KLE Technological University | Creating Value, Leveraging Knowledge

SCHOOL OF COMPUTER SCIENCE AND ENGINEERING

Minor Project Report

On

Cyber Threat Detection and Prevention using AI-ML Model (Hybrid Approach)

submitted in partial fulfillment of the requirements for the award of the degree of

Bachelor of Engineering

in

COMPUTER SCIENCE AND ENGINEERING

Submitted By

Mangalgouri P Kademani	**01FE23BCS422**
Yuvaraj P Rathod	**01FE23BCS423**
Nisha B Kubasad	**01FE23BCI401**
Ramaraddi G Maraddi	**01FE23BCS416**

Under the guidance of

Dr.Ashok Chikaraddi

School of Computer Science and Engineering

KLE Technological University, Hubballi

2024-2025

KLE Technological University | Creating Value, Leveraging Knowledge

SCHOOL OF COMPUTER SCIENCE AND ENGINEERING

2024-25

CERTIFICATE

This is to certify that project entitled "Cyber Threat Detection and Prevention using AI-ML Model (Hybrid Approach)" is a bonafied work carried out by the student team Mangalgouri P Kademani - 01FE23BCS422, Yuvaraj P Rathod - 01FE23BCS423, Nisha B Kubasad - 01FE23BCI401, Ramaraddi G Maraddi - 01FE23BCS416, in partial fulfillment of the completion of the 6th semester B.E. course during the year 2024 – 2025. The project report has been approved as it satisfies the academic requirement concerning the project work prescribed for the above-mentioned course.

Guide
Dr.Ashok Chikaraddi

Head, SoCSE
Dr. Vijayalakshmi.M.

External Viva-Voce

Name of the examiners Signature with date

1 _____ _____

2 _____ _____

ACKNOWLEDGEMENT

We would like to thank our faculty and management for their professional guidance towards the completion of the minor project work. We take this opportunity to thank Dr. Ashok Shettar, Pro-Chancellor, Dr. P.G Tewari, Vice-Chancellor and Dr. B.S.Anami, Registrar for their vision and support. We also take this opportunity to thank Dr. Meena S. M, Professor and Dean of Faculty, SoCSE and Dr.Vijayalakshmi M, Professor and Head, SoCSE for having provided us direction and facilitated for enhancement of skills and academic growth. We thank our guide Dr.Ashok Chikraddi and SoCSE for the constant guidance during interaction and reviews. We extend our acknowledgment to the reviewers for critical suggestions and inputs. We also thank Project coordinator Dr.Uday Kulkarni, and reviewers for their suggestions during the course of completion. We express gratitude to our beloved parents for constant encouragement and support.

Mangalgouri P Kademani(01FE23BCS422)
Yuvaraj P Rathod(01FE23BCS423)
Nisha B Kubasad(01FE23BCI401)
Ramaraddi G Maraddi(01FE23BCS416)

ABSTRACT

In today's digital landscape, cyber threats pose a significant risk to networks, systems, and sensitive information. This report presents a hybrid Artificial Intelligence (AI) and Machine Learning (ML) based approach for cyber threat detection and prevention that integrates both supervised and unsupervised learning techniques. The system utilizes the CICIDS2017 dataset, which includes a wide range of contemporary attack types, and employs an ensemble of Isolation Forest and Autoencoder models for anomaly detection. XGBoost is used for multi-class classification of identified threats, while a CNN-LSTM model captures temporal behavior for proactive threat mitigation. The proposed system is deployed with a Flask-based real-time monitoring dashboard that enables network administrators to visualize, analyze, and respond to cyber threats effectively. Evaluation metrics such as accuracy, precision, recall, and F1-score demonstrate that the hybrid model significantly improves threat detection performance, achieving up to 99% accuracy with CNN-LSTM and perfect classification with XGBoost. The integration of real-time visualization and automated mitigation further enhances the system's practical utility in dynamic network environments.

Keywords : *Cybersecurity, Intrusion Detection, Machine Learning, Deep Learning, Hybrid Model, Isolation Forest, Autoencoder, XGBoost, CNN-LSTM, Anomaly Detection, Threat Prevention, CICIDS2017, Real-Time Monitoring, Ensemble Learning, Flask Dashboard*

CONTENTS

LIST OF TABLES

LIST OF FIGURES

Chapter 1

INTRODUCTION

Cyber threat detection and prevention have become critical in today's highly interconnected world, where modern attacks target sensitive systems with increasing frequency and sophistication. Traditional security tools, such as firewalls and signature-based intrusion detection systems, are often reactive and fail to detect unknown or evolving threats[2]. To overcome these limitations, recent advancements in artificial intelligence and machine learning have introduced intelligent, adaptive models capable of identifying both known and novel attacks[7]. By integrating anomaly detection with classification techniques, hybrid models leverage the strengths of both unsupervised and supervised learning[18]. The present work builds upon these developments by proposing a hybrid AI-ML architecture that combines Isolation Forest, Autoencoder, XGBoost, and CNN-LSTM models to detect, classify, and respond to cyber threats in real-time, enhancing overall network security and resilience.

1.1 Preamble

The rapid digitalization of industries, businesses, and everyday life has led to an exponential increase in cyber threats, including DDoS attacks, data breaches, and zero-day exploits[4]. Conventional cybersecurity systems often rely on signature-based detection[6], which cannot cope with sophisticated and evolving attack vectors. As a result, there is an increasing demand for intelligent and adaptive systems capable of analyzing massive network traffic data and identifying potential threats proactively[15]. This project focuses on leveraging artificial intelligence (AI)[20] and machine learning (ML)[1] to create a hybrid model that improves cyber threat detection and prevention capabilities in real-time environments.

1.2 Motivation

In recent years, the frequency and complexity of cyber-attacks have escalated, making traditional defense mechanisms insufficient. Static rule-based systems struggle to identify unknown threats, leading to data loss, privacy breaches, and financial damage. The need for a smarter, automated system capable of identifying anomalies and classifying threats accurately has become critical. Motivated by this challenge, this project aims to design a hybrid AI-ML-based

approach that combines the strengths of unsupervised and supervised learning[5] for robust cyber threat detection and response.

1.3 Objectives

1. To collect the dataset from the different data sources to identify various cyber threats such as DDoS, SQL Injection, web attacks etc, for attack detection mechanism.
2. To preprocess, clean and normalize dataset for model training and train the models using AI-ML algorithms and do performance evaluation.
3. To develop a system to detect, classify and mitigate cyber threats.
4. To test the system in a real time network traffic.

1.4 Literature Review

This author Autor X. Meng evaluates AI methods including Isolation Forest, Autoencoder, CNN, and LSTM on network security datasets, proposing a hybrid approach for enhanced anomaly detection.[12] The combination of Isolation Forest and Autoencoder significantly improves zero-day threat detection with better precision-recall balance. However, the study lacks real-time integration, scalability analysis, and a practical deployment framework.

The author M. Nalini et al. propose a hybrid density-based enhancement of Isolation Forest with clustering-based pre-filtering, tested on CICIDS and UNSW-NB15 datasets. The approach improves recall and reduces false positives, making it effective for early-stage anomaly detection.[13] However, it lacks multi-class classification, temporal pattern analysis, and integration with deep learning models.

This author B. Jeon evaluates CNN, LSTM, and a combined CNN-LSTM model on the CICIDS2017 dataset to enhance intrusion detection. The CNN-LSTM architecture captures spatiotemporal patterns effectively, achieving 99% accuracy and strong generalization across attack types.[9] However, the study lacks integration with classical models, real-time processing, and has high computational requirements.

K. D. O. Ofoegbu et al. author present a real-time threat detection system using ensemble ML models and big data analytics on large network datasets. [14]The approach achieves over 90% accuracy and effectively classifies DDoS and brute-force attacks with improved scalability

and responsiveness. However, it lacks deep learning for temporal analysis and offers only a basic visualization dashboard

The author S. S. Dhaliwal et al. apply XGBoost with class-weighting on CICIDS2017 and NSL-KDD datasets to improve intrusion detection. [6]The model achieves 98–100% accuracy for attacks like DDoS, PortScan, and Brute Force, outperforming traditional methods. However, it lacks anomaly detection, ensemble or deep learning integration, and adaptability to evolving threats without retraining.

