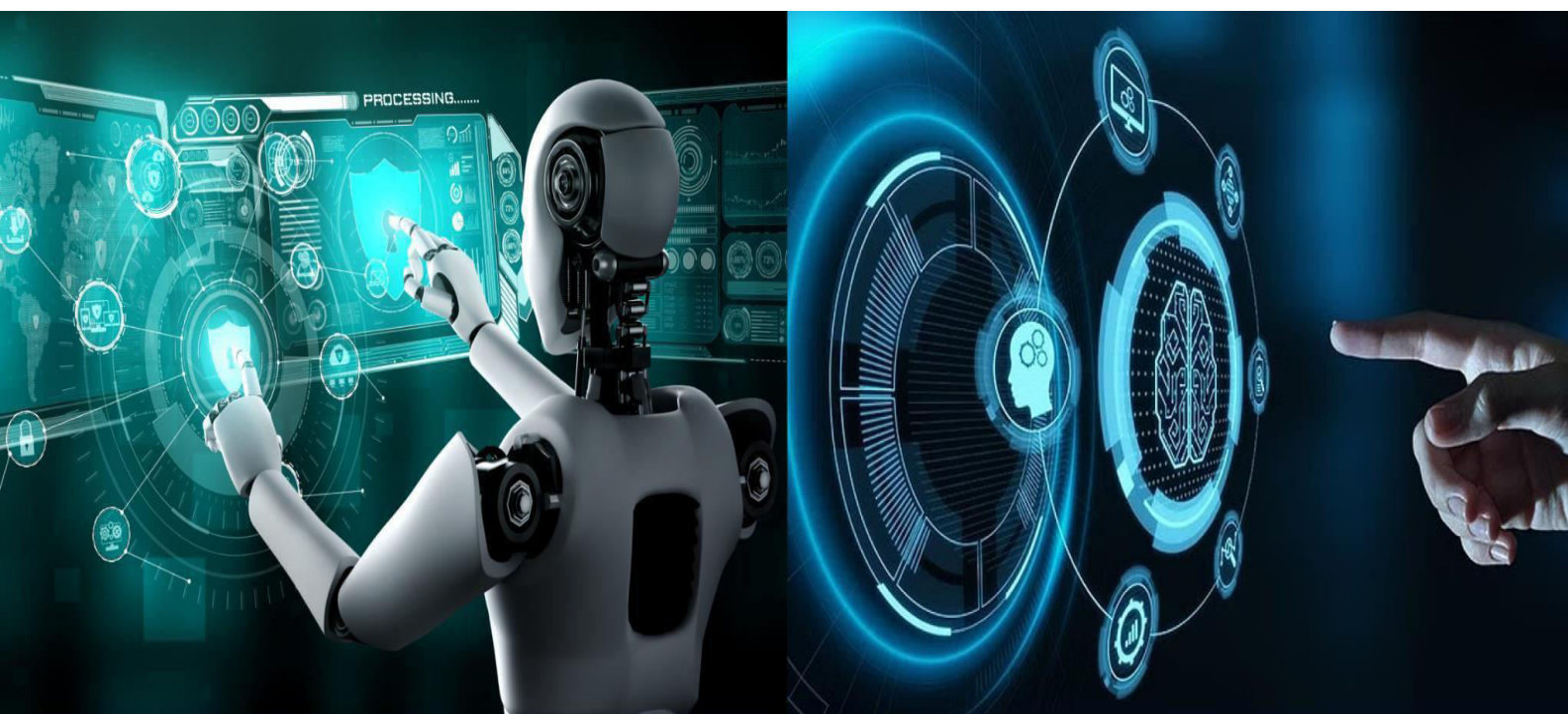


# International Journal of Innovative Research in Computer and Communication Engineering

(A Monthly, Peer Reviewed, Refereed, Scholarly Indexed, Open Access Journal)





# AI Based Cybersecurity Threat Detection Device

Sneha S N<sup>1</sup>, Sneha Yeri<sup>1</sup>, Trisha H R<sup>1</sup>, Vaishnavi K Y<sup>1</sup>, Dr Malatesh SH<sup>2</sup>

Student, Dept. of Computer Science and Engineering, MS Engineering College, Bengaluru, Karnataka, India<sup>1</sup>

HOD, Dept. of Computer Science and Engineering, MS Engineering College, Bengaluru, Karnataka, India<sup>2</sup>

