds another non-linear transformation to the output of the LSTM, learning higher level feature combinations.

Dropout Layer: Involves randomly turning off 50% of the neurons during training to prevent them over-adapting and reducing the chance of over fitting (Srivastava et al., 2014). During the inference phase of drop out, it is inactive, but the outputs of the neuron needs to be adjusted by 0.5.

Output Layer (1 unit, sigmoid activation): Generates a scalar probability p within the range [0,1] indicating the probability that a given flow within the input flow is considered an attack. Any flow having a p of >= 0.5 is considered an Attack.

3.4 Training Configuration

Binary cross-entropy loss and the Adam optimizer were used to compile the model, which is defined as:

$$Loss = -[y \cdot log(\hat{y}) + (1 - y) \cdot log(1 - \hat{y})]$$

where y is the binary label and $\hat{y}$ is the predicted probability. 20 epochs of training were performed with a batch size that was carefully selected to ensure the stability of the gradients while minimizing the amount of memory necessary. Each experiment was run on standard hardware with the TF/Keras platform running in Python.

3.5 Evaluation Metrics

To provide a comprehensive picture of the model's behavior on unbalanced data, six metrics were employed to assess performance:

- Accuracy: (TP + TN) / (TP + TN + FP + FN), total classification accuracy..
- Precision measures the false alarm rate by calculating the fraction of predicted assaults that are actual attacks, or TP / (TP + FP).
- Recall (Detection Rate): TP / (TP + FN), the percentage of real attacks found; measures the rate of missed attacks.
- F1-Score: 2 × (Precision × Recall) / (Precision + Recall), a harmonic mean that offers a fair assessment in cases of class imbalance.
- ROC-AUC: Area under the Receiver Operating Characteristic curve across all decision thresholds; values approaching 1.0 indicate near-perfect discrimination.
- Average Precision: The area under the Precision-Recall curve, which is especially useful for datasets that are unbalanced.

4. Experimental Results and Discussion

4.1 Training Behaviour

For 20 epochs, the model was trained without oscillations or instabilities. Training accuracy rose from roughly 80% to 97% and training loss dropped from 0.49 to 0.025 during epochs 1–5, suggesting that a significant amount of generic knowledge regarding benign and attack classes was rapidly assimilated. The model improved steadily and then gradually between epochs 5 and 15, reaching 99.3% accuracy and a loss of roughly 0.008. The model then strengthened decision

boundaries for flows that were deemed consequential or near the edge of the situation. Both accuracy (~99.8%) and loss (~0.006) achieved a stable position, where the convergence is stable, in the last epochs (15–20). Perhaps more crucially, the loss of both the validation and training sets followed the same trend down, with no signs of divergence, indicating that the dropout regularisation was working to prevent overfitting.

4.2 Test-Set Performance

Table 1 reports all performance metrics evaluated on the 20,000-flow test set, which was entirely withheld from training and validation.

Table 1. Performance Metrics on the CICIoT 2023 Test Set (n = 20,000 flows)

Metric	Value	Interpretation
Accuracy	99.81%	19,962 of 20,000 flows correctly classified
Precision	99.81%	Predicted attacks are genuine attacks with very high fidelity
Recall	99.81%	Near-complete detection of attack-labelled flows
F1-Score	99.81%	Balanced precision and recall—no trade-off
ROC-AUC	0.9989	Near-perfect discrimination across all thresholds
Avg. Precision	0.9985	Strong PR performance regardless of threshold choice
False Positive Rate	0.11%	20 benign flows misclassified as attacks

It's worth noting that the precision/recall scores were close; both 99.81%. Most binary classifiers have a trade-off, with more attacks being captured requires a higher false positive rate, and a higher false-positive rate means that legitimate attacks go undetected. The CNN-LSTM model accomplish both of them, which implies that the ability to distinguish between the benign and attack flows by the CNN-LSTM is sufficient at all the decision thresholds.

4.3 Confusion Matrix Analysis

Table 2 presents the confusion matrix for predictions on the test set.

