Journal of Advanced Zoology
ISSN: 0253-7214
Volume 45 Issue 3 Year 2024 Page 801-810

Iot Threat Mitigation: Leveraging Deep Learning For Intrusion Detection

Dr. Ch. Suresh Babu[1*], Boppa Sri Satya Sai Hruday[2], Jonnala Veera Venkata Sai Krishna[3], Chellinki Sandeep[4], Boddu Naveen[5]

[1]Dept. of IT, SR Gudlavalleru Engineering College, Gudlavalleru, Andhra Pradesh-521356*
Email:- Dr.sureshbabuch@gmail.com
[2]Dept. of IT, SR Gudlavalleru Engineering College, Gudlavalleru, Andhra Pradesh-521356
Email:- hruday.boppa@gmail.com
[3]Dept. of IT, SR Gudlavalleru Engineering College, Gudlavalleru, Andhra Pradesh-521356
Email:- saikrishnajonnala888@gmail.com
[4]Dept. of IT, SR Gudlavalleru Engineering College, Gudlavalleru, Andhra Pradesh-521356
Email:- sandeepchellinki1807@gmail.com
[5]Dept. of IT, SR Gudlavalleru Engineering College, Gudlavalleru, Andhra Pradesh-521356
Email:- naveenyadav3362@gmail.com

***Corresponding Author:** Dr. Ch. Suresh Babu*
**Dept. of IT, SR Gudlavalleru Engineering College, Gudlavalleru, Andhra Pradesh-521356*
Email:- Dr.sureshbabuch@gmail.com

Abstract

The growth of smart gadgets connected via the Internet of Things (IoT) in today's modern technology landscape has substantially improved our everyday lives. However, this convenience is juxtaposed with a concomitant surge in cyber threats capable of compromising the integrity of these interconnected systems. Conventional intrusion detection systems (IDS) prove inadequate for IoT due to the unique challenges they present. We propose and evaluate an intrusion detection system (IDS) based on a hybrid Convolutional Neural Network-Long Short-Term Memory (CNN-LSTM) model in this paper. The model is designed to capture both temporal and spatial patterns in network data, offering a robust solution for detecting malicious activities within IoT environments. The CNN-LSTM model displayed excellent accuracy, reaching 98% in both multi-class and binary classifications when trained on the UNSW-NB15 dataset. Furthermore, we explore the real-world applicability of the model through testing on Raspberry Pi, showcasing its effectiveness in IoT scenarios. The system is augmented with alert mechanisms, promptly notifying relevant parties upon intrusion detection. Our findings highlight the CNN-LSTM model's efficacy in strengthening IoT network security.

Keywords - IoT Security, Deep Learning, Intrusion Detection, Cybersecurity, Network Data Analysis, Raspberry Pi, Alert Mechanisms.

I. INTRODUCTION

The swift growth of the IoT has culminated in a moment of unmatched connectedness, integrating smart capabilities into ordinary things ranging from domestic appliances to industrial gear. This transformative integration, while enhancing efficiency and convenience, has concurrently exposed a multifaceted landscape of cybersecurity challenges. As IoT devices permeate diverse sectors, the need to fortify these interconnected ecosystems against malicious activities becomes not only a technological imperative but a crucial aspect of safeguarding sensitive data, privacy, and critical infrastructures. Amidst the promising landscape of IoT applications, security concerns loom large. The inherently heterogeneous and dynamic nature of IoT environments poses unique challenges for traditional security paradigms. Conventional intrusion detection systems often struggle to adapt to the diverse data formats, communication protocols, and contextual intricacies inherent in IoT networks. In this context, the development of advanced and adaptive security mechanisms is imperative and the IoT system's integrity, confidentiality, and availability must be ensured.