The author Bhoopesh Singh Bhati et al. propose an ensemble-based IDS using XGBoost, trained on the KDDCup99 dataset to detect both known and unknown attacks.[3] The model achieved 99.95% accuracy, showcasing high effectiveness in intrusion detection. However, reliance on the outdated KDDCup99 dataset limits generalizability to modern network threats, necessitating validation on contemporary datasets.

Kumar Saurabh et al. the author present LBDMIDS, an LSTM-based IDS designed for IoT networks, using stacked and bidirectional LSTMs trained on UNSW-NB15 and BoT-IoT datasets[16]. The models outperform traditional ML techniques in identifying complex attack patterns. However, real-time performance and scalability in large-scale IoT deployments remain untested.

This work introduces a hybrid deep learning IDS with an inverted hour-glass network and optimal feature selection, evaluated on NSL-KDD, KDD-CUP99, and UNSW-NB15 datasets. It achieved high accuracy rates (up to 99.967%), surpassing traditional models in detecting diverse attacks[10]. However, its real-time applicability and adaptability to evolving threats in dynamic networks remain to be explored.

This work proposes a hybrid deep learning IDS combining Autoencoder, LSTM, and CNN architectures, trained on the CIC IoT-2023 dataset to detect diverse IoT cyber-attacks[17]. The model demonstrated high accuracy and robustness across various attack types, enhancing IoT security. However, its scalability and real-time performance in large-scale IoT deployments require further investigation.

Xinwei Yuan et al. the author propose a framework enhancing adversarial robustness of deep learning-based IDS by combining DL models with adversarial example detectors and traditional ML classifiers[19]. The approach leverages low attack transferability to improve detection accuracy and reduce false positives. However, its effectiveness across diverse networks

and scalability for real-time detection need further validation.

1.5 Problem Definition

Design and develop a system for cyber threat detection and prevention using a hybrid AI-ML approach combining Isolation Forest, Autoencoder, XGBoost, and CNN-LSTM to detect anomalies, classify, and prevent cyber threats, like DDoS, SQL injection, and Web attacks that are rapidly evolving in networks.

Chapter 2

SOFTWARE REQUIREMENT SPECIFICATION

The major goal of the SRS paper is to offer readers with a full overview of our model, including all of its qualities and goals. This document describes the software requirements for the project.

2.1 Overview of SRS

The Software Requirements Specification (SRS) for this project outlines the essential hardware and software requirements, functional and nonfunctional needs, and use case scenarios for cyber threat detection and prevention using a hybrid AI-ML approach[8]. It ensures that the system meets performance, usability, and reliability standards, enabling accurate detection, classification, and real-time response to cybersecurity threats.

2.2 Requirement Specifications

Requirement specification tells how a software should work and what is expected fromthe software, these requirements should be relevant and detailed, there are functional and non-functional requirements[9]. it should also contain a description about the verification and working of the project.

2.2.1 Functional Requirements

Functional requirements explain the service that the software/project offers, about the input and output to the software, and about its behavior. Functional requirements specify what the system must accomplish.

1. **LaunchCyberAttack()**

   Input: AttackType, TargetIP.
   Processing: Simulate selected attack on target system.
   Output: Attack event logged, system behavior altered.

2. **DetectIntrusion()**
   Input: LiveNetworkTraffic, DetectionModel.
   Processing: Analyze packets for anomalies using ML techniques.
   Output: Intrusion alert generated and logged.

3. **NotifyAdmin()**
   Input: AlertDetails, AdminContact.
   Processing: Format and send a notification to the admin.
   Output: Admin notified via dashboard/email; event logged.

4. **AnalyzeAnomaly()**
   Input: SuspiciousTrafficData.
   Processing: Use ML model to evaluate anomaly behavior.
   Output: Threat likelihood score; data flagged for classification.

5. **ClassifyThreat()**
   Input: AnalyzedData, ClassificationModel.
   Processing: Classify threat type and assign severity.
   Output: Threat record with type, severity, and timestamp.

6. **MitigateAttack()**
   Input: ThreatID, MitigationRules.
   Processing: Apply countermeasure (e.g., block IP).
   Output: Attack neutralized or minimized; mitigation log updated.

7. **MonitorLogsAndAlerts()**
   Input: AdminLogin.
   Processing: Display logs and alerts from the system database.
   Output: Admin views system activity and threat history.

8. **ViewClassifiedThreats()**
   Input: AdminRequest.
   Processing: Fetch and display classified threats.
   Output: Admin sees categorized threat data with details.

9. **TriggerManualMitigation()**
   Input: ThreatID, ActionType, AdminID.
   Processing: Validate admin credentials, execute chosen action.
   Output: Threat manually mitigated; action logged.

10. **ConfigureSecurity()**
    Input: AdminSettingsUpdate.
    Processing: Apply new detection thresholds or response policies.
    Output: System updated; configuration change logged.

2.2.2 Use case diagram

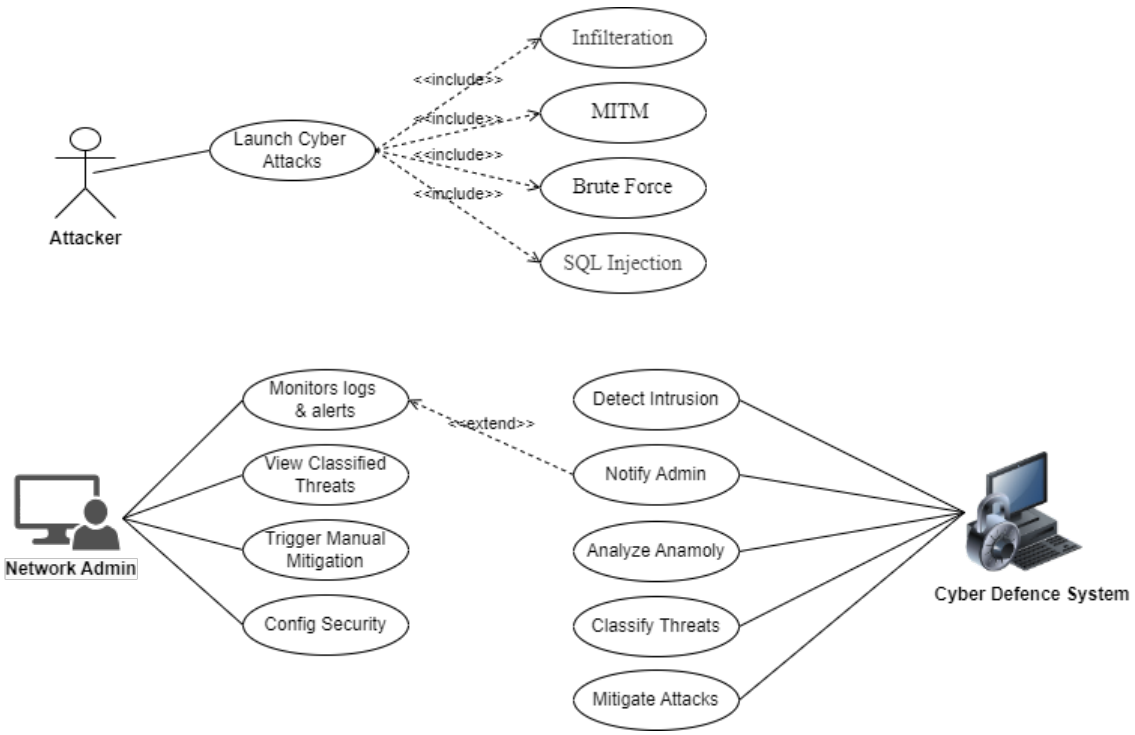

Figure 2.1: Use case diagram

2.2.3 Use Case descriptions

Use Case 1: Launch Cyber Attacks
Actors: Attacker
Description: The attacker initiates a cyber attack to compromise the system.
Prerequisites:

- Attacker has access to the target network.
- Attack tools are available.

Events in Progress:

1. The attacker selects an attack type (e.g., SQL Injection, MITM).

2. The attack is launched towards the network/system.
3. The system starts experiencing abnormal behavior.

Postconditions:

- System may detect and log the attack attempt.
- Attack progresses unless mitigated.

Use Case 2: Detect Intrusion

Actors: Cyber Defence System
Description: The system continuously monitors traffic and detects any unauthorized or abnormal behavior.
Prerequisites:

- Network traffic is being monitored.
- Detection models are deployed.

Events in Progress:

1. The system scans incoming data packets.
2. Anomalous behavior is flagged based on learned patterns.
3. Detection engine raises an alert internally.

Postconditions:

- An intrusion alert is generated.
- Alert is stored in logs for admin visibility.

