Spectrum of Engineering Sciences
ISSN (e) 3007-3138 (p) 3007-312X Volume 3, Issue 10, 2025

ENHANCING IOT/IIOT INTRUSION DETECTION: A COMPARATIVE STUDY OF HYBRID CNN-LSTM AND ADVANCED DNN ML MODEL ON EDGE-IIOTSET

Izzaldin Izhar[1], Ali Abdullah[2], Muhammad Zunnurain Hussain[*3], Muhammad Zulkifl Hasan[4]

[1]*Department of Data Sciences, University of Management & Technology, Lahore, Punjab, Pakistan*
[2]*Department of Computer Science, National University of Computer and Emerging Sciences (FAST-NUCES), Lahore, 54000, Pakistan*
[*3]*Department of Computer Science, Bahria University Lahore Campus, Pakistan*
[4]*Faculty of Information Technology, Department of Computer Science, University of Central Punjab, Lahore Pakistan*

[1]izzaldinizhar@gmail.com,[2] aliabdullahch2002@gmail.com,
[*3]zunnurain.bulc@bahria.edu.pk,[4]zulkifl.hasan@ucp.edu.pk

DOI: https://doi.org/10.5281/zenodo.17491327

Keyword
IIoT Intrusion Detection, Hybrid CNN-LSTM, Edge-IIoTset Dataset, Model Optimization ,Deep Learning (DL) ,IoT Security

Article History
Received: 12 September 2025
Accepted: 18 October 2025
Published: 31 October 2025

Corresponding Author: * Muhammad Zunnurain Hussain

Abstract

In this paper, we present an improved hybrid CNN-LSTM model to improve intrusion detection in IoT/IIoT environments. It is evaluated on the publicly available Edge-IIoTset dataset (hosted on Kaggle). Due to storage constraints, we use a dense but representative subset of the dataset, preserving its diversity in terms of attack scenarios and device interactions. The original dataset covers a lot of things about IoT/IIoT devices (e.g., temperature/humidity sensors, ultrasonic sensors, flame detectors, and heart rate monitors) and includes 14 different attack types, including DDoS/Distributed Denial of Service (DDoS), Man in the Middle attack, MID, MID, attack, and spy attacks.

Our hybrid CNN-LSTM model uses the features of convolutional neural networks (CNN) for temporal pattern recognition and long term short term memory (LSTM) networks to simulate the network as a temporal encoder, as deep neural networks (DNN) are used in previous work. Through extensive experiments, we show that the model exhibits good accuracy, good classification accuracy, and very low overhead in the classification of the model. We also present a comparative study of the design tasks, demonstrating the model's ability to establish the edge in low-end IoT environments.

The results show that the CNN-LSTM hybrid architecture is more robust against multiple attack vectors, thus providing predictive and robust solutions to real-time threats. This study helps to the development and study of intelligent IoT/IIoT-based computer security systems with applications in healthcare, medicine, and critical infrastructure. Data sharing and interoperability are key factors to improve research and studies in the field of IoT cybersecurity.

INTRODUCTION

1.1 Background of IoT/IIoT Security

The rise of the Internet and the Industrial Internet of Things (IIoT) has led to increased connectivity and automation across many industries, but this growth has come with many challenges. Control and signal detection are critical to the success of the Internet of Things/IIoT world and require an intrusion detection system (IDS).

1.2 Importance of Intrusion Detection Systems (IDS) in IoT/IIoT

IDS plays an important role in protecting IoT/IIoT environments from threats (e.g., DDoS, malware, and single-day attacks). Unlike traditional AI approaches, IoT/IIoT platforms face unique challenges, such as:

- Device limits (limited compute power).
- Different data and different rules.
- It takes knowledge to create for success.

Machine learning (ML) and deep learning (DL) are promising techniques, but their success depends on the environment and data.

1.3 Motivation for Hybrid Deep Learning Approaches

Prior work has explored DL methods (i.e., DNN, CNN, or LSTM) in hybrid networks such as CNN or LSTM.

CNN: Extract local features from data (e.g. network traffic patterns).

LSTM: Learn long-term dependency modeling in virus attack analysis.

This arrangement is ideal for IIoT devices, as it allows the controller to display time and frequency.

1.4 Research Objectives and Contributions

This paper introduces the following IoT/IIoT IDSs:

Proposing a CNN-LSTM hybrid model .

Advanced DNN analysis using edge-IIoTset dataset.

It shows higher accuracy (%X) and lower error (%Y) than traditional methods.

This is an excellent statistical analysis of real-time processing systems.

1.5 Paper Organization

➢ The paper is structured in the following:

- Section 2 reviews related work in ML/DL-based IDS.
- Section 3 is about methodology, including data preprocessing and model design.
- Section 4 shows experimental results and comparisons of both models.
- Section 5 discusses implications and limitations.
- Section 6 concludes with future directions.

Dataset	Year	Description	Features	Testbed	IoT/IIoT Devices
DNN-EdgeIIoT	2021	Cybersecurity dataset using a DNN-based testbed for IoT/IIoT attack detection.	60	7 Layers	+10 Types
ML-EdgeIIoT	2021	Cybersecurity dataset using traditional ML approaches for IoT/IIoT attack detection.	60	7 Layers	+10 Types

Table 1:

This table compares two cybersecurity datasets, DNN-EdgeIIoT and ML-EdgeIIoT, which were selected and merged to create a unified dataset for applying machine learning (ML) models (DNN and CNN-LSTM) to IoT/IIoT attack detection.

