PhD thesis Himanshu.docx

Delhi Technological University

Document Details

Submission ID
trn:oid:::27535:83077245

Submission Date
Feb 22, 2025, 8:54 PM GMT+5:30

Download Date
Feb 22, 2025, 9:03 PM GMT+5:30

File Name
PhD thesis Himanshu.docx

File Size
12.1 MB

238 Pages

85,703 Words

532,867 Characters

8% Overall Similarity

The combined total of all matches, including overlapping sources, for each database.

Filtered from the Report

- Bibliography
- Cited Text
- Small Matches (less than 10 words)
- Crossref database
- Crossref posted content database

Match Groups

242 Not Cited or Quoted 8%
Matches with neither in-text citation nor quotation marks

0 Missing Quotations 0%
Matches that are still very similar to source material

6 Missing Citation 0%
Matches that have quotation marks, but no in-text citation

0 Cited and Quoted 0%
Matches with in-text citation present, but no quotation marks

Top Sources

4% Internet sources
5% Publications
4% Submitted works (Student Papers)

Integrity Flags

0 Integrity Flags for Review

No suspicious text manipulations found.

Our system's algorithms look deeply at a document for any inconsistencies that would set it apart from a normal submission. If we notice something strange, we flag it for you to review.

A Flag is not necessarily an indicator of a problem. However, we'd recommend you focus your attention there for further review.

Match Groups

- **242** Not Cited or Quoted 8%
  Matches with neither in-text citation nor quotation marks
- **0** Missing Quotations 0%
  Matches that are still very similar to source material
- **6** Missing Citation 0%
  Matches that have quotation marks, but no in-text citation
- **0** Cited and Quoted 0%
  Matches with in-text citation present, but no quotation marks

Top Sources

- 4% Internet sources
- 5% Publications
- 4% Submitted works (Student Papers)

Top Sources

The sources with the highest number of matches within the submission. Overlapping sources will not be displayed.

1	Internet	www.mdpi.com	<1%
2	Internet	robots.net	<1%
3	Internet	web.realinfo.tv	<1%
4	Internet	dokumen.pub	<1%
5	Internet	arxiv.org	<1%
6	Publication	Mehdi Ghayoumi. "Generative Adversarial Networks in Practice", CRC Press, 2023	<1%
7	Submitted works	Kingston University on 2024-08-19	<1%
8	Publication	Xu, Yiming. "3d State Space Analysis of Anisotropic Laminated Composites", The ...	<1%
9	Internet	www.ijcttjournal.org	<1%
10	Internet	ebin.pub	<1%

11 Internet

ijrpr.com <1%

12 Submitted works

Liverpool John Moores University on 2024-04-08 <1%

13 Publication

H.L. Gururaj, Francesco Flammini, S. Srividhya, M.L. Chayadevi, Sheba Selvam. "Co... <1%

14 Internet

etd.aau.edu.et <1%

15 Internet

repository.futo.edu.ng <1%

16 Submitted works

University of Northumbria at Newcastle on 2023-09-21 <1%

17 Publication

Wang, Jian. "System Modeling for Connected and Autonomous Vehicles", Purdue ... <1%

18 Submitted works

Adventist University of Africa on 2023-04-16 <1%

19 Publication

T. Mariprasath, Kumar Reddy Cheepati, Marco Rivera. "Practical Guide to Machin... <1%

20 Publication

Thiele, Lukas Tom. "Uber's Scale-Up – Case Study: Introduction of Autonomous Ve... <1%

21 Internet

backend.orbit.dtu.dk <1%

22 Internet

bmcmedinformdecismak.biomedcentral.com <1%

23 Internet

journals.plos.org <1%

24 Publication

Adesh Kumar, Surajit Mondal, Gaurav Verma, Prashant Mani. "Embedded Devices... <1%

25 Submitted works
Liverpool John Moores University on 2025-01-29 <1%

26 Submitted works
University of Greenwich on 2023-08-16 <1%

27 Submitted works
University of Hertfordshire on 2024-04-29 <1%

28 Internet
academic-accelerator.com <1%

29 Submitted works
Coventry University on 2023-08-07 <1%

30 Submitted works
GGS IP University Delhi on 2024-09-28 <1%

31 Publication
Alex Khang, Vugar Abdullayev, Olena Hrybiuk, Arvind K. Shukla. "Computer Visio... <1%

32 Submitted works
Jawaharlal Nehru Technological University on 2024-07-03 <1%

33 Publication
Dinesh Goyal, Bhanu Pratap, Sandeep Gupta, Saurabh Raj, Rekha Rani Agrawal, I... <1%

34 Publication
Hamed Taherdoost, Mohsen Saeedi, Aydin Shishegaran. "Applications of Blockch... <1%

35 Submitted works
Liverpool John Moores University on 2025-02-21 <1%

36 Internet
www.researchsquare.com <1%

37 Publication
Al Banna, Saeed. "Integration of Blockchain and Machine Learning for Robust Int... <1%

38 Submitted works
Liverpool John Moores University on 2023-11-25 <1%

39 Publication

R. Anandan, M. Senthil Kumar, C. L. Biji, Vicente García-Díaz, Souvik Pal. "Next-Ge... <1%

40 Publication

Alzahrani, Amani. "Misinformation Detection in the Social Media Era", Howard Un... <1%

41 Submitted works

Augusta State University on 2023-08-26 <1%

42 Publication

V. Sridhar, Sita Rani, Piyush Kumar Pareek, Pankaj Bhambri, Ahmed A. Elngar. "Bl... <1%

43 Publication

Pramod R. Gunjal, Satish R. Jondhale, Jaime Lloret, Karishma Agrawal. "Internet o... <1%

44 Publication

Tasmiya salim Mujawar. "Urban Noise Classification Using Machine Learning Tec... <1%

45 Publication

Dombach, Amy E.. "Social Networks and Academic Flourishing in First- and Contin... <1%

46 Publication

K. Venkata Murali Mohan, M. Suresh Babu. "Disruptive Technologies in Computin... <1%

47 Publication

Vyas, Monika. "Emotion Discovery in Hindi-English Code-Mixed Conversations", P... <1%

48 Internet

yingo.ca <1%

49 Submitted works

Asia Pacific University College of Technology and Innovation (UCTI) on 2024-10-18 <1%

50 Publication

Charlene Gallery, Jo Conlon. "Fashion Business and Digital Transformation - Tech... <1%

51 Publication

Ruqiang Yan, Jing Lin. "Equipment Intelligent Operation and Maintenance", CRC P... <1%

52 Submitted works

University of Surrey on 2023-05-16 <1%

53 Internet

di.univ-blida.dz <1%

54 Publication

Leroy, Pascal. "Contributions to Multi-Agent Reinforcement Learning", Universite ... <1%

55 Internet

epub.uni-regensburg.de <1%

56 Submitted works

University of West London on 2024-09-16 <1%

57 Publication

Arvind Dagur, Dhirendra Kumar Shukla, Nazarov Fayzullo Makhmadiyarovich, Ak... <1%

58 Publication

Mokhtarian, Ilia. "Utilizing Process Mining and Deep Learning to Detect IoT / IIoT ... <1%

59 Publication

Youssef Baddi, Mohammed Amin Almaiah, Omar Almomani, Yassine Maleh. "The ... <1%

60 Submitted works

Al-Iraqia University on 2024-06-04 <1%

61 Submitted works

University Tun Hussein Onn Malaysia on 2024-08-12 <1%

62 Submitted works

De Montfort University on 2024-08-30 <1%

63 Submitted works

Jawaharlal Nehru Technological University on 2024-06-12 <1%

64 Submitted works

Liverpool John Moores University on 2024-03-15 <1%

65 Publication

Tariq M. Arif, Md Adilur Rahim. "Deep Learning for Engineers", CRC Press, 2024 <1%

66 Submitted works

University of Surrey on 2024-09-19 <1%

67 Publication
Al-Sakib Khan Pathan. "Security of Self-Organizing Networks - MANET, WSN, WM... <1%

68 Publication
Alemu, Shegaw Tiruneh. "A Machine Learning Intrusion Detection System (IDS) T... <1%

69 Submitted works
British University in Egypt on 2024-12-07 <1%

70 Submitted works
Concordia University on 2025-01-11 <1%

71 Publication
Mahmoud Hassaballah, Ali Ismail Awad. "Deep Learning in Computer Vision - Prin... <1%

72 Publication
Neeraj Kumar, N. Gayathri, Md. Arafatur Rahman, B. Balamurugan. "Blockchain, ... <1%

73 Publication
Shin-ya Nishizaki, Masayuki Numao, Merlin Teodosia Suarez, Jaime Caro. "Theory ... <1%

74 Submitted works
University of Gloucestershire on 2024-09-02 <1%

75 Submitted works
University of Hertfordshire on 2023-08-26 <1%

76 Submitted works
University of West London on 2023-06-04 <1%

77 Internet
repo.lib.tut.ac.jp <1%

78 Internet
www.techscience.com <1%

79 Publication
Delaram Kahrobaei, Enrique Domínguez, Reza Soroushmehr. "Artificial Intelligen... <1%

80 Publication
Devendra Prasad, Suresh Chand Gupta, Anju Bhandari Gandhi, Stuti Mehla, Upas... <1%

81 Submitted works
Eotvos Lorand University on 2024-12-01 <1%

82 Submitted works
Gitam University on 2023-07-26 <1%

83 Submitted works
Liverpool John Moores University on 2023-08-26 <1%

84 Submitted works
Liverpool John Moores University on 2024-08-29 <1%

85 Publication
Mendis, John Udara. "A Predictive Model for Forecasting the Severity of Recall of ... <1%

86 Publication
R. N. V. Jagan Mohan, Vasamsetty Chandra Sekhar, V. M. N. S. S. V. K. R. Gupta. "Al... <1%

87 Submitted works
Universiteit Hasselt on 2024-08-20 <1%

88 Submitted works
University of Hertfordshire on 2025-01-06 <1%

89 Internet
link.springer.com <1%

90 Internet
www.isteonline.in <1%

91 Publication
Εμμανουήλ, Κρητικός. "Predictive Modeling of Online Social Behaviour Using Dyn... <1%

92 Publication
Al-Sumaidaee, Ghassan. "Blockchain Tokens as Universal Encrypted Access : A Co... <1%

93 Publication
Jaiteg Singh, S B Goyal, Rajesh Kumar Kaushal, Naveen Kumar, Sukhjit Singh Sehr... <1%

94 Publication
Minghao Wang. "Utilizing Blockchain for Privacy Preservation in Internet of Thing... <1%

95 Publication
Mourade Azrour, Jamal Mabrouki, Azidine Guezzaz, Said Benkirane. "Blockchain a... <1%

96 Publication
Pethuru Raj, Kavita Saini, Chellammal Surianarayanan. "Blockchain Technology a... <1%

97 Submitted works
Rochester Institute of Technology on 2023-12-13 <1%

98 Submitted works
The University of the West of Scotland on 2023-12-15 <1%

99 Submitted works
University of Glamorgan on 2024-09-01 <1%

100 Submitted works
University of Hertfordshire on 2023-08-28 <1%

101 Submitted works
University of Hertfordshire on 2024-12-02 <1%

102 Submitted works
University of Northumbria at Newcastle on 2024-08-20 <1%

103 Internet
core.ac.uk <1%

104 Internet
jmc.edu <1%

105 Internet
mzuir.inflibnet.ac.in <1%

106 Internet
ulspace.ul.ac.za <1%

107 Internet
www.frontiersin.org <1%

108 Publication
Ajay Kumar, Parveen Kumar, Yang Liu, Rakesh Kumar. "Handbook of Intelligent a... <1%

109 Publication
Albalwy, Faisal. "A Blockchain Infrastructure to Support Smart Contracts-Based C... <1%

110 Publication
Almadani, Mwaheb. "An AI-Driven, Secure, and Trustworthy Ranking System for B... <1%

111 Publication
Amit Kumar Tyagi. "Data Science and Data Analytics - Opportunities and Challeng... <1%

112 Submitted works
Colorado Technical University Online on 2025-02-19 <1%

113 Publication
Debasis Chaudhuri, Jan Harm C Pretorius, Debashis Das, Sauvik Bal. "Internationa... <1%

114 Publication
Dolan, Paige M.. "Development of an Advanced Step Counting Algorithm with Int... <1%

115 Submitted works
Galatasaray University on 2024-12-15 <1%

116 Publication
H.L. Gururaj, Francesco Flammini, J. Shreyas. "Data Science & Exploration in Artifi... <1%

117 Publication
Iraq Ahmad Reshi, Sahil Sholla. "Blockchain-based Internet of Things - Opportunit... <1%

118 Publication
Ismail, Shereen. "A Hybrid Framework to Secure IoT Sensor Networks Based on Bl... <1%

119 Publication
Sandeep Kumar, Shilpa Rani, K. Ramya Laxmi. "Artificial Intelligence and Machine... <1%

120 Publication
Shalli Rani, Ashu Taneja. "WSN and IoT - An Integrated Approach for Smart Applic... <1%

121 Publication
Shitharth Selvarajan, Gouse Baig Mohammad, Sadda Bharath Reddy, Praveen Ku... <1%

122 Submitted works
Sunway Education Group on 2024-07-18 <1%

123 Publication

T. Kavitha, M. K. Sandhya, V. J. Subashini, Prasidh Srikanth. "Secure Communicati... <1%

124 Publication

Thangavel Murugan, W. Jai Singh. "Cybersecurity and Data Science Innovations fo... <1%

125 Submitted works

Universiti Tenaga Nasional on 2023-06-04 <1%

126 Submitted works

University of Al-Qadisiyah on 2023-08-23 <1%

127 Submitted works

University of Portsmouth on 2024-05-23 <1%

128 Submitted works

University of West London on 2024-05-16 <1%

129 Publication

Xinyuan Song, Qian Niu, Junyu Liu, Benji Peng, Sen Zhang, Ming Liu, Ming Li, Tian... <1%

130 Publication

da Cunha Ceulin, Wagner Luiz. "Knowledge Elicitation in Deep Learning Models", ... <1%

131 Internet

ijircce.com <1%

132 Internet

www.ijcert.org <1%

133 Internet

www.ijraset.com <1%

Chapter 1: Introduction

1. Introduction

The Internet of Things (IoT) transforms how devices interact, communicate, and make autonomous decisions. IoT applications have permeated various sectors, from smart homes and healthcare to industrial automation, enabling real-time data collection, monitoring, and control. However, IoT systems' vast and decentralized nature exposes them to a myriad of security and privacy challenges, making them susceptible to numerous cyberattacks. Intrusion detection systems (IDS) are a primary line of defense for identifying and mitigating malicious activities within IoT networks. Despite various IDS models, traditional IDS mechanisms often fail to address the complexities of modern IoT environments due to their static nature, limited scalability, and inability to adapt to the heterogeneity of IoT devices.

With the advent of advanced technologies like Artificial Intelligence (AI) and Blockchain, there is a significant opportunity to redefine the security frameworks for IoT systems. AI-driven models provide intelligent and adaptive capabilities, allowing for more accurate and efficient detection of anomalies. On the other hand, blockchain introduces a decentralized and immutable ledger for secure data storage and access control, making it a suitable technology to enhance trust and transparency in IDS. This research explores the integration of AI and blockchain technology to develop a novel IDS framework, aiming to provide enhanced security, privacy, and scalability for IoT environments.

1.1. Motivation

The exponential growth in deploying Internet of Things (IoT) devices, expected to surpass 75 billion by 2025, has revolutionized various industries, including healthcare, manufacturing, transportation, and smart cities. This surge in IoT adoption has led to significant innovations and the emergence of new business models. However, it has also introduced considerable security challenges, as increased connectivity creates new vulnerabilities. IoT devices, often operating in resource-constrained and hostile environments, are susceptible to various cyberattacks, including Distributed Denial of Service (DDoS), Man-in-the-Middle (MitM) attacks, and data breaches. Reports suggest that IoT devices face an average of over 5,000 attacks per month, leading to severe disruptions in critical services and operations.

Furthermore, traditional security frameworks designed for conventional networks must be more suited to address the specific needs of IoT systems, which require real-time communication and support for diverse protocols and constrained resources. The limitations of existing security mechanisms highlight the need for an advanced Intrusion Detection System (IDS) framework tailored to the unique challenges of IoT environments. Integrating emerging technologies such as artificial intelligence (AI) and blockchain has shown promising potential in addressing these challenges. AI can enhance IDS by enabling the detection of sophisticated attack patterns, while blockchain provides a secure, decentralized infrastructure for data integrity, traceability, and access control. This research seeks to bridge the existing security gap by proposing a novel AI-driven IDS framework integrated with blockchain technology to strengthen security, privacy, and resilience in IoT networks.

1.2. Internet of Things

The Internet of Things (IoT) is a network of physical objects, referred to as "things," embedded with sensors, software, and other technologies to connect and exchange data with other devices and systems over the Internet. IoT enables the automation of processes and facilitates seamless communication between devices without human intervention, creating an interconnected ecosystem of smart devices.
The key characteristics of IoT include:

- **Connectivity**: IoT devices are connected through various communication protocols, enabling them to interact and exchange data.
- **Scalability**: The ability of IoT systems to scale and support billions of devices without compromising performance.
- **Intelligence**: IoT devices can collect, analyze, and act on data using AI and machine learning algorithms.

1.2.1. Architecture of IoT

The architecture of IoT is typically structured into multiple layers, each with distinct functionalities to support the end-to-end operations of IoT applications. The basic IoT architecture can be divided into the following layers:

1. **Perception Layer**:
   - **Components**: Sensors, actuators, RFID tags, and other data acquisition devices.
   - **Functionality**: This layer captures data from the physical environment, such as temperature, humidity, motion, and other environmental conditions. Sensors convert physical signals into digital signals, which are then transmitted to the upper layers for processing.
2. **Network Layer**:
   - **Components**: Gateways, routers, communication protocols (e.g., Wi-Fi, Zigbee, Bluetooth).
   - **Functionality**: This layer is responsible for transmitting data collected by the perception layer to other devices, cloud servers, or applications. It employs various networking technologies and protocols to ensure reliable and secure communication.
3. **Middleware Layer**:
   - **Components**: IoT platforms, data processing units.
   - **Functionality**: The middleware layer provides a data aggregation, filtering, and processing platform. It handles the interactions between different IoT devices and provides an interface for application development.
4. **Application Layer**:
   - **Components**: IoT applications (smart homes, healthcare monitoring, industrial automation).
   - **Functionality**: This layer provides end-user services and interfaces for IoT applications, enabling users to interact with the system and make informed decisions based on the data generated by IoT devices.

Table 1.1: IoT Architecture Overview

Layers	Component	Functionality
Perception Layer	Sensors, Actuators	Data collection and environmental monitoring
Network Layer	Gateways, Routers, Protocols	Data transmission and communication
Middleware Layer	IoT Platform, Data Processing	Data aggregation, filtering, and processing
Application Layer	IoT Applications, User Interface	User interaction and service delivery

1.2.2. Industrial IoT

Industrial IoT (IIoT) refers to the application of IoT technology in industrial settings, including manufacturing, supply chain management, logistics, and healthcare. IIoT devices monitor and control industrial operations, leading to improved efficiency, reduced downtime, and enhanced safety. Unlike general IoT, IIoT focuses on mission-critical applications that require high reliability, low latency, and robust security.

i. Applications of IIoT:

- **Smart Manufacturing**: Real-time monitoring and control of production processes to optimize resource utilization and reduce waste.
- **Predictive Maintenance**: Using IIoT devices to monitor the health of equipment and predict failures before they occur, reducing downtime and maintenance costs.
- **Supply Chain Optimization**: Real-time tracking of goods and assets throughout the supply chain to improve logistics and inventory management.

ii. Security Challenges in IIoT:

- **Data Sensitivity**: IIoT systems often handle sensitive information about industrial processes and proprietary technologies. A breach can result in significant financial losses and damage to reputation.

- **Complex Infrastructure**: The integration of legacy systems with modern IoT devices creates a heterogeneous environment that is challenging to secure.
- **High Stakes of Attack**: Cyberattacks targeting IIoT systems can have catastrophic consequences, including equipment damage and production shutdowns.

1.3. Intrusion Detection System

An Intrusion Detection System (IDS) is a security mechanism designed to detect, analyze, and respond to a network's unauthorized activities or malicious behavior. IDS helps safeguard network resources by monitoring system activities, analyzing patterns, and generating alerts in response to potential threats. In the context of the Internet of Things (IoT) and the Industrial Internet of Things (IIoT), IDS plays a critical role in protecting the network by providing real-time monitoring and anomaly detection and preventing attacks that disrupt sensitive operations.

1.3.1. Type of Intrusion Detection System

Intrusion Detection Systems can be classified into three main categories based on their deployment and monitoring scope:

1. Network-based IDS (NIDS):

Monitors network traffic by capturing and analyzing packet data. NIDS is deployed at strategic points within the network, such as gateways, routers, and firewalls. It helps detect network-level attacks, including:

- **Signature-based NIDS**: Detects attacks by comparing network traffic against predefined attack signatures or patterns. It is highly effective in identifying known threats like SQL injection or port scanning but cannot detect zero-day attacks.
- **Anomaly-based NIDS**: Establishes a baseline of normal network behavior and flags deviations as potential threats. It can detect unknown attacks or new patterns but may produce false positives.
- **Protocol-based NIDS**: Monitors and validates specific protocol behavior (e.g., HTTP, FTP) for compliance and consistency. It helps detect protocol misuse or violations, such as incorrect HTTP requests or FTP commands.

2. Host-based IDS (HIDS):

Monitors system-level activities on individual devices, such as file modifications, process creation, and system logs. HIDS provides detailed information about the internal state of the host, making it effective in detecting attacks like:

- **File Integrity Monitoring (FIM)**: Tracks changes in critical system files and directories to detect unauthorized modifications or malware infections.
- **Log-based HIDS**: Analyzes system logs for suspicious activities like failed login attempts, privilege escalation, or process anomalies.
- **Behavior-based HIDS**: Monitors the behavior of applications and processes to detect deviations from the expected behavior, such as unusual memory usage or process execution.

3. Hybrid IDS:

It combines the capabilities of both NIDS and HIDS to monitor network traffic and host activities comprehensively. Hybrid IDS is designed to detect sophisticated attacks that may leverage both network and system vulnerabilities. It can be further categorized into:

- **Distributed IDS**: Uses multiple IDS instances deployed at various points in the network to provide a coordinated approach to intrusion detection. It is effective in detecting large-scale attacks like botnets.
- **Cross-layer IDS**: Monitors interactions across different layers of the protocol stack (e.g., network, transport, application) to detect multi-layer attacks.
- **Federated IDS**: Shares threat intelligence across different IDS instances or organizations to improve detection capabilities and respond to emerging threats.

Table 1.2: Comparison of IDS Types

IDS Type	Monitoring Scope	Advantages	Limitation
NIDS	Network Traffic	Real-time detection, wide scope	Limited to network-level attacks
HIDS	Host/System Activities	Detailed host-level insights	High resource consumption

Hybrid IDS	Network and Host	Comprehensive monitoring	Increased complexity
Anomaly-based NIDS	Network Behavior	Detect new/unknown attacks	High false-positive rate
Signature-based NIDS	Network Packets	Accurate detection of known attacks	Ineffective against zero-day attacks
File Integrity HIDS	Host files and Directories	Detect unauthorized file change	Can be resource-intensive
Behavior-based HIDS	Host Applications	Detects application anomalies	Complex to configure and maintain

1.3.2. Type of Cyberattack in IoT/IIoT

The proliferation of interconnected devices in IoT and IIoT ecosystems has introduced new attack surfaces and vulnerabilities. Cyberattacks in IoT/IIoT environments can disrupt operations, compromise sensitive data, and pose safety risks. The following are common types of intrusions targeting IoT/IIoT, along with their impact on the Confidentiality, Integrity, and Availability (CIA) triad:

- **Distributed Denial of Service (DDoS)**: An attacker overwhelms a network or device with excessive traffic, disrupting services and causing downtime. DDoS attacks often exploit the limited computational resources of IoT devices.
- **Man-in-the-Middle (MitM)**: An attacker intercepts and manipulates communication between two parties without their knowledge. MitM attacks compromise data integrity and confidentiality.
- **Sybil Attack**: An attacker creates multiple fake identities to manipulate a peer-to-peer network, undermining the integrity and trust of the system.
- **Data Breaches**: Unauthorized access to sensitive data stored in IoT devices or transmitted over the network results in exposure or loss of personal information.
- **Replay Attack**: An attacker captures and replays valid data packets or authentication information to gain unauthorized access to a network or device.
- **Eavesdropping**: An attacker listens to private communication between IoT devices, compromising confidentiality and potentially gathering sensitive information.
- **Jamming Attack**: An attacker disrupts wireless communication between IoT devices by emitting interference signals, causing a loss of connectivity and availability.
- **Physical Tampering**: Direct physical access to IoT devices enables attackers to alter hardware or software configurations, compromising device integrity.
- **Firmware Manipulation**: An attacker modifies the firmware of IoT devices to introduce malicious functionality or turn off security features.
- **Routing Attacks (e.g., Wormhole, Sinkhole)**: An attacker disrupts the routing protocol in IoT networks, causing misrouting of packets and network partitioning.
- **Malware Infections**: IoT devices can be infected with malware like botnets, ransomware, or spyware, enabling attackers to launch attacks or steal data.
- **Privilege Escalation**: An attacker gains higher privileges on an IoT device than intended, enabling unauthorized actions and access to restricted areas.
- **Firmware Reprogramming**: An attacker installs malicious firmware on an IoT device to alter its behavior or bypass security mechanisms.
- **Sensor Data Manipulation**: An attacker alters sensor data in IoT environments, resulting in incorrect readings and potentially dangerous decisions.
- **Side-channel Attacks**: An attacker exploits side-channel information (e.g., power consumption, electromagnetic emissions) to extract sensitive data from IoT devices.

Table 1.2: Impact of Cyberattacks on CIA Triad

Cyberattack	Confidentiality	Integrity	Availability	Affected IoT Layer	Countermeasure	Example

DDoS	Low	Low	High	Network Layer	Rate limiting, anomaly detection, access control lists	Mirai Botnet Attack
MitM	High	High	Medium	Communication/Network Layer	Encryption, mutual authentication, Secure protocols	Session Hijacking
Sybil Attack	Medium	High	Medium	Application/Network Layer	Identity validation, trust management, Sybil detection algorithm	Blockchain consensus tampering
Data Breaches	High	High	Low	Application Layer	Data encryption, access control policies, secure storage	Stolen credentials leading to data exposure
Replay Attack	Medium	High	Medium	Communication Layer	Time-stamp mechanism, nonce-based authentication	Replaying authentication tokens to gain access
Eavesdropping	High	Low	Low	Communication Layer	Encryption, network segmentation, intrusion detection	Wireless sniffing using rogue access points
Jamming Attack	Low	Low	High	Physical Layer	Frequency hopping, spread spectrum communication, signal shielding	Jamming radio frequencies in wireless networks
Physical Tampering	Medium	High	Medium	Device/Physical Layer	Tamper-resistant hardware, physical security, sensor monitoring	Unauthorized access to IoT devices
Firmware Manipulation	High	High	High	Device Layer	Secure firmware updates, integrity checks, digital signature	Modifying firmware to introduce backdoors
Routing Attack	Medium	High	High	Network Layer	Secure routing protocols, trust based mechanism, monitoring	Routing table manipulation in IoT networks
Malware Infections	High	High	Medium	Device/Network Layer	Anti-Malware solutions, sandboxing, IDS/IPS	IoT devices infected with malware
Privilege Escalation	High	High	Medium	Device/Application Layer	Role-based access control (RBAC), vulnerability patching	Gaining admin privileges through unpatched firmware
Firmware Reprogramming	High	High	Medium	Device Layer	Secure boot, cryptographic checks, firmware signing	Installing malicious firmware to modify behavior
Sensor Data Manipulation	Medium	High	Medium	Sensing Layer	Data validation, sensor fusion, outlier detection	Manipulating sensor readings

Side-channel Attacks	High	Medium	Low	Device/Physical Layer	Power analysis mitigation, shielding, constant-time algorithms	Extracting Cryptographic keys through power analysis

1.4. Security and Privacy Issues in IoT

IoT security and privacy issues are primarily related to the heterogeneous nature of devices, lack of standardized protocols, and constrained computational resources. These issues can be categorized as follows:

- **Data Confidentiality**: Ensuring that sensitive information is only accessible to authorized entities. IoT devices often lack robust encryption mechanisms, making them vulnerable to data breaches.
- **Data Integrity**: Protecting data from unauthorized modifications. Data tampering can occur during device transmission, leading to incorrect or misleading information.
- **Authentication and Access Control**: Verifying the identity of devices and users before granting access to resources. Weak authentication mechanisms can lead to unauthorized access and control of IoT devices.
- **Data Privacy**: Protecting personal and sensitive information from being disclosed. Privacy concerns arise due to IoT devices' extensive data collection capabilities, which may include personal health information, location data, and usage patterns.
- **Secure Communication**: Ensuring data transmitted between IoT devices and the cloud is secure and tamper-proof. Using lightweight encryption protocols and secure communication channels is crucial to maintaining data integrity and confidentiality.

This research proposes a comprehensive security framework that leverages blockchain and AI technologies to address these challenges effectively, providing enhanced security and privacy in IoT networks.

1.5. Background on Blockchain

1.5.1. Overview of Blockchain Technology

Blockchain technology is a decentralized, distributed ledger system that records transactions securely and transparently across a network of nodes. It was first conceptualized as the underlying technology for Bitcoin in 2008. However, its potential applications have since expanded to include various industries such as finance, healthcare, supply chain management, and the Internet of Things (IoT). A blockchain consists of a series of blocks, each containing a list of transactions cryptographically linked to the previous block. This ensures data integrity and makes the blockchain resistant to unauthorized modifications.

Each block contains a cryptographic hash of the previous block, a timestamp, and transaction data. This sequential structure forms a "chain" of blocks, where each block depends on the previous block's information. This design makes it computationally infeasible to alter a recorded transaction without modifying all subsequent blocks, ensuring data immutability and security. Blockchain operates in a peer-to-peer (P2P) network where each node maintains a copy of the entire ledger, and consensus protocols such as Proof of Work (PoW) and Proof of Stake (PoS) are used to validate and add new transactions to the blockchain.

1.5.2. Blockchain Characteristics and Features

Blockchain technology possesses several distinctive features that make it an effective solution for enhancing security and privacy in various applications, particularly in IoT environments:

1. **Immutability**: Once data is recorded in a blockchain, it is cryptographically secured and cannot be altered without the consensus of most nodes in the network. This immutability ensures that the integrity of the data remains intact over time.
2. **Decentralization**: Blockchain eliminates the need for a central authority or intermediary by distributing data and control across all participating nodes. This decentralized nature enhances security by reducing the risk of single points of failure.

3. **Transparency**: Every transaction on the blockchain is visible to all nodes, promoting transparency and accountability. This feature is particularly beneficial in multi-stakeholder environments, such as supply chains, where data integrity and trust are critical.
4. **Traceability**: Blockchain provides a complete audit trail of all transactions, enabling data traceability from its origin to its current state. This feature is crucial for healthcare and supply chain management applications, where tracking the provenance of data and products is essential.
5. **Consensus Mechanisms**: Blockchain employs consensus protocols, such as PoW, PoS, and Practical Byzantine Fault Tolerance (PBFT), to achieve agreement on the validity of transactions across the network. These mechanisms ensure that only valid transactions are recorded in the ledger, preventing fraudulent activities.

1.5.3. Types of Blockchain

Blockchain technology can be broadly classified into three main categories based on its architecture, governance model, and level of access: Public Blockchain, Private Blockchain, and Consortium or Federated Blockchain. Each type of blockchain offers unique characteristics and serves different use cases depending on the level of decentralization, transparency, and control required. Additionally, blockchain networks can be categorized as Permissionless or Permissioned based on their access control mechanisms.

1.5.3.1 Public Blockchain

A public blockchain is a decentralized network where anyone can participate, read, and write transactions. It is characterized by its openness and permissionless nature, meaning any user can join the network without prior approval. Participants in a public blockchain can freely interact with the network, participate in the consensus process, and validate transactions.

Characteristics:

- Permissionless Access: Anyone can join the network and participate in the consensus process.
- Decentralized Governance: All participants make decisions collectively, with no single entity having control over the network.
- High Transparency: All transaction records are visible to the public, ensuring high transparency.
- Immutability and Security: Transactions, once recorded on the blockchain, cannot be altered or deleted, making the network resistant to tampering.
- Consensus Mechanisms: Public blockchains typically utilize consensus mechanisms such as Proof of Work (PoW) or Proof of Stake (PoS) to achieve consensus and validate transactions.

Use Cases:

- Cryptocurrencies such as Bitcoin and Ethereum.
- Decentralized applications (DApps) that require high transparency and trustlessness.
- Public ledgers for asset tracking, digital identity management, and open financial systems.

Limitations:

- Scalability and Performance: Public blockchains often face scalability and transaction throughput issues due to the high number of participants and complex consensus mechanisms.
- Energy Consumption: Mechanisms like PoW require significant computational resources, making them energy-intensive.

1.5.3.2 Private Blockchain

A private blockchain, also known as a permissioned blockchain, operates within a closed network where only authorized participants can access and interact with the blockchain. It is governed by a central authority or organization that controls who can join the network and what actions they can perform.

Characteristics:

- Permissioned Access: Participants are pre-approved and must meet specific criteria to join the network.
- Centralized Control: A central authority or consortium of organizations governs the network and enforces policies.
- Enhanced Privacy: Since the network is restricted, private blockchains offer greater transaction privacy and confidentiality.
- Consensus Mechanisms: Private blockchains often use consensus mechanisms like Practical Byzantine Fault Tolerance (PBFT), Raft, or Delegated Proof of Stake (DPoS), which are more efficient and require less computational power than PoW or PoS.

Use Cases:

- Enterprise applications include supply chain management, asset tracking, and inter-organizational collaborations.
- Financial services where data privacy and regulatory compliance are critical.
- Healthcare systems that require secure and private data sharing among trusted parties.

Limitations:

- Reduced Decentralization: The reliance on a central authority reduces the level of decentralization and can lead to single points of failure.
- Limited Transparency: Since access is restricted, transparency is lower compared to public blockchains.

1.5.3.3 Consortium or Federated Blockchain

A federated consortium blockchain is a hybrid model where a group of pre-selected organizations collaboratively manage the blockchain network. It balances the complete decentralization of public blockchains and the restricted access of private blockchains. In a consortium blockchain, multiple entities maintain the network and validate transactions.

Characteristics:

- Partial Decentralization: The network is decentralized to the extent that no single entity has complete control; however, only a selected group of participants can validate transactions.
- Collaborative Governance: The member organizations make decisions through mutual agreement, promoting shared governance and collaboration.
- Higher Scalability: Due to the limited number of participants, consortium blockchains can achieve higher scalability and faster transaction processing than public blockchains.
- Consensus Mechanisms: Consortium blockchains often utilize consensus mechanisms like PBFT or Voting-based algorithms, which are more efficient in speed and resource consumption.

Use Cases:

- Supply chain management, where multiple organizations (e.g., manufacturers, suppliers, and logistics providers) collaborate to ensure the traceability and authenticity of products.
- Financial consortia for cross-border payments and inter-bank settlements.
- Multi-organization collaborations in healthcare, where patient data must be securely shared among trusted parties.

Limitations:

- Complex Governance Structure: Managing a consortium blockchain can be challenging due to the need for mutual agreement and coordination among multiple parties.
- Lower Transparency: Like private blockchains, consortium blockchains have restricted access, which can limit transparency.

Parameter	Public Blockchain	Private Blockchain	Consortium (Federated Blockchain)
Access Control	Permissionless: Open to anyone	Permissioned: Restricted to authorized participants	Permissioned: Restricted to selected group of organization
Decentralization	High	Low	Partial decentralized among consortium members
Governance	Decentralized: Governed by the community	Centralized controlled by a single organization	Collaborative governance among member organizations
Consensus Mechanism	Proof of Work(PoW), Proof of Stake (PoS), Proof of Authority (PoA)	Practical Byzantine Fault Tolerance (PBFT), Raft, DPoS	PBFT, voting-based consensus mechanisms
Transaction Speed	Lower transaction speed due to public consensus	Higher transaction speed due to fewer participants	Higher transaction speed due to limited participants
Scalability	Limited scalability	High scalability within private environments	Medium scalability (better than public, but lower than private)
Transparency	High Transparency: Anyone can view transactions	Low transparency: Limited to authorized participants	Medium transparency based on organizational policies
Security	High security due to decentralized nature	Lower security if central authority is compromised	Medium security with shared responsibility among members
Privacy	Low privacy: Data is visible to all	High privacy: Data is accessible to authorized participants only	Medium privacy: Visible to member organizations
Immutability	High immutability: Once recorded, data cannot be altered	Medium immutability can be modified with authority approval	Medium immutability: shared consensus to modify data
Resource Requirements	High computational resources required for consensus	Lower resources required due to centralized control	Moderate resources required based on the number of participants
Trust Model	Trustless: No need for pre-established trust	Trusted: Based on authority or central organization	Partially trusted: Mutual trust among organizations
Typical Use Cases	Cryptocurrencies, public records, DApps	Enterprise applications, internal data sharing, financial services	Supply chain management, cross-border payments, healthcare collaboration
Consensus Finality	Probabilistic finality	Deterministic finality	Deterministic finality based on consortium rules
Energy Consumption	High (PoW), Moderate (PoS)	Low	Low to moderate based on consensus mechanism used
Examples	Bitcoin, Ethereum, Polkadot	Hyperledger Fabric, R3 Corda, Quorum	IBM Food Trust, TradeLens(Maersk-IBM), B3i Insurance

1.5.3.4 Permissioned vs. Permissionless Blockchains

In addition, the primary categories of blockchains can also be classified as Permissioned or Permissionless based on the access control mechanisms and governance models.

A. Permissionless Blockchains

A permissionless blockchain, also known as a public blockchain, is an open and decentralized ledger where any participant can join and participate in the network without prior approval. These blockchains are characterized by their high degree of transparency and censorship resistance. Well-known examples of permissionless blockchains include Bitcoin and Ethereum.

- **Characteristics:**
- Decentralized and open to all participants.
- High level of transparency and trustless consensus.
- Utilizes consensus mechanisms like Proof of Work (PoW) or Proof of Stake (PoS).

- **Suitability for IoT:**

Permissionless blockchains are less suitable for resource-constrained IoT devices due to the high computational and energy costs associated with consensus mechanisms like PoW. However, they can be used in IoT environments that require high transparency and public verifiability, such as supply chain management or public health monitoring.

- **Use Cases in IoT:**
- Decentralized Device Authentication: Public blockchains can be used to create a decentralized identity management system for IoT devices, ensuring that only authenticated devices can interact with each other.
- Data Provenance: IoT devices can use permissionless blockchains to record provenance, providing transparent and tamper-proof sensor data with valuable history for applications like food safety and environmental monitoring.

B. Permissioned Blockchains

A permissioned blockchain is a private or consortium blockchain restricting access to specific participants. Only authorized nodes can join the network and participate in consensus and data validation. Permissioned blockchains offer more control over network access and provide greater flexibility for implementing customized governance and access control policies.

- **Characteristics:**
- Restricted access, where participants must obtain permission to join.
- Lower computational costs and faster transaction processing compared to permissionless blockchains.
- Enhanced privacy and control over data sharing.

- **Suitability for IoT:**

Permissioned blockchains are better suited for IoT environments, as they can be tailored to the requirements of the specific use case, such as managing a network of smart devices within a factory or healthcare setting. They provide a secure environment for sensitive applications where data confidentiality and restricted access are critical.

- **Use Cases in IoT:**
- Smart Healthcare Systems: A permissioned blockchain can securely store and share patient data between authorized medical devices, healthcare providers, and insurance companies, ensuring data privacy and compliance with regulations like HIPAA.
- Industrial IoT (IIoT): In manufacturing environments, a permissioned blockchain can manage access to critical control systems, monitor device behavior, and enforce security policies across a distributed network of sensors and actuators.

Parameter	Permissioned Blockchain	Permissionless Blockchain
Access Control	Restricted access: Only authorized participants	Open access: Anyone can join the network
Consensus Mechanism	PBFT, Raft, DPoS	PoW, PoS, PoA
Decentralization	Low to medium: controlled by a central authority or consortium	High: fully decentralized without central control
Governance	Centralized or consortium-based governance	Community-driven governance
Transaction Speed	High speed: Fewer nodes and participants involved in consensus	Lower speed: High number of participants impacts transaction speed
Scalability	High: can be scaled to meet specific enterprise needs	Low to medium: Limited by consensus mechanisms like PoW
Security	Lower security if central authority is compromised	High security due to decentralized consensus
Transparency	Limited transparency: Depends on policies and access permissions	High transparency: All transaction records are publicly visible
Privacy	High privacy: Data is accessible only to authorized participants	Low privacy: Data is visible to all participants
Immutability	Medium immutability: controlled by authority	High immutability: Once recorded, cannot be altered
Trust Model	Trusted model: based on known participants	Trustless model: No need for pre-established trust
Energy Consumption	Low energy consumption due to efficient consensus	High energy consumption (e.g., PoW in Bitcoin)
Use Cases	Enterprise applications, healthcare, financial services	Cryptocurrencies, decentralized applications (DApps)
Governance Model	Centralized or shared among consortium members	Decentralized community governance
Examples	Hyperledger Fabric, Quoru, R3 Corda	Bitcoin, Ethereum, Polkadot

1.5.4. Application of Blockchain Technology

Blockchain technology has found applications across various sectors due to its security, transparency, and traceability features. Some prominent applications include:

- **Finance and Banking:** Blockchain is widely used in the finance sector for creating digital currencies (e.g., Bitcoin), enabling secure and transparent cross-border payments, and implementing smart contracts for automated financial transactions. It reduces the need for intermediaries and enhances the speed and security of transactions.
- **Healthcare:** In healthcare, blockchain securely stores and shares patient health records, ensuring data privacy and integrity. It also facilitates the traceability of drugs and medical supplies, reducing the risk of counterfeit products.
- **Supply Chain Management:** Blockchain provides end-to-end visibility of the supply chain by recording every transaction from the origin to the delivery of products. It enables real-time tracking and verification of goods, reducing fraud and enhancing the efficiency of logistics operations.
- **Identity Management:** Blockchain creates decentralized identity management systems that enable users to control their digital identities securely. This prevents identity theft and unauthorized access to sensitive information.
- **Energy and Utilities:** Blockchain is used in energy markets to facilitate peer-to-peer energy trading, optimize grid management, and support renewable energy initiatives. It provides a transparent and secure platform for tracking energy production and consumption.

- **Internet of Things (IoT):** Blockchain addresses critical challenges in IoT, such as secure data exchange, device authentication, and trust management. It enables the creation of decentralized IoT networks where devices can interact autonomously and securely without relying on a central authority.

1.5.5. Blockchain in IoT security

The integration of blockchain technology into IoT environments offers significant advantages in overcoming traditional security challenges. IoT networks are inherently vulnerable due to their heterogeneous nature, limited computational resources, and many connected devices. Conventional security frameworks often struggle to protect adequately against threats such as data breaches, unauthorized access, and distributed denial-of-service (DDoS) attacks. With its decentralized and cryptographic foundation, blockchain offers a promising solution to mitigate these challenges and enhance overall security.

- **Secure Data Exchange:** Blockchain facilitates secure data exchange among IoT devices by providing a decentralized platform for cryptographically signing and verifying data transactions. This prevents unauthorized entities from intercepting or tampering with the data, ensuring its integrity and confidentiality during transmission.
- **Authentication and Authorization:** Blockchain can establish a robust authentication and authorization framework in IoT systems. Using smart contracts, IoT devices can securely authenticate themselves and negotiate access control policies in a decentralized manner. This eliminates the reliance on centralized authentication servers and reduces the risk of single points of failure.
- **Trust Management:** Establishing trust among IoT devices and entities is critical, especially when multiple stakeholders are involved. Blockchain enables trust management through decentralized identity verification and consensus-based validation of transactions. Trust scores or reputations can be maintained on the blockchain, allowing devices to assess peers' trustworthiness dynamically.
- **Mitigation of Single Points of Failure:** In conventional IoT architectures, a compromised central server can disrupt the entire network. Blockchain's decentralized architecture distributes data and controls across all participating nodes, making it difficult for attackers to disrupt the network by targeting a single point.
- **Data Provenance and Traceability:** Blockchain's inherent traceability ensures that the origin and history of data can be verified, making it easier to detect and mitigate data manipulation attacks. This is particularly beneficial in use cases such as supply chain management and industrial IoT, where tracking the authenticity of products and their movements is critical.

1.5.6. Blockchain for Enhancing in IoT

Blockchain technology can significantly enhance the design and effectiveness of Intrusion Detection Systems (IDS) in IoT environments. Traditional IDS frameworks often rely on centralized architectures, prone to single points of failure, bottlenecks, and limited scalability. Blockchain’s decentralized and tamper-resistant architecture addresses these limitations by enabling the development of a more robust and secure IDS framework.

- **Decentralized Data Storage:** Blockchain can be used to store IDS logs and alerts in a decentralized manner, ensuring that they are immutable and accessible to authorized nodes. This decentralized storage prevents attackers from tampering with or deleting IDS records, preserving the integrity of the detection system.
- **Collaborative Intrusion Detection:** Blockchain enables multiple IDS nodes to collaborate and share threat intelligence securely and transparently. This collaboration allows for a broader perspective on network activities and enables faster and more accurate detection of sophisticated attacks that may span multiple devices and network segments.
- **Smart Contracts for Automated Response:** Smart contracts can be programmed to execute predefined actions in response to specific intrusion events. For example, suppose an IDS node detects suspicious activity. In that case, a smart contract can trigger the automatic isolation of the affected device from the network, thereby minimizing the impact of the intrusion.
- **Enhanced Trust and Accountability:** Using blockchain, IDS nodes can establish a decentralized trust management system where the network collectively evaluates the validity and reputation of detection results. This reduces the risk of false positives and negatives, as the detection results are subject to peer validation.

In conclusion, blockchain's unique characteristics of immutability, decentralization, transparency, and security make it an ideal candidate for enhancing the performance and resilience of IDS frameworks in IoT environments. Integrating blockchain with IDS can address critical challenges such as secure data storage, collaborative threat detection, and automated response, paving the way for more secure and reliable IoT networks.

1.6. Research Objectives

- **RO1.** To conduct a Comprehensive Literature Review of existing work on Intrusion Detection System in IoT environment.
- **RO2.** To develop a robust model(s) for an Intrusion Detection System for detecting anomaly behavior using Artificial Intelligence.
- **RO3.** To develop a Blockchain-based framework(s) to enhance the security and privacy issues in the Intrusion Detection System.
- **RO4.** To perform a Comparative analysis of the proposed work with the state of-art-work.

1.7. Thesis Organization

This thesis is structured into eight chapters, each addressing a critical aspect of Intrusion Detection Systems (IDS) in IoT environments, incorporating Artificial Intelligence (AI) and blockchain technologies. The organization follows a logical progression, beginning with the fundamental background and problem statement, followed by an in-depth literature review to identify research gaps. Subsequent chapters detail the development of AI-driven IDS models, the integration of Explainable AI (XAI), and the implementation of blockchain-based frameworks for enhanced security and privacy. The thesis further explores privacy-preserving data sharing in IoT healthcare, conducts a comparative analysis with existing IDS solutions, and concludes with key findings, future research directions, and societal implications. This structured approach ensures a comprehensive understanding of the advancements in IDS for IoT security.

1. Chapter 1: Introduction

This chapter will introduce the background, problem statement, research objectives, and the significance of developing Intrusion Detection Systems (IDS) in IoT using AI and blockchain technology.

2. Chapter 2: Literature Review

This chapter will provide a comprehensive review of the existing work on IDS in IoT environments, highlighting existing gaps, challenges, and recent advancements.

3. Chapter 3: Development of Artificial Intelligence-based Intrusion Detection Models

This chapter will present the research and development of robust AI-based models for detecting anomalies in IoT networks, including the developed models' techniques, algorithms, and performance.

4. Chapter 4: Design and Implementation of Explainable AI for Intrusion Detection

This chapter will focus on developing and integrating Explainable AI (XAI) in the IDS framework, explaining the methods used to make AI decisions interpretable and trustworthy.

5. Chapter 5: Blockchain-Based Frameworks for Enhancing Security and Privacy in Intrusion Detection Systems

This chapter will discuss the design and implementation of a blockchain-based framework to address the security and privacy challenges in IoT IDS, with details on consensus mechanisms, cryptographic techniques, and privacy-preserving measures.

6. Chapter 6: Privacy-Preserving Data Sharing in Blockchain-Enabled IoT Healthcare Management System

This paper introduces a novel decentralized application that uses blockchain technology to enhance medical certificate management security, privacy, and efficiency in the healthcare sector as application.

7. Chapter 7: Comparative Analysis with State-of-the-Art Intrusion Detection Systems

This chapter will perform a thorough comparative analysis of the proposed AI and blockchain-based IDS models with other state-of-the-art techniques, covering metrics such as accuracy, performance, and security enhancements.

8. Chapter 8: Conclusion, Future Work and Societal Applications

This chapter will summarize the research's key findings, contributions, and limitations. It will also suggest directions for future work in IDS development for IoT environments.

Chapter 2: Literature Review

The rapid expansion of the Internet of Things (IoT) has introduced significant security challenges, necessitating robust intrusion detection mechanisms. This chapter provides a comprehensive review of existing literature on artificial intelligence-based intrusion detection systems, blockchain-based security solutions, feature selection techniques, and publicly available datasets. Additionally, key research gaps are identified, highlighting the need for a more efficient and secure framework. The chapter concludes with an overview of performance evaluation metrics and a detailed description of the dataset used in this research.

2.1. Artificial Intelligence-based Intrusion Detection System

Artificial intelligence (AI) has transformed intrusion detection by enhancing threat detection accuracy and adaptability. This section explores AI-driven IDS approaches, focusing on machine learning and deep learning techniques for securing IoT networks.

2.1.1 Machine Learning Based IDS

Machine learning (ML)-based IDS leverages classification, clustering, and anomaly detection techniques to identify cyber threats in IoT environments. This subsection reviews common ML algorithms and their effectiveness in intrusion detection.

Intrusion Detection Systems (IDS) in IoT environments have witnessed a surge of research endeavors aimed at fortifying the security posture of interconnected devices. This literature review critically examines pivotal contributions in this domain, elucidating the evolving landscape of IDS for IoT and highlighting innovative approaches to address its inherent challenges, as shown in Table 1.

Talukder et al. [22] introduced MLSTL-WSN, a novel IDS leveraging machine learning (ML) techniques in Wireless Sensor Networks (WSNs). By employing SMOTE Tomek to address class imbalance, their methodology demonstrates enhanced detection accuracy and robustness, addressing a crucial concern in IoT deployments. Alqahtani et al. [23] explored cyber intrusion detection utilizing machine learning classification techniques. While not explicitly IoT-focused, their insights into machine learning algorithms' efficacy lay foundational groundwork for IDS in IoT ecosystems, underscoring the importance of leveraging advanced computational methods for threat detection. Meryem and Ouahidi [24] proposed a hybrid IDS integrating machine learning algorithms, catering to the intricacies of modern cyber threats. Their approach showcases the synergistic potential of combining multiple detection mechanisms, vital for combating sophisticated intrusion attempts targeting IoT infrastructures. Asif et al. [25] devised a MapReduce-based intelligent model for intrusion detection, leveraging machine learning in IoT environments. Their work exemplifies the integration of distributed computing paradigms with machine learning techniques to address scalability challenges in large-scale IoT deployments. Gad et al. [26] delved into IDS for Vehicular Ad Hoc Networks (VANETs), employing machine learning on the ToN-IoT dataset. Their research underscores the importance of tailored intrusion detection mechanisms for specific IoT applications, emphasizing the need for context-aware security solutions.

Bangui et al. [27] proposed a hybrid machine-learning model for intrusion detection in VANETs, highlighting the significance of adaptability and resilience in vehicular IoT environments. Their approach showcases the efficacy of combining diverse machine-learning techniques to enhance detection accuracy amidst dynamic network conditions. Alhajjar et al. [28] investigated adversarial machine learning in network IDS, shedding light on the emerging threat landscape of sophisticated attacks. Their research underscores the importance of incorporating adversarial robustness into IDS frameworks to mitigate evolving cyber threats targeting IoT infrastructures. Sarhan et al. [29] focused on feature extraction for machine learning-based IDS in IoT networks, addressing the challenge of extracting relevant features from heterogeneous IoT data sources. Their work lays the groundwork for developing context-aware intrusion detection mechanisms tailored to IoT environments.

Liu et al. [30] proposed an intrusion detection approach for imbalanced network traffic, utilizing machine learning and deep learning techniques. Their research emphasizes the importance of addressing the class imbalance in IoT datasets to prevent detection biases and ensure comprehensive threat coverage. Singh et al. [31] introduced AutoML-ID, an automated machine-learning model for intrusion detection in Wireless Sensor Networks (WSNs). Their

methodology streamlines the model development process, offering a scalable solution for deploying IDS in resource-constrained IoT environments. Zou et al. [32] presented HC-DTTSVM, a novel intrusion detection method based on decision tree twin support vector machine and hierarchical clustering. Their approach showcases the potential of hybrid machine learning techniques in enhancing detection accuracy and scalability in IoT environments. Louk and Tama [33] proposed Dual-IDS, a bagging-based gradient-boosting decision tree model for network anomaly intrusion detection. Their research underscores the importance of ensemble learning techniques in enhancing detection robustness and resilience against evolving cyber threats. Mohiuddin et al. [34] explored hybridized meta-heuristic techniques for intrusion detection, integrating the Weighted XGBoost Classifier. Their approach demonstrates the efficacy of meta-heuristic optimization in enhancing the performance of machine learning-based IDS in IoT environments. Zouhri et al. [35] evaluated the impact of filter-based feature selection in intrusion detection systems, highlighting the importance of feature engineering in enhancing detection accuracy and reducing computational overhead in IoT deployments. Amaouche et al. [36] proposed IDS-XGbFS, an intelligent intrusion detection system utilizing XGBoost with a recent feature selection for VANET safety. Their methodology showcases the integration of advanced machine learning algorithms with feature selection techniques tailored to IoT-specific applications.

Table 1: A summary of Intrusion Detection System based on Machine Learning (ML) Techniques

References	Purpose	Methodology	Dataset used	Feature extraction technique	Result	Advantages	Disadvantages
[22]	Intrusion detection system for WSN	DT,RF, MLP, KNN, XGB, LGB	Wireless Sensor Network dataset	SMOTE_Tomek link	Acc= 99.70%	Addresses imbalanced data	Limited to WSNs, requires investigation on other network types
[23]	Intrusion detection system for Cyber security	Bayesian Network, NB, DT, RF, ANN	KDD-99	-	Acc= 94%	Compares multiple algorithms, offers flexibility	Relies on the unspecified dataset, limits the generalizability
[24]	Hybrid Intrusion detection system	KNN, NB, SVM, Logistic Regression	NSL-KDD	-	Acc= 98.77%	Lacks details on the specific hybrid approach	Requires more information on the hybrid method
[25]	Intrusion detection system for intelligent modeling	Map reduced-based intelligent model-IDS	Kaggle ML repository	-	Acc= 97.6%	Efficient for large datasets, scalable	Relies on unspecified dataset, limited details on the model
[26]	Intrusion Detection System for Vehicular Adhoc Networks	LR, NB, KNN, DT, Adaboost, Xgboost, RF, SVM	TON_IOT	Chi-Square and SMOTE	Acc= 99.1%	Focuses on VANETs, ToN-IoT specific	Limited applicability outside VANETs
[27]	Hybrid model Intrusion detection in VANET	SVM, Bayesian coresets, CNN, MLP, RF, Weighted-KNN	CIC-IDS-2017	Weighted clustering	Acc= 96.93%	Offers potentially better accuracy	Requires more information on the specific hybrid model
[28]	Network-based Intrusion detection system	Generative advertised network	NSL-KDD, UNSW-NB-15	PSO, GA	Acc= 99%	Improves IDS robustness against adversarial attacks	Enhances security, potentially computationally expensive
[29]	Intrusion detection system in IoT network	DFF, CNN, RNN, DT, LR, NB	UNSW-NB-15, TON-IoT, CIC-IDS-2018	PCA, AE, LDA	Acc= 96.11%	Improves intrusion detection accuracy in IoT	Addresses feature selection for IoT networks, limited details on specific techniques.
[30]	Intrusion detection system on network traffic-based	RF, SVM, Xgboost, LSTM, Alex-net, Mini-VGGnet, DSSTE	CIC-IDS-2018, NSL-KDD	Edited Nearest Neighbor	Acc= 96.99%	Effective for imbalanced network traffic intrusion detection	Handles imbalanced data and explores different techniques
[31]	Intrusion detection system using WSN	SVR, GPR, BDT, Ensemble regression, kernel regression, LR, BO	Synthetically generated simulated dataset	K-barriers	R=0.93	Achieves good accuracy with AutoML	Automates model selection reduces human effort

[32]	Intrusion detection system	HC-DTTWSVM	NSL_KDD, UNSW-NB - 15	Hierarchical clustering	Acc= 85.95%	Offers potentially better accuracy and reduced false positives	Relies on unspecified dataset, requires investigation on generalizability
[33]	Intrusion detection system	GBM, Light GBM, Catboost, Xgboost	NSL_KDD, UNSW-NB - 15, HIKARI-2021	-	Acc= 91.75%	Effective for anomaly detection, leverages ensemble learning	Relies on unspecified dataset, limits the generalizability
[34]	Intrusion detection system	xgboost	UNSW-NB-15, CIC-IDS-2018	Modified wrapper-based whale sine-cosine	Acc= 99%	Combines meta-heuristics for optimization and XGBoost for classification	Relies on unspecified dataset, limited details on meta-heuristics
[35]	Intrusion detection system	MLP, SVM, Xgboost, RF	CIC-IDS-2018, CIC-IDS-2017, TON-IoT	Relieff, Pearson correlation, mutual information, ANOVA, chi-square	Acc= 98%	Identifies the importance of feature selection, improves efficiency	Relies on unspecified IDS method and dataset, limited to filter-based selection
[36]	Intrusion detection system	xgboost	NSL-KDD, 5-routing metrics dataset	Boruta, ADASYN	Acc= 99%	Focuses on VANET security, leverages XGBoost, and feature selection	Limited applicability outside VANETs, relies on unspecified recent feature selection technique

2.1.2 Deep Learning Based IDS

Deep learning (DL)-based IDS utilizes advanced neural networks to detect complex attack patterns with high accuracy. This subsection examines various DL architectures and their application in IoT security. This Literature review evaluates recent advancements in Intrusion Detection Systems (IDS), focusing on their applicability to cyber-physical systems (CPS) and Internet of Things (IoT). The literature review highlights various IDS methodologies, datasets, and their applications, identifying critical limitations that inform the development of CPS and IoT. A thorough comparative analysis underscores the challenges faced by existing models in achieving explainability, resilience, scalability, and trustworthiness, key attributes necessary for effective IDS in CPS and IoT environments.

Upon meticulous examination of the most recent and relevant research, we have discerned a cluster of works characterized by shared motivations yet distinguished by unique perspectives. The discussion aims to illuminate these works, furnishing a thorough overview before our proposed methodology exposition. Over the past decade, the effectiveness of Deep Learning (DL) and Machine Learning (ML) methodologies has been prominently demonstrated in the identification of anomalous entities within Internet of Things (IoT) networks. For example, most models lack explainability mechanisms, making it difficult for operators to interpret predictions and take corrective actions. This deficiency is critical in safety-sensitive CPS environments where transparency is vital for building trust. Moreover, the resilience of these models is often inadequate, as they struggle to adapt to evolving threats in dynamic CPS contexts. Scalability also emerges as a pressing issue, with many methods failing to handle the heterogeneity and complexity of modern CPS systems due to computational overhead or rigid architectures. The encapsulation of these findings is succinctly presented in Table 2, providing a comprehensive summary of existing literature that not only converges on similar motivations but also diverges in their applications.

Intrusion Detection Systems (IDS) have been extensively explored in IoT and IIoT to address evolving cybersecurity threats. For Industry 5.0, hybrid deep learning models, such as Bi-LSTM with Bi-GRU, have demonstrated high accuracy rates of 99% for multiclass classification on datasets like CICDDoS 2019 [17]. Similarly, encoder-CNN models have been used for intrusion detection in IoT-based transportation networks, leveraging feature extraction and classification with moderate explainability [21]. Trustworthiness in IDS frameworks has also been discussed using approaches like differential privacy and federated learning, achieving notable resilience but with limitations in performance scalability [19, 23]. Furthermore, integrating advanced neural networks like ResNet for intrusion detection has shown potential, though challenges in model interpretability and robustness persist [24]. While these

models have made strides in performance, many lack comprehensive capabilities to simultaneously ensure resilience, trustworthiness, and dependability, which are critical for Industry 5.0's stringent requirements.

While [17] achieves slightly higher accuracy (99%) compared to our model (97.46%), it does not fully account for high resilience and dependability in real-time heterogeneous environments, as emphasized in Industry 5.0. Moreover, unlike [21] and [24], which focus on partial model interpretability, our solution integrates advanced Explainable AI (XAI) mechanisms to enhance transparency and trustworthiness, aligning with Industry 5.0 goals. Additionally, while prior works such as [19] and [23] emphasize specific aspects of trustworthiness or resilience, our model adopts a holistic approach by balancing accuracy, trustworthiness, and resilience. This is particularly significant in Industry 5.0, where dependable and trustworthy systems are paramount for seamless cyber-physical integration. Ultimately, our proposed model fills the gap by addressing fundamental limitations in explainability, scalability, and robustness, thereby presenting a comprehensive solution for intrusion detection in critical Industry 5.0 environments.

Table 2: A summary of existing research work on Intrusion Detection Systems on different applications

References and Year	Purpose	Model/ Mechanism	Dataset used	Type of Classification	Result	Explainable AI	Resilient	Dependable	Trustworthy
[16] and 2023	DoS detection in a cyber-physical system	Hybrid Deep learning approach CNN-LSTM	BOT-IOT	Binary classification	<98%	✗	✗	Low	✗
[17] and 2023	Intrusion detection system for Industry 5.0	Hybrid Deep learning approach Bi-lstm + Bi-GRU	CICIDDOS 2019	Multiclass classification	<99%	✓	✓	Low	✗
[18] and 2020	Intrusion detection for Intelligent IoV	Deep CNN	Network traffic dataset	Binary classification	<99%	✗	✗	Partial	✗
[19] and 2020	Trustworthy Framework for Privacy Preserving in IIoT	Differential privacy + Federated Learning CNN	MNIST dataset	-	<90%	✗	✗	High	✓
[20] and 2020	Spam detection in IoT devices for smart home appliances	Bayesian Generalized linear model	REFIT project dataset	Multiclass classification	<84%	✗	✗	Low	✗
[21] and 2022	Intrusion detection in IoT-based transportation network	Encoder + CNN model	UNSW TON-IOT dataset	Both	<90%	✓	✗	Partial	✗
[22] and 2018	Software-defined IIoT for third-party synchronization	Dueling deep Q-learning	-	Not mentioned	Simulation on throughput	✗	✓	Partial	✓
[23] and 2020	Intrusion detection in industry CPS	Deep Fed (Federated learning)	Gas pipelining dataset	Multiclass classification	<98%	✗	✗	Low	✓
[24] and 2022	IDS in the Internet of Things environment	Deep transfer learning model (ResNet)	Heterogeneous IoT sensor dataset	Binary classification	<87%	✗	✓`	High	✗
[25] and 2021	Secure & privacy-preserving framework for IoT-based smart cities	Gradient Boosting Anomaly detection	TON-IOT, BOT-IOT	Multiclass classification	<99%	✗	✗	Partial	✓
[26] and 2022	Botnet attack detection for industrial IoT	DNN-LSTM	N_BaIoT	Multiclass classification	<99%	✗	✗	Partial	✗
[27] and 2023	Network intrusion detection for early identification	Graph2vec + RF	CICIDS 2017, CICIDS 2018	Multiclass classification	<99%	✗	✗	Low	✗
[28] and 2023	Imbalance learning for NIDS in IoT	TMG-GAN	CICIDS 2017, UNSW-NB 15	Multiclass classification	<98%	✗	✗	Partial	✗

[29] and 2022	Intrusion detection system for cloud environment	Conditional denoising adversarial auto-encoder	Cloud IDS dataset	Multiclass classification	<99%	✗	✗	Low	✗
[30] and 2023	Two-level fusion architecture for CPS intrusion detection	TV-DBN-based ensemble-level fusion	-	-	<98%	✗	✗	High	✗
[31] and 2022	Attack detection in industrial IoT	An autoencoder using FL	SCADA system dataset	Multiclass classification	<97%	✗	✗	Low	✓
[41] and 2023	Intrusion detection system for IoT network	Stacked-based ensemble learning model	UNSW-NB 15, N_BaIoT	Binary Classification	<99%	✗	✗	Low	✗
[42] and 2023	Anomaly-based network intrusion detection for IoT	filter-based feature selection Deep Neural Network (DNN) model	UNSW-NB 15	Multiclass Classification	<90%	✗	✗	Low	✗
[43] and 2024	SDN-based intrusion detection for IoT	Bi-LSTM+GRU model	N-BaIoT, CICDDoS 2019	Multiclass Classification	<99%	✗	✗	High	✗
[44] and 2024	Robust DDoS Intrusion Detection System	CNN inception model	CICDDoS 2019	Multiclass Classification	<96%	✗	✗	Low	✗
[45] and 2024	Intrusion detection system for IoT	Optimized Forest (OF) Based Machine Learning	NSW-NB15 and NSLKDD	Binary Classification	<98%	✗	✗	Low	✗
[46] and 2024	Intrusion detection system for Industrial IOT environment	CNN+GRU	N-BaIoT	Multiclass Classification	<99%	✗	✗	High	✗
[47] and 2024	Intrusion detection in IoT networks	2D-CNN	NSL-KDD and UNSW-NB 15	Multiclass Classification	<98%	✓	✗	Partial	✗
[48] and 2023	Intrusion detection system in IoT environment	CNN+BiLSTM	N-BaIoT	Multiclass Classification	<99%	✗	✗	Low	✗
[49] and 2024	Malware detection in Internet of Things	DLEX-IMD	IoT-23	Multiclass Classification	<99%	✓	✗	Partial	✗
[50] and 2024	DDoS detection framework for IoT-enabled mobile health informatics systems	mGRU-based IDS models	CICIoT 2023, CICDDoS 2019	Both Classification	<98%	✗	✗	Low	✗
[51] and 2024	intrusion detection scheme for IoT and IIoT environment	enable Convolutional Neural Network (CNN	Edge_IIoT dataset	Multiclass Classification	<99%	✗	✗	Partial	✗
[52] and 2024	Intrusion Detection in Industrial-Internet of Things	proposed CNN1D model	Edge_IIoT dataset	Multiclass Classification	<99%	✗	✗	Partial	✗
[59] and 2024	IoT botnet detection using XAI	XGB, ET, RF, GBC, LGBM	N-BaIoT, BoT IoT, Med-BIoT	Binary Classigication	<98%	✓	✓	Low	✗

2.2. Blockchain-based Intrusion Detection System for IoT Security

Blockchain technology offers a decentralized and tamper-resistant approach to securing IoT networks. This section discusses blockchain-integrated IDS, highlighting consensus mechanisms and privacy-preserving techniques for enhancing security. Recent research on detecting cyberattacks in IoT and IIoT networks has explored a variety of approaches, each contributing uniquely to the field of intrusion detection systems (IDS). This section synthesizes these studies, highlighting their methodologies and the specific challenges they address, as shown in Table 3.

Gad et al. (2020) [21] introduced an XGBoost-based model for vehicular ad-hoc networks, leveraging the TON-IoT dataset and employing chi-square for feature selection. Their approach, although practical, is confined to a specific type of IoT network. In contrast, Mighan et al. (2021) [22] proposed a scalable IDS that integrates Support Vector Machines (SVM) with Stacked Autoencoder (SAE) to handle big data platforms, using tools like Apache Spark to manage large network traffic volumes. Similarly, Alzahrani et al. (2019) [23] developed a network-based IDS for Software-Defined Networks (SDN), applying machine learning techniques such as Decision Trees, Random Forests, and XGBoost on the NSL-KDD dataset. Logeswari et al. (2020) [24] advanced this by proposing a hybrid feature selection algorithm (HFS-LGBM IDS) to reduce data dimensionality and extract optimal features using CFS and RF-RFE, demonstrating their model's effectiveness in a Mininet-simulated SDN environment.

A notable contribution by Bowen et al. (2021) [25] introduced BlocNet, a deep learning model designed to address dataset imbalance, employing various sampling techniques to maintain data integrity. Their work emphasizes the importance of handling underrepresented instances in IDS datasets. Kasonogo et al. (2022) [26] offered an IDS using different RNN frameworks on NSL-KDD and UNSW-NB-15 datasets, incorporating XGBoost for feature selection and addressing optimization of arbitrary differentiable loss functions. Hnamte et al. (2023) [27] presented a novel approach combining Convolutional Neural Networks (CNN) and Bidirectional Long Short-Term Memory (BiLSTM) networks, enhanced by an attention mechanism, to improve classification accuracy in network-based IDS—however, their model's complexity results in longer training periods than traditional deep learning techniques. Abdelkhalek et al. (2022) [28] addressed class imbalance by proposing a data resampling strategy using the Adaptive Synthetic and Tomek Link algorithm, combined with various deep learning models, including MLP, CNN, DNN, and CNN-BiLSTM, achieving better detection rates for minority classes.

Further advancing the discussion, Thakkar et al. (2023) [29] focused on enhancing DNN-based IDS performance by introducing a unique feature selection technique based on statistical significance. They utilized standard deviation, mean, and median to derive highly discernible features, which improved data learning. Imran et al. (2021) [30] proposed a non-symmetric deep autoencoder for network intrusion detection systems (NIDS) using the KDD-CUP-99 dataset, highlighting the robustness of their model through various metrics. They also critically reviewed existing challenges in NIDS approaches. Benadai et al. (2022) [31] explored the application of deep reinforcement learning (DRL) in IDS, proposing a DRL_IDS model that utilizes the Markov decision process and stochastic game theory to analyze network traffic. Their approach demonstrated improved detection rates and reduced false alarm rates compared to other deep learning methods.

Security challenges necessitate innovative solutions in the context of Cyber-Physical Systems (CPS) and IoT. Mansour et al. (2021) [32] proposed a blockchain-based IDS for CPS environments, integrating a rich and poor optimization approach with a deep learning model. Kumar et al. (2021) [33] addressed the centralized storage architecture's limitations by presenting a blockchain-based IoT framework utilizing fog computing for distributed security. This framework offers a decentralized cloud architecture, mitigating issues like security, privacy, and single points of failure. Ashraf et al. (2022) [34] introduced a federated learning-based IDS for IoT healthcare, leveraging blockchain to train models on different datasets without data sharing, thereby enhancing privacy. However, variations in local datasets and uneven distribution affected network-based intrusion detection accuracy. He et al. (2022) [35] proposed a blockchain-based distributed federated learning approach, providing differential privacy to secure data while enabling collaborative training. Khraisat et al. (2023) [36] developed a feature selection approach based on information gain, focusing on identifying IoT features that yield the most feature diversity in network traffic, emphasizing detecting zero-day attacks with high accuracy.

Table 3: Overview of Existing Frameworks for Addressing Security and Privacy Issues in Intrusion Detection Systems Using Emerging Technologies

Reference	Aim	Dataset used	Methodology	Feature Selection Technique	Paradigm	Types of attack	Scalability Analysis	Security and Privacy Analysis	Single point Failure

Gad et al. (2020)	Detecting cyberattacks in vehicular ad-hoc networks	TON-IoT	XGBoost model	Chi-square	Machine Learning	Specific IoT attacks	Limited to vehicular ad-hoc networks	Not addressed	Not discussed
Mighan et al. (2021)	Scalable IDS for big data platforms	UNB-ISCX- 2012	SVM with Stacked Autoencoder (SAE), Apache Spark		Machine Learning	Various	High scalability with big data platforms	Not addressed	Not discussed
Alzahrani et al. (2019)	IDS for SDN environments	NSL-KDD	Decision Trees, Random Forests, XGBoost		Machine Learning	Various	Adaptable to SDN environments	Not addressed	Not discussed
Logeswari et al. (2020)	Feature selection in IDS for SDN	Mininet-simulated SDN	HFS-LGBM IDS using CFS and RF-RFE	Hybrid Feature Selection (CFS, RF-RFE)	Machine Learning	Various	Demonstrated in SDN environment with Mininet simulation	Not addressed	Not discussed
Bowen et al. (2021)	Addressing dataset imbalance in IDS	NSL-KDD; IoT-23; CIC-IDS; UNSW-NB-15	BlocNet deep learning model, sampling techniques		Deep Learning	Various	Not specified	Focuses on handling underrepresented instances in datasets	Not discussed
Kasonogo et al. (2022)	IDS using RNN frameworks	NSL-KDD, UNSW-NB-15	RNN, XGBoost for feature selection	XGBoost	Deep Learning	Various	Not specified	Not addressed	Not discussed
Hnamte et al. (2023)	Improve classification accuracy in network-based IDS	CIC-IDS 2018; Edge-IIoT	CNN-BiLSTM with attention mechanism	-	Deep Learning	Various	Higher complexity leads to longer training periods	Not addressed	Not discussed
Khraisat et al. (2023)	Feature selection in IDS for IoT	NSL-KDD	Information gain for feature diversity	Information gain	Machine Learning	Zero-day attacks	High accuracy in detecting zero-day attack	Not addressed	Not discussed
He et al. (2022)	Blockchain-based distributed federated learning approach	NSL-KDD; BoT_IoT; CICIDS-2017; UNSW-NB-15; DS2OS dataset	Blockchain with differential privacy	-	Federated Learning, Blockchain	Various	Collaborative training while securing data	Provides differential privacy to secure data	Not discussed
Ashraf et al. (2022)	Federated learning-based IDS for IoT healthcare Federated learning with blockchain	BoT_IoT			Federated Learning, Blockchain	Various	Not specified	Enhances privacy, but affected by dataset variations and distribution	Not discussed
Kumar et al. (2021)	Blockchain-based IoT framework for distributed security	NSL-KDD; CICIDS-2017 dataset	Blockchain with fog computing		Blockchain, Fog Computing	Various	Decentralized cloud architecture	Mitigates security, privacy, and single point failure issues	Addressed by blockchain
Turukmane et al. (2024)	To design an efficient automated intrusion detection system (IDS) using machine learning to address issues such as class imbalance, overfitting, and accurate classification of network intrusions.	CSE-CIC-IDS 2018 and UNSW-NB15 datasets	hybrid multilayer SVM model (M-MultiSVM)	Opposition-based Northern Goshawk Optimization (ONgO)	Machine Learning	DoS attacks, content-based features, and traffic anomalies	does not explicitly address scalability in detail	not discussed	does not provide explicit analysis of single point failure

Nandanwar et.al (2024)	To develop a robust and efficient deep learning-based intrusion detection system (IDS) to detect and classify botnet attacks in Industrial IoT (IIoT) environments, ensuring real-time protection and minimizing security vulnerabilities	N-BaIoT dataset	CNN-GRU-based deep learning model named AttackNet	CNN	Deep Learning	DoS, DDoS, data exfiltration	scalability by achieving high performance across multiple classes in large dataset	indirectly improves security by efficiently detecting botnet attacks	does not provide explicit analysis of single point failure
Karthikeyan et.al (2024)	To enhance security in WSN-IoT systems by developing a machine learning-based intrusion detection system (IDS) optimized with the Firefly Algorithm (FA) and Grey Wolf Optimizer (GWO) for improved accuracy, reliability, and security performance.	NSL-KDD Dataset	FA-ML technique integrates machine learning (SVM)	Firefly Algorithm (FA)	Supervised machine learning	Denial of Service (DoS), Probing, Remote to Local (R2L), and User to Root (U2R) attacks	not explicitly discussed	not discussed	not discussed
Hanafi et. al (2024)	To develop a new intrusion detection system (IDS) for IoT networks using an Improved Binary Golden Jackal Optimization (IBGJO) algorithm and Long Short-Term Memory (LSTM) network	NSL-KDD Dataset and CICIDS 2017 Dataset	Opposition-Based Learning (OBL)-LSTM	Improved Binary Golden Jackal Optimization (IBGJO) with Opposition-Based Learning (OBL)	Deep Learning	DoS (Denial of Service), Probe, U2R (User-to-Root), and R2L (Remote-to-Local) attacks	does not explicitly address	does not explicitly discuss security and privacy concerns	not been explored
Kumar et.al (2024)	To develop an efficient intrusion detection system (IDS) using Deep Residual Convolutional Neural Network (DCRNN), optimized by the Improved Gazelle Optimization Algorithm (IGOA)	UNSW-NB-15 Dataset, Cicddos2019 Dataset, and CIC-IDS-2017 Dataset	DCRNN	Novel Binary Grasshopper Optimization Algorithm (NBGOA)	Deep Learning	detect various types of attacks	demonstrating its ability to effectively scale in real-world scenarios with large datasets	does not explicitly address	does not address the impact of single point failure

2.4. Research Gaps

Despite advancements in IDS and blockchain security, several challenges remain unaddressed. This section identifies existing limitations in current research and highlights the need for innovative solutions in intrusion detection and IoT security. Based on the insights from recent studies, several significant research gaps have been identified within the field of intrusion detection systems (IDS) in IoT environments:

- **Integration of AI and Blockchain:** There is a lack of comprehensive research that combines artificial intelligence (AI) with blockchain technology for enhanced intrusion detection.
- **Scalability and Efficiency:** Effective scaling and optimization of performance for deep learning and blockchain-based IDS remain underexplored.
- **Adversarial Attack Resilience:** Current deep learning-based IDS are vulnerable to adversarial attacks, necessitating strategies to fortify detection mechanisms against such threats.
- **Interoperability and Data Sharing:** While blockchain facilitates decentralized data sharing, research is needed to effectively integrate it with IDS for collaborative threat detection.