Use Case 3: Notify Admin

Actors: Cyber Defence System
Description: When a threat is detected, the system alerts the network administrator.
Prerequisites:

- Anomaly or threat is detected.
- Admin alert settings are configured.

Events in Progress:

1. Alert is triggered by the system.
2. The system sends a notification to the admin (e.g., email, dashboard).

Postconditions:

- Admin is aware of the detected threat.
- Admin can take further action.

Use Case 4: Analyze Anomaly

Actors: Cyber Defence System
Description: The system analyzes flagged data to determine if it represents a threat.
Prerequisites:

- Suspicious activity is detected.

Events in Progress:

1. System gathers data from suspicious traffic.
2. Pre-trained model analyzes behavioral patterns.
3. Result is prepared for threat classification.

Postconditions:

- The anomaly is confirmed or dismissed.
- Data is ready for classification.

Use Case 5: Classify Threats

Actors: Cyber Defence System
Description: The system identifies the type and severity of threat based on the anomaly.
Prerequisites:

- An anomaly has been analyzed.

Events in Progress:

1. Classification model processes the input.
2. Attack type is determined (e.g., brute force, mimt).

Postconditions:

- Threat is labeled with its category and severity level.

Use Case 6: Mitigate Attacks

Actors: Cyber Defence System

Description: The system takes countermeasures to block or neutralize the attack.

Prerequisites:

- Threat is classified as actionable.

Events in Progress:

1. System determines mitigation strategy.
2. Automated scripts or rules are applied (e.g., IP blocking).

Postconditions:

- Attack is stopped or its impact is minimized.
- Admin is informed of the mitigation action.

Use Case 7: Monitor Logs and Alerts

Actors: Network Admin

Description: The admin reviews logs and alerts generated by the system.

Prerequisites:

- Admin has access to the dashboard.

Events in Progress:

1. Admin logs into the system.
2. Admin views the list of alerts and system logs.

Postconditions:

- Admin has insight into system activity and potential threats.

Use Case 8: View Classified Threats

Actors: Network Admin

Description: Admin checks the list of threats identified and classified by the system.

Prerequisites:

- Threats have been classified.

Events in Progress:

1. Admin accesses the "Classified Threats" tab.
2. Admin views threat details including category and timestamp.

Postconditions:

- Admin understands the current threat status and history.

Use Case 9: Trigger Manual Mitigation

Actors: Network Admin
Description: Admin manually responds to a detected threat when necessary.
Prerequisites:

- Threat has been reviewed and confirmed.

Events in Progress:

1. Admin selects a threat from the dashboard.
2. Admin chooses a manual action (block IP).

Postconditions:

- Selected action is executed by the system.
- Threat is mitigated manually.

Use Case 10: Configure Security

Actors: Network Admin
Description: Admin modifies system settings to adjust security policies and detection thresholds.
Prerequisites:

- Admin has configuration privileges.

Events in Progress:

1. Admin accesses the configuration panel.
2. Admin updates detection parameters or response rules.

3. Settings are saved.

Postconditions:

- System applies the new configuration.
- Detection behavior may be updated accordingly.

2.2.4 Non-Functional Requirements

Constraints that interfere with the operation of the system are known as nonfunctional requirements. If the software is delivered before even if the non functional requirements are not met then the software won't meet the users expectations.

1. **Real-Time processing:** Detect and classify attacks less than 1 min.
2. **Scalability:** Handle network traffic up to 10 Gbps without performance degradation.
3. **Efficiency:** Utilize less than 50% CPU and 40% RAM during peak traffic loads.
4. **Availability:** The system shall ensure a minimum uptime of 99% under normal operating conditions.
5. **Maintainability:** The system shall be modular and well-documented, allowing for ease of updates and integration with future AI/ML models or components.

2.3 Software and Hardware requirement specifications

2.3.1 Software requirements

- Linux or Windows 11
- Python 3.12.6
- TensorFlow and PyTorch for model development
- Scikit-learn for machine learning algorithms
- Pandas and NumPy for data processing

2.3.2 Hardware requirements

- PC with Intel Core i5 or i7 processor
- Minimum 8 GB of RAM
- Storage of at least 100 GB SSD
- Network Interface Card (NIC) supporting packet capture

Chapter 3

PROPOSED SYSTEM

The proposed system outlines a hybrid AI-ML-based approach for detecting and preventing cyber threats in real time. This methodology integrates both supervised and unsupervised learning techniques to enhance threat detection accuracy and responsiveness. The chapter details the system's architecture, data flow, detection pipeline, and model components[9]. By combining techniques such as Isolation Forest, Autoencoder, and LSTM, the system is designed to identify anomalies, classify threats, and generate alerts with minimal latency. Emphasis is placed on adaptability, scalability, and ease of deployment in various cybersecurity environments.

3.1 Description of Proposed System

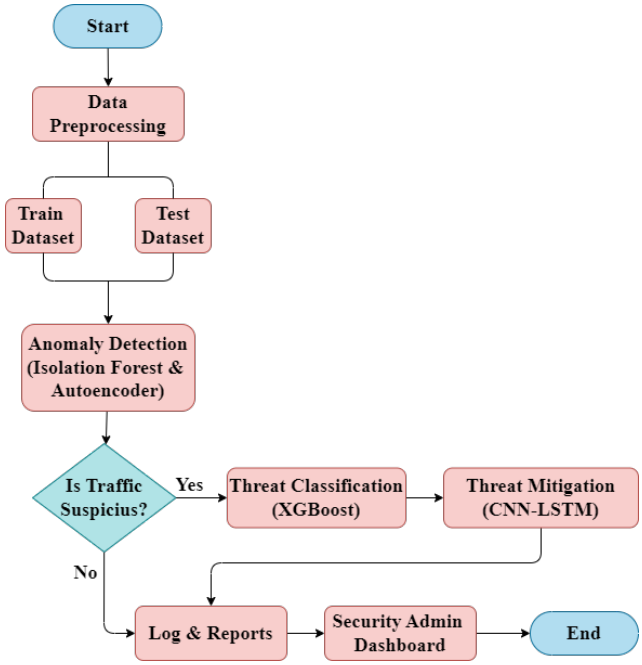

Figure 3.1: Proposed system for Hybrid AI-ML Based Cyber Threat Detection

The proposed system leverages a multi-stage AI-ML pipeline to provide a robust and intelligent solution for cyber threat detection and response. The flowchart in Figure 3.1 illustrates the sequential operations that are carried out by the system. These stages ensure efficient handling

of data from preprocessing to threat detection, classification, and mitigation. Each component in the workflow is designed to optimize detection accuracy while maintaining low latency and adaptability in dynamic cybersecurity environments. The workflow of the system is structured as follows:

- **Start:** The system is initialized and prepared to receive network traffic data for analysis.
- **Data Preprocessing:** Raw network traffic is collected and cleaned. Operations such as noise removal, normalization, and feature encoding are performed to prepare the data for model input.
- **Train and Test Dataset Split:** The preprocessed data is divided into two parts — one for training the models and the other for testing the system's accuracy.
- **Anomaly Detection (Isolation Forest & Autoencoder):** Unsupervised learning techniques are applied to identify abnormal patterns in the network traffic that deviate from normal behavior.
- **Is Traffic Suspicious?:** If the traffic is deemed normal, it is logged and the process ends for that sample. If it is suspicious, the traffic proceeds to the next phase.
- **Threat Classification (XGBoost):** The suspicious traffic is passed to a trained XGBoost classifier, which categorizes it into specific attack types such as DDoS, Port Scan, or Brute Force.
- **Threat Mitigation (CNN-LSTM):** A hybrid CNN-LSTM model is used to analyze the temporal behavior of the attack. This model aids in understanding and preventing further similar threats by learning sequential patterns.
- **Log & Reports:** The results of the detection and classification processes are stored in logs and compiled into system reports for administrative review and future analysis.
- **Security Admin Dashboard:** A real-time dashboard displays system status, ongoing threat statistics, model performance, and allows administrators to monitor and respond effectively.
- **End:** The detection and response cycle for that data batch is completed, and the system continues monitoring for new traffic.

This structured and modular approach allows for continuous monitoring and learning, enabling the system to respond proactively to evolving threats. By integrating unsupervised models like Isolation Forest and Autoencoder for anomaly detection, and supervised models

such as XGBoost and CNN-LSTM for classification and mitigation, the proposed system ensures a hybrid intelligence that adapts to various attack vectors. Furthermore, the use of a real-time dashboard empowers administrators to make informed decisions quickly, thereby enhancing overall network security posture.