In response to these challenges, this paper introduces a cutting-edge Intrusion Detection System (IDS) built upon a hybrid CNN-LSTM model. This innovative approach capitalizes on the advantages of both Convolutional Neural Networks (CNNs) and Long Short-Term Memory (LSTM) networks, aiming to address the nuanced security requirements of IoT environments. The CNN-LSTM model designed with dual purpose: to capture spatial features through CNNs, adept at recognizing patterns in data, and to understand temporal dependencies through LSTMs, which excel at modelling sequential information. This amalgamation allows the model to comprehend the intricacies of IoT network traffic, discerning normal behaviour from potentially malicious activities in a holistic and adaptive manner. The assessment of the proposed CNN-LSTM model extends beyond theoretical evaluations. We utilize the UNSW-NB15 dataset, a comprehensive repository of diverse network data encompassing various attack scenarios, to rigorously evaluate the model's accuracy in both multi-class and binary classifications. Our objective is to demonstrate the robustness of the model in distinguishing between benign and malicious network behavior across different attack categories [8].

Furthermore, recognizing the practical constraints and real-time demands inherent in IoT scenarios, we conduct experiments deploying the CNN-LSTM model on Raspberry Pi, a widely adopted platform in IoT applications. This real-world validation serves to illustrate the model's applicability, efficiency, and resource adaptability in authentic IoT environments. In subsequent sections, we provide a detailed exploration of the CNN-LSTM model's architecture, elucidating the underlying mechanisms that empower it to effectively tackle the intricacies of IoT network security. Additionally, we delve into the broader implications of IoT security breaches, emphasizing the pressing need for advanced intrusion detection mechanisms to fortify IoT ecosystems. Through this research, we aim not only to contribute a robust solution to the evolving challenges of securing IoT environments but also to stimulate discussions and innovations in the realm of adaptive and resilient IoT security. As we navigate the intricate landscape of IoT applications, the development of sophisticated security measures becomes pivotal in ensuring the continued growth and reliability of these interconnected systems.

II. LITERATURE REVIEW

The landscape of Intrusion Detection Systems (IDS) is vast and has witnessed significant evolution in response to the ever-growing sophistication of cyber threats. A comprehensive review of the literature reveals diverse approaches, methodologies, and algorithms employed in the quest for enhancing the security posture of networked systems. The following sections provide an overview of key themes in IDS literature, encompassing traditional methods, machine learning-based approaches, as well as the current issues provided by the IoT.

Traditional IDS methods have laid the foundation for intrusion detection research, primarily focusing on rule-based and signature-based detection. Rule-based IDS operate by defining explicit rules that characterize known attack patterns. While effective against known threats, their rigidity renders them susceptible to evasion through novel attack vectors. Signature-based IDS rely on a database of predefined signatures or patterns associated with known attacks, making them efficient for recognizing well-established threats. However, their efficacy diminishes in the face of polymorphic or zero-day attacks [4][6].

Anomaly-based detection methods depart from the predefined patterns of signature-based approaches, seeking to identify deviations from normal system behaviour. Machine learning, particularly unsupervised learning, plays a pivotal role in anomaly detection. One-class SVMs, clustering algorithms, and neural networks are commonly employed to model normal behaviour and flag deviations as potential intrusions. While anomaly detection excels at identifying novel threats, it also introduces challenges related to false positives and the dynamic nature of network behavior [2][3].

The integration of machine learning techniques has ushered in a new era in intrusion detection, leveraging the power of data-driven models to adapt and learn from evolving threats. Supervised learning algorithms, such as Support Vector Machines (SVM) and Random Forests, utilize labelled datasets to train models for accurate classification. Unsupervised learning, exemplified by clustering algorithms and autoencoders, excels in discerning anomalies without explicit labelling [1][5].

Recent advancements in deep learning have propelled the field of intrusion detection towards more sophisticated and adaptive solutions. CNNs and Recurrent Neural Networks (RNNs) offer the ability to extract temporal and spatial features, respectively, enhancing the detection of complex attack patterns. Hybrid models, such as the combination of CNNs and LSTMs networks, have shown promise in capturing both temporal and spatial aspects, particularly relevant in dynamic network environments [7][8].

The emergence of the Internet of Things presents a unique set of challenges for intrusion detection. The heterogeneity of IoT devices, diverse communication protocols, and the sheer volume of data generated necessitate tailored security measures. Traditional IDS models face limitations in adapting to the idiosyncrasies of IoT environments. Machine learning and deep learning-based IDS, especially those adepts at handling sequential and time-series data, have gained traction in mitigating IoT-specific security risks [9][10].