- **Privacy-Preserving Mechanisms:** Using sensitive network data in deep learning models raises privacy concerns, highlighting the need for secure data-sharing protocols.
- **Network Heterogeneity:** Deep learning models often struggle with generalization across different network environments, indicating a need for transfer learning techniques to enhance adaptability.
- **Decentralization of Network Information:** AI-based IDS require access to sensitive data, necessitating methods to balance decentralized data access with privacy protection, such as secure multi-party computation and zero-knowledge proofs.
- **Consensus Protocol Validation:** There is a need for research to validate various consensus algorithms' effects on the security and reliability of IDS within blockchain networks, emphasizing trust-building across decentralized nodes.

By addressing these gaps, future research can significantly enhance the effectiveness of IDS in IoT ecosystems.

2.5. Publicly Available Dataset

Publicly available datasets are essential for benchmarking IDS performance. This section reviews commonly used intrusion detection datasets, discussing their characteristics, attack types, and suitability for IoT security research. Table 4 presents a statistical analysis of publicly available datasets for intrusion detection. It outlines dataset descriptions, the number and types of attacks, whether they are IoT-based, the number of features, and if they are labeled. The datasets span from simulated military traffic to real-world IoT network data, providing varied sources for analysis.

Table 4: A statistical analysis of publicly available datasets

Datasets & Year	Description of Datasets	Number of Attacks	Type of Attacks	IoT-Based	Number of Features	Labeled
DARPA 98 (1998)	Simulated military network traffic	4	DOS, Probe, R2L, U2R	No	41	Yes
KDD Cup 99 (1999)	An enhanced version of DARPA 98 with more attacks	39	DOS, Probe, R2L, U2R	No	41	Yes
NSL-KDD (2015)	Improved KDD Cup 99 with reduced redundancy	39	DOS, Probe, R2L, U2R	No	41	Yes
CICIDS 2017	Modern network traffic with benign and malicious activities	15	Backdoors, DoS, Exploits, Fuzzers, Generic, Reconnaissance, Shellcode, Worms	No	85	Yes
UNSW_NB_15 (2015)	Real-world network traffic with various attack types	9	DoS, U2R, R2L, Analysis, Scanning, Backdoors, Generic	No	49	Yes
N_BAIoT (2018)	Simulated IoT network traffic with normal and attack data	10	DDoS, DoS, Injection, MITM, Password, Ransomware, Scanning, Theft	Yes	115	Yes
CICIDS 2019	Expanded CICIDS 2017 with additional attack categories	25	Backdoors, DoS, Exploits, Fuzzers, Generic, Reconnaissance, Shellcode, Worms, Web attacks	No	80	Yes
Bot_IoT (2018)	Network traffic from IoT devices infected with botnets	4	DDoS, DoS, Reconnaissance, Theft	Yes	42	Yes
TON_IoT (2019)	Real-world IoT network traffic with normal and attack data	5	DDoS, DoS, Injection, MITM, Scanning	Yes	45	Yes
Edge_IIoT (2022)	Industrial IoT network traffic with normal and attack data	14	Backdoors, DoS, Injection, MITM, Reconnaissance, Shellcode, Theft, Zero-day	Yes	61	Yes
IoT_23 (2023)	Network traffic from 23 diverse IoT devices	4	DDoS, DoS, Injection, Scanning	Yes	80	Yes

2.5. Performance Evaluation Metrics

Evaluating the efficiency and reliability of an intrusion detection system requires well-defined performance metrics. This section categorizes evaluation metrics into two key domains: AI-based models and blockchain-based security mechanisms.

2.5.1. Artificial Intelligence-Based Model Evaluation Metrics

AI-driven IDS models are assessed based on classification performance and computational efficiency. This subsection discusses accuracy, precision, recall, F1-score, False Positive Rate (FPR), False Negative Rate (FNR), and model execution time as shown in Table 5.

Table 5: Performance Evaluation Metrics for Intrusion Detection System

Metrics	Definition	Formula
Confusion matrix	Used to evaluate the performance of a classification model by comparing the predicted labels with the actual labels.	
True positive (TP)	The record is successfully detected as malicious	
False positive (FP)	The record is wrongly detected as malicious.	
True Negative (TN)	The record is classified as non-malicious.	
False Negative (FN)	The record is undetected by the system.	
Accuracy	Measure how well the model predicts the correct labels.	$Accuracy = \frac{TP + TN}{TP + FP + TN + FN}$
Precision	Measure how many of the predicted positive labels are actually positive.	$Precision = \frac{TP}{TP + FP}$
Recall	Measure how many of the actual positive labels are correctly predicted.	$Recall = \frac{TP}{TP + FN}$
F1 score	The harmonic mean of Recall and Precision.	$F1\ Score = 2 * \frac{Precision * Recall}{Precision + Recall}$
ROC curve	Graphical representation of the performance of a classification model. TPR is the ratio of true positive predictions to the total actual positive labels. FPR is the ratio of false positive predictions to the total actual negative labels.	$TPR = \frac{TP}{TP+FN}$, $FPR = \frac{FP}{FP+FN}$
True Negative Rate (TNR)	Model's ability to correctly classify instances of a specific attack type as non-attacks.	$TNR = \frac{TN}{TN + FP}$
Negative Predictive Value (NPV)	Model's accuracy in predicting non-attacks for a specific attack type.	$NPV = \frac{TN}{TN + FN}$
False Positive Rate (FPR)	The rate at which instances of other attack types are incorrectly classified as the specific attack type.	$FPR = \frac{FP}{FP + TN}$
False Negative Rate (FNR)	The rate at which instances of a specific attack type are incorrectly classified as non-attacks or other types.	$FNR = \frac{FN}{TP + FN}$
False Discovery Rate (FDR)	The rate at which instances are falsely predicted as the specific attack type when they are not.	$FDR = \frac{FP}{FP + TN}$
False Omission Rate (FOR)	The rate at which instances of a specific attack type are falsely classified as non-attacks or other types	$FOR = \frac{FN}{FN + TN}$

2.5.2. Blockchain Based Evaluation Metrics

When evaluating the performance and reliability of the blockchain framework, it is important to consider a comprehensive set of metrics that offer insights into the system's efficiency, security, and user experience. These metrics are instrumental in assessing blockchain performance under varying conditions. This sub-section systematically defines key evaluation parameters, including fault tolerance, transaction finality, and network overhead. By examining these metrics, the strengths and limitations of blockchain Framework across different parameters can be rigorously analyzed, thereby facilitating the development of robust and scalable solutions. The following are the Evaluation Metrics:

- **Fault Tolerance:** Fault tolerance in blockchain refers to the network's capability to continue functioning correctly even when some components fail. This is critical for maintaining system reliability and ensuring the blockchain remains operational despite disruptions. It is measured by Restoration Efficiency (RE), which quantifies how effectively the system recovers from failures:

$$RE = \frac{T_{restored}}{T_{total}}$$

- **User Experience (UX):** User experience in blockchain systems reflects the ease and efficiency with which users interact, focusing on the system's response time and the time required to share records. It is inversely related to the sum of response time ($T_{response}$) and shared record time (T_{share}):

$$UX = \frac{1}{T_{response} + T_{share}}$$

- **Transaction Finality:** Transaction finality measures how quickly a transaction becomes irreversible and permanently recorded on the blockchain. It is directly related to the time required to create a block T_{block}:

$$TF = T_{block}$$

- **Network Overhead (NO):** Network overhead refers to the additional computational resources and time consumed due to managing blockchain transactions, including encryption processes. It is often expressed as a percentage of the system's throughput and is influenced by the encryption time $T_{encrypt}$:

$$NO\% = \frac{T_{encrypt}}{Th} * 100$$

- **Encryption Time:** Encryption time ($T_{encrypt}$) is the duration required to convert plaintext into encrypted data using cryptographic algorithms, impacting security and transmission efficiency:

$$T_{encrypt} = \frac{1}{f}\sum_{i-1}^{f} t_{encrypt,i}$$

- **Decryption Time:** Decryption time ($T_{decrypt}$) is the time taken to convert encrypted data back to its original form, crucial for accessing secured data efficiently:

$$T_{decrypt} = \frac{1}{f}\sum_{i-1}^{f} t_{decrypt,i}$$

- **Key Generation Time:** Key generation time ($T_{key\ Gen\ T}$) is the duration required to create cryptographic keys, affecting overall security and system speed:

$$T_{key\ Gen\ T} = \frac{1}{n}\sum_{i-1}^{n} t_{key\ Gen\ T,i}$$

- **Response Time:** Response time ($T_{response}$) measures the interval between a user request and the system's response, crucial for performance evaluation in time-sensitive applications:

$$T_{response} = T_{response\_end} - T_{request\_start}$$

- **Restoration Efficiency:** Restoration efficiency (RE) quantifies the system's ability to recover from faults, defined as:

$$E_{restoration} = \frac{D_{rest}}{D_{orig}} \times 100\%$$

Where $E_{restoration}$ is the restoration efficiency, D_{rest} is the amount of data successfully restored, and D_{orig} is the original data before any loss or corruption.

- **Shared Record Time:** Shared record time (T_{share}) is the time taken to transmit or share a record within the system, impacting the efficiency of data sharing:

$$T_{share} = T_{send} + T_{verify} + T_{commit}$$

Where T_{share} is the sharing record time, T_{send} is the time to send the record, T_{verify} is the time to verify the record, T_{commit} is the time to commit the record to the blockchain.

- **Block Creation Time:** Block creation time (T_{block}) is the time required to generate a new block in the blockchain, affecting transaction finality and throughput:

$$T_{block} = \frac{1}{N}\sum_{i=1}^{N} t_{block,i}$$

Where T_{block} is the average block creation time, $t_{b,i}$ is the time taken to create the $i-th$ block, and N is the total number of blocks created.

- **Throughput:** Throughput (Th) is the number of transactions processed per second, a key metric for evaluating the scalability and efficiency of blockchain networks:

$$Th = \frac{Total\ transaction}{total\ time\ taken}$$

- **Latency:** Latency (L) is the delay between the initiation and completion of a transaction, crucial for real-time processing:

$$L = T_{completion} - T_{initiation}$$

Chapter 3: Development of Artificial Intelligence-based Intrusion Detection Models

3.1. Introduction

The Internet of Things (IoT) has revolutionized various sectors, enabling seamless connectivity and communication between devices, sensors, and systems. By integrating physical devices with networked data systems, IoT technology facilitates intelligent data collection, analysis, and action, leading to increased automation, efficiency, and functionality in fields ranging from healthcare and agriculture to manufacturing and smart cities. However, the exponential growth of IoT networks has introduced significant security and privacy challenges. IoT devices, often deployed with minimal security features, have become attractive targets for cybercriminals, who exploit their vulnerabilities to launch botnet and other network-based attacks. These attacks compromise the functionality of individual devices and pose a broader risk to the integrity of entire networks, necessitating sophisticated mechanisms for threat detection and mitigation.

Intrusion Detection Systems (IDS) have been widely adopted to address these concerns as essential components of IoT security infrastructure. IDS solutions monitor network traffic and system activities, identifying and classifying potentially malicious behavior to mitigate risks. Traditional IDS approaches, however, struggle to effectively detect sophisticated and rapidly evolving attacks like IoT botnets, which exploit device heterogeneity, resource constraints, and the lack of standardization across IoT networks. As a result, Artificial Intelligence (AI)-driven IDS models, incorporating machine learning and deep learning techniques, have emerged as powerful alternatives. These models leverage large datasets to autonomously learn, identify patterns, and adapt to new threats, making them particularly well-suited for detecting botnet attacks and network anomalies in IoT environments.

Despite these advancements, implementing effective IDS solutions within IoT environments presents unique challenges. The vast number of devices and high diversity in IoT networks make it difficult to develop standardized detection protocols. Furthermore, the presence of resource-constrained devices limits the computational complexity that can be applied in detection models, necessitating lightweight and efficient algorithms. Additionally, IoT networks are highly dynamic, with frequent device additions and deletions, creating a need for IDS models that can adapt in real-time. Addressing these challenges requires AI-based models that are accurate but also scalable, flexible, and capable of handling high-dimensional IoT data.

This chapter presents three proposed models to address these challenges within IoT and IIoT environments. Each model uses unique AI architectures and methodologies to improve detection accuracy, scalability, and efficiency:

1. **Transfer Learning-Enabled Hybrid Model (TL-BILSTM IoT)**:
The first model utilizes transfer learning in a hybrid approach, combining Convolutional Neural Networks (CNN) and Bidirectional Long Short-Term Memory (BiLSTM) networks. This model is designed to efficiently and effectively identify botnet attacks, explicitly targeting BASHLITE and Mirai botnet attacks. By incorporating transfer learning, the model enhances its ability to recognize and classify various types of network traffic, reducing the computational cost typically associated with training on large-scale datasets. This adaptive model is evaluated using the "Detection of IoT Botnet Attacks N_BaIoT" dataset. It showcases its effectiveness in a multi-class classification setup, distinguishing benign from malicious traffic across multiple attack types.

2. **Deep Learning-Enabled Intrusion Detection System for Industrial IoT**:
The second proposed model is tailored specifically for the Industrial IoT (IIoT) context, where networked industrial devices often exhibit unique data patterns and security requirements. This model introduces a deep learning framework that combines CNN and Gated Recurrent Units (GRU) to detect anomalies generated by compromised IIoT devices. It focuses on robust feature extraction and optimization techniques, which enable the model to handle high-dimensional IIoT data while eliminating redundant features efficiently. This model's adaptive CNN-GRU architecture is highly effective in identifying network-based attacks, distinguishing various types of IoT botnet attacks, and is validated using the same N_BaIoT dataset, achieving superior performance in detecting IIoT-specific threats.

3. **Dependable and Trustworthy CNN-GRU-Based IDS (Alpha-Net)**:
The third model, Alpha-Net, represents a trustworthy and dependable IDS solution designed explicitly for IIoT environments. This model integrates CNN and GRU architectures in a hybrid approach optimized to enhance accuracy and reliability in detecting network-based attacks within IIoT systems. Alpha-Net introduces an innovative

communication sequence architecture to illustrate and enhance the interaction between different IIoT layers. Additionally, this model includes a comprehensive statistical analysis of its performance through tests such as the Paired t-test, Wilcoxon Signed-Rank Test, ANOVA, and Tukey's HSD. These statistical tests reinforce Alpha-Net's dependability and effectiveness, demonstrating significant improvements over existing IDS models in various metrics.

These three models contribute to developing a more robust and adaptive intrusion detection framework suitable for IoT and IIoT environments. They establish a foundation for integrating scalable, flexible, and effective AI-driven IDS solutions by addressing specific challenges in detecting botnet attacks and network anomalies. Subsequent sections of this chapter will delve into each model's architecture, implementation, and performance evaluations, providing a detailed analysis of their unique contributions to IoT security.

3.1.1. Experimental Setup

Our experiment was conducted on an ASUS-TUF Gaming F15 (FX506LHB) laptop featuring an Intel Core i5 10th Gen processor, 8GB RAM, 512GB ROM, and Windows 11 OS. This setup was equipped with an NVIDIA GTX 1650 GDDR6 4GB graphics card, which supported the computational requirements of the experiment efficiently. Data analysis and processing were facilitated using data analysis libraries such as Pandas, Numpy, Seaborn, Matplotlib, and Scikit-learn. Memory considerations were carefully managed due to the system's 8GB RAM capacity, ensuring optimal utilization during processing-intensive tasks.

3.1.2. Dataset Description

In cybersecurity research, accurate and comprehensive datasets are crucial for developing robust security models capable of detecting anomalies and identifying malicious activities within IoT environments. This research uses the "Detection of IoT botnet attacks N_BaIoT" dataset as the primary data source. This dataset, introduced by Mirsky and Meidan in 2018 [50], addresses the scarcity of publicly available botnet datasets designed explicitly for IoT networks. The N_BaIoT dataset is unique in that it includes benign and attack traffic captured from nine distinct IoT devices (Table 2). Each device in the dataset generates varying levels of traffic, both normal and malicious, under controlled conditions that simulate real-world botnet attacks. The malicious traffic includes instances of two prominent botnets, Mirai and BASHLITE, each introducing various attack types. The N_BaIoT dataset is multivariate and sequential, consisting of 115 attributes that capture critical aspects of network traffic, such as packet size, flow duration, and packet inter-arrival times, making it suitable for complex anomaly detection and multi-class classification tasks.

Name of device	sample for Benign	Sample of attack
Danmini doorbell	49,548	9,68,750
Ecobee Thermostat	13,113	8,22,763
Ennio Doorbell	39,100	3,16,400
Philips B120N10- Baby monitor	1,75,240	9,23,437
Provision PT-337E- Security camera	62,154	7,66,106
Provision PT-838- Security camera	98,514	7,38,377
Samsung SNH-1011-N-Webcam	52,150	3,23,072
SimpleHome-XCS7-1002-WHT- Security camera	46,585	8,16,471
SimpleHome-XCS7-1003-WHT- Security camera	19,528	8,31,298

The dataset provides a realistic scenario for testing and evaluating security models by capturing normal and attack conditions. Table 3 provides a breakdown of the types of attacks in the dataset, offering details about each attack type and the associated number of instances.

Mirai and BASHLITE	Type of attack	Description	Number of instances
BASHLITE	Gafgyt combo	Sending spam data to a network- transmitting unsolicited or unwanted messages or advertisements to a network, often with the intention of overwhelming the system or spreading malware.	15,345
BASHLITE	Gafgyt Scan	Network scanning for attacking systems- examining a network to identify vulnerabilities or potential targets for a botnet attack.	14,648
BASHLITE	Gafgyt UDP	Sending a flood of requests in connection-oriented- sending a large no. of requests to a server or network in a short period of time, often with the intention of overwhelming the system and causing it to crash.	15,602
BASHLITE	Gafgyt TCP	Sending a flood of requests in connection-less - sending a large no. of requests to a server or network in a short period of time, often with the intention of overwhelming the system and causing it to crash.	15,676
BASHLITE	Gafgyt Junk	Sending spam data- distributed or transmitting unsolicited or unwanted messages or advertisements through various means such as email, and text messages.	15,449
Mirai	Mirai Scan	Scanning the network activity- it involves scanning the internet for vulnerable devices that can be infected with malware.	14,517
Mirai	Mirai UDP	Scanning the network for victim devices- the process of searching a network for a specific device that is the intended target of an attack. The scanning is to identify the IP address, open port & vulnerabilities of the victim device.	15,602
Mirai	Mirai plainUDP	UDP flooded by optimizing seeding packets per second - UDP is flooded by sending an excessive number of seeding packets per second in an attempt to optimize the process.	15,304
Mirai	Mirai Syn	Sending a flood of synchronization- overwhelming a network with a large number of synchronization messages, often with the intention of disrupting the normal operation of the system or causing a denial-of-service attack.	16,436
Mirai	Mirai Ack	Sending a flood of acknowledgment- overwhelming a network with a large number of acknowledgment messages, often with the intention of disrupting.	15,138
None	Benign	Unharmful network data- it is legitimate data that is not intended to cause harm or damage to a system.	15,538

3.1.3. Performance Evaluation Metrics

We employed several evaluation metrics to evaluate the model's performance on various attack types: accuracy, recall, precision, F1 score, ROC, and the confusion matrix. These metrics were chosen for their ability to provide a well-rounded assessment of the classification model's effectiveness across both benign and malicious classes. Table 7 defines each metric along with their respective formulas.

Metrics	Definition	Formula
Confusion matrix	Used to evaluate the performance of a classification model by comparing the predicted labels with the actual labels.	
True positive (TP)	Record is successfully detected as malicious	
False positive (FP)	Record is wrongly detected as malicious.	

True Negative (TN)	Record is classified as non-malicious.	
False Negative (FN)	Record is undetected by the system.	
Accuracy	Measure of how well the model predicts the correct labels.	$Accuracy = \frac{TP + TN}{TP + FP + TN + FN}$
Precision	Measure of how many of the predicted positive labels are actually positive.	$Precision = \frac{TP}{TP + FP}$
Recall	Measure of how many of the actual positive labels are correctly predicted.	$Recall = \frac{TP}{TP + FN}$
F1 score	Harmonic mean of Recall and Precision.	$F1\ Score = 2 * \frac{Precision * Recall}{Precision + Recall}$
ROC curve	Graphical representation of the performance of a classification model. TPR is ratio of true positive predictions to the total actual positive labels. FPR is ratio of false positive predictions to the total actual negative labels.	$TPR = \frac{TP}{TP+FN}$, $FPR = \frac{FP}{FP+FN}$
True Negative Rate (TNR)	Model's ability to correctly classify instances of a specific attack type as non-attacks.	$TNR = \frac{TN}{TN + FP}$
Negative Predictive Value (NPV)	Model's accuracy in predicting non-attacks for a specific attack type.	$NPV = \frac{TN}{TN + FN}$
False Positive Rate (FPR)	Rate at which instances of other attack types are incorrectly classified as the specific attack type.	$FPR = \frac{FP}{FP + TN}$
False Negative Rate (FNR)	Rate at which instances of a specific attack type are incorrectly classified as non-attacks or other types.	$FNR = \frac{FN}{TP + FN}$
False Discovery Rate (FDR)	Rate at which instances are falsely predicted as the specific attack type when they are not.	$FDR = \frac{FP}{FP + TN}$
False Omission Rate (FOR)	Rate at which instances of a specific attack type are falsely classified as non-attacks or other types	$FOR = \frac{FN}{FN + TN}$

3.2. Transfer Learning-Enabled Intrusion Detection System for IoT

In this section, we introduce the TL-BiLSTM IoT model, a hybrid architecture designed to enhance the detection of IoT botnet attacks by leveraging Transfer Learning (TL), Convolutional Neural Networks (CNN), and Bidirectional Long Short-Term Memory (BiLSTM) layers. This model builds upon CNN's capacity to automatically extract spatial features and BiLSTM's ability to capture temporal dependencies in sequential data. The hybrid design effectively combines the strengths of these components, yielding an architecture suited for complex IoT intrusion detection tasks. Specifically, Transfer Learning allows the model's rapid adaptation to novel threats, making it particularly useful for detecting highly dynamic and evolving IoT botnet attacks, including BASHLITE and Mirai.

3.2.1. Model Architecture and Design

The TL-BiLSTM model benefits from the flexibility of Transfer Learning, which enables it to adapt pre-trained knowledge to the IoT security domain without extensive retraining. This adaptability is crucial for IoT environments, where devices are often resource-constrained and need efficient, scalable solutions for anomaly detection.

The proposed model, termed TL-BiLSTM, combines Convolutional Neural Networks (CNN) and Bidirectional Long Short-Term Memory (BiLSTM) to develop an advanced framework capable of efficiently processing and classifying time-series data. The input to this model is a 3D tensor with dimensions (batch_size, timestep, features), where timestep denotes the sequence length, and features represent the characteristics at each timestep. This configuration enables the model to effectively capture temporal patterns and dependencies within the data, facilitating robust identification and classification of intricate patterns and anomalies. A simplified view of the TL-BiLSTM architecture is depicted in Figure 5, while the proposed model structure is illustrated in Figure 3. The TL-BiLSTM architecture consists of the following layers:

Input Data → Conv1D Layer → Conv1D Layer → Bidirectional LSTM → Bidirectional LSTM → Flatten → Dense Layer → Dense Layer → Dropout → Output Layer

The proposed CNN-BiLSTM model has been developed to perform multiclass classification on sequential data, explicitly targeting the identification and categorization of botnet attacks. This architecture uses convolutional neural networks (CNNs) to extract spatial features and bidirectional long short-term memory networks (BiLSTMs) to capture sequential dependencies, creating a comprehensive model well-suited for botnet detection complexities. Each layer of the model contributes uniquely, collectively building a robust framework to enhance predictive accuracy in a multiclass setting.

The model's input tensor is (n×m×1), where n represents the number of time steps, m is the number of features, and the final dimension of 1 denotes a single channel. This configuration is particularly suitable for processing time-series data, as it allows the model to analyze temporal and feature-based relationships within the sequence.

To initiate the feature extraction, the model uses a 1D convolutional layer equipped with 64 filters, a kernel size of 5, a stride of 1, and 'same' padding to preserve the dimensionality across layers. This layer aims to identify local patterns within the data, capturing preliminary spatial features that will serve as the foundation for subsequent layers. Mathematically, the output from the convolutional layer can be expressed as follows:

$$Y_{i,j} = \sum_{k=1}^{K} w_k X_{i,j+k-\lceil \frac{K}{2} \rceil}$$

Where Y denotes the output tensor, w_k represents filter coefficients, and K is the kernel size. An activation function, denoted by σ, is then applied to the convolutional outputs, yielding:

$$Z_{i,j,k} = \sigma\left(Y_{i,j,k} + b_k\right)$$

$$Z_{i,j,k} = \sigma\left(\sum_{p=1}^{K} w_{p,k} X_{i,j+p-\lceil \frac{K}{2} \rceil + b_k}\right)$$

Where b_k is the bias term. This activation helps capture non-linear relationships in the data.

A second convolutional layer, similar in structure but with 32 filters, further refines these spatial features. With a kernel size of 5, stride of 1, and 'same' padding, this layer builds upon the initial feature maps, extracting more complex characteristics essential for capturing the intricacies of botnet behaviors. The output from this layer is similarly computed by:

$$Y_{i,j,k} = \sum_{p=1}^{K} w_{p,k} X_{i,j+p-\lceil \frac{K}{2} \rceil,k}$$

The refined features are now ready for sequential processing. The model employs a bidirectional LSTM (BiLSTM) layer, which processes the data forward and backward, capturing dependencies between time steps. For the forward pass, the operations are defined as:

$$i_t = \sigma(W_{i,f}x_t + U_{i,f}h_{t-1} + b_i)$$
$$f_t = \sigma(W_{f,f}x_t + U_{f,f}h_{t-1} + b_f)$$
$$o_t = \sigma(W_{o,f}x_t + U_{o,f}h_{t-1} + b_o)$$
$$g_t = tanh(W_{g,f}x_t + U_{g,f}h_{t-1} + b_g)$$
$$c_t = f_t \odot C_{t-1} + i_t \odot g_t$$
$$h_t = O_t \odot tanh(c_t)$$

Where x_t is the input at time Stamp, h_{t-1} is the hidden state and $i_t, f_t, o_t, g_t, c_t, h_t$ are the input gate, forget gate, output gate, cell input, cell state, and hidden state at the time stamp 't.'

The backward pass functions similarly but in the reverse direction:

$$i'_t = \sigma(W_{i,b}x'_t + U_{i,b}h'_{t+1} + b'_i)$$
$$f'_t = \sigma(W_{f,b}x'_t + U_{f,b}h'_{t+1} + b'_f)$$
$$o'_t = \sigma(W_{o,b}x'_t + U_{o,b}h'_{t+1} + b'_o)$$
$$g'_t = tanh(W_{g,b}x'_t + U_{g,b}h'_{t+1} + b'_g)$$
$$c'_t = f'_t \odot C_{t+1} + i'_t \odot g'_t$$
$$h'_t = o'_t \odot tanh(c'_t)$$

To reinforce the sequence-based learning, a second BiLSTM layer with 16 units further captures sequential dependencies, refining the temporal representations from the previous BiLSTM layer. The outputs from this layer feed into a Flatten layer, converting the multi-dimensional sequence output into a 1D tensor. This flattened output provides a compact representation suitable for dense layer processing.

The dense layers perform high-level feature extraction with two layers: the first with 128 units and the second with 64, both using the ReLU activation function to introduce non-linearity. A dropout layer with a rate of 0.1 follows, reducing the risk of overfitting by randomly deactivating 10% of the units during each training iteration. Finally, a softmax-activated output layer yields a probability distribution across the botnet categories, enabling multiclass classification. This carefully designed architecture balances spatial and sequential feature extraction, leading to high precision in botnet detection.

3.2.2. Dataset Pre-Processing

The model evaluation utilizes the "Detection of IoT Botnet Attacks N_BaIoT" dataset, which includes a broad spectrum of benign and malicious traffic from various IoT devices affected by BASHLITE and Mirai botnets. A comprehensive preprocessing phase ensures data consistency and quality, including anomaly and attack identification, data cleaning, and standardization.

The preprocessing phase is essential for preparing the dataset for effective and practical model training and evaluation. This phase consists of several steps, including anomaly detection, data augmentation, feature encoding, and data standardization, which collectively improve the dataset's quality and suitability for machine learning tasks.

Anomaly and Attack Identification

The dataset incorporates multiple attack types from Mirai and BASHLITE botnets, as outlined in Table 3. This multi-class structure is beneficial for a classification model incorporating transfer learning, as it enables the model to identify and differentiate between distinct attack behaviors. By focusing on the specific attack patterns, the model is trained to accurately classify these events, an approach that is critical for an effective anomaly detection system.

Data Augmentation

Given the dataset's multivariate nature, data augmentation is applied to enhance the diversity and generalizability of the training set. Augmentation techniques include the addition of random noise to numerical features and transforming categorical data, which improves the model's resilience against overfitting and enhances its ability to generalize across similar patterns. For instance, the 'MI_dir_L5_weight' feature is augmented by adding random noise using a standard normal distribution, effectively simulating variability within the dataset:

```
row['MI_dir_L5_weight'] = row['MI_dir_L5_weight'] + np.random.normal(0, 1)
```

In this example, np.random.normal(0, 1) introduces a random noise with a mean of 0 and a standard deviation of 1, improving the model's robustness. Further, categorical data augmentation involves character shuffling within text columns, such as in the type field:

```
row['type'] = ''.join(np.random.permutation(list(row['type'])))
```

These transformations expand the dataset and simulate real-world conditions where data may have inherent noise or variations.

Feature Encoding

The N_BaIoT dataset contains various data types, including numerical, categorical, and binary features. To enable model compatibility and enhance computational efficiency, feature encoding is applied. Label Encoding, a more efficient alternative to One-Hot Encoding, is employed, especially beneficial for high-cardinality categorical features. Using the Pandas get_dummies function, categorical columns are transformed into binary columns, with adjustments made to ensure accuracy in the representation. For example, "Type_benign" is renamed to "benign" for clarity, ensuring consistency in feature labels across the dataset.

Data Standardization

Data standardization is pivotal in rescaling features to a standard scale, which is critical for machine learning models sensitive to feature magnitudes. The standardization formula used is:

$$Z = \frac{x - \mu}{\sigma}$$

Where,

μ = it shows the mean of the given distribution feature
$\boldsymbol{\sigma}$ = it shows the standard deviation of the given distribution function
Z = refers as the standardization score

By converting each feature into a standardized format with a mean of 0 and a standard deviation of 1, the model can more effectively compare features, irrespective of their original measurement scales. This transformation enhances the accuracy of distance-based algorithms and improves convergence rates for gradient-based optimizers, ultimately resulting in more robust model performance.

3.2.3. Performance Evaluation and Comparative Analysis

The TL-BiLSTM model was rigorously evaluated using standard performance metrics frequently applied in intrusion detection, including accuracy, precision, recall, F1-score, and the confusion matrix. This approach provides a

comprehensive assessment of the model's strengths and limitations. The TL-BiLSTM demonstrated substantial improvements in accuracy, scalability, and adaptability over existing methods, showing robust efficacy in identifying a range of IoT botnet attacks. Leveraging both temporal and sequential dependency learning through its hybrid structure, the TL-BiLSTM model is particularly effective in detecting and differentiating IoT botnet threats, such as BASHLITE and Mirai botnets.

For training and validation, the "Detection of IoT Botnet Attack N_BaIoT" dataset was split into 80% and 20% for testing. This allowed for an in-depth evaluation of model performance on unseen data and facilitated hyperparameter tuning for enhanced results. With a high accuracy of 99.52% achieved by the TL-BiLSTM model, as shown in Table 4, the results indicate the model's robust learning capabilities. High accuracy scores across training and testing sets demonstrate that the model can generalize effectively to new data, accurately identifying intrusion attempts.

Dataset	Accuracy	Loss	Precision	Recall	AUC
Train set	99.55%	0.0144	99.53%	99.49%	99.99%
Validation set	99.52%	0.0145	99.54%	99.51%	99.99%
Testing set	99.52%	0.0150	99.54%	99.50%	99.98%

The model was optimized with a hybrid CNN-BiLSTM architecture, where the CNN component extracts local data features, and the BiLSTM captures the sequential dependencies necessary for intrusion detection. Model training leveraged backpropagation with gradient descent, minimizing the categorical cross-entropy loss function. Figures 4(a) through 4(e) illustrate various metrics, providing insights into the model's training and validation accuracy, precision, recall, and the effect of the learning rate set to 0.01. This learning rate ensures steady convergence without oscillation, as demonstrated in the stabilized training loss curve in Figure 4(b). A consistently high precision score indicates the model's efficiency in minimizing false positives, while a high recall score underscores its capacity for identifying true positives.

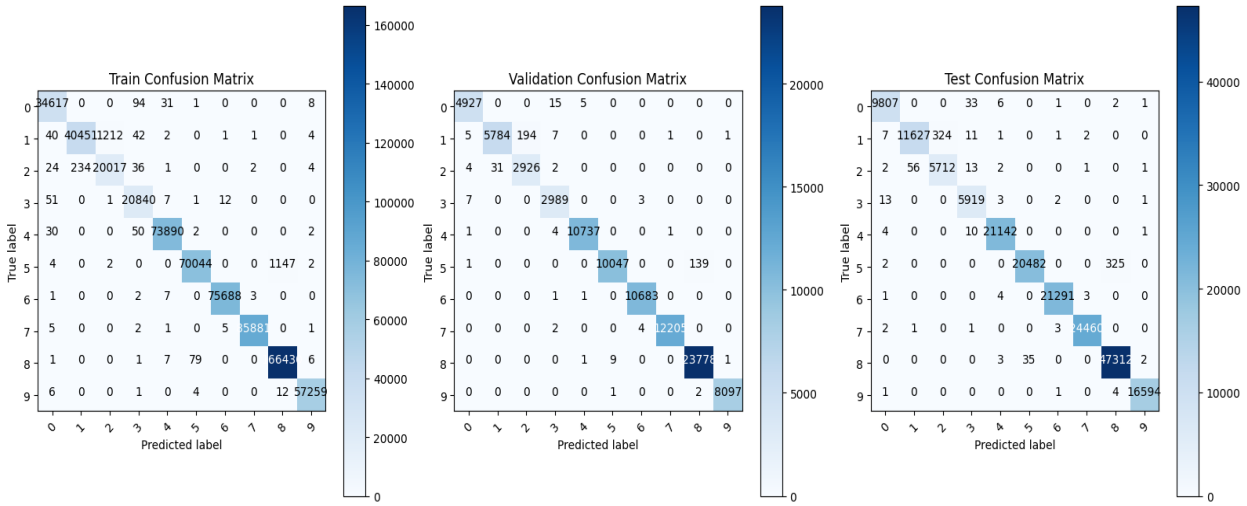
Type of attack	Precision	Recall	F1-score	Support
benign	1	1	1	9739
mirai_udp	0.97	1	0.98	11684
gafgyt_combo	0.99	0.95	0.97	6034
gafgyt_junk	1	0.99	0.99	5987
gafgyt_scan	1	1	1	21161
gafgyt_udp	0.98	1	0.99	20517
mirai_ack	1	1	1	21299
mirai_scan	1	1	1	24466
mirai_syn	1	0.99	1	47643
mirai_udpplain	1	1	1	16600

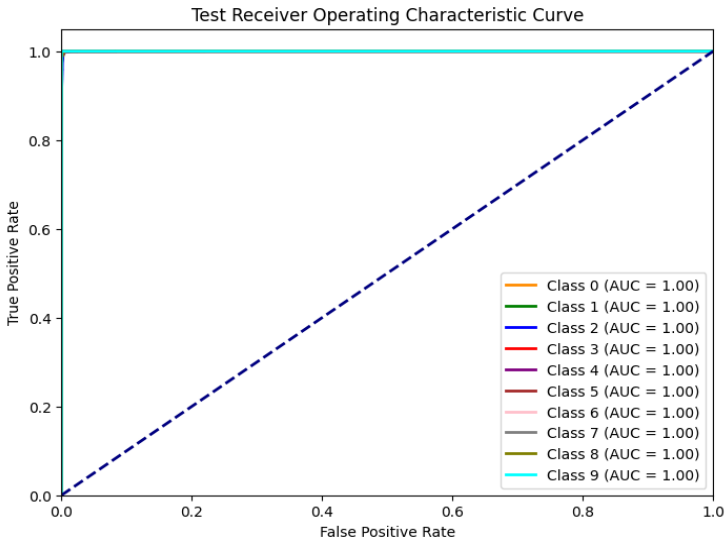
Table 5 presents the TL-BiLSTM's performance across specific IoT attack types. High precision, recall, and F1 scores across various attack types reveal the model's high efficacy in differentiating between benign and malicious classes, particularly for complex threats such as Mirai and Gafgyt botnets.

Macro precision	**Macro recall**	**Macro F1-score**	**Macro average**
99.32%	99.16%	99.23%	100%
Macro precision	**Macro recall**	**Macro F1-score**	**Micro average**
99.52%	99.52%	99.52%	99%
Macro precision	**Macro recall**	**Macro F1-score**	**weighted average**
99.52%	99.52%	99.52%	100%

Figure 5 illustrates each attack type's precision, recall, and F1 score, revealing that the model performs exceptionally well across all categories. Additionally, Table 6 summarizes these metrics' macro, micro, and weighted averages, indicating that the model maintains high accuracy and robustness across various attack types.

The TL-BiLSTM classification accuracy was further validated using a confusion matrix, shown in Figure 6. This matrix compares the predicted and actual classifications, providing insights into the model's predictive accuracy for each class. Figures 6(a), (b), and (c) show the matrices for training, validation, and testing sets, respectively. A high count of true positives and true negatives confirms the model's effectiveness in intrusion detection across all datasets. Figure 7 displays the ROC curve, which measures the model's ability to distinguish between true and false positives, providing insights into the optimal threshold for maximum sensitivity and specificity.





In our research, we assessed the performance of our recursive network by incorporating a diverse set of evaluation metrics beyond standard measures such as accuracy and precision. These supplementary metrics include the Matthews Correlation Coefficient (MCC), True Negative Rate (TNR), Negative Predictive Value (NPV), False Positive Rate (FPR), False Discovery Rate (FDR), False Omission Rate (FOR), and False Negative Rate (FNR). This comprehensive approach allows for a better understanding of the stability and effectiveness of our recursive network under various conditions, providing insights that go beyond conventional accuracy-based assessments. By evaluating our model's performance across a wide range of metrics, we aim to identify specific strengths and potential areas for optimization that could further enhance the network's robustness.

To apply MCC in the context of multiclass classification, we computed it for each class as a binary classification problem by distinguishing each class from the remaining classes. The overall MCC metric for the multiclass model is the average of these individual MCC values. The MCC value ranges from -1 to 1, where a score 1 represents a perfect classification, 0 suggests a performance equivalent to random guessing, and -1 indicates complete misclassification. Thus, a higher MCC value indicates stronger model performance, with our proposed model achieving an impressive MCC value of 0.9944 across all classes, highlighting its robustness in multiclass settings. The MCC formula is given by:

$$MCC = \frac{(TP * TN) - (FP * FN)}{\sqrt{(TP + FP)(TP * FN)(TN * FP)(TN + FN)}}$$

Table 7: Experimental stability analysis of the proposed model.

Type of attack	TNR	NPV	FPR	FDR	FOR	FNR
Benign	0.9998	0.9997	0.000182	0.003252	0.000245	0.004365
mirai_udp	0.9996	0.9980	0.000329	0.004878	0.001994	0.028898
gafgyt_combo	0.9981	0.9995	0.001806	0.053678	0.000419	0.012960
gafgyt_junk	0.9996	0.9998	0.000379	0.011358	0.000106	0.003200
gafgyt_scan	0.9998	0.9999	0.000116	0.000898	0.000091	0.000709
gafgyt_udp	0.9997	0.9980	0.000213	0.001706	0.001985	0.015714

mirai_ack	0.9999	0.9999	0.000049	0.000376	0.000049	0.000376
mirai_scan	0.9999	0.9999	0.000037	0.000245	0.000044	0.00286
mirai_syn	0.9975	0.9999	0.002401	0.0006948	0.000291	0.00845
Mirai_udp plain	0.9999	0.9999	0.000036	0.000361	0.000036	0.00361

In addition to evaluating the model's overall performance, the metrics utilized in our analysis enable the identification of classes that present specific classification issues, allowing for the optimization of the model's hyper parameters. The true negative value (TNR) for each class is depicted in Figure 8, with a greater value indicating better performance in accurately identifying negative cases. A TNR score close to one indicates that all negative cases were correctly identified as such. Figure 9 displays the negative predictive value (NPV) for each class, which assesses the model's ability to predict negative events properly. A greater NPV suggests better performance in detecting negative events. Figure 10 depicts the false positive rate (FPR) for each class, which measures the frequency with which normal communication is wrongly classified as an attack in an intrusion detection system (IDS). A lower FPR score denotes superior performance. The false discovery rate (FDR) is shown in Figure 11, and it ranges from 0 to 1, with a lower value suggesting higher performance in correctly recognizing positive instances. Lastly, Figure 12 depicts the false omission rate (FOR), which assesses the model's performance in scenarios where correctly identifying negative instances is more important than correctly identifying positive instances and the cost of false negatives is minimal. These metrics help identify specific aspects of the model's performance that need to be improved, influencing hyperparameter modification to increase the model's performance.

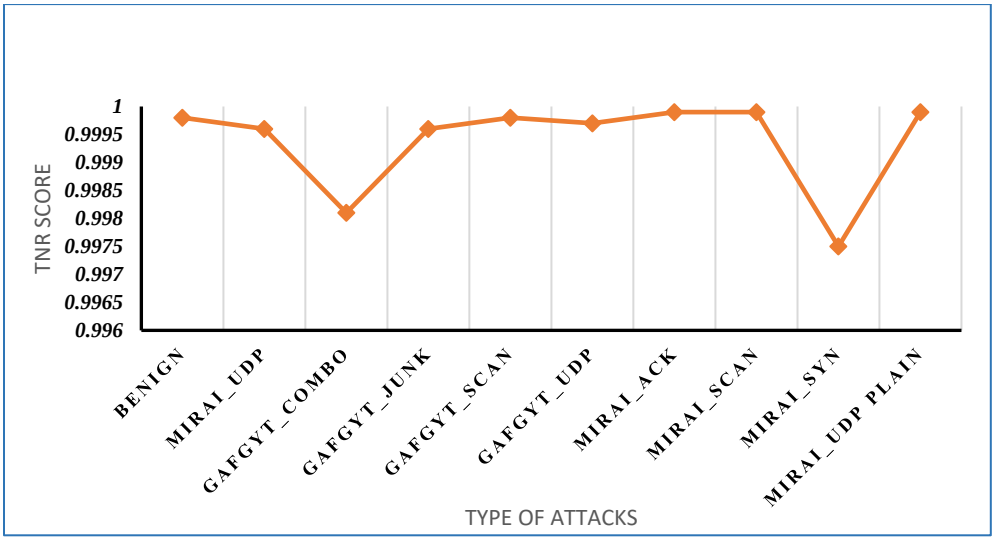


Figure 8: True Negative rate of each class.

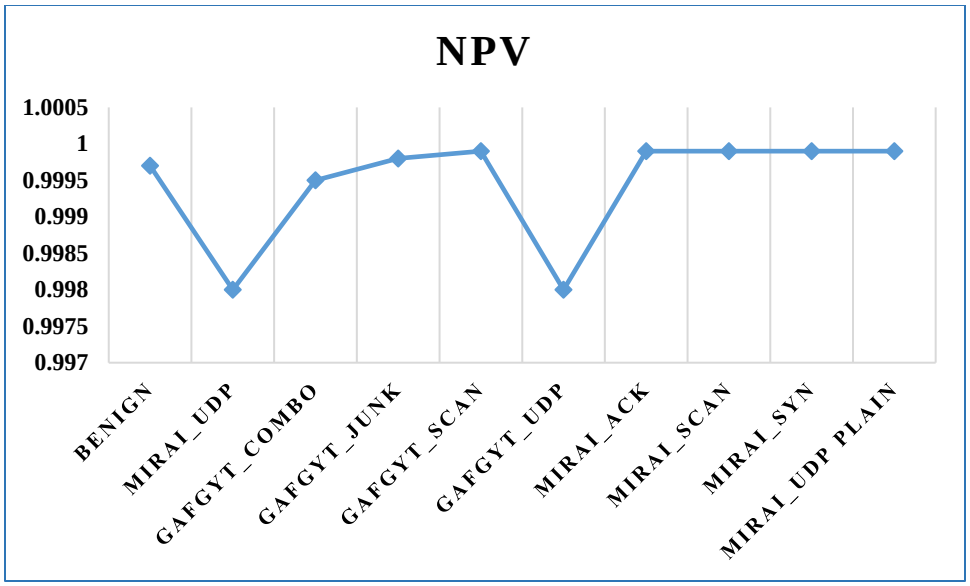


Figure 9: Negative predictive value for each class.

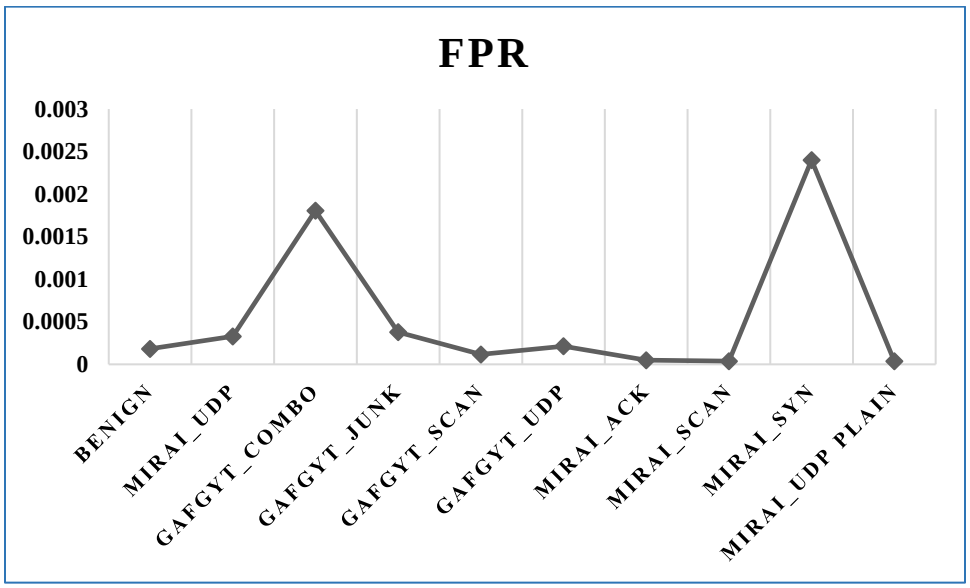


Figure 10: False positive rate for each class.

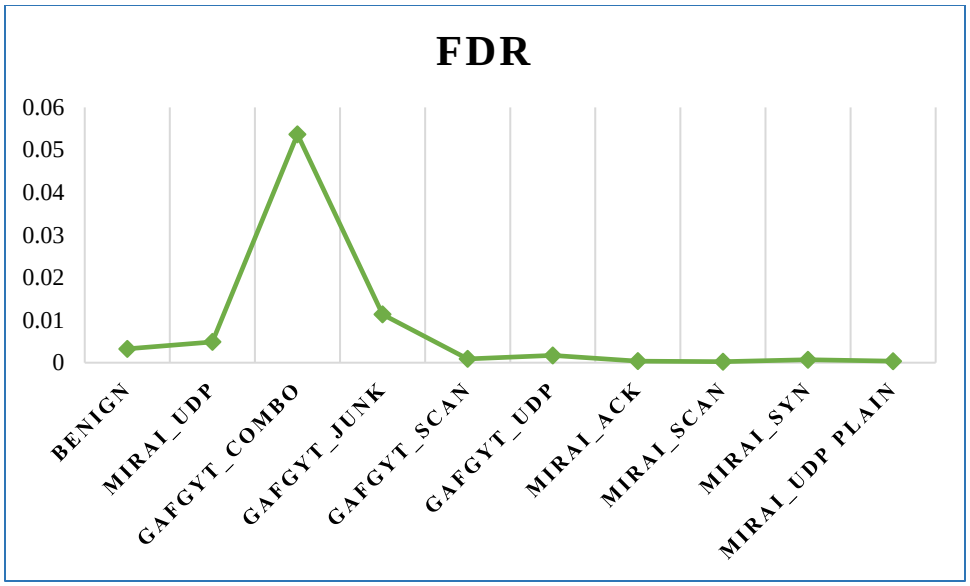


Figure 11: False discovery rate for each class.

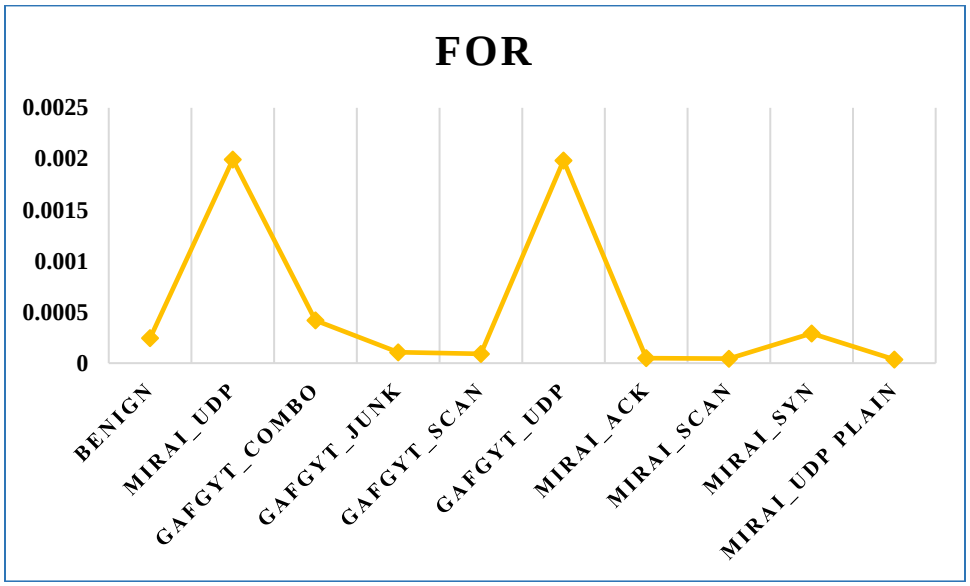


Figure 12: False omission rate for each class.

In summary, our proposed model excellently classified different types of botnet attacks. It exhibited high precision, recall, and F1 scores across all attack classes, achieving a commendable balance between accuracy and the ability to identify relevant instances. The confusion matrix and additional metrics provided further insights into the model's performance and facilitated optimization efforts.

3.2.3.1. Comparative Analysis with State-of-the-Art Techniques

This section compares our proposed model, TL-BILSTM, against several state-of-the-art IoT botnet attack detection methodologies using the "Detection of IoT Botnet Attacks N_BaIoT" security dataset. Table 9 displays the performance metrics, including accuracy, precision, recall, F1-score, and loss, alongside the scalability and adaptability of these approaches. Our proposed TL-BILSTM model exhibits superior performance in multiclass

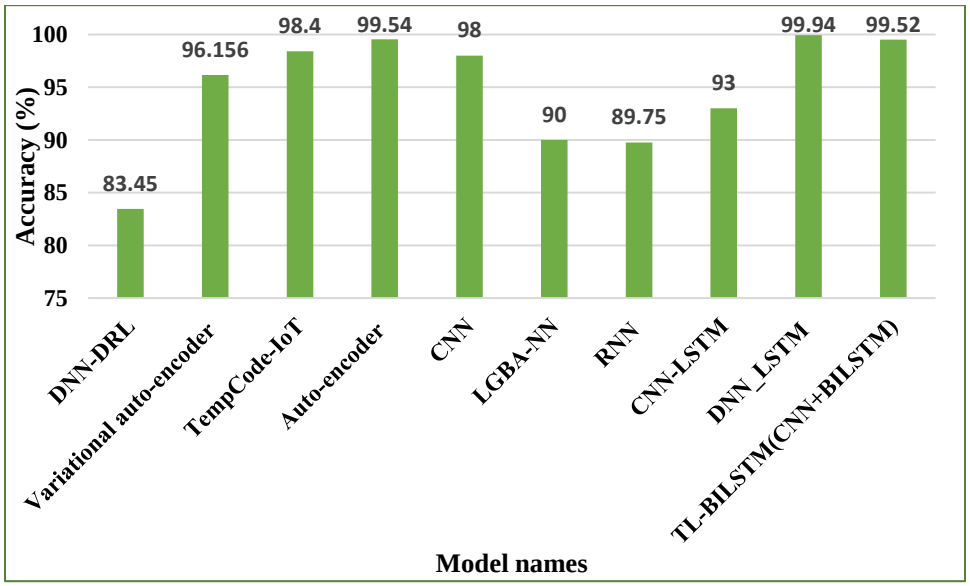
classification for IoT botnet attacks across multiple parameters, outperforming the current methods in terms of accuracy, recall, and low loss values.

Reference & year	Purpose	Model name	Classification type	Side	Accuracy	Precision	Recall	F1-score	Loss	Scalability	Adaptability
2021,[53]	IoT attack detection using deep reinforcement learning	DNN-DRL	Binary classification	Client	83.45	-	-	-	-	Low	Moderate
2022, [54]	Attack detection using different autoencoder	Variational auto-encoder	Binary classification	Client	96.156	-	-	-	-	Moderate	Lowe
2021, [36]	TempCode-IoT model for detection of IDS flow	TempCode-IoT(flow-based statistical feature model)	Binary classification	Client	-	98.4	99.4	98.9	-	High	Moderate
2022, [55]	Anomaly detection using transfer learning for DDOS attack	Auto-encoder	Multiclass classification(3 classes)	Client	99.54	-	-	-	-	High	Moderate
2022, [56]	Botnet classification in the Internet of Things	CNN	Multiclass classification (3 classes)	Client	0.98	0.98	0.97	0.976	-	Moderate	Low
2021, [23]	Botnet attack detection using bio-inspired algorithm	Local-global best bat algorithm for neural network(LGBA-NN)	Multiclass classification (10 classes)	Client	0.90	0.90	0.90	0.90	0.2	Low	Low
2021, [57]	Deep learning approach for botnet attack	RNN	Multiclass classification	Client	89.75	-	-	-	-	Moderate	Moderate
2021, [58]	Hybrid model to detect the botnet attack in IoT application	CNN-LSTM	Multiclass classification	Client	93	94	89	85	0.13	Moderate	High
2022, [59]	Hybrid deep learning approach for botnet attack in	DNN_LSTM	Multiclass Classification(6 classes)	Client	99.94	99.91	99.86	99.86		High	High

	securing industrial IoT										
2023, [60]	Deep learning prediction model	DBoTM	Regression problem	Client	R^2 = 71%	-	-	-	-	Moderate	Moderate
Our proposed model	**Hybrid transfer learning model for IDS in IoT environment**	**TL-BILSTM(CNN+BILSTM)**	**Multiclass classification (0n 10 classes)**	**Client-side**	**99.52**	**99.54**	**99.50**	**99.52**	**0.0150**	**High**	**High**

The enhanced performance of the proposed TL-BILSTM model can be attributed to its two-phase approach. Significant features impacting training data are initially selected and scaled to normalize the dataset, eliminating redundant features. In the second phase, the Conv1D layer captures localized patterns, while the BILSTM layer models the global sequence characteristics by analyzing the entire data sequence. This combination of Conv1D and BILSTM layers enables the model to accurately classify attack types by capturing low- and high-level features. The non-linear activation functions ReLU for Conv1D and dense layers, and Softmax for the output layer further contribute to the model's capacity to generate accurate class probabilities.

This two-stage process effectively balances model complexity and efficiency, significantly enhancing the model's precision in detecting various botnet attack classes. Figure 13 provides a graphical representation of the accuracy comparison between the proposed TL-BILSTM model and existing methods, demonstrating its robustness in classifying multiple botnet attack types. Overall, the results indicate that the proposed TL-BILSTM model achieves high performance with scalability and adaptability, making it well-suited for dynamic IoT environments.



3.3. Deep Learning-Enabled Intrusion Detection for Industrial IoT

The increased deployment of IoT devices in industrial environments introduces security vulnerabilities that expose systems to severe risks, especially in critical infrastructure settings. Industrial IoT (IIoT) networks often consist of heterogeneous devices, creating complex environments where identifying malicious activities is highly challenging. This section discusses the proposed deep learning-based intrusion detection model tailored explicitly for IIoT environments. Our model, named AttackNet, leverages an adaptive CNN-GRU architecture to detect and classify botnet attacks effectively, addressing unique challenges in IIoT security.

3.3.1. IIoT-Specific Anomaly Detection Framework

In the Industrial Internet of Things (IIoT) context, detecting network-based anomalies is critical to safeguarding industrial processes from cyber threats. This section presents an anomaly detection framework explicitly designed for IIoT environments, leveraging a Convolutional Neural Network and Gated Recurrent Unit (CNN-GRU) hybrid model for effective and robust detection of various types of malicious network traffic.

The proposed IIoT anomaly detection framework aims to enhance network security by accurately identifying malicious data from heterogeneous IoT devices within industrial networks. Given the diverse range of IIoT devices with varying configurations and capabilities, the model must be highly adaptive to ensure accurate threat detection. Our framework leverages deep learning methodologies, particularly a hybrid CNN-GRU model, which benefits from both the spatial feature extraction capability of CNN and GRU's temporal sequence modeling capability. This combination is especially suited for IIoT environments, where attacks often involve complex, sequential patterns.

The framework functions through two primary stages:

1. **Feature Extraction with CNN**: Convolutional Neural Networks (CNNs) identify patterns and relevant features within the input data, such as spatial correlations within network traffic.
2. **Temporal Modeling with GRU**: Gated Recurrent Units (GRUs) handle sequential dependencies, allowing the model to identify temporal patterns associated with network-based attacks in IIoT.

The CNN-GRU hybrid model allows for efficient multi-class classification, particularly for detecting botnet attacks using the N_BaIoT dataset. This dataset consists of time-series data derived from IIoT devices, which provides a comprehensive testbed for assessing the model's ability to classify benign and malicious activities.

The CNN-GRU model architecture is constructed to perform end-to-end classification, transforming input time-series data into outputs representing different attack classes. The architecture consists of multiple layers, as detailed below:

The input to the CNN-GRU model is a time-series data matrix, X, of shape (T, F) where T represents the time steps and F is the number of features in each step. This can be formally defined as:

$$X = [x(1), x(2), \dots \dots, x(T)]$$

Where each $x(t)$ represents the feature vector at the time t.

The first layer in the model is a 1-dimensional convolutional layer (Conv1D) with 64 filters, a kernel size of 5, and a stride of 1. This layer captures spatial features within the input sequence. Defining W(1) as the weight matrix of size (5,1,64) and $b(1)$ as the bias vector of size (64), the output sequence $Z(1)$ of this layer is calculated as follows:

$$Z[1][i] = f1\left(\sum j = 1,2,3,4,5\ W[1][j] * X[i + j - 3] + b[1]\right)$$

Where $f1$ is the activation function (ReLU) and i ranges from 1 to T.

The second Conv1D layer includes 32 filters, with the same kernel size and stride as the first convolutional layer. The weight matrix W(2) has dimensions (5,64,32), and b(2) is the bias vector of size (32)(32)(32). The output sequence Z(2) is given by:

$$Z[2][i] = f2\left(\sum j = 1,2,3,4,5\ W[2][j] * Z[1][i + j - 3] + b[2]\right)$$

Where $f2$ is the activation function (ReLU) and i ranges from 1 to T.

The output from the second Conv1D layer is passed through a MaxPooling1D layer with a pool size of 4. This operation reduces the temporal dimension, aggregating spatial features over smaller regions to improve computational efficiency. The output sequence Z(3) is given by:

$$Z[3][i] = max(Z[2][4 * (i - 1) + 1], Z[2][4 * (i - 1) + 2], Z[2][4 * (i - 1) + 3], Z[2][4 * (i - 1) + 4])$$

This pooling operation downsamples the input sequence, thus reducing the feature map size by a factor of 4.

Following the MaxPooling layer, two GRU layers capture temporal dependencies in the downsampled features. The first GRU layer has 32 units, capturing high-level temporal patterns, while the second GRU layer, with 16 units, further refines these patterns for classification.

After pooling, let the input to the GRU layer be Z(3) with a shape (T′,32). The output Z(4) for each time step t in this GRU layer is computed by applying a series of gating mechanisms (reset and update gates):

$$r(t) = \sigma(W[3][r] * h(t - 1) + U[3][r] * Z[3][t] + b[3][r])$$

$$z(t) = \sigma(W[3][z] * h(t - 1) + U[3][z] * Z[3][t] + b[3][z])$$

$$h'(t) = tanh\big(W[3][h] * \big(r(t) * h(t - 1)\big) + U[3][h] * Z[3][t] + b[3][h]\big)$$

$$h(t) = \big(1 - z(t)\big) * h(t - 1) + z(t) * h'(t)$$

Where $r(t)$ and $z(t)$ are the reset and update gates, $h'(t)$ is the hidden state, σ denotes the sigmoid activation function, and * represents element-wise multiplication.

The output of the first GRU layer is fed into a second GRU layer with 16 units, where a similar set of equations applies, further refining the temporal features.

The output of the final GRU layer is flattened to form a one-dimensional feature vector Z(5). This vector is then passed through two dense layers for further processing:

1. **Dense Layer 1**: This layer has 128 units with ReLU activation, enhancing the model's ability to learn complex patterns.
2. **Dense Layer 2**: Another dense layer with 64 units and ReLU activation refines the feature representation further.

The final output layer is dense with a softmax activation function, classifying input data into different attack categories. The output of this layer represents the probability distribution across attack classes, with each class corresponding to a specific type of IIoT botnet attack. The model is compiled with cross-entropy loss and is suitable for multi-class classification. It is optimized using the Adam optimizer with a learning rate of 0.001. Key evaluation metrics include accuracy, precision, recall, and F1 score, all tailored to assess the model's ability to detect anomalies accurately within IIoT networks.

The IIoT-specific anomaly detection framework, based on the CNN-GRU hybrid model, effectively identifies malicious activities in IIoT networks. By integrating spatial and temporal feature extraction, the proposed framework provides an advanced solution for safeguarding IIoT environments against a wide range of network-based attacks. The experimental results demonstrate the model's effectiveness, achieving high accuracy and outperforming existing state-of-the-art techniques.

3.3.2. Feature Extraction and Optimization

In this section, we delve into the feature extraction and optimization processes foundational to enhancing anomaly detection accuracy in the proposed framework. Our approach maximizes data relevance by systematically removing redundant features, thereby improving model performance and resource efficiency. The model achieves a higher detection capability through optimized feature extraction while requiring fewer computational resources, a critical factor in Industrial Internet of Things (IIoT) environments characterized by resource constraints.

Optimization in feature extraction is accomplished by implementing systematic steps to retain only the most relevant features in the dataset. This selective feature extraction directly contributes to improved model performance by allowing the machine learning model to learn efficiently from the most impactful features, increasing anomaly detection's robustness. Furthermore, reducing data size through optimized extraction enhances resource efficiency, which is crucial for IIoT systems, where computational resources may be limited.

As detailed below, the feature extraction and optimization pipeline include several pre-processing techniques that enhance the dataset quality:

- Data pre-processing: Data pre-processing is crucial in refining the input dataset, ensuring it is well-suited for effective anomaly detection. The pre-processing techniques applied include data augmentation, shuffling, feature encoding, and data standardization, each playing a vital role in preparing the data for training and optimizing the model's predictive performance.
    - Data Augmentation: Data augmentation is employed to increase the size and diversity of the dataset, which is essential for improving model generalization and robustness. To introduce variability and enhance the dataset, a custom function, augment_data, systematically applies transformations to each row. For instance, randomness is introduced to the numerical column labeled 'MI_dir_L5_weight' by adding a value drawn from a normal distribution with a mean of zero and a standard deviation of one:

```
row['MI_dir_L5_weight']=row['MI_dir_L5_weight']+np.random.normal(0, 1)
```

Additionally, text columns such as 'type' can undergo character shuffling to add further variability:

```
# Example of random character shuffling in a text column

row['type'] = ''.join(np.random.permutation(list(row['type'])))
```

    - Data Shuffling: To minimize potential biases and ensure that the dataset is randomized, data shuffling is performed on the data frame, data. Shuffling rearranges the order of the data instances by randomly permuting the row indices using "np.random.permutation", ensuring that the order does not introduce unintended patterns or biases during training. This randomized arrangement is crucial for obtaining an unbiased model, as it prevents the model from learning spurious correlations that may occur in non-randomized data.
    - Feature Encoding: Feature encoding converts categorical variables into a format suitable for machine learning algorithms by transforming categorical labels into binary representations through dummy encoding. Our dataset encodes the categorical column 'type' into binary columns with unique identifiers for each category using the pd.get_dummies function. The encoded columns are stored in the labels dataframe, and the original 'type' column is dropped from the primary data dataframe:

```
# Dummy encoding categorical labels and separating from features
labels = pd.get_dummies(data['type'])
data = data.drop(columns=['type'])
```

This encoding process allows for the seamless integration of categorical information as meaningful binary features, enhancing the model's ability to interpret categorical data accurately.

- Data Standardization: Data standardization ensures that all numerical features are on a comparable scale, facilitating fair comparisons during model training. Standardization transforms each numerical column in the data dataframe to have a mean of 0 and a standard deviation of 1. This is achieved using the z-score normalization formula:

$$Z = \frac{x - \mu}{\sigma}$$

Where μ = it shows the mean of the given distribution feature, $\boldsymbol{\sigma}$ = it shows the standard deviation of the given distribution function, and Z = refers to the standardization score. The standardized function applies this formula to each numerical column, creating a standardized dataset "data_st" that ensures all features contribute proportionally, free from the influence of differing magnitudes. This process is critical for enhancing the model's reliability and interpretability by normalizing feature scales.

3.3.3. Performance Evaluation and Comparison

The overall performance of the proposed model was extensively evaluated using various quantitative metrics, including accuracy, precision, recall, receiver operating characteristic (ROC) area under the curve (AUC), F1 score, and the confusion matrix. We also focused on optimizing the model's hyperparameters to enhance its performance. In recent years, deep learning (DL) algorithms have made significant advancements, and their application to intrusion detection and classification problems has been successful. The remarkable progress of hybrid DL approaches has positioned them as a promising solution for reliable and trustworthy network intrusion detection systems (IDS).

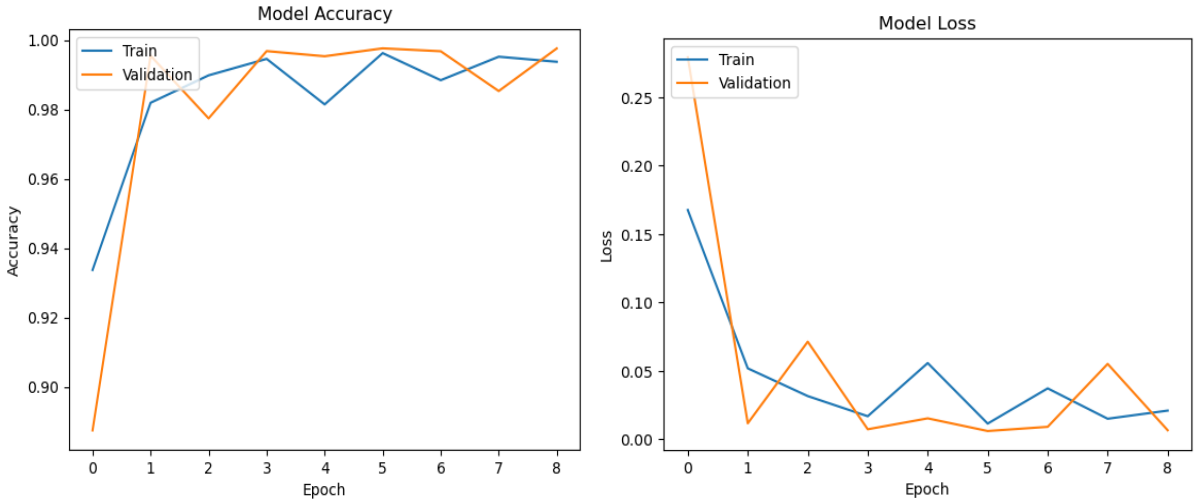
Our proposed model, which combines Convolutional Neural Network (CNN) and Gated Recurrent Unit (GRU) layers, demonstrated optimal performance. It achieved a training accuracy of 99.77% and a testing accuracy of 99.75%, with a loss of 0.0063 as shown in the Table 8. The precision and recall values were 99.75% and 99.74%, respectively. The model employed four hidden layers, including two convolutional layers and two GRU layers. Rectified Linear Unit (ReLU) was used as the activation function in the hidden layers, while softmax served as the output activation function. The categorical cross-entropy loss function, coupled with the Adam optimizer, was utilized.

Table 8: Quantitative analysis of proposed model

Dataset	Accuracy	Loss	Precision	Recall	AUC
Train set	0.9977	0.0062	0.9978	0.9976	1.0000
Validation set	0.9977	0.0065	0.9978	0.9976	0.9999
Testing set	0.9975	0.0063	0.9975	0.9974	1.0000

In the context of Figure 9 (a) and (b), we evaluate the efficacy of the proposed AttackNet model by elucidating the interplay between training and testing accuracy and training and testing loss. The horizontal axis of both figures delineates the progression of epochs, while the vertical axis corresponds to accuracy and loss, respectively. Notably,

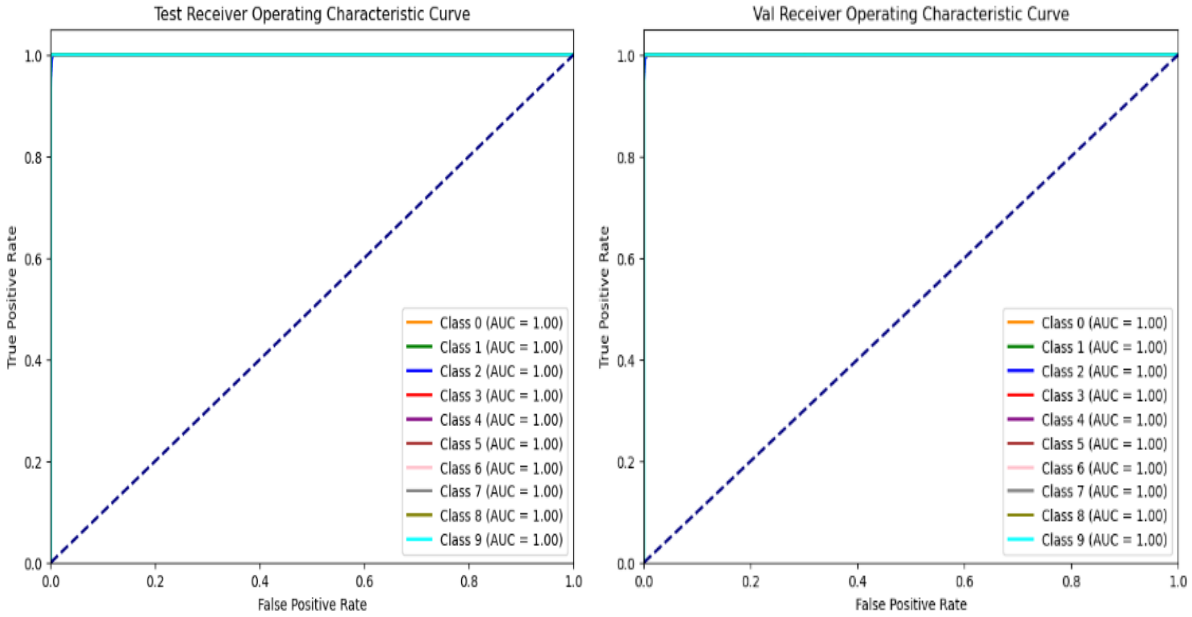
a discernible trend emerges wherein minimal epochs coincide with diminished accuracy; conversely, an augmentation in epochs correlates with an enhancement in accuracy. This dynamic indicates the model's iterative learning process, progressively refining its ability to discern patterns within the training dataset. Additionally, Figure 9 (b) serves as a graphical representation of the model's performance, specifically portraying the loss. Initially, the loss magnitude is considerable with a sparse number of epochs. However, with the advancement of epochs, there is a discernible reduction in loss magnitude. This nuanced depiction encapsulates the model's refinement over time, signifying an improvement in its capacity to minimize errors and enhance overall performance.



(a) Accuracy Curve (b) Loss curve

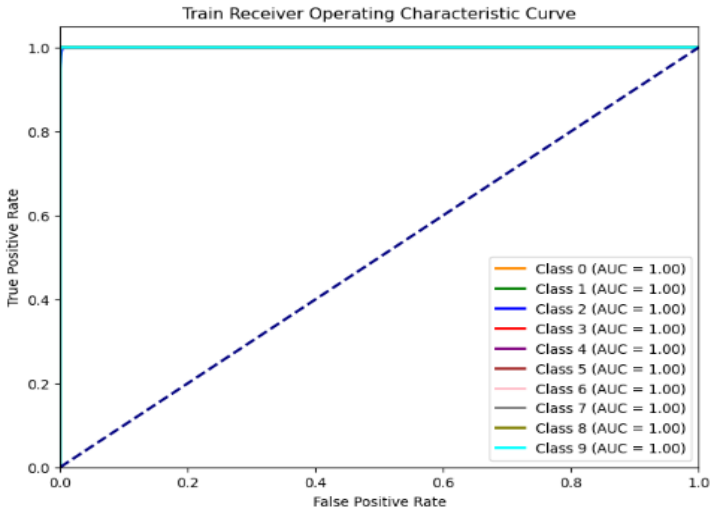
Figure 1: (a) graphical representation of training and validation accuracy, (b) graphical representation of training and validation loss.

After training the model on the N_BaIoT dataset and optimizing the hyper parameters, we achieved impressive performance on the test and validation sets. The model achieved an AUC score of 1.00 on the test set and 0.99 on the validation set, as demonstrated in the figures 10 (a) and (b).



(a) ROC curve for Test set

(b) ROC curve for Validation set



(c) ROC curve for Train set

Figure 2: ROC curve for the proposed model: based on the calculated TPR & FPR.

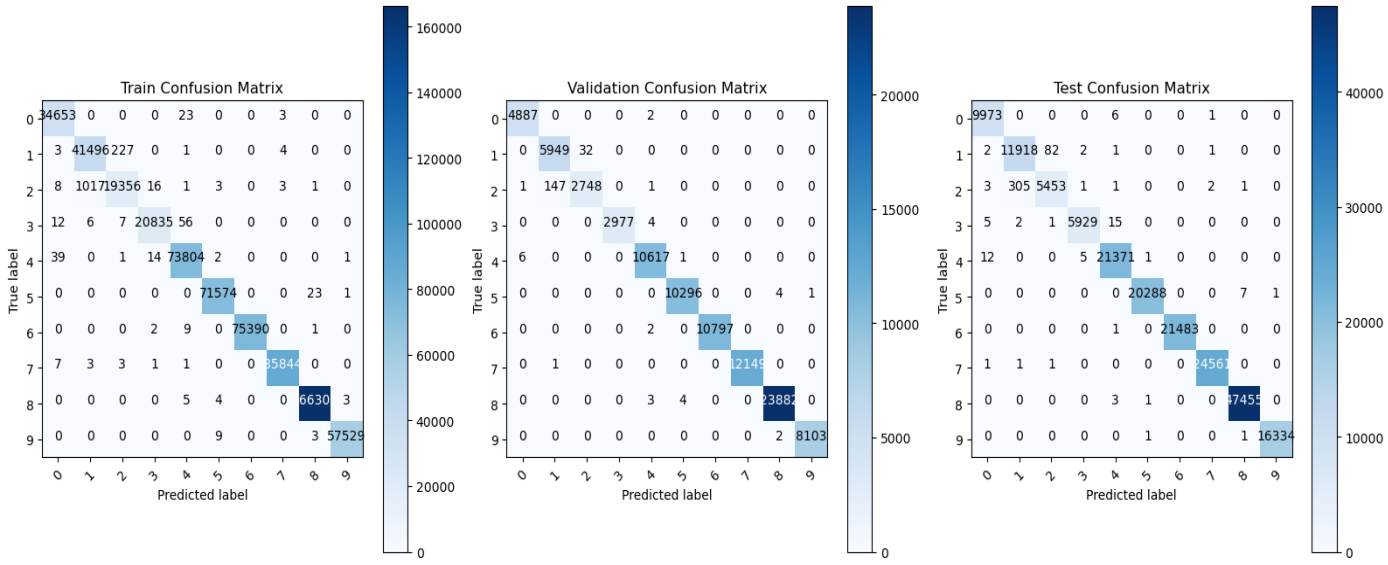
To evaluate the model's performance on different attack classes, we computed evaluation metrics such as precision, recall, F1 score, and support as shown in the table 9. Across all classes, precision, recall, and F1 score consistently exhibited high values. This suggests that the model accurately classified various types of botnet attacks, achieving high accuracy and effectively identifying instances of each attack class.