3.2 Description of Target Users

The following are the system's target users:

- **Network Security Administrators:** For monitoring and mitigating real-time threats.
- **SOC Teams:** To analyze alerts, logs, and dashboards for faster incident response.
- **Cybersecurity Researchers:** For experimenting with hybrid AI-ML models on threat datasets.
- **IT Managers:** To ensure secure infrastructure and reduce attack risks.
- **Government and Defense Agencies:** For real-time, automated threat detection in critical systems.
- **SMEs and Enterprises:** For affordable and scalable threat monitoring without heavy manpower.

3.3 Advantages of Proposed System

1. **High Accuracy:** Combines Isolation Forest, Autoencoder, and CNN-LSTM models to achieve accurate detection and classification of both known and unknown threats.
2. **Real-Time Detection:** Capable of analyzing network traffic in real-time to detect anomalies and launch timely mitigation.
3. **Hybrid Model Approach:** Leverages both supervised and unsupervised learning techniques for more robust and adaptive detection.
4. **Low False Positives:** Improved anomaly filtering reduces unnecessary alerts, allowing for efficient monitoring.
5. **Modular Design:** Supports integration with dashboards, reports, and logging systems for administrative use.
6. **Scalable Architecture:** Can be adapted to various network sizes and security requirements across different organizations.

3.4 Scope (Boundary of proposed system)

- Adapt models for IoT devices, which are highly vulnerable and resource-constrained.
- Provides a framework for future expansion using threat intelligence feeds and advanced techniques like Explainable AI.
- Designed to be scalable and extendable to new environments such as IoT, cloud, and edge networks.
- Supports continuous learning and model updates using feedback and new threat data.

Chapter 4

SYSTEM DESIGN

This chapter provides an overview of the system design for the proposed hybrid AI-ML-based cyber threat detection and prevention system. It explains the architecture, key modules, and the logical flow of data between components. The design focuses on integrating anomaly detection and classification models to ensure accurate and real-time threat identification. Detailed diagrams, algorithm flow, and component-wise interactions are presented to highlight system functionality, modularity, and scalability. The use of benchmark datasets and the design's adaptability to different environments are also discussed.

4.1 Architecture of the system

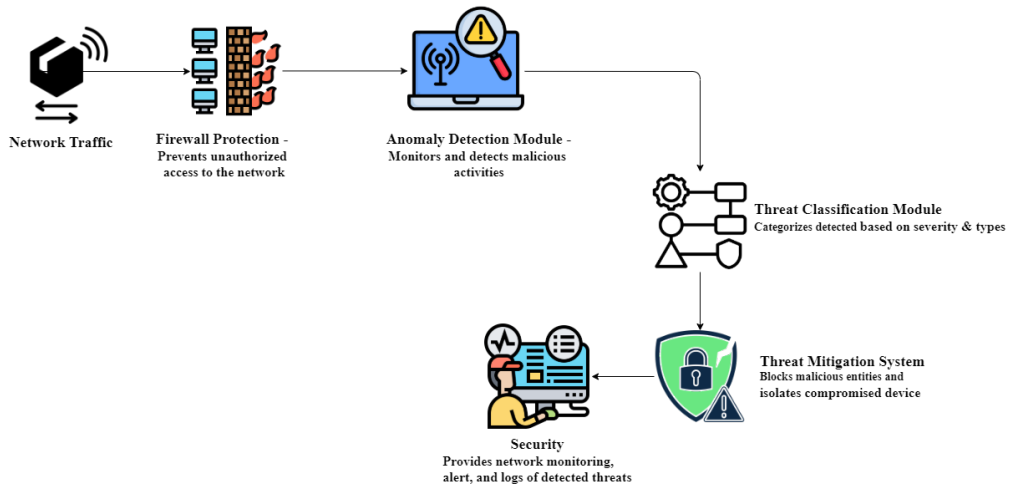

Figure 4.1: Proposed System Architecture for Hybrid AI-ML Based Cyber Threat Detection

The architecture of the proposed hybrid AI-ML cyber threat detection system, as illustrated in Figure 4.1, is designed to ensure accurate, scalable, and real-time identification of network anomalies and malicious activities. The following components form the core of the system:

- **Input Source (Network Logs, Traffic):** This module acts as the system's entry point, collecting raw network data from firewalls, firewalls , and other sources. The data includes connection logs, packet traces, and system activities.

- **Data Preprocessing:** Raw data is cleaned and normalized to remove noise and irrelevant entries. Feature extraction and transformation steps are applied to convert logs into a structured format suitable for machine learning models.
- **Anomaly Detection Module (Isolation Forest, Autoencoder):** This module uses unsupervised models to identify abnormal traffic patterns. Isolation Forest isolates outliers, while Autoencoders detect anomalies based on reconstruction error.
- **Classification Module (XGBoost):** This supervised learning module classifies known threats into categories such as DoS, Sql etc. CNN captures spatial patterns while LSTM detects sequential behaviors in time-series data.
- **Threat Analysis and Decision System:** The outputs of anomaly detection and classification models are analyzed together. A decision layer fuses results to confirm the presence and severity of a threat.
- **Alert and Monitoring Dashboard:** Generates real-time alerts for security analysts. Visual dashboards display detected threat types, timestamps, severity levels, and system health status.

The overall working of the proposed hybrid AI-ML system is illustrated in Algorithm. 1. The process begins with data preprocessing and splitting into training and testing sets. Anomaly detection is carried out using Isolation Forest and Autoencoder models. If traffic is found suspicious, it is classified using XGBoost and further analyzed with a CNN-LSTM model for mitigation. The results are logged and visualized on a real-time security dashboard for prompt action.

Algorithm 1 Network Threat Detection and Classification System

Require: Raw network data from firewalls, firewalls , and other sources

Ensure: Real-time alerts with threat classification and severity

1: Initialize input sources
2: Collect raw network data (connection logs, packet traces, system activity)
3: Preprocess the data: clean noise, normalize, extract and transform features
4: Perform anomaly detection using Isolation Forest and Autoencoder
5: If anomaly detected, classify the traffic using XGBoost model.
6: Fuse anomaly detection and classification results to confirm threat and severity
7: Generate real-time alerts and update the monitoring dashboard
8: **Optional:** Trigger automated security responses or SIEM integration
9: **return** Threat alerts with classified categories and severity =0

4.2 Sequence Diagram

The Figure 4.2 is a sequence diagram that illustrates the interactions between the components of the proposed threat detection system during the processing of suspicious network traffic.