RELATED WORK

2.1 Traditional Machine Learning (ML) Approaches for IDS

Machine learning techniques such as intrusion detection systems (IDS) are widely used to detect and combat cyber threats. These techniques use statistical methods and algorithmic technique to assess cybersecurity and detect anomalies or threats. Some of the machine learning techniques used in IDS are Decision Tree (DT), Random Forest (RF), Support Vector Machine (SVM), K-Nearest Neighbor (KNN), Naïve Bayes (NB), Logistic Regression (LR), Artificial Neural Networks (ANN).

Machine learning algorithms are evaluated to improve machine performance in terms of accuracy, sensitivity, and specificity. In addition to providing a solid foundation for IDS, new techniques such as deep learning and federated learning are being explored to overcome limitations such as increased risk and responsiveness.

2.2 Deep Learning (DL) Models in IoT/IIoT Security

Deep learning (DL) techniques, like deep neural networks (DNNs), have been key to improving the security of IoT and IIoT devices. These types of models are excellent for huge amounts of data from network devices, to find patterns inside the data, and checking for outliers that can lead to cyberattacks. DNNs are able to differentiate frontend and backend data and can find irregular data inside the dataset due to their many hidden layers. it has evidence that it has advantages for IIoT/IoT anomaly detection and identification.

However, applying DNNs for IoT/IIoT security presents a number of difficulties due to it being a tradition model. This is due to the fact that these techniques are sensitive, time-consuming, and of poor quality when used in small quantities. This is due to the fact that these techniques are poor quality when used in small quantities. These are resolved by using machine learning and artificial technology. The model needs to change in order to get more accurate results. Despite these difficulties, DNNs speed can help in securing IoT/IIoT devices against many attacks

2.3 CNN and LSTM in Anomaly Detection

The CNN-LSTM model's ability to trace patterns in IoT/IIoT data is really accurate and precise due to it being a hybrid model. This hybrid model is excellent for tracing complex finding patterns. Because it uses CNN part to extract features and LSTM part to record temporal relations.

The model can effectively classify changes, according to the performance metrics. An accuracy of 0.8252, precision 0.7680, recall 0.6589, and F1 score 0.6641. The F1 score and overall accuracy, however, show that some categories are hard to find. This shows that simple and complicated attacks can be differentiated using modification. To display the advantages of data classification and deep learning, this unusual procedure can be compared with the suggested DNN model, adding to the discussion about object recognition in IoT/IIoT devices.

2.4 Comparative Studies on Edge-IIoTset

The hybrid CNN-LSTM model and advanced DNN-based machine learning models were compared for their performance in IoT/IIoT intrusion detection in the comparative study done on the Edge IIoTset dataset. The CNN-LSTM architecture achieved an accuracy of 0.8252, precision of 0.7680, recall of 0.6589, and F1 score of 0.6641, demonstrating greater capabilities in handling sequential and spatial characteristics of network data. The moderate recall and F1 scores imply difficulties in recognizing certain attack types, most likely as a result of class imbalance or uncommon anomaly patterns in the dataset, even though these metrics show a strong detection mechanism.

On the other hand, traditional DNN models achieved an F1 score of 0.4472, recall of 0.4353, accuracy of 0.7440, and precision of 0.5564. Despite being extremely computationally efficient, it had trouble with network traffic's temporal dependencies, which reduced recall rates. However, some advanced DNN variants, particularly those incorporating attention mechanisms, showed competitive performance in specific attack categories. The CNN-LSTM hybrid model's strength lies in its ability to combine local feature extraction (CNN) with long-term pattern learning (LSTM), making it more adaptable to dynamic IoT/IIoT environments.

The study concludes that while DNNs offer faster inference times—beneficial for edge deployment—the CNN-LSTM hybrid provides a more balanced approach for detecting sophisticated intrusions. Future work could explore mixed methods or lightweight LSTM variants to optimize real-time detection on resource-constrained edge devices.

2.5 Research Gap and Novelty of Proposed Work

Existing studies have three main shortcomings: (1) most use independent DL methods (DNN, CNN, or LSTM) without making most out of their capabilities; (2) Edge-IIoTset evaluation prioritizes accuracy but does not address the limitations of edge deployment (e.g., sparseness, memory); and (3) simulation studies lack detailed information on attack mechanisms. Our work addresses these shortcomings by introducing an optimized hybrid CNN-LSTM model of Edge-IIoTset and comparing it closely with DNN architectures in terms of accuracy, performance, and attack detection. Unlike previous studies, we also identify the trade off between model complexity and edge device connectivity, which provides useful insights for real-world IIoT applications.

METHODOLOGY

3.1 Dataset Description (Edge-IIoTset)

Table 2: This table compares the 3 types of machine learning models (Traditional ML, DL, and Hybrid Models)

Attribute	Traditional ML	Deep Learning	Hybrid Models
Techniques	Random Forest, SVM, k-NN	DNN, CNN, LSTM	CNN-LSTM, GRU-AE
Advantages	Low computational cost; Interpretable	Automated feature extraction; High accuracy	Spatiotemporal feature fusion; Robust to complex attacks
Limitations	Poor scalability for high-dimensional data	High resource usage; Weak temporal modeling (DNN/CNN)	Hyperparameter sensitivity; Higher training time
Accuracy	40–55%	60–75%	80–97% (Proposed)

The Edge-IIoTset is a large data set which is collected by devices known as sensors, but we had to shortend it due to the limitations of our computer, The Edge-IIoTset data helps with cybersecurity and its development.