III. PROPOSED METHODOLOGY

Figure 1 illustrates the innovative procedure within the proposed framework, delineating the entire operational process of IDS. Specifically, the devised framework comprises five levels, showcasing uniqueness in the following stages:

3.1 Utilized Dataset:

Developed by IXIA's Perfect Storm in partnership with the UNSW Cyber Range Lab, the UNSW-NB15 dataset offers a collection of moderately intense attack simulations and network traffic data. This dataset, which was published in 2015 [9], simulates over 2,50,000 packets sent through the network. It encompasses nine attack categories (Reconnaissance, Exploits, Shellcode, Backdoors, Worms, DoS (Denial of Service), Fuzzers, Generic, Analysis), alongside normal/regular packets. Notably, over 87% of the packets belong to the normal type, resulting in a highly imbalanced dataset. A breakdown of packet distribution is provided in Table 1.
Further information about this dataset can be referenced in [11].

Table 1. UNSW-NB15 Dataset Packets Distribution

Category	Records Count	Percentage
Backdoor	2,329	0.10
Worms	174	0.01
Reconnaissance	13,987	0.55
Fuzzers	24,246	0.95
DoS	16,353	0.64
Exploits	44,525	1.75
Analysis	2,677	0.11
Normal	2,218,761	87.35
Generic	215,481	8.48
Shellcode	1,511	0.06
Total	**2,540,044**	**100**

3. 2. Data-Preprocessing

Data preprocessing entails the collection and manipulation of digital data, altering the values within a specified dataset. It involves modifying the information perceived by the observer to enhance the process of acquiring information. Generally, datasets display a significant variation in their min and max values. Normalization is a crucial step in simplifying algorithms and facilitating effective utilization of data for algorithmic classification, particularly within the realm of neural networks [10]. The fundamental approach to normalization is data scaling, employing algorithms based on minimum and maximum values. This process entails converting data into a specified range, typically within the intervals of (1, 1) or (0, 1). Furthermore, prior to normalization, the standardization function is often employed.

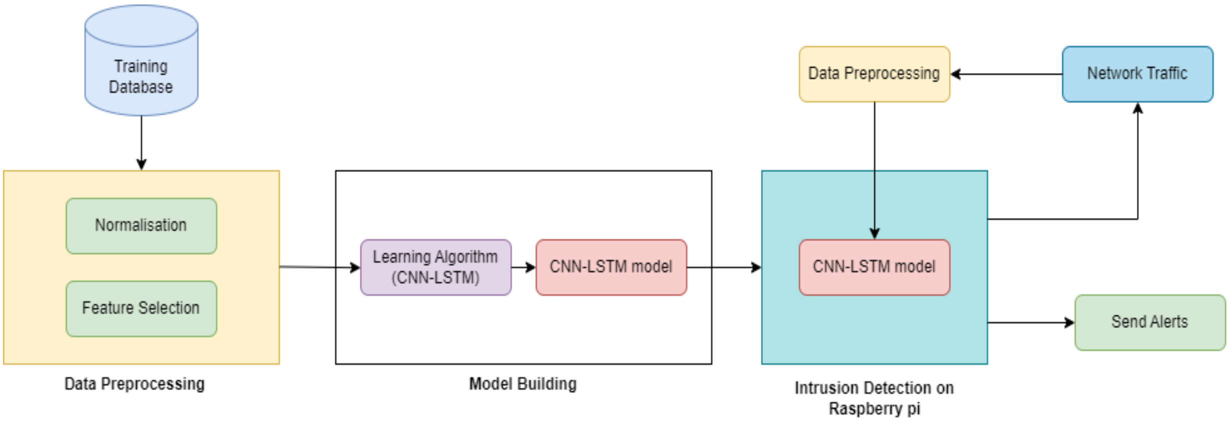


Figure 1. Proposed CNN-LSTM-Based Framework for Intrusion Detection in the IoT Environment

3.3Learning Component

The outputs from the CNN and LSTM layers are concatenated to form a comprehensive feature representation. This hybrid integration ensures that the model captures both spatial nuances, crucial for recognizing attack patterns, and temporal dynamics, vital for discerning evolving threats over time.