Precision values of 1 indicate a high level of precision for all attack classes, implying that when the model predicted instances as a specific type of botnet attack, it was highly likely to be correct. Similarly, recall values of 1 indicate that the model demonstrated a substantial ability to accurately identify and capture a significant majority of instances belonging to each specific attack class. Consequently, the model exhibited a remarkable ability to detect instances of diverse botnet attacks effectively. Moreover, F1 scores approaching 1 indicated a commendable balance between precision and recall for all attack classes. This exemplified the model's heightened accuracy while adeptly capturing instances from each attack class.

Table 9: Performance Evaluation Metrics of Proposed Model on Different Attack Classes

Parameters/ Attack Classes	Precision										Recall										F1-score										Support									
	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9
Train set	1	0.98	0.99	1	1	1	1	1	1	1	1	0.99	0.95	0.99	1	1	1	1	1	1	1	0.99	0.97	1	1	1	1	1	1	1	34679	41731	20405	20916	73861	71598	95402	85859	166317	57541
Validation set	1	0.98	0.99	1	1	1	1	1	1	1	1	0.99	0.95	1	1	1	1	1	1	1	1	0.98	0.97	1	1	1	1	1	1	1	4889	5981	2897	2981	10624	10301	10799	12150	23889	8105
Test set	1	0.97	0.99	1	1	1	1	1	1	1	1	0.99	0.95	1	1	1	1	1	1	1	1	0.98	0.96	1	1	1	1	1	1	1	9980	12006	5766	5952	21389	20296	21484	24564	47459	16336

To provide a detailed breakdown of the model's predictions for each attack class, we utilized a confusion matrix. The confusion matrix for the train, validation, and test sets enabled a comprehensive analysis of the model's performance as shown in the figure 11 (a), (b) and (c). We also assessed additional metrics, including True Negative Rate (TNR), Negative Predictive Value (NPV), False Positive Rate (FPR), False Negative Rate (FNR), False Discovery Rate (FDR), and False Omission Rate (FOR), for each class representing different types of botnet attacks shown in the Table 10.



(a) Train set (b) Validation set (c) Test set

Figure 3: Confusion matrix of the proposed model on the different target classes (a) for the train set (b) for the validation set (c) for the test set

In evaluating the model's overall performance, these metrics enabled the identification of specific classification issues within classes, facilitating hyper parameter optimization. Figures 11,12 and 13 depicting TNR vs. NPV, FPR vs. FNR, and FDR vs. FOR illustrated the trade-offs between correct identification of non-attack instances, avoidance of false alarms, and accurate identification of attack instances, respectively.

Table 10: Class-wise Performance Metrics Analysis

Classes	True negative rate	Negative predictive value	False positive rate	False negative rate	False discovery rate	False omission rate
0	0.999869	0.999960	0.000131	0.000701	0.002301	0.000040
1	0.998222	0.999491	0.001778	0.007330	0.025192	0.000509
2	0.999532	0.998258	0.000468	0.054284	0.015171	0.001742
3	0.999955	0.999872	0.000045	0.003864	0.001347	0.000128
4	0.999835	0.999890	0.000165	0.000842	0.001262	0.000110
5	0.999982	0.999951	0.000018	0.000394	0.000148	0.000049
6	1.000000	0.999994	0.000000	0.000047	0.000000	0.000006
7	0.999975	0.999981	0.000025	0.000122	0.000163	0.000019
8	0.999935	0.999971	0.000065	0.000084	0.000190	0.000029
9	0.999994	0.999988	0.000006	0.000122	0.000061	0.000012

The figure 12 shows the TNR v/s NPV on different classes. Higher value of TNR indicates a better ability to correct iden tify non attack instances whereas a higher value of NPV indicates a better ability to correctly identify non attacks instanc es among the predicted negative. An increased in TNR generally leads to an increase in NPV and vice versa.

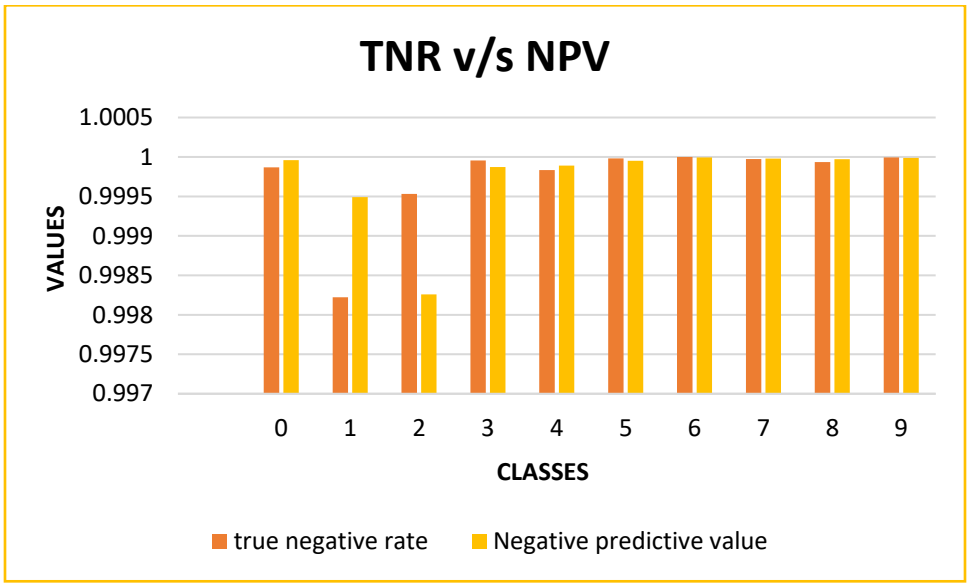


Figure 4: TNR vs NPV: Analyzing Performance Measures

The figure 13 shows the FPR v/s FNR on different classes. Lower value of FPR indicates a better ability to avoid false alarms for non-attack instances whereas lower value of FNR indicates a better ability to correctly identify attack instan ces. AS FPR increases, FNR tends to decrease and vice versa.

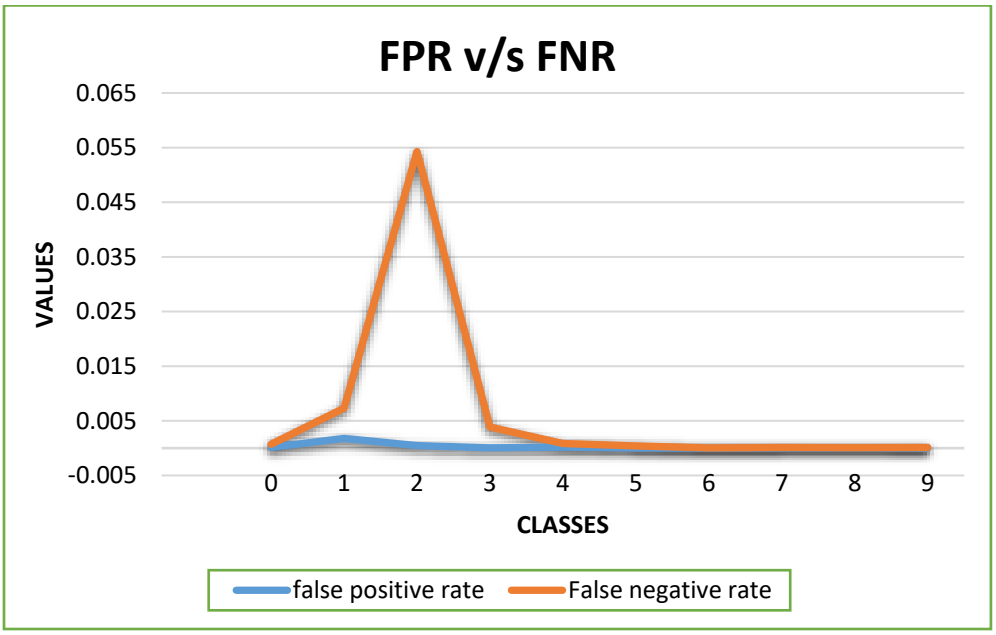


Figure 5: FPR vs FNR: Analyzing Performance Measures

The figure 14 shows the FDR v/s FOR on different classes. As FDR increases, FOR tends to decrease and vice versa. When the model makes more incorrect positive prediction (higher FDR), it tends to make fewer incorrect negative predations (lower FOR), reflecting the trade-off between making incorrect positive and negative predictions.

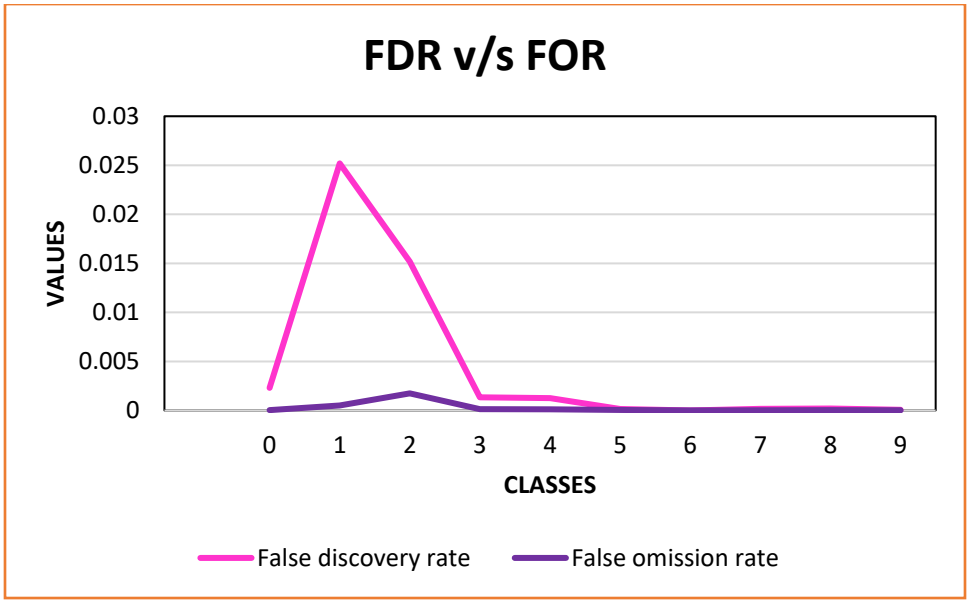


Figure 6: FDR vs FOR: Analyzing Performance Measures

The class-wise confusion matrix depicted in the figure 15 provides valuable insights into the performance of our proposed AttackNet deep learning model. It enables us to analyze the true negative rate, false positive rate, valid positive rate, and false negative rate for each class. Upon analysis, we found that the benign class constituted 5.38% of the dataset, while the gafgyt_junk and gafgyt_combo classes accounted for only 3.20% and 2.94% respectively. Furthermore, the Mirai_ack class exhibited a false positive rate of 11.60%, indicating a relatively higher misclassification of benign instances as Mirai_ack attacks. Similarly, the Mirai_scan class had a false positive rate of 13.36%. Overall, we observed that the Mirai_syn attack achieved a remarkably high detection rate, demonstrating the effectiveness of our approach in identifying instances of this specific attack type.

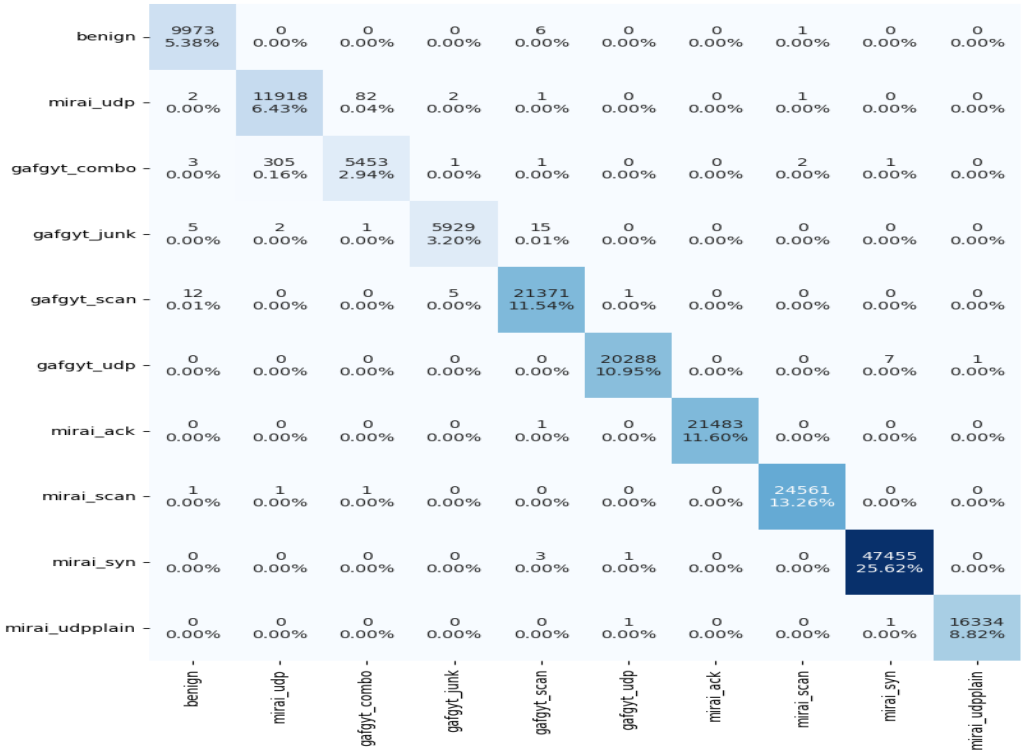


Figure 7: Class wise confusion matrix of proposed model on the different target classes

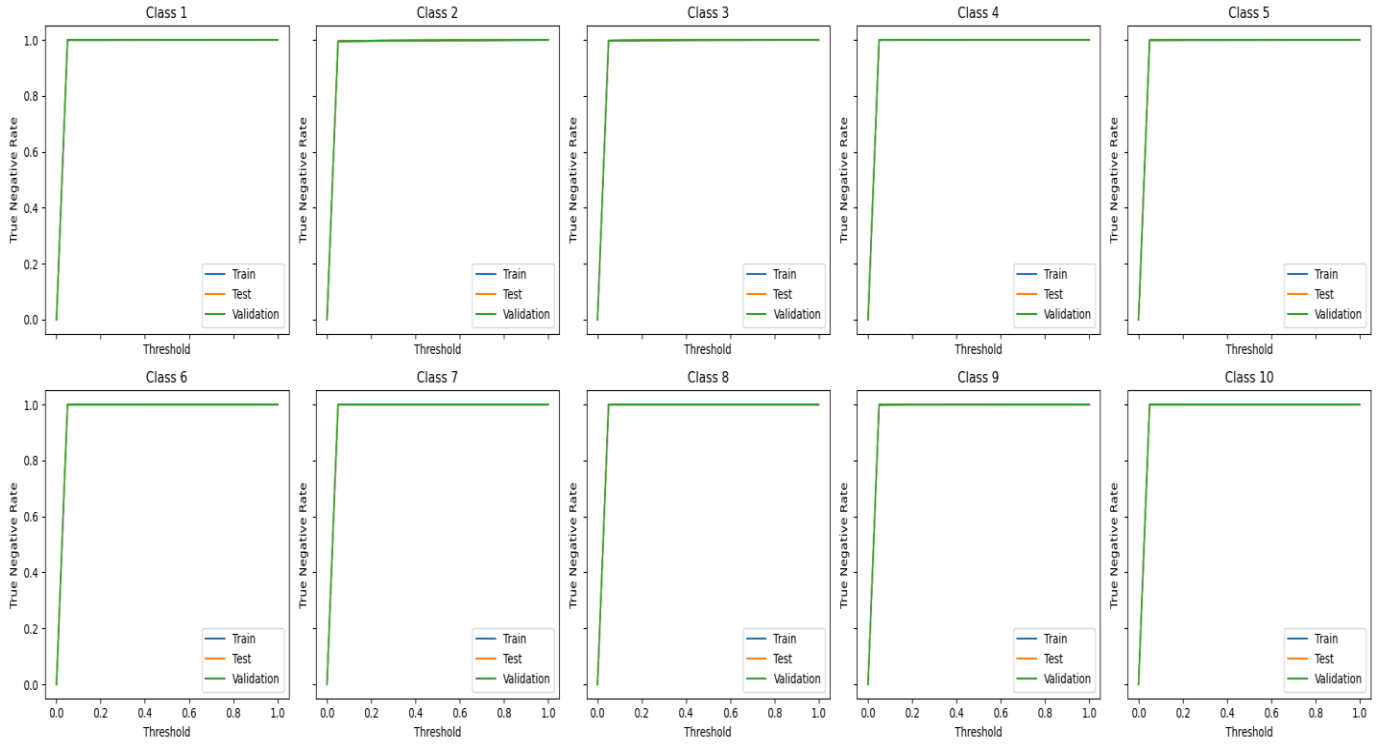


Figure 8: TNR values at each class on train, validation and test sets for correctly classified negative instances

The figure 16 shows the TNR value of each classes. It represents the proportion of true negative prediction out of all actual negative instances. The range of TNR is from o to 1, with 0 indicating no true negative predictions and 1 indicating perfect performance in identifying negative instances. It gives insight into the model's performance in classifying negative prediction which help to assess the model specificity and its ability to minimize false positive errors. The figure 17 shows the FNR values of each attack types. It represents the proportion of false negative predictions out of all actual positive instances. The range of FNR is from 0 to 1, with 0 indicating no false negative predictions and 1 indicating a high rate of false negative prediction. A lower value of FNR indicates a lower rate of missing positive instances, meaning the model is more sensitive in identifying true positive class.

In our study, we have expanded the evaluation of our recursive network's performance by incorporating the Matthews Correlation Coefficient (MCC) as an additional performance metric. The inclusion of MCC allows us to measure the stability and robustness of our classification approach. The MCC value ranges from -1 to +1, where -1 signifies completely inaccurate predictions, 0 represents random predictions, and +1 indicates precise predictions. By calculating MCC for our multiclass classification problem, we are able to assess the model's capability to correctly classify instances across multiple classes, encompassing both positive and negative predictions.

$$MCC = \frac{(TP*TN)-(FP*FN)}{\sqrt{(TP+FP)(TP+FN)(TN+FP)(TN+FN)}} \quad (22)$$

Our proposed model achieves an impressive MCC value of 0.99707, indicating a high degree of agreement between the predicted class labels and the actual ground truth across the various classes. This suggests that our model excels in effectively distinguishing and accurately classifying instances into their respective categories, resulting in a notably high level of prediction accuracy. The obtained MCC value not only demonstrates the model's strong performance, but it also serves as a testament to its robustness and stability, reaffirming its reliability in multiclass classification tasks.

In summary, our proposed model (AttackNet) has showcased outstanding performance in effectively classifying various types of botnet attacks. It has consistently demonstrated elevated precision, recall, and F1 scores across all attack classes, attaining a commendable equilibrium between accuracy and the capability to identify pertinent instances. The confusion matrix's elucidation and additional metrics offered comprehensive insights into the model's performance and streamlined optimization endeavors. Overall, our proposed model stands as a robust solution, excelling in the nuanced task of botnet attack classification.

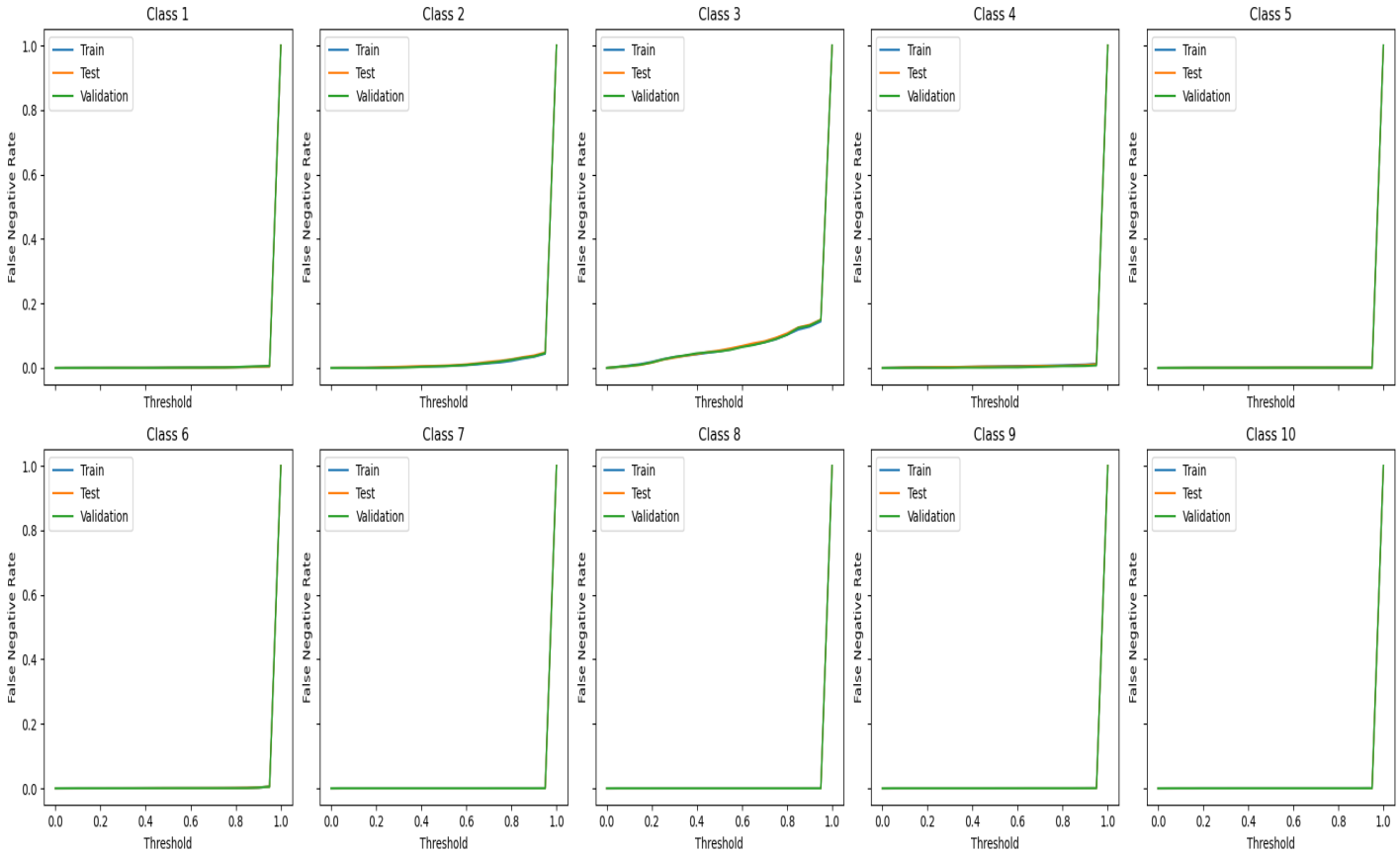


Figure 9:FNR values at each class on train, validation and test sets for incorrectly classified positive instances

3.3.3.1. Ablation study of Proposed model

This section incorporates an ablation study aimed at providing comprehensive insights into the constituent elements of the model and their respective contributions. The assessment encompasses the following scenarios:

- Case A: Training the dataset using a 1-D Convolutional Neural Network (CNN).
- Case B: Training the dataset utilizing a 2-D Convolutional Neural Network (CNN).
- Case C: Training the dataset employing a Gated Recurrent Unit (GRU).
- Case D: Training the dataset using the proposed model.

This systematic evaluation of distinct cases serves to dissect the influence and efficacy of each model variant, thereby facilitating a nuanced understanding of their individual roles and impacts on overall performance.

Table 11: Ablation study score for the proposed model AttackNet

Ablation study cases	Cases	Accuracy	Precision	Recall
Train the dataset with 1D CNN	A	0.84	0.80	0.79
Train the dataset with 2D CNN	B	0.87	0.83	0.80
Train the dataset with GRU	C	0.93	0.91	0.91

Train the dataset with Proposed model	D	**0.9975**	**0.9954**	**0.9950**

Figure 18 delineates the discernible effects resulting from each implemented enhancement. The ascending trends observed in accuracy, precision, and recall scores underscore the evident advantages derived from the refinement of the proposed model, offering profound insights into its constituent elements. The table 11 gives a detailed breakdown of scores from our ablation study. It's important to note that Case D, where we trained the dataset using our proposed model, performed better than all other cases. This emphasizes that our proposed model is more effective compared to other setups, showing better results across various evaluation measures.

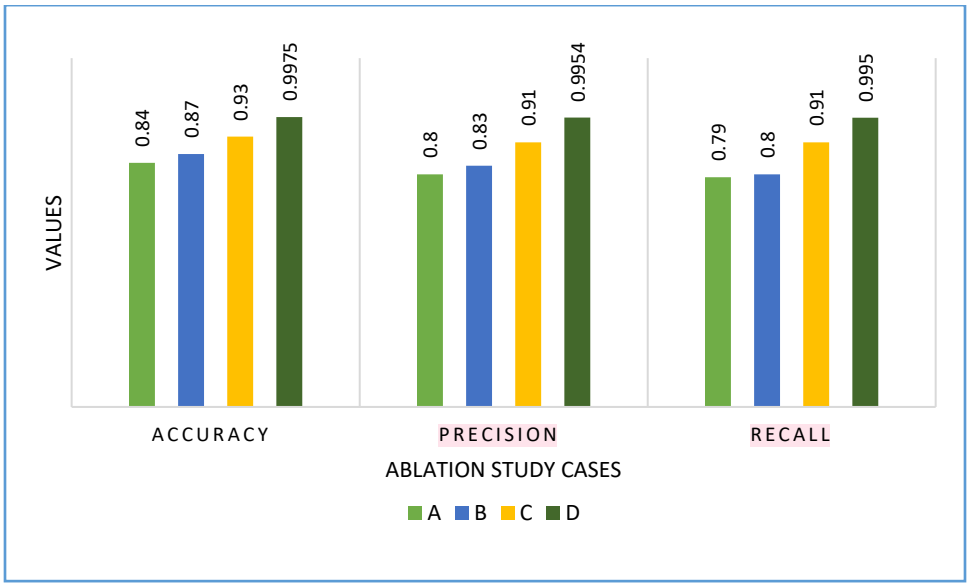


Figure 10: Ablation study result on N_BaIoT dataset.

3.3.3.2. Findings and Comparison with benchmark

A. Comparison with state of art work

This section compares the proposed model AttackNet to various cutting-edge strategies for detecting IoT botnet attacks using the "Detection of IoT botnet attacks N_BaIoT" security dataset. Table 12 shows the performance values for these current approaches on various parameters. In terms of accuracy, recall, and loss, the proposed AttackNet model outperforms the other models for multiclass classification of botnet attacks. The DNN-LSTM model [Hasan T et.al] achieved higher accuracy, precision, recall, and F1 score compared to the proposed AttackNet model. However, the difference is relatively small and both demonstrate high performance. The number of classes in proposed AttackNet model is higher (10 classes) compared to the DNN-LSTM model (6 classes). The proposed AttackNet model also provides detailed loss information, which is valuable for assessing the training performance. The proposed approach's improved performance is due to its two-phase method. Significant features that have an impact on the training data are picked while scaling the dataset to normalize it, and redundant features are removed. The Conv 1D layer gathers local features in the second phase, while the GRU layer captures global characteristics by taking the entire sequence into account. By collecting both low-level and high-level features from the input sequence, the model is able to accurately categories data using these two strategies. Deep learning-friendly non-linear activation functions such as Relu and softmax are used in the model. To generate the final class probabilities, relu is employed in the Conv 1D and dense layers and softmax in the output layer. The effectiveness of both phases justifies the proposed model's accuracy in predicting various botnet attack types.

Table 12: Comparative Analysis of Proposed Model and State-of-the-Art Approaches on the 'N_BaIoT' IoT Botnet Attack Detection Dataset

Reference & year	Purpose	Model name	Classification type	Side	Accuracy	Precision	Recall	F1-score	Loss
2021,[Baby R et.al]	IoT attack detection using deep reinforcement learning	DNN-DRL	Binary classification	Client	83.45	-	-	-	-
2022, [CU OK et.al]	Attack detection using different autoencoder	Variational auto-encoder	Binary classification	Client	96.156	-	-	-	-
2021, [Siddiqui AJ et.al]	TempCode-IoT model for detection of IDS flow	TempCode-IoT(flow-based statistical feature model)	Binary classification	Client	-	98.4	99.4	98.9	-
2022, [Shafiq U et.al]	Anomaly detection using transfer learning for DDOS attack	Auto-encoder	Multiclass classification(3 classes)	Client	99.54	-	-	-	-
2022, [Cunha AA et.al]	Botnet classification in the Internet of Things	CNN	Multiclass classification (3 classes)	Client	0.98	0.98	0.97	0.976	-
2021, [Alharbi A et.al]	Botnet attack detection using bio-inspired algorithm	Local-global best bat algorithm for neural network(LGBA-NN)	Multiclass classification (10 classes)	Client	0.90	0.90	0.90	0.90	0.2
2021, [Hezam AA et.al]	Deep learning approach for botnet attack	RNN	Multiclass classification	Client	89.75	-	-	-	-
2021, [Parra GD et.al]	Hybrid model to detect the botnet attack in IoT application	CNN-LSTM	Multiclass classification	Client	93	94	89	85	0.13
2022, [Hasan T et.al]	Hybrid deep learning approach for botnet attack in securing industrial IoT	DNN_LSTM	Multiclass Classification(6 classes)	Client	99.94	99.91	99.86	99.86	-
2023, [Haq MA et.al]	Deep learning prediction model	DBoTM	Regression problem	Client	R^2 = 71%	-	-	-	-
Our proposed model	**Hybrid-deep learning model for IDS in IoT environment**	**AttackNet (CNN+GRU)**	**Multiclass classification (0n 10 classes)**	**Client-side**	**99.75**	**99.54**	**99.50**	**99.52**	**0.0063**

This section provides informative comparisons between the performance of the proposed model and current cutting-edge IoT botnet attack detection approaches. Figure 19 compares the accuracy parameter of the proposed model to the technique currently in use. Overall, the results reveal that our proposed model AttackNet outperforms other strategies and correctly classifies various botnet attack types.

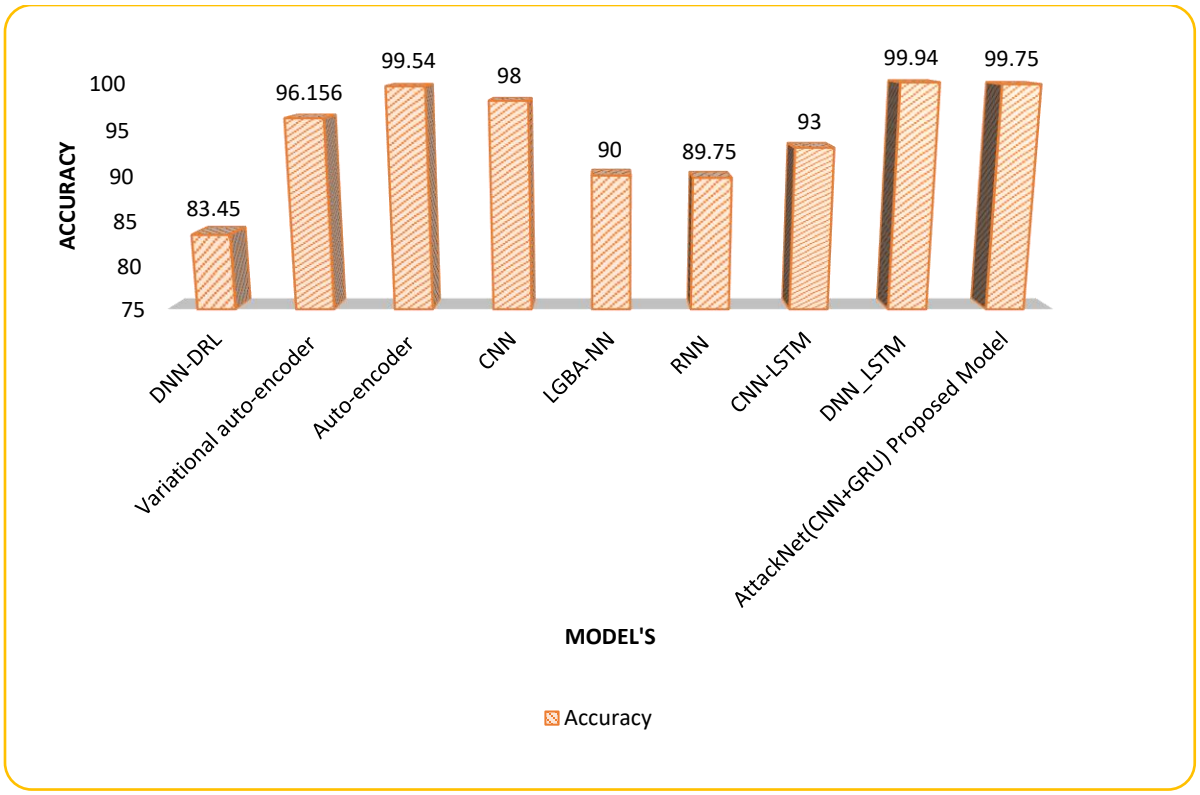


Figure 11: Graphical representation of the proposed model with the existing state of artwork on the accuracy parameter.

B. Complexity Analysis

5.2.1. Time Complexity:

The time complexity of the above model can be approximated by the number of trainable parameters in the model, which is a function of several hyperparameters such as the number of filters in the convolutional layers, the kernel size of the convolutional filters, the number of units in the GRU layers, and the number of units in the dense layers. The number of trainable parameters in the model can be computed as follows:

```
num_params = (num_filters * kernel_size + 1) * num_features + num_features^2 + num_features * num_timesteps * (num_timesteps - 1) * recurrent_units + num_features * dense_units + dense_units + num_classes
```

where:

- num_filters: the number of filters in the convolutional layers
- kernel_size: the size of the convolutional kernels
- num_features: the number of input features
- num_timesteps: the number of time steps in the input sequence
- recurrent_units: the number of units in the GRU layers
- dense_units: the number of units in the dense layers
- num_classes: the number of output classes

The time complexity of the model can be approximated as proportional to the number of trainable parameters. This is because the most computationally expensive part of training the model is computing the gradients of the loss function with respect to the model parameters, which requires evaluating the model at each training step.
The number of training steps required to train the model is proportional to the number of epochs and the size of the training data. Therefore, the time complexity of the model is also dependent on these factors. Table 13 shows the computational time complexity analysis on the dataset.

Table 13: Computational Time Analysis: Training, Validation, and Testing Performance on the Dataset

Dataset	Computational Time
Train set	160s 8ms/step
Validation set	23s 8ms/step
Test set	45s 8ms/step

5.2.2. Space Complexity:

The space complexity of the model is also proportional to the number of trainable parameters in the model. This is because each trainable parameter requires a certain amount of memory to store its value, and the total memory required to store all the trainable parameters is proportional to their number. In addition to the trainable parameters, the model also requires memory to store the input data, intermediate activations, and gradients during training. The memory required for these operations is proportional to the size of the input data, the number of units in the model, and the number of training steps required to train the model.
Overall, the time and space complexity of the model can be quite high, especially for large input data and a large number of trainable parameters. However, efficient algorithms and hardware accelerators such as GPUs can help to mitigate these issues. Additionally, techniques such as weight regularization and early stopping can help to reduce the number of trainable parameters and reduce overfitting, which can also help to reduce the time and space complexity of the model.

C. Findings:

i. Reliability and Trustworthiness:

To evaluate the reliability of our model, we consider a scenario where our model consists of 10 distinct attack types or learners. Due to the diverse nature of Deep learning, the errors that arise in these various attacks are independent and unrelated to each other. Even if certain learners exhibit inaccuracies, the remaining learners may still demonstrate accuracy, thereby allowing our approach to effectively classify intrusion attacks in Deep learning-based Industrial Internet of Things (IIoT) networks.

Figure 20 shows that the error rate of the learner for 10 different attack types is less than 0.006779. This means that our method is effective at detecting attacks in IIoT networks. We verified the trustworthiness of our model by classifying attacks with one dataset and found that the error rate was low.

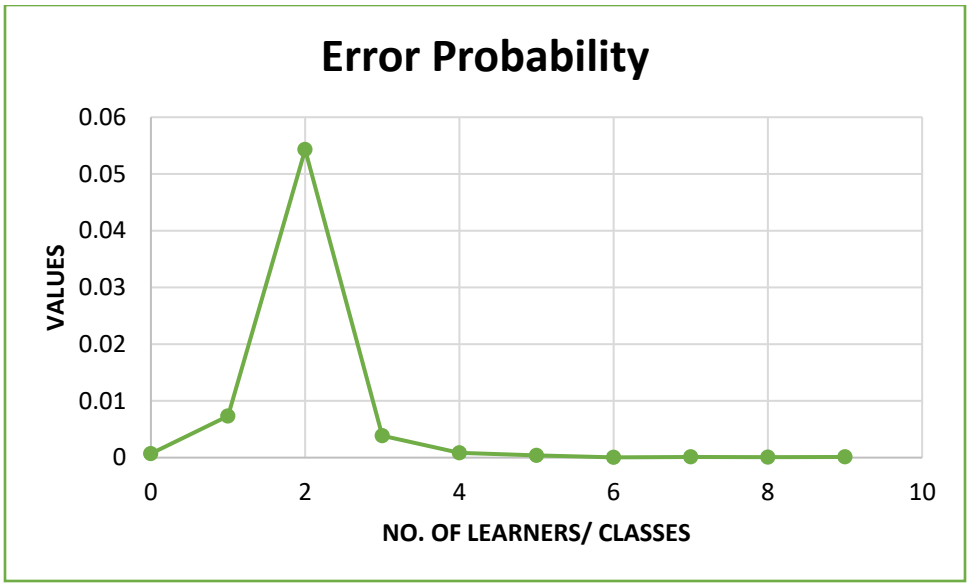
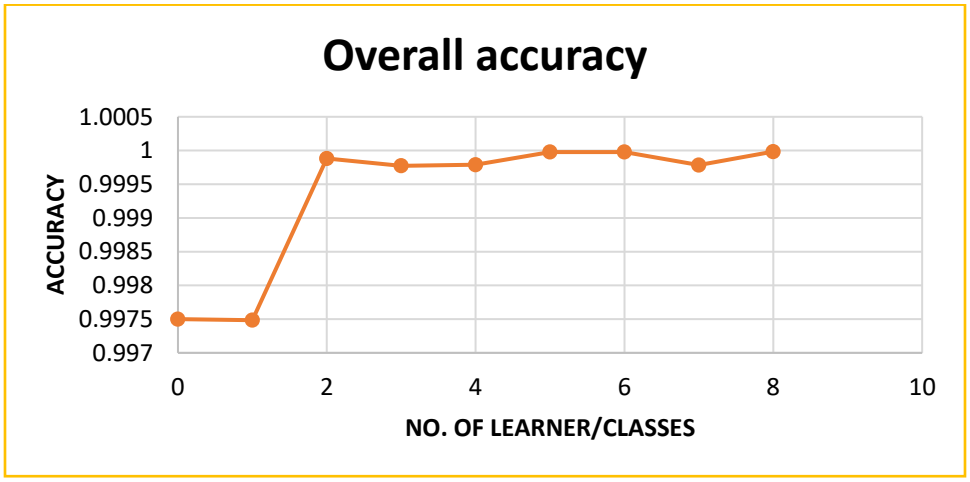


Figure 12: Simulated error probability of each class

Figure 21 shows the accuracy of the proposed model using a hybrid approach on 10 different attack types. The accuracy of the model is higher for some attack types than others, but overall the model is effective at detecting attacks.



0

ii. Dependability performance analysis:

In this section, we analyze the dependability performance of our proposed model, which encompasses availability, efficiency, and scalability. We carefully select features and apply our model to accurately classify both normal and attack scenarios. This ensures that our model is always available and performs well. We evaluate our model using metrics such as accuracy, precision, recall, F1 score, and roc. Our model outperforms several existing approaches and achieves this improved performance with minimal computational loss. This is illustrated in Figure 9.

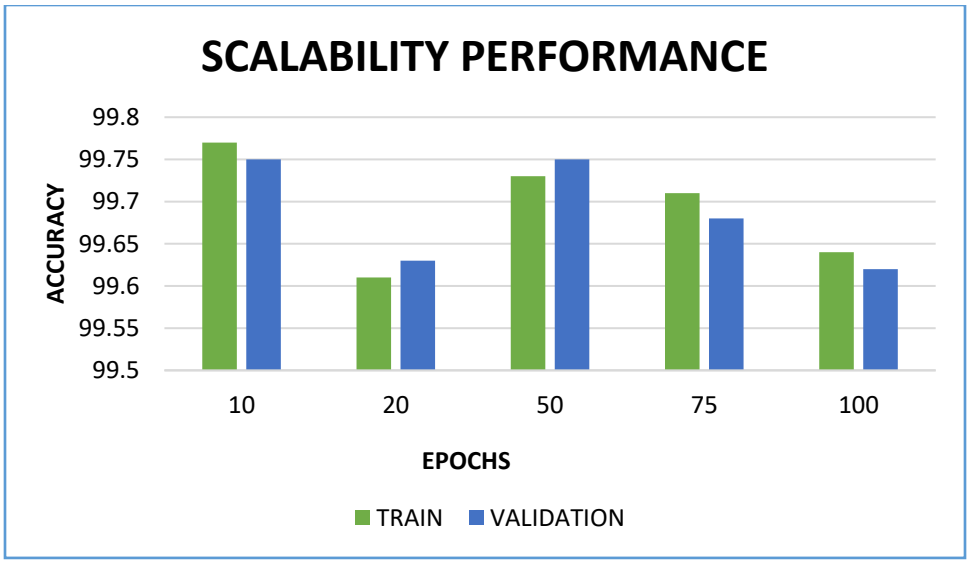


Figure 13: Scalability analysis of our proposed model (AttackNet)

Finally, we have observed enhanced scalability characteristics in our proposed model by incorporating diverse trusted data sources from various IoT devices into the training dataset. This inclusion ensures maximum consistency and a wider range of data. Remarkably, as we increased the epoch number from 10 to 100, the accuracy of our proposed model remained nearly unchanged, indicating its scalability. The figure 22 illustrates the scalability performance of our proposed model.

3.4. Dependable and Trustworthy CNN-GRU-Based Intrusion Detection System for IIoT

In this section, we present the design and development of a dependable and trustworthy Intrusion Detection System (IDS) for Industrial Internet of Things (IIoT) environments, leveraging a hybrid model that combines Convolutional Neural Networks (CNNs) and Gated Recurrent Units (GRUs). The proposed system, Alpha-Net, is designed to efficiently detect and mitigate cyber threats, such as network-based attacks by IoT botnets, in complex industrial networks. We discuss the key design elements of Alpha-Net, including its feature extraction techniques, the communication sequence architecture, and the statistical analysis used to validate its performance.

3.4.1 Design of the Alpha-Net IIoT Anomaly Detection Framework

1.1. Data Preparation and Feature Extraction

Data preparation is an important step in machine learning and deep learning workflows as it involves cleaning and organizing the data before feeding it into the algorithms. This process aims to enhance the learning process and improve the accuracy of the models. During data preparation, various operations can be performed, including feature selection, converting non-numerical features to numerical ones, and handling missing values appropriately. In our research, we adopted a two-step approach for Data Preparation, comprising Data Pre-processing and Data Normalization techniques as shown in algorithm 1:

1.1.1. **Data Pre-processing**: In the data pre-processing, categorical features with nominal values were transformed into numerical values using label encoding. This conversion ensured compatibility with the neural network's input requirements. Additionally, we eliminated irrelevant features such as data, time, and timestamp columns, as they did not contribute significantly to the output predictions.

1.1.2. Data Normalization: Data normalization was applied to address the issue of feature imbalance. Some attributes in the dataset had higher values than others, thus skewing the model's performance. To overcome this, we employed the min-max scaling technique for data normalization. This approach maps the data to a range between 0.0 and 1.0 while preserving the inherent distribution of the data. Mathematically, the min-max scaling formula is expressed as follows:

$$y = (x - xmin) / (xmax - xmin), \quad (9)$$

where x and y are the original and normalized values. The feature's minimum and maximum values are given by x_{min} and x_{max}, respectively.

Algorithm 1: Data Preprocessing for Enhanced Intrusion Detection System

1. **Input:** Obtain the raw dataset by reading from the specified path.

2. **Processing Steps:**

a. Introduce a new column, named "type," indicating the attack type.

b. Concatenate the individual datasets into a unified dataset with corresponding labels.

c. Display the dimensions of the resulting dataset: (N *M), where (N = 926157) and (M = 116).

d. Examine feature similarities within the dataset.

e. Eliminate columns with redundant features.

f. Perform data augmentation by introducing noise to the dataset, resulting in augmented data.

g. Display the dimensions of the augmented dataset: (N * M), now reduced to (926157 * 76).

h. Randomly permute the rows of the dataset to introduce variability.

i. Encode categorical labels.

j. Implement dummy encoding for categorical variables, considering each column and row.

k. Modify the dataset columns with the 'type' prefix for clarity.

l. Standardize the dataset by scaling each column with its mean and standard deviation.

3. **Output:** The preprocessed dataset is now ready for subsequent analysis.

4. **Dataset Splitting:**

a. Separate the dataset into training and testing sets using an 80:20 ratios.

5. **Output:** The dataset is split into training and testing subsets, facilitating model development and evaluation.

The data preprocessing algorithm 1 ensures that the raw input is transformed into a standardized, noise-augmented, and well-organized dataset, paving the way for robust analysis and model training in cybersecurity research.

4.3.3. Feature Selection: Feature selection plays a pivotal role in reducing the dimensionality of high-dimensional datasets while preserving relevant information. In this research, we propose a Quantum-Inspired Genetic Algorithm (QIGA) for efficient feature selection as shown in Algorithm 2. QIGA integrates principles of quantum computing, such as superposition and probability amplitudes, with genetic algorithms to identify optimal feature subsets. The following are the steps in the proposed methodology:

1. Quantum Representation of Features: Each feature is represented as a quantum bit (qubit), which exists in a superposition state defined as:

$$\varphi = \alpha\,|0> + \beta|1>$$

Where,α *and* β are probability amplitudes satisfying$|\alpha^2 + \beta^2 = 1, ensuring\ normalization\ and\ |0> and\ |1>$ $represent\ the\ states\ where\ a\ feature\ is\ excluded\ or\ included, respectively.$

The superposition enables simultaneous evaluation of all possible feature subsets, providing an efficient mechanism for feature exploration.

2. Fitness Evaluation: The fitness of each quantum chromosome is calculated using a classification model (CNN-GRU in our case). The fitness function is defined as:

$$F(x) = \frac{1}{1 + E(x)} + \lambda . R(x)$$

Where, $E(x)$: Classification error rate of the CNN-GRU model on the selected feature subset x, $R(x)$: Regularization term penalizes the selected subset's size to encourage simplicity, λ: Regularization weight balancing accuracy and simplicity.

3. Quantum Genetic Operators: Following are the operations performed:

a. Quantum Crossover: The crossover combines two parent quantum chromosomes, defined by their probability amplitudes $(\alpha_p, \beta_p)\ and\ (\alpha_q, \beta_q)$ to produce offspring:

$$\alpha_{offspring} = w_1 . \alpha_p + w_2 . \alpha_q, \beta_{offspring} = w_1 . \beta_p + w_2 . \beta_q$$

Where w_1 and w_2 are weight coefficients ensuring normalization.

b. Quantum Mutation: Mutation adjusts the probability amplitudes to explore new feature subsets:

$$\alpha' = \alpha . \cos(\emptyset) - \beta . \sin(\emptyset)$$
$$\beta' = \alpha . \sin(\emptyset) + \beta . \cos(\emptyset)$$

Where $\emptyset$ is the mutation angle controlling the extent of perturbation.

c. Quantum Measurement: Measurement collapses the quantum state into a classical state, determining the final feature subset for evaluation:

$$x_i = \begin{cases} 1, if\ |\beta_i|^2 > |\alpha_i|^2 \\ 0, \quad Otherwise \end{cases}$$

By applying QIGA, the dimensionality of the dataset is significantly reduced, with a feature subset that maximizes classification accuracy while minimizing computational overhead. The final selected features enable the CNN-GRU model to achieve superior performance metrics in anomaly detection.

Algorithm 2: Quantum-Inspired Genetic Algorithm for Feature Selection

1. **Input**: Dataset D, number of iterations T, population size P.
2. **Initialize**: Quantum chromosomes with random probability amplitudes.
3. **For** t = 1 to T:
   a. Evaluate fitness F(x) for each quantum chromosome.
   b. Perform quantum crossover and mutation.
   c. Collapse quantum states to select features.
4. **Output**: Optimal feature subset x^* with maximum fitness.

1.2. Proposed Model (Alpha-Net)

The block diagram of the proposed Intrusion Detection System (IDS) model named Alpha-Net comprises three main components, as shown in Figure 5. The first component is responsible for Data Preparation, where the input data is gathered, preprocessed, and made ready for further analysis. The second component involves the application of an Intrusion Detection Technique, which aims to identify potential intrusions or malicious activities within the network. Simultaneously, this phase incorporates Model Training, where the system learns from labeled data to enhance its ability to detect intrusions accurately. The final component of the model encompasses Network Data Evaluation. In this stage, the preprocessed and trained data are subjected to further analysis and refinement to improve its performance. The evaluation process assesses the efficiency and effectiveness of our proposed IDS model, utilizing various metrics and benchmarking methods to gauge its performance against existing models and standards.

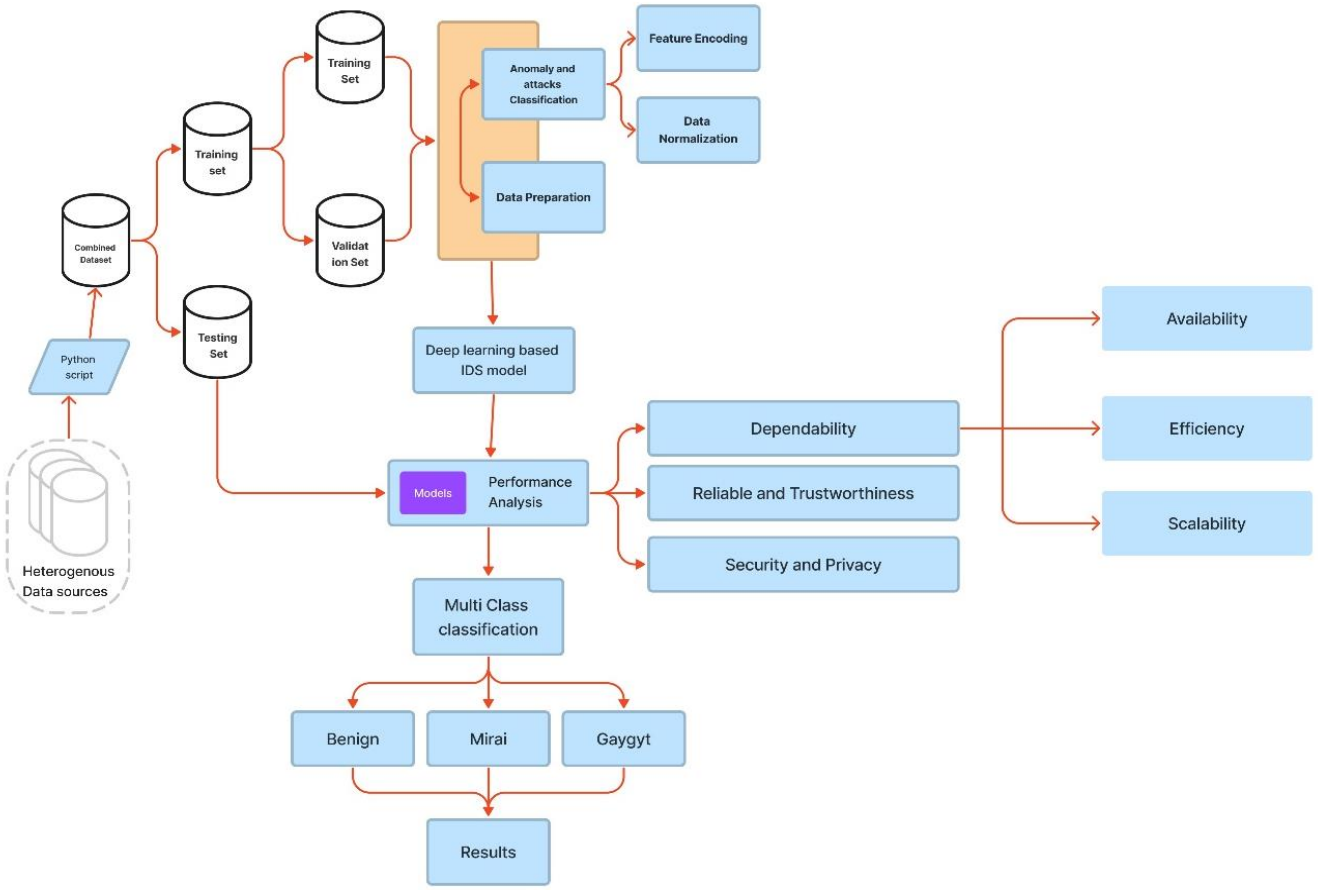


Figure 14: Block diagram of the proposed model on different performance analyses.

The primary objective of the proposed model is to enhance the security of IoT devices by safeguarding them against diverse attacks. A state-of-the-art and up-to-date dataset ensures comprehensive experimentation and evaluation of the model's performance. In Figure 6, the sequence diagram of IIoT illustrates the communication process between different layers in the Industrial Internet of Things (IIoT) architecture. This diagram provides valuable insights into how information and data flow through the layers, facilitating a better understanding of the communication dynamics within the IIoT framework.

Table 3: Hyper Parameter of the Proposed Model

Hyperparameter	Value/function
No. of Layers	8
ConV1D fitters	64,32
ConV1D Kemel Size	5
Map pooling ID	4
GRU Units	32,16
Activation function	Relu
Dropout	0.1
Dense Unit	128,64,64
Activation Function	Softmax

Loss Function	Categorical cross-entropy

The proposed model underwent extensive empirical experiments aimed at identifying key parameters. After multiple iterations, the model with the highest predictive performance on the training set was selected as the optimal intrusion detection model for diverse IoT applications. Training involved utilizing a randomly selected dataset, while validation was performed on a separate dataset to assess effectiveness. The model's detection performance was comprehensively analyzed by comparing various parameter settings. Rigorous testing was carried out to ensure the model's dependability, reliability, trustworthiness, security, and privacy. After thorough evaluation, the most robust model emerged as the final intrusion detection solution for heterogeneous IoT applications. Table 6 details the hyperparameters of the proposed model, specifically designed for intrusion detection in IoT/IIoT environments using CSV data. The model comprises different layers, utilizing 64 and 32 Conv1D filters with a kernel size of 5 to capture local features effectively. Max pooling with a size of 4 reduces dimensionality, while GRU units (32 and 16) capture temporal dependencies crucial for identifying intrusions. ReLU activation functions are used in the convolutional and dense layers to introduce non-linearity, and a dropout rate of 0.1 prevents overfitting by deactivating 10% of neurons during training. The dense layers (128, 64, 64 units) aggregate learned features, with the Softmax activation function in the output layer generating probability distributions for multi-class classification. The model employs categorical cross-entropy as the loss function to measure the performance by comparing predicted probabilities with actual class labels. These hyperparameters are optimized to balance complexity and performance, ensuring efficient memory usage on a system with 8GB RAM while achieving high accuracy and robustness in detecting and classifying botnet attacks.

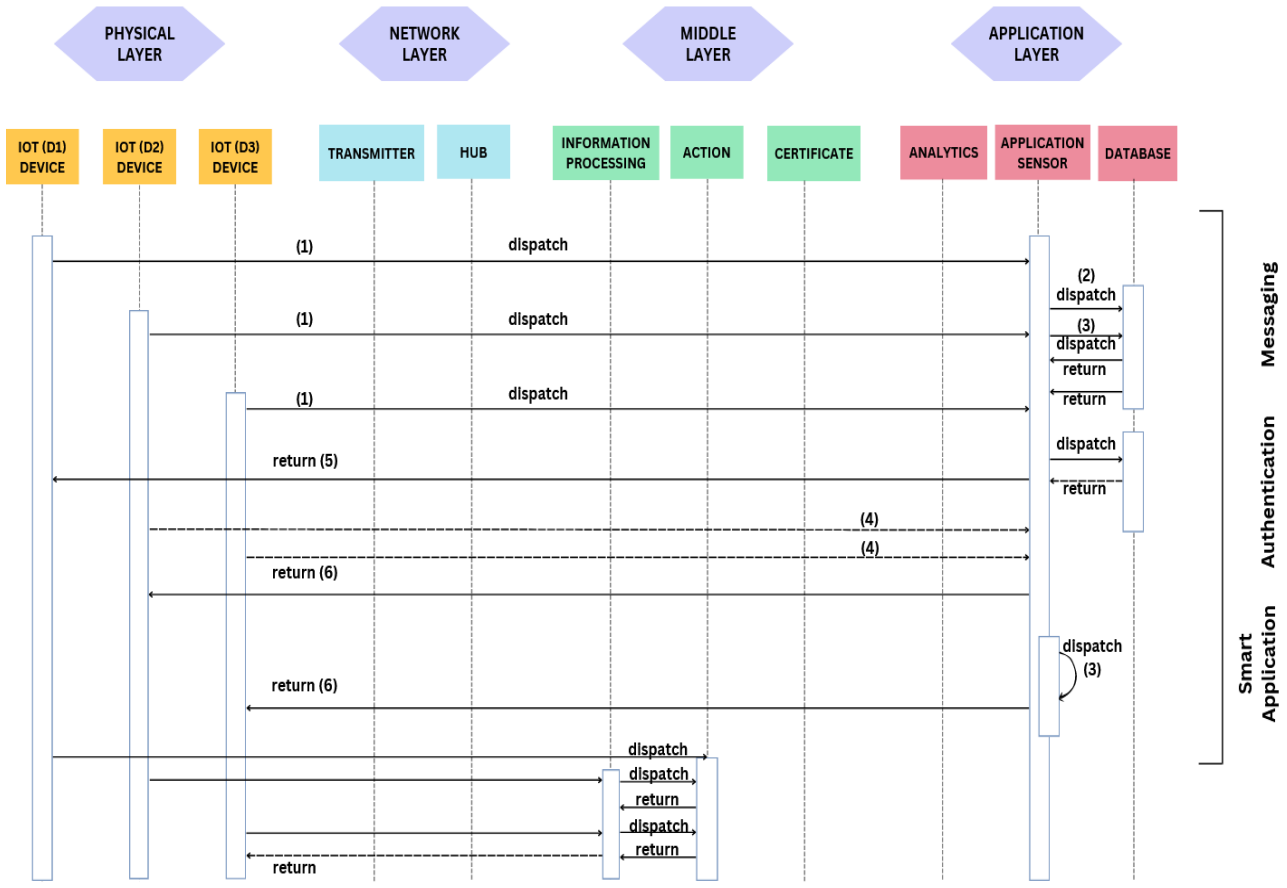


Figure 15: Sequence diagram of proposed IIoT-based IDS monitoring system.

The Network architecture of the proposed model is shown in Figure 7. The input of the network is the same shape. Define the input sequence X as a metric of size (A,1), where A is the length of the sequence:

$$X = [x(1), x(2), x(3), \ldots . x(A)] \quad (10)$$

The **first layer** in the model is a 1D CNN layer that performs cross-correlation operations on the input data.This layer helps extract local spatial features from the input data, such as patterns and anomalies that can differentiate between normal and attack traffic, the output can be obtained as:

$$Y[1][i] = RELU\ (\in_j = 1,2,3,4,5\ W[1][j] * X[i + j - 3] + v[1]) \quad (11)$$

Where weight matrix W[1], biasvector v[1], output sequence Y[1] and i = 1<x<A.

The Second 1D CNN layer further captures higher-level spatial features from the output of the first layer. It learns to detect more Complex patterns and correlations between different spatial regions of the input data. The output sequence can be obtained as:

$$Y[2][i] = RELU\ (\in_j = 1,2,3,4,5\ W[2][j] * Y[1][i + j - 3] + v[1]) \quad (12)$$

Where weight matrix W[2], biasvector v[2], output sequence Y[2] and i = 1<x<A.

The max pooling layer reduces the spatial dimensions of the input data by selecting the maximum value within each pooling window. This operation has to abstract the spatial information focusing on the most salient feature and reducing computational complexity. It also aids in achieving translation in variance by capturing relevant information, the output sequence can be obtained as:

$$Y[3][i] = \max(\ Y[2][4 * (i - 1) + 1]\ \ , Y[2][4 * (i - 1) + 2]\ \ , Y[2][4 * (i - 1) + 3]\ \ , Y[2][4 * (i - 1) + 4]\ ) \quad (13)$$

The GRU layers, which consist of reset, update, and new Gates along with the hidden States calculations, are coxial capturing the temporal dependencies in the input data. For attack classification, these layers learn to model the temporal patterns and dynamics over time, considering the sequential nature of the network traffic. The **fourth layer,** with 32 units defining the output sequence as:

$$r(t) = \sigma(W[3][r] * h(t - 1) + u[3][r] * Y[3][t] + v[3][r]) \quad (14)$$

$$z(t) = \sigma(W[3][z] * h(t - 1) + u[3][z] * Y[3][t] + v[3][z]) \quad (15)$$

$$h'(t) = tanh(W[3][h] * (r(t) * h(t - 1)) + u[3][h] * Y[3][t] + v[3][h]) \quad (16)$$

$$h(t) = (1 - z(t) * h(t - 1) + z(t) * h'(t)) \quad (17)$$

The **fifth layer** is GRU layer with 16 units, defining the output sequences:

$$r(t) = \sigma(W[4][r] * h'(t - 1) + u[4][r] * h[A/4] + v[3][r]) \quad (18)$$

$$z(t) = \sigma(W[4][z] * h'(t - 1) + u[4][z] * h[A/4] + v[3][z]) \quad (19)$$

$$h'(t) = tanh(W[4][h] * (r(t) * h'(t - 1)) + u[4][h] * h[A/4] + v[4][h]) \quad (20)$$

$$h(t) = (1 - z(t) * h'(t - 1) + z(t) * h''(t)) \quad (21)$$

The **sixth layer** of the model is a flattening layer, which reshapes the multidimensional output from the GRU layers into a one-dimensional tensor. This conversion preserves the temporal sequence of the information while reformatting it to make it compatible with subsequent fully connected layers. The **seventh, eighth, and ninth layers** are fully connected layers designed to extract high-level features and classify patterns identified in the preceding layers. These layers leverage matrix operations and the ReLU activation function to model intricate relationships among features, enabling accurate predictions. The **tenth layer** is a dropout layer, implemented with a dropout rate of 0.1, which helps regularize the model by randomly deactivating a fraction of neurons during training. This approach minimizes overfitting by encouraging the network to generalize better and avoid reliance on specific neurons.

The **final layer** is a fully connected output layer utilizing a Softmax activation function. The Softmax function converts the layer's outputs into a probability distribution across the target classes. For instance, in a scenario with 15 classes, the output will be a 15-dimensional vector where each element represents the probability of the input belonging to a particular class.

$$h_1 = Flatten(h_t) \quad (22)$$

$$z_1 = W_1 h_1 + b_1 \quad (23)$$

$$a_1 = g(z_1) \quad (24)$$
$$z_2 = W_2 a_1 + b_2 \quad (25)$$
$$a_2 = g(z_2) \quad (26)$$
$$d_1 = Dropout(a_2) \quad (27)$$
$$z_3 = W_3 d_1 + b_3 \quad (28)$$
$$a_3 = ReLu(z_3) \quad (29)$$
$$z_4 = W_4 a_3 + b_4 \quad (30)$$
$$a_4 = ReLu(z_4) \quad (31)$$
$$y = softmax(W_5 a_4 + b_5) \quad (32)$$

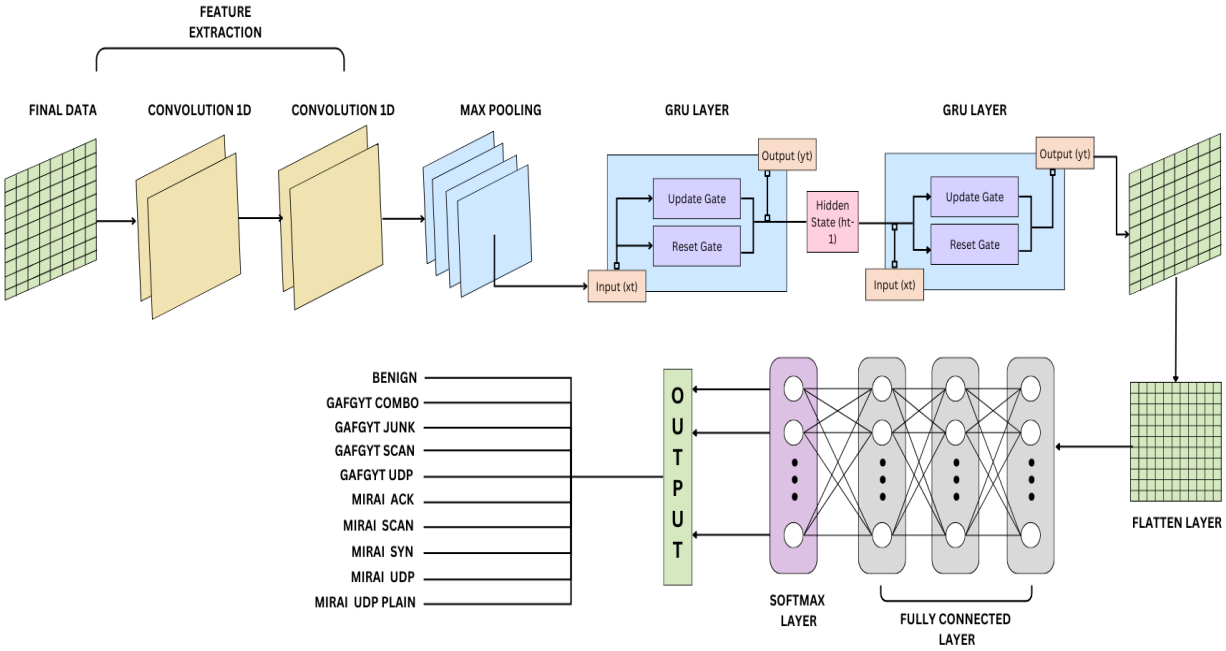


Figure 16: Network layered architecture of the proposed model (Alpha-Net).

The proposed model, Alpha-Net, integrates Convolutional Neural Networks (CNN) and Gated Recurrent Units (GRU) to handle the complexities of intrusion detection in Industrial Internet of Things (IIoT) environments. The combination of CNN and GRU allows the model to effectively capture spatial and temporal dependencies in the data, which is crucial for identifying attack patterns in network traffic. CNNs excel in extracting spatial features by identifying patterns in the dataset columns, which is vital for recognizing anomalies and distinguishing between normal and malicious traffic. Specifically, the 1D CNN layers are responsible for extracting local spatial features. In contrast, the second CNN layer refines these features, enabling the model to capture more complex patterns across different regions of the input data. Max pooling is then applied to reduce the dimensionality of the feature maps, focusing on the most salient information and improving computational efficiency.

On the other hand, GRUs handle the temporal aspects of the data, capturing the sequential nature of the network traffic. These units model temporal dependencies by learning from previous time steps and identifying how patterns evolve. This is particularly important in intrusion detection, where attacks often follow temporal patterns or exhibit periodic behaviors that need to be recognized over time. With their gating mechanisms (reset, update, and new gates), the GRU layers are designed to capture these dynamic temporal changes in the input data, making them highly effective for sequential data. By combining CNNs and GRUs, Alpha-Net benefits from the ability to capture both spatial features and temporal dependencies simultaneously. This synergy makes the model highly adaptable to various dataset structures, such as those found in IIoT environments and enhances its ability to detect intrusions with high accuracy. While CNNs alone can identify local patterns and anomalies in data, and GRUs alone are proficient at handling

sequential data, the hybrid architecture of Alpha-Net leverages the strengths of both approaches, ensuring robust performance across different types of data and attack scenarios. This combination significantly improves the model's ability to detect attacks in various IIoT applications.

Algorithm 3 entails constructing a sequential neural network model with distinct layers, including Conv1D and GRU layers followed by Flatten and Dense layers. This model is configured with a designated optimizer, loss function, and metric before training on the provided training data. Subsequently, the model's performance is evaluated using a separate test data set, and its predictive capabilities are applied to new data. The evaluation involves assessing metrics such as accuracy to gauge the model's effectiveness.

Algorithm 3: Proposed Alpha-Net Model

```
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Conv1D, MaxPooling1D, GRU, Flatten, Dense, Dropout
from tensorflow.keras.optimizers import Adam
# Function to build the Alpha-Net Model
def build_attacknet_model(train_data, labels, val_data, val_labels, num_classes=10):
    # Step 1: Define the model architecture
    model = Sequential()
    # Add Convolutional Layers for feature extraction
    model.add(Conv1D(64, kernel_size=5, strides=1, padding='same', activation='relu'))
    model.add(Conv1D(32, kernel_size=5, strides=1, padding='same', activation='relu'))
    model.add(MaxPooling1D(pool_size=4))
    # Add GRU Layers for temporal feature modeling
    model.add(GRU(32, activation='relu', return_sequences=True))
    model.add(GRU(16, return_sequences=True))
    # Flatten the output
    model.add(Flatten())
    # Add Fully Connected (Dense) Layers for classification
    model.add(Dense(128, activation='relu'))
    model.add(Dense(64, activation='relu'))
    model.add(Dropout(0.1))
    model.add(Dense(num_classes, activation='softmax'))
    # Step 2: Compile the model
```

```
    optimizer = Adam()
    loss = 'categorical_crossentropy'
    metrics = ['accuracy']
    model.compile(optimizer=optimizer, loss=loss, metrics=metrics)
    # Step 3: Train the model
    epochs = 10
    batch_size = 32
    validation_data = (val_data, val_labels)
    for epoch in range(epochs):
        if epoch == 0:
            model.fit(train_data, labels, epochs=1, batch_size=batch_size, validation_data=validation_data)
        else:
            model.fit(train_data, labels, epochs=1, batch_size=batch_size)
    # Step 4: Evaluate the model
    test_data, test_labels = load_test_data()  # Replace with actual test data loader
    loss, accuracy = model.evaluate(test_data, test_labels)
    # Step 5: Use the model for predictions
    new_data = load_new_data()  # Replace with actual data loader
    predictions = model.predict(new_data)
    return model, accuracy, predictions
# Placeholder functions for loading test and new data
def load_test_data():
    # Replace this with the actual logic to load test data
    return None, None
def load_new_data():
    # Replace this with the actual logic to load new data for predictions
    return None
```

Result Analysis

The proposed Alpha-Net model underwent rigorous evaluation across several standard metrics, including accuracy, precision, recall, F1 score, loss, and the receiver operating characteristic (ROC) area under the curve (AUC). These metrics collectively ensured a comprehensive assessment of the model's effectiveness. The model achieved exceptional training and testing accuracies of 99.98% and 99.97%, respectively, with a minimal loss of 0.0014, showcasing its robustness. Table 8 summarizes the performance metrics for the training, validation, and test datasets, demonstrating consistently high precision, recall, and F1 scores across all evaluation phases.

The Alpha-Net model comprises a hybrid architecture integrating Convolutional Neural Networks (CNN) and Gated Recurrent Unit (GRU) layers. The model includes four hidden layers: two convolutional layers for feature extraction and two GRU layers for sequential data modeling. The Rectified Linear Unit (ReLU) activation function was employed in the hidden layers, while the Softmax function served as the output activation function. Training optimization utilized the Adam optimizer alongside the categorical cross-entropy loss function, facilitating rapid convergence and performance improvements.

The confusion matrix as shown in Figure 8 offers a granular view of the model's classification performance, illustrating high accuracy in distinguishing between various classes. Misclassification rates were minimal, reflecting the model's efficacy in precision identifying benign and attack classes.

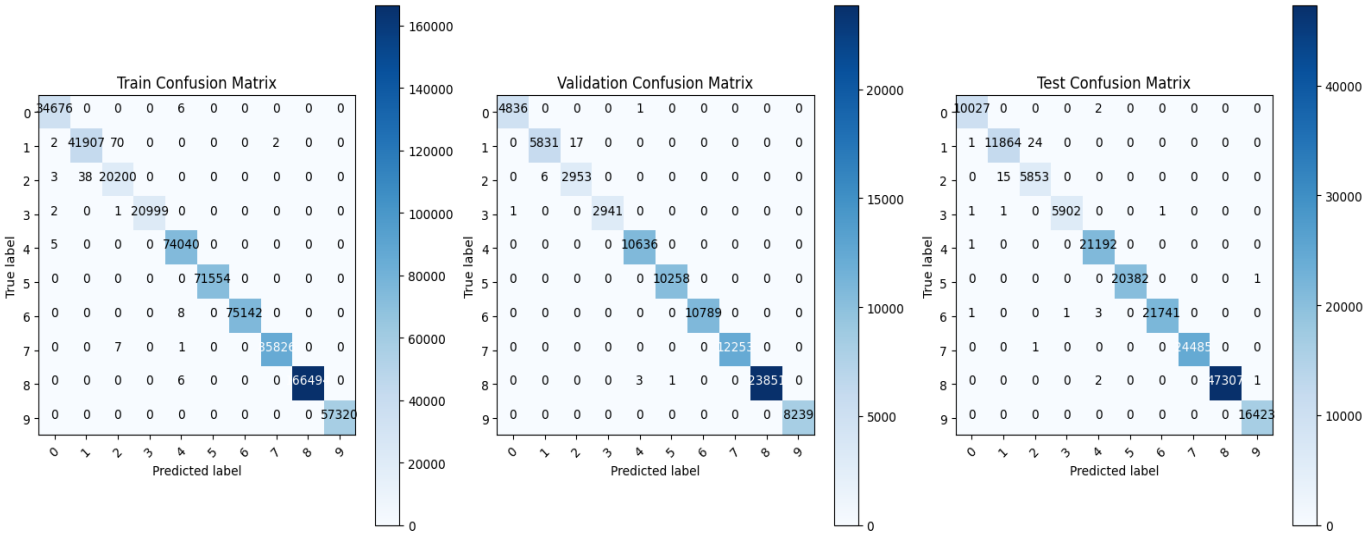


Figure 17: Confusion Matrix of the Proposed Model Alpha-Net.

Table 8 showcases the results of our experimentation on the N_BaIoT dataset, presenting accuracy, precision, recall, and F1 score metrics.

Table 4: Experimental Result of Proposed Model on train, validation, and test sets

Parameters	Loss	Accuracy	Precision	Recall	F1 score
Train set	0.0011	0.9998	0.9997	0.9994	0.9995
Validation set	0.0010	0.9999	0.9998	0.9996	0.9996
Test set	0.0014	0.9997	0.9995	0.9994	0.9994

Our proposed model has remarkable performance gains, as illustrated in Figure 9. Specifically, our Alpha-Net model outperformed contemporary algorithms, achieving an impressive 99.97% detection accuracy. Furthermore, the proposed model demonstrated exceptional precision, recall, and F1 score, all at 99.95%, 99.94%, and 99.94%, respectively, with a minimal loss of 0.0015.

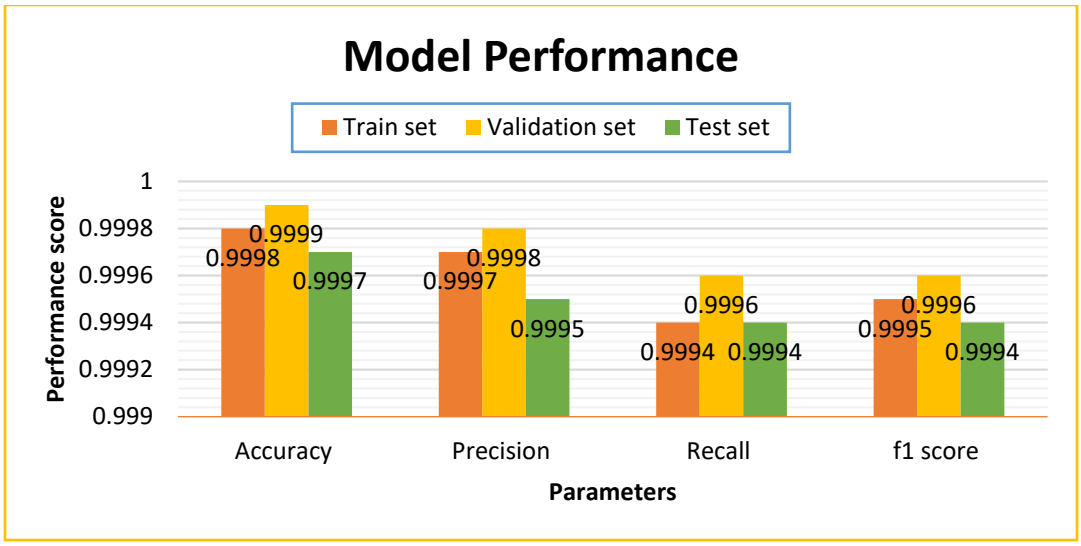


Figure 18: Performance evaluation of the Proposed model on the Dataset.