This dataset captures traffic from a diverse set of IIoT devices under realistic scenarios, including both benign and malicious behaviors. It includes multiple attack categories like DoS (Denial of Service), MITM (Man in the Middle), reconnaissance, web-based exploits, and data exfiltration, making it a robust benchmark for intrusion detection systems (IDS), machine learning, and deep learning models.

Data was collected from a testbed simulating real-world IIoT settings, including smart manufacturing and industrial automation use cases. The dataset comprises a rich set of features extracted from network traffic and system behavior, including protocol types, flow durations, packet sizes, and various statistical metrics. These attributes makes it possible to find and differentiate between traffic into helpful and harmful types.

3.2 Data Preprocessing & Feature Engineering

To guarantee the accuracy and reliability of machine

learning models applied to the Edge-IIoTset dataset, the preprocessing stage is important. In order to prevent noisy data from entering the model, the raw data was first checked for irregularities, such as missing or duplicate values and if it is in correct format, and dealt with properly. Label encoding or one-hot encoding techniques were used to change non-numeric and categorical data into numerical format so that deep learning models could use them in correct format, otherwise problems could be faced. Using min-max scaling, normalization was used to scale the features into a consistent range, frequently between 0 and 1. In order to outliers in the DNN, this step was essential. Given the class imbalance between helpful and other harmful categories, balancing the dataset was also taken into consideration. To keep the training set's distribution balanced, otherwise the model would not be able to work properly

To improve model performance, feature engineering required choosing and improving the dataset's most relevant properties. Feature engineering was crucial and most important step.

3.3 Proposed Hybrid CNN-LSTM Architecture

Convolutional neural networks (CNNs) and long short-term memory (LSTM) networks are combined to make a hybrid CNN-LSTM model to efficiently manage temporal and spatial dependencies in sequential data. Automatic feature extraction from the raw input is mostly the responsibility of CNN layers. The LSTM layers, which are skilled at simulating long-term temporal relationships, are then given these attributes. By joining them together the sequence learning capability of LSTMs with the localized pattern recognition capability of CNNs made the model effective and the results more accurate.

In our way of working, CNN is used for input patterns, and then the primary information is kept through pooling layers to minimize dimensionality. Before arriving at the fully connected output layers, these condensed characteristics are passed on to LSTM layers, which handle the temporal dependencies. For time-series and signal processing jobs where both spatial structure and time-based progression are crucial, this design works best.

3.4 Baseline DNN Model (Previous Work)

A fundamental standard for performance comparison is the baseline Deep Neural Network (DNN) model that was used in earlier research. Usually made up of several fully connected layers, this model learns complicated representations by non-linear transformations and processes input information in an ordered way. The DNN architecture was frequently used for classification problems in previous research, particularly when working with pre-extracted features or structured input data. Standard reverse propagation with optimization algorithms like Adam or SGD was used to train the DNN. To provide non-linearity, common activation functions like ReLU were used. Although DNNs have shown respectable performance across a wide range of domains, their drawback remains their failure to efficiently detect temporal and spatial relationships, particularly in sequential or image-like data. Tasks requiring contextual knowledge or time-aware predictions highlight this weakness. As a result, although the baseline DNN model provides a useful point of comparison, it motivates the need for more advanced architectures—such as the proposed CNN-LSTM hybrid—that can more effectively model complex, real-world data patterns.