**ABSTRACT:** In this work, several machine learning classifiers were evaluated for cybersecurity threat detection, including Decision Tree, Boosted Tree, Logistic Regression Tree, and Support Vector Machine (SVM). Each model was trained and validated in MATLAB using a cyber threat data to identify the most efficient algorithm for real-time deployment. While Boosted Trees and SVM demonstrated strong predictive performance, they required higher computational resources, making them less suitable for embedded environments. Logistic Regression Trees offered reliable binary classification but showed reduced accuracy for complex threat patterns. Considering portability, execution speed, interpretability, and accuracy requirements, the Decision Tree model was selected for implementation on Raspberry Pi. The model enables real-time threat prediction, and results are displayed on an LCD with an audible buzzer alert for detected threats, making the device a low-cost and hardware-integrated cybersecurity solution.

**KEYWORDS:** Cybersecurity Threats, Machine Learning Models, Decision Tree, Boosted Tree, Logistic Regression, Support Vector Machine (SVM), Raspberry Pi Deployment, MATLAB Application, LCD Display, Buzzer Alert System, Training Dataset, Preprocessing, Normalization, Cross-Validation, Feature Extraction, Real-Time Detection, Model Evaluation Metrics, Hardware Integration, Embedded Systems, Threat Classification.

## I. INTRODUCTION

The rapid growth of digital networks, smart devices, and online services has increased the exposure to cyber threats such as malware, intrusion attacks, phishing, and ransomware. Traditional cybersecurity systems mainly rely on rule-based or signature-based detection approaches, which effectively identify known attacks but struggle with newly emerging and zeroday threats. To address these limitations, artificial intelligence and machine learning are increasingly being integrated into cybersecurity solutions, enabling systems to learn patterns, detect anomalies, and classify unknown threats more accurately. In this project, AI techniques are combined with embedded systems to build a **real-time cybersecurity threat detection device** using a Raspberry Pi. The device uses machine learning models trained in MATLAB, particularly a Decision Tree algorithm, to classify network activities as safe or malicious with high accuracy. The trained model is converted using MATLAB Coder and deployed onto Raspberry Pi for fast edge-level inference without dependency on cloud servers or powerful computers. The **Raspberry Pi** processes incoming input values and performs threat classification locally, making the system portable, efficient, and suitable for real-time monitoring in small networks and IoT environments.

To enhance usability, a custom **MATLAB App** provides a graphical interface for data entry, communication, and visualization. The detection results are displayed on an onboard **LCD screen**, allowing users to monitor threat classification in real time without needing an external system. Additionally, a **buzzer module** generates an audible alert whenever a threat is detected, enabling quick user awareness and timely response. The integration of these components makes the system low-cost, user-friendly, and deployable across diverse environments such as educational institutions, home networks, laboratories, and IoT-based infrastructures. Overall, this project demonstrates how AI-driven cybersecurity models, combined with embedded devices, can offer an affordable, scalable, and intelligent alternative to traditional security tools while ensuring real-time protection and continuous adaptability.

## II. METHODOLOGY

### 1. Dataset Preparation

- Use Intrusion Detection Data.
- Preprocess data: normalization, feature extraction, splitting into training/testing sets.

### 2. Model Training in MATLAB

- Train using the Decision Tree algorithm.



## International Journal of Innovative Research in Computer and Communication Engineering (IJIRCCE)

(A Monthly, Peer Reviewed, Refereed, Scholarly Indexed, Open Access Journal)

- Evaluate model performance using accuracy, precision, recall, F1-score.
- 3. Model Deployment on Raspberry Pi**
  - Convert trained MATLAB model using MATLAB Coder Toolbox.
  - Deploy optimized model onto Raspberry Pi.
- 4. MATLAB Application Development**
  - Create a GUI app to allow manual entry or batch entry of data.
  - Send input values to Raspberry Pi via network communication (Wi-Fi).
- 5. Onboard Processing**
  - Raspberry Pi runs inference using the deployed Decision Tree model, Boosted tree algorithm, Logistic regression algorithm, Support vector machine algorithm.
  - Classifies input as Threat or Benign.

### III. REQUIREMENTS

#### Functional Requirements

- **Real-Time Threat Detection**

The system must continuously monitor incoming network data and classify it as benign or malicious using the deployed machine learning model on Raspberry Pi. This ensures real-time cyber intrusion tracking and immediate response.