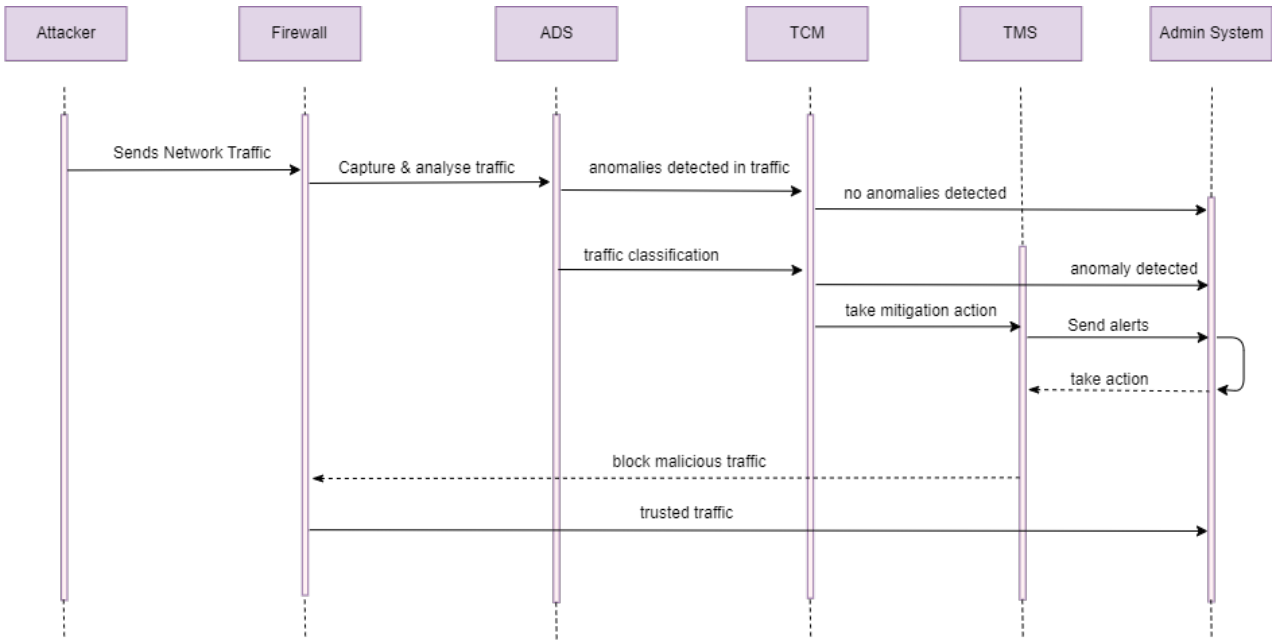

Figure 4.2: Sequence Diagram

- **Sending Malicious Traffic:** The attacker initiates the process by sending suspicious or malicious traffic into the network.
- **Traffic Capture and Analysis:** The firewall captures incoming traffic and forwards it to the ADS for analysis.
- **Anomaly Detection:** The ADS inspects the traffic using models like Autoencoder and Isolation Forest to identify any abnormal patterns.
- **Traffic Classification:** If an anomaly is detected, the TCM classifies the threat type (e.g., DoS, R2L) using a supervised model such as XGBoost.
- **Threat Mitigation:** Upon confirmation, the TMS takes mitigation actions, possibly using a CNN-LSTM model and defense tools like Wireshark or Windows Defender.
- **Alert and Logging:** The TMS sends alerts to the Admin System and logs the event. The administrator may take additional manual actions.
- **Trusted Traffic Handling:** If no anomaly is detected, the traffic is marked as trusted and allowed through the system.

Chapter 5

IMPLEMENTATION

In the implementation chapter, this section describes in detail the methods used to develop the proposed network threat detection and prevention system. It includes data collection, preprocessing, anomaly detection, threat classification, response handling, and visualization[11]. Each stage is implemented using appropriate machine learning and deep learning techniques, along with the necessary tools and frameworks to ensure accuracy, efficiency, and real-time threat response capabilities.

5.1 Proposed Methodology

The proposed approach outlines how to detect and mitigate network threats using a combination of anomaly detection and classification models as shown in Figure 5.1. The process includes capturing network traffic, preprocessing data, detecting anomalies, classifying attack types, and initiating appropriate responses. The methodology is supported by detailed flow and sequence diagrams that explain each step of the system's workflow.

5.2 Description of Modules

1. **Module Name: Input Source Handling**
   Input: Network logs, packet data, system traffic.
   Output: Raw data streams for analysis.
   Description: This module collects incoming network traffic from firewalls , firewalls, or other monitoring tools. It acts as the entry point for the system's analysis pipeline.

2. **Module Name: Rule-Based Filtering**
   Input: Raw network traffic.
   Output: Filtered traffic after blocking known malicious sources.
   Description: Known malicious IP addresses or blacklisted domains are blocked using predefined firewall rules or IPTable configurations.

3. **Module Name: Anomaly Detection**
   Input: Filtered traffic logs.

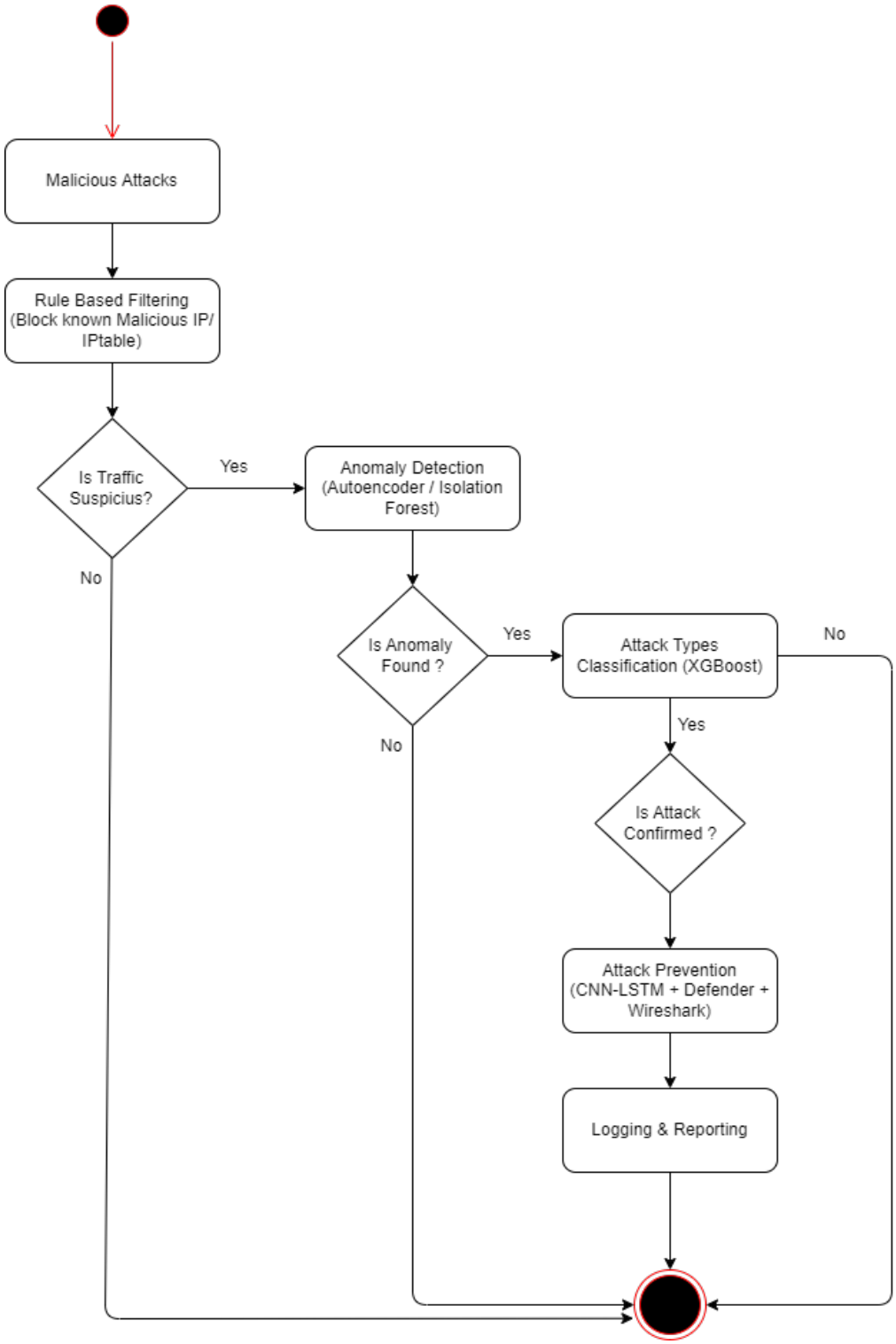

Figure 5.1: Activity Diagram

Output: Flags indicating anomalous behavior.
Description: Machine learning models like Isolation Forest and Autoencoders detect unusual patterns in network traffic, helping to identify novel or zero-day threats.

4. **Module Name: Threat Classification**
   Input: Detected anomalies.
   Output: Specific attack categories (e.g., DoS, SQL).
   Description: Supervised models such as XGBoost or CNN-LSTM are used to classify the nature of the threat based on feature vectors derived from traffic data.

5. **Module Name: Decision System**
   Input: Classification and anomaly results.
   Output: Final threat verification and response strategy.
   Description: This module combines results from detection and classification to confirm threats and determine severity before triggering actions.

6. **Module Name: Threat Mitigation and Response**
   Input: Verified threat alerts.
   Output: Action commands (block IP, alert admin, etc.).
   Description: Uses predefined security tools and automation (e.g., windows Defender) to mitigate active threats and prevent further intrusion.

7. **Module Name: Logging and Monitoring Dashboard**
   Input: System status and threat logs.
   Output: Visual reports and real-time alerts.
   Description: Displays all system activity, detected threats, and current security posture for administrators through an interactive interface.

Below Algorithm 2 present the workflow of the proposed hybrid AI-ML-based cyber threat detection system. It begins by capturing and preprocessing network data, followed by rule-based filtering. Anomalies are detected using unsupervised models, and suspicious traffic is classified with XGBoost. If a threat is confirmed, mitigation actions are triggered, with all results logged and shown on a real-time dashboard.