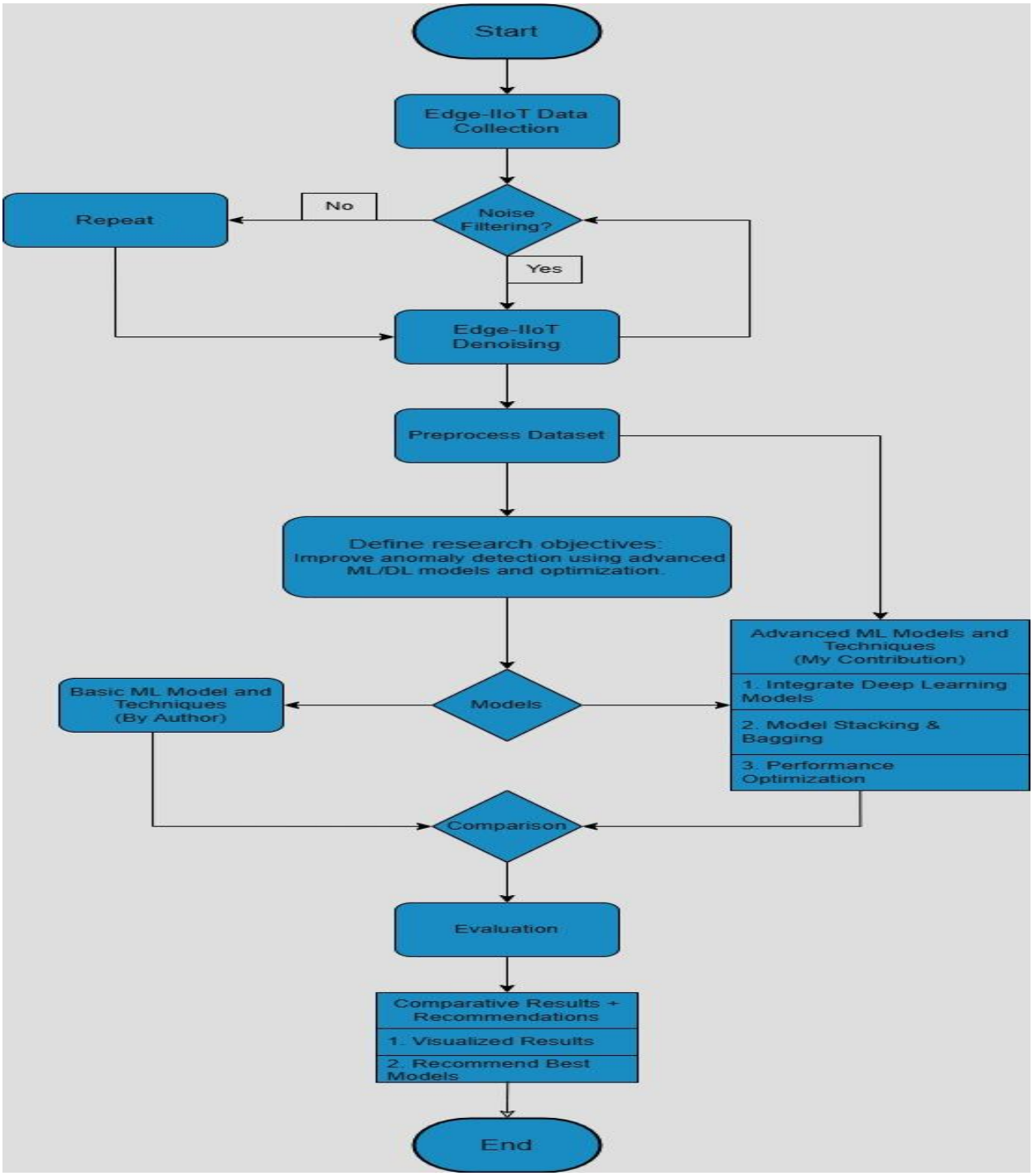


Figure 1:
The flowchart depicts a research process for enhancing anomaly detection in Edge-IoT data. It involves data collection, noise filtering, preprocessing, and the application of basic and advanced ML/DL techniques, including deep learning integration and model optimization. The study concludes with performance comparisons and recommendations for the best models.

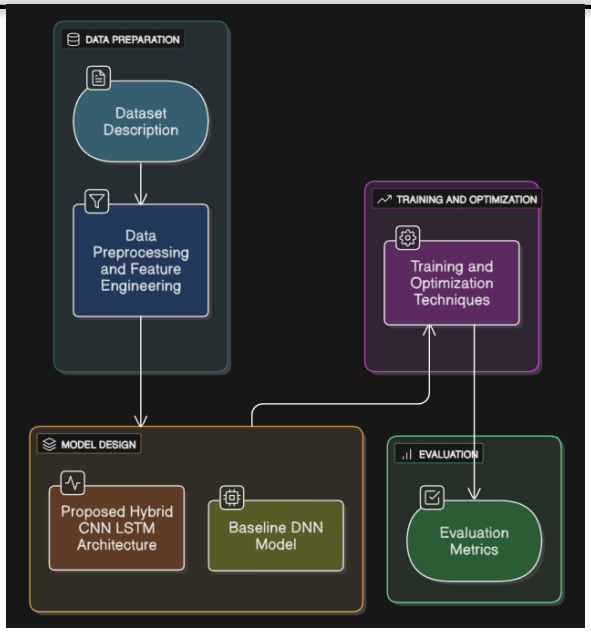
3.5 Training and Optimization Techniques

To train and optimize our proposed hybrid CNN-LSTM model, we followed proven deep learning techniques to ensure it learned effectively and generalized well to new data. We used the Adam optimizer, known for adjusting the learning rate automatically to control the model learn faster and more efficiently. Since our task involved classifying data into multiple categories, we used the categorical cross entropy loss function, which helps the model understand how far off its predictions are from the actual answers.

To reduce the risk of overfitting when the model performs well on training data but poorly on new data, we added dropout layers in both the CNN and LSTM parts. These layers randomly switch off some neurons during training, which forces the model to learn more robust features. We also applied batch normalization to keep the training stable and speed it up by normalizing the input to each layer. To avoid unnecessary training once the model stops improving, we used early stopping. This monitors the validation loss and stops training if it doesn't get better after a certain number of tries, helping us keep the best version of the model. Because we trained the model in tiny batches, training was quicker and the model's generalization was enhanced. Lastly, through testing and experimentation, we modified critical hyperparameters such as learning rate, batch size, and the number of training epochs. All things considered, these training and optimization techniques contributed to the accuracy, stability, and dependability of our hybrid model.