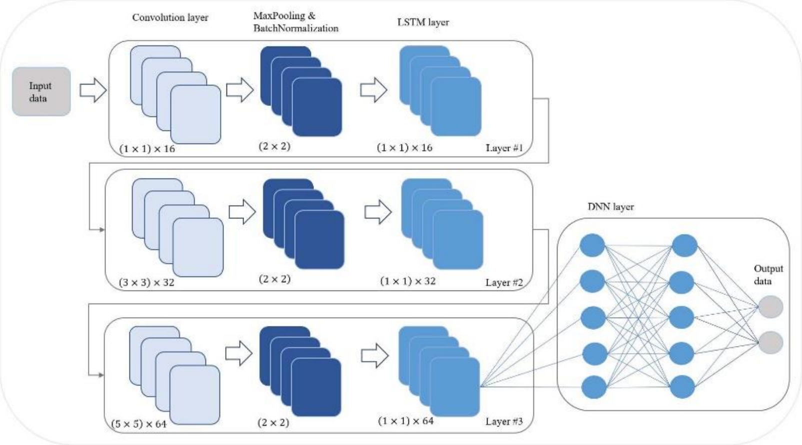


Figure 2. CNN_LSTM Architecture

CNN-LSTM - Hybrid Model Architecture and Working

Figure 2 depicts the CNN-LSTM model layers structure architecture. The following are the details of within each layer and how input is transformed as it moves through the layers in your CNN-LSTM model:

1. Conv1D Layer:

Input: The input to the Conv1D layer is a sequence of data with spatial features.

Operation: The Conv1D layer applies a convolution operation to capture local patterns in the data. It uses a set of filters to scan the input sequence.

$$Conv(a,b) = \sum(c,d)\,(Input(a + c, b + d) * Filter(c,d)) + Bias$$

Where a, b represents spatial coordinates within input data and c, d are the relative positions of the filter elements.

$Bias$ – Additional degree of freedom or flexibility in modeling.

Output: The Conv1D layer produces a succession of feature maps, each of which reflects the existence of certain patterns or features in the input sequence. The number and size of feature maps are determined by the no. of filters and the size of kernal.

2. MaxPooling1D Layer:

Input: The input to the MaxPooling1D layer is the sequence of feature maps generated by the Conv1D layer.

Operation: By picking the greatest value inside a frame, maxpooling down samples the feature maps. This method shrinks the spatial dimension while retaining the most important aspects.

$$MaxPooling(i,j) = max(Input(i,j), Input(i + 1,j), Input(i,j + 1), Input(i + 1,j + 1))$$

Where i,j represents the spatial coordinates within the data

Output: The down sampled series of feature maps with reduced spatial dimensions produced by the MaxPooling1D layer.

Now this output of MaxPooling1D goes through the LSTM layers, gets computed there and at the end finally gives the output.

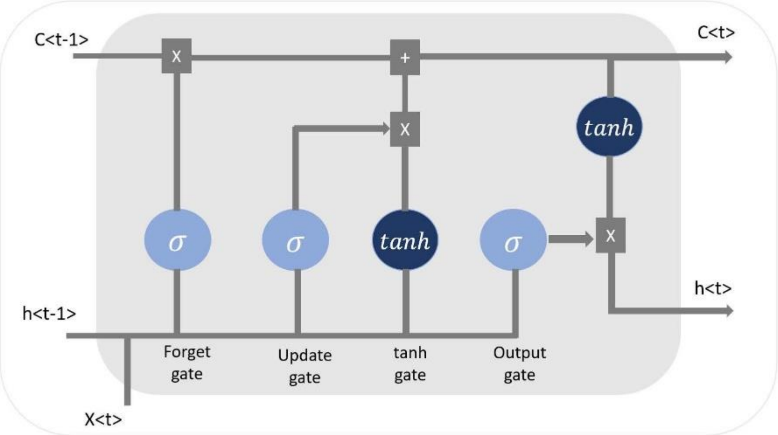


Figure 3. LSTM Structure

3. LSTM Layer:

Input: The sequence of down sampled feature maps from the MaxPooling1D layer is input to the LSTM layer.