- **Automated Alerting and Notification**

When a threat is detected, the system must immediately trigger a buzzer alarm and display alert messages on the onboard LCD, ensuring quick user awareness without manual supervision.

- **MATLAB Application Interaction**

The system must allow users to input data or network parameters through a MATLAB interface, send them to Raspberry Pi for inference, and receive response outputs for visualization and analysis.

- **Local Embedded Inference**

The Raspberry Pi must perform classification locally without needing cloud connectivity, enabling low-latency processing for environments with limited or unstable network access.

- **Result Display and User Feedback**

The system must show classification results (“Threat” or “Safe”) on both MATLAB and the on-device LCD screen. Audible feedback must be provided through a buzzer when malicious activity is detected.

- **Logging and Data Storage**

All prediction outputs, timestamps, and detected events should be stored for later analysis, allowing performance evaluation, debugging, and system monitoring.

#### Non-Functional Requirements

- **Performance and Real-Time Responsiveness**

The system must process and classify input data with minimal latency to ensure immediate cybersecurity alerts and accurate detection results on Raspberry Pi.

- **Scalability**

The architecture should support retraining with new threat datasets, additional model upgrades, and future features like remote monitoring or cloud dashboards without affecting performance.

- **Reliability and Availability**

The device must run continuously with consistent malware detection accuracy, ensuring uninterrupted monitoring and reliable alerting even during long-term operation.

- **Security and Data Integrity**

System communication between MATLAB and Raspberry Pi must be secure, ensuring that transmitted data, stored logs, and detection outputs are protected from unauthorized access or manipulation.

- **Usability and User Experience**

The MATLAB app, LCD interface, and buzzer alerts must be simple, clear, and easy to understand, enabling non-technical users to identify threats instantly without needing cybersecurity expertise.

- **Low Power and Hardware Efficiency**

The device should operate efficiently on Raspberry Pi hardware, minimizing processing overhead and power consumption.



## International Journal of Innovative Research in Computer and Communication Engineering (IJIRCCCE)

(A Monthly, Peer Reviewed, Refereed, Scholarly Indexed, Open Access Journal)

### IV. SYSTEM ARCHITECTURE

System architecture refers to the high-level structure of a complex system, encompassing its components, relationships, interactions, and principles governing its design and operation.

#### 1. Intrusion Detection Data:

A large collection of network traffic records containing normal and malicious activities used as the foundational input for training models.

#### 2. Preprocessing:

Raw dataset is cleaned, normalized, encoded, and transformed to remove noise and prepare high-quality structured data for model training.

#### 3. Model Training:

Machine learning algorithms learn patterns from preprocessed data to detect intrusions by creating a predictive classification or detection model.

#### 4. Model Evaluation:

The trained model is tested on validation data to measure accuracy, precision, recall, and overall reliability before deployment.

#### 5. Model Conversion:

The evaluated model is converted into a lightweight, hardware-compatible format suitable for execution on Raspberry Pi devices.

#### 6. Raspberry Pi Deployment:

Converted model is deployed onto the Raspberry Pi, enabling real-time intrusion detection on a low-cost embedded processing platform.

#### 7. Input Receiver:

Raspberry Pi collects real-time network or sensor inputs, forwarding them to the inference engine for intrusion detection processing.

#### 8. Network Communication:

Raspberry Pi communicates detection results or alerts across the network, enabling remote monitoring and interaction with other components.

#### 9. Buzzer Alert System:

When an intrusion is detected, the Raspberry Pi triggers a buzzer to provide immediate audible warning of suspicious network activity.

#### 10. End-to-End Detection Workflow:

All components work together seamlessly to collect data, analyze threats, and generate alerts using efficient machine learning-based intrusion detection.