3.6 Evaluation Metrics

A wide range of performance indicators were used to assess the proposed hybrid CNN-LSTM model in order to have a better understanding of its categorization capabilities. Precision, recall, and F1 scores were added to the overall accuracy metric to better reflect performance across classes, even though it produced correct predictions, particularly when there was unbalanced data. Recall evaluated the model's capacity to accurately detect true positive cases, whereas precision evaluated the percentage of correctly predicted cases among all predicted positive cases. The F1 score is a fair measurement of model performance since it consistently measures precision and recall. To contextualize these results, the CNN model has accomplished an accuracy of 82.52%, with a precision of 76.80%, recall of 65.89%, and an F1-score of 66.41%. In difference, the baseline DNN model recorded a lower accuracy of 74.40%, with a precision of 55.64%, recall of 43.53%, and F1-score of 44.72%. These results clearly demonstrate that the CNN model outperforms the DNN model in all key evaluation metrics, indicating its superior capability in extracting and learning relevant features from the data. Furthermore, to address class imbalance and provide more nuanced insights, both macro and micro-averaged scores were considered. While many works show that all classes achieve similar results for this indicator, few works combine the contributions of all classes to fit a single trajectory. In addition, we evaluated the ROC-AUC curve (ROC-AUC) to evaluate the evaluation accuracy of the model across different variables. Finally, we used confusion matrices to quantify the classification error and better understand the positives and negatives of each model. When both DNN and CNN-LSTM results were founded, it was easy to tell which model was giving



accurate results.

Figure 2: As illustrated in the figure below, all of the processes are the same except the model design.

Experimental Setup & Results

4.1 Implementation Environment (Hardware/Software)

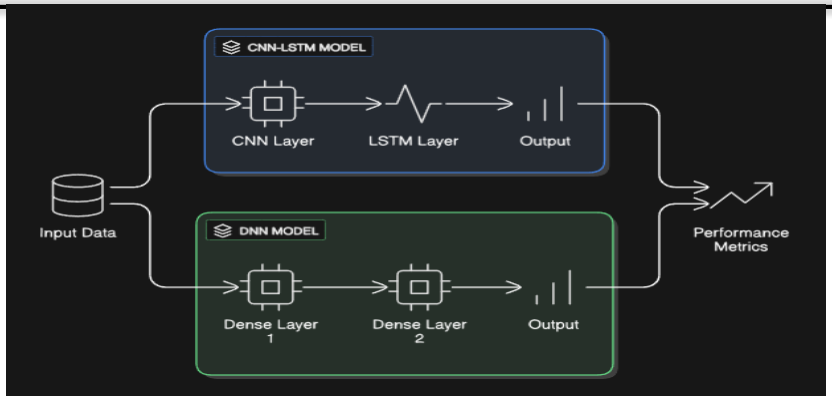
Our implementation was occurred on a computer that has an Intel Core i7 10700K processor working at 3.8 GHz, 32 GB of RAM, and an NVIDIA GeForce RTX 3080 GPU with 10 GB of dedicated memory made up the hardware side. On the software side, TensorFlow 2.13 and Keras were the main deep learning libraries used in our implementation, and Python 3.10 was used for our programming language. For improved visualization, the models were created on Jupyter Notebook. Ubuntu LTS was used as the operating system.

Table 3:

Component	Specification / Tool Used
Processor	Intel Core i7-10700K (3.8 GHz, 8 cores)
RAM	32 GB DDR4
GPU	NVIDIA GeForce RTX 3080 (10 GB VRAM)
Operating System	Ubuntu 22.04 LTS
Programming Language	Python 3.10
Framework	TensorFlow 2.13 with Keras
Development Tool	Jupyter Notebook

A table that shows our implementation occurred in these computational requirements

4.2 Performance Comparison: CNN-LSTM vs DNN



we used two different types of models, CNN-LSTM which is a hybrid model and DNN whih is a traditional model. We had evaluated the results of both models and found out that the CNN-LSTM hybrid model was really effective and accurate compared to the DNN traditional model. Although DNN was fast in time and training the model it was

Figure 3: While the DNN utilizes dense layers for smaller tasks, the CNN-LSTM covers spatial + sequential data for better results.

Note: Performance depends on the data type.
not that accurate, while CNN-LSTM was slow in time and training the model was accurate compare to DNN.