Algorithm 2 Hybrid AI-ML-Based Cyber Threat Detection and Prevention Flow

Require: Trained anomaly detection and classification models, real-time network traffic data

Ensure: Detected and classified threats with corresponding mitigation actions

Initialize network data sources and load AI-ML models.
`traffic_data` ← capture real-time network packets and system logs
Preprocess `traffic_data`: clean noise, extract relevant features
Apply rule-based filtering to block known malicious IPs and domains
Perform anomaly detection on preprocessed traffic using unsupervised models (e.g., Isolation Forest, Autoencoder)
if anomaly_score > threshold **then**
    Perform classification using XGBoost
    Determine threat category
end if
Correlate anomaly and classification results to assess threat severity
if threat is confirmed **then**
    Trigger mitigation actions (block IP)
    Log the incident in threat database
end if
Update monitoring dashboard with current status and threats
return Identified threat types and corresponding mitigation actions
=0

Chapter 6

TESTING

The testing approach evaluated the performance and functionality of the proposed cybersecurity system as shown in Table 6.1. It focused on detecting and mitigating network threats through modules like Firewall, Intrusion Detection System (IDS), Anomaly Detection, Threat Classification, and Automated/Manual Threat Mitigation. Multiple scenarios were tested, including detection of scanning behavior, blocking of malicious IPs, auto-mitigation of all types of detected attacks, log access by administrators, handling of botnet command traffic, and correct classification of attack types. Special attention was given to edge cases such as false positives (normal traffic misclassified as threats), system failures to detect anomalies, and admin-triggered manual controls.

6.1 Test Cases

Table 6.1: Test Cases for Proposed Cybersecurity System

TestCase ID & Name	**TestCase Description**	**TestCase Steps**	**Test Data**	**Expected Output**	**Actual Output**	**Status**
TC-01: Firewall Block IP	Test whether the firewall correctly blocks a known malicious IP address.	i. Configure firewall rules to block IP `10.10.1.11`. ii. Simulate traffic from IP `10.10.1.11`.	IP: 10.10.1.11	Access is denied and IP is blocked.	Access denied and IP logged.	Pass

TestCase ID & Name	**TestCase Description**	**TestCase Steps**	**Test Data**	**Expected Output**	**Actual Output**	**Status**
TC-02: IDS Detect Scan	Validate that IDS detects scanning behavior from connected devices.	i. Start the IDS module to scan all connected IPs for suspicious activity. ii. Check IDS logs for detection results.	Connected IPs with scan behavior	IDS should detect scanning activity and log the event.	Scan address detected and logged.	Pass
TC-03: Anomaly Detection Spike	Verify if anomaly detection triggers under high traffic spike.	i. Generate a sudden burst of traffic from a specific IP. ii. Observe the anomaly detection engine.	Traffic volume 10x higher than normal	Anomaly should be detected and flagged.	Anomaly detected and flagged.	Pass
TC-04: Threat Classification - SQL Injection	Confirm that SQL Injection attacks are classified correctly by the system.	i. Launch a SQL Injection attack simulation. ii. Check the classification result from the system.	Simulated SQL Injection	Attack is classified as a SQL Injection attack.	Correctly classified as SQL Injection.	Pass
TC-05: Auto Mitigation - DoS/DDoS	Validate automatic mitigation of DoS/DDoS traffic.	i. Inject known DoS/DDoS signature traffic into the system. ii. Observe whether the system mitigates the threat.	DoS/DDoS behavior pattern	System should automatically block and log the threat.	Threat blocked automatically.	Pass

TestCase ID & Name	**TestCase Description**	**TestCase Steps**	**Test Data**	**Expected Output**	**Actual Output**	**Status**
TC-06: Admin Log Access	Check if the admin can view all stored threat logs.	i. Log in to the admin dashboard. ii. Navigate to the threat logs section.	Admin login credentials	All threat logs are listed with timestamps.	Logs displayed successfully.	Pass
TC-07: Anomaly Detection	Evaluate if the system detects unexpected anomalous behavior and notifies the admin.	i. Simulate unexpected traffic behavior from a connected IP. ii. Observe the anomaly detection system for alerts. iii. Confirm that the system logs the anomaly. iv. Verify that a notification is successfully sent to the admin.	Unusual traffic pattern from IP: 10.10.1.11	Anomaly should be detected and a notification should be sent to the admin.	Anomaly detected and alert successfully sent to the admin.	Pass
TC-08: Misclassified Web Attacks	Test the accuracy of Web Attacks detection by the threat classifier.	i. Simulate a Web Attacks attempt through network traffic. ii. Check the classification result. iii. Confirm the classification matches the attack type.	Web Attacks.	The attack should be classified as Web Attacks.	Correctly classified as Web Attacks.	Pass

TestCase ID & Name	**TestCase Description**	**TestCase Steps**	**Test Data**	**Expected Output**	**Actual Output**	**Status**
TC-09: False Positive - Normal Traffic Misclassified	Validate that the system does not falsely classify legitimate traffic as an attack.	i. Send normal user traffic (e.g., browsing, file download) from IP: 10.10.1.20. ii. Monitor the system's detection and classification process. iii. Confirm that no alerts or misclassifications are triggered.	Normal HTTP/HTTPS traffic from IP: 10.10.1.20	Traffic should pass without being flagged as malicious.	Traffic allowed and not flagged; no false positives recorded.	Pass
TC-10: Admin Manual Block IP	Check manual blocking of an IP address by admin.	i. Admin selects malicious IP `10.10.1.46` from logs. ii. Admin clicks the "Block" button. iii. System confirms and logs the block action. iv. Verify IP is added to firewall block list.	IP: 10.10.1.46	IP should be added to the firewall block list.	IP successfully added to firewall block list.	Pass

Chapter 7

RESULTS & DISCUSSIONS

The hybrid AI-ML system effectively detected and classified cyber threats in real-time from various network data sources. It accurately identified multiple attack types and triggered timely mitigation actions. The monitoring dashboard provided clear, real-time updates, demonstrating the system's reliability and effectiveness in enhancing network security.

7.1 Model Training and Evaluation

Several machine learning and deep learning models were developed and evaluated using the CICIDS2017 dataset in order to accurately identify and stop cyberthreats. XGBoost, CNN-LSTM, Autoencoder, Isolation Forest, and a proposed ensemble model that combines autoencoder and isolation forest predictions to improve robustness and reliability.

The Autoencoder was trained using only typical traffic data in a semi-supervised fashion. Reconstruction error was employed as a threshold to identify anomalies during in-ference. By allocating anomaly scores according to feature-space isolation, the Isolation Forest functioned similarly in an unsupervised environment.

The Ensemble model uses a majority voting technique for binary classification to combine the results of the autoencoder and isolation forest models. By utilizing each algorithm's complementary strengths, this method increases general ro-bustness.

A multi-class classification assignment was used to train XGBoost for supervised learning, and it demonstrated amazing precision and recall for all attack types.The CNN-LSTM model was used to capture temporal dependencies in the data by taking advantage of the sequential pattern of network traffic flows.

Table 7.1: Model Performance Metrics for Cyber Threat Detection

Model	**Accuracy**	**Precision**	**Recall**	**F1-score**
Isolation Forest	0.53	0.51	0.53	0.51
Autoencoder	0.77	0.77	0.77	0.76
Ensemble (ISF + AE)	0.80	0.78	0.78	0.79
XGBoost	1.00	1.00	1.00	1.00
CNN-LSTM	0.99	0.99	0.99	0.99

According to the results from Table 7.1, the Ensemble model provides the most balanced