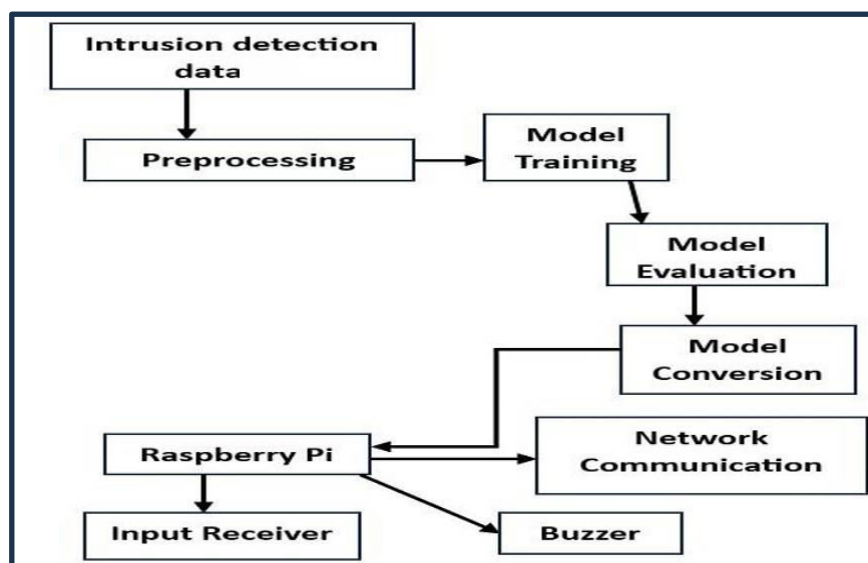


Fig 1: System architecture of AI Based Cybersecurity Threat Detection Device



## International Journal of Innovative Research in Computer and Communication Engineering (IJIRCCCE)

(A Monthly, Peer Reviewed, Refereed, Scholarly Indexed, Open Access Journal)

### V. WORK FLOW

- **Start**

The system starts when the AI-based cybersecurity threat detection device is powered ON and initialized for operation.

- **System Initialization**

All required hardware components such as **Raspberry Pi**, **LCD display**, and **buzzer** are initialized, and network connectivity is established.

- **Input Network Data**

Network or intrusion-related data is provided to the system either through the **MATLAB application** or collected via the Raspberry Pi's network interface.

- **Data Forwarding to Raspberry Pi**

The input data is securely transmitted to the Raspberry Pi, which acts as the embedded processing unit for real-time analysis.

- **Decision Tree Model Execution**

The Raspberry Pi runs the **Decision Tree machine learning model** that was trained in MATLAB and deployed using MATLAB Coder.

- **Feature Analysis and Inference**

The Decision Tree model analyzes extracted features from the input data and performs inference to classify the network activity.

- **Threat Decision Block**

The system checks whether the classified result indicates a **cyber threat or benign activity**.

- **No Threat Condition (Safe)**

If no threat is detected, the classification result is displayed on the **LCD display**, indicating normal or safe network activity.

- **Threat Detected Condition**

If a threat is detected, the system immediately activates the **buzzer alert**, providing an audible warning to notify the user.

- **End of Detection Cycle**

The system completes one detection cycle and returns to monitoring mode, ready to process new network input data continuously.

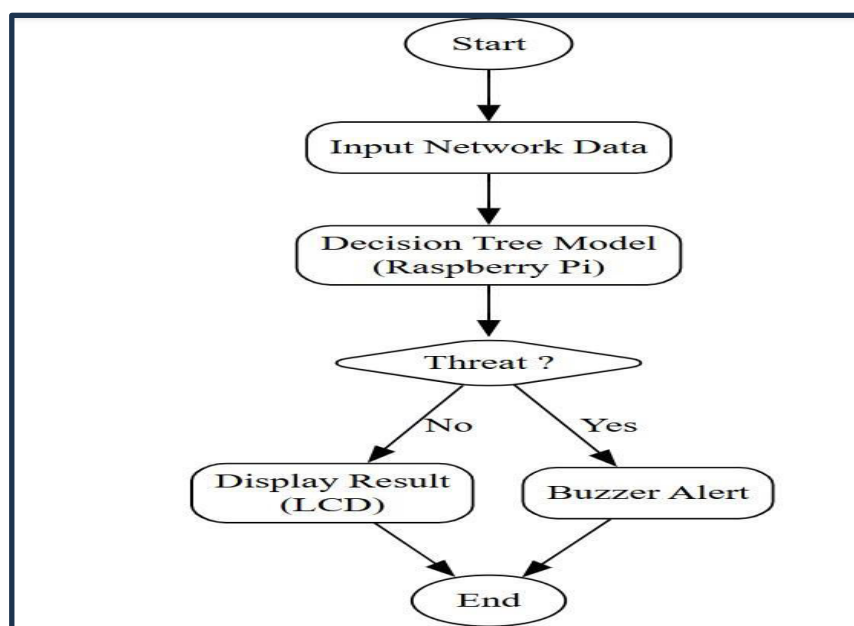


Fig 2:Flowchart of AI Based Cybersecurity Threat Detection Device



## International Journal of Innovative Research in Computer and Communication Engineering (IJIRCCCE)

(A Monthly, Peer Reviewed, Refereed, Scholarly Indexed, Open Access Journal)