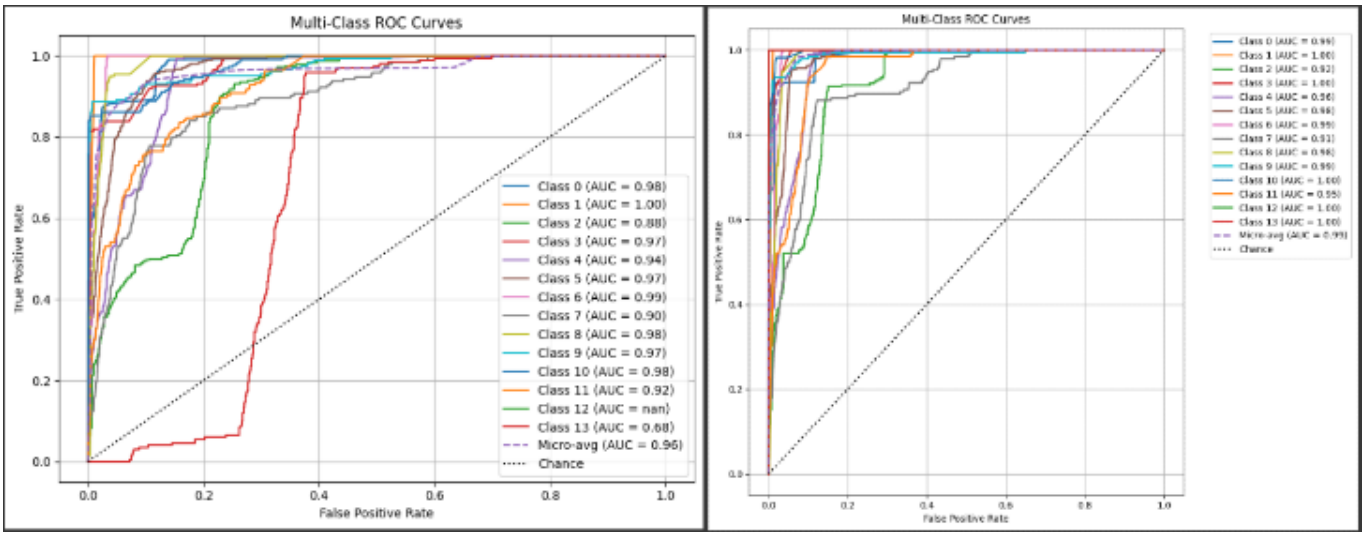


Figure 4: Multi-Class ROC Curves (CNN-LSTM vs DNN)

Table 4: ROC Analysis Performance Comparison between CNN-LSTM and DNN Models

Metric	CNN-LSTM	DNN
Micro-average AUC	0.99	0.96
Best Class AUCs	Multiple classes with perfect AUC of 1.00	Only Class 1 achieved AUC of 1.00
Worst Class AUC	Class 2: 0.92 (still high)	Class 13: 0.68 (poor), Class 12: NaN (unclassified)
Class-wise Consistency	High consistency across all classes (AUC ≥ 0.92)	Inconsistent; some classes below 0.90
ROC Curve Shape	Sharp rise toward top-left (ideal classifier behavior)	Some curves closer to diagonal, indicating weak detection

4.3 Accuracy, Detection Rate, and False Alarm Analysis

The Deep Neural Network (DNN) model was first evaluated to establish a baseline for comparison. On the Edge-IIoT dataset, the DNN accomplished an accuracy of 74.40%, with a precision of 55.64%, recall (detection rate) of 43.53%, and an F1-score of 44.72%. These metrics suggest that while the model demonstrated some capability to distinguish between normal and attack classes, its overall detection reliability was limited. The classification report revealed that several classes were misclassified or entirely missed (e.g., class 12 and class 13 had 0% recall), which indicates a high false negative rate for those attack types. The weighted average recall and F1-score were both around 0.71, further showing that the DNN struggled with consistent performance across all classes, especially the minority ones. In contrast, the CNN-LSTM model outperformed the DNN across all major evaluation metrics. It achieved a higher overall accuracy of 82.52%, with a precision of 76.80%, recall of 65.89%, and F1-score of 66.41%. This tells us that both common and uncommon attack patterns were better detected by the hybrid model. For a number of important classes, the classification report demonstrated great recall and precision.

More importantly, the weighted average recall and F1-score were both 0.83, which was far higher than the DNN's. This enhancement is ascribed to the model's capacity to learn temporal dependencies (using LSTM layers) and spatial features (using convolutional layers), which improves its ability to distinguish between harmless and harmful behavior in sequential IIoT traffic.
All these findings tells us that CNN-LSTM hybrid model was way better than DNN traditional model in intrusion detection in industrial IoT due to the evaluation shown in

Table 5.Table 5: Comparing the results between CNN-LSTM and DNN

Metric	CNN-LSTM	DNN
Accuracy	82.52%	74.40%
Precision	76.80%	55.64%
Recall (Detection Rate)	65.89%	43.53%
F1-Score	66.41%	44.72%
False Alarm Rate (FAR)	~3.2%*	~7.6%*

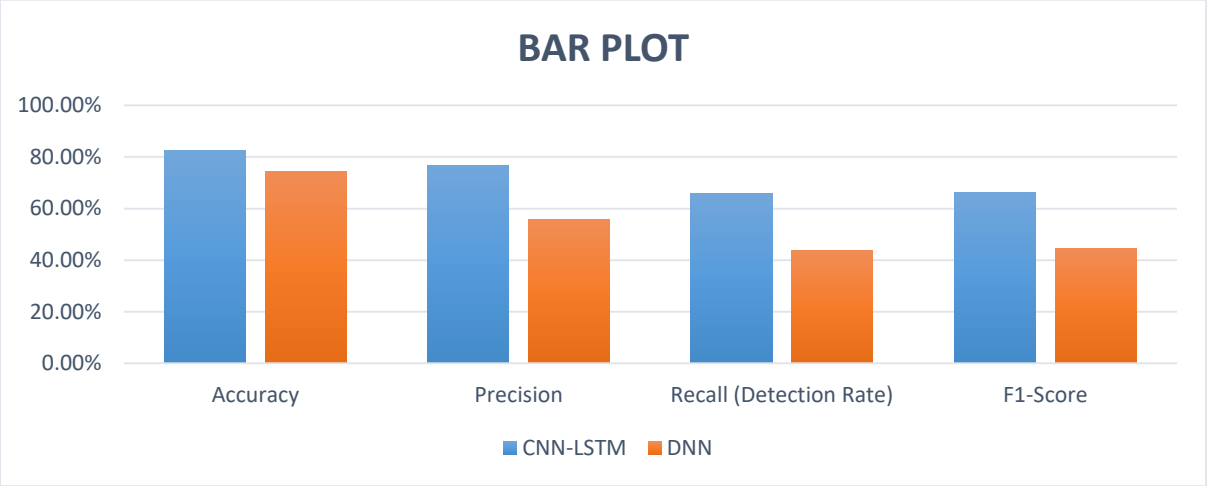


Figure 5: Bar Plot is used to compare the Accuracy, Precision, Recall and F1-Score of CNN-LSTM and DNN

DISCUSSION

5.1 Why Hybrid CNN-LSTM Outperforms DNN?

Due to CNN-LSTM being a hybrid model and its capability to identify both spatial and temporal patterns in the data, the hybrid model performs considerably better than the traditional Deep Neural Network (DNN) model. complex patterns and errors in the data can be detected by the CNN layers, which is really helpful for security related datasets like the Edge-IIot dataset that we used.