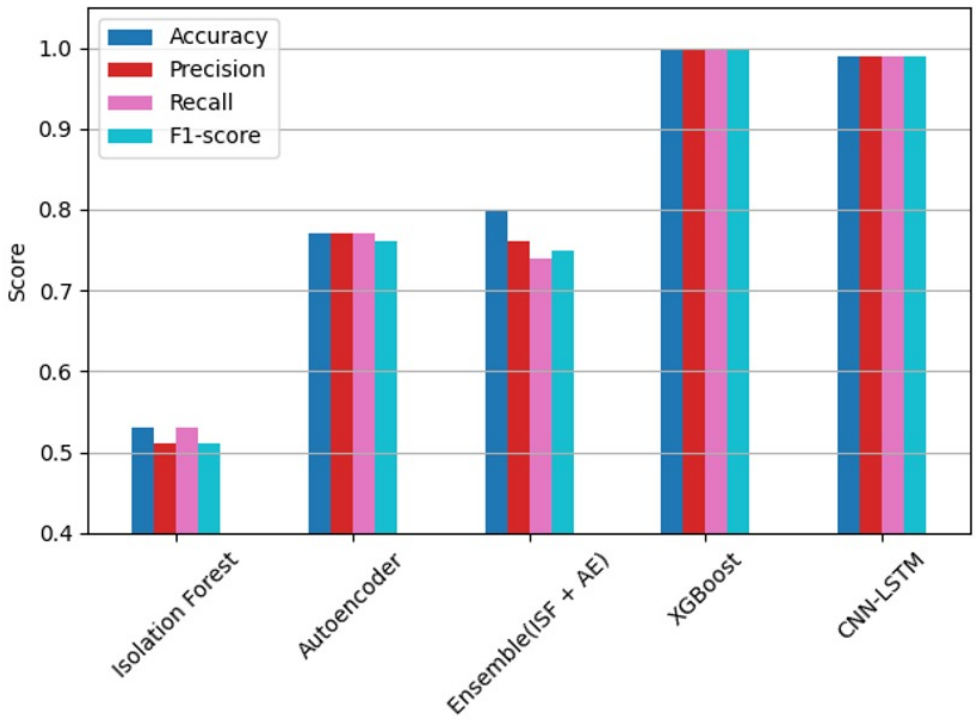

Figure 7.1: Performance Metrics of Cyber Threat Detection and Prevention Models

and dependable performance, particularly when applied to wider network settings, while XG-Boost and CNN-LSTM show nearly perfect classification on the test set. It performs better in F1-score than the independent Autoencoder and Isolation Forest, which makes it a good fit for real-time threat detection systems where robustness and consistency are essential.

Accuracy, precision, recall, and F1-score are used to com- pare the performance of five models in Figure 7.1 While Autoen- coder and Isolation Forest perform moderately, their ensemble (IF + AE) slightly increases accuracy. XGBoost and CNN- LSTM perform better in anomaly detection tasks, as seen through their near-perfect results on all measures.

7.2 System Implementation

A Flask-based web application that combines ML and DL trained models is used to construct the proposed hybrid cyber threat detection system. Real-time anomaly detection and result visualization via an easy-to-use user interface. A real-time network security dashboard with system IP addresses, connected devices, and a list of malicious IPs found is displayed in the Figure 7.2 This configuration offers fast visibility into network security risks as well as centralized monitoring.

The statistical distribution of the different types of cyber- attacks that have been identified in the network environment under observation is shown in the pie chart in Figure 7.3.

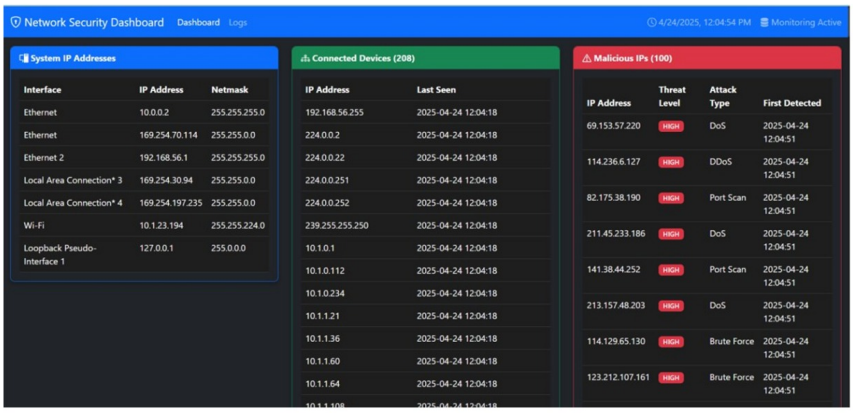

Figure 7.2: Network Security Dashboard Overview

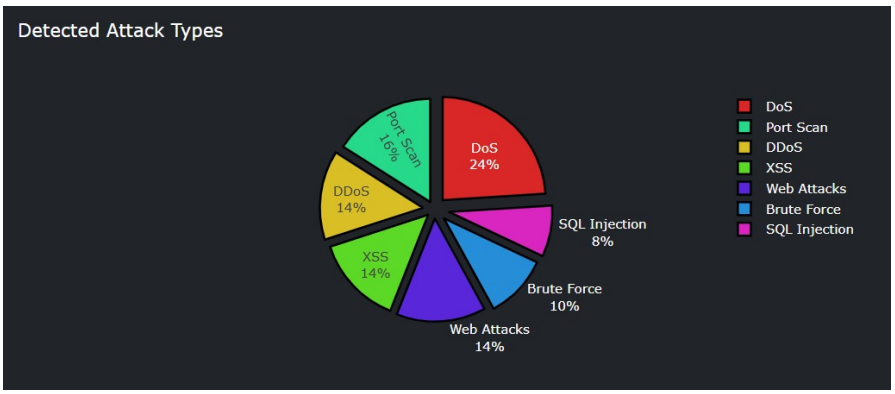

Figure 7.3: statistical distribution of Detected Attack Types

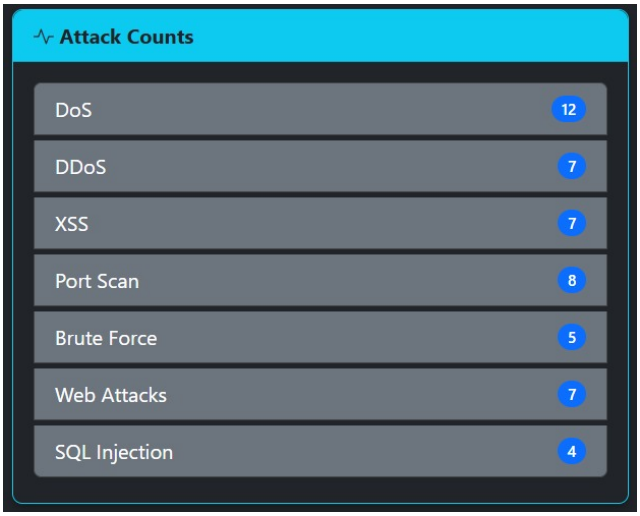

Figure 7.4: Recorded Counts of Detected Attack Types

Figure 7.5: Threat Mitigation Interface of the Network Security Dashboard

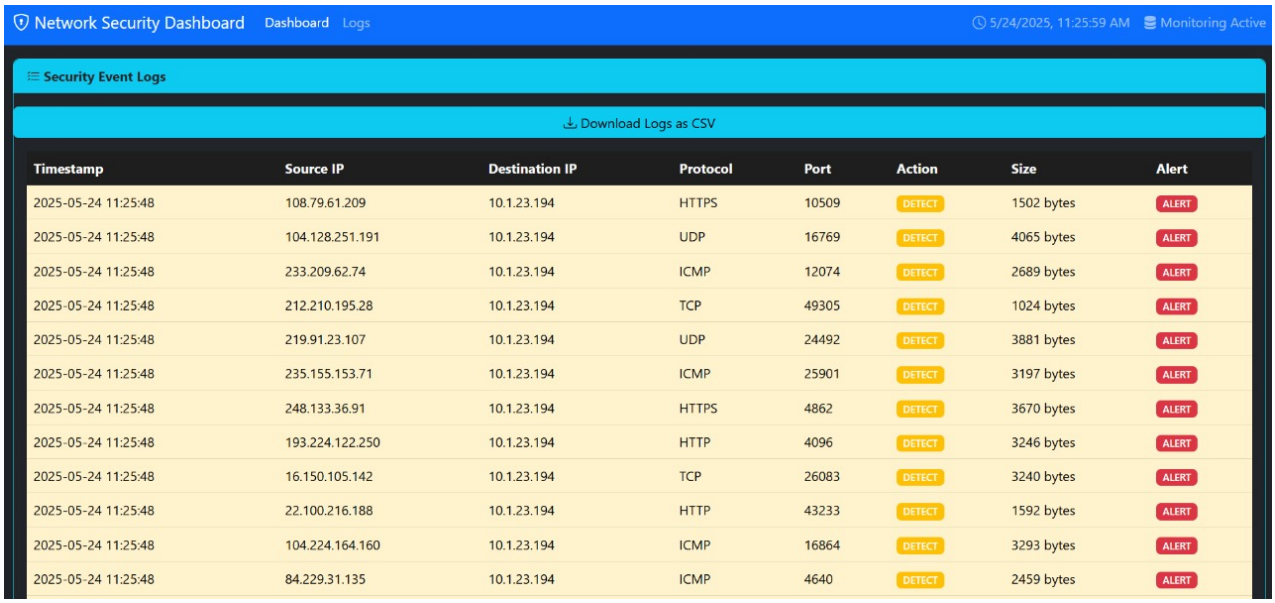

Figure 7.6: Real-time Network Security Dashboard displaying active security event logs

The frequency of the different sorts of attacks that the network monitoring system has detected is displayed in a bar- style list in Figure 7.4. In order to prioritize defense measures in the network security plan, these counts offer a quantitative viewpoint on threat occurrences. This Figure 7.5 shows how the network security dashboard's threat mitigation features work. Administrators can use it to enter a malicious or suspicious IP address and implement a mitigation measure, such blocking the address.