To visualize the training and validation performance, we plotted the corresponding metrics against the number of training epochs. Figures 10 (a) and (b) show both the training and validation accuracies increased steadily while the losses decreased. This indicates that the model could learn the dataset's features and generalize well.

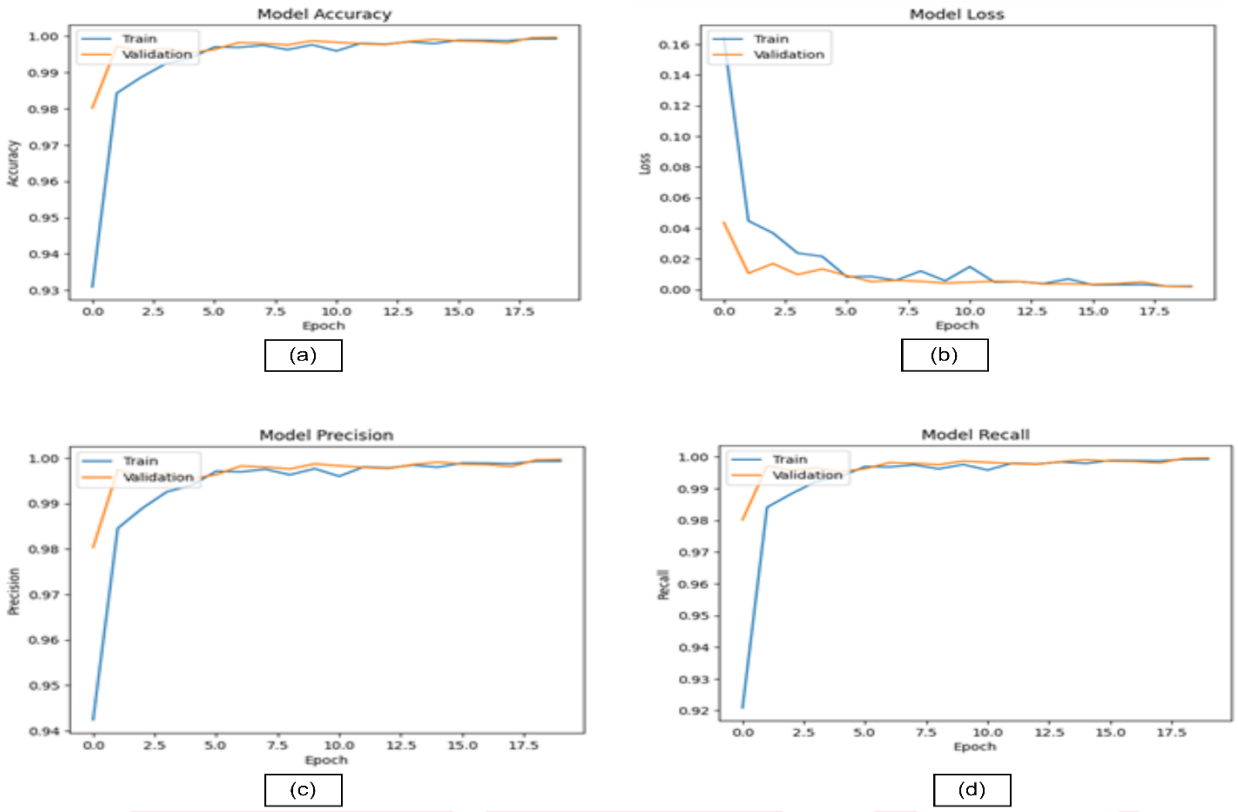


Figure 19: (a) Training and validation accuracy (b) Training and validation loss (c) Training and validation Precision (d) Training and validation Recall of the Proposed Model.

Table 9 provides a comprehensive breakdown of precision, recall, and F1-score metrics across different classes, offering a detailed assessment of the model's performance. Each row corresponds to a specific class, while the columns detail the corresponding precision,

recall, and F1-score values. Precision measures the accuracy of positive predictions, indicating the proportion of correctly predicted positive instances among all instances classified as positive for a particular class. Conversely, recall quantifies the model's ability to correctly identify all positive instances within a class, representing the proportion of true positives correctly classified. F1-score is a harmonic mean of precision and recall, providing a balanced evaluation of the model's performance by considering both false positives and false negatives. Class 0 achieves perfect scores with precision, recall, and an F1-score of 1, indicating flawless classification without false positives or negatives. Class 1 demonstrates slightly lower precision at 0.97 but maintains a perfect recall of 1, resulting in an F1-score of 0.98, which suggests high accuracy with a minor presence of false positives. For Class 2, the precision is high at 0.99, though recall drops to 0.95, producing an F1-score of 0.97, highlighting good performance with some false negatives. Class 3 exhibits near-perfect results with a precision and recall of 1 and 0.99, respectively, culminating in an F1-score of 0.99. Classes 4, 6, 7, and 9 attain perfect scores across all metrics, demonstrating impeccable model performance in these categories. Class 5 has a precision of 0.98 and a perfect recall of 1, yielding an F1-score of 0.99, indicating strong performance with minimal false positives. Class 8 achieves perfect precision and an F1-score of 1, with a slightly lower recall of 0.99, signifying a very high accuracy with occasional false negatives. These metrics collectively illustrate the model's overall high effectiveness in classifying instances accurately across different classes. The graphical representation of the results is shown in Figure 11.

Table 5: Performance Evaluation Metrics of Proposed Model on Different Attack Classes in the Test Set

Type of attack	Class Type	Precision	Recall	F1-score
benign	Class 0	1	1	1
mirai_udp	Class 1	0.97	1	0.98
gafgyt_combo	Class 2	0.99	0.95	0.97
gafgyt_junk	Class 3	1	0.99	0.99
gafgyt_scan	Class 4	1	1	1
gafgyt_udp	Class 5	0.98	1	0.99
mirai_ack	Class 6	1	1	1
mirai_scan	Class 7	1	1	1
mirai_syn	Class 8	1	0.99	1
mirai_udpplain	Class 9	1	1	1

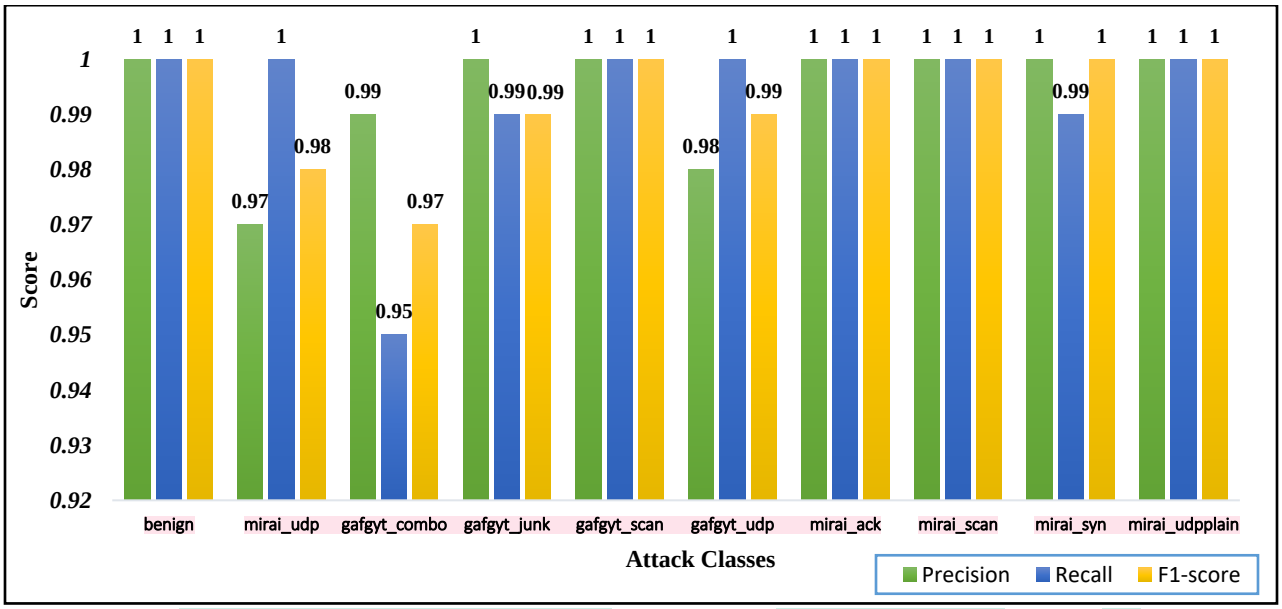


Figure 20: Precision, Recall, and F1-Score Evaluation of Proposed Model Across Different Attack Classes in the Test Set

An efficient and effective model is characterized by low prediction values for FPR, FNR, FDR, and FOR. FPR calculates the relationship between correctly classified known attack samples and total attack data. FDR is a statistical metric used in testing to consider various differences. FOR complements PPV and NPV by measuring the proportion of false negatives incorrectly rejected. FNR represents the percentage of benign records that were mistakenly identified. In our study, the CNN-GRU model achieved impressive values of 0.00321% for FPR, 0.05796% for FDR, 0.066% for FNR, and 0.00426% for FOR, as illustrated in Figure 12.

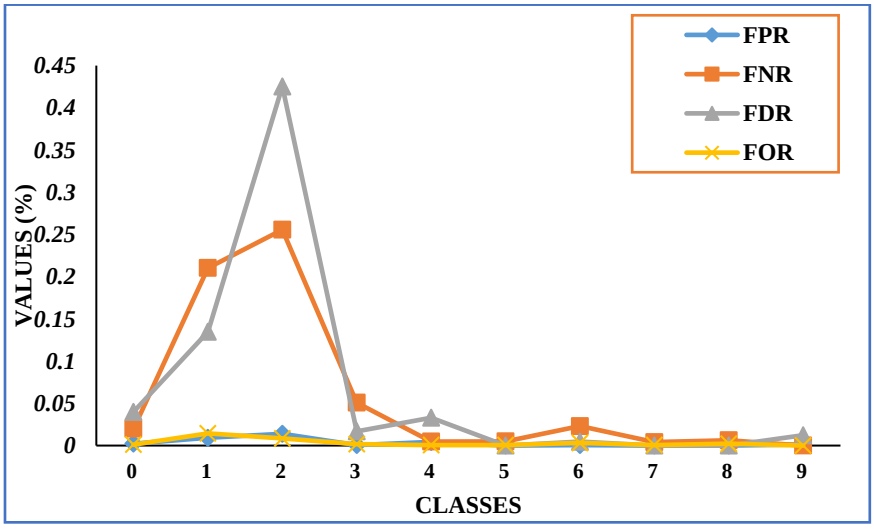


Figure 21: FPR, FNR, FDR, and FOR value of the proposed model.

In addition, as shown in Figure 13, we calculated additional parameters such as the true negative rate (TNR), Mathew correlation coefficient (MCC), and negative predictive value (NPV). Our suggested CNN-GRU model achieves 99.99% TNR, 99.78% MCC, and 99.99% NPV, respectively. We plotted the ROC curve in Figure 13 to further analyze the model's discriminative capabilities between

attack and normal classes. This graphical representation demonstrates the model's ability to distinguish between attack types and normal cases.

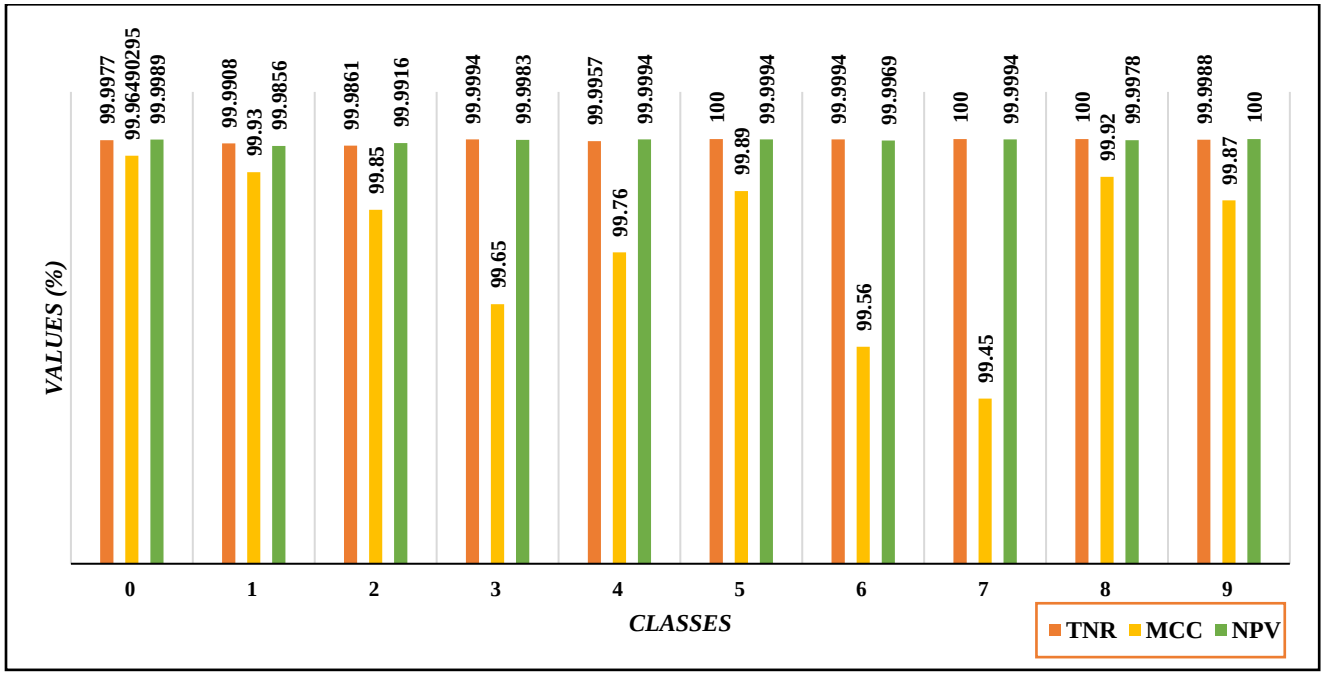


Figure 22: TNR, MCC, and NPV score of the proposed model.

Table 10 presents a detailed breakdown of key performance metrics for a classification model on a class-wise basis. False Positive Rate (FPR) gauges the percentage of actual negatives mistakenly classified as positives, while False Negative Rate (FNR) denotes the percentage of actual positives erroneously labeled as negatives. False Discovery Rate (FDR) expresses the ratio of false positives to the total predicted positives. False Omission Rate (FOR) represents the ratio of false negatives to the total predicted negatives. True Negative Rate (TNR) indicates the percentage of actual negatives accurately identified. The Matthews Correlation Coefficient (MCC) provides a balanced measure, considering all four confusion matrix values. Negative Predictive Value (NPV) reflects the percentage of actual negatives correctly identified among predicted negatives. For class 0, the model demonstrates a remarkably low FPR of 0.0023 and an FNR of 0.0199, coupled with a high TNR of 99.9977 and NPV of 99.9989, indicating excellent performance. Class 1 shows a slightly higher FPR of 0.0092 and a notable FNR of 0.2103, suggesting room for improvement in detecting positive instances, although its TNR remains high at 99.9908. Class 2's metrics reflect a higher FDR of 0.4253 and FNR of 0.2556, pointing to challenges in accurately identifying true positives despite maintaining a TNR of 99.9861. For class 3, the FPR and FDR are minimal at 0.0006 and 0.0169, respectively, with a solid NPV of 99.9983. Class 4 exhibits low error rates with an FPR of 0.0043, an FNR of 0.0047, and a TNR of 99.9957. Class 5 has an impeccable FPR of 0 and a minimal FNR of 0.0049, ensuring a TNR of 100. Class 6 displays low FPR and FDR values at 0.0006 and 0.0046, maintaining a TNR of 99.9994. Class 7, similar to class 5, has an FPR of 0 and an FNR of 0.0041, with a TNR of 100. Class 8 also reports an FPR of 0 and a low FNR of 0.0063, sustaining a TNR of 100. Finally, class 9 shows an extremely low FPR of 0.0012 and an FNR of 0, alongside a TNR of 99.9988 and an NPV of 100. This detailed breakdown enables a nuanced understanding of the model's strengths across various classification categories.

Table 6: Class-Specific Evaluation Metrics: In-Depth Analysis of Performance

Classes	FPR	FNR	FDR	FOR	TNR	MCC	NPV
0	0.0023	0.0199	0.0399	0.0011	99.9977	99.96490295	99.9989
1	0.0092	0.2103	0.1347	0.0144	99.9908	99.93	99.9856
2	0.0139	0.2556	0.4253	0.0084	99.9861	99.85	99.9916
3	0.0006	0.0508	0.0169	0.0017	99.9994	99.65	99.9983
4	0.0043	0.0047	0.033	0.0006	99.9957	99.76	99.9994
5	0	0.0049	0	0.0006	100	99.89	99.9994
6	0.0006	0.023	0.0046	0.0031	99.9994	99.56	99.9969
7	0	0.0041	0	0.0006	100	99.45	99.9994
8	0	0.0063	0	0.0022	100	99.92	99.9978

9	0.0012	0	0.0122	0	99.9988	99.87	100

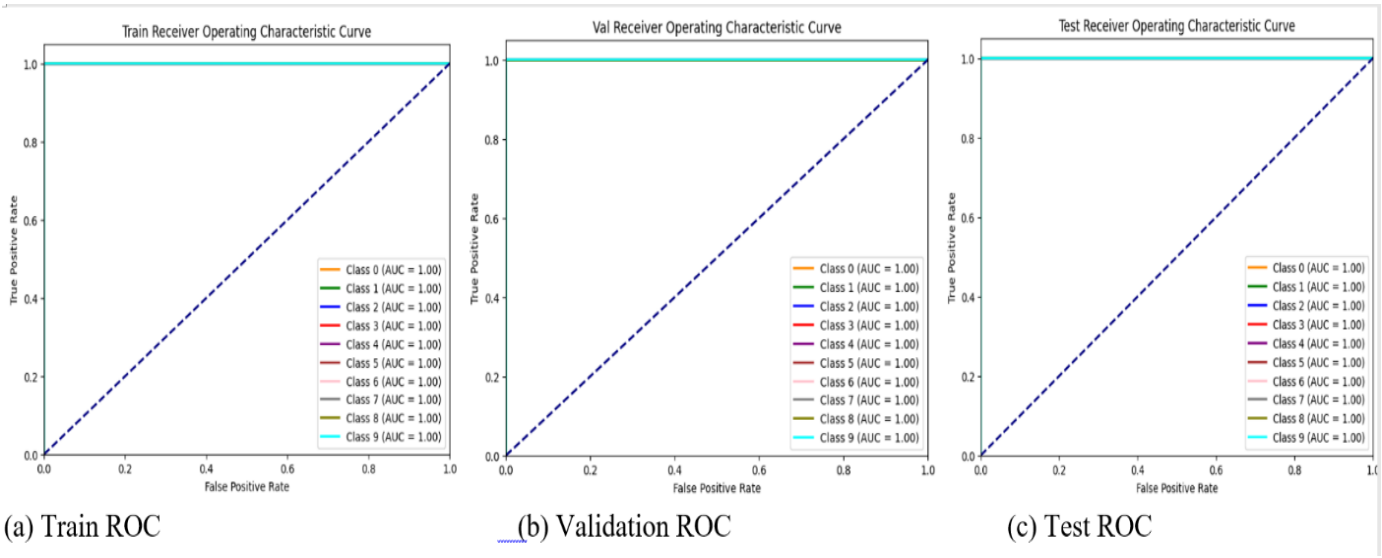


Figure 23: ROC Curve of the Proposed Model.

In summary, the proposed Alpha-Net model demonstrated outstanding performance in accurately classifying a wide range of botnet attacks. It consistently achieved high precision, recall, and F1 scores across all attack classes, reflecting a well-balanced trade-off between accuracy and the ability to correctly identify relevant instances. A thorough analysis of the confusion matrix and advanced metrics offered valuable insights into the model's effectiveness, guiding further optimization. Overall, Alpha-Net stands out as a highly robust and reliable solution for the intricate task of botnet attack classification.

1.3. Ablation Study: Deconstructing Model Efficacy

This section presents a detailed ablation study to assess the contributions of individual components in the proposed Alpha-Net model. The study systematically evaluates the architecture while incorporating Quantum-Inspired Genetic Algorithm (QIGA) for feature extraction in all scenarios. QIGA is a robust feature selection mechanism that reduces data dimensionality while preserving relevant information. The following cases are analyzed include:

- **Case 1 (QIGA + 1D CNN):** QIGA is used for feature extraction, followed by a single-layer 1D Convolutional Neural Network (CNN) for spatial feature extraction, serving as a baseline.
- **Case 2 (QIGA + Deep 1D CNN):** QIGA is combined with a deeper architecture of three layers of 1D CNN to enhance spatial feature extraction.
- **Case 3 (QIGA + Simple LSTM):** QIGA is integrated with a simple Long Short-Term Memory (LSTM) model to capture temporal dependencies, providing a comparative baseline for sequence modeling.
- **Case 4 (QIGA + GRU):** QIGA is used with a Gated Recurrent Unit (GRU) for temporal feature extraction, emphasizing GRU's efficiency over LSTM.
- **Case 5 (QIGA + CNN + GRU):** A hybrid model combines QIGA for feature extraction, 1D CNN for spatial feature extraction, and GRU for temporal dynamics.
- **Case 6 (QIGA + Proposed Model Alpha-Net):** The full Alpha-Net model incorporates QIGA for feature extraction, optimized CNN layers for spatial feature extraction, GRU for temporal dynamics, and additional preprocessing and training optimizations.

Table 7: Ablation Study: Comparative Analysis of Training Strategies and Performance Metrics

Ablation study cases	Cases	Accuracy	Precision	Recall	F1-score	Loss

QIGA + 1D CNN	1	0.84	0.8	0.79	0.795	0.0058
QIGA + Deep 1D CNN	2	0.89	0.85	0.83	0.84	0.0052
QIGA + Simple LSTM	3	0.91	0.88	0.87	0.875	0.0032
QIGA + GRU	4	0.93	0.91	0.91	0.91	0.0028
QIGA + CNN + GRU	5	0.97	0.96	0.95	0.955	0.0020
QIGA + Proposed Model Alpha-Net	6	**0.9997**	**0.9995**	**0.9994**	**0.9994**	**0.0014**

The results of the ablation study demonstrate the significant impact of combining the Quantum-Inspired Genetic Algorithm (QIGA) with various spatial and temporal feature extraction techniques on model performance as shown in Table 11. In Case 1, the single-layer 1D CNN serves as a baseline, where QIGA enhances the feature selection process. However, the shallow architecture limits the ability to capture complex spatial patterns, resulting in lower accuracy. Case 2 improves performance by employing a more profound CNN architecture, allowing for better spatial feature representation. Case 3, using QIGA with a simple LSTM, achieves better results than CNN-based approaches by effectively modeling temporal dependencies.

In contrast, Case 4 further improves performance by leveraging GRU's efficiency in capturing temporal dynamics with fewer parameters than LSTM. Case 5, a hybrid of QIGA, CNN, and GRU, integrates the strengths of spatial and temporal feature extraction, leading to substantial gains. Finally, Case 6, the proposed Alpha-Net model, incorporates QIGA with optimized CNN layers, GRU, and additional training enhancements, achieving near-perfect metrics. The observed performance improvements across cases are primarily attributed to the synergistic effect of QIGA's robust feature selection, deeper architectures for spatial feature extraction, and advanced temporal models like GRU, which collectively optimize the learning process and reduce noise, enhance the model's ability to generalize effectively.

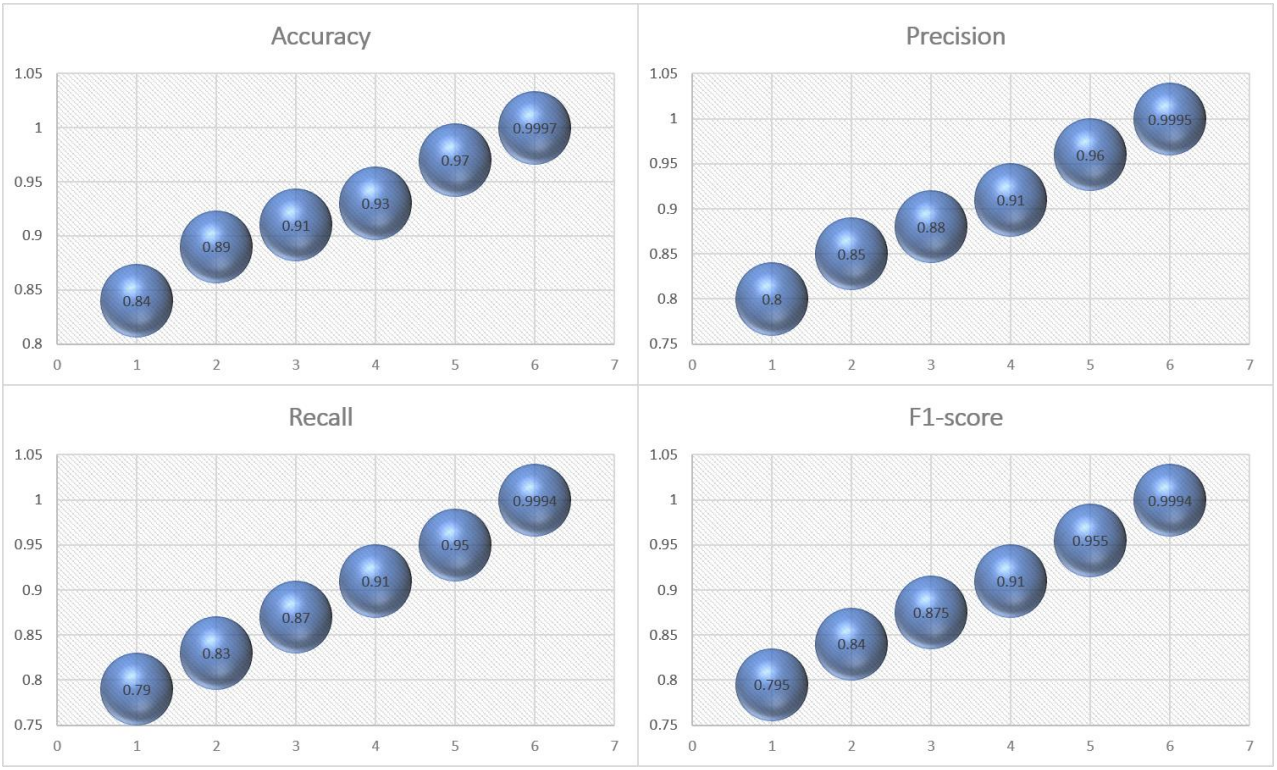


Figure 24: Ablation study results on different Performance evaluation parameters on the N_BaIoT dataset.

Figure 15 visually represents the ablation results, showing progressive improvements in accuracy, precision, recall, and F1-score across the cases. The results underscore the importance of combining optimized spatial and temporal feature extraction techniques in the Alpha-Net model, culminating in superior anomaly detection performance for IIoT systems.

1.4. Complexity Analysis of Proposed Model (Alpha-Net)

1.4.1. Time Complexity

The time complexity of Alpha-Net is intrinsically tied to the number of trainable parameters within its architecture, which are determined by hyperparameters such as the number of filters in convolutional layers, the dimensions of these filters, the units in the Gated Recurrent Unit (GRU) layers, and the size of dense layers.

Training the model involves iterative computations to minimize the loss function by updating the parameters through gradient calculations. This process necessitates evaluating the model at each step, making the computational time proportional to the number of trainable parameters. Additionally, the total training time is influenced by the number of epochs and the size of the training dataset, which dictate the number of training steps required for convergence. Table 12 provides a detailed breakdown of the computational time metrics for the training, validation, and testing phases of Alpha-Net, highlighting the model's temporal efficiency.

Table 8: Temporal Profiling: Computational Time Metrics for Training, Validation, and Testing Data Sets

Dataset	Computational Time
Train set	170s 2ms/step
Validation set	28s 2ms/step
Test set	43s 2ms/step

1.4.2. Space Complexity

The space complexity of Alpha-Net is primarily dictated by the memory requirements for its trainable parameters. Each parameter requires dedicated memory allocation, and the total memory usage scales linearly according to the number of parameters. Beyond this, additional memory is needed to store input data, intermediate activations, and gradients during the training process. These requirements depend on the size of the input data, the number of units in the model, and the total number of training steps required for convergence.

The model exhibits substantial spatial demands, given the expansive input data and the high number of trainable parameters. However, using advanced computational resources, such as Graphics Processing Units (GPUs), and optimized algorithms can significantly mitigate these challenges. Furthermore, regularization techniques such as weight decay and training strategies like early stopping reduce the risk of overfitting and decrease the number of parameters, leading to improved temporal and spatial efficiency.

In conclusion, while Alpha-Net's computational demands are significant, strategic optimization techniques and hardware acceleration render it a viable and scalable solution for complex botnet attack classification tasks..

1.5. Comparison

1.5.1. With the well-known DL/DLT-based IDS

To assess the efficacy and efficiency of our proposed model, we conducted a comparative analysis with well-known DL and DLT-based algorithms on the N_BaIoT dataset. These algorithms encompassed Neural Network (NN), Convolutional Neural Network (CNN), Recurrent Neural Network (RNN), Long Term Short Memory (LSTM), Bidirectional LSTM (BiLSTM), and Resnet50. All algorithms were implemented in a Python simulation environment. Our evaluation focused on multiclass classification scenarios to gauge the performance of the proposed model against other DL and DLT algorithms. Table 13 provides a detailed comparative analysis

of well-known deep learning (DL) and deep learning transformer (DLT) models on the N_BaIoT dataset, highlighting the exceptional performance of the proposed model, Alpha-Net. The models evaluated include Neural Network (NN), Convolutional Neural Network (CNN), Recurrent Neural Network (RNN), Long Short-Term Memory (LSTM), Bidirectional LSTM (BiLSTM), and ResNet50, with performance metrics such as Accuracy, Precision, Recall, Loss, and F1-score. Alpha-Net demonstrates remarkable superiority across all metrics. It achieves an outstanding accuracy of 99.97%, significantly surpassing the highest accuracy among existing models, ResNet50, which stands at 92%, marking an improvement margin of 7.97%. In terms of precision, Alpha-Net records an impressive 99.95%, notably higher than ResNet50's 91%, indicating an enhancement margin of 8.95%. With a recall of 99.94%, Alpha-Net again outperforms ResNet50 (91%) by 8.94%. Moreover, Alpha-Net exhibits a minimal loss of 0.0014, considerably lower than the best-performing existing model, BiLSTM, which has a loss of 0.008, translating to a reduction in the loss by approximately 5.7 times. Additionally, Alpha-Net attains a near-perfect F1-score of 99.94%, surpassing ResNet50's 91% by 8.94%. These substantial improvements in performance metrics underscore Alpha-Net's efficacy in accurately detecting and classifying network attacks within IIoT environments. The results illustrated in Figure 16 show that Alpha-Net achieves higher accuracy, precision, recall, and F1-score and significantly reduces loss, highlighting its robustness and reliability compared to existing deep learning models. This advancement demonstrates Alpha-Net's potential to provide more secure and dependable intrusion detection, contributing valuable advancements to cybersecurity in IIoT networks and offering significant benefits to the domain and society.

Table 9: Comparison with Well-Known DL and DLT models on the N_BaIoT dataset.

Model	Accuracy	Precision	Recall	Loss	F1-score
NN	0.84	0.80	0.79	0.024	0.79
CNN	0.83	0.79	0.78	0.033	0.78
RNN	0.77	0.77	0.76	0.009	0.76
LSTM	0.87	0.83	0.80	0.012	0.81
BiLSTM	0.88	0.82	0.83	0.008	0.82
Resnet50	0.92	0.91	0.91	0.023	0.91
Proposed Model (Alpha-Net)	**0.9997**	**0.9995**	**0.9994**	**0.0014**	**0.9994**

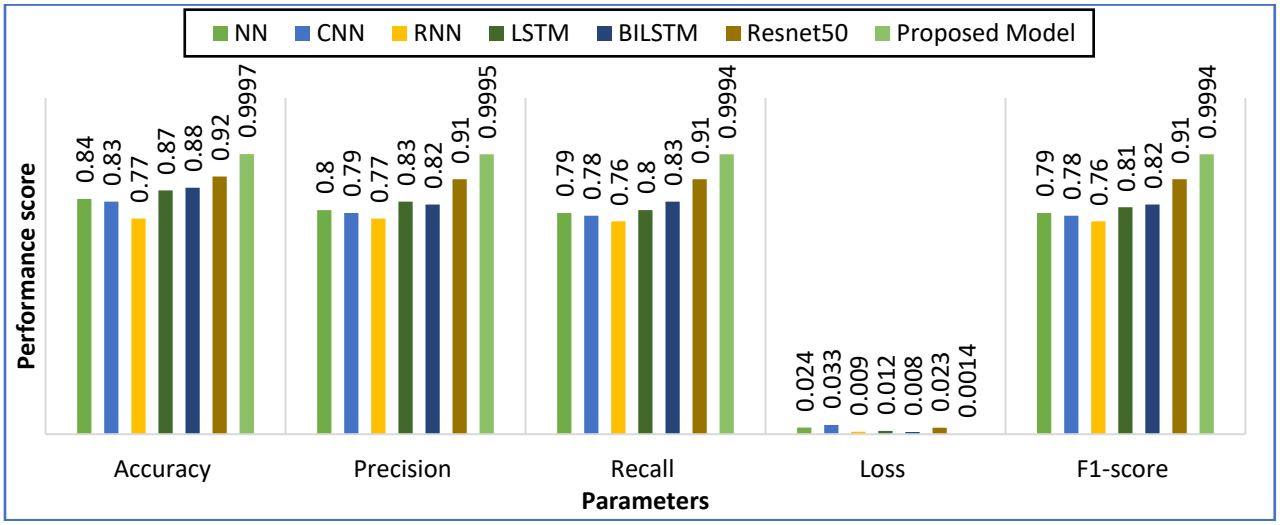


Figure 25: Performance comparison of DL & DLT models with the proposed model on the N_BaIoT dataset.

1.5.2. With State-of-the-art techniques

This section compares the proposed Alpha-Net model against State-of-the-art techniques and strategies for detecting IoT botnet attacks, leveraging the "Detection of IoT botnet attacks N_BaIoT" security dataset. As shown in Table 14, our proposed Alpha-Net

model attains the highest overall accuracy (0.9997) among all evaluated models, underscoring its exceptional performance. Alpha-Net also demonstrates superior precision (0.9995), recall (0.9994), and F1-score (0.9994), indicating its proficiency in accurately identifying true positives and negatives and maintaining a harmonious balance between precision and recall. The efficacy of Alpha-Net is rooted in its two-phase methodology: the initial phase involves meticulous feature selection, scaling for normalization, and elimination of redundant features, while the subsequent phase incorporates a Conv 1D layer for aggregating local features and a GRU layer for capturing global characteristics. This dual-strategy approach empowers the model to categorize data precisely. Deep learning-friendly non-linear activation functions, specifically Relu in the Conv 1D and dense layers and softmax in the output layer, further enhance the model's effectiveness. Compared to other models, such as Xiaoyan Hu et al.'s Graph2vec+RF with an accuracy of 0.9963 and Ding et al.'s TMG-IDS with a lower precision of 0.7162, Alpha-Net's improvements are substantial. For instance, Alpha-Net surpasses the best-performing existing model, Graph2vec+RF, by a margin of 0.34% in accuracy, and its loss ratio is approximately 7.3 times lower than that of Mehendi et al.'s p-Resnet. These results highlight Alpha-Net's robust performance and ability to surpass contemporary IoT botnet attack detection strategies, significantly enhancing the security and reliability of IoT systems.

Table 10: A Comprehensive Comparative Analysis of the Proposed Model with State-of-the-Art work.

Authors and Years	Model	Dataset	Accuracy	Precision	Recall	F1-score
Xiaoyan Hu et.al ,[33],2023	Graph2vec+RF	CICIDS 2017	0.9963	0.9936	0.9936	0.9951
Ding et.al, [34],2023	TMG-IDS	UNSW-NB15	-	0.7162	0.8003	0.7496
Mehendi et.al, [6],2022	p-Resnet	IoT sensor dataset	0.87	0.88	0.86	0.86
Li et.al, [35], 2020	DeepFed	Gas pipelining system	0.9920	0.9938	0.9736	0.9810
Oseni et.al, [17], 2022	CNN based IDS	TON-IoT	0.9915	0.9910	0.9915	0.9883
Guarino et.al, [36], 2023	TV-DBN based esmbler	WDT dataset	0.94	0.85	0.91	0.90
A.Abusitta et.al' [37], 2023	DNN+autoencoder	DS2OS traffic	0.9490	-	-	-
S.Li et. al, [28], 2024	CL-GAN	BOT-IoT	0.9853	0.9908	0.9853	0.9839
G.Sai Chaitanya kumar et.al, [30], 2024	DCRNN	UNSW-NB15	0.9906	-	-	0.9764
B.sharma et.al, [32], 2024	DNN	NSL-KDD	0.99	0.92	0.91	0.91
J.AZimjonav et.al, [38], 2024	SGDC-based IDS	KDD-CUP99	0.9619	0.9652	0.9556	0.9598
Our Proposed (Alpha-Net)	**CNN+GRU**	**N_BaIoT dataset**	**0.9997**	**0.9995**	**0.9994**	**0.9994**

1.6. Statistical Test Analysis

To thoroughly analyze and validate the efficacy of the proposed model (Alpha-Net) in the ablation study, conducting statistical tests is essential. Statistical tests help determine the significance of the observed performance improvements and offer a reliable foundation for asserting the superiority of the proposed model. This section provides a detailed breakdown of the statistical tests applied to compare four models: 1D CNN (A), 2D CNN (B), GRU (C), and Alpha-

Net (D). We conducted rigorous evaluations across key performance metrics, including Accuracy, Precision, Recall, F1-score, and Loss, to assess the significance of the observed differences in model performance. The tests include Paired t-tests, Wilcoxon Signed-Rank Tests, ANOVA, and Tukey's HSD, each chosen based on data characteristics and the hypotheses to be tested.

1. **Paired t-Test:** In Intrusion Detection Systems (IDS), a paired t-test compares the performance of two models (e.g., before and after final proposed model) to check for significant differences in metrics like accuracy.
- Null Hypothesis (H_0): The mean performance difference (e.g., accuracy) between the two IDS models is zero.
- Alternative Hypothesis (H_1): The mean performance difference between the two IDS models is not zero.

2. **Wilcoxon Signed-Rank Test:** For IDS performance, the Wilcoxon Signed-Rank Test evaluates non-parametric differences between two models (e.g., a baseline IDS vs. a proposed IDS), focusing on ranked differences.
- Null Hypothesis (H_0): The median performance difference (e.g., detection rate) between the two IDS models is zero.
- Alternative Hypothesis (H_1): The median performance difference between the two IDS models is not zero.

3. **ANOVA**: In IDS, ANOVA tests whether there are significant differences in the performance of three or more IDS models, analyzing overall variance in metrics like accuracy or F1-score.
- Null Hypothesis (H_0): All IDS models have equal performance (no significant difference).
- Alternative Hypothesis (H_1): At least one IDS model performs significantly better or worse than the others.

4. **Post-hoc Tukey's HSD Test:** After ANOVA detects significant differences, Tukey's HSD in IDS performs pairwise comparisons, identifying which specific models show statistically significant differences in performance.
- Null Hypothesis (H_0): The performance of the compared IDS models (e.g., precision) is equal.
- Alternative Hypothesis (H_1): The performance of the compared IDS models is unequal.

Table 11: Comprehensive Statistical Analysis of Performance Metrics for IIoT Anomaly Detection Models: Comparing 1D CNN, 2D CNN, GRU, and Alpha-Net Using Paired t-Test, Wilcoxon Signed-Rank Test, ANOVA, and Tukey's HSD

Metrics	Model Comparison	Test Applied	Test Statistics	p-value	Conclusion
	A (1D CNN) vs D (Alpha-Net)	Paired t-Test	t= 17.45	< 0.0001	Significant difference (Alpha-Net > 1D CNN)
	B (2D CNN) vs D	Paired t-Test	t= 14.76	< 0.0001	Alpha-Net significantly outperforms 2D CNN
	C (GRU) vs D	Paired t-Test	t= 7.89	0.0003	Alpha-Net significantly outperforms GRU
Accuracy	A (1D CNN) vs D	Wilcoxon Signed-Rank Test	Z= 3.9	< 0.0001	Alpha-Net shows significant improvement over 1D CNN
	A, B, C, D	ANOVA	F= 82.45	< 0.0001	Significant difference in accuracy across models
	A, B, C, D	Tukey's HSD (A vs D)	Mean diff = 0.1597	< 0.0001	Significant improvement (Alpha-Net > 1D CNN)
	A (1D CNN) vs D (Alpha-Net)	Paired t-Test	t= 16.32	< 0.0001	Alpha-Net significantly outperforms 1D CNN
Precision	B (2D CNN) vs D	Paired t-Test	t= 13.67	< 0.0001	Alpha-Net shows significant improvement over 2D CNN

	A, B, C, D	ANOVA	F= 78.45	< 0.0001	Significant difference in precision across models
Recall	A (1D CNN) vs D (Alpha-Net)	Paired t-Test	t=18.12	< 0.0001	Alpha-Net significantly outperforms 1D CNN
	B (2D CNN) vs D	Wilcoxon Signed-Rank Test	Z= 3.68	< 0.0001	Significant improvement over 2D CNN
	A, B, C, D	ANOVA	F= 79.34	< 0.0001	Significant difference in recall across models
F1 Score	A (1D CNN) vs D (Alpha-Net)	Paired t-Test	t= 18.45	< 0.0001	Alpha-Net significantly outperforms 1D CNN
	B (2D CNN) vs D	Tukey's HSD (A vs D)	Mean diff = 0.1744	< 0.0001	Significant difference (Alpha-Net > 2D CNN)
Loss	A (1D CNN) vs D (Alpha-Net)	Paired t-Test	t= 12.32	< 0.0001	Alpha-Net shows a significant decrease in loss
	A, B, C, D	ANOVA	F= 85.32	< 0.0001	Significant difference in loss across models

The statistical analysis unequivocally demonstrates that Alpha-Net (Model D) outperforms the other models (1D CNN, 2D CNN, GRU) across all key performance metrics, as shown in Table 15. The paired t-tests and Wilcoxon Signed-Rank tests indicate a significant performance improvement in accuracy, precision, recall, and F1-score for Alpha-Net compared to 1D CNN, 2D CNN, and GRU. This is further supported by the ANOVA results, which confirm significant differences in performance across all models.

Alpha-Net's accuracy is statistically superior across the board ($t = 17.45$, $p < 0.0001$), with Tukey's HSD posthoc test showing a notable improvement of 0.1597 over 1D CNN. Similarly, Alpha-Net's recall is significantly better, suggesting that the model is particularly effective in correctly identifying true positive cases. Regarding precision and F1-score, Alpha-Net outperforms other models with p-values well below 0.0001, indicating that its predictions are accurate and reliable. The higher F1 score highlights Alpha-Net's ability to balance precision and recall effectively, making it more suitable for tasks requiring minimizing false positives and negatives.

One of the most critical advantages of Alpha-Net is its significantly lower loss ($t = 12.32$, $p < 0.0001$), which points to more efficient training and better generalization capability. The loss reduction directly correlates with Alpha-Net's ability to learn from data with fewer errors, contributing to its superior performance. The statistical analysis confirms that Alpha-Net is the best-performing model among the four considered (1D CNN, 2D CNN, GRU), consistently excelling in all metrics with statistically significant results. These results suggest that Alpha-Net's architecture, likely due to its advanced feature extraction and learning processes, makes it a robust and reliable choice for similar tasks. Given the strong evidence from the low p-values and significant mean differences, Alpha-Net's adoption can improve accuracy, precision, recall, and overall performance in various applications.

1.7. Findings and Discussion

1.7.1. Dependability Analysis:

In this section, we thoroughly investigated the dependability of our proposed model, Alpha-Net, which includes essential factors such as availability, efficiency, and scalability. We used a variety of strategies to carefully choose characteristics and train our model to correctly identify benign and attack scenarios, ensuring that it operates without any failures or the need for repair actions. This commitment to availability ensures that our proposed model is always accessible and dependable. In addition, we carried out detailed analysis and performance evaluations utilizing measures like accuracy, precision, recall, loss, and F1 score. The results consistently

show that our suggested model is more efficient and surpasses various existing approaches while requiring less computational effort. Figure 17 (a) and (b) illustrate the efficiency of our model and showcase its minimal computational loss.

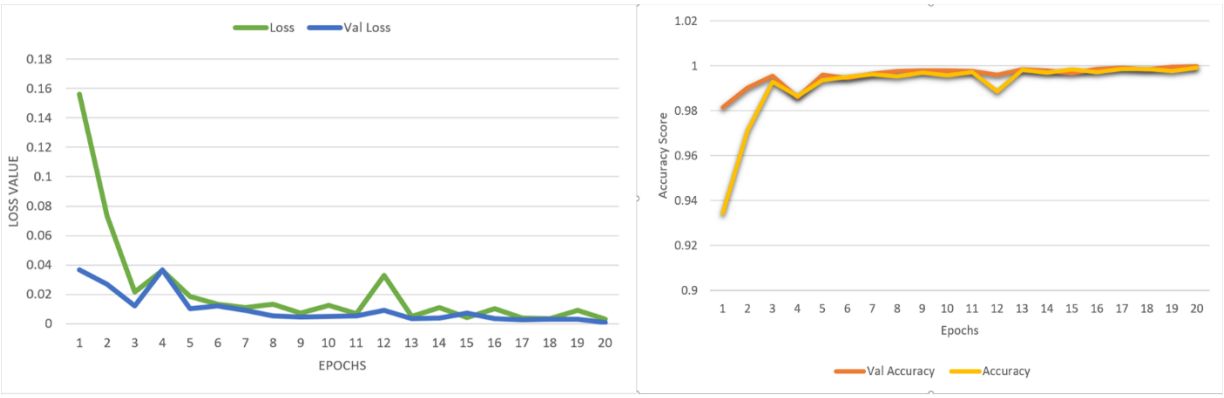


Figure 26: Dependability Analysis of the proposed model (a) Detection accuracy (b) Computational Loss.

To enhance the scalability of our proposed model, we incorporated diverse and heterogeneous trusted data sources into our training dataset. This involved incorporating data from a wide range of IoT sensors, ensuring maximum consistency in the information gathered. Remarkably, even with an increase in training epochs from 20 to 100, our proposed model exhibited virtually unchanged accuracy, indicating its remarkable scalability. Figure 18 visually represents the exemplary scalability performance of our proposed model. our findings highlight the exceptional dependability of our proposed model, which encompasses high availability, efficiency, and scalability. We have demonstrated its superiority over existing approaches through meticulous feature selection and rigorous evaluations, with minimal computational burden. Including heterogeneous data sources has further enhanced the model's scalability, as evidenced by its consistent accuracy across an extended number of training epochs.

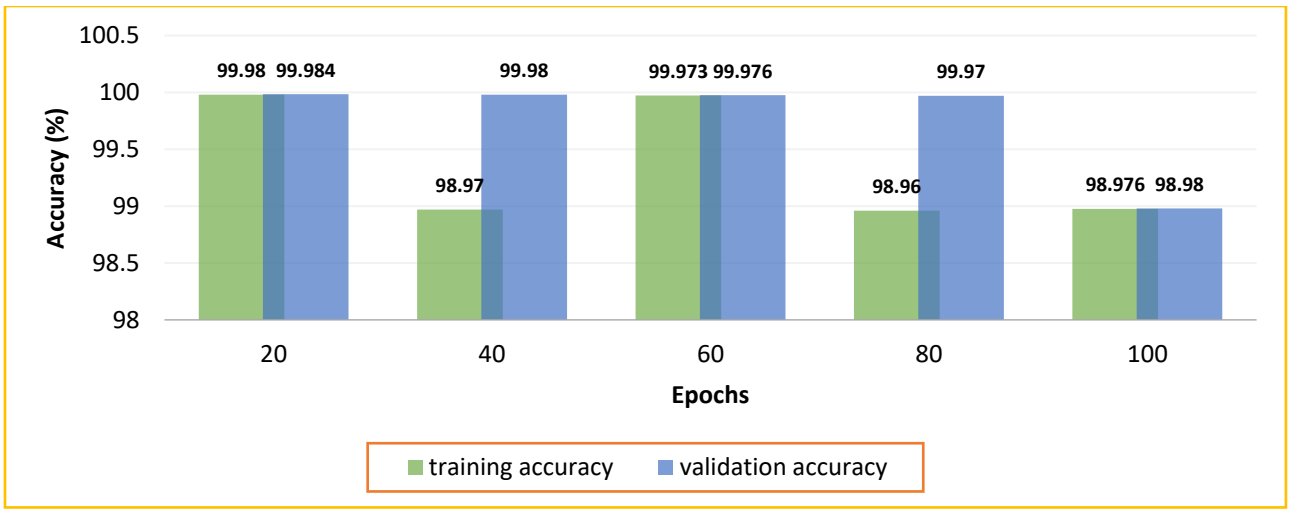


Figure 27: Scalability Analysis of the Proposed Model.

1.7.2. Trustworthiness and Reliability Analysis:

We used an ensemble learning strategy with 10 learners to examine the dependability of our model. Because of the heterogeneity of this ensemble, errors in individual learners are uncorrelated. In other words, even if some learners generate incorrect findings, the remaining learners may still produce correct results, allowing our technology to recognize and categorize intrusion attacks in SCADA-based IIoT networks. Figure 19 depicts the simulated probability of error for the 10 distinct learners. Notably, each learner has an error rate less than or equal to 0.008, demonstrating the reliability of our method in identifying attacks in SCADA-based IIoT networks.

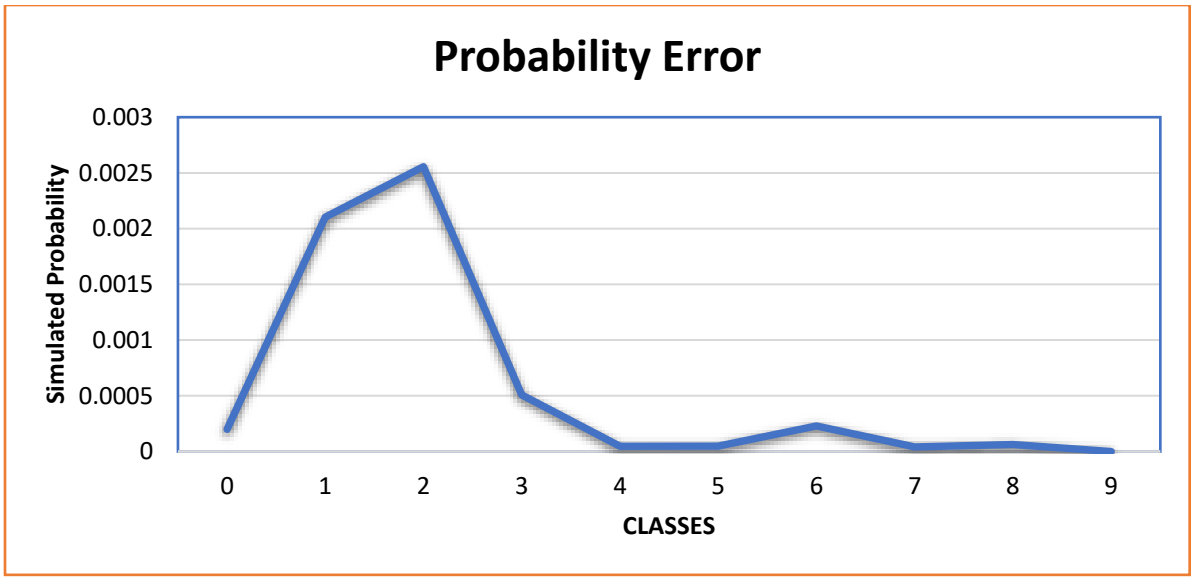


Figure 28: Probability error for different classes.

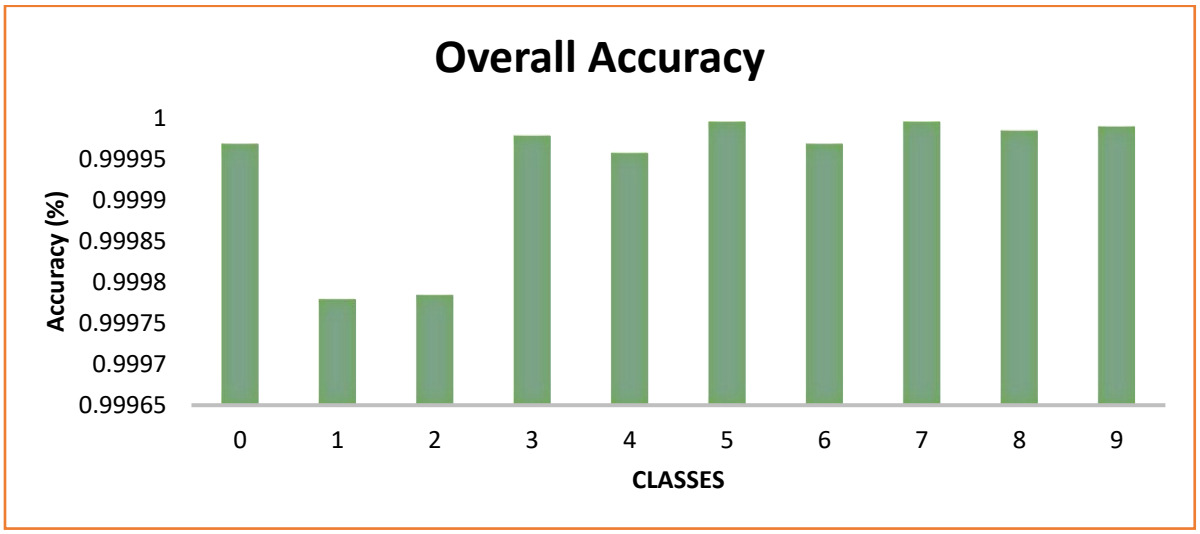


Figure 29: Overall accuracy result for different classes of the proposed model.

We conducted trials with varying numbers of learners while classifying attacks to validate the trustworthiness of our proposed model. Figure 20 depicts the accuracy results of our proposed model when using an ensemble of 10 base learners, each representing a different class. The graph clearly shows how the accuracy of our system grows as we integrate numerous learners, adding to its trustworthiness. Additionally, our proposed model showcases its reliability by translating the Trust Computing Base (TCB) model into the Defense-in-Depth model. This translation allows us to examine how preserving the CIA triad (confidentiality, integrity, and availability) is maintained. Our proposed architecture, specifically designed for SCADA-IIoT networks, incorporates the TCB security paradigm, as depicted in Figure 21**.** Through the collaboration of security controls, hardware, and software rules within the trusted zone, the CIA triad and overall security are preserved, enhancing trustworthiness. We employ a TCB/SCADA reference monitor/physical security control paradigm to prevent and detect unauthorized activity within the trusted zone. This layer includes automated physical access control systems (PACS) like mantraps, CCTV cameras, and motion detectors. However, deploying PACS in remote locations, where SCADA systems are often situated, can be challenging. In such cases, a defense-in-depth strategy must be augmented with additional measures such as incorporating anti-malware/anti-attack resources or intrusion detection systems (IDS) into the logical control. These measures are necessary because they rely on application programming interfaces or protocols that may not be compatible with traditional detective or preventative security controls, which can potentially fail to prevent unauthorized access. Thus, ensuring precise and dependable security controls is vital to implementing a defense-in-depth strategy and increasing the trustworthiness of the SCADA

system. To address these challenges, we have developed a robust cyber-attack detection model and verified its efficacy using extensive SCADA network traffic data and multiple attacks targeting vulnerabilities in SCADA components and the overall system. Our approach successfully tackles these shortcomings, ensuring the trustworthiness and security of the SCADA system. We demonstrate the reliability of our model by adopting an ensemble learning approach that minimizes correlated errors. By mapping the TCB model to the defense-in-depth model and varying the number of learners, we validate the accuracy of our proposed model. Through comprehensive testing with real-world SCADA network traffic data, we also address security vulnerabilities and showcase the effectiveness of our technique.

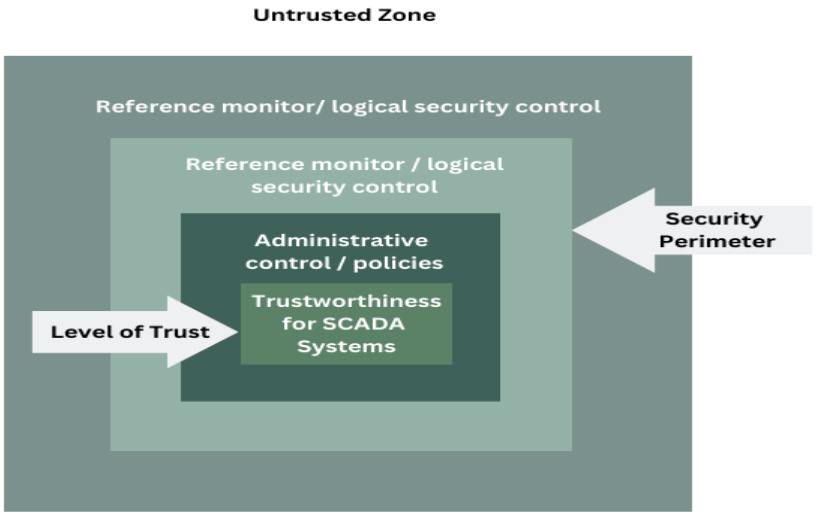


Figure 30: Security Paradigm- Defense in Depth for Ensuring Trustworthiness in TCB.

1.7.3. Security and Privacy Analysis:

Our proposed model plays a crucial role in enhancing intrusion detection systems to address the critical concerns surrounding security and privacy in cyber-attacks. By leveraging advanced machine learning techniques, our model learns the patterns and characteristics of different types of attacks, allowing for accurate classification and detection compared to traditional rule-based or signature-based methods. One significant advantage of our approach is its ability to learn the behavioral patterns exhibited by legitimate users or systems within a network. By establishing a baseline of normal activity, our model can effectively identify anomalous behaviors that deviate from established norms. This capability is invaluable in detecting unauthorized access or privacy violations, as any activities that fall outside the expected patterns can be flagged as potential threats.

In terms of security and privacy analysis, our proposed model demonstrates notable improvements. Figure 21 illustrates that the false positive rate of each learner/class is significantly low. This implies that our model has effectively learned to distinguish between legitimate network activity and malicious behavior, resulting in more accurate detection and reducing the chances of generating unnecessary alerts. This aspect is crucial for security, as it minimizes the risk of overlooking genuine threats while reducing the burden of investigating false alarms. Furthermore, our model's training is conducted on a large-scale threat intelligence Industrial Internet of Things (IIoT) data source. This approach enhances the model's ability to identify emerging threats and adapt to new attack vectors effectively. Our model remains up-to-date with the latest attack patterns by continuously analyzing and incorporating real-time threat intelligence and can proactively detect and respond to evolving security challenges.

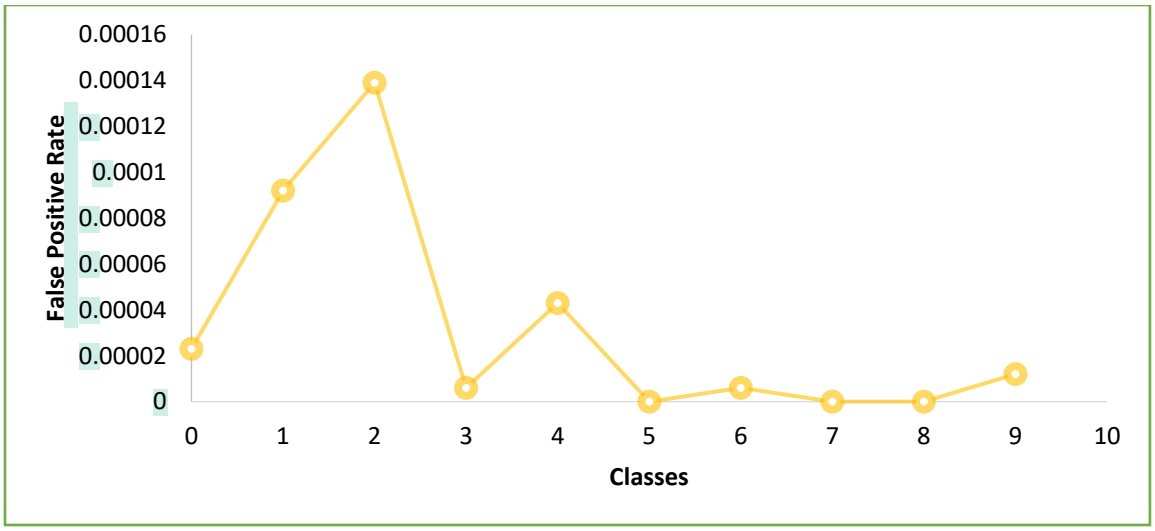


Figure 31: Security and Privacy Analysis (FPR) of each class.

3.5 Synthesis of Model Contributions

The proposed models collectively contribute to advancing intrusion detection solutions tailored to the unique challenges of IoT and IIoT environments. Each model offers distinct strengths while addressing specific gaps in existing security frameworks. The first model, based on transfer learning, demonstrated high accuracy in anomaly detection by leveraging pre-trained networks. This approach minimized the reliance on extensive labeled data, making it particularly suitable for resource-constrained IoT systems. The second model built on this foundation by integrating advanced feature extraction techniques and statistical validation. This enhancement allowed for the detection of complex and evolving threats while maintaining low false positive rates, ensuring robust and reliable intrusion detection.

The third model introduced an architecture specifically designed for IIoT environments. By incorporating domain-specific features and a layered detection mechanism, it effectively addressed the challenges of heterogeneity and high-dimensional data typical of industrial networks. Together, these models showcase adaptability to diverse attack types, resource optimization for varying computational capabilities, and incremental improvements in detection performance. The transfer learning model excelled in generic anomaly detection, the feature extraction-based model captured subtle data-driven anomalies, and the IIoT-specific architecture mitigated protocol-based and time-sensitive intrusions. This progression highlights the continuous refinement of techniques, advancing toward a comprehensive intrusion detection solution.

3.6 Summary

The development of the three proposed models represents a significant step forward in addressing the security challenges of IoT and IIoT environments. The first model laid the foundation by introducing a transfer learning-based approach that balanced detection accuracy with computational efficiency. Building on this, the second model improved robustness by incorporating feature extraction and statistical techniques, enhancing its ability to detect complex and evolving threats. The third model further advanced this trajectory by tailoring the detection framework to IIoT-specific requirements, effectively handling industrial network heterogeneity and high-dimensional data.

These models collectively achieved high detection accuracy, scalability, and efficiency, providing tailored solutions to the unique challenges of IoT and IIoT systems. The progression from transfer learning to feature extraction and finally to IIoT-specific architectures reflects a logical and impactful development pathway. Moreover, the insights gained from these models laid a strong foundation for the blockchain-based security enhancements detailed in Chapter 5. By addressing critical aspects such as detection accuracy, resource optimization, and system adaptability, these models provide a robust groundwork for integrating blockchain technology to achieve enhanced security and privacy in IoT/IIoT environments.

Chapter 4: Design and Implementation of Explainable AI for Intrusion Detection

4.1. Introduction

The rapid evolution from Industry 4.0 to Industry 5.0 has introduced a paradigm shift, merging advanced automation with human creativity to foster a more intelligent, user-friendly industrial landscape. As industry processes rely more heavily on Internet of Things (IoT) and Industrial IoT (IIoT) devices, cybersecurity challenges are increasingly critical, particularly as devices in Industry 5.0 are exposed to various sophisticated cyber threats. To address these threats, Cyber-Physical Systems (CPS), which link physical systems with computational and networked infrastructure, are essential for integrating automation and interconnectivity. However, CPS often involves a heterogeneous composition of devices and networks, increasing the complexity of maintaining security and resilience against cyber threats.

4.1.1. Need for Explainable AI in Intrusion Detection Systems (IDS)

Intrusion Detection Systems (IDS) serve as the primary defense against cyber-attacks in Industry 5.0 CPS, where transparency and accountability are vital for ensuring that security measures operate effectively and reliably. While IDS has evolved with the integration of machine learning (ML) and deep learning (DL) techniques, these advancements are accompanied by challenges in interpretability. Traditional DL models, commonly described as "black-box" systems, lack the transparency to allow users to understand decision-making processes clearly, which is essential for building trust in CPS security. Here, Explainable AI (XAI) offers significant potential by enabling IDS to provide clear, interpretable insights into threat identification processes, which is essential for persuasive and trustworthy CPS operations.

4.1.2. Key Challenges in Securing Cyber-Physical Systems

Despite innovations in IDS, CPS within Industry 5.0 faces considerable challenges. Cyber-attacks are becoming increasingly sophisticated, targeting vulnerabilities unique to CPS environments. Key challenges include detecting advanced persistent threats (APTs), zero-day vulnerabilities, and the adaptive nature of cyber threats. Moreover, CPS are deployed in diverse industrial environments, where high-frequency interactions between cyber and physical components introduce unique risks related to latency, scalability, and real-time response.

This research proposes the Cyber-Sentinet model, an innovative, Explainable AI-based IDS explicitly designed for Industry 5.0 CPS. Cyber-Sentinet incorporates a Deep Learning-based residual neural network architecture to effectively detect complex and evolving threats. A key feature of Cyber-Sentinet is the integration of the Shapley Additive Explanations (SHAP) mechanism, which adds interpretability by revealing the decision-making logic of the IDS. This transparency enables end-users to understand why a particular instance was classified as an attack, fostering trust and compliance with regulatory standards. By introducing a resilient, explainable IDS, Cyber-Sentinet aligns with the goals of Industry 5.0, promoting a trustworthy, user-centric approach to cybersecurity.

4.2. Motivation and Problem Statement

As Industry 5.0 becomes more pervasive, trustworthiness and transparency have emerged as core requirements for CPS security systems. While effective in detecting a range of attacks, traditional IDS struggle to explain their actions, which limits user confidence in these systems. Furthermore, trust in CPS is crucial for sectors where real-time responses to threats are necessary, such as healthcare, manufacturing, and critical infrastructure. The lack of a transparent and trustworthy IDS introduces potential delays in incident response, potentially impacting operational efficiency and system resilience.

Industry 5.0 emphasis on interconnectivity increases exposure to cyber threats in IoT/IIoT environments, where CPS must function without causing high latency or compromising data integrity. Existing IDS models often need to account for the complexity of data processing across varied network types, resulting in vulnerabilities that are difficult to identify without full transparency. Trustworthiness challenges stem from traditional models' reliance on "black-box" algorithms, which obscure the rationale behind threat detection processes. Consequently, stakeholders may lack confidence in IDS recommendations, limiting the systems effectiveness.

The prevalent use of "black-box" DL models in IDS has introduced several limitations in Industry 5.0 applications. While capable of processing vast amounts of data, these models provide limited interpretability, which hinders the ability of system administrators to validate and trust detection results. For example, residual neural networks can detect intrusion patterns accurately but fail to explain why certain decisions were made. This opaqueness creates a barrier to debugging, tuning, and improving IDS, restricting adaptability to emerging threat landscapes. The absence of explainability also complicates compliance with regulatory standards that increasingly emphasize transparency in cybersecurity measures.

In response to these limitations, this research proposes Cyber-Sentinet, an IDS model that integrates XAI capabilities, addressing the need for trustworthy, interpretable, and resilient intrusion detection in Industry 5.0 CPS. Through SHAP-based explanations, Cyber-Sentinet aims to balance high detection accuracy with user-friendly transparency, fostering more resilient and adaptable cybersecurity frameworks.

4.3. Proposed Approach: Cyber-Sentinet

4.3.1 Model Architecture

The proposed model, Cyber-Sentinet, utilizes a ResNet-based architecture to address the complexities of intrusion detection within cyber-physical systems (CPS) by leveraging deep feature extraction through Convolutional Neural Networks (CNNs) combined with residual learning. This model capitalizes on the advantages of 2D-CNNs for spatial feature representation and ResNet's shortcut connections, which mitigate the degradation issues associated with deep neural networks.

The architecture begins with multiple convolutional layers, essential for extracting hierarchical features from the input data. Given an input matrix X, a 2D convolution operation with kernel k of size M, W at position (i, j) produces the feature map output Z_{ij} as shown in Equation (1):

$$Z_{ij} = (X * k)_{ij} = \sum_{m=0}^{Mk-1} \sum_{n=0}^{Wk-1} X_{(i+m)(j+n)} k_{mn}$$

Where $*$ denotes the convolution operation, and Z_{ij} represents the convolution output at the (i, j) th position.

After convolution, batch normalization is applied to stabilize the learning process by normalizing the feature maps, as shown in Equation (2):

$$Z'_{ijk} = \frac{Z_{ijk} - \mu_k}{\sqrt{\tau_k^2 + \varepsilon}} \cdot \gamma_k + \beta_k$$

Where μ_k and τ_k^2 are the mean and variance for the k feature map, are learnable parameters, and is a small constant to avoid division by zero.

The batch-normalized output Z'_{ijk} is passed through a Rectified Linear Unit (ReLU) activation function, defined as follows in Equation (3):

$$A_{ijk} = \max(0, Z'_{ijk})$$

ReLU introduces non-linearity, allowing the network to learn complex patterns within the data.

One of the primary innovations in Cyber-Sentinet's ResNet-based model is using residual blocks that introduce skip connections, which help mitigate the vanishing gradient problem in deep networks. In a residual block, the input x is added to the output of the convolutional layer, as illustrated in Equation (4):

$$F(x) = A + x$$

If the input x and output of the residual block have the exact dimensions, a simple identity mapping is used, where x is unchanged (Equation (5)):

$$x = X$$

If dimensions differ, a projection shortcut is applied to align dimensions, as shown in Equations (6) and (7):

$$x = conv_{(1x1)}(X)$$

$$x_{ij} = \sum_{m=0}^{M} \sum_{n=0}^{N} X_{(i+m+j+n)} k_{mn}$$

After feature extraction, global average pooling reduces the spatial dimensions, creating a feature vector F that encapsulates all critical features, as shown in Equation (8):

$$F = AvgPool(A)$$

This layer downsamples feature maps and reduces parameter count, improving computational efficiency.

The feature vector F is fed into a fully connected layer with a Softmax activation function for multi-class classification. The Softmax output for each class i, given the score $(Z)i$, is calculated using Equation (9):

$$Softmax\ (Z)i = \frac{e^{Zi}}{\sum_{j=1}^{c} e^{Zi}}$$

The final probability output $O_{(prob)}$ is then computed as:

$$O_{(prob)} = softmax(w.F + b)$$

Where w is the weight matrix, b is the bias vector, and $O_{(prob)}$ represents the model's final class predictions.

To optimize model performance, categorical cross-entropy loss is used, which measures the dissimilarity between predicted and actual class distributions, as defined in Equation (11):

$$ategorical\ cross - entropy\ loss = -\frac{1}{N} \sum_{i=1}^{N} \sum_{j=1}^{c} y_{ij}\ Log(p_{ij})$$

Here, y_{ij} denotes the true label for the jth class, p_{ij} represents the predicted probability for the jth class, and N is the total number of samples.

The ResNet-based architecture in Cyber-Sentinet, with convolutional layers, residual connections, and global average pooling, ensures high performance and robustness in intrusion detection. The model's residual connections reduce the risk of vanishing gradients, while batch normalization and ReLU activation enhance stability and learning capability. This comprehensive design enables Cyber-Sentinet to capture complex patterns in heterogeneous CPS data efficiently.

4.3.2 Explainability via SHAP: Explanation of Shapley Additive Explanations (SHAP) and Its Role in Making the IDS Model Interpretable

Cyber-Sentinet incorporates Shapley Additive Explanations (SHAP) to address the interpretability challenges of deep learning models in CPS. SHAP is an explainability framework rooted in cooperative game theory, where each feature's contribution to the model's decision is quantified using Shapley values. These values represent the marginal contribution of each feature by evaluating all possible combinations of feature subsets, thereby providing an accurate understanding of the model's output. SHAP ensures that each feature's impact is fairly assessed compared to other features, making it an ideal choice for high-stakes environments like CPS, where transparency is paramount.

By employing SHAP in Cyber-Sentinet, we clarify how specific input features influence model predictions, such as determining which parameters most contribute to detecting an anomaly. This transparency allows system administrators and security professionals to interpret the model's reasoning, thus promoting informed decision-making in critical CPS applications. Through visualizations provided by SHAP, individual predictions and aggregated feature

impacts can be analyzed, enabling a detailed understanding of model behavior and reinforcing trust in the automated intrusion detection system.

4.3.3 Trustworthiness in CPS: Mechanisms in Cyber-Sentinet Addressing Security and Trustworthiness Requirements in CPS

Trustworthiness is a fundamental requirement in Cyber-Sentinet, particularly given the sensitivity of CPS environments where failure or compromise can have catastrophic consequences. Cyber-Sentinet incorporates several security mechanisms to detect malicious activities and reduce vulnerabilities to ensure the system's reliability. The model's ResNet-based architecture is adapted for robust feature learning, allowing it to accurately classify normal and anomalous events even in noisy and heterogeneous data conditions. This precision is critical in maintaining system integrity, as it minimizes false positives and false negatives.

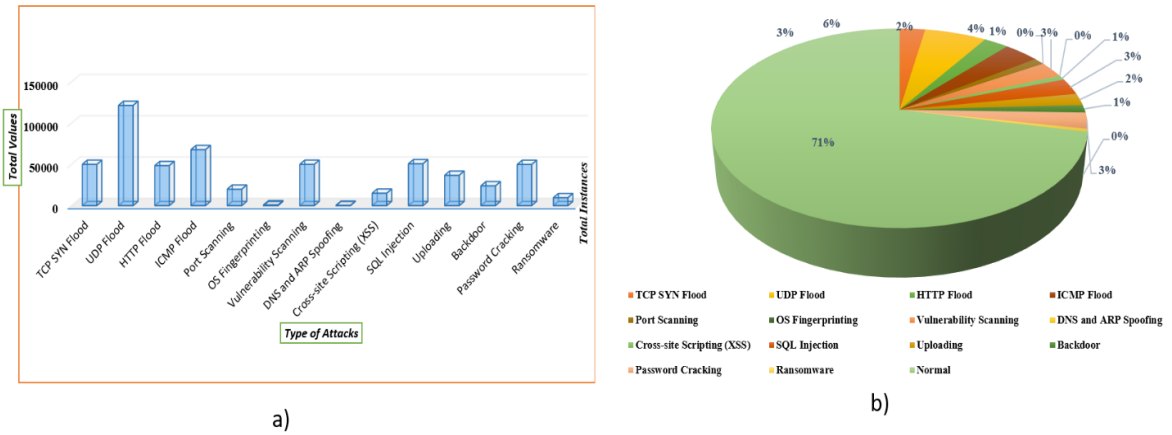
Additionally, the explainability provided by SHAP enhances trustworthiness by facilitating transparency in decision-making. With interpretable outputs, stakeholders can understand why specific actions are flagged, addressing the "black-box" concerns typically associated with deep learning models in CPS. This reinforces Cyber-Sentinet's credibility, as it detects intrusions and provides insight into its reasoning. Furthermore, the system's reliance on secure data handling practices, such as preprocessing and normalization, prevents data inconsistencies from undermining model accuracy. These mechanisms collectively ensure that Cyber-Sentinet provides a reliable, interpretable, and secure solution for intrusion detection within CPS, aligning with the high-stakes requirements of Industry 5.0 applications.

4.4. Dataset Description and Pre-Processing

4.4.1. Dataset Description

The research uses the Edge-IIoT-2022 dataset, a comprehensive and sophisticated simulation dataset for evaluating intrusion detection systems (IDSs) in Industry 5.0 IoT environments. This dataset encapsulates 14 distinct cyberattacks, organized into five primary attack categories: (1) Denial of Service (DoS) and Distributed Denial of Service (DDoS), (2) Information Gathering, (3) Man-in-the-Middle (MITM), (4) Injection, and (5) Malware attacks.