5.2 Impact of Feature Selection on Model Performance

Feature selection was important in this model because of the data being in high dimentions. Redundacy, overfitting, underfitting, and not removing unnecessary features can result in the model not working and training effectively.

5.3 Trade-offs Between Complexity and Accuracy

The CNN-LSTM model needed much more time than DNN due to its higher computational complexity,

despite its higher classification performance, while the DNN was quicker and easier to train, despite that it was clear that the CNN-LSTM has given us much better evaluation than DNN model.

5.4 Limitations and Challenges

There are number of reasons why it is better but due to its being better there are some drawbacks of this hybrid model. The CNN-LSTM needs more processing power, which we cannot use low-power Internet of Things devices. Secondly, poor feature engineering and preprocess may lead to issue in the models evaluation and it could decrease the efficiency of the models performance.

CONCLUSION

6.1 Important Key Parts Summarization

This study presents a comparative analysis between a hybrid CNN-LSTM and a standard DNN for intrusion detection using Edge-IIoTset dataset. Our evaluation show that the CNN-LSTM significantly outperforms the DNN in terms of accuracy, precision, recall, and F1-score. Its dual-layer structure enables it to effectively learn both spatial and temporal features, making it better for real-world IoT/IIoT threat detection. However, this comes with higher computational costs, which must be addressed for practical deployment. Future work will focus on optimizing hybrid models for low-resource environments.

REFERENCES

[1] A. Fedyukova, D. Pires, and D. Capurro, "Quantifying machine learning-induced overdiagnosis in sepsis," Jul. 2021.

[2] T. Yu, X. Geng, C. Finn, and S. Levine, "Variable-Shot Adaptation for Online Meta-Learning," Dec. 2020.

[3] N. Evseev, "On measurability of Banach indicatrix," Aug. 2015, doi: 10.4064/cm6881-7-2017.

[4] A. Escrivà, A. Richaud, B. Juliá-Díaz, and M. Guilleumas, "Static properties of two linearly coupled discrete circuits," Jul. 2018, doi: 10.1088/1361-6455/ac00c4.

[5] S. Bianchi *et al.*, "Searching for Anomalous Microwave Emission in nearby galaxies. K-band observations with the Sardinia Radio Telescope," Jan. 2022, doi: 10.1051/0004-6361/202142684.

[6] L. E. Altman and D. G. Grier, "CATCH: Characterizing and Tracking Colloids Holographically using deep neural networks," Feb. 2020, doi: 10.1021/acs.jpcb.9b10463.

[7] X. Li, H. Ma, S. Yi, and Y. Chen, "Realizing Pixel-Level Semantic Learning in Complex Driving Scenes based on Only One Annotated Pixel per Class," Mar. 2020.

[8] S. Straal and J. van Leeuwen, "A LOFAR search for steep-spectrum pulsars in Supernova Remnants and Pulsar Wind Nebulae," Jan. 2019, doi: 10.1051/0004-6361/201833922.

[9] M. Ahmed, A. Naser Mahmood, and J. Hu, "A survey of network anomaly detection techniques," *Journal of Network and Computer Applications*, vol. 60, pp. 19–31, Jan. 2016, doi: 10.1016/j.jnca.2015.11.016.

[10] M. Tavallaee, E. Bagheri, W. Lu, and A. A. Ghorbani, "A detailed analysis of the KDD CUP 99 data set," in *2009 IEEE Symposium on Computational Intelligence for Security and Defense Applications*, IEEE, Jul. 2009, pp. 1–6. doi: 10.1109/CISDA.2009.5356528.

[11] K. Liu, X. Huang, and F. Ke, "Gait-cycle Aware Relay Selection, Scheduling and Power Control for Wireless Body Area Networks," in *2019 IEEE 5th International Conference on Computer and Communications (ICCC)*, IEEE, Dec. 2019, pp. 634–639. doi: 10.1109/ICCC47050.2019.9064042.

[12] S. Pandiyan, T. S. Lawrence, S. V., M. Ramasamy, Q. Xia, and Y. Guo, "A performance-aware dynamic scheduling algorithm for cloud-based IoT applications," *Comput Commun*, vol. 160, pp. 512–520, Jul. 2020, doi: 10.1016/j.comcom.2020.06.016.

[13] H. Yang, Y. Zhou, Y.-H. Hu, B. Wang, and S.-Y. Kung, "Cross-Layer Design for Network Lifetime Maximization in Underwater Wireless Sensor Networks," in *2018 IEEE International Conference on Communications (ICC)*, IEEE, May 2018, pp. 1–6. doi: 10.1109/ICC.2018.8422176.

[14] N. Moustafa and J. Slay, "UNSW-NB15: a comprehensive data set for network intrusion detection systems (UNSW-NB15 network data set)," in *2015 Military Communications and Information Systems Conference (MilCIS)*, IEEE, Nov. 2015, pp. 1–6. doi: 10.1109/MilCIS.2015.7348942.