Operation: The LSTM layer uses gates such forget gate, input gate, output gate and memory cells to analyze the sequence data and record long-term dependencies.

Figure 3 shows the LSTM structure.

For a single time, step (t), the equations are as follows:

i. Forget Gate (ft):

Whether the previous cell state$(Ct - 1)$ information should be forgotten or not determined by the forget gate.

$$ft = \sigma(Wf * [ht - 1, Xt] + bf)$$

σ - sigmoid activation function.

ii. Input Gate (it):

What additional information should be added to the cell state is determined by the input gate.

$$it = \sigma(Wi * [ht - 1, Xt] + bi)$$

σ - sigmoid activation function.

iii. Candidate Cell (Ct):

The candidate cell state is calculated based on the input data and it gate.

$$Ct = tanh(Wc * [ht - 1, Xt] + bc)$$

where tanh is the hyperbolic tangent activation function.

iv. Cell State (Ct):

The cell state from the previous time step (Ct-1) is integrated with the input gate (it), forget gate (ft) and the candidate cell state (Ct) outputs to change the present cell state.

$$Ct = ft * Ct - 1 + it * Ct$$

where * represents element-wise multiplication.

v. Output Gate (ot):

The concealed state is determined by the output gate, which selects what information from the cell state should be output.

$$ot = \sigma(Wo * [ht - 1, Xt] + bo)$$

σ - sigmoid activation function.

vi. Hidden State ($\boldsymbol{ht}$):

LSTM layer is the output of Hidden State.

$$ht = ot * tanh(Ct)$$

where * represents element-wise multiplication.

Output: LSTM layer output is a sequence of hidden states. Each hidden state contains information about the data up to that point in the sequence, allowing the model to capture temporal patterns and dependencies.

4. Flatten Layer:

Input: Flatten layer input is the sequence of hidden states from the LSTM layer.

Operation: The Flatten layer reshapes the 3D tensor into a 1D vector. This operation prepares the data for the final fully connected layer.

Output: Flatten layer output is a 1D vector that may be viewed as a feature vector that summarizes the temporal data gathered from the LSTM layer.

5. Dense (Fully Connected) Layer:

Input: The input to Dense layer is 1D vector obtained from the Flatten layer.

Operation: The Dense layer computes the input attributes weighted sum, applies a function for activation (for example, softmax for multi-class classification), and generates the final output.

i. Weighted Sum ($\boldsymbol{z}$):

This is the total of the weighted inputs plus a bias term.

$$z = \sum(i)\,(Wi * xi) + b$$

where b is the bias, Wi are the weights and xi are the inputs.

ii. Activation Function ($\boldsymbol{a}$):

The model is introduced non-linearity by the activation function.

$$a = f(z)$$

where activation function is f (e.g., Relu, sigmoid, softmax, etc.).

Output: The Dense layer produces either a list of class probabilities (for categorization activities) or a continuous value (for regression operations). Each element in the output vector represents a classification or regression prediction.

The Flatten Layer and Dense Layer combine together to the give the output layer. In summary, the Conv1D layer extracts spatial features, the MaxPooling1D layer down samples the features, the LSTM layer models temporal dependencies, the Flatten layer prepares data for the last layer, and Dense layer provides the end predictions. The input data undergoes a series of transformations, capturing spatial and temporal patterns, before producing the model's output.