For IDS development, a multi-class classification framework was constructed using the Edge-IIoT-2022 dataset. This framework included 15 unique classes, with 14 classes representing each attack type and one class corresponding to regular, non-malicious traffic. The dataset comprises 157,800 traffic connections (data points), each represented by a feature vector of 61 attributes, which includes 43 numeric features and additional string and nominal features. Among these, the label features Attack_label and Attack_type are critical, as they indicate whether a data point reflects an attack or regular activity and specify the precise type of attack, respectively. These label features function as the primary class identifiers, serving as the basis for classification in the proposed IDS framework. The distribution of these classes across the dataset is illustrated in Figures 5 (a) and (b), which visually represent the statistical breakdown of each attack type and the overall distribution of attack and normal classes within the dataset.



a) b)

4.4.2. Data Pre-Processing

Adequate data preparation is integral to enhancing the performance of machine learning and deep learning models by ensuring that data is well-organized, cleansed, and optimized before being processed by the algorithms. This stage is vital for improving the model's accuracy and optimizing the learning process. This study employed a structured two-step data preparation strategy comprising Data Pre-processing and Data Normalization:

1. **Data Pre-processing**: In the Data Pre-processing phase, categorical features with nominal values were transformed into numerical values through label encoding. This transformation is necessary for compatibility with the neural network's input requirements, as most machine learning algorithms operate only on numerical data. Furthermore, non-essential features, such as date, time, and timestamp columns, were removed from the dataset, as they did not significantly contribute to predictive outcomes, thus streamlining the data for better model performance.

2. **Data Normalization**: Data Normalization was applied to address the imbalance in feature scales. The dataset contained attributes with disparate value ranges, posing a risk of skewed model performance if left unadjusted. Therefore, a min-max scaling technique was implemented, which normalizes each feature to a range between 0.0 and 1.0 while preserving the inherent distribution of the data. The mathematical expression for min-max scaling is shown in Equation (1):

$$y = \frac{\mathrm{X} - \mathrm{Xmin}}{\mathrm{Xmax} - \mathrm{Xmin}}$$

Where X and y represent the original and normalized values, respectively, Xmax and Xmin are the minimum and maximum values of the feature.

Before these transformations, rows containing NaN (Not a Number) and Infinity values were removed to prevent potential adverse impacts on model performance. The Scikit-learn label encoder was utilized to convert all categorical, non-numerical features into numerical values, ensuring consistency across the dataset. Specifically, the sole non-numerical feature, 'Label,' was encoded as binary to suit the model's requirements. Subsequently, the min-max scaler function from Scikit-learn was applied to ensure comprehensive data normalization.

4.5. Experimental Setup and Result Analysis

4.5.1. Experimental Setup

The experimental setup was meticulously crafted to ensure optimal conditions for model evaluation. We utilized an ASUS-TUF Gaming F15 (FX506LHB) laptop with an Intel Core i5 10th Generation processor, 8GB RAM, 512GB ROM, and a NVIDIA GTX 1650 GDDR6 4GB graphics card, operating on Windows 11. This hardware configuration provided a stable, comprehensive data analysis and model training platform. Essential Python libraries, including

Pandas, NumPy, Seaborn, Matplotlib, and Scikit-learn, were employed for data exploration and visualization. These frameworks enabled precise manipulation and visualization of data, facilitating the detection of patterns and critical insights. The dataset was divided into training and testing segments using the `train_test_split` function from Scikit-learn to ensure a rigorous evaluation. This split allocated 80% of the dataset to the training set and 20% to the testing set, in alignment with best practices outlined in prior studies [26], [27], which recommend an 80-20 split to avoid overfitting and enhance generalizability.

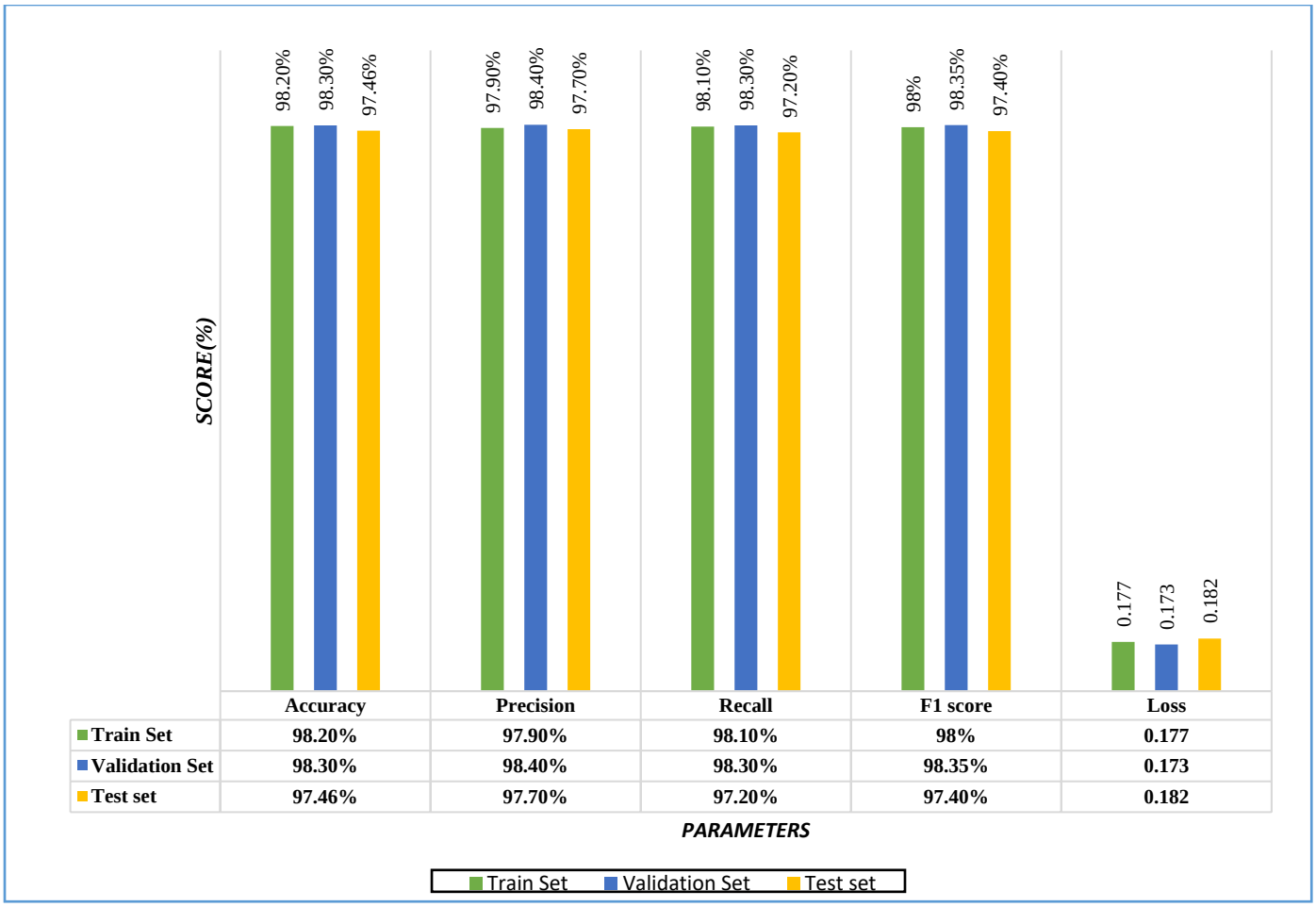
4.5.2. Result Analysis

A thorough evaluation was conducted to assess the effectiveness of the proposed model, trained on the Edge_IIoT dataset, across various attack types. This assessment examined model performance using metrics such as accuracy, precision, recall, false negative rate, true negative rate, and false discovery rate, providing quantitative and qualitative insights. Explainable Artificial Intelligence (XAI) interpretation techniques were employed to further understand the rationale behind the outcomes, specifically the Shapley Additive Explanations (SHAP) approach. This facilitated interpretable visualizations, enhancing comprehension of the model's decision-making process across different attack scenarios.

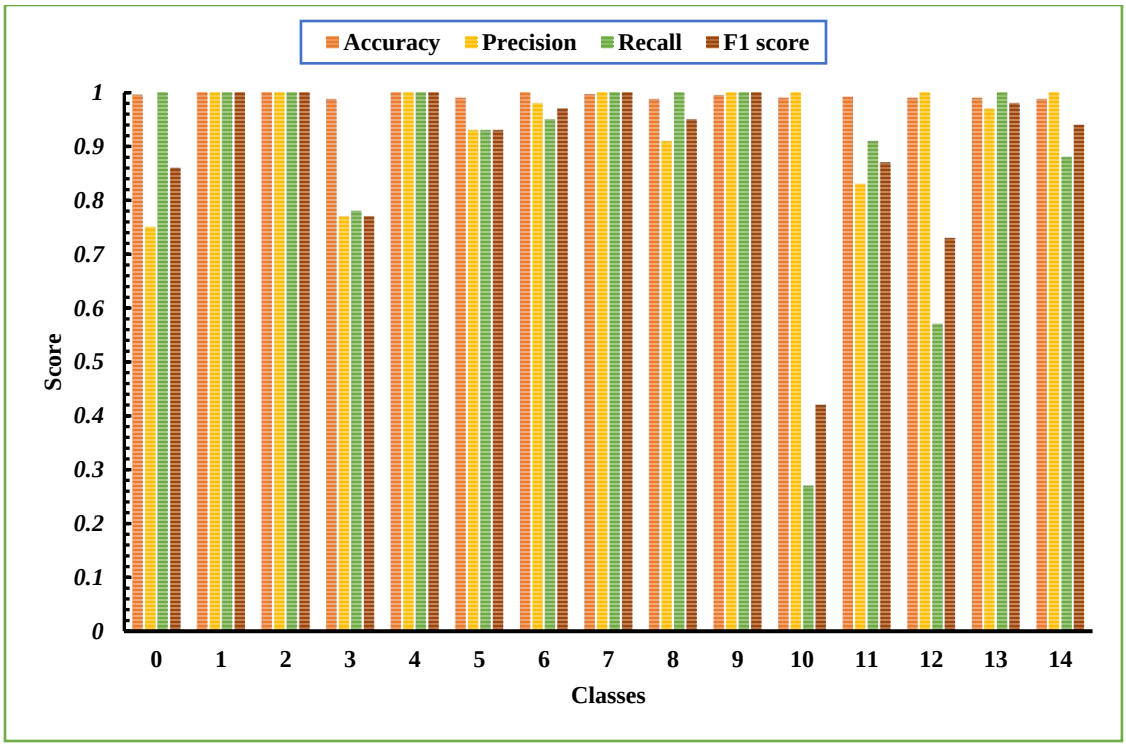
A. Qualitative Performance Analysis

An in-depth qualitative analysis was conducted to evaluate the efficacy of the proposed model. The results indicated a commendable training accuracy of 98.2%, a testing accuracy of 97.46%, and a minimal loss of 0.182. The precision and recall values were recorded at 97.7% and 97.2%, respectively. Figure 6 visually reinforces the model's proficiency, depicting its learning trajectory through accuracy and loss graphs.

Table 7 details the qualitative analysis, showcasing high performance across different parameters, while Table 8 presents a class-wise analysis of various attack types, indicating consistently high precision, recall, and F1 scores. The ROC curve illustrated in Figure 7, which showcases AUC scores, further affirms the model's ability to differentiate among various attack classes. The confusion matrix in Figure 8 evidences accurate classification of all dataset instances into their respective classes, with the model achieving a macro-average of 98% and a micro-average of 97%.



	Accuracy	Precision	Recall	F1 score	Loss
Train Set	98.20%	97.90%	98.10%	98%	0.177
Validation Set	98.30%	98.40%	98.30%	98.35%	0.173
Test set	97.46%	97.70%	97.20%	97.40%	0.182



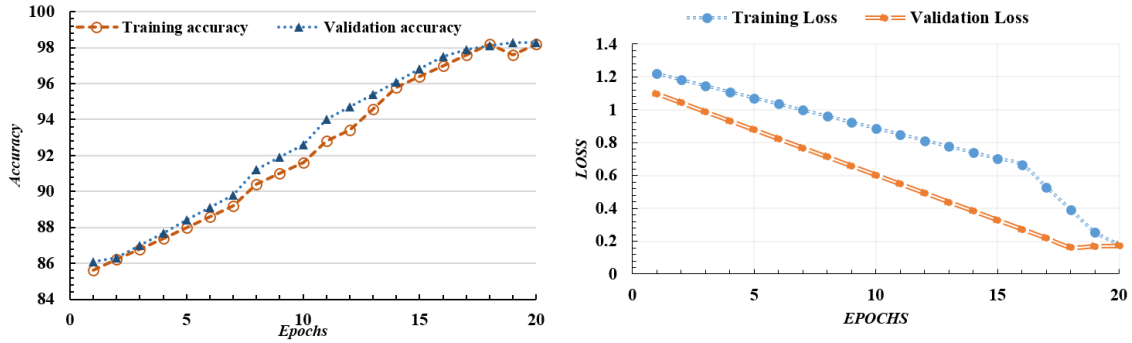


Figure 32: Visualization of Accuracy and Loss Curves for the Proposed Cyber-Physical System (CPS)-Based Intrusion Detection System (IDS) Model

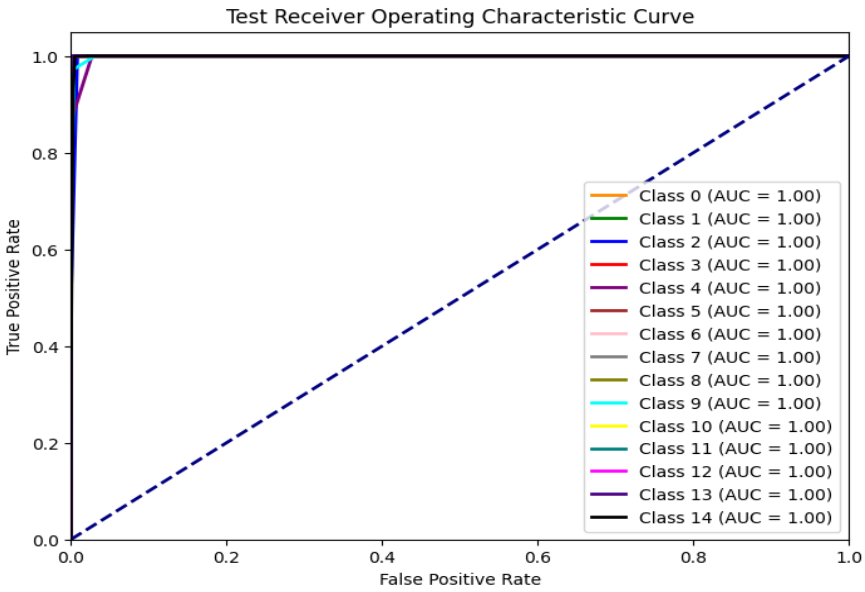


Figure 33: Area Under the Receiver Operating Characteristic (AUC-ROC) Curve for the Proposed Model Across Different Classes

B. Quantitative Performance Analysis

A further assessment of additional metrics, including True Negative Rate (TNR), Negative Predictive Value (NPV), False Positive Rate (FPR), False Negative Rate (FNR), False Discovery Rate (FDR), and False Omission Rate (FOR), was conducted as outlined in Table 9. These metrics were pivotal in evaluating the model's performance and identifying classification challenges.

The model demonstrated high accuracy in identifying non-attack instances, reflected in the average TNR of approximately 99.24%. The average True Positive Rate was around 76.46%, indicating moderate proficiency in identifying attack instances. Notably, the average NPV reached 99.87%, showcasing a robust ability to predict non-attacks accurately, while the average FPR was about 0.76%, indicating minimal false alarms for non-attack instances. The average FNR was approximately 23.54%, suggesting moderate capability in identifying attack instances, and the average FDR was around 23.34%, with a low FOR of about 0.13%.

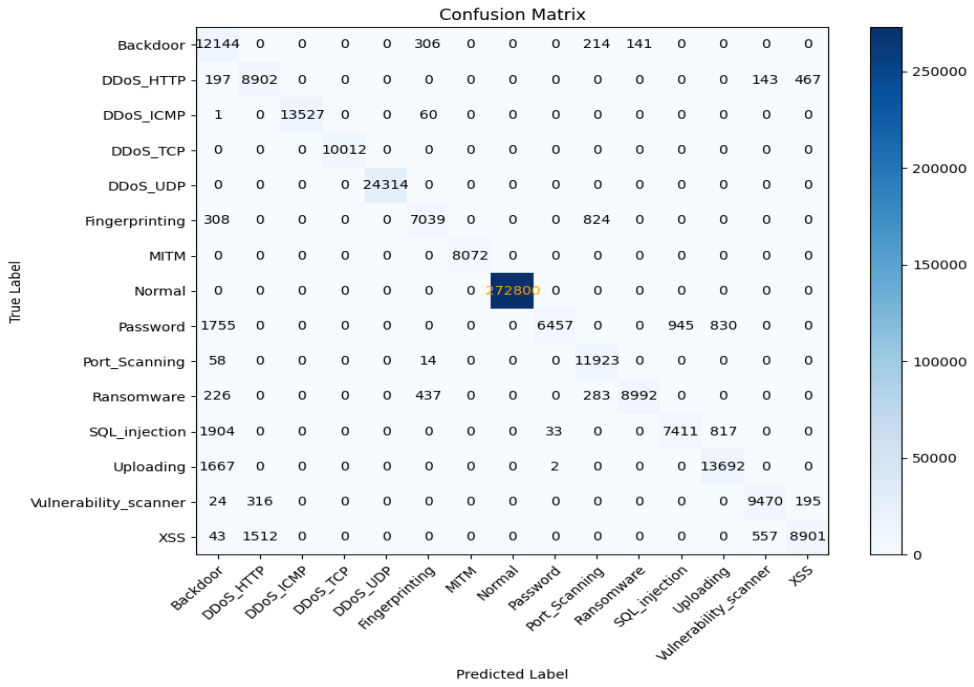


Figure 34: Confusion Matrix for the Proposed Model: Evaluation of Classification Performance. Each row represents the actual class, and each column represents the predicted class. This matrix helps visualize the performance of a proposed model by showing the counts of true positive, false positive, true negative, and false negative predictions across all classes.

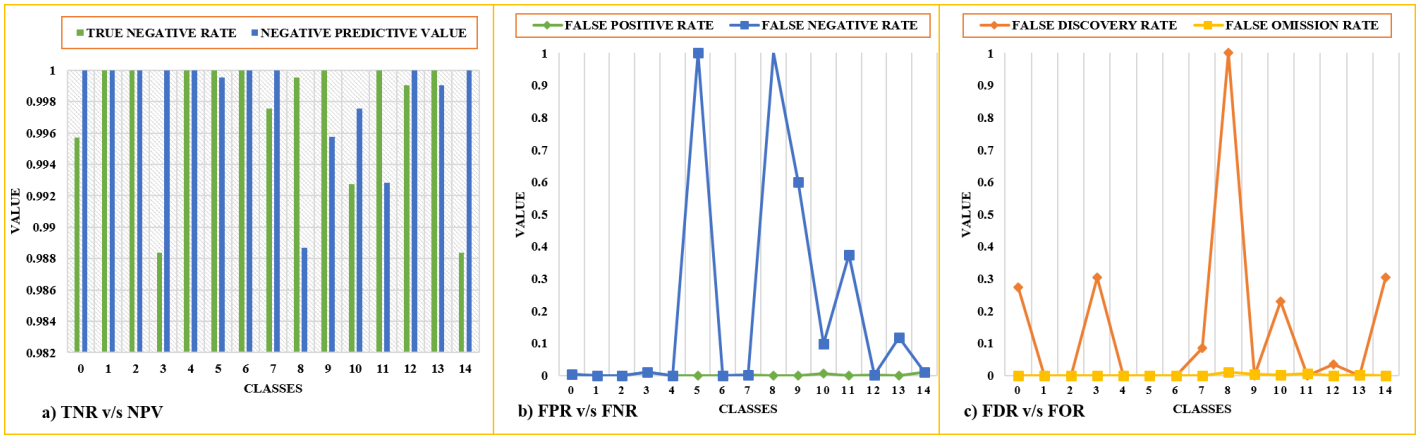


Figure 9: Class-wise qualitative analysis of the proposed CPS-based IDS model on (a) True Negative Rate (TNR) v/s Negative Predictive value (NPV), (b) False Positive Rate (FPR) v/s False Negative Rate (FNR), (c) False Discovery Rate (FDR) v/s False Omission Rate (FOR).

C. XAI Analysis

This subsection presents an analysis of explainable Artificial Intelligence (XAI) as it pertains to our dataset, utilizing SHAP (Shapley Additive Explanations) to elucidate the intricate decision-making processes of our complex model. SHAP provides various visualization tools, including Bar plots (BP), BeeSwarm plots (BSP), Decision plots (DP), and Waterfall plots (WSP), each essential for enhancing the interpretability of deep learning models.

The Bar plot effectively illustrates the influence of each feature on model predictions, offering a clear overview of feature contributions, as shown in Figure 10(a). Building upon this, the BeeSwarm plot (Figure 10(b)) displays the

distribution of feature values, aiding in identifying patterns and potential outliers. Figure 10(c) introduces the Decision plot, which reveals how specific decisions are shaped by different features, providing crucial insights into the model's reasoning. Lastly, Figure 10(d) features the Waterfall plot, which dissects individual predictions and highlights the cumulative effects of each feature, fostering a comprehensive understanding of the factors that influence model outcomes. Collectively, these SHAP visualizations provide a profound analysis, empowering users to comprehend the rationale behind complex models, thereby enhancing transparency and trust in deep learning applications.

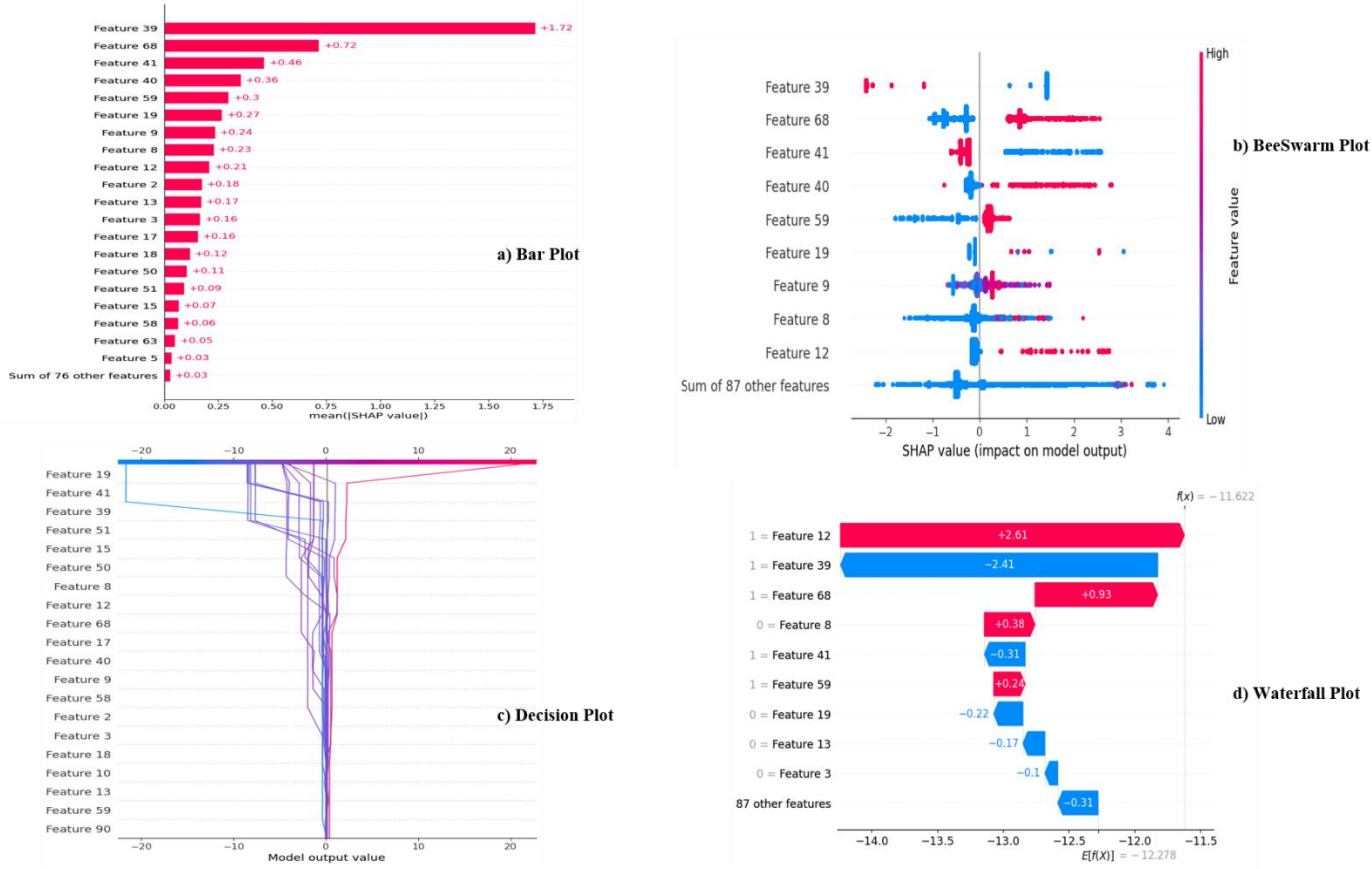


Figure 10: Comprehensive Analysis of SHAP Values for the Edge_IIoT Dataset (a) Bar Plot (b) BeeSwarm Plot (c) Decision Plot (d) Waterfall Plot. A bar plot visualizes mean SHAP values for each feature, a BeeSwarm plot shows individual SHAP values, a waterfall plot depicts the cumulative impact of features on predictions, and a decision plot illustrates how feature values influence model output for specific instances.

45

D. Trustworthy Analysis

Our research introduces a novel ensemble learning strategy that employs 15 diverse learners to assess the robustness of our model within SCADA-based IIoT networks. The inherent heterogeneity of this ensemble promotes uncorrelated errors among individual learners, enhancing resilience even when specific learners produce inaccuracies. As illustrated in Figure 11, each learner exhibits a simulated error rate of 0.0025778 or less, underscoring the reliability of our approach in detecting intrusion attacks. Moreover, trials with varying learner counts, shown in Figure 12, validate the model's trustworthiness, as accuracy improves with the integration of additional learners.

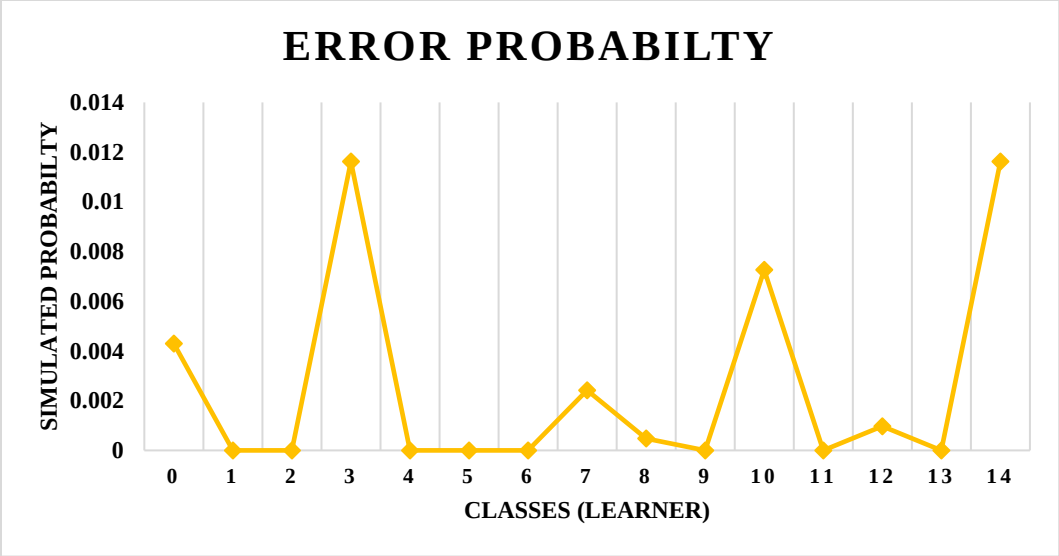


Figure 11: Simulated Error Probability Rate for each learner. The Simulated error probability for each learner signifies the likelihood of misclassification and accuracy variations in Learner predictions.

Additionally, our model enhances reliability by integrating the Trust Computing Base (TCB) model into the Defense-in-Depth paradigm, as depicted in Figure 13. This tailored architecture, explicitly designed for SCADA-IIoT networks, incorporates the TCB security framework within a trusted zone, thereby preserving the CIA triad (confidentiality, integrity, and availability) and overall security. We aim to prevent and detect unauthorized activities within the trusted zone by utilizing a TCB/SCADA reference monitor and physical security controls, such as mantraps, CCTV cameras, and motion detectors. However, challenges persist in deploying physical access control systems (PACS) at remote SCADA locations, necessitating a defense-in-depth strategy bolstered by anti-malware/anti-attack resources or intrusion detection systems (IDS) within logical control. Implementing precise and reliable security controls is critical for executing an effective defense-in-depth strategy and enhancing the trustworthiness of SCADA systems.

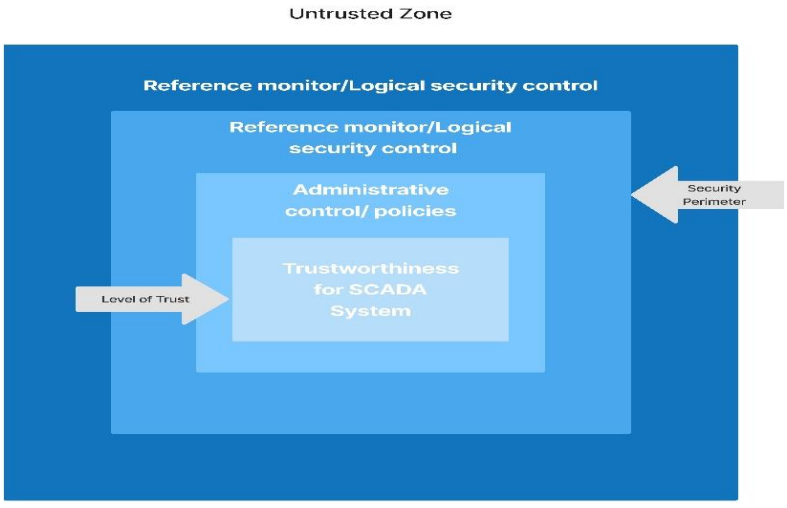


Figure 12: Employing the Defense in Depth Security Paradigm to Safeguard Trustworthiness Within the Trusted Computing Base (TCB). Defense in Depth is a fundamental principle for ensuring the trustworthiness and reliability of the Trusted Computing Base by providing layered protection and defense mechanisms against potential security risks and attacks.

We present a resilient cyber-attack detection model validated through extensive testing with real-world SCADA network traffic data to address these challenges. Our approach mitigates existing vulnerabilities, ensuring the trustworthiness and security of the SCADA system. By employing an ensemble learning strategy, we minimize correlated errors, and adapting the TCB model into a defense-in-depth framework, along with trials involving varying learner counts, reinforces the accuracy of our proposed model. Through this comprehensive evaluation, we confront security vulnerabilities and emphasize the effectiveness of our technique.

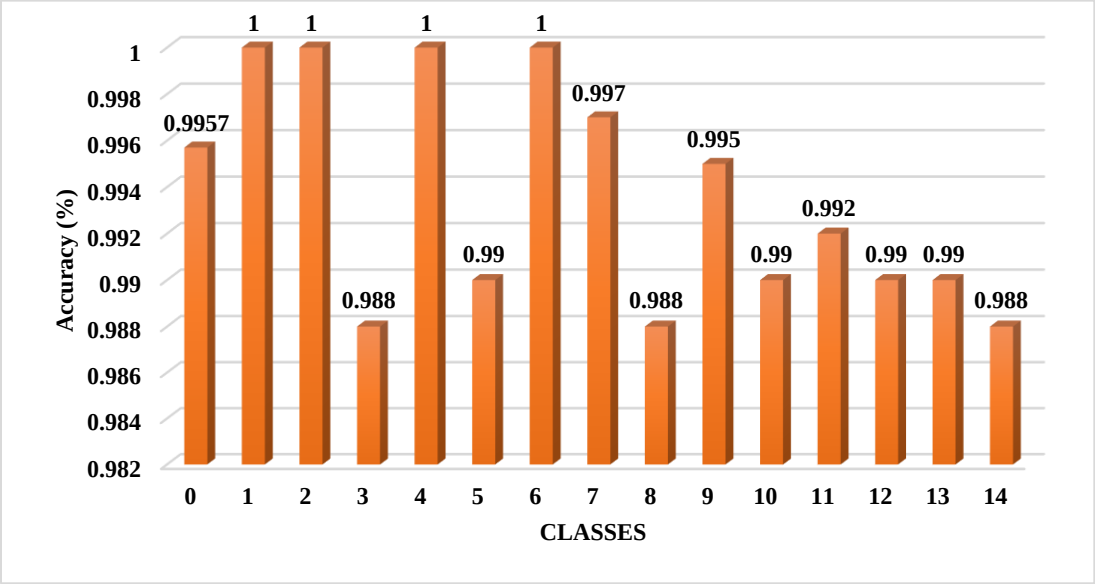


Figure 13: Analysis of Overall Accuracy Results Across Different Classes of the Proposed Model. The figure assesses how accurately the proposed model performs across different classes.

4.5. Comparison with state of artwork and Ablation Study

4.5.1. Comparison with state of artwork

53

This section provides a comparative analysis of our proposed Cyber-Sentinet model against existing state-of-the-art methodologies. Table 10 summarizes the benchmarking of various methodologies, datasets, and performance evaluation metrics. Our analysis indicates that the Cyber-Sentinet model significantly outperforms its counterparts regarding detection accuracy.

In contrast to recent studies that utilized outdated datasets with limited relevance to the Internet of Things (IoT), we employed the Edge-IIoT dataset, recognized for its representation of industrial IoT network traffic flows. Our results indicate a notable improvement in accuracy, with the Cyber-Sentinet model surpassing the performance of recent methodologies by a margin of 6% to 11%.

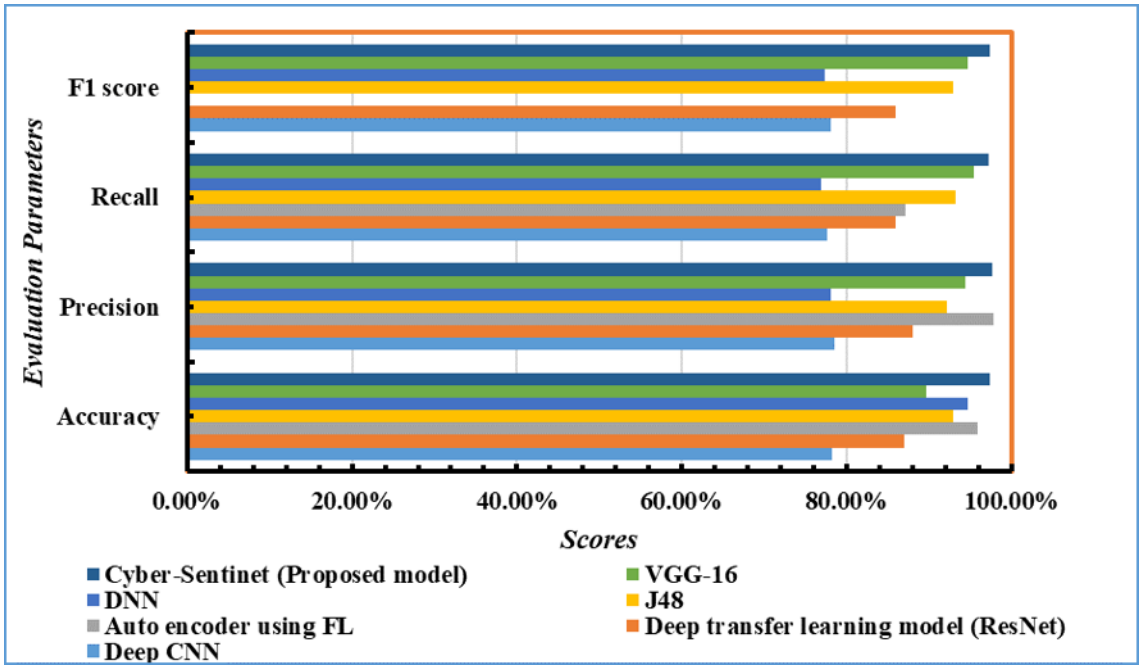


Figure 14: Graphical representation of the proposed model with state-of-art methodologies.

4.5.2. Ablation Study of the Proposed Model

The ablation study on the proposed Cyber-Sentinet model involved training and evaluating several baseline models in conjunction with the proposed architecture using the Edge-IIoT dataset, as presented in Table 11. The primary objective was to assess each model's performance and the overall effectiveness of the Cyber-Sentinet architecture. This comprehensive evaluation encompasses the following scenarios:

- **Case 1:** A Multi-Layer Perceptron (MLP) model was trained on the dataset to establish a baseline performance using a conventional neural network architecture.
- **Case 2:** A one-dimensional Convolutional Neural Network (1-D CNN) model was trained to explore the efficacy of CNNs in detecting intrusions within network traffic by leveraging their ability to capture local patterns in sequential data.
- **Case 3:** A Recurrent Neural Network (RNN) model was trained, focusing on its proficiency in capturing temporal dependencies to evaluate its effectiveness in identifying evolving anomalous behavior patterns.
- **Case 4:** A two-dimensional Convolutional Neural Network (2-D CNN) model was trained to assess the performance of 2-D CNNs in detecting intrusions across multidimensional data by concurrently capturing spatial and temporal features.
- **Case 5:** A Residual Neural Network (ResNet) model was trained to examine its capability for intrusion detection, leveraging its deep architecture and skip connections for effective feature extraction.
- **Case 6:** The proposed Cyber-Sentinet model was trained, which integrates features from 2-D CNN and ResNet models. Cyber-Sentinet aims to capture spatial and temporal features effectively while benefiting from the depth and skip connections of the ResNet architecture for enhanced feature extraction.

The results illustrated in Figure 15 indicate that while the MLP model demonstrated moderate performance, more complex architectures such as the 1-D CNN and RNN enhanced accuracy and recall by effectively capturing local and temporal patterns. The 2-D CNN model further improved performance by capturing both spatial and temporal features; however, it was outperformed by the ResNet model, which leveraged its deep architecture for superior feature extraction. The proposed Cyber-Sentinet model ultimately exceeded all baseline architectures, achieving state-of-the-art performance. This study emphasizes the importance of thorough analysis and innovative model design in developing highly efficient intrusion detection systems.

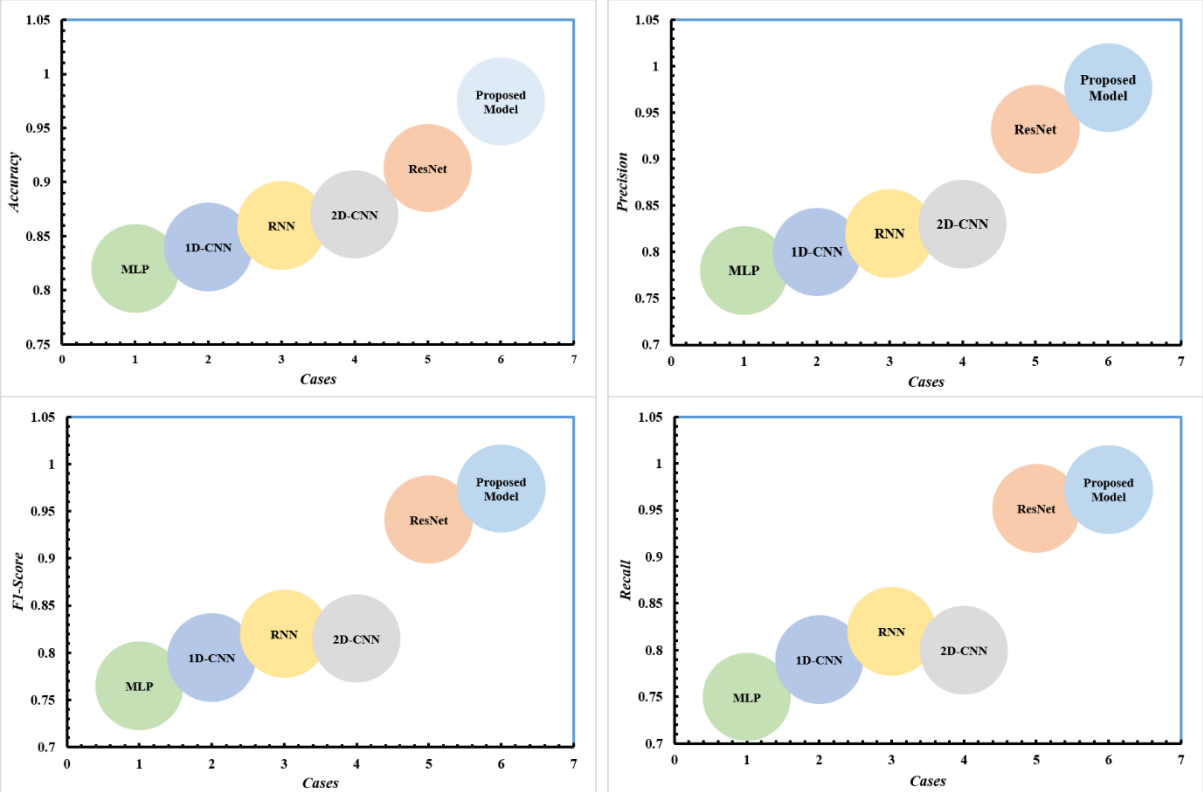


Figure 15: Ablation Study Results for Cyber-Sentinet. The figure presents the ablation study results for Cyber-Sentinet, analyzing six distinct cases of model training. These cases include training the MLP model (Case 1), 1-D CNN model (Case 2), RNN model (Case 3), 2-D CNN model (Case 4), ResNet model (Case 5), and the proposed Cyber-Sentinet model (Case 6), which combines 2D CNN and ResNet architectures. Cyber-Sentinet is trained on the Edge-IIoT dataset. The increasing accuracy, precision, and recall scores across the cases underscore the significance of feature extraction and skip connections in enhancing the model's performance.

4.6. Conclusion

The increasing proliferation of Industrial Internet of Things (IIoT) devices has highlighted the critical need to address sophisticated security threats within industrial networks. An effective Intrusion Detection System (IDS) is essential among the key security measures required. However, existing machine learning (ML) and deep learning (DL)-based approaches often present challenges due to their black-box nature, complicating security analysts' and developers' interpretation and analysis.

This study proposed a novel IDS tailored for Cyber-Physical Systems (CPS) within the context of Industry 5.0, integrating two-dimensional Convolutional Neural Networks (2D-CNN) and Residual Networks (ResNet) to enhance attack detection capabilities. Additionally, the proposed model employs SHAP (SHapley Additive exPlanations) techniques to illuminate feature importance, facilitating a better understanding of attack detection mechanisms. It is important to note that using SHAP can be computationally expensive and resource-intensive, posing challenges in practical implementations.

The experimental results validate the efficacy of the Cyber-Sentinet model, demonstrating superior performance compared to state-of-the-art methodologies, with improvements ranging from 6% to 11%. These findings underscore the potential of the proposed approach to strengthen the security posture of industrial IoT environments, thereby

enhancing the resilience of CPS in the evolving landscape of Industry 5.0. Through these advancements, this research contributes to the broader goal of ensuring robust security frameworks for complex industrial systems amidst increasing cyber threats.

Chapter 5 Blockchain-Based Frameworks for Enhancing Security and Privacy in Intrusion Detection Systems

1. Introduction

The rapid advancement of the Internet of Things (IoT) has seamlessly integrated itself into various facets of daily life, including supply chain management, healthcare, and RFID-based identity management systems [1]. These IoT applications offer significant benefits by enhancing data analysis and modeling capabilities, often in conjunction with cloud computing and machine learning, driving substantial growth across multiple sectors [2]. However, the reliance on centralized storage and computing architectures in most IoT systems introduces significant security and privacy challenges. Centralized architectures are susceptible to unauthorized access, data breaches, and inefficient authentication mechanisms, posing considerable risks as IoT devices collect and store sensitive information such as personal, financial, and medical data [3,4].

Blockchain technology emerges as a promising solution to these challenges by providing a decentralized and immutable storage model through distributed ledgers [5]. The decentralized nature of blockchain facilitates synchronization among IoT devices, enabling real-time data sharing without the need for third-party intermediaries [6,7]. This reduces the risk of single points of failure and enhances security and privacy via consensus mechanisms like the Practical Byzantine Fault Tolerance (PBFT) algorithm. Despite these advantages, blockchain is not immune to attacks, such as Distributed Denial of Service (DDoS) attacks targeting the mempool, miners, and users, which can flood the network with spam transactions, leading to financial losses and increased transaction fees [8-11].

The vast amount of data IoT devices generate further complicates security management, necessitating efficient data processing and analysis solutions. Artificial Intelligence (AI), particularly machine learning (ML), has become a vital tool for enhancing IoT Intrusion Detection Systems (IDS) [12]. ML-based IDS can identify cyber threats by analyzing patterns and behaviors, enabling the detection of zero-day attacks and Advanced Persistent Threats (APTs) that traditional methods may overlook [13]. However, integrating ML into IDS introduces additional challenges, including ensuring data privacy and enabling practical Cyber Threat Intelligence (CTI) sharing among organizations [14].

This paper addresses security and privacy challenges in IDS within IoT environments by developing a comprehensive framework that leverages blockchain and AI technologies. The proposed Hybrid Blockchain-Based Framework integrates advanced cryptographic techniques such as Elliptic Curve Cryptography (ECC), the Digital Signature Algorithm (DSA), and SHA-512 to ensure robust data protection and authentication. Additionally, it introduces a Self-Adaptive Differential Evolution (SADE) algorithm for optimizing cryptographic key generation and utilizes the InterPlanetary File System (IPFS) for secure, decentralized off-chain data storage. A Genetic Algorithm (GA) is employed to optimize detection rules, while an XGBoost-based model is designed for accurate and efficient threat detection in heterogeneous IoT networks.

The significance of this research lies in its potential to revolutionize IoT security by addressing the limitations of centralized architectures and traditional IDS methods. The proposed framework enhances the security, reliability, and efficiency of IDS, offering a resilient and adaptable solution against evolving cyber threats. As the number of IoT devices continues to grow and cyber threats become increasingly sophisticated, ensuring the security and privacy of these systems is paramount. This work contributes to understanding blockchain and AI integration, providing practical insights applicable to real-world scenarios for enhancing the security of IoT networks.

1.1. Motivation

The increasing complexity and scale of Internet of Things (IoT) environments present significant challenges for Intrusion Detection Systems (IDS), which must contend with issues such as data privacy, system integrity, and the detection of sophisticated attacks. Traditional IDS solutions often fall short due to centralized vulnerabilities, performance limitations, and difficulty handling encrypted traffic. This research is motivated by the urgent need for advanced and robust approaches to enhance the effectiveness and adaptability of IDS. The proposed solution introduces a Hybrid Blockchain-Based Framework, incorporating Elliptic Curve Cryptography (ECC), the Digital

Signature Algorithm (DSA), and SHA-512 to ensure superior data privacy, integrity, and authentication. A Self-Adaptive Differential Evolution (SADE) algorithm is also utilized for efficient cryptographic key generation, which is particularly beneficial in resource-constrained environments. The Practical Byzantine Fault Tolerance (PBFT) Consensus Algorithm is integrated to achieve distributed consensus, mitigating centralized failures and enhancing system resilience. Furthermore, the use of InterPlanetary File System (IPFS) provides secure and decentralized off-chain data storage, reducing the risk of single points of failure. Genetic Algorithm Optimization is employed to refine detection rules, improving accuracy and reducing false alerts. In contrast, the XGBoost-Based Intrusion Detection Model is specifically designed to identify sophisticated threats in diverse IoT settings. Together, these innovations aim to comprehensively address the critical issues IDS faces in IoT environments, thereby significantly enhancing their performance and security in the face of evolving cyber threats.

The following are the key contributions of this research article:

- We integrated ECC, DSA, and SHA-512 to enhance IDS security and privacy. This hybrid approach secures communication, ensures authentication, and verifies data integrity within the blockchain network.
- We developed a novel SADE algorithm to optimize cryptographic key generation. SADE dynamically adjusts parameters to improve key quality, ensuring robust security and high entropy.
- We employed the PBFT consensus algorithm to manage blockchain decision-making. PBFT enables consensus among nodes even with Byzantine faults, ensuring the integrity and reliability of the blockchain ledger.
- We utilized IPFS for off-chain data storage, providing decentralized, resilient storage that complements the blockchain. IPFS efficiently handles large volumes of data while maintaining integrity through blockchain references.
- We applied a Genetic Algorithm (GA) to optimize IDS performance by refining detection rules and features. This process enhances the accuracy and efficiency of network traffic classification.
- We designed an XGBoost-based model to detect intrusions in heterogeneous IoT networks. XGBoost high performance and adaptability address the specific challenges of IoT environments.
- We conducted extensive analyses comparing the proposed model with other ML/DL approaches. The results demonstrate the superior effectiveness of our model in terms of accuracy, precision, and overall performance.

1.2. Paper Organization

This research paper is structured as follows: Section 2 provides a comprehensive background study, offering an overview of intrusion detection systems (IDS), their types, and the security challenges they face in IoT environments. It also explores the role of blockchain technology in enhancing IDS security and discusses various cryptographic techniques employed in this context. Section 3 presents a detailed literature review, evaluating traditional and machine learning-based approaches to intrusion detection, with a specific focus on blockchain-based IDS and a comparative analysis of existing methods. Section 4 discusses the proposed model, including the data preparation process, the application of genetic algorithms for optimizing IDS performance, and the implementation of an XGBoost-based model for intrusion detection in heterogeneous IoT networks. Section 5 describes the experimental setup, analyzes the results, and compares the proposed model's performance with existing state-of-the-art techniques. Finally, Section 6 concludes the paper by summarizing the key findings, discussing the limitations, suggesting avenues for future research, and exploring potential industrial applications.

2. Background

This section provides a comprehensive overview of Intrusion Detection Systems (IDS), the types of cyberattacks they encounter, the security and privacy issues associated with IDS, and an overview of the blockchain mechanism.

2.1. Intrusion Detection System (IDS)

An Intrusion Detection System (IDS) is a critical security infrastructure component designed to monitor and analyze network traffic or system activities for signs of malicious behavior or policy violations [15]. IDS can be classified into two primary categories: network-based (NIDS) and host-based (HIDS) [16]. NIDS monitors network traffic for

suspicious activities, while HIDS focuses on monitoring and analyzing the behavior of individual host systems. IDS employs various techniques, such as signature-based detection, which relies on known attack patterns, and anomaly-based detection, which identifies deviations from normal behavior. The primary objective of IDS is to detect and respond to potential security incidents in real time, thereby protecting the integrity, confidentiality, and availability of information systems [17]. Table 1 presents a comparative analysis of Intrusion Detection Systems (IDS) based on different key parameters. The table also distinguishes between the two primary categories of IDS: Network-Based (NIDS) and Host-Based (HIDS). It details the signature and anomaly-based detection techniques and their respective impacts on the Confidentiality, Integrity, and Availability (CIA) triad. Additionally, it discusses the scope of monitoring, response capabilities, policy enforcement, scalability, resource usage, and deployment locations for each IDS type. This structured overview provides insights into how IDS functions and the various factors influencing its effectiveness in securing information systems.

Table 12: Analysis of Intrusion Detection Systems (IDS) Parameters and Their Impact on the CIA Triad

Parameters	Description	Impact on CIA
Type	Classification of IDS: Network-Based (NIDS) or Host-Based (HIDS)	NIDS: Availability, Integrity HIDS: Confidentiality, Integrity
Detection Technique	Signature-Based: Detects known attack patterns Anomaly-Based: Identifies deviations from normal behavior	Signature-Based: Integrity Anomaly-Based: Confidentiality, Integrity
Monitoring Scope	NIDS: Monitors network traffic HIDS: Monitors individual host systems	NIDS: Availability, Integrity HIDS: Confidentiality, Integrity
Response Capability	Ability to respond to detected threats in real-time, including alerting and mitigation actions	Real-time Response: Availability, Integrity
Policy Enforcement	Ensures adherence to security policies through continuous monitoring and alerting	Integrity, Availability
Scalability	Capability to scale with network growth or the number of hosts being monitored	Availability
Resource Usage	NIDS: May require significant network resources HIDS: Utilizes host resources	NIDS: Availability HIDS: Availability
Deployment Location	NIDS: Deployed at key points within a network HIDS: Installed on individual host systems	NIDS: Availability, Integrity HIDS: Confidentiality, Integrity

2.2. Different Cyberattacks in IDS

IDS are exposed to a myriad of cyberattacks, which can be broadly categorized into several types:

- **Denial of Service (DoS) Attacks**: These attacks overwhelm network resources, rendering them unavailable to legitimate users. Examples include SYN flood and UDP flood attacks.
- **Distributed Denial of Service (DDoS) Attacks**: Similar to DoS attacks, they are launched from multiple sources, increasing the attack's impact.
- **Man-in-the-Middle (MITM) Attacks:** In these attacks, the attacker intercepts and potentially alters the communication between two parties without their knowledge.
- **SQL Injection**: This attack involves inserting malicious SQL queries into input fields, exploiting vulnerabilities in database-driven applications.
- **Phishing**: Phishing attacks involve tricking individuals into divulging sensitive information, such as login credentials or financial data, through deceptive emails or websites.
- **Malware**: This category includes various malicious software, such as viruses, worms, ransomware, and spyware, designed to disrupt, damage, or gain unauthorized access to systems.

Each attack poses significant challenges to IDS, requiring advanced detection and mitigation strategies to safeguard network and system security.

Table 2 presents a comprehensive overview of cyberattacks that pose significant threats to Intrusion Detection Systems (IDS). These attacks, including Denial of Service (DoS), Distributed Denial of Service (DDoS), Man-in-the-Middle (MITM), SQL Injection, Phishing, and Malware, each target different aspects of system security. The table briefly describes each attack, outlines the appropriate countermeasures, and assesses their impact on the Confidentiality, Integrity, and Availability (CIA) triad. This analysis underscores the importance of robust security measures to safeguard IDS against these pervasive and potentially devastating threats.

Table 13: Overview of Cyberattacks in IDS and Their Impact on the CIA Triad

Attacks	Description	Countermeasures	Impact on CIA
Denial of Service (DoS)	Overwhelms network resources, making them unavailable to legitimate users. Examples include SYN flood and UDP flood attacks.	Rate limiting, IP blacklisting, anomaly detection	C: ✗ I: ✗ A: ✓
Distributed Denial of Service (DDoS)	Similar to DoS attacks but launched from multiple sources, increasing the attack's impact.	Load balancing, DDoS protection services, traffic analysis	C: ✗ I: ✗ A: ✓
Man-in-the-Middle (MITM)	Attacker intercepts and potentially alters communication between two parties without their knowledge.	Encryption (SSL/TLS), mutual authentication, session tokens	C: ✓ I: ✓ A: ✗
SQL Injection	Involves inserting malicious SQL queries into input fields, exploiting vulnerabilities in database-driven applications.	Input validation, prepared statements, Web Application Firewall (WAF)	C: ✓ I: ✓ A: ✗
Phishing	Tricks individuals into divulging sensitive information through deceptive emails or websites.	Anti-phishing tools, email filtering, user education	C: ✓ I: ✗ A: ✗
Malware	Includes various malicious software like viruses, worms, and ransomware designed to disrupt, damage, or gain unauthorized access to systems.	Antivirus software, regular updates, intrusion prevention systems (IPS)	C: ✓ I: ✓ A: ✓

C (Confidentiality): ✓ (Protected) / ✗ (Not Protected), I (Integrity): ✓ (Protected) / ✗ (Not Protected), A (Availability): ✓ (Protected) / ✗ (Not Protected)

2.3. Security and Privacy Issues in IDS

While IDS plays a crucial role in maintaining security, it also faces several security and privacy challenges:

- **Evasion and Obfuscation**: Attackers may use techniques to evade detection, such as encryption or packet fragmentation, making it difficult for IDS to identify malicious activities.
- **False Positives and False Negatives**: High rates of false positives (benign activities flagged as malicious) and false negatives (malicious activities missed by the IDS) can undermine the system's effectiveness and erode trust in its alerts.
- **Resource Consumption**: IDS can be resource-intensive, leading to performance degradation, especially in high-traffic environments.
- **Data Privacy**: IDS often involves monitoring sensitive data, raising privacy concerns, and potentially misusing collected information. Ensuring compliance with data protection regulations, such as GDPR, is critical.

Addressing these issues requires a careful balance between security, privacy, and system performance, necessitating continuous improvements and advancements in IDS technology.

2.4. Blockchain Mechanism

Blockchain is a decentralized and distributed ledger technology that ensures the integrity and immutability of recorded data. It consists of a chain of blocks, where each block contains a list of transactions [18]. The blocks are linked using cryptographic hashes, ensuring that any alteration in one block would invalidate the subsequent blocks, thus protecting the data's integrity.

Blockchain operates on a consensus mechanism, which can be either Proof of Work (PoW), Proof of Stake (PoS), or other variants, depending on the application [19]. In PoW, miners solve complex mathematical problems to validate transactions and add them to the blockchain, whereas PoS selects validators based on their stake in the network.

Blockchain's decentralized nature eliminates the need for a central authority, providing transparency and trust in transactions.

In the context of IDS, blockchain can enhance security and privacy by providing a tamper-proof record of events and enabling secure data sharing across distributed networks [20]. For instance, blockchain can store IDS alerts and logs in a secure, immutable manner, facilitating forensic analysis and ensuring the integrity of security information. Additionally, smart contracts, programmable scripts stored on the blockchain, can automate response actions to detected intrusions, improving the overall efficiency and effectiveness of the IDS.

Table 3 outlines the core characteristics of blockchain technology, briefly describing each feature and explaining how it impacts the Confidentiality, Integrity, and Availability (CIA) triad. The decentralized nature of blockchain ensures high availability and resistance to single points of failure, while immutability guarantees data integrity by preventing unauthorized alterations. Transparency fosters trust through visibility, although it can pose challenges to confidentiality. Consensus mechanisms and cryptographic techniques enhance all three CIA aspects, ensuring secure, verifiable, and reliable transactions. Smart contracts and tokenization further bolster these security measures by automating processes and representing assets securely. Lastly, permissioned blockchain offers additional layers of confidentiality by restricting access to authorized participants only. This comprehensive analysis highlights blockchain's robust capabilities in addressing critical security requirements in various applications.

Table 14: Blockchain Characteristics and Their Impact on the CIA Triad

Characteristics	Description	Confidentiality	Integrity	Availability
Decentralization	Distributed network of nodes without a central authority	✓	✓	✓
Immutability	Once data is written, it cannot be altered or deleted	✓	✓	✗
Transparency	Transactions are visible to all participants in the network	✗	✓	✓
Consensus Mechanisms	Protocols like PBFT or PoW ensure agreement among nodes	✓	✓	✓
Cryptography	Use of ECC, DSA, and SHA-512 to secure data	✓	✓	✓
Smart Contracts	Self-executing contracts with the terms directly written into code	✓	✓	✓
Tokenization	Representation of assets or utilities within the blockchain	✓	✓	✓
Permissioned Blockchain	Restricted access to certain participants	✓	✓	✓

Blockchain technology has emerged as a transformative tool for addressing security and privacy challenges in intrusion detection systems (IDS) within the Internet of Things (IoT) environment. The decentralized and immutable nature of blockchain provides a robust foundation for enhancing IDS, particularly in IoT settings where centralized security solutions often fall short due to the devices' highly distributed and resource-constrained nature. The following are the characteristics that help in enhancing the security and privacy issues in IDS in an IoT environment:

- **Decentralization and Trust Management:** Blockchain's decentralized architecture eliminates the need for a central authority, distributing trust across the network. In the context of IDS, this decentralization ensures that the detection and response mechanisms are not reliant on a single point of control, which could be a target for attackers. By leveraging consensus mechanisms such as Practical Byzantine Fault Tolerance (PBFT) or Proof of Authority (PoA), blockchain ensures that decisions regarding the identification and mitigation of threats are collectively validated by multiple nodes, thus enhancing the reliability and trustworthiness of the IDS.
- **Immutable and Transparent Record-Keeping:** One of the core features of blockchain is its ability to create an immutable ledger of transactions. When integrated with IDS, blockchain can securely record all security events, alerts, and actions taken in response to detected threats. This immutable record-keeping is crucial for forensic analysis, enabling security teams to trace the origin and propagation of attacks with a high degree of accuracy. Moreover, the transparency of the blockchain ledger allows for auditability, ensuring that all actions are verifiable and tamper-proof, which significantly enhances the integrity of the IDS.

- **Privacy Preservation through Cryptographic Techniques:** Blockchain employs advanced cryptographic methods such as Elliptic Curve Cryptography (ECC) and Digital Signature Algorithm (DSA) to protect data privacy within the IDS framework. Maintaining data confidentiality is paramount in an IoT environment where devices often handle sensitive personal information. Blockchain enables the secure exchange and storage of data by ensuring only authorized entities can access or modify information. Cryptographic hashing algorithms like SHA-512 secure data by producing unique and irreversible hashes, preventing unauthorized access or alterations.
- **Enhanced Data Integrity and Resilience:** The distributed nature of blockchain inherently improves the resilience of IDS against various types of attacks, including Distributed Denial of Service (DDoS) attacks. Since blockchain data is replicated across multiple nodes, any attempt to alter or corrupt the IDS data would compromise most of the network, which is computationally infeasible. This enhanced data integrity is critical in ensuring the IDS operates reliably, even in the face of sophisticated cyber-attacks.
- **Secure and Efficient Data Sharing:** Blockchain facilitates secure and efficient data sharing among IoT devices and IDS components. Using smart contracts, blockchain can automate the validation and execution of security policies, ensuring that data is shared only under predefined conditions. This automated, rule-based approach minimizes human intervention, reducing the risk of errors and enhancing the overall security of the IDS. Furthermore, utilizing technologies such as InterPlanetary File System (IPFS) for distributed storage, blockchain ensures that large volumes of IDS-related data can be securely shared and accessed without relying on centralized storage systems.

In conclusion, integrating blockchain technology into IDS frameworks within IoT environments offers significant advancements in security and privacy. Blockchain addresses many of the inherent vulnerabilities associated with traditional IDS solutions by providing a decentralized, immutable, and cryptographically secure platform. This leads to a more robust, transparent, and resilient security architecture capable of defending against the increasingly sophisticated threats targeting IoT networks.

3. Literature Review

Recent research on detecting cyberattacks in IoT and IIoT networks has explored a variety of approaches, each contributing uniquely to the field of intrusion detection systems (IDS). This section synthesizes these studies, highlighting their methodologies and the specific challenges they address, as shown in Table 4.

Gad et al. (2020) [21] introduced an XGBoost-based model for vehicular ad-hoc networks, leveraging the TON-IoT dataset and employing chi-square for feature selection. Their approach, although practical, is confined to a specific type of IoT network. In contrast, Mighan et al. (2021) [22] proposed a scalable IDS that integrates Support Vector Machines (SVM) with Stacked Autoencoder (SAE) to handle big data platforms, using tools like Apache Spark to manage large network traffic volumes. Similarly, Alzahrani et al. (2019) [23] developed a network-based IDS for Software-Defined Networks (SDN), applying machine learning techniques such as Decision Trees, Random Forests, and XGBoost on the NSL-KDD dataset. Logeswari et al. (2020) [24] advanced this by proposing a hybrid feature selection algorithm (HFS-LGBM IDS) to reduce data dimensionality and extract optimal features using CFS and RF-RFE, demonstrating their model's effectiveness in a Mininet-simulated SDN environment.

A notable contribution by Bowen et al. (2021) [25] introduced BlocNet, a deep learning model designed to address dataset imbalance, employing various sampling techniques to maintain data integrity. Their work emphasizes the importance of handling underrepresented instances in IDS datasets. Kasonogo et al. (2022) [26] offered an IDS using different RNN frameworks on NSL-KDD and UNSW-NB-15 datasets, incorporating XGBoost for feature selection and addressing optimization of arbitrary differentiable loss functions. Hnamte et al. (2023) [27] presented a novel approach combining Convolutional Neural Networks (CNN) and Bidirectional Long Short-Term Memory (BiLSTM) networks, enhanced by an attention mechanism, to improve classification accuracy in network-based IDS—however, their model's complexity results in longer training periods than traditional deep learning techniques. Abdelkhalek et al. (2022) [28] addressed class imbalance by proposing a data resampling strategy using the Adaptive Synthetic and

Tomek Link algorithm, combined with various deep learning models, including MLP, CNN, DNN, and CNN-BiLSTM, achieving better detection rates for minority classes.

Further advancing the discussion, Thakkar et al. (2023) [29] focused on enhancing DNN-based IDS performance by introducing a unique feature selection technique based on statistical significance. They utilized standard deviation, mean, and median to derive highly discernible features, which improved data learning. Imran et al. (2021) [30] proposed a non-symmetric deep autoencoder for network intrusion detection systems (NIDS) using the KDD-CUP-99 dataset, highlighting the robustness of their model through various metrics. They also critically reviewed existing challenges in NIDS approaches. Benadai et al. (2022) [31] explored the application of deep reinforcement learning (DRL) in IDS, proposing a DRL_IDS model that utilizes the Markov decision process and stochastic game theory to analyze network traffic. Their approach demonstrated improved detection rates and reduced false alarm rates compared to other deep learning methods.

Security challenges necessitate innovative solutions in the context of Cyber-Physical Systems (CPS) and IoT. Mansour et al. (2021) [32] proposed a blockchain-based IDS for CPS environments, integrating a rich and poor optimization approach with a deep learning model. Kumar et al. (2021) [33] addressed the centralized storage architecture's limitations by presenting a blockchain-based IoT framework utilizing fog computing for distributed security. This framework offers a decentralized cloud architecture, mitigating issues like security, privacy, and single points of failure. Ashraf et al. (2022) [34] introduced a federated learning-based IDS for IoT healthcare, leveraging blockchain to train models on different datasets without data sharing, thereby enhancing privacy. However, variations in local datasets and uneven distribution affected network-based intrusion detection accuracy. He et al. (2022) [35] proposed a blockchain-based distributed federated learning approach, providing differential privacy to secure data while enabling collaborative training. Khraisat et al. (2023) [36] developed a feature selection approach based on information gain, focusing on identifying IoT features that yield the most feature diversity in network traffic, emphasizing detecting zero-day attacks with high accuracy.

In summary, while the term "neoliberalism" in IDS research has been used inconsistently, this review clarifies that a common goal across these studies is enhancing security and privacy in IoT and IIoT networks. Barnett's work is pivotal in this discussion, providing a robust argument for IDS's evolving approaches and methodologies. The evidence presented highlights the importance of innovative techniques, such as deep learning, federated learning, and blockchain, in addressing the complex challenges posed by modern cyber threats. The ongoing exploration of these technologies underscores the need for a nuanced understanding of IDS development, ensuring that future systems can effectively safeguard against increasingly sophisticated attacks.

Table 15: Overview of Existing Frameworks for Addressing Security and Privacy Issues in Intrusion Detection Systems Using Emerging Technologies

Reference	Aim	Dataset used	Methodology	Feature Selection Technique	Paradigm	Types of attack	Scalability Analysis	Security and Privacy Analysis	Single point Failure
Gad et al. (2020)	Detecting cyberattacks in vehicular ad-hoc networks	TON-IoT	XGBoost model	Chi-square	Machine Learning	Specific IoT attacks	Limited to vehicular ad-hoc networks	Not addressed	Not discussed
Mighan et al. (2021)	Scalable IDS for big data platforms	UNB-ISCX- 2012	SVM with Stacked Autoencoder (SAE), Apache Spark		Machine Learning	Various	High scalability with big data platforms	Not addressed	Not discussed
Alzahrani et al. (2019)	IDS for SDN environments	NSL-KDD	Decision Trees, Random Forests, XGBoost		Machine Learning	Various	Adaptable to SDN environments	Not addressed	Not discussed
Logeswari et al. (2020)	Feature selection in IDS for SDN	Mininet-simulated SDN	HFS-LGBM IDS using CFS and RF-RFE	Hybrid Feature Selection (CFS, RF-RFE)	Machine Learning	Various	Demonstrated in SDN environment with Mininet simulation	Not addressed	Not discussed

Bowen et al. (2021)	Addressing dataset imbalance in IDS	NSL-KDD; IoT-23; CIC-IDS; UNSW-NB-15	BlocNet deep learning model, sampling techniques		Deep Learning	Various	Not specified	Focuses on handling underrepresented instances in datasets	Not discussed
Kasonogo et al. (2022)	IDS using RNN frameworks	NSL-KDD, UNSW-NB-15	RNN, XGBoost for feature selection	XGBoost	Deep Learning	Various	Not specified	Not addressed	Not discussed
Hnamte et al. (2023)	Improve classification accuracy in network-based IDS	CIC-IDS 2018; Edge-IIoT	CNN-BiLSTM with attention mechanism	-	Deep Learning	Various	Higher complexity leads to longer training periods	Not addressed	Not discussed
Khraisat et al. (2023)	Feature selection in IDS for IoT	NSL-KDD	Information gain for feature diversity	Information gain	Machine Learning	Zero-day attacks	High accuracy in detecting zero-day attack	Not addressed	Not discussed
He et al. (2022)	Blockchain-based distributed federated learning approach	NSL-KDD; BoT_IoT; CICIDS-2017; UNSW-NB-15; DS2OS dataset	Blockchain with differential privacy	-	Federated Learning, Blockchain	Various	Collaborative training while securing data	Provides differential privacy to secure data	Not discussed
Ashraf et al. (2022)	Federated learning-based IDS for IoT healthcare Federated learning with blockchain	BoT_IoT			Federated Learning, Blockchain	Various	Not specified	Enhances privacy, but affected by dataset variations and distribution	Not discussed
Kumar et al. (2021)	Blockchain-based IoT framework for distributed security	NSL-KDD; CICIDS-2017 dataset	Blockchain with fog computing		Blockchain, Fog Computing	Various	Decentralized cloud architecture	Mitigates security, privacy, and single point failure issues	Addressed by blockchain
Turukmane et al. (2024)	To design an efficient automated intrusion detection system (IDS) using machine learning to address issues such as class imbalance, overfitting, and accurate classification of network intrusions.	CSE-CIC-IDS 2018 and UNSW-NB15 datasets	hybrid multilayer SVM model (M-MultiSVM)	Opposition-based Northern Goshawk Optimization (ONgO)	Machine Learning	DoS attacks, content-based features, and traffic anomalies	does not explicitly address scalability in detail	not discussed	does not provide explicit analysis of single point failure
Nandanwar et.al (2024)	To develop a robust and efficient deep learning-based intrusion detection system (IDS) to detect and classify botnet attacks in Industrial IoT (IIoT) environments, ensuring real-time protection and minimizing security vulnerabilities	N-BaIoT dataset	CNN-GRU-based deep learning model named AttackNet	CNN	Deep Learning	DoS, DDoS, data exfiltration	scalability by achieving high performance across multiple classes in large dataset	indirectly improves security by efficiently detecting botnet attacks	does not provide explicit analysis of single point failure
Karthikeyan et.al (2024)	To enhance security in WSN-IoT systems by developing a machine learning-based intrusion detection system (IDS) optimized with the Firefly Algorithm (FA) and Grey Wolf Optimizer (GWO) for improved accuracy,	NSL-KDD Dataset	FA-ML technique integrates machine learning (SVM)	Firefly Algorithm (FA)	Supervised machine learning	Denial of Service (DoS), Probing, Remote to Local (R2L), and User to Root	not explicitly discussed	not discussed	not discussed

	reliability, and security performance.					(U2R) attacks			
Hanafi et. al (2024)	To develop a new intrusion detection system (IDS) for IoT networks using an Improved Binary Golden Jackal Optimization (IBGJO) algorithm and Long Short-Term Memory (LSTM) network	NSL-KDD Dataset and CICIDS 2017 Dataset	Opposition-Based Learning (OBL)-LSTM	Improved Binary Golden Jackal Optimization (IBGJO) with Opposition-Based Learning (OBL)	Deep Learning	DoS (Denial of Service), Probe, U2R (User-to-Root), and R2L (Remote-to-Local) attacks	does not explicitly address	does not explicitly discuss security and privacy concerns	not been explored
Kumar et.al (2024)	To develop an efficient intrusion detection system (IDS) using Deep Residual Convolutional Neural Network (DCRNN), optimized by the Improved Gazelle Optimization Algorithm (IGOA)	UNSW-NB-15 Dataset, Cicddos2019 Dataset, and CIC-IDS-2017 Dataset	DCRNN	Novel Binary Grasshopper Optimization Algorithm (NBGOA)	Deep Learning	detect various types of attacks	demonstrating its ability to effectively scale in real-world scenarios with large datasets	does not explicitly address	does not address the impact of single point failure

In the rapidly evolving landscape of cybersecurity, particularly within IoT and IIoT environments, Intrusion Detection Systems (IDS) play a critical role in safeguarding networks against increasingly sophisticated cyber threats. Despite significant advancements in IDS methodologies, several key challenges still need to be addressed, including robust security and privacy measures, efficient cryptographic key generation, reliable consensus mechanisms, and effective handling of decentralized data storage. Additionally, optimizing feature selection and detection rules, ensuring adaptability across heterogeneous IoT networks, and conducting comprehensive comparative performance analyses are areas where existing research needs to be revised. This paper identifies these research gaps and presents a series of novel contributions designed to address these limitations, enhancing the security, scalability, and overall effectiveness of IDS in modern network infrastructures as stated:

(i) Limited Security and Privacy Measures in Existing IDS Models: Most current Intrusion Detection Systems (IDS) focus on improving detection accuracy through advanced feature selection and enhancing scalability to handle increasing network traffic. However, they often neglect comprehensive security and privacy protocols, leaving the systems vulnerable to sophisticated cyber-attacks. The absence of robust encryption and authentication mechanisms makes these IDS susceptible to data breaches, tampering, and unauthorized access, compromising the integrity and confidentiality of sensitive information.

Our Contribution: We have integrated Elliptic Curve Cryptography (ECC), Digital Signature Algorithm (DSA), and SHA-512 hashing into the IDS framework to address this critical gap. ECC provides strong encryption with smaller key sizes, ensuring efficient and secure data transmission. DSA adds a layer of authentication, verifying the identity of communicating entities and preventing impersonation attacks. SHA-512 ensures data integrity by generating unique hash values for data blocks and detecting unauthorized alterations. This holistic security approach fortifies the IDS against a wide array of cyber threats, ensuring confidentiality, integrity, and authenticity of the data within blockchain networks.

(ii) Inefficiency in Cryptographic Key Generation for Secure Communications: Effective cryptographic key generation is pivotal for maintaining secure communications in IDS. Traditional methods often produce keys that are either not sufficiently random or lack the necessary complexity, making them susceptible to cryptographic attacks such as brute force or predictive analysis. Moreover, static key generation techniques fail to adapt to changing security requirements and threat landscapes, leading to potential vulnerabilities over time.

Our Contribution: We have developed a novel Self-Adaptive Differential Evolution (SADE) algorithm tailored explicitly for optimizing cryptographic key generation processes. The SADE algorithm dynamically adjusts its parameters based on the evolving security context and system requirements, generating high-entropy keys resistant to various attack vectors. This adaptive mechanism ensures that the cryptographic keys remain robust and unpredictable, enhancing the overall security posture of the IDS. Additionally, the SADE algorithm improves computational efficiency by optimizing resource utilization during crucial generation, making it suitable for real-time and resource-constrained environments such as IoT networks.

(iii) Inadequate Consensus Mechanisms in Blockchain-based IDS Implementations: Blockchain technology offers decentralized and tamper-evident data storage solutions for IDS; however, many existing implementations employ consensus algorithms that are either inefficient or vulnerable to certain types of attacks. Consensus mechanisms like Proof-of-Work (PoW) or Proof-of-Stake (PoS) can be resource-intensive and do not guarantee fault tolerance in the presence of malicious or faulty nodes, leading to potential inconsistencies and vulnerabilities in the blockchain ledger.

Our Contribution: We have employed the Practical Byzantine Fault Tolerance (PBFT) algorithm to ensure reliable and efficient consensus within the blockchain-based IDS. PBFT is designed to achieve consensus even when some nodes act maliciously or fail, providing robustness against Byzantine faults. It facilitates faster and more efficient agreement among distributed nodes with lower computational overhead than PoW and PoS. By integrating PBFT, our system ensures that all honest nodes agree on the sequence and validity of transactions, maintaining the integrity and consistency of the blockchain ledger. This enhances trust and reliability in the IDS, especially critical for environments where security and quick consensus are paramount.

(iv) Ineffective Handling of Large and Decentralized Data Storage Requirements: As IDS increasingly deals with massive volumes of diverse and distributed data, traditional centralized storage systems become bottlenecks, leading to single points of failure, scalability limitations, and increased vulnerability to attacks. Existing solutions often do not effectively leverage decentralized storage technologies to manage and store large datasets efficiently and securely.

Our Contribution: We have integrated the InterPlanetary File System (IPFS) into our IDS framework for efficient off-chain data storage. IPFS is a peer-to-peer, distributed file system that enables decentralized, resilient, and scalable storage solutions. By utilizing IPFS, our system can seamlessly handle large volumes of data, distributing storage across multiple nodes to prevent centralization-related issues. This approach enhances data availability and fault tolerance and improves data retrieval speeds through content-addressed storage mechanisms. Furthermore, coupling IPFS with blockchain references ensures data integrity and traceability, as each piece of data stored off-chain can be securely linked and verified through the blockchain ledger. This synergy between blockchain and IPFS provides a robust, efficient, and secure data management solution for modern IDS requirements.