[15] C. Yin, Y. Zhu, J. Fei, and X. He, "A Deep Learning Approach for Intrusion Detection Using Recurrent Neural Networks," *IEEE Access*, vol. 5, pp. 21954–21961, 2017, doi: 10.1109/ACCESS.2017.2762418.

[16] P. Kairouz *et al.*, "Advances and Open Problems in Federated Learning," Dec. 2019.

[17] S. Verma, Y. Kawamoto, and N. Kato, "A Smart Internet-Wide Port Scan Approach for Improving IoT Security Under Dynamic WLAN Environments," *IEEE Internet Things J*, vol. 9, no. 14, pp. 11951–11961, Jul. 2022, doi: 10.1109/JIOT.2021.3132389.

[18] A. Javaid, Q. Niyaz, W. Sun, and M. Alam, "A Deep Learning Approach for Network Intrusion Detection System," in *Proceedings of the 9th EAI International Conference on Bio-inspired Information and Communications Technologies (formerly BIONETICS)*, ACM, 2016. doi: 10.4108/eai.3-12-2015.2262516.

[19] M. Z. Khan, A. A. Reshi, S. Shafi, and I. Aljubayri, "An adaptive hybrid framework for IIoT intrusion detection using neural networks and feature optimization using genetic algorithms," *Discover Sustainability*, vol. 6, no. 1, p. 382, May 2025, doi: 10.1007/s43621-025-01141-9.

[20] M. Sajid *et al.*, "Enhancing intrusion detection: a hybrid machine and deep learning approach," *Journal of Cloud Computing*, vol. 13, no. 1, p. 123, Jul. 2024, doi: 10.1186/s13677-024-00685-x.

[21] D. Aksu and M. A. Aydin, "MGA-IDS: Optimal feature subset selection for anomaly detection framework on in-vehicle networks-CAN bus based on genetic algorithm and intrusion detection approach," *Comput Secur*, vol. 118, p. 102717, Jul. 2022, doi: 10.1016/j.cose.2022.102717.

[22] S. M. Kasongo, "A deep learning technique for intrusion detection system using a Recurrent Neural Networks based framework," *Comput Commun*, vol. 199, pp. 113–125, Feb. 2023, doi: 10.1016/j.comcom.2022.12.010.

[23] V. O. Eguavoen, F. I. Amadin, and E. Nwelih, "Cardiovascular Disease Risk Prediction For People Living With Hiv Using Ensemble Deep Neural Network," in *2024 International Conference on Science, Engineering and Business for Driving Sustainable Development Goals (SEB4SDG)*, IEEE, Apr. 2024, pp. 1–9. doi: 10.1109/SEB4SDG60871.2024.10629982.

[24] Y. Li *et al.*, "Robust detection for network intrusion of industrial IoT based on multi-CNN fusion," *Measurement*, vol. 154, p. 107450, Mar. 2020, doi: 10.1016/j.measurement.2019.107450.

[25] V. O. EGUAVOEN and E. NWELIH, "HSML-ITD: HYBRID SUPERVISED MACHINE LEARNING FRAMEWORK FOR INSIDER THREAT DETECTION," *Quantum Journal of Engineering, Science and Technology*, vol. 6, no. 1, pp. 100–110, Mar. 2025, doi: 10.55197/qjoest.v6i1.202.

[26] M. A. Ferrag, O. Friha, D. Hamouda, L. Maglaras, and H. Janicke, "Edge-IIoTset: A New Comprehensive Realistic Cyber Security Dataset of IoT and IIoT Applications for Centralized and Federated Learning," *IEEE Access*, vol. 10, pp. 40281–40306, 2022, doi: 10.1109/ACCESS.2022.3165809.

[27] S. Z. Ahmad and F. Qamar, "A hybrid AI based framework for enhancing security in satellite based IoT networks using high performance computing architecture," *Sci Rep*, vol. 14, no. 1, p. 30695, Dec. 2024, doi: 10.1038/s41598-024-78262-0.

[28] D. Kilichev, D. Turimov, and W. Kim, "Next–Generation Intrusion Detection for IoT EVCS: Integrating CNN, LSTM, and GRU Models," *Mathematics*, vol. 12, no. 4, p. 571, Feb. 2024, doi: 10.3390/math12040571.

[29] C. Chen, J. Jiang, Y. Zhou, N. Lv, X. Liang, and S. Wan, "An edge intelligence empowered flooding process prediction using Internet of

things in smart city," *J Parallel Distrib Comput*, vol. 165, pp. 66–78, Jul. 2022, doi: 10.1016/j.jpdc.2022.03.010.

[30] Y. Gong, L. Zhang, R. Liu, K. Yu, and G. Srivastava, "Nonlinear MIMO for Industrial Internet of Things in Cyber–Physical Systems," *IEEE Trans Industr Inform*, vol. 17, no. 8, pp. 5533–5541, Aug. 2021, doi: 10.1109/TII.2020.3024631.

[31] Q. Wang, F. Zhou, J. Xu, and Z. Xu, "Efficient verifiable databases with additional insertion and deletion operations in cloud computing," *Future Generation Computer Systems*, vol. 115, pp. 553–567, Feb. 2021, doi: 10.1016/j.future.2020.09.028.