3. 4. Real-World Applicability

a. Raspberry Pi as a Simulation Platform for IoT:

The choice of Raspberry Pi as a simulation platform stems from its characteristics as a cost-effective, low-power, and versatile single-board computer. In the context of IoT, where devices often operate with resource constraints, Raspberry Pi serves as an apt representation. Its emulation of limited computational resources allows us to gauge the adaptability of the intrusion detection model to the conditions commonly encountered by IoT devices.

b. IoT-Specific Validation:

The deployment of the intrusion detection model on Raspberry Pi goes beyond mere simulation; it aims to validate the model's real-world performance, resource - constrained IoT situations. This real-world applicability assessment is crucial for ensuring the effectiveness of the model in practical IoT environments. By testing the model on Raspberry Pi, we can observe its behavior and efficiency in detecting network intrusions under conditions representative of typical IoT deployments.

3. 5. Alert Mechanisms

a. Integration of Twilio for Real-Time SMS Alerts:

In enhancing the real-world responsiveness of our intrusion detection system, we have integrated Twilio, a cloud communications platform, to facilitate real-time SMS alerts. This addition ensures that administrators receive immediate notifications on their mobile devices, allowing for swift responses to potential security threats. Twilio's reliability and speed in delivering SMS alerts contribute to the overall effectiveness of our alert mechanism.

b. Use of SMTP for Email Notifications:

Complementing the SMS alerts, our system is equipped with an email notification mechanism using the Simple Mail Transfer Protocol (SMTP). This dual-alert system ensures comprehensive coverage, as administrators are notified through both SMS and email channels. Email notifications provide a detailed account of the detected intrusion, enabling administrators to assess the situation more thoroughly and take informed actions.

c. Significance of Alert Mechanisms:

The integration of robust alert mechanisms is paramount in fortifying the intrusion detection system's real-world effectiveness. Timely alerts empower administrators to respond promptly to potential security threats, mitigating risks and minimizing the impact of intrusions. The combination of SMS and email notifications offers a multi-faceted approach, ensuring that relevant parties are informed through channels that best suit their accessibility and response requirements.

IV. EVALUATION METRICS AND RESULTS

We give a complete evaluation of the CNN-LSTM Hybrid model's performance in intrusion detection alongside CNN and LSTM in this section. The models were rigorously tested and validated, with the findings described in Tables 2,3,4.

Table 2. The CNN model produced the following results

Metrics	Binary Classification	Multi-Class Classification
Training Accuracy	90.34%	90.34%
Validation Accuracy	90.04%	90.07%
Test Accuracy	90.06%	90.06%
Training Loss	12.72%	12.78%
Validation Loss	12.73%.	12.79%
Test Loss	8.31%	8.35%

Table 3. The LSTM model produced the following results

Metrics	Binary Classification	Multi-Class Classification
Training Accuracy	91.21%	91.01%
Validation Accuracy	91.50%	91.20%
Test Accuracy	91.15%	91.13%
Training Loss	11.63%	11.63%
Validation Loss	11.72%	11.72%
Test Loss	6.41%	6.41%

Table 4. The CNN-LSTM Model produced the following results

Metrics	Binary Classification	Multi-Class Classification
Training Accuracy	98.4%	98.23%
Validation Accuracy	98.14%	98.45%
Test Accuracy	98.23%	98.09%
Training Loss	3.24%	3.65%
Validation Loss	3.5%	3.23%
Test Loss	3.67%	3.37%