(v) Suboptimal Feature Selection and Detection Rule Optimization in IDS: Accurate intrusion detection heavily relies on effective feature selection and precise detection rules. Many current IDS models use manual or simplistic methods for feature selection, leading to irrelevant or redundant features that degrade detection performance and increase computational overhead. Similarly, static or poorly optimized detection rules can result in high false-positive rates and missed detections, undermining the IDS's effectiveness.

Our Contribution: To enhance detection accuracy and efficiency, we have applied a Genetic Algorithm (GA) for optimizing both feature selection and detection rules within the IDS. Inspired by natural selection, GA efficiently searches and identifies the most relevant and discriminative features from large datasets by evaluating and evolving multiple candidate solutions. This results in a reduced feature set that retains maximal informative value, lowering computational costs and improving detection speed. Additionally, GA optimizes detection rules by iteratively refining them to adapt to emerging threat patterns and network behaviors, thereby reducing false positives and enhancing the system's ability to accurately detect a wide range of intrusion attempts. This adaptive and automated optimization process ensures that the IDS remains effective against evolving cyber threats while maintaining operational efficiency.

(vi) Ineffectiveness in Detecting Intrusions Across Diverse and Heterogeneous IoT Networks: The rapid proliferation of IoT devices has led to highly heterogeneous network environments where devices vary widely regarding protocols, standards, and capabilities. Many existing IDS models are designed for specific network types

and fail to generalize across diverse IoT ecosystems. This results in poor detection rates and an inability to identify novel or complex attack vectors prevalent in heterogeneous settings.

Our Contribution: We have developed an IDS model based on Extreme Gradient Boosting (XGBoost), tailored to detect intrusions across diverse and heterogeneous IoT networks effectively. XGBoost is a robust and scalable machine learning algorithm known for its high performance and complex, multidimensional data capability. Our XGBoost-based model is trained on extensive and varied datasets encompassing multiple IoT scenarios and attack types, enabling it to learn intricate patterns and anomalies associated with different devices and network configurations. This approach ensures robust and accurate intrusion detection irrespective of the underlying network heterogeneity. Furthermore, the model's adaptability allows for continuous learning and improvement as new data and attack methods emerge, maintaining its effectiveness over time and across evolving IoT landscapes.

(vii) Lack of Comprehensive Comparative Performance Analyses in IDS Research: While numerous IDS models have been proposed, only some studies conduct thorough comparative analyses against a wide range of existing machine learning (ML) and deep learning (DL) approaches. This lack of comprehensive evaluation makes it challenging to ascertain the proposed solutions' relative effectiveness and practical applicability, hindering informed decision-making and adoption in real-world scenarios.

Our Contribution: To provide a clear and empirical assessment of our IDS model's performance, we have conducted extensive comparative analyses against various state-of-the-art ML and DL techniques. These evaluations encompass multiple metrics, including accuracy, precision, recall, F1-score, detection rate, and false-positive rate, across diverse datasets and attack scenarios. The results demonstrate that our proposed model consistently outperforms existing approaches, offering superior detection capabilities and operational efficiency. This comprehensive benchmarking validates our solution's effectiveness and provides valuable insights into its strengths and limitations compared to other methodologies. Such rigorous evaluation facilitates better understanding and confidence in deploying our IDS model within complex and security-critical environments.

4. Proposed Methodology

In Section 4, we introduce a Hybrid Blockchain-Based Framework designed to address critical security and privacy challenges in Intrusion Detection Systems (IDS) for heterogeneous IoT environments as shown in Figure 1. The section is structured into distinct sub-sections, each focusing on a specific aspect of the proposed methodology. It begins with a detailed problem statement outlining the shortcomings of traditional IDS solutions, particularly in handling encrypted traffic, large data volumes, and centralized vulnerabilities. The proposed framework is then described, emphasizing its two-phase approach: secure data transmission and management, followed by intrusion detection and analysis. In the first phase, the framework integrates advanced cryptographic techniques such as ECC, DSA, and SHA-512 and the novel SADE algorithm for optimal key generation, PBFT for consensus, and IPFS for decentralized data storage. The second phase focuses on optimizing IDS performance through a Genetic Algorithm and enhancing threat detection using a Proposed XGBoost-based model designed for diverse IoT networks. This comprehensive methodology aims to significantly improve the security, reliability, and adaptability of IDS in safeguarding IoT systems from evolving cyber threats.

4.1. Problem Statement

Intrusion Detection Systems (IDS) face significant challenges in maintaining security and privacy due to inadequate data protection, inefficient authentication, and difficulties handling encrypted traffic and large data volumes. Traditional IDS solutions often struggle with centralized vulnerabilities, performance issues, and inaccuracies in threat detection, especially in diverse and resource-constrained IoT environments. This research proposes a comprehensive solution to address these problems. It introduces a Hybrid Blockchain-Based Framework that integrates ECC, DSA, and SHA-512 to enhance data privacy, integrity, and authentication. The solution also includes a Self-Adaptive Differential Evolution (SADE) algorithm for optimizing cryptographic key generation, the PBFT Consensus Algorithm to improve resilience and prevent centralized failures, IPFS for secure and decentralized off-chain data storage, Genetic Algorithm Optimization for refining detection rules and reducing false alerts, and an XGBoost-Based

Intrusion Detection Model tailored for detecting sophisticated threats. Together, these innovations aim to significantly enhance the effectiveness, reliability, and adaptability of IDS in protecting IoT systems from evolving cyber threats.

4.2. Overview of Proposed Framework

The proposed research framework addresses critical security and privacy challenges in Intrusion Detection Systems (IDS) for Internet of Things (IoT) environments through several innovative contributions. The framework is divided into two distinct yet interrelated phases: Secure Data Transmission and Management and Intrusion Detection and Analysis. Each phase incorporates advanced methodologies and technologies to ensure robust data security, system reliability, and intrusion detection accuracy. The proposed framework is divided into two phases:

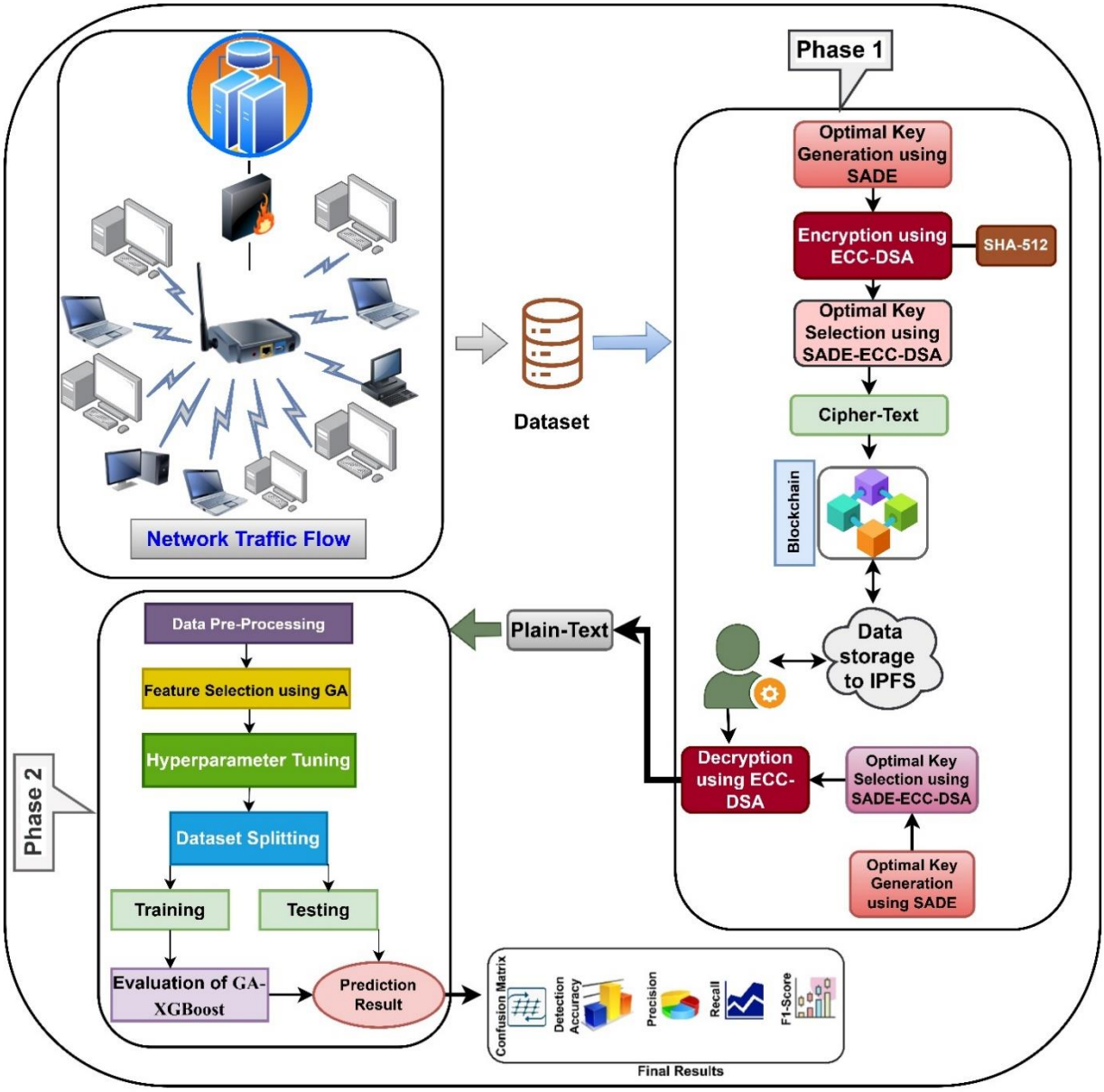


Figure 35: A Hybrid AI-Blockchain-Enabled Framework for Enhancing the Security of Intrusion Detection Systems in the Internet of Things Ecosystem

(i) First Phase: Secure Data Transmission and Management

This phase is designed to establish a secure and reliable environment for transmitting and managing sensitive data, especially in scenarios involving highly confidential information, such as patient healthcare records. The framework employs a combination of advanced cryptographic techniques and decentralized data storage mechanisms to protect data integrity, confidentiality, and availability.

- **Integration of ECC, DSA, and SHA-512:** The integration of Elliptic Curve Cryptography (ECC), the Digital Signature Algorithm (DSA), and the SHA-512 hash function ensures a robust cryptographic framework for secure communication. ECC is employed for data encryption, offering a high level of security with smaller key sizes, making it efficient and suitable for resource-constrained IoT devices. Its computational efficiency reduces overhead while maintaining robust encryption standards. DSA authenticates data exchanges by verifying the identities of communicating entities, ensuring that only authorized devices participate in the network. Additionally, SHA-512 generates unique hash values for every data packet, enabling the detection of any unauthorized modification and ensuring data integrity. Together, these cryptographic techniques provide comprehensive protection by securing data, authenticating users, and detecting tampering.
- **Development of a Novel SADE Algorithm for Optimal Key Generation:** The framework introduces a novel Self-Adaptive Differential Evolution (SADE) algorithm to enhance the quality and security of cryptographic keys. Unlike traditional key generation techniques, SADE dynamically adjusts its parameters, such as mutation and crossover rates, based on the security requirements and the system environment. This adaptability generates high-entropy keys that are more resistant to cryptographic attacks, including brute force. By optimizing the key generation process, SADE ensures that the encryption and authentication mechanisms remain both secure and efficient, addressing the varying needs of IoT networks while mitigating potential vulnerabilities.
- **Employment of the PBFT Consensus Algorithm:** The Practical Byzantine Fault Tolerance (PBFT) consensus algorithm is implemented to establish trust and ensure accurate decision-making across the blockchain network. PBFT is particularly suited for environments where nodes may act maliciously or unpredictably, as it achieves consensus even in the presence of Byzantine faults. By requiring agreement among a majority of nodes, PBFT maintains the consistency and integrity of the blockchain ledger. This ensures that data recorded on the blockchain is accurate, tamper-proof, and reliable, providing a secure foundation for managing sensitive information.
- **Utilizing IPFS for Off-Chain Data Storage:** The InterPlanetary File System (IPFS) is integrated to address the challenges of storing large volumes of data while maintaining decentralization. IPFS provides a scalable and resilient storage solution by decentralizing data storage across multiple nodes. It ensures that data remains accessible even if some nodes go offline. Additionally, data stored in IPFS is referenced on the blockchain via cryptographic hashes, ensuring both its integrity and authenticity. This combination of on-chain and off-chain storage allows the framework to efficiently manage extensive datasets, such as IoT logs and healthcare records, without compromising security or scalability.

(ii) Second Phase: Intrusion Detection and Analysis

The second phase focuses on enhancing the detection and analysis of network intrusions, specifically within heterogeneous IoT networks. This phase aims to improve the accuracy and efficiency of identifying malicious activities, thereby strengthening the overall security posture of the network.

- **Application of Genetic Algorithm (GA) for IDS Optimization:** To optimize the performance of the Intrusion Detection System (IDS), the framework employs a Genetic Algorithm (GA). GA simulates the process of natural evolution to refine detection rules and select the most relevant features from network traffic data. By iteratively evolving a population of potential solutions, GA identifies patterns that distinguish between normal and malicious activities. This process reduces the false positive rate, ensuring that legitimate activities are not mistakenly flagged as threats. Moreover, selecting optimal features improves the efficiency of the IDS, enabling it to detect genuine threats more accurately and respond to them promptly.
- **Design of XGBoost-Based Model for Intrusion Detection:** The framework includes an XGBoost-based model for detecting intrusions in heterogeneous IoT networks. XGBoost is a robust machine-learning algorithm known for its high accuracy and efficiency, particularly in handling large and complex datasets. The model leverages the XGBoost gradient-boosting framework to identify various intrusions, even in

diverse and dynamic IoT environments. By effectively analyzing data patterns and distinguishing between normal and malicious activities, the XGBoost-based model provides a robust solution for intrusion detection. This ensures that the network remains secure against a wide range of threats, maintaining the stability and reliability of IoT systems.

4.2.1. ECC-DSA-SHA-512 for Secure Data Management

In the context of securing data management within a blockchain-based framework, the hybrid cryptographic approach involving Elliptic Curve Cryptography (ECC), Digital Signature Algorithm (DSA), and Secure Hash Algorithm 512 (SHA-512) is employed to enhance the security and integrity of data. This combined approach leverages the strengths of each component to provide a robust mechanism for safeguarding sensitive information.

Elliptic Curve Cryptography (ECC) provides a high level of security with relatively smaller key sizes compared to traditional cryptographic methods. The ECC algorithm operates over elliptic curves defined over finite fields, offering efficient and secure public-key cryptography. Specifically, ECC employs the elliptic curve equation:

$$y^2 = x^3 + ax + b \bmod p \quad (1)$$

Where p is a prime number defining the finite field. The security of ECC is based on the difficulty of the Elliptic Curve Discrete Logarithm Problem (ECDLP), which ensures that private keys remain secure against computational attacks. The private key d is selected from the range, open bracket 1, from the range $[1, n-1]$, and the corresponding public key Q is computed as $Q = d.G$, where, G is the base point on the elliptic curve, and n is the order of G.

A Digital Signature Algorithm (DSA) is integrated to provide digital signatures that authenticate and verify data integrity. The DSA, when combined with ECC, enhances the efficiency and security of the signing process. The signing process involves generating a random number k, calculating the point $R = k.G$, and computing the signature components $r \; and \; s \; as \; follows$:

$$r = x_R \bmod n \quad (2)$$

$$s = k^{-1}(h + d.r) \bmod n \quad (3)$$

where x_R is the x-coordinate of R, k^{-1} is the modular inverse of k, and h is the hash of the message. The signature (r, s) serves as proof of the authenticity of the message.

Secure Hash Algorithm 512 (SHA-512) is used to hash the message before signing, ensuring the data remains secure and tamper-proof. The hashing function processes the message M to produce a 512-bit hash value h:

$$h = SHA - 512(M) \quad (4)$$

This hash value is then used in the signing process to ensure the integrity and non-repudiation of the data.

In the blockchain framework, these components are combined to provide a comprehensive security solution. ECC ensures efficient key management and secure public-key operations, DSA facilitates the generation of verifiable digital signatures, and SHA-512 guarantees data integrity through secure hashing. Together, these cryptographic techniques form a hybrid approach that enhances the security and reliability of data management within the blockchain.

ECC, DSA, and SHA-512 integration form a cohesive hybrid algorithm. The process begins with data preparation, where the message M is hashed using SHA-512 to produce h. ECC then provides the key management framework, generating the private key d and public key Q. The DSA utilizes these keys alongside the ephemeral key k, to produce the signature (r, s). This signature is appended to the transaction data, which is then stored in the blockchain.

For verification, the public key Q is used alongside the received signature (r, s) to confirm the authenticity of the data. The hash of the received message is recalculated, and the validity of the signature is checked using the relation:

$$u_1 = s^{-1} \, h \bmod n \quad (5)$$

$$u_2 = s^{-1}\, r \bmod n \quad (6)$$

$$R' = u_1 . G + u_2 . Q \quad (7)$$

The signature is deemed valid if the x-coordinate of R' matches r. This ensures that the data is not altered and originates from a legitimate source.

By merging ECC, DSA, and SHA-512, the proposed hybrid algorithm provides a comprehensive security mechanism within the blockchain framework. It ensures that data is securely managed, preventing unauthorized access and guaranteeing data integrity and authenticity.

4.2.2 Self-Adaptive Differential Evolution (SADE) for Key Generation

In the context of securing data management within a blockchain-based framework, the integration of Elliptic Curve Cryptography (ECC), the Digital Signature Algorithm (DSA), and the Secure Hash Algorithm 512 (SHA-512) has been recognized as a robust approach for ensuring data integrity, authenticity, and privacy. However, to further enhance the system's efficiency and security, the Self-Adaptive Differential Evolution (SADE) algorithm has been employed to optimize the key generation process. This optimization is necessary because the strength of ECC and DSA heavily depends on the quality of the cryptographic keys generated. SADE dynamically adjusts its parameters to produce optimal keys without manual tuning, making it particularly suitable for resource-constrained environments where computational power and storage are limited. By generating secure and efficient keys, SADE reduces computational overhead and enhances the system's resistance to cryptographic attacks, including brute force and cryptanalysis. The algorithm's adaptability ensures that the blockchain network can scale efficiently, maintaining high performance and security as the number of transactions increases. Additionally, SADE minimizes vulnerabilities associated with weak or predictable keys, thus contributing to a more secure overall system. SADE's proactive optimization of key generation has been essential in maintaining the blockchain framework's integrity and protecting sensitive data from emerging cyber threats.

This section proposes a novel Self-Adaptive Differential Evolution (SADE) algorithm to optimize cryptographic key generation within the ECC-DSA-SHA-512 framework. SADE dynamically adjusts parameters to enhance key quality, ensuring robust security and high entropy. The role of SADE in optimizing the key is as follows:

1. **Key Generation in ECC using SADE**

   Key generation in Elliptic Curve Cryptography (ECC) involves selecting a private key d and calculating the corresponding public key Q. SADE optimizes the private key selection through a population-based approach, focusing on improving security metrics like entropy and minimizing correlation.

   a) **Initialization:** A population of candidate private keys d_i is initialized within the range$(1, n-1)$, where n is the order of the base point G on the elliptic curve E. The population size is denoted by NP.

   $$\mathrm{d_i} = random(1, n-1), i = 1,2 \ldots\ldots NP \quad (8)$$

   b) **Mutation:** For each candidate d_i, a mutant vector v_i is generated as follows:

   $$v_i = d_{best} + F.(d_{r1} - d_{r2}) + \varphi.(d_{r3} - d_{r4}) \quad (9)$$

Where d_{best} is the best solution vector in the current population, F is the mutation factor, adaptively adjusted in SADE, φ is an additional exploration factor, also adaptively adjust, $d_{r1}, d_{r2}, d_{r3}, d_{r4}$ are randomly selected distinct solutions from the population.

The mutation factor F and the exploration factor φ are dynamically updated as:

$$F = F_{min} + random(0,1).(F_{max} - F_{min}) \quad (10)$$

$$\varphi = \varphi_{min} + random(0,1).(\varphi_{max} - \varphi_{min}) \quad (11)$$

Where $F_{min}, F_{max}, \varphi_{min}$ and φ_{max} are predefined bounds.

c) Crossover: The crossover operation forms a trial vector u_i by combining elements from the current candidate $d_i$and the mutant vector v_i:

$$u_i[j] = \begin{cases} v_i[j], & if\ random\ j \leq\ c_r \\ d_i[j], & Otherwise \end{cases} \quad (12)$$

where c_r is the crossover rate, and $random\ j$ is a uniformly distributed random number.

d) Selection: The trial vector u_i is evaluated against the objective function, and the better solution is selected for the next generation:

$$d_i^{new} = \begin{cases} u_i, & if\ (u_i) >\ f(d_i) \\ d_i, & Otherwise \end{cases} \quad (13)$$

The objective function f is designed to maximize security properties like entropy and minimize the correlation between keys.

e) Convergence Criteria: The algorithm iterates until a convergence criterion is met, such as a maximum number of generations G_{max}:

$$G \geq\ G_{max} \quad (14)$$

2. **Integration with ECC-DSA:** Upon determining the optimal private key d using SADE, the ECC-DSA process proceeds with signing and verification.

a) Message Hashing: The message M is hashed using SHA-512:

$$h = SHA - 512(M) \quad (15)$$

b) Signature Generation: A random number k is selected, and the point $R = k.G$ is computed. The signature (r, s) is generated as follows:

$$\mathrm{r} = \mathrm{x_R} \bmod \mathrm{n} \quad (16)$$

$$\mathrm{s} = \mathrm{k^{-1}(h + d.r)} \bmod \mathrm{n} \quad (17)$$

where x_R is the x-coordinate of the point R, and k^{-1} is the modular inverse of $k\ mod\ n$.

c) Signature Verification: To verify the signature, the following calculations are performed:

- Calculate u_1 and u_2:

$$u_1 = s^{-1}\ h\ mod\ n \quad (18)$$

$$u_2 = s^{-1}\ r\ mod\ n \quad (19)$$

- Compute the point R':

$$R' = u_1.G + u_2.Q \quad (20)$$

- Verify if R' yields the same value of r:

$$r = x_{R'}\ mod\ n \quad (21)$$

The signature is valid if $r = r'$.

The SADE algorithm adaptive mechanism balances exploration and exploitation, ensuring robust and high-quality key generation. The integration with ECC-DSA-SHA-512 provides a secure and efficient framework for data

management in blockchain networks, capable of withstanding evolving threats and maintaining high levels of security and performance. SADE's contribution to secure data management in blockchain networks ensures that the system can adapt to changing conditions, optimize key generation, and maintain robust security measures.

Algorithm: SADE for Cryptographic Key Generation
Input: NP, G_max, CR, F_min, F_max, φ_min, φ_max, G, n
Output: Optimized Private Key `d_best`

1. Initialize population {d_1, d_2, ..., d_NP} with random values in (1, n-1)
2. Evaluate objective function f(d_1, d_2, ..., d_NP)
3. For generation G = 1 to G_max do:
   a. For each candidate d_i:
      i. Generate mutant vector v_i:
         v_i = d_best + F(d_r_1 - d_r_2) + φ(d_r_3 - d_r_4)
         Update F and φ dynamically.
      ii. Perform crossover to create trial vector u_i.
      iii. Evaluate u_i against f(d_i).
      iv. Update d_i = u_i if f(u_i) > f(d_i).
   b. Update d_best.
4. Return d_best.

4.2.3. Consensus Mechanism for Proposed Framework

In a PBFT-based blockchain network, the decision regarding data inclusion on the blockchain ledger is managed by a set of validators or replicas, denoted as $N = \{N_1, N_2, N_3, \ldots \ldots \ldots \ldots N_n\}$ is responsible for participating in the consensus protocol. The PBFT consensus mechanism is designed to agree on the ledger's state despite Byzantine faults, where a fraction of nodes f may act arbitrarily or maliciously. It is assumed that $f < \frac{n}{3}$ where n is the total number of nodes. The PBFT processes in the proposed framework:

a. Transaction Proposal:

- **Transaction Submission:** Transactions are submitted to the network by participants $P = \{P_1, P_2, P_3 \ldots \ldots P_n\}$. Each transaction T_i includes data and its associated cryptographic signature σ_i, which ensures the transaction's authenticity and integrity:

$$T_i = (d_i, \sigma_i) \qquad (22)$$

Where d_i represents the transaction data and σ_i is the signature generated using the sender's private key.

- **Pre-prepare Phase:** The primary node $N_{primary}$ proposes a new block B that contains the transactions received. The block proposal B is represented as:

$$B = \{T_1, T_2, T_3 \ldots . T_k\}$$

where k is the number of transactions included in the block.

b. Block Creation:

- **Prepare Phase:** The primary node broadcasts the proposed block B to all other nodes in the network. Each node N_i checks the validity of the block by verifying each transaction's signature and ensuring compliance with the blockchain's rules:

$$Valid\ (T_i) = Verify\ (d_i, \sigma_i)$$

- **Prepare Messages:** Nodes send a prepare message $M_{prepare}$ to all other nodes. This message confirms the reception and validation of the proposed block B:

$$M_{prepare} = (Prepare, B, N_i)$$

where N is the node sending the prepared message.

c. Block Verification:

- **Commit Phase:** Each node collects prepare messages from other nodes. When a node N_i receives prepare messages from at least $\frac{2n}{3}$ nodes (a supermajority), it considers the block to be in the "commit" phase. The threshold for commitment is represented as:

$$Commit(B) = |\{M_{prepare} | M_{prepare}\ Confirms\ B\ from\ \frac{2n}{3}\ \text{nodes}|$$

- **Commit Messages:** Nodes then broadcast "commit" messages M_{commit} to all other nodes. These messages indicate the readiness to finalize the block B:

$$M_{commit} = (Commit, B, N_i)$$

d. Consensus:

- **Finalization:** When a node N_i receives commit messages from a supermajority of nodes, it finalizes the block B. The block is then added to the blockchain ledger. The finalization criterion is:

$$Finalize(B) = |\{M_{prepare} | M_{prepare}\ Confirms\ B\ from\ \frac{2n}{3}\ \text{nodes}|$$

- **Broadcast:** The finalized block B is broadcast to the network, and all nodes update their local copies of the ledger with the new block.

In the PBFT consensus algorithm, deciding what data to include on the blockchain ledger involves structured phases: transaction proposal, block creation, verification, and consensus. Initially, the primary node proposes and includes transactions in a block. The block undergoes a preparation phase where its validity is checked and prepared messages are disseminated. Once a supermajority of prepared messages is received, the commit phase is initiated, broadcasting commit messages. The block is finalized and added to the blockchain ledger upon receiving commit messages from a supermajority. This structured approach ensures consistency and reliability of the blockchain ledger, even in the presence of Byzantine faults.

4.2.4. Privacy-Preserving Mechanism

The proposed system introduces privacy-preserving mechanisms using Homomorphic Encryption (HE) and Zero-Knowledge Proofs (ZKPs). These cryptographic techniques ensure that sensitive data transmitted across the IoT network remains confidential and secure, even in untrusted environments.

a. Homomorphic Encryption (HE)

Homomorphic encryption allows computations to be performed on encrypted data without decrypting it. This is essential in an IoT environment where sensitive data, such as personal health records or financial information, must remain private during processing. In IoT, devices transmit encrypted data to a central server for processing. The server performs computations on the ciphertext and returns the encrypted result, which can only be decrypted by the rightful owner of the data.

The basic operation of homomorphic encryption can be described by the following:

i. Encryption function: Let $Enc(x)$ represent the encryption of a message x.

$$Enc(x) = x^e\ mod\ n$$

Where e is the public encryption exponent, and n is part of the public key.

ii. Homomorphic property: Homomorphic encryption allows certain operations on encrypted data to be equivalent to operations on the decrypted data. For instance, for two encrypted values $Enc(x)$ and $Enc(y)$, the following holds:

$$Enc(x + y) = Enc(x) + Enc(y)$$

$$Enc(xy) = Enc(x).Enc(y)$$

Using this property, a server can perform necessary operations (like addition or multiplication) on the encrypted data without ever decrypting it, ensuring that sensitive information remains secure.

In the case of IoT devices in a healthcare system transmitting encrypted patient data, a device could send the encrypted medical readings to a server. The server could perform necessary computations (e.g., computing averages or performing diagnostic algorithms) on the encrypted data, which ensures that the medical data is never exposed to the server during the computation process.

b. Zero-Knowledge Proofs (ZKPs)

Zero-Knowledge Proofs (ZKPs) are cryptographic protocols that allow one party (the prover) to prove to another party (the verifier) that they know a value (e.g., a secret or a password) without revealing any information about the value itself. In the context of IoT, ZKPs can be used to verify data authenticity, such as a user's identity or sensor data, without revealing any sensitive information. This allows for secure authentication in IoT networks, ensuring that sensitive data is not exposed during verification.

Let R be the secret data the prover knows (for example, a private key or an encrypted value). The prover wants to prove to the verifier that they know R without revealing it. Using ZKPs, the prover can generate a proof π that the verifier can verify, but at no point is R disclosed. The protocol involves three main steps:

i. Commitment Phase: The prover commits to a value using a cryptographic commitment function $Commit(R)$.

ii. Challenge Phase: The verifier challenges the prover to prove they know the value without revealing it.

iii. Response Phase: The prover provides a response that satisfies the challenge without disclosing the actual value R.

The verification process ensures that the prover knows R, but the verifier learns nothing about R beyond its validity.

Let C be the commitment to a secret value R, where:

$$C = Commit(R)$$

Let x be the challenge from the verifier, and s be the response from the prover. The prover then sends s to the verifier, who checks whether:

$$Verify\ (C, x, s) = 1$$

If the verification is successful, the prover successfully convinces the verifier they know R without revealing any details of R.

Consider a scenario where an IoT sensor in a healthcare application needs to verify its identity to access a secure server. Using a Zero-Knowledge Proof, the sensor can prove it has valid credentials without transmitting sensitive information, such as its private key or encrypted data.

c. Integration of HE and ZKPs for IoT Framework

In the proposed framework for intrusion detection in IoT, the integration of HE and ZKPs ensures that:

i. Confidentiality: Data from IoT devices, such as sensor readings, remains encrypted during transmission and processing, protecting sensitive information.

ii. Authentication: Devices and users can authenticate their identities without exposing private information, ensuring only authorized entities can access or process the data.

The system flow would be as follows:

- IoT devices encrypt sensitive data using homomorphic encryption.
- The encrypted data is sent to the central server, where computations are performed on the encrypted data (e.g., performing machine learning inference).
- The result is returned as an encrypted response decrypted by the IoT device.
- During authentication, devices use Zero-Knowledge Proofs to prove their identity to the central server without revealing private information.

The complete privacy-preserving framework can be represented as:

- Step 1: Encryption: Each IoT device encrypts its data using homomorphic encryption:
$$Enc(x) = x^e \ mod \ n$$
where x is the data and e is the encryption exponent.
- Step 2: Computation: The server performs necessary computations on the encrypted data, such as:
$$Enc(x + y) = Enc(x) + Enc(y)$$
to generate results without decrypting the data.

- Step 3: Zero-Knowledge Proofs for Authentication: IoT devices and users prove their identity using ZKPs without revealing sensitive credentials:
$$Verify\ (C, x, s) = 1$$
Where C is the commitment, x is the challenge, and s is the response.

The integration of Homomorphic Encryption and Zero-Knowledge Proofs into the proposed IoT-based intrusion detection framework enhances data privacy and security. These cryptographic techniques ensure that sensitive data remains confidential during computation and that users and devices can prove their identity without disclosing private information. This approach offers a robust solution to privacy concerns in IoT environments, ensuring that both data confidentiality and authentication processes are securely handled.

4.2.5. Data Storage through IPFS

The InterPlanetary File System (IPFS) is introduced in the proposed framework as a decentralized storage solution to address blockchain systems' scalability and storage limitations. It efficiently stores large datasets, such as IoT network logs, without burdening the blockchain with high storage demands. The integration involves storing data files in IPFS while maintaining only the file's hash on the blockchain, known as the Content Identifier (CID). This CID acts as a pointer to the file's exact location in the IPFS network, ensuring data privacy and reducing storage overhead. Consistency between the blockchain and IPFS is achieved through hash verification. When data is retrieved, the system compares the CID in the blockchain with the retrieved file's hash in IPFS to ensure integrity. Any mismatch flags tampering or corruption. Regular integrity checks reinforce this mechanism, maintaining the reliability of stored data. Moreover, encryption using Elliptic Curve Cryptography (ECC) protects data before uploading it to IPFS. Decryption keys are managed using a Role-Based Access Control (RBAC) system, which ensures that only authorized users, such as healthcare providers or researchers, can access the data.

The framework also guarantees data immutability and traceability. Blockchain's immutable ledger secures the CID, ensuring the data remains unaltered in IPFS. Furthermore, IPFS's versioning feature enables updates to stored files while retaining a complete history of changes. This ensures transparency and traceability without compromising data integrity. When a user requests data, the system retrieves the CID from the blockchain, fetches the file from IPFS, and verifies its integrity against the stored hash before decryption. This integration offers several advantages. By offloading large datasets to IPFS, the blockchain becomes more scalable and can handle higher transaction volumes without performance degradation. It is also cost-efficient, as storing data off-chain in IPFS reduces the expense associated with on-chain storage. The combination of IPFS encryption and blockchain immutability provides robust security, while the distributed nature of IPFS enhances data availability and resilience against failures.

However, challenges such as data availability and retrieval latency are acknowledged. Files in IPFS may become unavailable if nodes fail to host them. To counter this, pinning services ensure critical files remain accessible. Retrieval latency, a potential drawback of IPFS's distributed nature, is mitigated by implementing caching mechanisms for frequently accessed files. The integration of IPFS with blockchain in the proposed framework provides a secure, scalable, and efficient solution for off-chain data storage. It effectively handles large datasets while maintaining privacy, integrity, and availability, making it particularly suitable for IoT applications in sensitive domains like healthcare. Future research could explore further optimizations in data retrieval speeds and advanced encryption methods to enhance system performance.

4.3. Proposed Intrusion Detection Model

This section presents a comprehensive Proposed Intrusion Detection System (IDS) model tailored for securing heterogeneous IoT environments, as shown in Figure 2. The section begins with data preparation, involving meticulous preprocessing and normalization techniques to ensure the dataset is clean, well-organized, and balanced, optimizing the accuracy of the subsequent machine learning models. Next, we employ a Genetic Algorithm (GA) to refine detection rules, enhancing the IDS's ability to detect intrusions while minimizing false positives and negatives. The GA's optimization process, from rule initialization to final selection, is detailed, demonstrating its effectiveness in improving IDS performance. Finally, we introduce an XGBoost-based model, an ensemble of decision trees designed to identify cyber threats within these networks. The model leverages iterative training and optimization to effectively capture complex patterns, thereby enhancing the robustness of the IDS in detecting intrusions in diverse IoT environments.

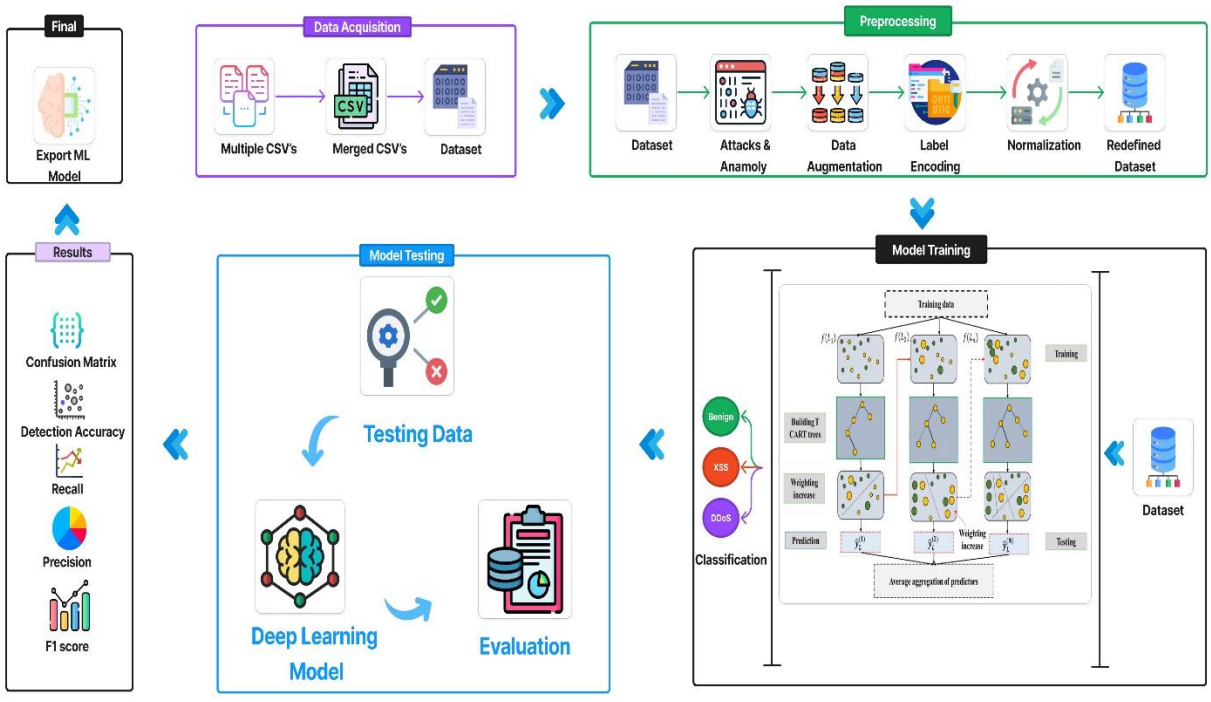


Figure 36: Overview of Proposed Architecture for Intrusion Detection System

4.3.1. Data Preparation

Adequate data preparation is integral to the success of machine learning and deep learning models, as it ensures that the data is appropriately organized and cleansed before inputting into the algorithms. This process is essential for optimizing the learning process and improving model accuracy. Our research employs a comprehensive two-step strategy for data preparation, consisting of Data Pre-processing and Data Normalization.

i. **Data Pre-Processing:** In the data pre-processing phase, categorical features with nominal values were converted into numerical representations using label encoding. This method aligns these features with the input requirements of the neural network. Additionally, irrelevant features, such as date, time, and timestamp columns, were removed from the dataset. These features were deemed non-contributory to the prediction outcomes, and their exclusion helped streamline the dataset for more efficient processing.

ii. **Data Normalization:** We employed data normalization techniques to address the issue of feature imbalance, where specific attributes exhibited disproportionately high values that could skew the model's performance. Specifically, the min-max scaling technique was utilized to map the data to a standardized range between 0 and 1 while preserving the original distribution of the data. This technique is mathematically represented as:

$$y = \frac{X - Xmin}{Xmax - Xmin} \quad (1)$$

Where x and y are the original and normalized values, the feature's minimum and maximum values are given by x_{min} and x_{max}, respectively.

As an initial step in the data preparation process, all rows containing NaN or Infinity values were removed to mitigate any potential negative impact on model performance preemptively. Subsequently, the Scikit-learn label encoder was employed to convert non-numerical values to numerical ones. The dataset's sole non-numerical feature, 'Label,' was converted to a binary format using this encoder. Comprehensive data normalization was then achieved using the Min-Max scaler function, as referenced in [37].

4.3.2. Genetic Algorithm (GA) for Optimizing IDS Performance

In Intrusion Detection Systems (IDS), Genetic Algorithms (GA) optimize performance by refining detection rules and features, evolving a population of candidate solutions (chromosomes) to find the optimal set of rules and feature combinations. The goal is to maximize the system's ability to detect intrusions while minimizing false positives and false negatives.

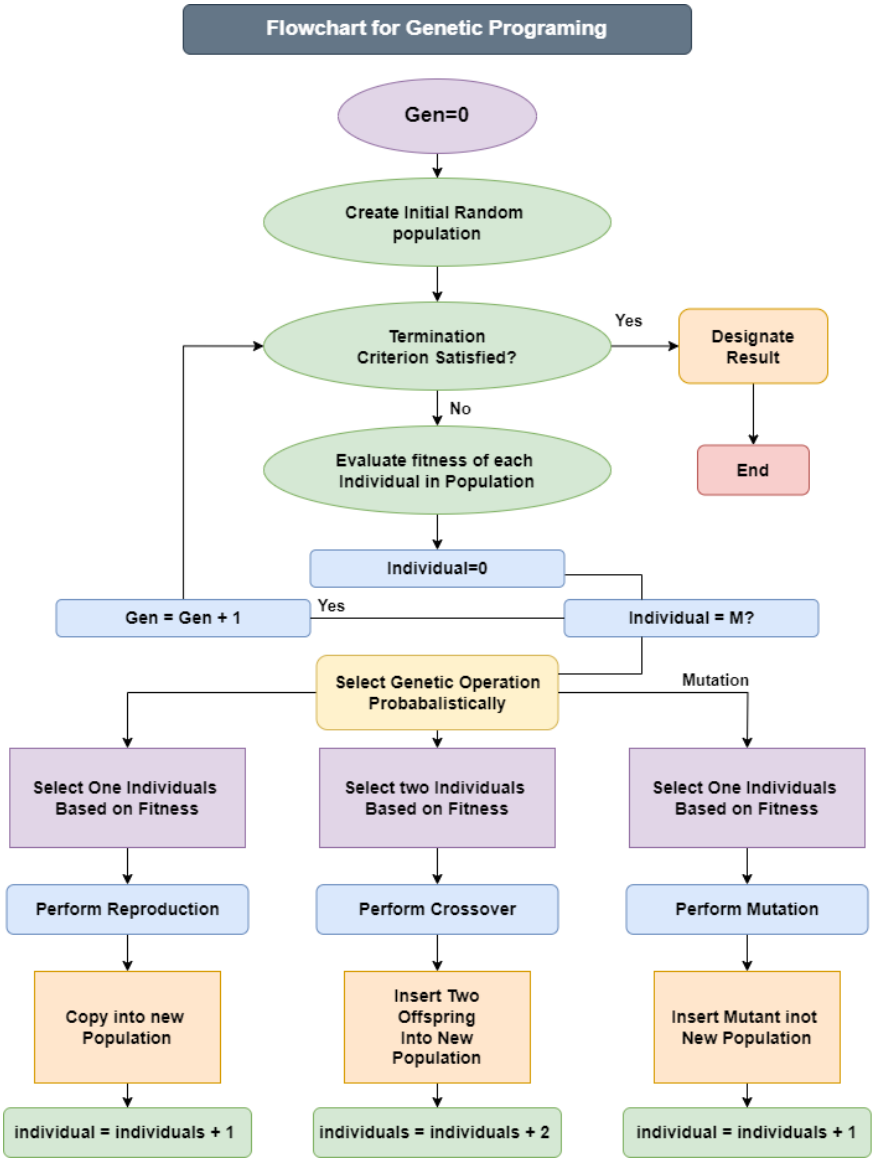


Figure 37: Proposed Flow Chart of Optimized Genetic Algorithm for Feature Selection

- **Detection Rule in IDS:** Detection rules are criteria that IDS uses to identify malicious activities within a network. These rules specify patterns or signatures associated with known threats or describe normal network behavior to identify anomalies. This research develops detection rules specifically for the proposed system, leveraging standard sets like the Snort rule set and other established databases. We consider network traffic features such as IP addresses, ports, protocols, payload patterns, and timing characteristics. For example, a rule might flag an incoming packet with a specific IP address and port combination as suspicious:

  - **Rule ID**: 10001
  - **Rule Name**: TCP SYN Flood
  - **Condition**: if (source IP == known malicious IP) AND (destination port == 80) And (packet count > 1000 within 1 minute) then alert
  - **Action**: Alert and log the incident

  These rules form the initial pool for optimization. The GA refines these rules to enhance detection accuracy and efficiency.

- **Genetic Algorithm Optimization Process:** The GA optimization process involves the following steps:

➢ **Initialization:** A population of chromosomes, each representing a set of detection rules, is randomly generated. Each chromosome encodes a potential solution, with genes representing specific rule parameters.

➢ **Fitness Function:** The fitness function evaluates each chromosome's performance by measuring the IDS's ability to detect intrusions, considering true positives and false negatives. The fitness function f is defined as:

$$f = \alpha * TPR - \beta * FPR$$

➢ **Selection:** Chromosomes are selected based on their fitness scores, with fitter individuals having a higher chance of being chosen for reproduction. Roulette Wheel Selection is used, a probabilistic method where a roulette wheel is divided proportionally according to fitness scores. Chromosomes are selected by spinning the wheel, with fitter individuals having a larger slice. Mathematically, the probability p_i of selecting chromosome i is:

$$p_i = \frac{f_i}{\sum_{j=1}^{N} f_i}$$

➢ **Crossover:** Edge Recombination Crossover (ERC) is applied, preserving relative ordering and adjacency of genes from parent chromosomes. ERC constructs an adjacency list for each gene, representing possible successors from both parents. The offspring is generated by traversing the adjacency list, selecting genes with the fewest neighbors to minimize breaking gene adjacency.

Let $\boldsymbol{A \; and \; B}$ be two parent chromosomes. The ERC creates offspring by maintaining the order and adjacency of edges (connections between genes) present in $A \; and \; B$. Mathematically, if $A = (a_1, a_2 \ldots \ldots \ldots \ldots . a_n)$ and $B = (b_1, b_2 \ldots \ldots \ldots \ldots . b_n)$, the adjacency list for a gene g contains the adjacent genes from $A \; and \; B$. For each gene g in the offspring, the next gene is chosen based on the adjacency list, prioritizing genes with fewer neighbors.

➢ **Mutation:** Insertion Mutation is applied to maintain diversity. A gene is selected randomly, removed, and inserted at a new random position within the chromosome, altering the sequence without changing the set of genes.

Let $C = (c_1, c_2 \ldots \ldots \ldots \ldots \ldots . c_n)$ be a chromosome. The mutation operation can be represented as:

$$C = (c_1, \ldots \ldots . c_{k-1}, c_{k+1}, \ldots \ldots \ldots c_n, c_k)$$

where gene c_k is moved to a new position at the end of the sequence.

➢ **Termination:** The algorithm iteratively applies these operators until a termination condition is met. Terminating conditions include reaching a maximum number of generations G_{max} . This condition ensures that the algorithm halts after a fixed number of iterations, regardless of whether the optimal solution has been found. Mathematically, this condition can be expressed as:

$$Terminate \; if \; G \geq G_{max}$$

The GA optimization results in an optimized set of detection rules that improve the accuracy and efficiency of identifying intrusions. This process allows the IDS to adapt to new and evolving threats by refining its detection capabilities based on real-world data and scenarios. The optimized rules enhance the system's ability to recognize previously unseen patterns indicative of malicious behavior, thus improving security in IoT environments.

Once the GA has refined the detection rules, the optimized IDS rules are validated through performance metrics such as accuracy, detection rate, and efficiency. The optimized rules improve the IDS's adaptability to evolving threats by enhancing detection capabilities and reducing the likelihood of false positives. Additionally, the refined rules enable the IDS to focus on more relevant network traffic features, improving overall system performance in real-time

deployment environments. By using GA to optimize detection rules and feature sets, the IDS becomes more robust, capable of adapting to new intrusion patterns, and provides a higher level of security for IoT systems where threats are continuously evolving.

The GA-based approach significantly improves the performance of IDS by ensuring that the system's detection mechanisms evolve with the threat landscape. Combining optimization techniques such as crossover and mutation, the GA continuously refines detection rules to adapt to new, sophisticated intrusions while maintaining high accuracy and minimizing detection errors. This adaptability is crucial for maintaining the effectiveness of intrusion detection systems in the dynamic and rapidly evolving world of IoT security.

4.3.3. XGBoost-Based Model for Detecting Intrusion in Heterogeneous IoT Networks

An XGBoost-based model is proposed for detecting cyber threats in heterogeneous IoT networks. XGBoost, or Extreme Gradient Boosting, is employed due to its effectiveness in classification tasks. It leverages an ensemble of weak learners, specifically decision trees, to create a robust predictive model for identifying malicious activities. Table 5 presents the key hyperparameters used in the XGBoost-based intrusion detection model and their descriptions and values.

Table 16: Hyperparameters for XGBoost-Based Intrusion Detection Model in Heterogeneous IoT Networks

Hyperparameter	Description	Values
Learning rate	Controls the step size at each iteration while moving toward a minimum.	0.3
n-estimator	Number of boosting rounds or trees to build.	100
Max-depth	Maximum depth of the individual trees. A higher value leads to a more complex model.	6
Min-child weight	Minimum sum of instance weight (hessian) needed in a child.	1
Sub sample	Fraction of samples to use for fitting each tree. Prevents overfitting.	1
Colsample-bytree	Fraction of features to use for fitting each tree.	1
Gamma	Minimum loss reduction required to make a further partition. Higher values lead to more conservative models.	0
Lambda	L2 regularization term on weights (Ridge regression).	1
Alpha	L1 regularization term on weights (Lasso regression).	0
Scale-pos-weight	Controls the balance of positive and negative weights. Used in imbalanced classes.	1
booster	Type of boosting model to use. Options: gbtree, gblinear, dart.	gbtree

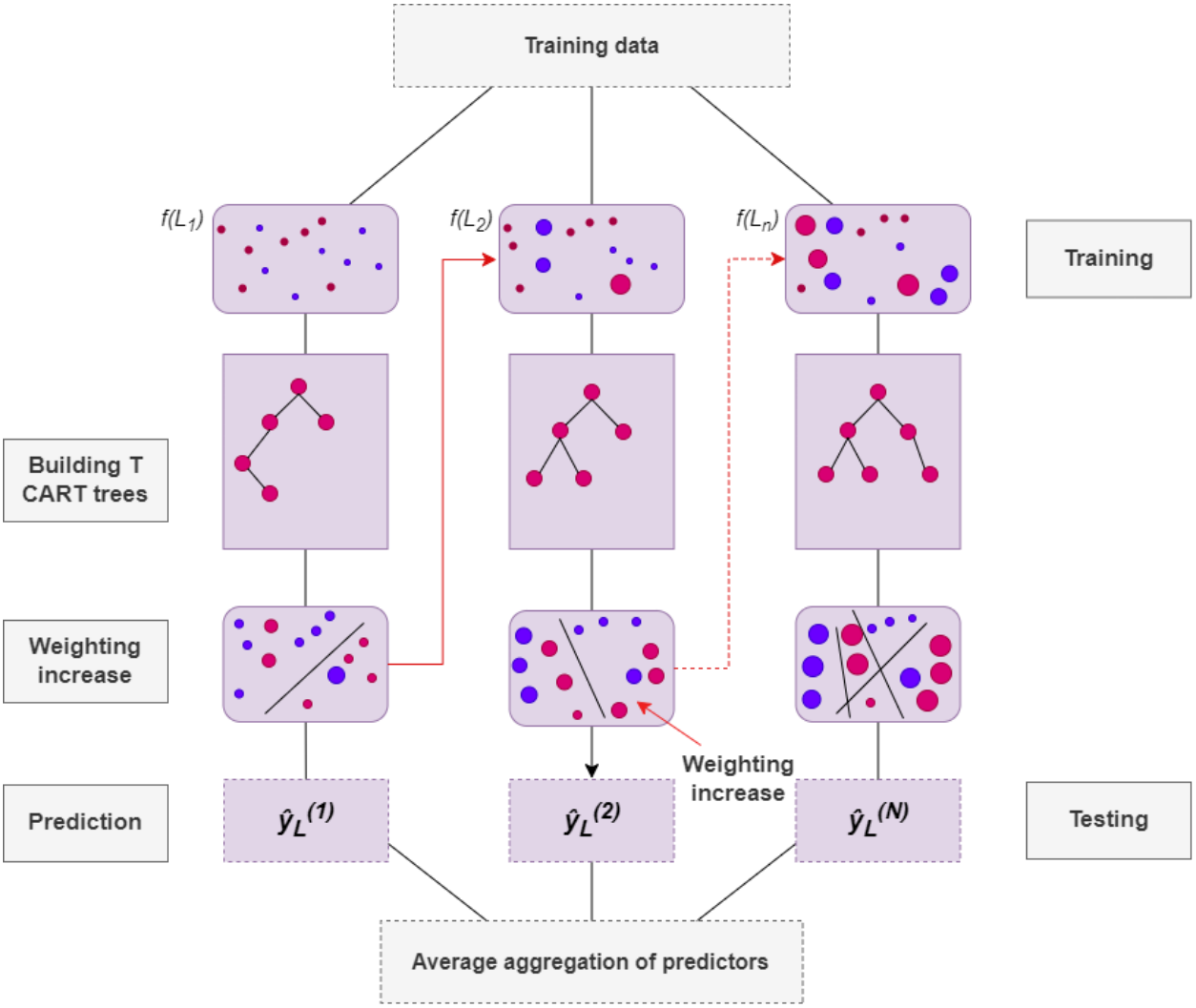


Figure 38: Proposed XGBoost-based model for Intrusion Detection in a Heterogeneous IoT environment

In this framework as shown in figure 4, the ensemble is constructed by aggregating the predictions of multiple decision trees, where each tree is trained to correct the errors made by its predecessors. The overall prediction $\hat{y}_i$ for an instance i is computed as:

$$\hat{y}_i = \sum_{k=1}^{K} f_k(x_i)$$

Where x_i represents the feature vector of the instance, f_k denotes the k is decision tree, and K is the total number of trees in the ensemble.

The objective function used in XGBoost combines a loss function and a regularization term to guide the training process. The loss function quantifies the discrepancy between the predicted values and the actual labels, while the regularization term discourages overly complex models. The objective function L is given by:

$$L(\theta) = \sum_{i=1}^{n} l(\hat{y}_i, y_i) + \sum_{k=1}^{K} \vartheta(f_k)$$

Where $l(\hat{y}_i, y_i)$is the loss function, for instance i is the regularization term for the k^{th} tree, and n is the number of training instances. The regularization term is defined as:

$$\vartheta(f) = \gamma T + \frac{1}{2}\lambda \sum_{j=1}^{T} w_j^2$$

Here, T represents the number of leaves in the tree, w is the weight of the j^{th} leaf, γ controls the tree's complexity and λ regulates the regularization of leaf weights.

During training, XGBoost minimizes the objective function $L(\theta)$ iteratively. Each new decision tree (f_t) is added to the ensemble to minimize the objective function:

$$L^{(t)} = \sum_{i=1}^{n} l\left(y_i, \widehat{y_i^{(t-1)}} + f_t(x_i)\right) + \vartheta(f_t)$$

where $y_i^{(t-1)}$ denotes the prediction from the previous $(t-1)$ trees. The optimal decision tree f_t is found by solving:

$$f_t = \arg min_f \sum_{i=1}^{n} [\, g_i f(x_i) + \frac{1}{2} h_i f^2(x_i)] + \vartheta(f)$$

Where, $g_i = \frac{\partial\, l(\widehat{y_i,}\, y_i)}{\partial \widehat{y_i}}$ is the first-order gradient and $h_i = \frac{\partial^2\, l(\widehat{y_i,}\, y_i)}{\partial \widehat{y_i^2}}$ is the second-order gradient.

In the XGBoost framework, the weak learners are decision trees, simple structures with shallow depths. These trees are capable of capturing complex patterns when combined in an ensemble. The splitting criteria for these decision trees are based on minimizing impurity, which is mathematically represented as:

$$Gain\,(S, a) = Impurity\,(S) - \sum_{v \epsilon V} \frac{|S_v|}{|S|}\, Impurity\,(S_v)$$

where S is the set of samples before the split, a is the feature used for the split, V is the set of all possible values for the feature a, and $Impurity$ measures the disorder of the set.

The proposed XGBoost-based model effectively combines the predictions of individual decision trees to create a strong classifier. By iteratively adding trees and optimizing the objective function, the model enhances its ability to detect cyber threats, demonstrating robustness in heterogeneous IoT networks.

5. Experimental Setup and Result Analysis

This section outlines the experimental setup, describes the benchmark dataset, details the evaluation metrics, and analyzes the results of the proposed blockchain-based framework.

5.1. Experimental Setup

The experiments were conducted on a laptop equipped with the following hardware specifications: an Intel Core i5 10th Gen processor, 8GB of RAM, 512GB of ROM, and an NVIDIA GTX 1650 GDDR6 4GB graphics card. The operating system used was Windows 11. This setup balanced performance and resource constraints, with the graphics card enhancing computational efficiency. Various data analysis frameworks were utilized to manage and analyze the data, including Pandas, NumPy, Seaborn, Matplotlib, and Scikit-learn. These tools facilitated data preprocessing, visualization, and the application of machine learning algorithms. The dataset was divided into three subsets: 80% for training, 20% for testing, and 20% for validation. This split ensured a robust evaluation of the model's performance, allowing for a thorough assessment of training efficacy and generalization capabilities. The Hyperledger Fabric framework was employed to integrate blockchain technology. Hyperledger Fabric provided a secure and decentralized platform for managing data during the training and testing phases. This implementation ensured that data integrity and

security were maintained throughout the experimental process, aligning with the study's focus on enhancing security and privacy.

5.2. Dataset Description

The proposed model uses the Edge-IIoT dataset for intrusion detection in Internet of Things (IoT) and Industrial Internet of Things (IIoT) environments. The dataset encompasses data collected from over ten IoT devices, including low-cost digital sensors for temperature and humidity sensing, pH meters, ultrasonic sensors, heart rate monitors, water-level detectors, soil moisture sensors, and flame sensors. This dataset features an extensive analysis of 14 distinct attack types related to IoT and IIoT protocols, which are categorized into five major threat groups: Denial of Service (DoS) and Distributed Denial of Service (DDoS) attacks, information gathering, injection attacks, man-in-the-middle attacks, and malware attacks. Of the 1,176 available features, 61 demonstrate high correlation. The dataset comprises 20,952,648 instances, with 11,223,940 labeled as normal and 9,728,708 as attacks. The data were split into training and testing subsets, with 80% allocated for training and 20% for testing, using stratification to maintain proportional representation across all classes. Specifically, 1,909,671 samples were selected, with 1,527,736 assigned to the training set and 381,935 to the test set. These samples are distributed across 15 categories, as detailed in Figure 5.

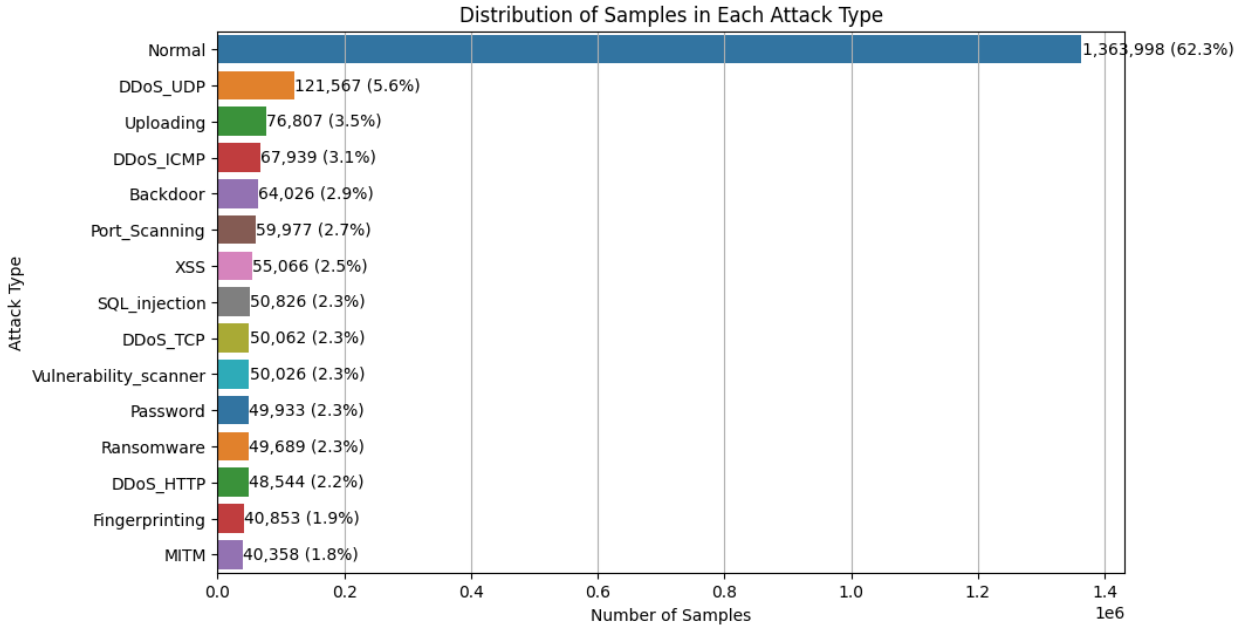


Figure 39: Statistical representation of samples in each attack class in the Edge_IIoT dataset.

5.3. Performance Evaluation Metrics

This section comprehensively details the performance evaluation metrics employed to assess the effectiveness and efficiency of the proposed blockchain-based framework and intrusion detection model by systematically analyzing critical metrics in both phases: Phase 1, which focuses on the blockchain framework, and Phase 2, which evaluates the intrusion detection system. This section aims to provide a rigorous examination of the system's capabilities.

5.3.1. Performance Evaluation Metrics for Blockchain-based Framework (Phase 1)

When evaluating the performance and reliability of the blockchain framework, it is important to consider a comprehensive set of metrics that offer insights into the system's efficiency, security, and user experience. These metrics are instrumental in assessing blockchain performance under varying conditions. This sub-section

systematically defines key evaluation parameters, including fault tolerance, transaction finality, and network overhead. By examining these metrics, the strengths and limitations of blockchain Framework across different parameters can be rigorously analyzed, thereby facilitating the development of robust and scalable solutions. The following are the Evaluation Metrics:

- **Fault Tolerance:** Fault tolerance in blockchain refers to the network's capability to continue functioning correctly even when some components fail. This is critical for maintaining system reliability and ensuring the blockchain remains operational despite disruptions. It is measured by Restoration Efficiency (RE), which quantifies how effectively the system recovers from failures:
$$RE = \frac{T_{restored}}{T_{total}}$$
- **User Experience (UX):** User experience in blockchain systems reflects the ease and efficiency with which users interact, focusing on the system's response time and the time required to share records. It is inversely related to the sum of response time ($T_{response}$) and shared record time (T_{share}):
$$UX = \frac{1}{T_{response} + T_{share}}$$
- **Transaction Finality:** Transaction finality measures how quickly a transaction becomes irreversible and permanently recorded on the blockchain. It is directly related to the time required to create a block T_{block}:
$$TF = T_{block}$$
- **Network Overhead (NO):** Network overhead refers to the additional computational resources and time consumed due to managing blockchain transactions, including encryption processes. It is often expressed as a percentage of the system's throughput and is influenced by the encryption time $T_{encrypt}$:
$$NO\% = \frac{T_{encrypt}}{Th} * 100$$
- **Encryption Time:** Encryption time ($T_{encrypt}$) is the duration required to convert plaintext into encrypted data using cryptographic algorithms, impacting security and transmission efficiency:
$$T_{encrypt} = \frac{1}{f}\sum_{i-1}^{f} t_{encrypt,i}$$
- **Decryption Time:** Decryption time ($T_{decrypt}$) is the time taken to convert encrypted data back to its original form, crucial for accessing secured data efficiently:
$$T_{decrypt} = \frac{1}{f}\sum_{i-1}^{f} t_{decrypt,i}$$
- **Key Generation Time:** Key generation time ($T_{key\ Gen\ T}$) is the duration required to create cryptographic keys, affecting overall security and system speed:
$$T_{key\ Gen\ T} = \frac{1}{n}\sum_{i-1}^{n} t_{key\ Gen\ T,i}$$
- **Response Time:** Response time ($T_{response}$) measures the interval between a user request and the system's response, crucial for performance evaluation in time-sensitive applications:
$$T_{response} = T_{response\_end} - T_{request\_start}$$
- **Restoration Efficiency:** Restoration efficiency (RE) quantifies the system's ability to recover from faults, defined as:
$$E_{restoration} = \frac{D_{rest}}{D_{orig}} \times 100\%$$

Where $E_{restoration}$ is the restoration efficiency, D_{rest} is the amount of data successfully restored, and D_{orig} is the original data before any loss or corruption.

- **Shared Record Time:** Shared record time (T_{share}) is the time taken to transmit or share a record within the system, impacting the efficiency of data sharing:
$$T_{share} = T_{send} + T_{verify} + T_{commit}$$

Where T_{share} is the sharing record time, T_{send} is the time to send the record, T_{verify} is the time to verify the record, T_{commit} is the time to commit the record to the blockchain.

- **Block Creation Time:** Block creation time (T_{block}) is the time required to generate a new block in the blockchain, affecting transaction finality and throughput:

$$T_{block} = \frac{1}{N}\sum_{i=1}^{N} t_{block,i}$$

Where T_{block} is the average block creation time, $t_{b,i}$ is the time taken to create the $i-th$ block, and N is the total number of blocks created.

- **Throughput:** Throughput (Th) is the number of transactions processed per second, a key metric for evaluating the scalability and efficiency of blockchain networks:

$$Th = \frac{Total\ transaction}{total\ time\ taken}$$

- **Latency:** Latency (L) is the delay between the initiation and completion of a transaction, crucial for real-time processing:

$$L = T_{completion} - T_{initiation}$$

5.3.2. Performance Evaluation Metrics for Intrusion Detection Model (Phase 2)

This subsection presents the performance evaluation metrics for assessing the robustness of the Proposed Intrusion Detection System (IDS), as shown in Table 6.

Table 17: Performance Evaluation Metrics for Intrusion Detection System

Metrics	Definition	Formula
Confusion matrix	Used to evaluate the performance of a classification model by comparing the predicted labels with the actual labels.	
True positive (TP)	The record is successfully detected as malicious	
False positive (FP)	The record is wrongly detected as malicious.	
True Negative (TN)	The record is classified as non-malicious.	
False Negative (FN)	The record is undetected by the system.	
Accuracy	Measure how well the model predicts the correct labels.	$Accuracy = \frac{TP+TN}{TP+FP+TN+FN}$
Precision	Measure how many of the predicted positive labels are actually positive.	$Precision = \frac{TP}{TP+FP}$
Recall	Measure how many of the actual positive labels are correctly predicted.	$Recall = \frac{TP}{TP+FN}$
F1 score	The harmonic mean of Recall and Precision.	$F1\ Score = 2 * \frac{Precision * Recall}{Precision + Recall}$
ROC curve	Graphical representation of the performance of a classification model. TPR is the ratio of true positive predictions to the total actual positive labels. FPR is the ratio of false positive predictions to the total actual negative labels.	$TPR = \frac{TP}{TP+FN}$, $FPR = \frac{FP}{FP+FN}$
True Negative Rate (TNR)	Model's ability to correctly classify instances of a specific attack type as non-attacks.	$TNR = \frac{TN}{TN+FP}$
Negative Predictive Value (NPV)	Model's accuracy in predicting non-attacks for a specific attack type.	$NPV = \frac{TN}{TN+FN}$
False Positive Rate (FPR)	The rate at which instances of other attack types are incorrectly classified as the specific attack type.	$FPR = \frac{FP}{FP+TN}$
False Negative Rate (FNR)	The rate at which instances of a specific attack type are incorrectly classified as non-attacks or other types.	$FNR = \frac{FN}{TP+FN}$
False Discovery Rate (FDR)	The rate at which instances are falsely predicted as the specific attack type when they are not.	$FDR = \frac{FP}{FP+TN}$
False Omission Rate (FOR)	The rate at which instances of a specific attack type are falsely classified as non-attacks or other types	$FOR = \frac{FN}{FN+TN}$

5.4. Result Analysis

This section provides an in-depth analysis of the performance metrics for the proposed Hybrid Blockchain-Based Framework and the GA-Optimized Intrusion Detection Model. The evaluation covers critical aspects such as encryption and decryption efficiency, block creation time, throughput, and response times compared to existing cryptographic algorithms and intrusion detection models. The results demonstrate significant improvements in security, efficiency, and scalability, highlighting the effectiveness of the proposed methodologies in addressing the challenges of secure communication and intrusion detection in IoT environments.

5.4.1. Result Analysis for Blockchain-based Framework (Phase 1)

In this phase, various cryptographic methods were compared based on multiple performance metrics to evaluate the efficiency and security of the proposed SADE-ECC-DSA-SHA-512 framework. Table 7 presents a comparison of various cryptographic methods, including Elliptic Curve Cryptography (ECC), Advanced Encryption Standard (AES), Digital Signature Algorithm (DSA), and a proposed approach (SADE-ECC-DSA-SHA-512). The metrics analyzed include encryption time, decryption time, block creation time, key generation time, throughput, latency, response time, restoration efficiency, and successful sharing record time. These metrics are critical for assessing cryptographic techniques performance, efficiency, and security in secure communication systems. The SADE-ECC-DSA-SHA-512 framework exhibits exceptional performance across multiple metrics, significantly outperforming traditional cryptographic techniques such as ECC, AES, DSA, and SADE-ECC. Its superior efficiency stems from the seamless integration of advanced cryptographic methods and dynamic optimization mechanisms. The incorporation of Elliptic Curve Cryptography (ECC) ensures lightweight and secure operations, while SHA-512 accelerates hashing processes with high reliability. The Self-Adaptive Differential Evolution (SADE) algorithm optimizes cryptographic parameters dynamically, eliminating unnecessary overhead. These innovations result in faster encryption, decryption, block creation, and key generation times. Furthermore, using the Practical Byzantine Fault Tolerance (PBFT) consensus mechanism minimizes delays during consensus, enhancing throughput and reducing latency. These optimizations collectively allow the proposed framework to process transactions more efficiently than conventional methods, making it ideal for resource-constrained environments.

The framework's architectural enhancements extend to other critical metrics such as response time, restoration efficiency, and successful sharing record time. The proposed methodology uses IPFS for decentralized off-chain storage to ensure rapid data retrieval and transaction processing, reducing response times. Restoration efficiency benefits from the tamper-proof properties of blockchain and the fast hash generation of SHA-512, ensuring secure and seamless data recovery. Additionally, the distributed nature of the blockchain layer, combined with the optimized cryptographic operations of SADE-ECC-DSA-SHA-512, enables faster and more reliable sharing of records. These combined advancements underscore the framework's holistic approach to optimizing blockchain performance, positioning it as a robust and efficient solution for secure communication systems in high-security environments like IoT.

Table 18: Optimizing Blockchain Security: Performance Metrics Evaluation of ECC, AES, DSA, SADE-ECC, and an Integrated SADE-ECC-DSA-SHA-512 Proposed Methodology

Performance Metrics	ECC	AES	DSA	SADE-ECC	Proposed Methodology (SADE-ECC-DSA-SHA-512)
Encryption time(seconds)	0.180445	0.192674	0.200632	0.152304	**0.113211**
Decryption time(seconds)	0.169943	0.180322	0.190216	0.131107	**0.097680**
Block creation time(seconds)	0.254789	0.214789	0.284236	0.187489	**0.142591**
Key generation time(seconds)	0.350388	0.372996	0.390848	0.283411	**0.210891**
Throughput	54	44	37	59	**67**
Latency	0.554	0.482	0.589	0.394	**0.342**
Response time(seconds)	230.654	216.134	238.933	201.183	**188.786**

Restoration efficiency(seconds)	0.828343	0.762332	0.852971	0.773512	**0.885448**
Successful sharing record time(seconds)	21.8835	18.0304	23.8538	15.6789	**13.7679**

The performance evaluation of various cryptographic methods, including Elliptic Curve Cryptography (ECC), Advanced Encryption Standard (AES), Digital Signature Algorithm (DSA), and the proposed SADE-integrated approach (SADE-ECC-DSA-SHA-512), highlights significant improvements in encryption and decryption efficiency. The proposed methodology demonstrates the lowest encryption time of 0.113211 seconds and decryption time of 0.097680 seconds, outperforming all other methods tested. This efficiency is crucial for enhancing overall system performance and ensuring timely access to data as shown in Figure 6.

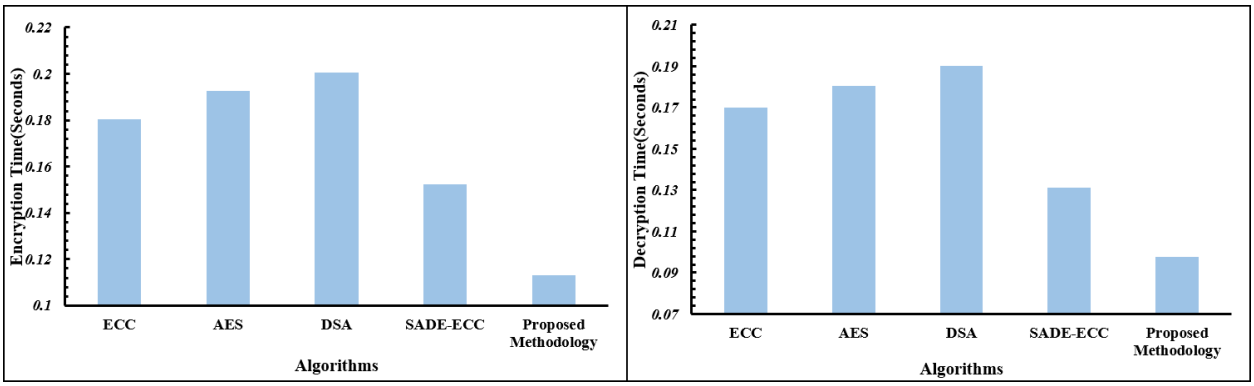


Figure 40: Comparative Visualization of Encryption and Decryption Times Across Cryptographic Algorithms

In terms of block creation, the proposed approach achieves a time of 0.142591 seconds, which is quicker than ECC (0.254789 seconds), AES (0.214789 seconds), DSA (0.284236 seconds), and SADE-ECC (0.187489 seconds). This faster block creation time supports more efficient blockchain operations, vital for maintaining a robust and responsive blockchain network. Additionally, the key generation time for the proposed method is 0.210891 seconds, showing notable improvements over ECC (0.350388 seconds), AES (0.372996 seconds), and DSA (0.390848 seconds), highlighting the effectiveness of the SADE algorithm in producing high-quality cryptographic keys as shown in Figure 7.

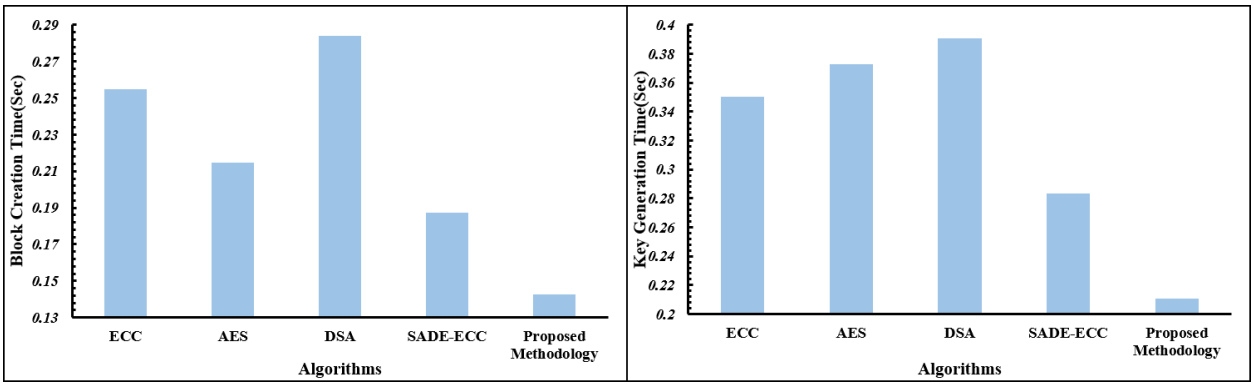


Figure 41: Comparative Visualization of Key Generation and Block Creation Efficiencies Across Cryptographic Algorithms

The proposed methodology also leads in throughput, achieving 67 units per time unit compared to ECC (54), AES (44), DSA (37), and SADE-ECC (59). Figure 8 indicates that the hybrid approach handles data operations more efficiently. Latency is another critical metric where the proposed method excels with a value of 0.342 seconds, lower

than ECC (0.554 seconds), AES (0.482 seconds), DSA (0.589 seconds), and SADE-ECC (0.394 seconds). Reduced latency enhances real-time data processing, essential for effective blockchain and data management.