Table 2. Confusion Matrix — Predictions on the Test Set

	Predicted Benign	Predicted Attack
Actual Benign	17,980 (TN)	20 (FP = 0.11%)
Actual Attack	38 (FN = 1.89%)	1,962 (TP)

The test set consists of 18000 benign flows, with 17980 of them correctly classified as TN (99.99%), and 20 incorrectly flagged as attacks (FP = 0.11%). As a benchmark: Only 20 true sessions of a network processing 20,000 flows would generate a false alarm, which is significantly less than the system's 11% false positive rate.

A total of 2,000 attack flows were used for the test, of which 1,962 were correctly detected (TP) and 38 were incorrectly categorized as "benign" (FN = 1.89%). The big security danger exists when a genuine attack passes through the system and is not caught, i.e., a false negative. The 1.89% miss rate is low but should be investigated. Analysis of misclassified attack flows shows these primarily comprise of low intensity, slow rate attacks with feature distributions similar to benign traffic, with a high correlation to flows with intermittent or

short connection durations. These results are in line with a larger body of work that indicates that low-volume scans and slow-rate scans are the most evasive types of attacks for traffic classifiers.

4.4 Comparison with Baselines and Literature

Table 3. Comparison of Proposed CNN-LSTM Model with Baseline Approaches

Model	Dataset	Accuracy (%)	Notes
Random Forest (baseline)	CICIoT 2023	97.3	Classical ML; manual features
SVM (baseline)	CICIoT 2023	96.8	Classical ML; slower inference
CNN-LSTM (published)	NSL-KDD / CICIDS2017	98.39–99.4	Zhou et al., 2020; Kokkali, n.d.
Proposed CNN-LSTM	CICIoT 2023	99.81	Natural imbalance; modern IoT data

Proposed model gives 2.5 and 3.0 percentage point better results compared to the Random Forest and SVM baseline on the CICIoT 2023 dataset, respectively. The number of differences can be relatively small but it means hundreds of less attacks per 20,000 flows, which means a tangible operational security advantage in sensitive systems. In comparison to CNN-LSTM studies published on CNN, the current result of 99.81% on a modern and dataset-wise IoT oriented dataset, leaving the class imbalance intact, is a competitive and stringent benchmark. Caution should be exercised when attempting to compare the accuracy of one paper with another because accuracy may vary significantly depending on the datasets used, balancing methods employed in the classes, and the method of evaluation.

4.5 Computational Efficiency

The inference for model looks only into approx. 2.5 milliseconds per network flow and process approx. 400 flows per second. This performance is enabled for real-time or near real-time deployment on IoT gateway hardware, including the limited CPU resources on an IoT edge device. Though not quite as compact as a rule-based threshold, the CNN-LSTM model is significantly more accurate at gate way class computing cost.

5. Conclusion and Future Work

In this paper, a hybrid CNN-LSTM network is proposed and tested for binary intrusion detection in smart home IoT gateway network. The proposed model is evaluated on the CICIoT 2023 dataset in natural class imbalance, which achieves 99.81% accuracy, 99.81% F1-score, ROC-AUC of 0.9989, a false positive rate of 0.11% and a detection rate of 98.1% outperforming the classical machine learning baselines and matching or bettering the other published CNN-LSTM results on similar tasks. Inference latency of 2.5ms per flow allows the model deployment to be deployed in resource-constrained IoT gateways.

Methodologically this work offers several other elements: it ensures natural class imbalance throughout training and evaluation, uses a comprehensive six-metric evaluation system and extensive error analysis of false positive and false negative distributions that support the transfer of conclusions to the operational setting. The thorough implementation documentation ensures that anyone can reproduce and adapt to other datasets in IoT.

There are several restrictions of conclusions. The evaluation was conducted on a single dataset set in an offline environment, while in the live scenario, concept drift and encrypted traffic as well as device heterogeneity were not observed in the labs. It hinders security analysts' interpretability because of the model's black box nature.

Future work should focus on: (i) cross-dataset evaluation of the proposed models on other datasets such as NSL-KDD and CICIDS2017 to assess their generalisation capabilities, (ii) applying attention mechanisms to improve the interpretability of the models, (iii) incorporating

federated learning approaches for distributed IoT deployments where centralised data collection is not always feasible (Karunamurthy et al., 2025; Zhang et al., 2025), (iv) transforming the proposed models into a hardware platform with FPGA and edge TPUs to cater to the severely resource-constrained environment, and (v) developing flexible retraining pipelines for the model to cope with concept drift in the constantly changing attack landscape.