### VI. FUTURE SCOPE

The Future Scope of this project includes:

- ❖ Add cloud monitoring to check threats online.
- ❖ Make the device smarter to detect more threats.
- ❖ Connect it to smart devices to keep them safe.
- ❖ Create an app to get alerts on your phone.
- ❖ Make it work with more networks and systems.
- ❖ System is suitable for small to medium-scale applications where real-time monitoring is necessary.
- ❖ It can be extended to enterprise networks by integrating with IoT devices.
- ❖ The model can be retrained with additional data for continuous improvement.
- ❖ Future scalability may include cloud integration and remote monitoring dashboards.

### VII. CONCLUSION

This project successfully demonstrates the design and implementation of an **AI-Based Cybersecurity Threat Detection Device** by integrating machine learning techniques with embedded systems. The primary objective of developing a low-cost, portable, and real-time cybersecurity solution has been achieved using a **Decision Tree algorithm trained in MATLAB** and deployed on a **Raspberry Pi**. Unlike traditional signature-based or rule-based security systems, the proposed system is capable of intelligently classifying network activities as benign or malicious, thereby improving detection accuracy and adaptability to evolving cyber threats.

The use of MATLAB for data preprocessing, model training, evaluation, and deployment ensures reliable performance and ease of development. By converting the trained model using MATLAB Coder and executing it locally on the Raspberry Pi, the system eliminates dependency on cloud infrastructure, resulting in low latency and efficient real-time operation. The integration of an **LCD display** provides clear visual feedback of detection results, while the **buzzer alert system** enables immediate audible notification when a threat is detected, enhancing practical usability and user awareness.

The system architecture proves to be suitable for small networks, IoT environments, laboratories, and educational institutions where affordable and embedded cybersecurity solutions are required. Performance evaluation shows that the selected Decision Tree model offers a good balance between accuracy, interpretability, and computational efficiency, making it well-suited for embedded deployment.

Overall, this project highlights the effectiveness of combining artificial intelligence with embedded hardware to create an intelligent, scalable, and user-friendly cybersecurity device. The solution not only addresses the limitations of existing systems but also provides a strong foundation for future enhancements such as model retraining, cloud integration, and expanded threat classification capabilities.

### REFERENCES

- [1] "A Machine Learning Approach to Cybersecurity Threat Detection" by J. Kim et al. (IEEE, 2020).
- [2] "Raspberry Pi-Based Network Intrusion Detection System" by M. A. Al-Amin et al. (IJCSIT, 2019).
- [3] "IP Address Classification Using Machine Learning" by S. Kumar et al. (IJCNIS, 2020).
- [4] "Anomaly-Based Intrusion Detection System Using Machine Learning" by A. L. Buczak et al. (IEEE, 2016).
- [5] "Cybersecurity Threat Detection Using Deep Learning" by Y. Xin et al. (IEEE, 2018).
- [6] "A Survey on Cybersecurity Threat Detection Using Machine Learning" by R. Vinayakumar et al. (IEEE, 2019).
- [7] "Real-Time Network Intrusion Detection System Using Raspberry Pi" by S. S. Patil et al. (IJERT, 2020).
- [8] "IP Address-Based Threat Detection Using Machine Learning" by A. K. Jain et al. (IJCNIS, 2020).
- [9] "Cybersecurity Threat Detection Using Convolutional Neural Networks" by Y. Liu et al. (IEEE, 2020).
- [10] "Anomaly-Based Cybersecurity Threat Detection Using Machine Learning" by M. A. Ferrag et al. (IEEE, 2020).



INTERNATIONAL  
STANDARD  
SERIAL  
NUMBER  
INDIA



# INTERNATIONAL JOURNAL OF INNOVATIVE RESEARCH

IN COMPUTER & COMMUNICATION ENGINEERING



9940 572 462



6381 907 438



ijircce@gmail.com



[www.ijircce.com](http://www.ijircce.com)

Scan to save the contact details