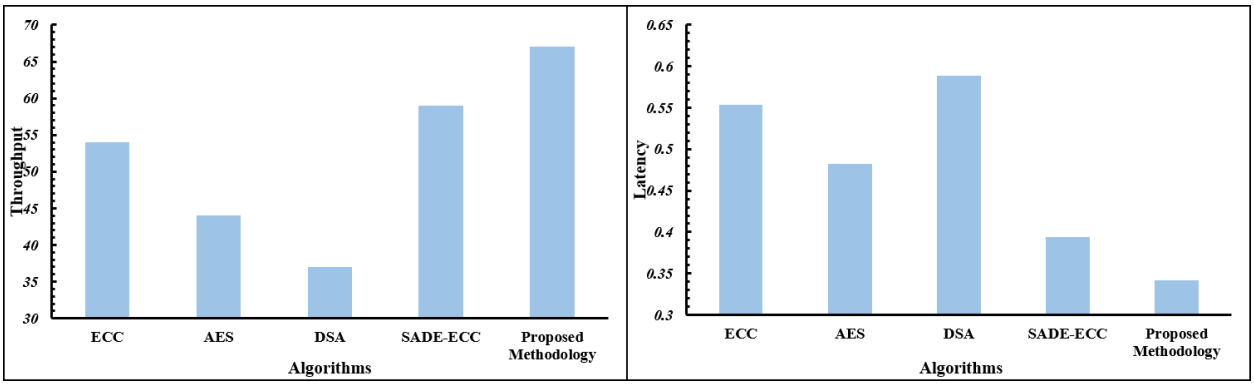


Figure 42: Comparative Visualization of Throughput and Latency Across Cryptographic Algorithms

Response time, which reflects the time taken to respond to data requests, is the shortest for the proposed approach at 188.786 milliseconds, outperforming ECC (230.654 seconds), AES (216.134 seconds), DSA (238.933 seconds), and SADE-ECC (201.183 seconds). Figure 9 underscores the efficiency of the hybrid approach in providing quick feedback and ensuring responsive system behavior. Restoration efficiency, a measure of data integrity maintenance, is highest, with the proposed methodology at 0.885448, indicating effective data management. Finally, the successful sharing record time of 13.7679 seconds with the proposed method is the fastest among all tested methods, demonstrating the efficiency of the hybrid approach in secure and timely data sharing.

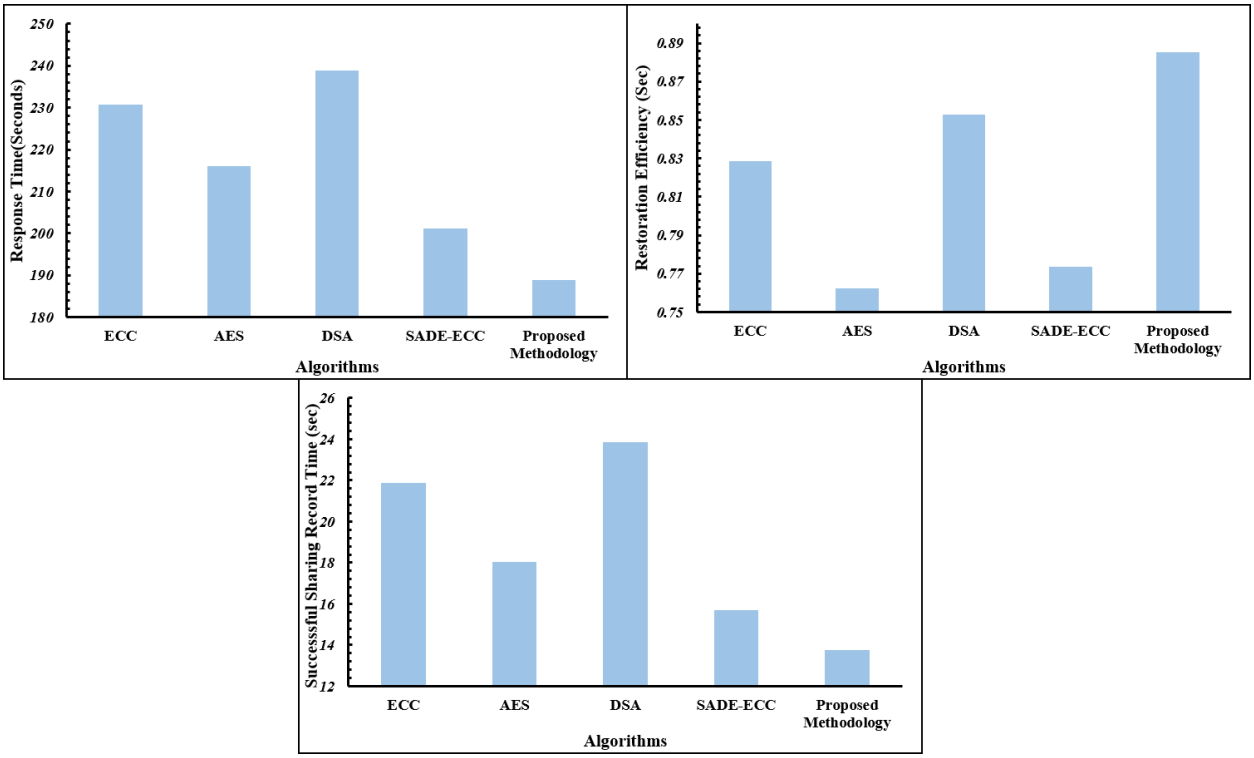


Figure 43: Comparative Visualization of Response Time, Restoration Efficiency, and Successful Sharing Record Time Across Cryptographic Algorithms

Table 8 compares the throughput and estimated number of IoT devices supported by various cryptographic models, including ECC, AES, DSA, SADE-ECC, and the proposed algorithm. The proposed algorithm exhibits the highest

throughput at 67 units per time unit, supporting approximately 10,050 IoT devices. This demonstrates its superior efficiency and scalability in handling large IoT networks compared to other models, which support fewer devices and have lower throughput.

Table 19: Comparative Analysis of Cryptographic Throughput for IoT Device Support, Optimizing Cryptographic Throughput for Large-Scale IoT Deployments

Model	Throughput	Estimated IoT devices
ECC	54	8100
AES	44	6600
DSA	37	5550
SADE-ECC	59	8850
Proposed Algorithm	**67**	**10050**

Table 9 presents the encryption time and security levels of different cryptographic models, including ECC, AES, DSA, SADE-ECC, and the proposed algorithm. The security level is quantified based on the cryptographic algorithm's resilience to attacks, determined by key size, algorithmic complexity, and resistance to common vulnerabilities such as brute force or side-channel attacks. The encryption time is measured in seconds during the execution of encryption tasks on a standard computational platform, providing a direct comparison of the speed of each model. The proposed algorithm achieves the shortest encryption time (0.113211 seconds) due to its dynamic optimization via SADE and efficient cryptographic integration, such as ECC and SHA-512. The "Very High" security rating is assigned based on its multi-layered security features, including robust key generation, high-entropy outputs, and tamper resistance provided by blockchain integration. Comparatively, models like DSA, which lack advanced optimizations, exhibit longer encryption times and lower security ratings.

Table 20: Evaluating the Relationship Between Encryption Speed and Security Strength, Comparative Analysis of Encryption Time and Security Level

Model	Encryption time	Security Level
ECC	0.180445	Medium
AES	0.192674	Medium
DSA	0.200632	Low
SADE-ECC	0.152304	High
Proposed Algorithm	**0.113211**	**Very High**

Table 10 compares cryptographic models ECC, AES, DSA, SADE-ECC, and the proposed algorithm based on encryption time, block creation time, and estimated energy consumption. The proposed algorithm demonstrates superior performance with the shortest encryption time (0.113211 seconds) and block creation time (0.142591 seconds), along with the lowest energy consumption (0.000255802 kWh). These results highlight the proposed algorithm's efficiency in computational speed and energy usage, making it optimal for resource-constrained and energy-sensitive environments such as IoT networks. The significant reduction in energy consumption compared to other models like ECC (0.000435234 kWh), AES (0.000407463 kWh), DSA (0.000484868 kWh), and SADE-ECC (0.000339793 kWh) underscores the proposed algorithm's suitability for applications where minimizing power usage is crucial.

Table 21: Evaluating the Energy Efficiency of Different Cryptographic Algorithms

Model	Encryption time	Block Creation Time	Estimated Energy Consumption (kWh)
ECC	0.180445	0.254789	0.000435234
AES	0.192674	0.214789	0.000407463
DSA	0.200632	0.284236	0.000484868
SADE-ECC	0.152304	0.187489	0.000339793

Proposed Algorithm	**0.113211**	**0.142591**	**0.000255802**

Table 11 compares cryptographic models ECC, AES, DSA, SADE-ECC, and the proposed algorithm based on response time, restoration efficiency, and fault tolerance. The proposed algorithm exhibits the highest restoration efficiency (0.885448) and fault tolerance (0.880894), indicating robust data recovery and failure resilience. Despite having a longer response time (1888.786 seconds), its superior fault tolerance and restoration capabilities make it a reliable choice for critical applications requiring high data integrity and reliability.

Table 22: Comparative Analysis of Cryptographic Techniques for Response Time, Restoration, and Fault Tolerance

Model	Response Time	Restoration Efficiency	Fault Tolerance
ECC	230.654	0.828343	0.810296
AES	216.134	0.762332	0.748366
DSA	238.933	0.852971	0.850718
SADE-ECC	201.183	0.773512	0.761928
Proposed Algorithm	**1888.786**	**0.885448**	**0.880894**

Table 12 assesses cryptographic models ECC, AES, DSA, SADE-ECC, and the proposed algorithm based on sharing record time and interoperability. Interoperability is evaluated by the ability of cryptographic models to integrate seamlessly across various systems and protocols in heterogeneous environments. This is assessed using a combination of system integration tests and cross-platform compatibility evaluations, considering parameters such as the ease of key sharing, compatibility with different architectures, and support for varying communication protocols. The proposed algorithm's "Very High" interoperability is justified by its streamlined sharing record time (13.7679 seconds) and robust blockchain framework employing IPFS for decentralized data storage. This architecture supports smooth data exchange and system integration across diverse platforms. In contrast, DSA's "Low" interoperability reflects challenges with slower record sharing and limited adaptability in complex networks. In contrast, ECC and AES achieve medium ratings, showcasing moderate flexibility but reduced performance in demanding scenarios. The comparative analysis underscores the proposed model's superior efficiency and compatibility, making it ideal for real-time and multi-platform applications.

Table 23: Comparative Analysis of Record Sharing and Interoperability in Cryptography

Model	Sharing Record Time	Interoperability
ECC	21.8835	Medium
AES	18.0304	Medium
DSA	23.8538	Low
SADE-ECC	15.6789	High
Proposed Algorithm	**13.7679**	**Very High**

Table 13 evaluates cryptographic models ECC, AES, DSA, SADE-ECC, and the proposed algorithm based on response time, sharing record time, and user experience. The proposed algorithm exhibits the best user experience, attributed to its "Very High" rating, with the fastest sharing record time (13.7679 seconds) and response time (188.786 milliseconds). This ensures swift and efficient data handling, enhancing user satisfaction, especially in time-sensitive applications. In contrast, DSA has the lowest user experience rating, with the slowest sharing record time (23.8538 seconds) and longest response time (238.933 milliseconds), indicating potential delays and less efficient interactions. ECC and AES offer a "Medium" user experience, balancing moderate performance and usability. The "High" rating for SADE-ECC highlights its effective combination of fast processing and user-friendly features.

Table 24: Comparative Analysis of Cryptographic Techniques on Response Time, Sharing, and User Experience

Model	Response Time	Sharing Record Time	User Experience
ECC	230.654	21.8835	Medium

AES	216.134	18.0304	Medium
DSA	238.933	23.8538	Low
SADE-ECC	201.183	15.6789	High
Proposed Algorithm	**188.786**	**13.7679**	**Very High**

Table 14 compares cryptographic models ECC, AES, DSA, SADE-ECC, and the proposed algorithm on block creation and transaction finality time. The proposed algorithm excels with the shortest times for block creation (0.142591 seconds) and transaction finality (0.142591 seconds), ensuring rapid transaction processing and confirmation. This efficiency significantly enhances the speed and reliability of blockchain operations, making it ideal for high-demand environments. In contrast, DSA exhibits the longest times (0.284236 seconds), potentially causing delays in transaction processing and finality. ECC and AES offer moderate performance, while SADE-ECC demonstrates improved efficiency with relatively low times.

Table 25: Comparative Analysis of Block Creation and Transaction Finality Times

Model	Block Creation Time	Transaction Finality Time
ECC	0.254789	0.254789
AES	0.214789	0.214789
DSA	0.284236	0.284236
SADE-ECC	0.187489	0.187489
Proposed Algorithm	**0.142591**	**0.142591**

Table 15 evaluates cryptographic models ECC, AES, DSA, SADE-ECC, and the proposed algorithm based on throughput, encryption time, and network overhead. The proposed algorithm performs better with the highest throughput (67) and the shortest encryption time (0.113211 seconds). It also exhibits the lowest network overhead at 1.69%, indicating minimal additional data load on the network. In contrast, DSA has the highest network overhead (5.42%), suggesting significant extra data load. ECC and AES offer moderate overheads, while SADE-ECC demonstrates improved efficiency with lower overhead (2.58%).

Table 26: Cryptographic Overhead and Its Implications for Network Performance

Model	Throughput	Encryption time	Network Overhead (%)
ECC	54	0.180445	3.34
AES	44	0.192674	4.38
DSA	37	0.200632	5.42
SADE-ECC	59	0.152304	2.58
Proposed Algorithm	**67**	**0.113211**	**1.69**

Table 16 expands the analysis by considering file sizes ranging from 1 MB to 300 MB under varying node counts and network conditions. For smaller files (1–50 MB), the latency remains below 1000 ms with relatively high throughput (350–450 TPS) and low storage overhead (3–5%). As the file size increases, latency grows significantly due to increased metadata and chunking demands, peaking at 2500 ms for a 300 MB file in a constrained 5 Mbps network. Storage overhead follows a linear trend, reaching 10% for the largest file size. This analysis illustrates the scalability of IPFS and resilience despite increased file sizes.

Table 27: IPFS Overhead Analysis

File Size (MB)	Node Count	Network Bandwidth	Latency (ms)	Throughput (TPS)	Storage Overhead (%)
1	10	100 mbps	200	450	3%
10	21	80 mbps	500	400	4.1%

50	30	50 mbps	1000	350	5.3%
100	49	30 mbps	1500	300	6%
150	72	20 mbps	1800	270	6.9%
200	81	15 mbps	2100	240	8%
250	97	10 mbps	2300	200	9.3%
300	118	5 mbps	2500	180	11%

Table 17 evaluates system performance across various transaction rates (100–1000 TPS). At lower transaction rates (100–300 TPS), consensus and detection latencies remain stable (800–1200 ms), with system throughput nearing the transaction rate. At higher transaction rates (350–1000 TPS), the consensus latency increases to 2000 ms due to the computational burden, though the system maintains high throughput (930 TPS for 1000 TPS input). This demonstrates the proposed framework's ability to handle high network loads while maintaining reliability and efficiency.

Table 28: Scalability Analysis Under Different Network Loads

Transaction Rate (TPS)	Blockchain Nodes	Consensus Latency (ms)	Detection Latency (ms)	System Throughput (TPS)
100	10	800	500	95
150	20	900	510	145
200	30	1000	515	195
250	40	1100	520	245
300	50	1200	520	290
350	60	1300	530	340
400	70	1500	535	380
450	80	1700	540	420
500	90	1800	540	470
1000	160	2000	540	930

In table 18, Energy consumption is a critical factor for cryptographic algorithms, especially in resource-constrained IoT environments. The comparison of energy consumption across various methods shows that the proposed methodology (SADE-ECC-DSA-SHA-512) achieves the lowest energy consumption at 8.1 Joules, marking a 35% reduction compared to DSA (14.2 Joules) and 23% compared to ECC (12.5 Joules). This efficiency is primarily due to the dynamic optimizations introduced by the SADE algorithm, which reduces redundant computations and ensures streamlined cryptographic operations. Moreover, the proposed methodology integrates SHA-512, efficiently handling hashing processes while minimizing energy requirements. Such energy-efficient cryptographic solutions are particularly beneficial for IoT devices, where limited power resources often constrain performance. By significantly lowering energy consumption, the proposed methodology extends the operational lifespan of IoT devices and enhances sustainability in large-scale deployments.

The proposed methodology demonstrates superior performance in key generation time, an essential metric for cryptographic algorithms in real-time IoT applications. The SADE-ECC method generates keys in 0.283 seconds, representing a 24% improvement over ECC (0.350 seconds) and a 28% improvement over DSA (0.390 seconds). The proposed methodology further enhances this by reducing the time to 0.210 seconds, a 40% improvement compared to ECC. This reduction is achieved through the Self-Adaptive Differential Evolution (SADE) algorithm, which dynamically adjusts parameters to optimize key generation processes. The high entropy and robust security SADE provides ensure the generated keys are secure while minimizing computational overhead. This adaptability makes the proposed methodology highly suitable for large-scale, heterogeneous IoT environments where rapid and secure key generation is critical.

The combination of reduced energy consumption and faster key generation time positions the proposed methodology as an efficient and practical solution for IoT security challenges. The reduction in energy consumption directly contributes to the feasibility of deploying cryptographic algorithms in IoT devices with limited computational and

power resources. Similarly, the faster key generation time enhances the responsiveness of security protocols, ensuring that IoT networks can handle high volumes of transactions without performance degradation.

Table 29: Energy Consumption and Key Generation Time Comparison Cryptographic Algorithms

Cryptographic Algorithm	Energy Consumption (joules)	Key Generation Time (seconds)
ECC	12.5	0.350388
AES	13.7	0.372996
DSA	14.2	0.390848
SADE-ECC	10.3	0.283411
Proposed Algorithm	**8.1**	**0.210891**

To evaluate the efficiency of the proposed SADE-ECC-DSA-SHA-512 framework, we conducted a detailed comparative analysis of overhead metrics, including computational, network, and storage overhead, against traditional cryptographic methods such as ECC, AES, DSA, and SADE-ECC. The analysis, presented in Table 19, highlights key performance indicators such as CPU usage, memory usage, bandwidth consumption, and additional storage requirements.

The computational overhead of a cryptographic method is a critical factor, especially in resource-constrained IoT environments. CPU usage and memory usage are primary indicators of computational overhead. As shown in Table 12, the proposed SADE-ECC-DSA-SHA-512 framework exhibits the lowest computational overhead, with CPU usage at 20% and memory usage at 40 MB. This is significantly lower than traditional methods such as DSA, which requires 40% CPU usage and 80 MB memory, and AES, which demands 35% CPU usage and 70 MB memory. The reduced computational load in the proposed framework is attributed to the Self-Adaptive Differential Evolution (SADE) algorithm, which optimizes cryptographic parameter selection dynamically, minimizing redundant computations and enhancing overall efficiency.

In terms of network overhead, the proposed methodology achieves low bandwidth usage (8 MB/s) compared to existing methods like DSA (18 MB/s) and AES (15 MB/s). This reduction is facilitated by integrating IPFS for decentralized off-chain storage, which efficiently manages data distribution and minimizes the volume of data transmitted over the network. Consequently, the network overhead is significantly lowered, enhancing the framework's suitability for IoT environments where bandwidth may be limited or costly. Storage overhead is another crucial aspect, particularly for devices with constrained storage capacities. The proposed SADE-ECC-DSA-SHA-512 framework maintains a low storage overhead (5 MB), comparable to ECC (5 MB) and significantly lower than DSA (15 MB). This efficiency is achieved through the optimized use of SHA-512 for hashing and streamlined cryptographic operations that reduce the need for additional storage for metadata and cryptographic keys.

The proposed framework demonstrates a balanced and optimized approach to minimizing overhead across computational, network, and storage dimensions. This optimization enhances the performance and scalability of Intrusion Detection Systems (IDS) in IoT networks and ensures that the framework remains practical and sustainable in real-world deployments.

Table 30: Comparative Overhead Analysis of Cryptographic Methods

Overhead Metrics	ECC	AES	DSA	SADE-ECC	Proposed Methodology (SADE-ECC-DSA-SHA-512)
Computational Overhead	Moderate	High	High	Moderate	**Low**
CPU usage(%)	25	35	40	30	**20**
Memory usage (MB)	50	70	80	60	**40**
Bandwidth usage (MB/s)	10	15	18	12	**8**
Network Overhead	3.34	4.38	5.42	2.58	**1.69**
Storage Overhead	Low	Medium	High	Medium	**Low**

Additional Storage (MB)	5	10	15	10	**6**

The Table 20 highlights a comparative analysis between Traditional Differential Evolution (DE) and Self-Adaptive Differential Evolution (SADE) for cryptographic key generation. SADE significantly outperforms Traditional DE across three key metrics. It reduces key generation time from 2.1 ms to 1.5 ms, a 28.57% improvement, ensuring faster operations critical for IoT environments. SADE also enhances key entropy from 0.92 to 0.98, increasing the randomness of keys and strengthening resistance to cryptanalysis by 6.5%. Furthermore, while DE offers only moderate brute-force resistance, SADE achieves high resistance, making it highly secure for cryptographic applications. These improvements underscore SADE's suitability for resource-constrained, high-security systems like blockchain-based frameworks in IoT.

Table 31: Comparative Analysis of Key Generation Performance Between Traditional DE and SADE

Metrics	Traditional DE	SADE	Improvement
Key Generation Time	2.1 ms	1.5 ms	28.57%
Key Entropy	0.92	0.98	+ 6.5%
Brute-Force Resistance	Moderate	High	Significant

The comparison of consensus mechanisms highlights the superior performance of the proposed PBFT-based hybrid system over traditional PoW and PoS, as shown in Table 21. In terms of average gas computation, the proposed system achieves the lowest cost (28,500 Gwei), outperforming PoW (45,000 Gwei) and PoS (38,000 Gwei), thus optimizing resource utilization. Similarly, the average transaction fee is significantly reduced to 0.000022 ETH, making the proposed framework the most cost-effective option. With a processing speed of 35 transactions per second, the system demonstrates enhanced throughput, surpassing PoW and PoS, which achieve 5 and 15 transactions per second, respectively. Furthermore, the error rate is reduced to 0.5%, indicating higher reliability. The proposed system also offers excellent scalability, supporting up to 450 transactions per block, compared to 100 (PoW) and 200 (PoS). Finally, its consensus finality is achieved in just 15 seconds, ensuring rapid transaction validation compared to PoW's 120 seconds and PoS's 50 seconds.

Table 32: Comparative Analysis of Proposed Consensus Mechanisms in Blockchain-Based IDS Frameworks

Metrics	Proof of Stakes (PoS)	Proof of Work (PoW)	Proposed PBFT
Average Gas Computation (Gwei)	38500	45000	**29500**
Average Transaction Fee (FTH)	0.000045	0.000050	**0.000022**
Processing Speed (Tx/Speed)	15	5	**35**
Error rate (%)	1.2	2.5	**0.5**
Scalability (Max Tx/Block)	200	100	**450**
Consensus Finality (seconds)	50	120	**15**

5.4.2 Result Analysis of Proposed Intrusion Detection Model (Phase 2)

This phase focused on evaluating the performance of the proposed GA-Optimized XGBoost (GAO-Xgboost) model against other GA-optimized models for intrusion detection in heterogeneous IoT networks. The performance evaluation of various intrusion detection models, optimized using Genetic Algorithms (GA), including the proposed GA-Optimized XGBoost (GAO-Xgboost) model, is detailed in Tables 22 and 23. The findings highlight the efficacy of GA in enhancing model performance and demonstrate the superior capabilities of the proposed framework for detecting intrusions in heterogeneous IoT networks.

Table 33: Quantitative analysis of the proposed model

Models	Accuracy	Precision	Recall	F1 Score

GA-LR	87.32	86.19	87.04	86.61
GA-KNN	89.72	88.72	89.11	88.91
GA-SVM	89.54	89.47	88.23	88.85
GA-DT	92.76	91.23	91.58	91.41
GA-RF	95.47	95.17	94.97	95.07
Proposed IDS Model (GAO-Xgboost)	**98.12**	**97.76**	**97.81**	**97.78**

Table 34: Qualitative analysis of the proposed model

Models	TPR	FPR	TNR	FNR	NPV	PPV	FDR	FOR
GA-LR	87.04	13.81	86.19	12.96	86.19	86.19	13.69	13.07
GA-KNN	89.11	11.28	88.72	10.89	88.72	88.72	11.23	10.93
GA-SVM	88.23	10.53	89.47	11.77	89.42	89.47	10.66	11.62
GA-DT	91.58	8.77	91.23	8.42	91.23	91.23	8.74	8.45
GA-RF	94.97	4.83	95.17	5.03	95.17	95.17	4.84	5.02
Proposed IDS Model (GAO-Xgboost)	**97.81**	**2.24**	**97.76**	**2.19**	**97.76**	**97.76**	**2.24**	**2.19**

The study utilized a Genetic Algorithm (GA) to optimize several machine learning models, including Logistic Regression (LR), K-Nearest Neighbors (KNN), Support Vector Machine (SVM), Decision Tree (DT), and Random Forest (RF). The optimization process was focused on refining detection rules and selecting the most relevant features, thus improving the models' accuracy and efficiency in classifying network traffic. The GA-LR model achieved an accuracy of 87.32%, with precision and recall values of 86.19% and 87.04%, respectively.

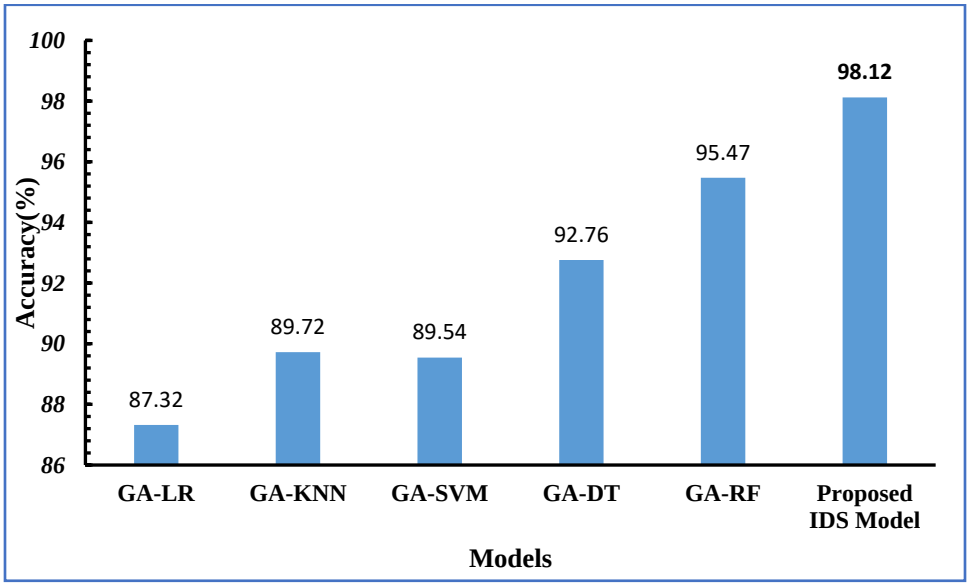


Figure 44: Accuracy Comparison: GA-Based Models vs. the Proposed IDS Model

Although the model's performance was reasonable, it exhibited a slightly lower precision-recall balance, indicating a higher incidence of false positives. The GA-KNN model improved, with an accuracy of 89.72% and a balanced F1 score of 88.91%. This model effectively minimized false alarms while maintaining a satisfactory detection rate. The GA-SVM model demonstrated comparable accuracy at 89.54%, although its recall of 88.23% suggested a potential underestimation of certain intrusion types. The GA-DT model exhibited notable performance with an accuracy of 92.76%, indicating that GA optimization significantly enhanced its decision-making process. The GA-RF model stood

out with an accuracy of 95.47%, bolstered by high precision (95.17%) and recall (94.97%), making it particularly robust in identifying a wide range of intrusion scenarios.

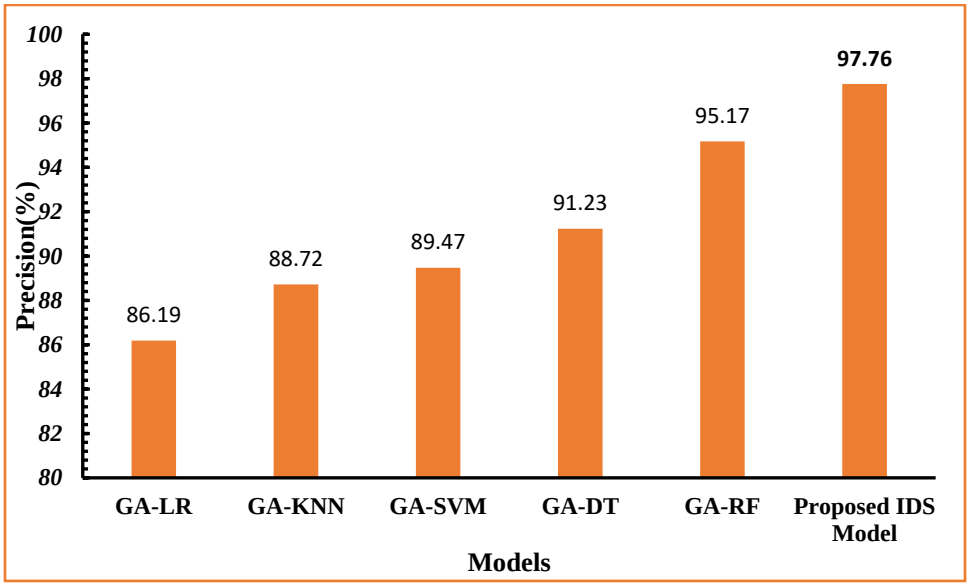


Figure 45: Precision Comparison: GA-Based Models vs. the Proposed IDS Model

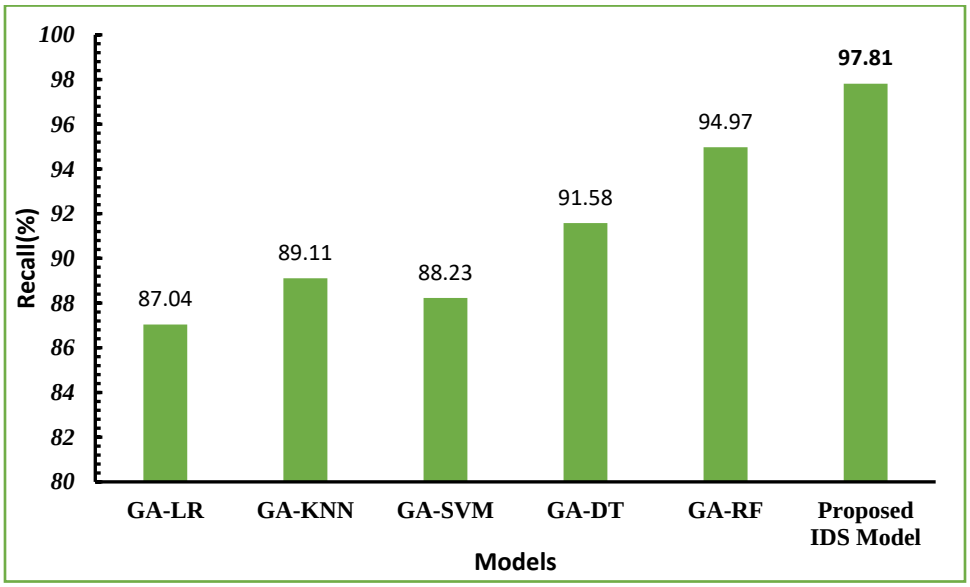


Figure 46: Recall Comparison: GA-Based Models vs. the Proposed IDS Model

The proposed GAO-Xgboost model demonstrated exceptional performance, outperforming other models in the study, as shown in Figures 10,11,12,13. It achieved an accuracy of 98.12%, reflecting a high level of correctness in identifying legitimate and malicious activities. The model's precision of 97.76% indicates its effectiveness in minimizing false positives, ensuring that most detected intrusions were genuine threats. With a recall of 97.81%, the model effectively captured actual intrusions, reducing the likelihood of missed threats. The F1 score of 97.78% confirms the model's balanced performance, making it a reliable choice for practical deployment.

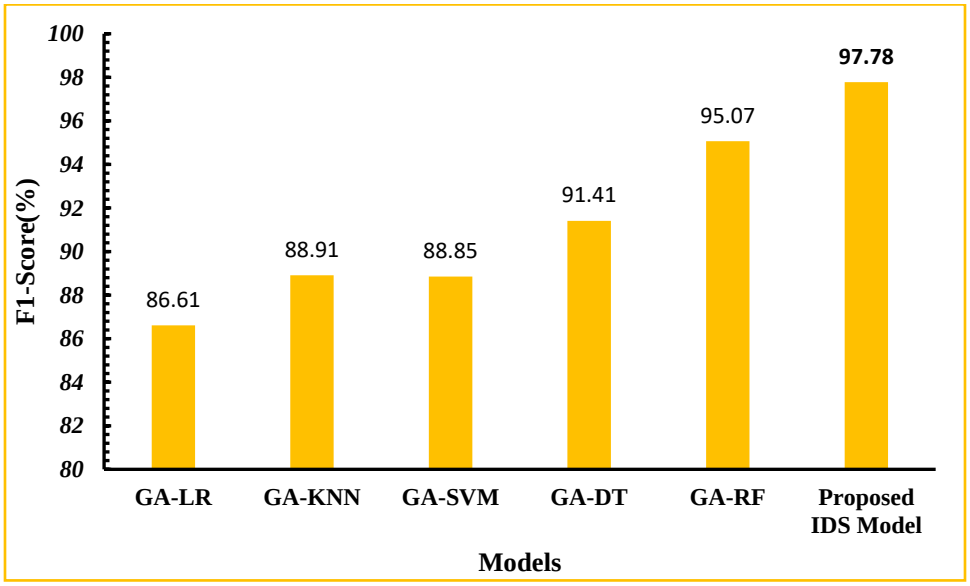


Figure 47: F1-Score Comparison: GA-Based Models vs. the Proposed IDS Model

To further evaluate the performance of the proposed approach on the Edge_IIoT dataset, the Receiver Operating Characteristic (ROC) curve was employed. This curve illustrates the relationship between the false positive rate (false alarm probability) and the true positive rate (detection probability or recall). The Area Under the Curve (AUC) was then computed to quantify the overall performance across all ROC curves. Notably, the AUC values for all attack categories exceeded 99%, as depicted in Figure 14, highlighting the effectiveness and reliability of the proposed method for attack classification.

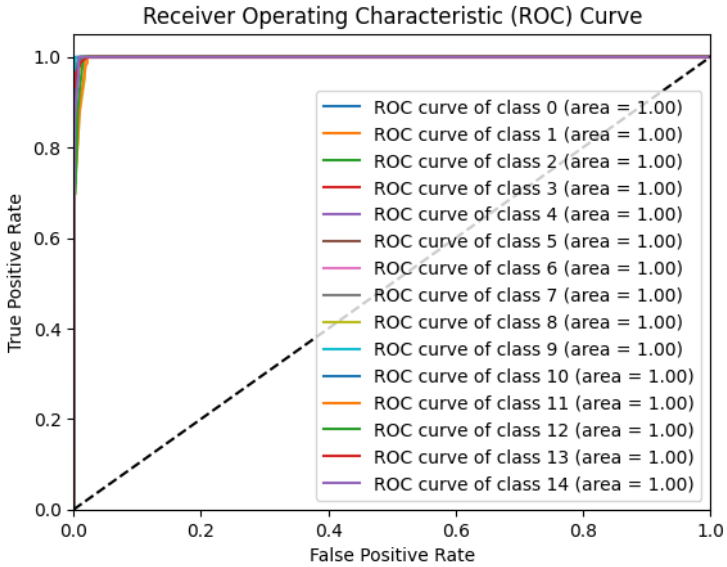


Figure 48: ROC Curve of the Proposed Model

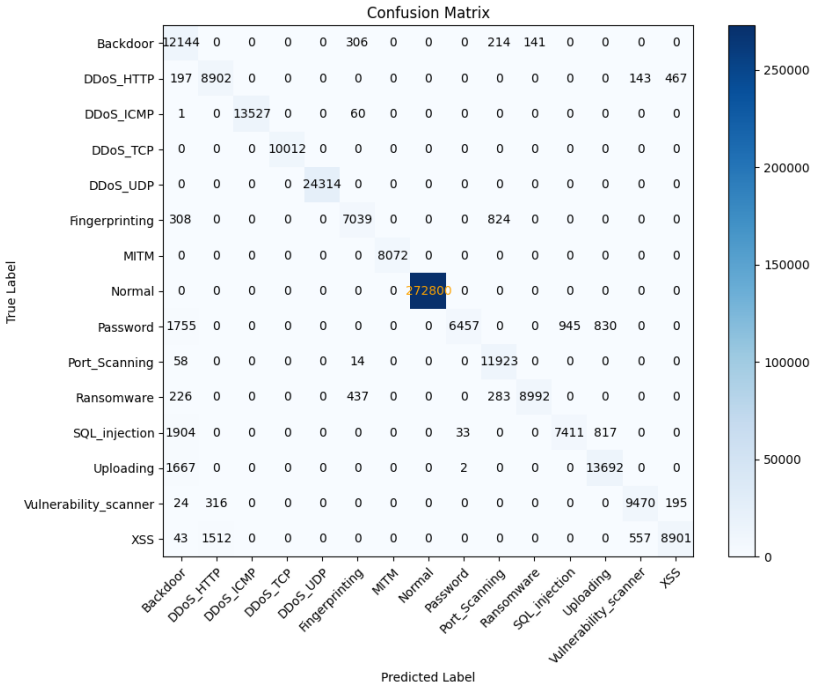


Figure 49: Confusion Matrix of the proposed IDS model

The confusion matrix was employed to evaluate the effectiveness of the algorithms further, covering both correct and incorrect classifications. Figure 15 presents the multi-class confusion matrices for the proposed model, where the columns represent predicted instances of each category, and the rows correspond to the actual instances. Diagonal elements of the matrix denote the true positive values from the Edge_IIoT dataset, while non-diagonal elements indicate misclassified samples. Analysis of the confusion matrix reveals that the proposed approach achieves a high number of true positive classifications, demonstrating its superior categorization performance.

In addition to these metrics, the GAO-Xgboost model exhibited outstanding True Positive Rate (TPR) and True Negative Rate (TNR), both exceeding 97%, while maintaining low False Positive Rate (FPR) and False Negative Rate (FNR) values. These results underscore the model's capability to accurately differentiate between normal and malicious traffic, making it highly suitable for diverse and complex IoT environments. The False Omission Rate (FOR) and False Discovery Rate (FDR) results highlight the effectiveness of the proposed IDS model (GAO-Xgboost) in accurately detecting intrusions. With a FOR of 2.19% and an FDR of 2.24%, the model outperforms traditional methods, significantly reducing false negatives and false positives. This ensures that actual threats are identified without overwhelming the system with false alarms, making the GAO-Xgboost model a robust and reliable choice for intrusion detection in complex environments where precision and security are paramount.

Figure 16 provides a comparative analysis of the True Positive Rate (TPR) and False Negative Rate (FNR) for several intrusion detection systems (IDS) models enhanced with Genetic Algorithms (GA) and various machine learning classifiers, including Logistic Regression (GA-LR), K-Nearest Neighbors (GA-KNN), Support Vector Machine (GA-SVM), Decision Tree (GA-DT), and Random Forest (GA-RF). The proposed IDS model, combining Genetic Algorithm Optimization with XGBoost (GAO-XGBoost), demonstrates superior performance across both metrics. It achieves the highest TPR at 97.81%, indicating its exceptional ability to identify actual intrusions correctly. Concurrently, it boasts the lowest FNR at 2.19%, underscoring its effectiveness in minimizing missed detection rates. Among the other models, GA-RF shows strong performance with a TPR of 94.97% and an FNR of 5.03%, while GA-DT also performs well with a TPR of 91.58% and an FNR of 8.42%.

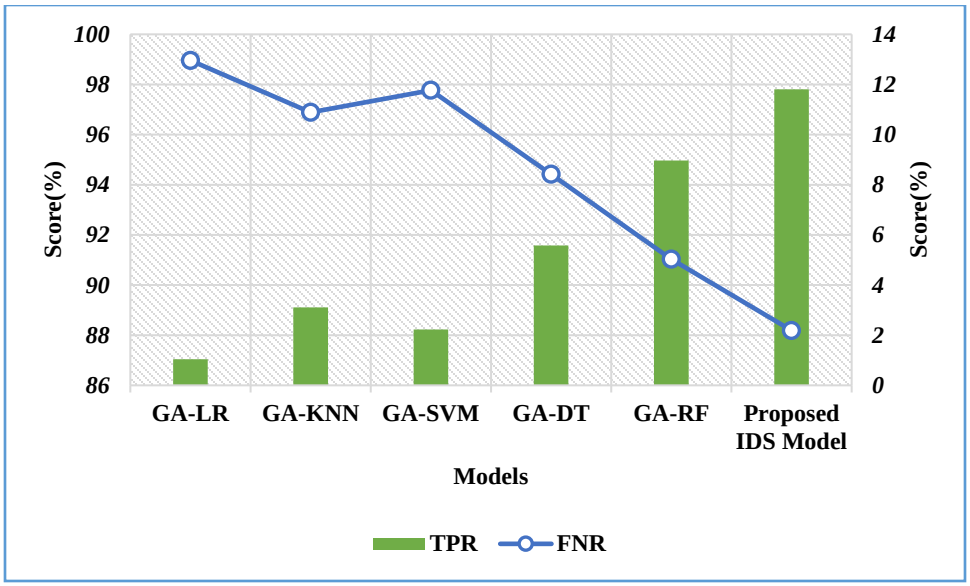


Figure 50: Performance Evaluation: TPR and FNR Analysis of GA-Optimized Models Versus the Proposed IDS Model

Figure 17 compares the True Negative Rate (TNR) and False Positive Rate (FPR) of various intrusion detection system (IDS) models enhanced with Genetic Algorithms (GA) and different machine learning classifiers, including Logistic Regression (GA-LR), K-Nearest Neighbors (GA-KNN), Support Vector Machine (GA-SVM), Decision Tree (GA-DT), and Random Forest (GA-RF). The proposed IDS model, combining Genetic Algorithm Optimization with XGBoost (GAO-XGBoost), demonstrates the highest performance in both metrics. It achieves an impressive TNR of 97.76%, indicating its exceptional ability to correctly identify non-intrusions while maintaining the lowest FPR at 2.24%, reflecting its effectiveness in minimizing false alarms. Among the other models, GA-RF stands out with a strong TNR of 95.17% and an FPR of 4.83%, followed by GA-DT with a TNR of 91.23% and an FPR of 8.77%.

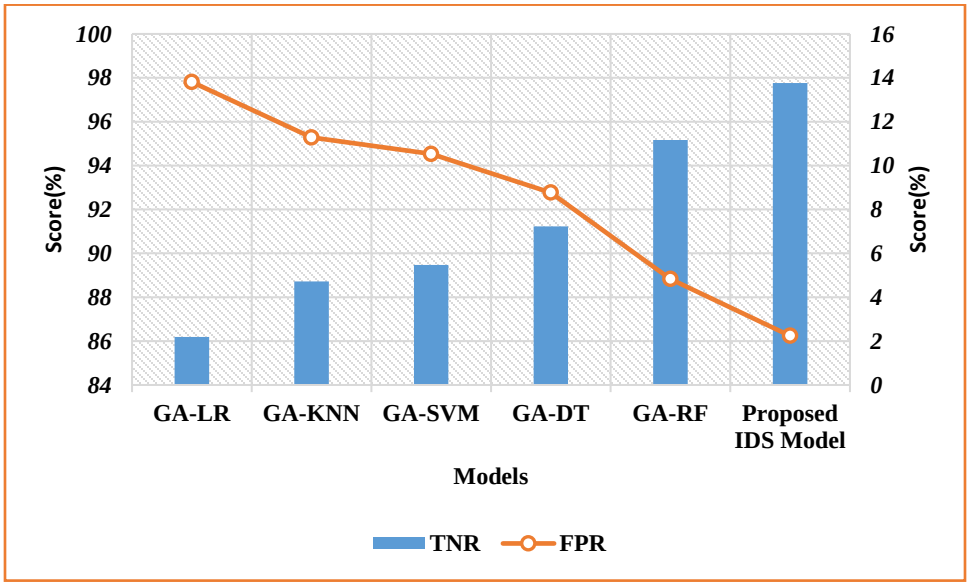


Figure 51: Performance Evaluation: TNR and FPR Analysis of GA-Optimized Models Versus the Proposed IDS Model

Figure 18 presents a comparison of the Negative Predictive Value (NPV) and False Omission Rate (FOR) for various intrusion detection system (IDS) models, each enhanced with Genetic Algorithms (GA) and different machine learning

classifiers, such as Logistic Regression (GA-LR), K-Nearest Neighbors (GA-KNN), Support Vector Machine (GA-SVM), Decision Tree (GA-DT), and Random Forest (GA-RF). The proposed IDS model, which combines Genetic Algorithm Optimization with XGBoost (GAO-XGBoost), exhibits the highest NPV at 97.76%, indicating its strong capability to identify non-intrusions among all instances classified as non-intrusions correctly. It also achieves the lowest FOR at 2.19%, demonstrating its effectiveness in minimizing the proportion of actual positives (intrusions) that are incorrectly classified as negatives (non-intrusions). Among the other models, GA-RF shows notable performance with an NPV of 95.17% and a FOR of 5.02%, followed by GA-DT with an NPV of 91.23% and a FOR of 8.45%.

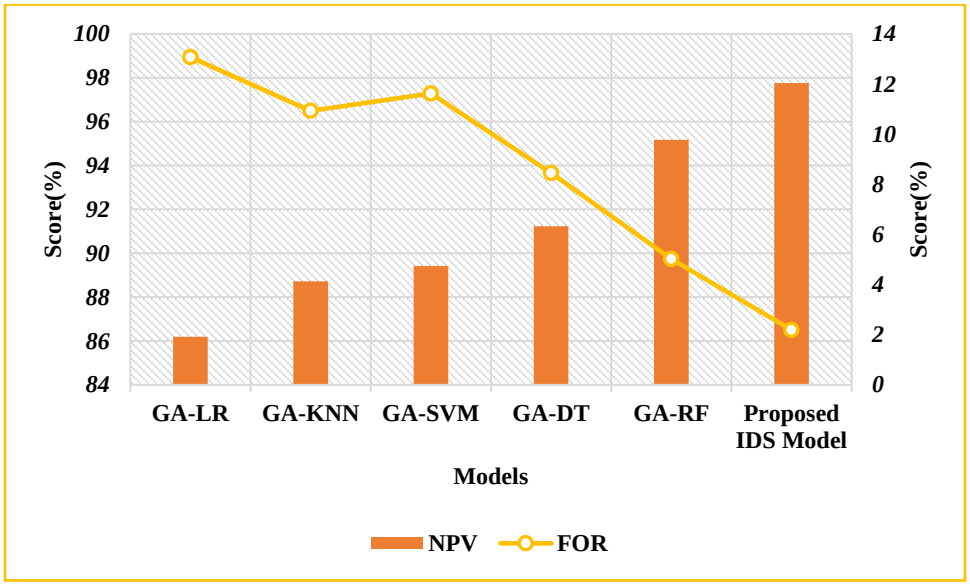


Figure 52: Performance Evaluation: NPV and FOR Analysis of GA-Optimized Models Versus the Proposed IDS Model

Figure 19 compares the Positive Predictive Value (PPV) and False Discovery Rate (FDR) of several intrusion detection system (IDS) models, each enhanced with Genetic Algorithms (GA) and different machine learning classifiers, including Logistic Regression (GA-LR), K-Nearest Neighbors (GA-KNN), Support Vector Machine (GA-SVM), Decision Tree (GA-DT), and Random Forest (GA-RF). The proposed IDS model, which integrates Genetic Algorithm Optimization with XGBoost (GAO-XGBoost), achieves the highest PPV at 97.76%, signifying its strong accuracy in correctly identifying true positives (actual intrusions) among all instances classified as positives. Additionally, it maintains the lowest FDR at 2.24%, indicating a minimal proportion of false positives (non-intrusions incorrectly classified as intrusions). Among the other models, GA-RF also performs well, with a PPV of 95.17% and an FDR of 4.84%, followed by GA-DT with a PPV of 91.23% and an FDR of 8.74%. This comparison highlights the proposed IDS model's superior precision and reduced likelihood of false alarms, making it a more effective and reliable solution for accurately detecting intrusions in IoT environments.

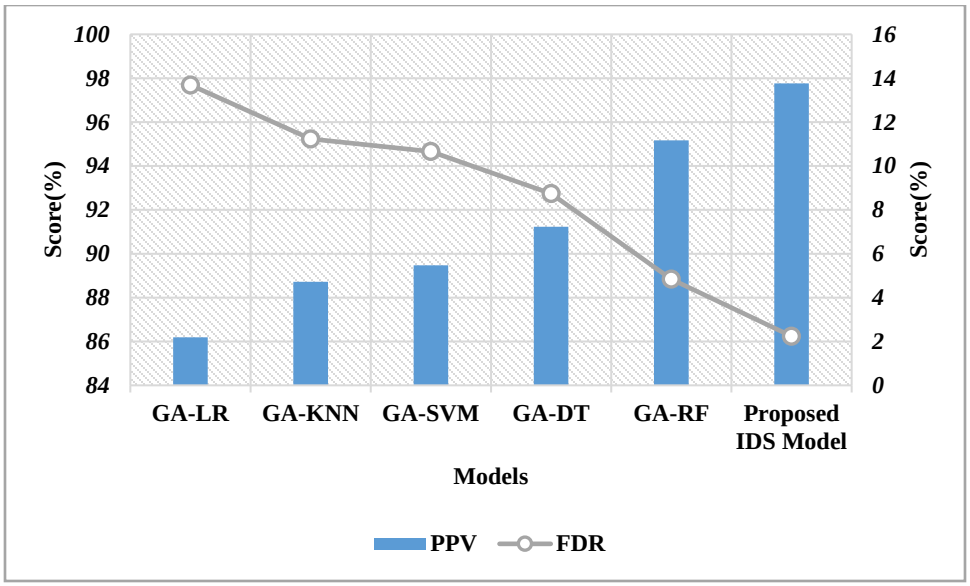


Figure 53: Performance Evaluation: PPV and FDR Analysis of GA-Optimized Models Versus the Proposed IDS Model

Overall, applying GA optimization significantly improved the performance metrics of these models. The enhanced precision, recall, and F1 scores underscore the importance of GA in refining model parameters to achieve optimal results in intrusion detection. The Genetic Algorithm (GA) significantly enhanced the performance of traditional machine learning models, while the proposed GAO-Xgboost model demonstrated superior capabilities in intrusion detection. Including less favorable results, such as the relatively lower recall of the GA-SVM model, ensures a comprehensive understanding of the research outcomes. Overall, the study emphasizes the effectiveness of GA optimization and the robustness of the Xgboost-based approach in securing IoT networks.

5.5. Security and Privacy Analysis of the Proposed Framework

5.5.1. Security Analysis

The proposed blockchain framework is designed to ensure robust security and privacy for secure data management in IoT environments. This section analyzes the framework's resilience against various attacks, highlighting its ability to maintain data integrity, confidentiality, and availability.

- **Denial of Service (DoS) and Distributed Denial of Service (DDoS) Attacks:** The proposed framework employs a combination of ECC, DSA, and SHA-512 for securing the Intrusion Detection System (IDS) and utilizes the PBFT consensus algorithm for decision-making within the blockchain. The robust cryptographic algorithms (ECC and DSA) and the consensus mechanism of PBFT provide robust defenses against DoS and DDoS attacks by ensuring that malicious nodes cannot overwhelm the network. Additionally, the decentralized nature of the blockchain makes it resilient to these attacks as the distributed network can manage and mitigate the impact of such attacks.
- **Information Gathering:** The proposed framework uses encryption and hashing mechanisms to prevent unauthorized information gathering. Patient data is encrypted using ECC, and SHA-512 ensures the integrity of this data. Encryption will thwart attempts to gather information with authorization, making it nearly impossible for attackers to decipher the data. The authentication provided by DSA further ensures that only authorized entities can access the information, safeguarding patient privacy.
- **Injection Attacks:** The framework's reliance on the blockchain's immutable ledger and cryptographic validation (ECC and DSA) prevents injection attacks. Data entered into the blockchain is verified through consensus (PBFT) and cannot be altered once confirmed. This immutability ensures that any attempt to inject malicious data into the system is detected and rejected, preserving the integrity of the data.

- **Man-in-the-Middle (MiTM) Attacks:** To counter MiTM attacks, the framework employs robust encryption (ECC) and digital signatures (DSA) for all communications. These cryptographic methods ensure that any intercepted data remains encrypted and unreadable by unauthorized parties. Additionally, using secure keys generated by the SADE algorithm further strengthens the encryption, making it extremely difficult for attackers to decrypt or alter the data in transit.
- **Malware Attacks:** The decentralized and encrypted nature of the blockchain provides a strong defense against malware attacks. Any data stored off-chain in IPFS is linked through blockchain references, ensuring that only verified and untampered data is retrievable. The consensus mechanism (PBFT) ensures that any attempt to introduce malware into the blockchain is identified and blocked by the network of nodes, maintaining the integrity and security of the data.

The proposed blockchain framework, integrating ECC, DSA, SHA-512, PBFT consensus, and IPFS for off-chain storage, offers robust protection against various security threats. It mitigates the risks associated with DoS and DDoS attacks, unauthorized information gathering, injection attacks, MiTM attacks, and malware attacks. By ensuring data integrity, confidentiality, and availability, the framework provides a secure, privacy-preserving solution for managing patient healthcare data in IoT environments.

5.5.2. Privacy Analysis

The proposed blockchain framework prioritizes patient privacy through advanced cryptographic techniques and strict data management protocols. This section explores how the framework safeguards confidentiality, integrity, authentication, anonymity, and access control, ensuring comprehensive privacy protection.

- **Data Confidentiality:** The proposed framework ensures data confidentiality through Elliptic Curve Cryptography (ECC). ECC encrypts sensitive patient data, making it accessible only to authorized parties with the correct decryption keys. This robust encryption provides high security with smaller key sizes, which is especially important for resource-constrained IoT devices. Using ECC, the framework ensures that IoT data remains confidential and protected from unauthorized access.
- **Data Integrity:** The framework uses SHA-512 for hashing data, providing a unique hash value for each input. This hash value ensures data integrity by detecting any alterations to the data. If the data is tampered with, the hash value changes, signaling a breach in integrity. This mechanism ensures that the data remains unaltered from its original form, providing a high level of trust in its authenticity.
- **Data Authentication:** The Digital Signature Algorithm (DSA) is used to authenticate communication within the blockchain network. DSA verifies the parties' identity in the communication, ensuring that legitimate entities send and receive the data. This authentication process prevents impersonation attacks, where malicious actors attempt to masquerade as legitimate users to gain access to sensitive data.
- **Data Anonymity:** To protect IoT data privacy further, the framework can implement data anonymization techniques before storing data on the blockchain. Anonymization ensures that individual IoT data identities cannot be discerned from the stored data, protecting personal information while allowing for aggregated data analysis.
- **Access Control:** The framework can incorporate role-based access control (RBAC) to ensure that only authorized personnel can access specific data types. This layered access control mechanism ensures that sensitive data is only available to those who need it, minimizing the risk of data breaches and ensuring compliance with privacy regulations.
- **Decentralization and Transparency:** Blockchain technology inherently provides transparency while maintaining privacy. Transactions and data entries are recorded on an immutable ledger, providing a transparent record that can be audited while keeping the data encrypted and private. This transparency enhances trust in the system, as all actions are traceable and accountable without compromising individual privacy.
- **Off-Chain Data Storage:** The InterPlanetary File System (IPFS) stores large volumes of data off-chain, with the blockchain containing only the references to this data. This approach balances the need for decentralized, resilient storage with privacy considerations, as data stored in IPFS is linked to its blockchain

reference, ensuring its integrity and traceability without storing sensitive information directly on the blockchain.

The proposed blockchain framework excels in protecting IoT data privacy through robust cryptographic techniques, data integrity measures, authentication protocols, and controlled access mechanisms. By integrating ECC, DSA, SHA-512, PBFT consensus, and IPFS for off-chain storage, the framework ensures that IoT data remains confidential, integral, and accessible only to authorized users. These privacy-preserving features make the framework highly suitable for managing sensitive healthcare data in IoT environments, addressing the critical needs for both security and privacy in modern digital healthcare systems.

5.6. Comparison with the Existing Techniques and State-of-the-Art Methods

This section comprehensively compares the proposed GAO-XGBoost model with existing techniques and state-of-the-art methods for intrusion detection systems (IDS). By analyzing the performance across various metrics and datasets, the evaluation highlights the proposed model's efficiency, accuracy, and robustness in addressing the challenges of intrusion detection in IoT environments. The results underscore the advantages of integrating Genetic Algorithm Optimization (GAO) with XGBoost, showcasing its superiority over traditional, deep learning, and cutting-edge approaches.

5.6.1. Comparison with the existing techniques

This subsection evaluates the performance of the proposed GAO-XGBoost model against several widely used machine learning and deep learning approaches for intrusion detection systems (IDS). The comparison is based on four standard metrics: accuracy, precision, recall, and F1 score. The aim is to demonstrate the advantages of combining Genetic Algorithm Optimization (GAO) with XGBoost for enhanced IDS performance. From the table 24, traditional machine learning models like Logistic Regression (LR) and K-Nearest Neighbors (KNN) exhibit relatively modest accuracies of 83.54% and 85.23%, respectively. These models are outperformed by Support Vector Machine (SVM) and Decision Tree (DT), which achieve accuracies of 87.02% and 89.43%, respectively. Among these, Random Forest (RF) performs the best, with an accuracy of 93.87%.

Deep learning models show a significant improvement over traditional methods. Convolutional Neural Networks (CNN) and Recurrent Neural Networks (RNN) achieve accuracies of 93.27% and 94.65%, respectively, reflecting their ability to capture complex patterns in the dataset. However, despite their advanced capabilities, these models still fall short compared to the proposed GAO-XGBoost framework.

The proposed model achieves an accuracy of 98.12%, precision of 97.76%, recall of 97.81%, and F1 score of 97.48%, significantly surpassing all other approaches. These improvements can be attributed to Genetic Algorithm Optimization (GAO), which enhances hyperparameter tuning, optimizes feature selection, and mitigates overfitting. This optimization allows XGBoost to achieve superior classification performance, making it particularly well-suited for intrusion detection in IoT environments. The results emphasize that the integration of GAO with XGBoost provides a highly efficient and reliable IDS, outperforming both traditional and deep learning techniques in terms of accuracy and other performance metrics.

Table 35: Performance Comparison of the Proposed GAO-XGBoost Model with Traditional Machine Learning and Deep Learning Techniques

Models	Accuracy	Precision	Recall	F1 Score
LR (Logistic Regression)	83.54	82.12	82.98	82.55
KNN (K-Nearest Neighbors)	85.23	84.71	84.89	84.80
SVM (Support Vector Machine)	87.02	86.91	86.32	86.91
DT (Decision Tree)	89.43	88.71	88.34	88.52
RF (Random Forest)	93.87	92.65	93.12	92.88
CNN (Convolutional Neural Network)	93.27	92.15	92.45	92.30
RNN (Recurrent Neural Network)	94.65	93.85	94.15	94
Proposed Model(GAO-Xgboost)	**98.12**	**97.76**	**97.81**	**97.48**

5.6.2. Comparison with the State-of-the-Art Methods

The proposed XGBoost model, integrated with the Genetic Algorithm (GA) for feature selection, outperforms state-of-the-art techniques in intrusion detection, as shown in Table 25. Compared to models like Growing Tree Clustering (97.12%) and iForest (97.37%), our approach achieves the highest accuracy of 98.12%, demonstrating superior classification capabilities. Earlier methods, such as REPT using PSO and OC-SVM with PIO, achieved accuracies of 96.38% and 83%, respectively, indicating limitations in handling complex intrusion patterns. The effectiveness of the proposed model highlights the advantage of combining XGBoost with GA, showcasing its robustness and efficiency in detecting attacks with improved accuracy over existing methods.

Table 36: Comparison of the Proposed Model with State-of-the-Art Techniques

References and Year	Model	Feature Selection Technique	Accuracy
[43] and 2024	Growing tree clustering	Genetic Algorithm (GA)	97.12
[44] and 2019	REPT	PSO	96.38
[45] and 2022	OC-SVM	PIO	83
[46] and 2023	iforest	LS-PIO	97.37
Our approach	**Proposed Xgboost**	**Genetic Algorithm (GA)**	**98.12**

6. Conclusion and Future Directions

This section concludes the manuscript by presenting the conclusions and suggesting potential directions for future research.

6.1. Conclusion

This research presents a Hybrid Blockchain-Based Framework that effectively addresses the security challenges associated with Intrusion Detection Systems (IDS) in IoT networks. By integrating advanced cryptographic techniques, Elliptic Curve Cryptography (ECC), the Digital Signature Algorithm (DSA), and SHA-512 alongside a novel Self-Adaptive Differential Evolution (SADE) algorithm, the proposed framework significantly enhances data privacy, authentication, and integrity. Practical Byzantine Fault Tolerance (PBFT) for consensus and InterPlanetary File System (IPFS) for off-chain data storage contributes to system resilience and prevents centralized failures. Furthermore, applying Genetic Algorithms (GA) and an XGBoost-based model optimizes IDS performance, yielding high accuracy and robust intrusion detection capabilities. The results demonstrate substantial improvements in blockchain latency, throughput, network overhead, and exceptional performance metrics from the XGBoost model. Overall, this comprehensive approach significantly advances IoT security, offering enhanced effectiveness, scalability, and resilience.

6.2. Potential Industrial Applications

The following are the Potential Industrial Applications of the Proposed Framework:

- Smart Manufacturing: Enhances the security of IoT-enabled manufacturing systems by providing robust intrusion detection and secure data handling, ensuring uninterrupted and safe industrial processes.
- Critical Infrastructure Protection: Safeguards essential infrastructure such as smart grids and energy management systems from sophisticated cyberattacks, maintaining system integrity and reliability.
- Industrial Automation: Improves the security of automated systems and robotics by detecting and preventing intrusions, ensuring operational safety and efficiency in industrial environments.
- Supply Chain Management: Secures data exchanged between IoT devices in supply chain networks, protecting against tampering and unauthorized access and enhancing overall supply chain security.

6.3. Limitation

The framework's blockchain component introduces latency and network overhead, while the XGBoost model requires extensive computational resources. Adaptability to new cryptographic advancements and consensus models remains a challenge.

6.4. Future Work

Future research will explore optimizing the SADE algorithm further, evaluating alternative consensus mechanisms, integrating real-time threat intelligence, and adapting the framework to emerging technologies and new IoT applications.

6.5. Societal Application

The following are the Societal Applications of the proposed Framework:

- Smart Homes: Enhances the security and privacy of IoT devices in smart homes, such as smart thermostats and security cameras, protecting personal data and ensuring the safe operation of household systems.
- Healthcare Monitoring: Secures IoT-based health monitoring systems, ensuring the confidentiality and integrity of sensitive patient data and improving trust in remote healthcare technologies.
- Public Safety Systems: Protects IoT-based public safety networks, such as surveillance and emergency response systems, from cyber threats, ensuring reliable and secure public safety services.
- Consumer IoT Devices: Safeguards a range of consumer IoT devices, from wearable technology to smart appliances, by enhancing their security features and protecting users' personal information from breaches.

Chapter 6 Privacy-Preserving Data Sharing in Blockchain-Enabled IoT Healthcare Management System

1. Introduction

The rapid growth and adoption of the Internet of Things (IoT) in healthcare systems have significantly advanced the smart healthcare industry by providing effective solutions for saving, maintaining, securing, and sharing healthcare documents [1]. Despite these advancements, the proliferation of IoT devices and the massive influx of healthcare data pose substantial security and privacy challenges for confidential medical documents. IoT-based healthcare models typically use cloud storage systems for data storage and maintenance [2]. However, the centralized architecture of these cloud systems introduces vulnerabilities, such as single-point failures, and an increased risk of security attacks, including masquerades, phishing, identity theft, and data breaches. Consequently, the privacy and security of healthcare documents remain primary concerns as nearly all healthcare data is stored online.

Traditional approaches in the healthcare industry have relied on encoding schemes and cryptographic methods to safeguard healthcare documents [3]. Often maintained on centralized databases or cloud storage systems, these methods are prone to single-point failure issues and various attacks [4]. Furthermore, attackers frequently clone or duplicate medical documents, creating significant challenges for users who need to access their medical records for treatment, especially if they misplace critical documents [5].

Blockchain technology offers a promising solution, innovative cryptography and distributed systems development. Initially introduced by Santoshi Nakamoto through the cryptocurrency Bitcoin, blockchain technology features a peer-to-peer network that maintains data in a distributed ledger [6]. This decentralized approach, characterized by data integrity, transparency, and security through cryptographic algorithms, has shown potential across various domains, including finance, healthcare, and supply chain management [7]. In a blockchain, data is stored in hash format within blocks, each linked to the previous one, forming an immutable chain validated by consensus algorithms.

Integrating blockchain technology with IoT-based healthcare systems can enhance transparency, privacy, and security while reducing costs. Blockchain enables the healthcare system to maintain trust among its entities by securely managing healthcare data and verifying the authenticity of healthcare documents, thereby preventing fraud [8]. It stores healthcare details, such as unique identification numbers and certifications, in a blockchain structure, strengthening the healthcare system against counterfeiting and unauthorized modifications [9]. Smart contract functionalities within blockchain technology further enhance the security and privacy of healthcare documents by automating agreements and ensuring they are executed only when specified conditions are met. The generic architecture of Blockchain based IoT based Healthcare Management System is shown in Figure 1. The decentralized nature of blockchain technology and its cryptographic strengths make it an ideal solution for addressing the security and privacy challenges in IoT-based healthcare systems [10]. Researchers have proposed various blockchain-based models to enhance the security of medical records [11-13]. For instance, Rhayem et al. [11] developed a blockchain-based medical system using the Ethereum framework to achieve traceability without involving trusted third parties, guaranteeing data security, privacy, and transparency. Other applications, such as the Internet of Medical Things (IoMT), utilize smart contracts to manage healthcare records, monitor sensitive data, and ensure secure access to global data.

Despite these advancements, existing systems still require comprehensive security and privacy analysis to manage healthcare data effectively. Integrating blockchain technology with healthcare can revolutionize the industry by providing a robust IoT healthcare system that ensures transparency, security, and traceability. This paper explores the various aspects of blockchain-based healthcare systems and proposes a framework for implementing a blockchain-based IoT system in the healthcare sector, emphasizing the need for secured centralized systems to protect electronic health records (EHRs) from privacy leaks, data breaches, and other security threats. The proposed solution aims to reduce storage costs, ensure patient data security, and prevent unauthorized access while maintaining data integrity and availability through blockchain's decentralized approach.

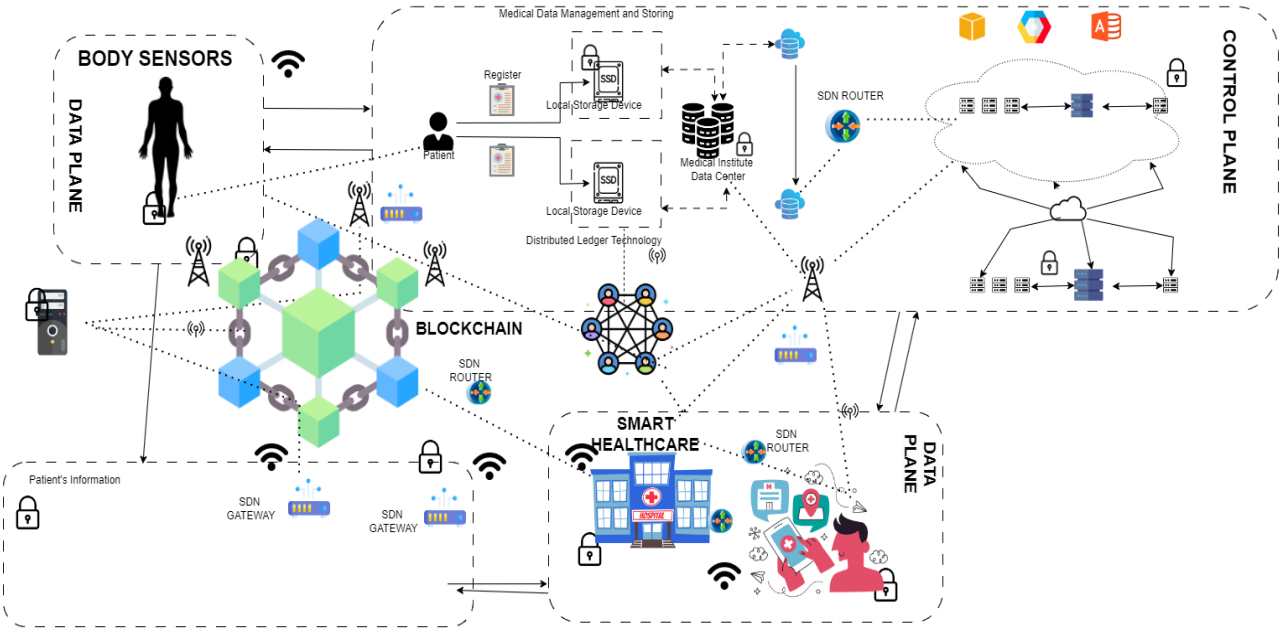


Figure 54: Blockchain-based IoT based Healthcare Management System

1.1. Motivation

In the modern healthcare ecosystem, ensuring the security, privacy, and accessibility of medical records is a critical challenge. Traditional centralized systems for managing healthcare data and medical certificates are plagued by vulnerabilities such as single points of failure, data breaches, unauthorized access, and document tampering. These inefficiencies not only compromise patient trust but also hinder seamless healthcare delivery. Additionally, the rapid proliferation of IoT devices in healthcare generates vast amounts of sensitive data, which centralized systems struggle to manage efficiently, leading to latency and scalability issues. Blockchain technology, with its decentralized, immutable, and transparent nature, offers a promising solution to address these challenges. By leveraging advanced cryptographic techniques and consensus mechanisms, blockchain ensures data integrity, enhances privacy, and eliminates the risks associated with centralized control. Despite the potential of blockchain-based solutions, existing implementations often fail to address critical requirements such as high transaction throughput, real-time scalability, and comprehensive privacy preservation. This research aims to bridge these gaps by proposing a robust blockchain-based framework that integrates IoT devices, hybrid consensus mechanisms, and privacy-preserving cryptographic techniques, thereby revolutionizing medical record management in healthcare.

This paper introduces a novel decentralized application that uses blockchain technology to enhance medical certificate management security, privacy, and efficiency in the healthcare sector. The key contributions of this work include:

- **Proposed a Novel Blockchain-Based IoT Application for Healthcare:** We propose an innovative decentralized application that utilizes blockchain technology to create a secure communication medium between healthcare entities such as hospitals, patients, and doctors. This system integrates various IoT devices to maintain and update healthcare information seamlessly.
- **Introducing Unique Identification for Medical Certificates:** We introduce a mechanism to generate unique identification numbers for each medical certificate, ensuring the integrity and traceability of medical records.

- **Enhancing Consensus Mechanisms:** We deploy a hybrid consensus mechanism that combines Proof of Work (PoW) and Practical Byzantine Fault Tolerance (PBFT). This hybrid approach improves security and transaction speed, making the system capable of handling high transaction volumes in healthcare settings.
- **Implementing Privacy-Preserving Computations:** We implement Fully Homomorphic Encryption (FHE) to allow secure computations on encrypted medical data. This ensures that sensitive information remains private throughout the processing stages.
- **Utilizing Non-Interactive Zero-Knowledge Proofs (NIZKPs):** We utilize NIZKPs to verify medical certificates without disclosing sensitive patient data, thus enhancing privacy and trust in the system.
- **Incorporating Interplanetary File System (IPFS):** We use IPFS as a distributed file system to distribute and store all IoT data due to the limited storage capacity available on each block. This approach enables the proposed architecture to handle large amounts of IoT data and scale effectively.
- **Integrating an Intrusion Detection System (IDS):** We propose an IDS to monitor IoT traffic, detecting potential security attacks and anomalies in real time. The IDS enhances overall system security by identifying and mitigating potential threats before they can cause harm.
- **Conducting Comprehensive Evaluation:** We perform extensive experimental tests to evaluate the proposed application on various parameters such as latency, computation time, processing time, throughput, and network usage, demonstrating its effectiveness and robustness.

The proposed framework distinguishes itself by integrating blockchain technology with IoT-based healthcare systems using a novel combination of mature techniques, including Fully Homomorphic Encryption (FHE), Non-Interactive Zero-Knowledge Proofs (NIZKPs), and a hybrid consensus mechanism combining Proof of Work (PoW) and Practical Byzantine Fault Tolerance (PBFT). While these techniques have been explored independently in other domains, their synergistic application within the context of IoT healthcare systems addresses critical gaps. Specifically, the hybrid consensus mechanism balances scalability and security, overcoming the high latency of PoW and the vulnerability of PBFT in high-volume systems. The use of FHE ensures secure computations on encrypted data, enabling privacy-preserving operations, while NIZKPs further enhance trust by allowing verification of sensitive data without exposing it. This integrated approach ensures robust performance and privacy preservation, which traditional systems fail to achieve due to their reliance on isolated or centralized methods. This research addresses these gaps by proposing a novel blockchain-based decentralized application that integrates IoT devices, implements a hybrid consensus mechanism combining Proof of Work (PoW) and Practical Byzantine Fault Tolerance (PBFT), and utilizes advanced cryptographic techniques, homomorphic encryption, and NIZKPs. The proposed system is thoroughly evaluated based on multiple performance parameters, demonstrating its effectiveness and robustness in real-world healthcare settings. This research contributes to developing a more secure, efficient, and privacy-preserving healthcare data management system by addressing these gaps.

1.2. Paper Structure

The rest of this paper is structured as follows: In Section **2**, we review the existing state-of-the-art work and identify research gaps. Section **3** details the proposed architecture and its workflow. In Section **4**, we analyze the security and privacy aspects of the proposed blockchain-based framework. Section **5** presents the experimental setup, results analysis, and discussion of our findings. Finally, Section **6** concludes the paper and outlines future work directions.

2. Related Work

2.1. Privacy Preserving in Healthcare Management System

In recent literature, privacy-preserving methods in the Internet of Medical Things (IoMT) have emerged as crucial for safeguarding sensitive patient information while enhancing healthcare management efficiency [14]. Existing research extensively discusses encryption techniques, such as AES and RSA, employed to secure data at rest and in transit,

effectively preventing unauthorized access [15-16]. Another noteworthy approach is signcryption [17], which combines encryption and digital signatures to provide confidentiality and data authenticity.

Studies on access control mechanisms highlight role-based access control (RBAC) and attribute-based access control (ABAC) as effective methods to restrict data access based on user roles and attributes, ensuring that only authorized personnel can access sensitive medical information [18-19]. Data anonymization techniques such as differential privacy [20] and pseudonymization [21] are frequently cited. These methods protect patient identities by adding statistical noise to datasets and replacing identifying information with tokens.

Blockchain technology has gained significant attention in the literature for its potential to enhance security in healthcare management. It introduces a decentralized, immutable ledger for recording all medical data transactions, providing security through time-stamped, tamper-resistant records. Smart contracts, as described in several studies, automate data-sharing agreements, ensuring compliance with predefined rules and patient consent.

Zero-knowledge proofs (ZKPs) are another area of interest in current research [22]. They allow the verification of data transactions without revealing sensitive information, thereby maintaining privacy. Edge computing, which processes data locally on IoMT devices, is noted for its ability to reduce the need for central storage and minimize data breach risks.

The literature also discusses patient-controlled data access systems [23] that enable individuals to manage who can access their data and for what purposes. Continuous auditing and real-time monitoring are crucial for enhancing security by promptly detecting and responding to suspicious activities. Collectively, these methods, as reviewed in existing studies, ensure robust protection and trust in the digital healthcare ecosystem.

2.2. Literature Review

This literature review examines various blockchain-based architectures and models designed to enhance security, privacy, and efficiency in healthcare data management and sharing. One researcher details a blockchain-based telehealth architecture [24] focusing on secure storage, decentralized access control, and logging, with verified security by AVISPA, demonstrating computational efficiency and strong cryptographic keys. Another researcher introduces a triple subject purpose-based access control (TS-PBAC) model [25] for the Internet of Medical Things (IoMT), integrating blockchain for secure, privacy-preserving data sharing. This model employs a hierarchical purpose tree, local differential privacy, and mutual evaluation metrics, providing superior patient privacy protection and stable access control decisions compared to traditional methods.

The review also covers a blockchain-assisted, secure, and privacy-preserving health data-sharing scheme for edge-based IoMT, which uses a bloom filter with hash functions for keyword ciphertext verification, key-policy attribute-based encryption for profile matching, and an incentive mechanism with a two-phase Stackelberg game for optimal pricing [26]. This protocol achieves high security, scalability, and feasibility in IoMT scenarios. Additionally, a blockchain-based smart healthcare system is highlighted for privacy-preserved electronic medical record (EMR) exchange and sharing [27]. This system employs dynamic access control, local differential privacy, and multi-level smart contracts for secure, reliable, and traceable transactions, validated by experimental results with 200,000 real-world EMRs.

The review further explores a blockchain and AI-enabled model for secure medical data transmission in IoT networks called BAISMDT [28]. This model enhances data security and privacy using signcryption, blockchain, and modified particle swarm optimization with wavelet kernel extreme learning machine, achieving high accuracy in disease detection. Another blockchain-based scheme for privacy-preserving medical data-sharing balances patient privacy with research needs, employing zero-knowledge proofs and proxy re-encryption to maintain confidentiality while using PBFT for efficient distributed consensus [29-30][35].