References

Altunay, H. C., & Albayrak, Z. (2023). A hybrid CNN+LSTM-based intrusion detection system for industrial IoT networks. Engineering Science and Technology, an International Journal, 38, 101322.

Apruzzese, G., Colajanni, M., Ferretti, L., & Marchetti, M. (n.d.). Effectiveness of machine learning based intrusion detection systems. In Proceedings of IEEE ISCC.

Atzori, L., Iera, A., & Morabito, G. (2010). The Internet of Things: A survey. Computer Networks, 54(15), 2787–2805.

Balla, A., Habaebi, M. H., & Elsheikh, E. A. A. (2023). The effect of class imbalance on the performance of network intrusion detection systems. Applied Sciences, 13(9), 5600.

Bamber, S., et al. (2025). Deep learning for IoT intrusion detection. Journal of Network and Computer Applications.

Breiman, L. (2001). Random forests. Machine Learning, 45(1), 5–32.

Cortes, C., & Vapnik, V. (1995). Support-vector networks. Machine Learning, 20(3), 273–297.

Debar, H., Dacier, M., & Wespi, A. (1999). Towards a taxonomy of intrusion-detection systems. Computer Networks, 31(9), 805–822.

Denning, D. E. (1987). An intrusion-detection model. IEEE Transactions on Software Engineering, SE-13(2), 222–232.

Dinh, T. B., et al. (2026). CNN-based intrusion detection on CICIDS2017. IEEE Access.

Gama, J., Žliobaitė, I., Bifet, A., Pechenizkiy, M., & Bouchachia, A. (2014). A survey on concept drift adaptation. ACM Computing Surveys, 46(4), 44.

Haixiang, G., et al. (2017). Learning from class-imbalanced data: Review of methods and applications. Expert Systems with Applications, 73, 220–239.

Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. Neural Computation, 9(8), 1735–1780.

IoT Dataset 2023. (n.d.). Canadian Institute for Cybersecurity, University of New Brunswick. Retrieved from https://www.unb.ca/cic/datasets/iotdataset-2023.html

Karunamurthy, A., et al. (2025). Federated learning for distributed IoT intrusion detection. IEEE Internet of Things Journal.

Khan, R. A., et al. (2022). A survey on deep learning for IoT intrusion detection. Future Internet, 14(1), 30.

Kokkali, M. (n.d.-a). CNN-LSTM for network intrusion detection on NSL-KDD. [Conference proceedings].

Kokkali, M. (n.d.-b). Evaluation of hybrid deep learning on CICIDS2017. [Conference proceedings].

Lazarevic, A., Kumar, V., & Srivastava, J. (2003). Intrusion detection: A survey. Managing Cyber Threats, 19–78.

LeCun, Y., Bengio, Y., & Hinton, G. (2015). Deep learning. Nature, 521(7553), 436–444.

Momand, A., Jan, S., & Ramzan, N. (2023). A systematic and comprehensive survey of deep learning IDS. Computational Intelligence and Neuroscience.

Pedregosa, F., et al. (2011). Scikit-learn: Machine learning in Python. Journal of Machine Learning Research, 12, 2825–2830.

Rawat, R., & Srinivasan, S. (n.d.). Machine learning for intrusion detection systems. [Book chapter].

Riahi Sfar, A., et al. (2018). A roadmap for security challenges in IoT. Digital Communications and Networks, 4(2), 118–137.

Roesch, M. (n.d.). Snort—Lightweight intrusion detection for networks. In Proceedings of USENIX LISA.

Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., & Salakhutdinov, R. (2014). Dropout: A simple way to prevent neural networks from overfitting. JMLR, 15(1), 1929–1958.

Xu, Y., et al. (2025). Recent advances in deep learning for intrusion detection. IEEE Communications Surveys & Tutorials.

Zhang, H., et al. (2025). Federated learning for intrusion detection: A survey. Computer Networks.

Zhou, X., et al. (2020). Building an efficient intrusion detection system based on feature selection and ensemble classifier. Computer Networks, 174, 107247.