A network security dashboard displaying many real-time security events is shown in the Figure 7.6. The timestamp, protocol type, port number, source and destination IP addresses, and data size are all recorded in each log entry.

Chapter 8

CONCLUSION AND FUTURE SCOPE

This work aimed to identify the most effective approach for detecting cyber threats using a hybrid combination of machine learning and deep learning models. Utilizing the CICIDS2017 dataset, several models were implemented and evaluated, including CNN-LSTM, XGBoost, Autoencoder, and Isolation Forest. Among these, the supervised models—XGBoost and CNN-LSTM—demonstrated superior performance, achieving near-perfect results across key evaluation metrics. In contrast, the unsupervised models, Autoencoder and Isolation Forest, yielded average results when used independently. However, their ensemble combination showed a notable improvement in detection capability, offering a more balanced and robust solution. This highlights the potential of hybrid approaches that integrate both supervised and unsupervised learning for more comprehensive threat detection. The trained models were also successfully integrated into a real-time cyber threat detection and prevention system using a Flask-based dashboard, showcasing the system's viability in real-world cybersecurity environments.

Future enhancements to the system can focus on further improving accuracy, scalability, and real-time responsiveness. Potential directions include integrating external threat intelligence feeds to provide broader context for threat detection, incorporating live packet capture technologies for real-time data analysis, and applying advanced ensemble techniques such as stacking or blending to improve predictive performance. Additionally, implementing adaptive or online learning methods could enable the system to evolve continuously by learning from new types of attacks, ensuring long-term effectiveness in dynamic and evolving network environments.

REFERENCES

[1] Giovanni Apruzzese et al. "On the effectiveness of machine and deep learning for cyber security". In: *2018 10th international conference on cyber Conflict (CyCon)*. IEEE. 2018, pp. 371–390.

[2] Sikha Bagui et al. "Using machine learning techniques to identify rare cyber-attacks on the UNSW-NB15 dataset". In: *Security and Privacy* 2.6 (2019), e91.

[3] Bhoopesh Singh Bhati et al. "An improved ensemble based intrusion detection technique using XGBoost". In: *Transactions on emerging telecommunications technologies* 32.6 (2021), e4076.

[4] Jonathan Cook, Sabih Ur Rehman, and M Arif Khan. "Security and privacy for low power iot devices on 5g and beyond networks: Challenges and future directions". In: *IEEE Access* 11 (2023), pp. 39295–39317.

[5] Selçuk Demir and Emrehan Kutlug Sahin. "An investigation of feature selection methods for soil liquefaction prediction based on tree-based ensemble algorithms using AdaBoost, gradient boosting, and XGBoost". In: *Neural Computing and Applications* 35.4 (2023), pp. 3173–3190.

[6] Sukhpreet Singh Dhaliwal, Abdullah-Al Nahid, and Robert Abbas. "Effective intrusion detection system using XGBoost". In: *Information* 9.7 (2018), p. 149.

[7] Malik Al-Essa and Annalisa Appice. "Dealing with imbalanced data in multi-class network intrusion detection systems using xgboost". In: *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*. Springer. 2021, pp. 5–21.

[8] Jiaorong Fan. "Analyzing the applicability of isolation forest for detecting anomalies in time series data". B.S. thesis. J. Fan, 2025.

[9] Byungkak Jeon. "Enhancing Intrusion Detection Systems Using Deep Learning Techniques: A Comparative Study on CICIDS 2017 Dataset". PhD thesis. 2023.

[10] Neeraj Kumar and Sanjeev Sharma. "A hybrid modified deep learning architecture for intrusion detection system with optimal feature selection". In: *Electronics* 12.19 (2023), p. 4050.

[11] Ainomugisha Maxima. "Integration and analysis of unstructured data towards database optimization and decision making using deep learning techniques". PhD thesis. Kampala International University, 2024.

[12] Xianghui Meng. "Advanced AI and ML techniques in cybersecurity: Supervised and unsupervised learning, and neural networks in threat detection and response". In: *Applied and Computational Engineering* 82 (2024), pp. 24–28.

[13] M Nalini et al. "Enhancing early attack detection: novel hybrid density-based isolation forest for improved anomaly detection". In: *International Journal of Machine Learning and Cybernetics* (2024), pp. 1–19.

[14] Kingsley David Onyewuchi Ofoegbu et al. "Real-Time Cybersecurity threat detection using machine learning and big data analytics: A comprehensive approach". In: *Computer Science & IT Research Journal* 4.3 (2024).

[15] Ahmed Saaudi et al. "Insider threats detection using CNN-LSTM model". In: *2018 International conference on computational science and computational intelligence (CSCI)*. IEEE. 2018, pp. 94–99.

[16] Kumar Saurabh et al. "Lbdmids: LSTM based deep learning model for intrusion detection systems for IOT networks". In: *2022 IEEE World AI IoT Congress (AIIoT)*. IEEE. 2022, pp. 753–759.

[17] Bambang Susilo, Abdul Muis, and Riri Fitri Sari. "Intelligent Intrusion Detection System Against Various Attacks Based on a Hybrid Deep Learning Algorithm". In: *Sensors* 25.2 (2025), p. 580.

[18] Tala Talaei Khoei and Naima Kaabouch. "A comparative analysis of supervised and unsupervised models for detecting attacks on the intrusion detection systems". In: *Information* 14.2 (2023), p. 103.

[19] Xinwei Yuan et al. "A simple framework to enhance the adversarial robustness of deep learning-based intrusion detection system". In: *Computers & Security* 137 (2024), p. 103644.

[20] Zhimin Zhang et al. "Artificial intelligence in cyber security: research advances, challenges, and opportunities". In: *Artificial Intelligence Review* (2022), pp. 1–25.

Appendix A

APPENDIX

Cyber Threat Detection and Prevention using AI-ML Model (Hybrid Approach) is a cyber security system developed to identify and prevent various cyber threats using artificial intelligence and machine learning techniques. The dataset used includes network traffic details such as IP addresses, ports, protocols, and labeled threat types like malware, phishing, and DoS attacks. It was trained using Python libraries such as TensorFlow, Keras, and Scikit-learn.

Cyber Threat Detection and Prevention using AI-ML Model (Hybrid Approach) was evaluated using performance metrics such as accuracy, precision, recall, F1-score. The model outputs are provided. The system was tested to demonstrate real-time detection. Code snippets for data preprocessing, model training, and prediction are included as part of the implementation.

Cyber Threat Detection and Prevention using AI-ML Model (Hybrid Approach) was developed on a system with Intel i7 processor and 16GB RAM. The tools and environments used include Jupyter Notebook/google colab, Visual Studio Code. A list of terms and acronyms such as AI-ML is included for reference. Relevant links, libraries, and sources used during the project are also documented to support further research and development.

Chapter 9

PLAGIARISM REPORT

ORIGINALITY REPORT

12%	6%	10%	6%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	Submitted to B.V. B College of Engineering and Technology, Hubli Student Paper	5%
2	"Proceedings of 5th International Ethical Hacking Conference", Springer Science and Business Media LLC, 2025 Publication	3%
3	Thangaprakash Sengodan, Sanjay Misra, M Murugappan. "Advances in Electrical and Computer Technologies", CRC Press, 2025 Publication	1%
4	www.coursehero.com Internet Source	<1%
5	"Intelligent Systems Design and Applications", Springer Science and Business Media LLC, 2024 Publication	<1%
6	Chetna Kaushal, Amandeep Kaur, Mohit Angurala, Aryan Chaudhary. "Zero-Trust Learning - ", Apple Academic Press, 2025 Publication	<1%
7	cit3.cdn.swin.edu.au Internet Source	<1%