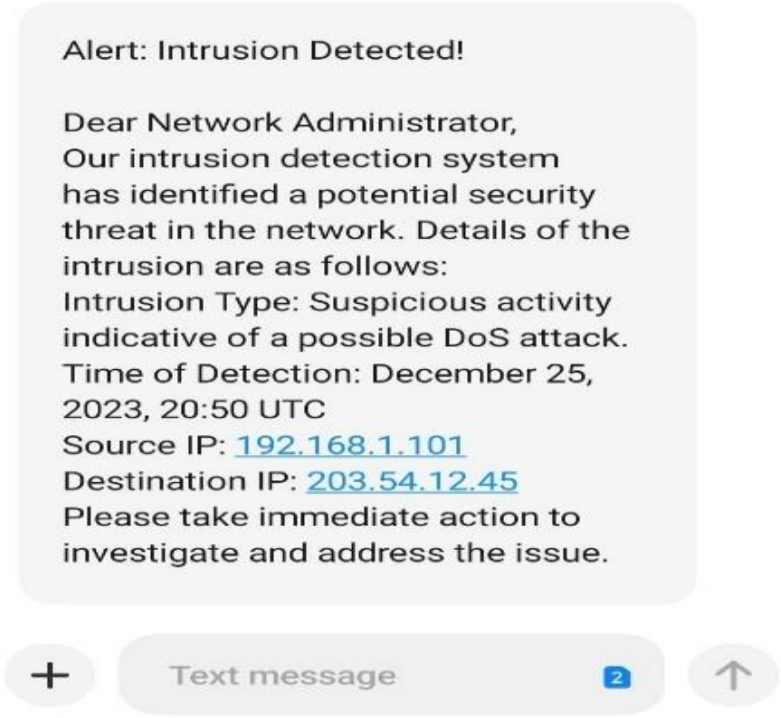
The CNN-LSTM hybrid model outperforms both the LSTM and CNN models, marking a significant advancement in intrusion detection capabilities. The comprehensive evaluation of performance metrics unequivocally underscores the effectiveness of the CNN-LSTM Hybrid model in accurately identifying and classifying network intrusions. Exceptionally high training, validation, and test accuracies, each surpassing 98%, underscore capacity of the model to discern complex structures and anomalies within the network data. Furthermore, the consistently low training and validation losses signify the robust learning and generalization abilities of the CNN-LSTM Hybrid model. This characteristic is particularly crucial in intrusion detection scenarios where the model's capacity to adapt to diverse and evolving threats is paramount. The hybrid architecture, leveraging both convolutional and sequential learning, enables the model to get temporal and spatial dependencies concurrently, contributing to its superior performance. In conclusion, the CNN-LSTM Hybrid model not only achieves exceptional accuracy in both multi-class and binary classifications but also exhibits resilience and adaptability across various datasets. These findings position the CNN-LSTM Hybrid model as a formidable solution for bolstering IoT network security, offering a nuanced and effective approach to intrusion detection in dynamic and complex environments.

Intrusion Detection Alerts:

To validate the real-world applicability of our intrusion detection system, we implemented alert mechanisms using Python modules. When an intrusion is detected, alerts are promptly sent to the relevant personnel through both SMS and email channels.

SMS Alert:

The SMS alert provides a concise notification about the intrusion, including the type of intrusion, time of detection, source, and destination IP addresses.

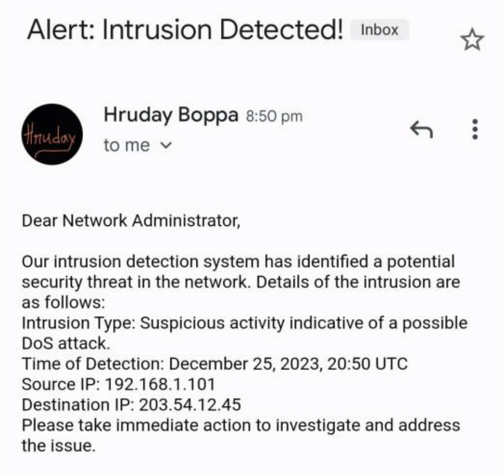


Alert: Intrusion Detected!

Dear Network Administrator,
Our intrusion detection system has identified a potential security threat in the network. Details of the intrusion are as follows:
Intrusion Type: Suspicious activity indicative of a possible DoS attack.
Time of Detection: December 25, 2023, 20:50 UTC
Source IP: 192.168.1.101
Destination IP: 203.54.12.45
Please take immediate action to investigate and address the issue.

Figure 4. SMS Alert for Intrusion Detection

Email Alert: Simultaneously, an email alert is dispatched with detailed information about the intrusion as showing in figure 5. The email includes specifics such as the intrusion type, timestamp, source, and destination IP addresses. This comprehensive alert mechanism ensures that the responsible parties are informed promptly, allowing for immediate investigation and mitigation of potential security threats.



Alert: Intrusion Detected!

Hruday Boppa 8:50 pm
to me

Dear Network Administrator,

Our intrusion detection system has identified a potential security threat in the network. Details of the intrusion are as follows:
Intrusion Type: Suspicious activity indicative of a possible DoS attack.
Time of Detection: December 25, 2023, 20:50 UTC
Source IP: 192.168.1.101
Destination IP: 203.54.12.45
Please take immediate action to investigate and address the issue.

Figure 5. Email Alert for Intrusion Detection

These alert mechanisms, integrated into the CNN-LSTM Hybrid model, not only demonstrate its effectiveness in intrusion detection but also emphasize its practical utility in real-world scenarios, contributing to the enhancement of IoT network security.

V. CONCLUSION

In conclusion, the proposed CNN-LSTM Hybrid model emerges as a formidable intrusion detection system, demonstrating exceptional accuracy in detecting malicious activities within IoT environments. Trained on the UNSW-NB15 dataset, the model exhibited classification rates exceeding 98% in both multi-class and binary classifications. The real-world applicability of the model was validated through testing on Raspberry Pi, showcasing its effectiveness in practical IoT scenarios. The integration of alert mechanisms, utilizing Python modules Twilio for SMS and SMTP for email notifications, enhances the system's responsiveness to potential threats. As we look ahead, future work involves fortifying the model's robustness through synthetic data generation and integrating advanced deep learning techniques for even greater accuracy. Feature selection methods will be explored to refine the intrusion detection system, ensuring optimal performance across diverse datasets. The ultimate goal is real-world deployment, collaborating with industry partners for widespread adoption, and contributing to a safer and more secure IoT landscape. The journey continues towards enhancing the sophistication and intrusion detection systems reliability in the ever-evolving landscape of cybersecurity.

REFERENCES

1. Moustafa, N., & Slayman, M. S. (2006). Machine learning methods for network intrusion detection: A comparative analysis. Journal of Network and Computer Applications, 30(1), 36-56.
2. Ma, Junfeng, and Sung-Bae Cho. "Anomaly detection for wireless sensor networks using a one-class support vector machine." International Journal of Distributed Sensor Networks 10.4 (2014): 159415.
3. Chandola, Varun, Arindam Banerjee, and Vipin Kumar. "Anomaly detection: A survey." ACM computing surveys (CSUR) 41.3 (2009): 1-58.
4. Kumar, S., & Spafford, E. H. (1994). A pattern matching model for misuse intrusion detection. Proceedings of the 17th National Computer Security Conference, 11-21.
5. Guyon, I., & Elisseeff, A. (2003). Random forests for network intrusion detection. Proceedings of the 3rd IEEE Symposium on Computational Intelligence for Security and Defense Applications, 30-37.
6. Garcia-Teodoro, P., Diaz-Verdejo, J., Maciá-Fernández, G., & Vázquez, E. (2009). Anomaly-based network intrusion detection: Techniques, systems and challenges. Computers & Security, 28(1-2), 18-28.
7. Zhang, L., Gu, G., & Rong, L. (2017). Anomaly detection in network traffic using long short-term memory networks. IEEE Transactions on Network and Service Management, 64(10), 1444-1455.
8. S. Hanif, T. Ilyas, M. Zeeshan, Intrusion detection in iot using artificial neural networks on unswnb15 dataset, IEEE 16th International Conference Smart Cities, Improving Quality of Life Using ICT & IoT AI (HONET-ICT) (2019) 152 156.
9. Li, M., Li, J., Wang, C., Wang, Y., & Guo, S. (2023). IoT Intrusion Detection Taxonomy, Reference Architecture, and Analyses. Sensors, 23(19), 11055.

10. Khan, M. A., Cheema, M. A., Bashir, M. K., & Hussain, S. (2022). Dependable Intrusion Detection System for IoT: A Deep Transfer Learning-based Approach. arXiv preprint arXiv:2204.04837.
11. N. Moustafa and J. Slay, "UNSW-NB15: A comprehensive data set for network intrusion detection systems (UNSW-NB15 network data set).," In Proc. IEEE Military Commun. Inf. Syst. Conf. (MilCIS), pp. 1-6, 2015.