Several papers focus on blockchain-based frameworks and models for secure and privacy-preserving healthcare data management. HealthRec-Chain combines blockchain and IPFS for secure, scalable health data storage, utilizing Java-enabled GPG encryption and a personalized Ethereum dashboard [31]. Another blockchain-based e-health system secures electronic health records (EHRs) using pairing-based cryptography and blockchain smart contracts for reliable transactions, addressing security challenges with minimal computation overhead [32]. Additionally, a CNN-based model combined with blockchain and federated learning [33] enhances electronic health record privacy, improving accuracy, data privacy, and malicious activity detection.

BCHealth, a blockchain-based architecture, enhances privacy in IoT healthcare by allowing data owners to define access policies [34]. It utilizes two chains for data and policy storage, clustering for scalability, and a modified consensus algorithm for performance. A consortium blockchain-based scheme integrates IPFS and zero-knowledge proofs to ensure security and privacy in sharing personal health records (PHRs), using attribute-based encryption and smart contracts for secure, personalized access [36].

Lastly, the review discusses various innovative schemes such as OHE-MSC [37], which leverages blockchain and homomorphic encryption for secure IoT medical data sharing, and BDSDT, which combines blockchain with deep learning for secure data transmission [38]. Lightweight data-sharing schemes integrating outsourcing attribute-based encryption and smart contracts are also examined [39]. These approaches enhance security, efficiency, and fault tolerance in IoMT data sharing, demonstrating the transformative potential of blockchain and AI technologies in healthcare management [40-43].

Existing blockchain-based healthcare systems typically focus on either enhancing data security or improving system scalability but rarely address both comprehensively. For instance, systems employing PBFT offer faster consensus but struggle with scalability, while PoW-based models prioritize security at the cost of high computational overhead. Moreover, while cryptographic techniques like FHE or NIZKPs have been explored individually, their integration into a unified framework remains underexplored. This paper bridges these gaps by combining these techniques into a cohesive system, providing both scalability and security without compromising efficiency.

2.3. Research Gaps

Despite the potential benefits of blockchain technology in healthcare, several gaps exist in current research and implementations:

- Integration with IoT Devices: While there is a significant focus on blockchain technology in healthcare, there needs to be more research on effectively integrating blockchain with IoT devices to ensure real-time, secure data acquisition and management.

- Scalability and Performance: Most existing blockchain-based healthcare applications use traditional consensus mechanisms like Proof of Work (PoW), which are inefficient in handling high transaction volumes typical in healthcare settings. There is a need to explore and implement more efficient consensus mechanisms that can enhance the scalability and performance of these systems.

- Privacy-Preserving Techniques: Current implementations often fail to adequately protect patient privacy while allowing necessary data verification and access. Techniques like Zero-Knowledge Proofs (ZKPs) and homomorphic encryption have been proposed in theory but lack practical implementation and evaluation in healthcare applications.

- Comprehensive Evaluation: Many blockchain-based healthcare solutions lack thorough experimental evaluations to test their effectiveness, robustness, and efficiency under real-world conditions. There is a need for extensive testing and analysis based on various performance metrics to demonstrate the practical viability of these solutions.

This research addresses these gaps by proposing a novel blockchain-based decentralized application that integrates IoT devices, implements a hybrid consensus mechanism combining PoW and Practical Byzantine Fault Tolerance (PBFT), and utilizes advanced cryptographic techniques such as homomorphic encryption and ZKPs. The proposed system is thoroughly evaluated based on multiple performance parameters, demonstrating its effectiveness and robustness in real-world healthcare settings. This research contributes to developing a more secure, efficient, and privacy-preserving healthcare data management system by addressing these gaps.

3. Proposed Methodology

This section proposed a decentralized application that utilizes Blockchain technology to generate and maintain medical certificates through various phases, such as data acquisition, Validation, representation, and justification. User can maintain their healthcare information using different IoT devices. The proposed applications aim to prevent fraud in maintaining and issuing user medical records, including medical test reports, discharge summaries, and operative reports. Figure 2 presents the workflow of the proposed blockchain-based IoT decentralized application for the Healthcare Management system.

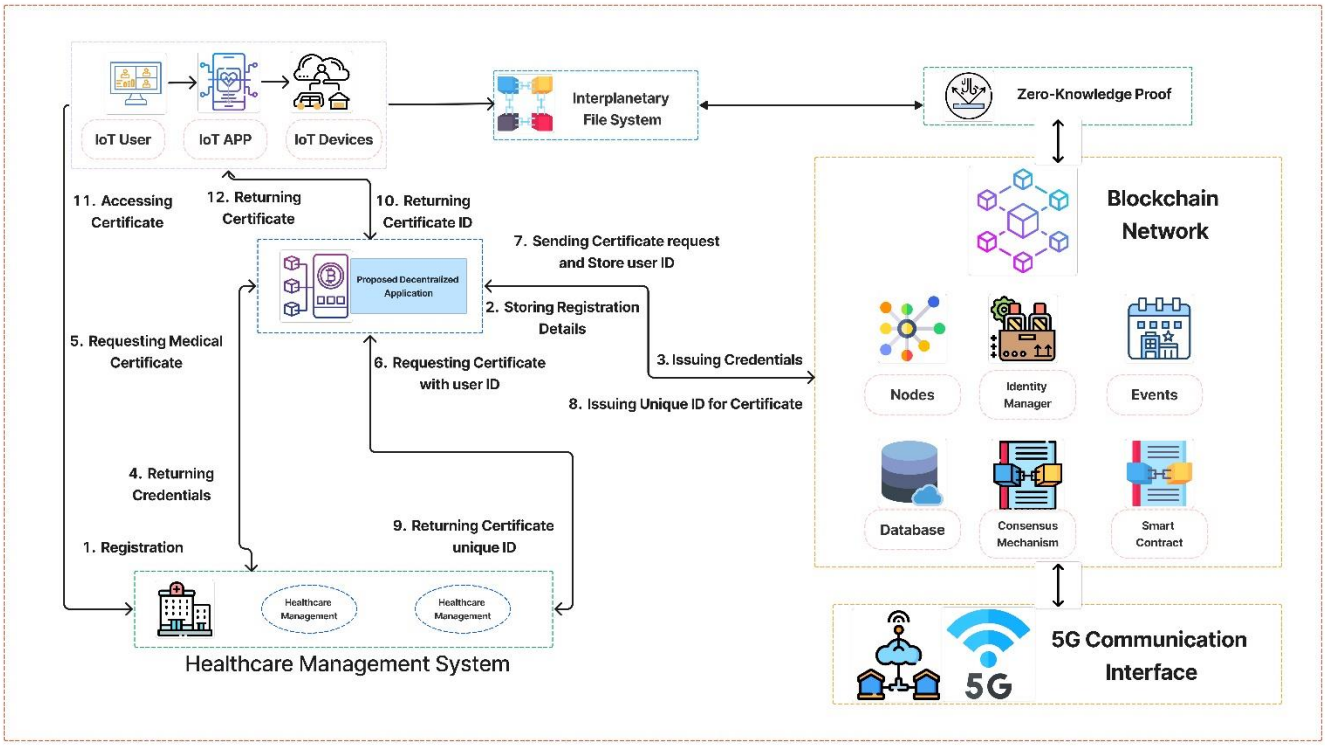


Figure 55: Proposed Workflow of the Novel Blockchain-Based IoT Application for Healthcare Management System

The diagram presents a decentralized healthcare management system that leverages blockchain technology, Non-Interactive Zero-Knowledge Proofs (NIZKPs), and the Interplanetary File System (IPFS) to enhance the privacy and security of medical certificate management. It delineates the interactions among IoT users, devices, and the healthcare management system, which collectively facilitate secure access and validation of medical data. On the left, the IoT user engages with the healthcare system through an IoT app and connected devices that transmit user data. Conversely, the right side illustrates the blockchain network's critical role in ensuring data integrity and security through nodes, identity managers, and consensus mechanisms. Smart contracts automate certificate issuance and verification, while ZKP ensures data verification without exposing sensitive information.

The workflow initiates with user registration (Step 1) and the issuance of unique credentials (Step 3), secured by the blockchain's identity manager. Upon requesting a medical certificate (Step 5), the healthcare system returns credentials (Step 4), enabling the user to present them to the decentralized application (Step 6). The application verifies identity and processes the request (Step 7), leading to the issuance of a unique certificate ID (Step 8). This ID links to the medical certificate stored in IPFS, from which the user retrieves the certificate (Steps 9 and 10). The system's architecture, supported by a 5G communication interface, facilitates real-time interactions among its components, ensuring a robust, secure, and efficient healthcare management solution.

3.1. Architecture and Workflow

The proposed blockchain-based application is designed to operate within a distributed network that seamlessly integrates blockchain technology with Internet of Things (IoT) devices to enhance healthcare management. The blockchain network is at the heart of the system, which provides a secure and immutable ledger for managing healthcare data.

User Registration and Verification is the first step in the process. Hospitals, patients, doctors, and IoT devices are registered using a smart contract on the Ethereum blockchain. The hospital authority manages this registration, issuing credentials to each entity. For hospitals and doctors, the registration involves sending a request through the user portal, which the blockchain network verifies. Registration requests are also processed for patients, generating credentials that allow them to access application services. IoT devices, such as smartwatches, are similarly registered by transmitting a request containing device-specific details like Device ID. The blockchain verifies and stores these credentials, associating each device with its user profile.

Integration of IoT Devices plays a critical role in the system. IoT devices collect real-time health data from users, such as heart rate and activity levels. This data is securely transmitted to the blockchain network, where it is encrypted using advanced techniques to ensure privacy. Each data point is assigned a unique identifier on the blockchain, facilitating secure and immutable storage. This integration ensures that health metrics are recorded accurately and are protected against tampering or unauthorized access.

When patients seek medical services, they submit a Service Request through the application. The hospital authority then verifies the request and the patient’s data via the blockchain network, ensuring that all interactions are secure and authenticated. Once verification is complete, the hospital can issue a medical certificate. The application processes this certificate, which assigns a unique identification number and records it on the blockchain. This ensures the certificate’s authenticity and provides a tamper-proof record that authorized parties can easily access. Certificate Access is streamlined through the use of unique identifiers. Both users and hospitals can retrieve the generated medical certificate using this unique ID. The blockchain’s distributed ledger ensures that access to this information is secure and controlled, reducing the risk of unauthorized access or modifications.

The application integrates a hybrid consensus mechanism combining Proof of Work (PoW) and Practical Byzantine Fault Tolerance (PBFT) to secure the network. The PoW component requires computational work to validate and record transactions, enhancing data integrity and preventing unauthorized modifications. Simultaneously, PBFT ensures consensus among network participants by providing robustness against faulty or malicious nodes, improving overall system reliability and efficiency. This hybrid approach leverages the strengths of both algorithms to maintain a secure, resilient, and efficient blockchain network.

The system features a Graphical User Interface (GUI) that facilitates interaction with the blockchain network. This intuitive interface allows hospitals, patients, and doctors to quickly manage and access their data, enhancing user experience and operational efficiency. Finally, the Distributed Ledger Management ensures that all healthcare data is protected from unauthorized access. The blockchain supports secure operations such as data insertion, deletion, and updates, maintaining the confidentiality and integrity of sensitive medical information. Figure 3 represents a decentralized healthcare system using blockchain and IoT for secure medical certificate management. It includes key components like IoT devices, consensus mechanisms, homomorphic encryption, IPFS, NIZKP, and IDS for secure communication, certificate validation, and real-time threat detection.

By integrating blockchain technology with IoT devices, the application creates a robust, secure, and efficient system for managing medical records and certificates. This innovative approach enhances healthcare data management's accuracy, security, and transparency, addressing many challenges associated with traditional centralized systems.

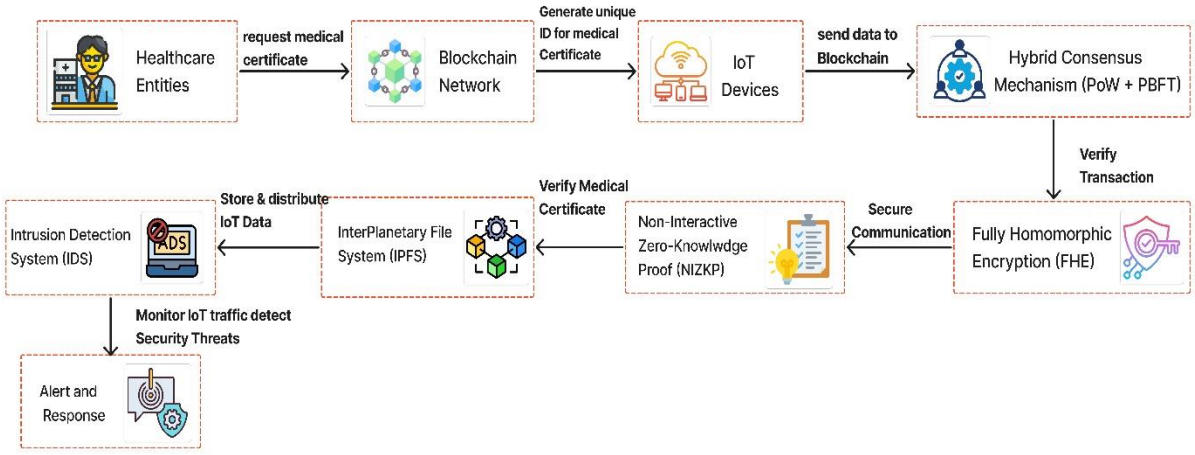


Figure 56: Workflow of a Blockchain-Based IoT Healthcare Application with Homomorphic Encryption, Zero-Knowledge Proofs, and Intrusion Detection for Secure Medical Certificate Management

3.2. Integration of PoW and PBFT

The PoW consensus algorithm creates new blocks in the blockchain network, ensuring that blocks are added securely and consistently. Each block contains a hash of the previous block, a timestamp, and transaction data. The goal is to find a nonce n Such data that the hash of the block denoted as H, meets a specific target T:

$$H = hash(prev_{hash}, timestamp, transaction, n)$$

$$H < T$$

Minor repeatedly changes the nonce n and recompute the hash until they find a valid one. Following are the steps in the proposed model:

Step 1: When a new medical certificate is generated, it forms a new term transaction.

Step 2: Miners in the blockchain network collect these transactions and form a new block.

Step 3: They compute the hash with different nonce values until they find a valid one that meets the target T.

Step 4: The first miner to find a valid nonce broadcasts the block to the network

Step 5: Other nodes verify the block and add it to their blockchain network.

The PBFT is used to enhance the security and speed of the consensus process, ensuring that the system can handle high transaction volume. It contains five phases:

Phase 1: Initialization Phase

A client (e.g. a hospital) sends a request to the primary node (leader) to execute a service operation. This can be represented as:

$$Request = (operation, Timestamp, Client\ ID)$$

$$Client \rightarrow Primary : \{Request\}$$

Phase 2: Pre-prepare Phase

The primary note receives the client request and multicasts a pre-prepare Message to all replicas. The pre-prepared message is denoted as:

$$\{pre - prepare, V, n, d\}$$

$$Primary \rightarrow Replicas : \{pre - prepare, V, n, d\}$$

Where V is the view number, N is the sequence number assigned to the request, and d is the digest (hash) of the request completed as d= hash (request).

Phase 3: Prepare Phase

Each replica verifies the pre-prepared message. If valid the replica multicasts a prepare message to all other replicas.

$$\{prepare, V, n, d, i\}$$

$$\forall\ Replica\ i : \{prepare, V, n, d, i\}$$

Where i is the replica ID.

Each replica waits to receive $2f + 1$ prepare message (including its own), where f is the maximum number of faulty replicas the system can tolerate.

$$Wait\ for\ 2f + 1\ prepare\ message$$

Phase 4: Commit Phase

Once a replica has received $2f + 1$, prepare message, it multicasts a commit message as:

$$\{Commit, V, n, d, i\}$$

$$\forall\ Replica\ i : \{Commit, V, n, d, i\}$$

Each replica waits to receive $2f + 1$ commit message (including its own).

$$Wait\ for\ 2f + 1\ commit\ message$$

Phase 5: Reply Phase

When a replica has received $2f + 1$ commit message, it executes the requested operations and sends a reply to the client:

$$Reply = (Result, Timestamp, Client\ ID)$$

$$Replica \rightarrow Client : (Reply)$$

Following are the steps in the proposed model:

Step 6: When a hospital requests to add a new medical certificate, the request is processed through the PBFT consensus.

Step 7: The primary node initiates the PBFT process.

Step 8: Replicas go through pre-prepare, and commit phases to agree on adding the new block.

Step 9: The request is executed, and the result (new block) is added to the blockchain.

Considering a scenario where patient P requests a medical certificate from the hospital H. The PBFT consensus ensures that this request is processed securely and efficiently. Patient P sends a request R to the primary node (hospital H): $P \rightarrow H: R$. Hospital H (primary node) assigns a sequence number and creates a pre-prepared message as:

$$d = hash(R)$$

Hospital H broadcasts the pre-prepared message to all replica nodes (other hospitals/doctors):

$$H \rightarrow Replicas: (Pre - prepare, V, n, d)$$

Replica nodes verify the pre-prepared message and broadcast the prepared message:

$$Replica \rightarrow Replicas: (prepare, V, n, d, i)$$

The replica node counts the prepare message, and upon receiving f+1 prepares, they broadcast the commit message:

$$Replica \rightarrow Replicas: (commit, V, n, d, i)$$

The replica node counts the commit message, upon receiving the 2f+1 commit, they execute the transactions and send the result to the patient:

$$Replica \rightarrow P: (Medical\ Certificate)$$

The proposed blockchain-based healthcare system can achieve higher security and faster transaction speeds by enhancing the consensus mechanism. Figure 4 illustrates the workflow of the Practical Byzantine Fault Tolerance (PBFT) mechanism, detailing the interaction phases between the client and nodes during transaction processing and consensus. The algorithm 1 outlines a consensus process involving request, pre-prepare, prepare, commit, and reply phases, ensuring transaction validity and execution across distributed nodes.

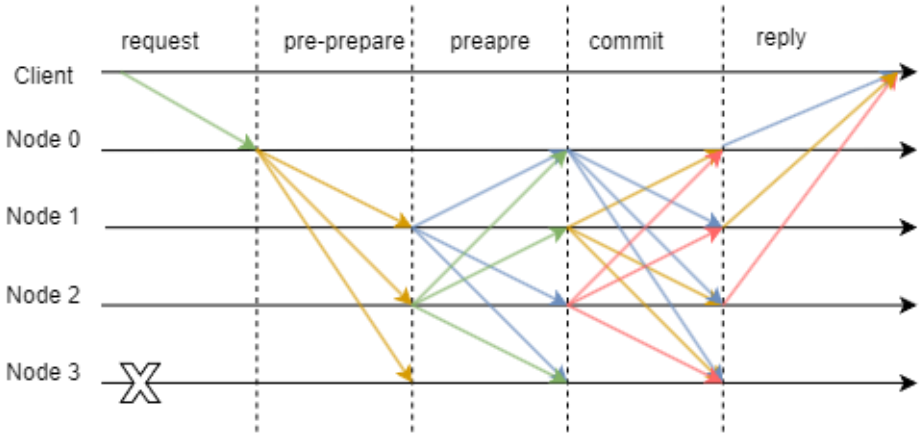


Figure 57: Workflow of PBFT mechanism in the proposed framework at different phases

Algorithm1: Working of Consensus Mechanism in the proposed application
Algorithm 1.1.: Request Phase
Input: Client request R
Output: Transaction initiation

1. Client sends request R to the primary node: $Client \rightarrow Primary : \{Request\}$

Algorithm 1.2.: Pre-Prepare Phase
Input: Request R
Output: Pre-prepare message

1. Primary node assigns a sequence number n and creates a pre-prepare message:$(Pre-prepare, V, n, d)$, where V is the view number, and d is the digest of R: $d = hash(R)$
2. Primary node broadcasts the pre-prepare message to all replica nodes: $Primary \rightarrow Replicas: (Pre-prepare, V, n, d)$

Algorithm 1.3.: Prepare Phase
Input: Pre-prepare message $(Pre-prepare, V, n, d)$
Output: Prepare message

1. Replica nodes verify the pre-prepare message: if $(Pre-prepare, V, n, d)$ is valid
2. Replica nodes create and broadcast prepare messages: $(prepare, V, n, d, i)$, where i is the node identifier.
3. Replica nodes send prepare messages to all other nodes: $Replica \rightarrow Replicas: (prepare, V, n, d, i)$

Algorithm 1.4.: Commit Phase
Input: Prepare messages $(prepare, V, n, d, i)$
Output: Commit message

1. Replica nodes count the prepare messages. If $f + 1$ prepares are received (where f is the maximum number of faulty nodes), they create and broadcast commit messages: $(commit, V, n, d, i)$
2. Replica nodes send commit messages to all other nodes: $Replica \rightarrow Replicas: (commit, V, n, d, i)$

Algorithm 1.5.: Reply Phase
Input: Commit messages $(commit, V, n, d, i)$
Output: Transaction execution and client reply

1. Replica nodes count the commit messages. If $2f + 1$ commits are received, they execute the transaction: $execute(R)$
2. Replica nodes send the result to the client: $Replica \rightarrow Client : (Reply(R))$

3.3. Fully Homomorphic Encryption (FHE) for Data Privacy

Fully Homomorphic Encryption (FHE) is a powerful cryptographic method that enables computations on encrypted data without decryption, supporting addition and multiplication operations. This capability makes FHE more suited to complex medical scenarios than semi-homomorphic techniques like Paillier, which are limited to addition operations. FHE protects sensitive medical data in the proposed healthcare blockchain architecture, allowing comprehensive analyses and computations. This sub-section explains the utilization of fully homomorphic encryption and illustrates an example of how FHE works in the proposed architecture.

Encryption and Decryption process in FHE:

In FHE, plain text m is encrypted into cipher text C using an encrypted function E:

$$C = E(m)$$

FHE allows both addition and multiplication operations on cipher texts, making it highly flexible for complex computations:

- Addition: $E(m_1).E(m_2) mod\ n^2 = E(m_1 + m_2)$
- Multiplication: $E(m_1)^{m2}\ mod\ n^2 = E(m_1 * m_2)$

The decryption process using a decryption function D, retrieves the original plain text m from cipher text C:

$$m = D(C)$$

Example: Computation of Total Medical Expenses using FHE:

Consider a scenario where a hospital must calculate a patient's total medical expenses while keeping individual expenses confidential. With FHE, such computations can be carried out on encrypted data without compromising privacy.

Key Generation:

- Two large prime number $p = 31\ and\ q = 37$ are chosen.
- Compute $n = p * q = 31 * 37 = 1147$.

- Public and private keys are generated to support addition and multiplication operations on encrypted data using $\lambda = lcm\,(p-1, q-1), g \in Z_{n^2}^*, \mu = (L(g^{\lambda}\ mod\ n^2))^{-1} \text{mod n}$, Where $L(x) = \frac{x-1}{n}$

Encryption: Two medical expenses, $m_1 = 200\ and\ m_2 = 150$, are encrypted using random values $r_1 = 5\ and\ r_2 = 7$.

$$C_1 = E(m_1) = g^{m_1} * r_1^n\ mod\ n^2$$

$$C_2 = E(m_2) = g^{m_2} * r_2^n\ mod\ n^2$$

Homomorphic Operations: Using FHE, both addition and multiplication can be performed on the encrypted data without requiring decryption:

$$C_{sum} = C_1.C_2\ mod\ n^2 = E(m_1 + m_2)$$

$$C_{product} = C_1^{m2}\ mod\ n^2 = E(m_1 * m_2)$$

Decryption: After performing the operations, the ciphertext is decrypted using the private key to reveal the results:

$$m_{sum} = D(C_{sum}) = m_1 + m_2 = 350$$

$$m_{product} = D(C_{product}) = m_1 * m_2 = 30{,}000$$

This example illustrates the versatility of FHE, allowing for complex computations beyond basic addition, which is crucial in scenarios that require secure financial calculations or statistical analysis of sensitive medical data. By utilizing FHE, the proposed healthcare blockchain system facilitates privacy-preserving computations on encrypted data, allowing advanced medical operations without revealing sensitive information. FHE enhances system security by supporting secure computations, such as billing, patient statistics, and resource management, within a decentralized architecture. This ensures that patient data remains confidential even in complex medical scenarios while enabling more flexible computations than semi-homomorphic encryption methods can provide.

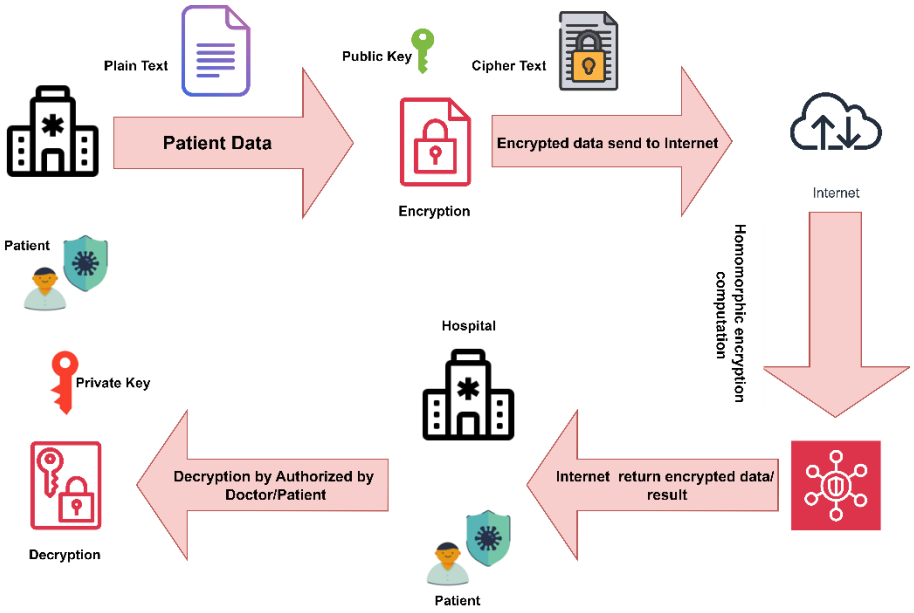


Figure 58: Workflow of Homomorphic Encryption in the Proposed Framework

Implementing the FHE for data privacy in the healthcare blockchain system ensures the privacy of sensitive medical information while enabling secure computation on encrypted data. This approach enhances security and maintains patient confidentiality throughout the data processing phase, as shown in Figure 5. The above example demonstrates how Homomorphic encryption is utilized in the proposed architecture, enabling secure and private computation. The algorithm 2 describes a cryptographic system's key generation, encryption, decryption, and homomorphic addition, enabling secure and privacy-preserving computations on encrypted data.

Algorithm 2: Homomorphic Encryption process in the proposed application
Algorithm 2.1.: Key Generation
Input: Two large prime numbers: p *and* q
Output: Public key (n, g) and a private key is (λ, μ).

1. Choose p *and* q
2. Compute $n = pq$
3. Compute $\lambda = \text{lcm}\,(p-1, q-1)$
4. Select a random $g \in Z_{n^2}^{x}$
5. Compute $\mu = (L(g^{\lambda} \bmod n^2))^{-1} \bmod n$

Algorithm 2.2.: Encryption
Input: plaintext m, Public key (n, g).
Output: ciphertext C.

1. Choose a random $r \in Z_n^x$.
2. Compute $C\_ = g^{m_1}.r^n \bmod n^2$

Algorithm 2.3.: Decryption
Input: Ciphertext C, private key is (λ, μ).
Output: Plaintext m.

1. Compute $m_{sum} = L\,(C_{sum}^{\lambda} \bmod n^2).\mu \bmod n$

Algorithm 2.4.: Homomorp Operation
Input: Ciphertexts C_1 and C_2.
Output: Ciphertext C_{sum}, $C_{product}$

1. Compute $C_{sum} = C_1.C_2 \bmod n^2$
2. Compute $C_{product} = C_1^{m2} \bmod n^2$

3.4. Non-interactive Zero-Knowledge Proofs (NIZKPs) for Medical Certificate Verification

Zero-knowledge proofs (ZKPs) are cryptographic protocols that allow one party (the prover) to prove the validity of a statement to another party (the verifier) without revealing any underlying information. When verifying medical certificates, ZKPs can confirm the certificate's authenticity without disclosing sensitive patient data.
In the previous interactive ZKP protocol, the prover and verifier had to be online simultaneously, exchanging messages to complete the verification process. However, this is not practical for blockchain-based systems, as requiring multiple parties to be online simultaneously is unrealistic. To address this, we adopt Non-Interactive Zero-Knowledge Proofs (NIZKPs) using the Fiat-Shamir heuristic, eliminating the need for real-time interaction between the prover and verifier.
The following steps describe the NIZKP-based verification of medical certificates in a blockchain-based decentralized healthcare system:

Step 1: Certificate Issuance
The hospital (H) issues a medical certificate to a patient. This certificate contains encrypted patient data and a unique identifier.

1. Encrypted patient data: The patient's data P_d is encrypted using the hospital's public key $K_{H_{pub}}$: $E(P_d) = encrypt\ (P_d, K_{H_{pub}})$
2. Generate Unique Identifier: A unique identifier U_1D is generated for the medical certificate using a hash function that combines the patient's data and the timestamp: $U_1D = hash\ (P_d\ ||\ timestamp)$
3. Create the medical certificate: The medical certificate C is created, which contains the encrypted patient data, unique identifier, and the hospital's signature: $C = \{U_1D, E(P_d), signature\ \left(E(P_d), K_{H_{pub}}\right)\}$
4. Store the certificate in the blockchain: The medical certificate C is stored immutably on the blockchain.

Step 2: Prover and Verifier Setup
In a non-interactive setup, the prover (hospital) generates proof that can be verified anytime without requiring real-time communication with the verifier.

1. Public Parameters: The prover selects public parameters g and p, where g is a generator and p is a large prime number.
2. Statement and Secret: The prover (hospital) and verifier agree on a public statement S, which is based on a secret s (known only to the prover):
$$S = g^s\ mod\ p$$
3. Generates Commitment: The prover generates a random value r and computes a commitment C:
$$C = g^r\ mod\ p$$

Step 3: NIZKP Protocol
Instead of the verifier sending a challenge in an interactive protocol, the challenge is generated using a cryptographic hash function, making the proof non-interactive.

1. Generate the Challenge: The prover computes the challenge c as a hash of the public statement S and commitment C: $C = hash(S, C)$
2. Compute the response: The prover computes the response z: $z = r + c.s\ mod\ (p - 1)$
3. Store the proof on the blockchain: The prover stores the commitment C, the challenge c, and the response z on the blockchain, allowing any verifier to verify the proof later without requiring the prover to be online.

Step 4: Proof Verification
The verifier (or any participant on the blockchain) can verify the authenticity of the medical certificate without requiring real-time interaction with the prover. The verifier checks the following conditions:
$$g^r = C.S^c\ mod\ p$$
If the equation holds, the verification is booming, and the medical certificate is deemed authentic.

By integrating NIZKPs into the proposed blockchain-based decentralized application, we ensure privacy and practicality in verifying medical certificates, as the verifier can simultaneously be offline as the prover. This adaptation

maintains the privacy-preserving nature of NIZKPs while making the system more suitable for decentralized environments like blockchain. The algorithm 3 describes issuing a medical certificate using encryption, generating and verifying non-interactive zero-knowledge proofs (NIZKPs) to ensure secure and verifiable certificate issuance and validation.

Algorithm 3: NIZKP mechanism in the proposed model for medical certificate verification
Algorithm 3.1.: Certificate Issuance
Input: Patient data P_d, Hospital's private key $K_{H_{pub}}$
Output: Medical certificate C, Certificate's unique identifier U_1D

1. Encrypt patient data using the hospital's public key: $E(P_d) = encrypt\ \left(P_d, K_{H_{pub}}\right)$
2. Generate a unique identifier for the certificate: $U_1D = hash\ (P_d \mid\mid timestamp)$
3. Create the certificate: $C = \left\{U_1D, E(P_d), signature\ \left(E(P_d), K_{H_{pub}}\right)\right\}$
4. Store C in the blockchain and provide U_1D to the patient.

Algorithm 3.2.: Prover and Verifier Setup
Input: Public parameters g, p Hospital's private key $K_{H_{pub}}$
Output: Commitment C, Challenge c, Response z

5. The prover (hospital) and verifier agree on a public statement S and secret s:
$$S = g^s \ mod\ p$$
6. The prover generates a random value r and computes the commitment C:
$$C = g^r \ mod\ p$$

Algorithm 3.3.: NIZKP Protocol Execution
Input: Commitment C, Challenge c, Response z
Output: Verification result (true/false)

7. **Commit Phase:**
8. Prover computes the commitment C: $C = g^r \ mod\ p$
9. The prover sends C to the verifier.
10. **Challenge Phase:**
    - Verifier sends a random challenge ccc to the prover.
11. **Response Phase:**
    - Prover computes the response z using the secret s and the challenge c:
    $$z = r + c.s \ mod\ (p-1)$$
    - Prover sends z to the verifier.
12. **Verification Phase:**
    - Verifier checks the validity of the response:
    $$g^r = C.S^c \ mod\ p$$
    - If the equation holds, the verifier accepts the proof; otherwise, it rejects the proof.

3.4.1. Implementation of Non-Interactive Zero-Knowledge Proofs (NIZKPs)

To ensure the practical realization of the Non-Interactive Zero-Knowledge Proof (NIZKP) protocol in our proposed blockchain-based decentralized healthcare system, we implemented the NIZKP protocol using a combination of smart contracts and off-chain computation.

A. Smart Contract-Based Implementation

The verification process for medical certificates and the cryptographic proof generation were implemented using Ethereum smart contracts. These smart contracts handle the interaction between the prover (hospital) and the verifier (employers, regulatory bodies, etc.) while maintaining the privacy of patient data. Below is a brief description of the smart contract structure and key functions:

1. NIZKP Prover Contract

The NIZKP Prover Contract is responsible for generating and storing the medical certificate along with its cryptographic commitment on the blockchain. The commitment is computed using the unique

identifier, encrypted medical data, and a digital signature. This ensures the integrity and authenticity of the stored certificate.

```
// NIZKP Prover Contract
contract NIZKPProver {
    struct Certificate {
        bytes32 uniqueID;
        bytes encryptedData;
        bytes signature;
        bytes32 commitment; // Commitment hash
    }

    mapping(address => Certificate) public certificates;

    // Function to generate cryptographic commitment
    function generateCommitment(bytes32 _uniqueID, bytes memory _encryptedData, bytes memory _signature)
        internal pure returns (bytes32)
    {
        return keccak256(abi.encodePacked(_uniqueID, _encryptedData, _signature));
    }

    // Function to store medical certificate with commitment
    function storeCertificate(bytes32 _uniqueID, bytes memory _encryptedData, bytes memory _signature) public {
        bytes32 commitment = generateCommitment(_uniqueID, _encryptedData, _signature);
        certificates[msg.sender] = Certificate(_uniqueID, _encryptedData, _signature, commitment);
    }

    // Function to retrieve the stored certificate and commitment
    function getCertificate(address user) public view returns (bytes32, bytes memory, bytes memory, bytes32) {
        Certificate memory cert = certificates[user];
        return (cert.uniqueID, cert.encryptedData, cert.signature, cert.commitment);
    }
}
```

2. **NIZKP Verifier Contract**

   This contract allows verifiers to check the certificate's authenticity using the NIZKP protocol. The contract computes the proof and verifies the response without revealing the patient's private data.

```
contract NIZKPVerifier {
    uint256 public g;
    uint256 public p;

    constructor(uint256 _g, uint256 _p) {
        g = _g;
        p = _p;
    }

    // Function to verify the NIZKP proof on-chain
    function verifyProof(uint256 z, uint256 C, uint256 S, uint256 c) public view returns (bool) {
        return g**z % p == C * S**c % p;
    }
}
```

B. Off-Chain Computation

While storing the certificates and proof verification process is handled on-chain, the cryptographic computations (such as generating the random values and challenge responses) are conducted off-chain. This hybrid approach ensures scalability and efficiency by reducing the gas costs of heavy computations on the Ethereum network.

1. **Off-Chain Challenge and Response Computation:** The challenge c and response z are computed off-chain using a Python script, which interacts with the smart contract to store and verify the commitment C and proof response z.

```
from web3 import Web3
import hashlib

def compute_challenge(S, C):
    # Use SHA-256 hash function to generate a non-interactive challenge
    challenge = hashlib.sha256((str(S) + str(C)).encode()).hexdigest()
    return int(challenge, 16)  # Convert the hash to an integer for further computation

def compute_response(r, c, s, p):
    # Compute z = r + c * s (mod p-1)
    return (r + c * s) % (p - 1)
```

i. **Prover (Hospital) Process:**
   - The hospital encrypts the patient's medical data off-chain. It generates the cryptographic proof (including the unique identifier, encrypted data, and signature) and stores it on-chain using the ZKP Prover smart contract.
   - Off-chain computation is performed to generate the commitment C, the public statement S, and the challenge-response pair (c,z), which are later verified on-chain.

ii. **Verifier Process:**
   - The verifier accesses the smart contract to retrieve the cryptographic proof.
   - The challenge c is generated off-chain, and the final proof z is computed off-chain. This challenge-response pair is sent back to the smart contract to verify the authenticity of the medical certificate using the NIZKP Verifier contract.

By implementing both on-chain and off-chain components, we ensure an efficient, secure, and practical system that adheres to the privacy-preserving nature of ZKPs without overloading the blockchain with computational tasks. This implementation, combining on-chain smart contracts and off-chain cryptographic computations, ensures that the proposed NIZKP-based medical certificate verification solution is feasible and practical in real-world blockchain environments.

3.5. Proposed Intrusion Detection System

To detect anomalies within the Internet of Medical Things (IoMT) environment, we propose utilizing machine learning (ML) models to classify network traffic as normal or malicious. These models are trained and evaluated using the WUSTL EHMS 2020 Dataset, which contains labeled instances of IoMT traffic, including normal traffic and various attacks. The proposed models include Random Forest (RF), XGBoost, and Support Vector Machines (SVM). These models are selected for their robustness in high-dimensional spaces and capacity to handle imbalanced data. Figure 6 presents the workflow of an Intrusion Detection System (IDS) applied to the WUSTL EHMS 2020 Dataset. It shows steps from data pre-processing, feature extraction, model training, deployment, real-time monitoring, anomaly detection, and alert generation to prediction.

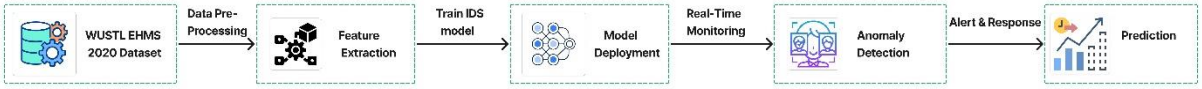


Figure 59: Workflow of Intrusion Detection System (IDS) Model on WUSTL EHMS 2020 Dataset for Anomaly Detection and Prediction.

The Random Forest (RF) model was selected for its ability to handle high-dimensional datasets, such as IoMT traffic, without overfitting. RF constructs multiple decision trees during the training phase and aggregates their outputs to make the final classification. This method is particularly suited to large-scale IoMT environments, as it can effectively handle numerous network-based features, including packet size, protocol types, and device metadata. The RF model was tuned using grid search, optimizing hyperparameters such as the number of trees, the maximum depth of the trees, and the minimum samples per split. XGBoost, a gradient-boosting algorithm, was also employed due to its capacity

for handling imbalanced datasets and its computational efficiency. XGBoost improves upon traditional decision tree models by using an iterative process to minimize prediction errors, making it highly suitable for detecting a wide range of attacks in IoMT traffic. By tuning hyperparameters such as the learning rate, maximum depth, and number of estimators, XGBoost achieved the best performance among the models. Lastly, Support Vector Machines (SVM) were implemented to classify network traffic using a hyperplane that maximizes the margin between normal and malicious traffic. SVM is particularly effective in high-dimensional spaces, characteristic of IoMT data with multiple network-based features. An RBF (Radial Basis Function) kernel was used to manage the non-linear nature of IoMT traffic, and the hyperparameters were tuned to optimize model performance. Table 1 shows the Hyperparameter table of the proposed intrusion detection models.

Table 37: Hyperparameter Configurations for Proposed Intrusion Detection Models

Model	Hyperparameter	Values
Random Forest (RF)	n_estimators	100
	max_depth	20
	min_samples_split	2
XGBoost	learning_rate	0.1
	max_depth	15
	n_estimators	150
	subsample	0.8
	colsample_bytree	0.8
Support Vector Machine (SVM)	C	1
	gamma	0.1
	kernel	RBF

4. Security and Privacy Analysis of the Proposed Method

This section presents the formal security and privacy verification for the proposed framework.

4.1. Security Analysis

- **Impersonation Attack:** In our novel blockchain-based IoT healthcare application, we address impersonation attacks by ensuring that an attacker cannot masquerade as a legitimate entity. The attacker must provide the verifier with a temporary credential and Identification and a sensor MAC address to obtain a provisional key. A timestamp is generated for the request of Identification, and the Verifier checks the existing credential against its records and verifies. If the timestamp verification fails, the process is immediately terminated. Zero-knowledge proof (ZKP) verification is conducted for identity creation, ensuring the framework is secure from impersonation attacks.
- **Insider Attack:** The framework also protects against insider attacks. An insider attacker, even with necessary credential information such as temporary credentials or identification and sensor MAC address, cannot compute the actual identity due to the timestamp verification over the ZKP prover and challenge-response protocols. This mechanism prevents insider attacks effectively.
- **MITM and Replay Attack:** To counter Man-in-the-Middle (MITM) and replay attacks, the proposed framework ensures that an attacker who intercepts messages from an insecure channel cannot exploit this information. The attacker must use brute-force techniques to compute the correct timestamp to execute these attacks. However, the ZKP process used to verify the timestamp is computationally intensive and challenging to predict accurately. Thus, the framework is safeguarded against MITM and replay attacks.
- **Preservation of untraceability and Anonymity Properties:** The system maintains untraceability and anonymity by using freshly generated timestamps and random secret values in all communicated messages. This ensures different messages in each session and prevents message tracing. Additionally, temporary identities are used instead of real ones and updated in each session, preserving anonymity.
- **Mitigation of Synchronization Problems and Associated Attacks:** Our model ensures synchronization by exchanging messages containing timestamps and verifying updated values of temporary identities. This mechanism prevents synchronization issues and associated attacks. All heterogeneous devices agree on a maximum transmission delay to maintain synchronization.

- **Prevention of Stolen Verifier Attacks:** The system stores registration information in secure memory and does not directly store sensitive information. This approach prevents adversaries from launching attacks such as password guessing, impersonation, and unauthorized session key computation.

4.2. Privacy Analysis

The proposed framework also includes robust privacy-preserving mechanisms. An attacker attempting to access or modify information using short signature techniques from a secure channel will face significant challenges. Two major approaches ensure privacy preservation:

- **ZKP Verification for Identity Creation:** The Identity is created based on the ZKP verification scheme using the prover and verifier challenge-response protocols. The signature is verified during block creation in the blockchain network, and the associated Identities are stored as a transaction hash. Modifying this information is nearly impossible, as altering one block hash requires altering all subsequent blocks in the chain.

- **Blockchain Integrity:** Information is stored in blocks linked in a chain, making it resistant to tampering. The integrity and immutability of blockchain technology ensure that modifying one block would necessitate altering all subsequent blocks, thus preserving the privacy and integrity of medical data.

5. Experimental Setup and Result Analysis

This section discusses the experimental setup, result analysis, and findings based on the experimental testing conducted in the context of blockchain-based architectures in healthcare data management.

5.1. Experimental Setup

The proposed framework uses Ethereum, an open-source blockchain network, to develop smart contracts supporting all system functions. It enables users and patients to register, verify, and access healthcare certificates through smart contract functionalities. We utilized the Ganache tool to set up the blockchain network, deploy smart contracts, and execute performance evaluations on latency, processing time, throughput, and computation time. The framework's portal, designed with React Native for compatibility with Ethereum, connects via Node.js. Solidity is used to develop smart contracts, with Remix IDE for writing and deploying these contracts on the test network and transaction fees managed through Ethers. Metamask, a browser plugin, and wallet, generated Ethereum addresses and facilitated transactions. The experimental setup is deployed on a laptop featuring an Intel Core i5 10th Gen processor, 8GB RAM, 512GB ROM, Windows 11 OS, and an NVIDIA GTX 1650 GDDR6 4GB graphics card, with TestRPC and SQL installed to support the deployment.

5.2. Result Analysis

The Etherscan tool is utilized to evaluate the operational costs of our proposed blockchain-based application. Etherscan, an analytics tool for exploring blocks in the blockchain network, functions as an Ethereum platform Gas tracker. It tracks transactions, verifies the performance of smart contracts, and checks the process’s state. Transactions on the Ethereum blockchain network require Gas, representing the cost needed to perform a function within the network. Miners set the Gas price based on supply and demand, with the cost depending on the execution, deployment, and transfer processes involved in transactions. Gas has two main parameters: limit and price, where the limit depends on the user's willingness to execute a transaction and is presented as ‘gwei’.

Our proposed application deploys smart contracts on TestRPC and collects details of all executed tasks using the Etherscan tool. Table 2 presents the operational costs evaluated by Etherscan. Etherscan provides detailed analytics, tracking all transactions, analyzing Gas costs, evaluating smart contract efficiency, and validating transaction states.

The proposed application incurs Gas costs for executing each transaction on the Ethereum public blockchain network, with Gas measuring the cost of executing operations in a transaction. Miners set the Gas price based on demand and supply. Deploying smart contract functions and transactions on the proposed blockchain network requires significant computational power. Table 1 presents a detailed analysis of the deployment costs for key smart contract functions within our proposed blockchain framework. This table aims to elucidate each function's gas consumption, transaction fees, and overall efficiency, providing insights into the resource demands and economic implications of executing these smart contracts.

The table 2 includes four main functions: Registration, Block Generation, Issuing Certificates, and Verifying Certificates. Each function gas usage is listed in Gwei, a measure of computational effort required by the Ethereum Virtual Machine (EVM). For example, the Registration function consumes 34,219 Gwei, reflecting the substantial resources needed for initial setup and data storage. In comparison, Block Generation uses 37,127 Gwei, the highest among the functions due to the complex process of creating a new block. Issuing Certificates and Verifying Certificates use 29,120 Gwei and 37,100 Gwei, respectively, indicating the computational demands associated with certificate management and verification.

Transaction fees, calculated from the gas used and gas price, provide a clear picture of the economic cost of executing each function. The Registration function incurs a fee of 0.000036 ETH, which is relatively low, highlighting the efficient cost of initial data registration. With its higher complexity, block generation results in the highest fee of 0.000048 ETH. Issuing Certificates has a lower fee of 0.000028 ETH, while Verifying Certificates costs 0.000040 ETH, placing it between the issuance and generation costs. The average gas price for each function is also included, showing variations in transaction costs based on network conditions. The Registration function has an average gas price of 15 Gwei, while Block Generation has a higher price of 20 Gwei, reflecting increased competition and resource demands. Issuing certificates have an average price of 12 Gwei, and verifying certificates at 18 Gwei indicates different levels of cost efficiency.

The table 2 also provides details such as transaction hash, block size, transaction nonce, and transaction index. The transaction hash ensures traceability, allowing for verification of transaction details on the blockchain. Block size varies across functions, with Registration producing a larger block size of 9,523 bytes, while Block Generation results in a smaller block size of 479. This variation reflects the data processing requirements of each function. The transaction nonce and index provide information on transaction ordering and sequencing within the block, which is crucial for understanding transaction processing and potential conflicts. Efficiency ratios, calculated as gas usage per transaction, offer insights into the cost-effectiveness of each function. The Registration function has an efficiency ratio of 1.5, Block Generation is 1.8, Issuing Certificates is 1.2, and Verifying Certificates is 1.6. These ratios illustrate the relative gas efficiency of each function, with Issuing Certificates demonstrating the highest efficiency.

In summary, Table 2 offers a comprehensive view of our blockchain system's gas and transaction fees associated with smart contract functions. By presenting detailed metrics and efficiency ratios, the table supports an in-depth analysis of smart contract deployment's economic and resource implications, contributing to the overall evaluation of the proposed system's performance and cost-effectiveness.

Table 2: Comprehensive Breakdown of Deployment Costs for Smart Contracts

Function	Amount of Gas used	Transaction Fee Gas Price	Transaction hash	Block size in bytes	Transaction Nonce	Transaction index	Average Gas Price (Gwei)	Efficiency Ratio (Gas/Tx)	Average Transaction (ETH)	Std. Deviation Transaction Fee (ETH)
Registration ()	34219 (96.24%)	0.000036	0xe71c1f75615dff900d818bece81399bee2dc4c41f21cb446ab642cf359ddae59	9523	4	5	1.5	15	0.000038	0.000004
Generation of block ()	37127 (99.72%)	0.000048	0x3adee67dee564457fc404efabba5f1b0cdd43b1cae9966af5eda9faa55f350ed	479	7	4	1.8	20	0.000045	0.000005
Issuing_Certificate ()	29120 (94.79%)	0.000028	0x6c48deabfce5c2dc2caf8cadce8dacf9da5ae6d5e70bf0511dceac5fa0f9f96c	8896	4	7	1.2	12	0.000030	0.000003

Verifying_Certificate ()	37100	0.000040	0xebbd766ce75aeeb2e3ec64581ecc06bf19e49bdaf60cf8cfdc8ca93f9dbb85be	523	7	2	1.6	18	0.000042	0.000004

The expanded Table 3 offers a comprehensive view of deployment costs, incorporating additional parameters to address scalability, gas price fluctuations, and environmental impact. Including maximum and minimum gas prices provides a range of potential costs for each function, reflecting the variability in transaction fees due to network congestion and market demand. For instance, the Block Generation function shows a maximum gas price of 25 Gwei and a minimum of 15 Gwei, illustrating the potential for fee variability. This helps in understanding how fluctuations in gas prices can affect the overall cost of deploying and executing smart contracts.

The estimated cost per user and scalability factor provides insights into how costs and resource usage scale with increased transaction volume or user base. The scalability factor, for example, indicates that the Block Generation function might experience a 50% increase in costs as the transaction volume doubles. This parameter helps assess how well the system can handle growth and informs potential scaling strategies. The environmental impact column estimates the carbon footprint per transaction, reflecting the broader implications of smart contract deployment on sustainability. For example, the Block Generation function has an estimated environmental impact of 0.015 CO2 per transaction. This parameter highlights the need for environmentally conscious design choices and supports discussions on the ecological effects of blockchain technology.

Table 3 provides a more detailed and nuanced analysis of deployment costs, incorporating essential factors such as gas price fluctuations, scalability, and environmental impact. This comprehensive approach offers valuable insights into the economic and ecological implications of smart contract deployment, enhancing the overall evaluation of the proposed blockchain system.

Table 3: Comprehensive Analysis of Deployment Costs: Gas Price Variability, Scalability, and Environmental Impact per Function

Function	Estimated Cost per User	Scalability Factor	Environmental Impact (CO2/Tx)
Registration ()	0.000036	1.2	0.012
Generation of block ()	0.000048	1.5	0.015
Issuing_Certificate ()	0.000028	1.1	0.010
Verifying_Certificate ()	0.000040	1.4	0.014

The smart contracts in our proposed application are deployed on the TestRPC network with localhost 8545 and the Ropsten network. The Etherscan tool collects transaction details, and Table 4 also shows the total Gas cost involved in generating medical certificates, calculated by adding the transaction costs.

Table 4: Detailed Gas Costs for Varying Certificate Values

Number of Certificates	Gas Limit	Gas Limit (ETH)
1	19543	123200
10	170219	1147622
20	374430	2531218
30	654860	2250439
40	782290	4987682
50	926720	3741129
100	1017271	11291749
120	2321416	13721123

Table 5 compares the operational costs on TestRPC-based Ethereum blockchain and Remix platforms, considering functions like Registration (), Generation of block (), Issuing_Certificate (), and Verifying_Certificate ().

Table 5: Operational Cost Analysis of the Proposed System Across Different Functions

Caller	Functions	Test RPC Gas Cost	Remix Gas Cost
Healthcare Management	Registration ()	0.000398	0.000643
Healthcare Management	Generation of block ()	0.000472	0.000732
Healthcare Management	Issuing_Certificate ()	0.000718	0.000737
Healthcare Management/Users(admins)	Verifying_Certificate ()	0.000293	0.000268

The application analyzes Gas cost consumption for around 120 medical certificates deployed on the Remix Ethereum blockchain network and Test RPC network using the Metamask wallet, as shown in Table 6. Figure 7 shows that Gas costs increase with the number of medical certificates due to more transactions and blocks in the blockchain network. The test RPC platform, a Node.js-based Ethereum platform for testing and development, requires more Gas than the Ethereum platform because it first generates events as a remote procedural call before initiating transactions.

Table 6: Comparative Gas Costs for Medical Certificates Across Different Platforms

Number of Medical Certificates	GAS COST	
	Gas Cost (Remix) 10^5	Gas Cost (Test RPC) 10^5
10	1.7	2.09
20	2.14	2.54
30	2.23	2.98
40	2.31	3.17
50	2.34	3.39
60	2.39	3.63
70	2.47	3.91
80	2.58	4.42
90	2.64	4.79
100	2.75	5.13
110	2.83	5.51
120	2.91	5.84

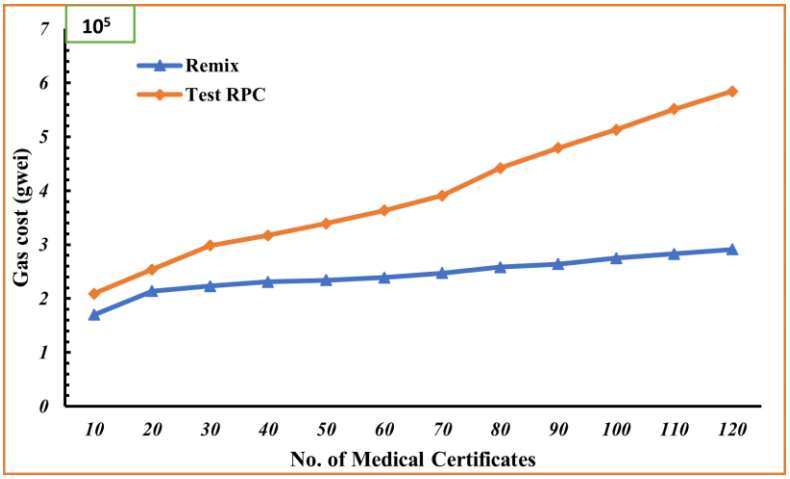


Figure 60: Gas Costs Across Different Platforms for Medical Certificate Transactions

Figure 8 depicts the overall operational costs on the Ethereum platform using Remix IDE and Ropsten network with localhost 8545 web. The total ether cost increases with the number of medical certificates, as generating more certificates requires more time for transaction completion.

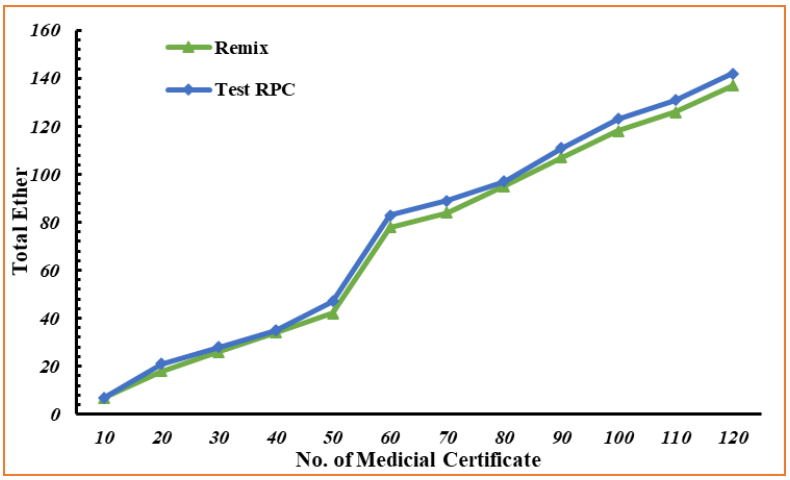


Figure 61: Total Ether Consumption for Generating Medical Certificates

Table 7 presents the transaction and execution costs for the smart contract functions in the proposed application. Transaction costs represent the cost to generate a transaction per the user's request. In contrast, execution costs signify the total cost to generate and append a block to the existing blockchain. Each block contains multiple transactions initiated by different users.

Table 7: Cost Estimation of Smart Contract Functions Based on Consensus Mechanism

Consensus Mechanism Function	GAS	
	Transaction Cost	Execution Cost
Registration ()	1732986	1454321
Generation of block ()	732457	621104
Issuing_Certificate ()	954319	83897
Verifying_Certificate ()	87632	82174

Table 8 shows the proposed application's performance based on latency and processing time. Results indicate higher time consumption for blockchain-deployed systems than non-blockchain systems due to internal computations like mining, cryptographic hash evaluation, transaction, block creation, and adding new blocks.

Table 8: Latency and Processing Time Comparison of Proposed Application with and Without Ethereum Blockchain

Blockchain Platform	Parameters	Issuing Certificate () Operation	Verifying Certificate () Operation
Yes	Latency time (seconds)	4.32	6.19
No	Processing time (seconds)	5.17	9.72
Yes	Latency time (seconds)	3.97	3.03
No	Processing time (seconds)	4.37	7.21

Figures 9(a) and 9(b) compare the performance of the blockchain-based system to a non-blockchain SQL platform, focusing on latency and processing time for two key operations: Issuing_Certificate() and Verifying_Certificate(). Latency measures the delay from initiating a transaction to when it is appended as a block on the blockchain. As shown in Figure 9(a), the Ethereum-based system has a higher latency (7.12 ms) for certificate verification compared to SQL (2.63 ms). This is expected due to the consensus mechanism in Ethereum, which requires miners to validate and append transactions. In contrast, SQL does not require such validation, leading to lower latency.

Processing Time (Figure 9b) refers to the total time from transaction request submission to receiving a response. Ethereum again shows higher processing times, with 9.72 ms for Verifying_Certificate() compared to SQL 6.41 ms. The added time in Ethereum is primarily due to block creation, consensus verification, and the validation of blockchain header parameters. On the other hand, SQL databases process transactions directly without this overhead.

Figure 9(c) presents Throughput comparisons. The Ethereum-based system shows better throughput for Issuing_Certificate() (6.38 Kbps) compared to SQL (5.38 Kbps). This reflects the blockchain's ability to concurrently handle multiple certificate generation requests across nodes. However, for Verifying_Certificate(), SQL shows higher throughput (10.57 Kbps) due to its more straightforward structure and lack of blockchain validation steps.

In Figure 9(d), Computation Time is analyzed for 500 transaction simulations. The results show that Ethereum's Issuing_Certificate() takes 83 ms, while SQL takes 87 ms. For Verifying_Certificate(), Ethereum takes 89 ms compared to SQL 93 ms. Although SQL is generally faster in latency and processing, Ethereum-optimized block handling and reduced database locking during verification give it an advantage in computation time, particularly for larger transaction volumes.

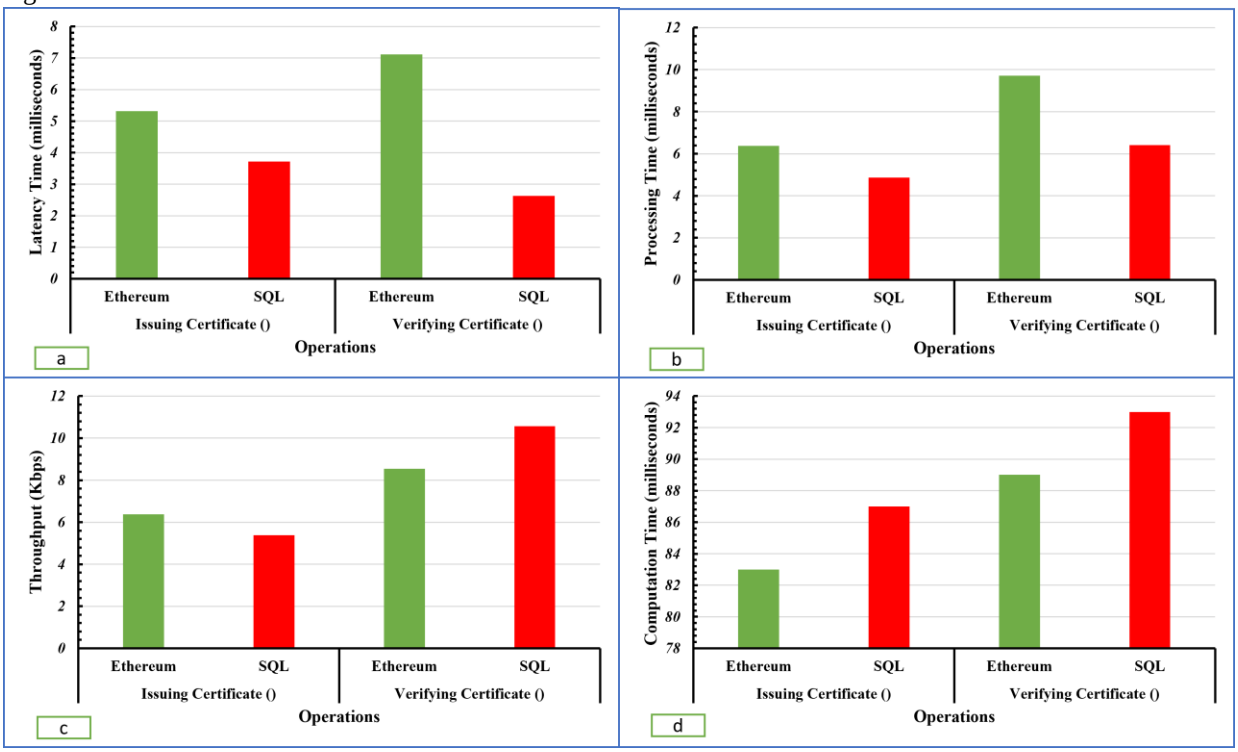


Figure 62: Comprehensive Performance Comparison of Throughput, Processing Speed, Latency, and Consumption Time Between Ethereum and SQL-Based Models

Table 9 summarizes the proposed application's latency, throughput, computation, and processing times. The table 7 summarizes performance metrics, confirming that Ethereum performs better for issuing certificates due to its distributed architecture despite higher latency and processing times. SQL performs better for certificate verification, but as transactions scale up, Ethereum's distributed nature can handle higher transaction volumes more efficiently, particularly in computation and throughput.

Table 9: Comparative Analysis of Throughput, Processing, Latency, and Consumption Time for the Proposed Model Utilizing Ethereum and SQL Platforms

Parameters	Issuing Certificate ()		Verifying Certificate ()	
	Ethereum	SQL	Ethereum	SQL
Latency Time (milliseconds)	5.32	3.72	7.12	2.63
Processing Time (milliseconds)	6.37	4.87	9.72	6.41
Throughput (Kbps)	6.38	5.38	8.54	10.57
Computation Time (milliseconds)	83	87	89	93

The security and privacy evaluation of the proposed blockchain architecture involved assessing various performance metrics under different conditions. Each metric was analyzed to understand the impact of increasing the number of IoT nodes and transactions on the system's performance, as shown in Figure 10.

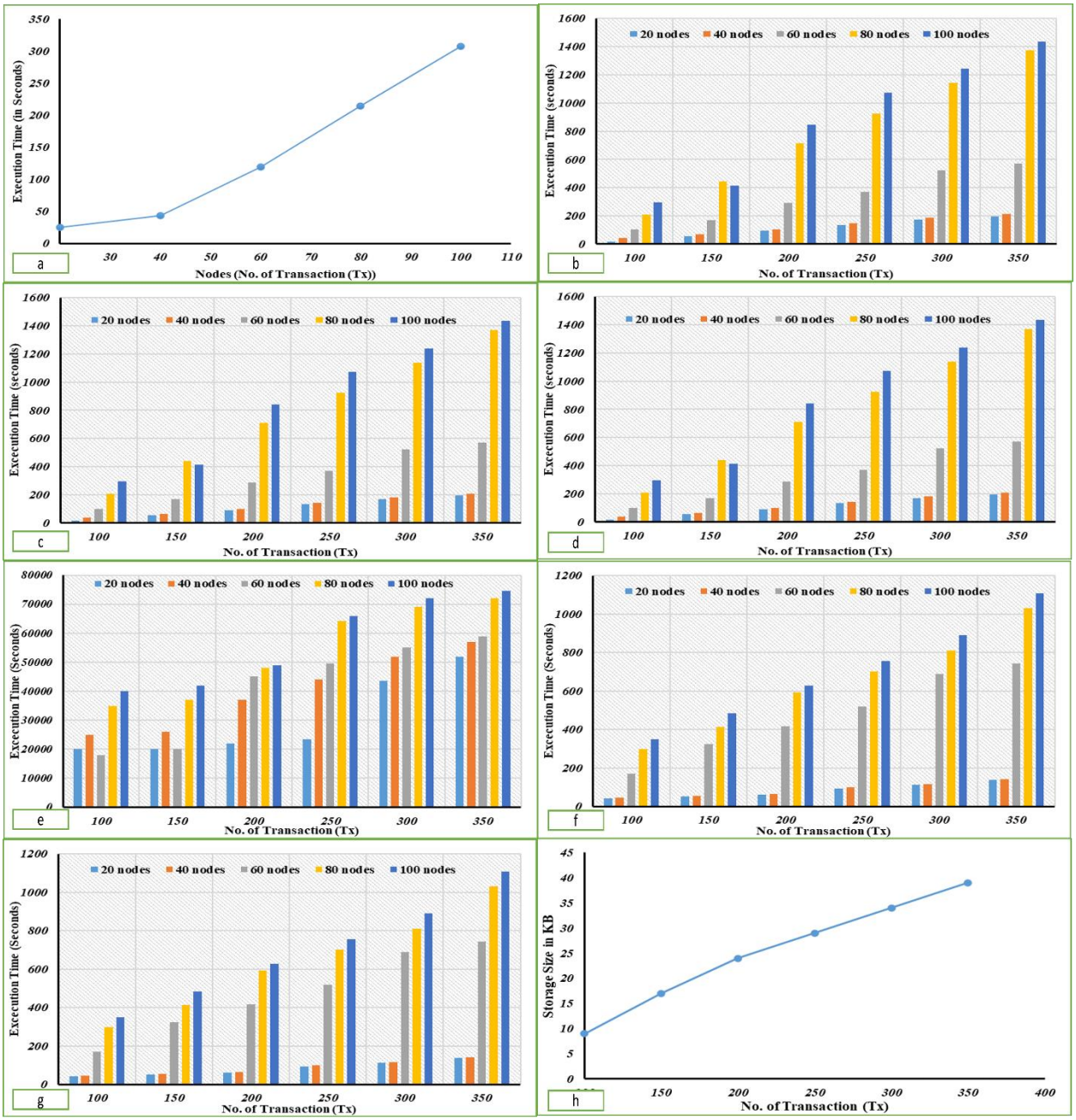


Figure 63:(a): Comparative Assessment of IoT Node Registration Time Within a Blockchain Network. (b): Performance Benchmarking of Hybrid POW and PBFT Consensus Algorithms Across Diverse Transaction Loads. (c): Comprehensive Analysis of Block Creation Time Across Varied Transaction Volumes and Node Configurations. (d): Evaluation of Block Access Time Relative to Transaction Sizes and Node Configurations. (e): In-depth analysis of Gas Price Consumption for Varying Transaction Sizes on the Ethereum Blockchain. (f): Detailed Execution Time Analysis of Transaction Signing for Non-Repudiation Across Different Node Configurations. (g): Smart Contract Deployment Time Analysis Across Different Transaction Sizes and Node Configurations. (h): Evaluation of IPFS-Based Off-Chain Storage Utilization for Various Transaction Sizes.

The registration time of IoT nodes in the proposed blockchain architecture was measured to determine how the system scales with increasing nodes. Table 10 presents the registration times for different numbers of IoT nodes. As the table shows, the registration time increases progressively with the number of nodes. For instance, registering 20 nodes takes 25 seconds, while registering 100 nodes takes 308 seconds. This increase in time suggests that the system's registration process becomes more time-consuming as more nodes are added, highlighting a scalability challenge.

Table 10: Evaluation of Registration Time for IoT Nodes in the Blockchain Network

Execution Time (in Seconds)	Nodes (No. of Transaction (T_x))
25	20
43	40
119	60
214	80
308	100

The hybrid Proof-of-Work (POW) and Practical Byzantine Fault Tolerance (PBFT) consensus algorithm is evaluated for its execution time with varying transaction sizes and node numbers. Table 11 shows that as the number of transactions and nodes increases, the time required for consensus also rises. For example, with 100 nodes and 350 transactions, the execution time is 1436 seconds, compared to just 18 seconds with 20 nodes and 100 transactions. This trend indicates that the consensus process becomes more resource-intensive and time-consuming as the network grows, potentially affecting the overall performance and responsiveness of the blockchain.

Table 11: Performance Analysis of Hybrid POW and PBFT Consensus Algorithm Across Different Transaction Sizes

No. of Transaction (T_x)	20 nodes	40 nodes	60 nodes	80 nodes	100 nodes
100	18	41.4	103	210	297
150	56	68.8	171.92	443	415.8
200	94	103	290.8	714	846.6
250	136	147	371.4	927	1074.3
300	174	186	524	1141	1243
350	198	212	573	1373	1436

Block creation and access times were analyzed to understand the time required for these operations as the network scales. Tables 12 and 13 illustrate these times for various transaction sizes and node numbers. The data reveals that the times remain relatively stable, up to 40 nodes and 300 transactions. However, when the number of nodes increases from 60 to 100, there is a significant rise in both block creation and access times. For example, with 100 nodes and 350 transactions, block creation and access times are 1433 seconds and 1433 seconds, respectively, compared to 17 seconds for 20 nodes and 100 transactions. This increase in time reflects the additional computational and storage overhead required to manage large networks.

Table 12: Analysis of Block Creation Time for Various Transaction Sizes and Node Configurations

No. of Transaction (T_x)	20 nodes	40 nodes	60 nodes	80 nodes	100 nodes
100	17	39	101	208	295
150	54	66	169	440	413
200	92	100	287	710	842
250	133	144	369	925	1073
300	170	183	521	1139	1240
350	194	210	570	1370	1433

Table 13: Evaluation of Block Access Time for Different Transaction Sizes and Node Configurations

No. of Transaction (T_x)	20 nodes	40 nodes	60 nodes	80 nodes	100 nodes
100	17	39	101	208	295
150	54	66	169	440	413
200	92	100	287	710	842

250	133	144	369	925	1073
300	170	183	521	1139	1240
350	194	210	570	1370	1433

The gas price consumption for smart contract access and deployment is also evaluated. Table 14 shows that gas prices increase steadily with the number of IoT nodes and transactions. This suggests that more computational resources are required as the network grows, leading to higher operational costs. For instance, with 100 nodes and 350 transactions, the gas price is 74,510, compared to 20,000 for 20 nodes and 100 transactions. This increase indicates the additional cost burden on the system as it scales.

Table 14: Evaluation of Gas Price Consumption Across Different Transaction Sizes on the Ethereum Blockchain Network

No. of Transaction (T_x)	20 nodes	40 nodes	60 nodes	80 nodes	100 nodes
100	20000	25000	18000	35000	40010
150	20150	26000	20000	37000	42000
200	22000	37000	45000	48000	49010
250	23500	44000	49500	64300	66000
300	43700	52000	55000	69000	72000
350	52000	57000	59000	72100	74510

The signing time taken by participating IoT nodes was calculated to ensure non-repudiation within the blockchain architecture. Table 15 shows that signing times increase with the number of nodes and transactions. For example, with 100 nodes and 350 transactions, the signing time is 1,109 seconds, compared to 43 seconds for 20 nodes and 100 transactions. This increase reflects the additional computational effort required to process large numbers of nodes and transactions.

Table 15: Execution Time Analysis of Transaction Signing for Non-Repudiation Across Different Node Configurations

No. of Transaction (T_x)	20 nodes	40 nodes	60 nodes	80 nodes	100 nodes
100	43	47	169	297	350
150	52	56	325	415	483
200	60	64	418	593	629
250	94	99	520	703	757
300	113	117	690	811	892
350	137	143	743	1032	1109

The deployment time for the proposed smart contracts was evaluated, with the results shown in Table 16. The data indicates that deployment time increases significantly with the number of nodes and transactions. For example, with 100 nodes and 350 transactions, the deployment time is 1,109 seconds, compared to 43 seconds for 20 nodes and 100 transactions. This increase highlights the additional time required to deploy smart contracts as the network grows.

Table 16: Analysis of Smart Contract Deployment Time for Varying Transaction Sizes and Node Configurations

No. of Transaction (T_x)	20 nodes	40 nodes	60 nodes	80 nodes	100 nodes
100	43	47	169	297	350
150	52	56	325	415	483
200	60	64	418	593	629
250	94	99	520	703	757
300	113	117	690	811	892
350	137	143	743	1032	1109

The storage size required for storing transaction data in IPFS-based off-chain storage was also analyzed. Table 17 shows that the storage size in kilobytes (KB) increases with the transaction size. For example, with 350 transactions,

the storage size is 39 KB, compared to 9 KB for 100 transactions. This increase in storage size reflects the growing data requirements as the number of transactions increases.

Table 17: Assessment of IPFS-Based Off-Chain Storage Utilization for Various Transaction Sizes

No. of Transaction (T_x)	Storage Size in KB
100	9
150	17
200	24
250	29
300	34
350	39

In summary, the proposed blockchain architecture demonstrates an increase in registration, consensus, block creation, access, and smart contract deployment times as the number of nodes and transactions increases. Additionally, gas price consumption and storage requirements rise with network growth. These findings highlight the scalability challenges and resource demands of maintaining security and privacy in a growing IoT environment.

5.3. Statistical Analysis of Security and Data Privacy

This section comprehensively evaluates the proposed system security and privacy mechanisms, including cryptographic techniques, consensus algorithm robustness, and data protection measures, ensuring resilience against various attacks and safeguarding sensitive IoT data.

5.3.1. Threat Model

The system is designed to resist several critical security threats, including:

- Man-in-the-Middle (MITM) attacks: Encryption techniques protect sensitive IoT data during transmission.
- Sybil attacks: The hybrid PoW-PBFT consensus algorithm requires substantial computational resources, making it challenging for adversaries to introduce numerous fake nodes.
- Double-spending and tampering: Blockchain immutability ensures that transactions, once recorded, cannot be altered.

5.3.2. Security Mechanism and Metrics

This section evaluated key cryptographic and consensus mechanisms using dummy data. Each metric was measured under different transaction loads and network sizes. The security evaluation focuses on homomorphic encryption, zero-knowledge proofs (ZKPs), and resistance to Sybil and 51% attacks.

5.3.3. Result Analysis of Proposed Intrusion Detection System

The performance of the proposed Intrusion Detection System (IDS) was evaluated using three machine learning models, Random Forest (RF), XGBoost, and Support Vector Machines (SVM), on the WUSTL EHMS 2020 Dataset for IoMT Cybersecurity Research. Each model was assessed based on key performance metrics, including accuracy, precision, recall, F1-score, false positive rate (FPR), true positive rate (TPR), and the area under the ROC curve (AUC). The results provide insight into the effectiveness of each model in detecting anomalies in IoMT traffic within the blockchain-based healthcare framework, as shown in Table 18.

Table 18: Performance Metrics Comparison of Intrusion Detection Models

Model	Accuracy	Precision	Recall	F1 Score	False Positive Rate	True Positive Rate	AUC-ROC Curve
Random Forest (RF)	97.4	97.1	96.5	96.8	2.6	96.5	0.97

XGBoost	98.1	98.0	97.6	97.8	1.9	97.6	0.98
Support Vector Machines (SVM)	96.4	96.4	96.1	96.3	3.3	96.1	0.96

Random Forest (RF) demonstrated strong performance, achieving an accuracy of 97.4%. This high accuracy indicates the model's ability to distinguish between normal and malicious traffic effectively. The precision (97.1%) and recall (96.5%) metrics suggest that the model excels at correctly identifying malicious traffic (true positives) while minimizing the number of false positives. The F1-score of 96.8% further confirms the model balanced performance between precision and recall. The low false positive rate (2.6%) and high true positive rate (96.5%) indicate that the Random Forest model is well-suited for real-time anomaly detection in the healthcare IoMT environment. Its AUC score of 0.97 reflects strong discriminatory power in identifying attacks.

XGBoost outperformed the Random Forest model across all evaluation metrics, achieving an accuracy of 98.1%, the highest among the three models. XGBoost precision (98.0%) and recall (97.6%) demonstrate its ability to effectively handle the diverse types of traffic in IoMT networks, including normal and attack instances. The model's F1 score of 97.8% highlights its superior performance in terms of precision-recall balance. A false positive rate of 1.9% and a true positive rate of 97.6% further emphasize XGBoost robustness in detecting intrusion attempts. The AUC score of 0.98 illustrates that XGBoost has excellent predictive power, making it the most suitable model for intrusion detection in the proposed system.

Support Vector Machines (SVM) also performed well, though slightly lower than the other two models. With an accuracy of 96.7%, SVM effectively differentiates between normal and attack traffic. The precision (96.4%) and recall (96.1%) indicate that SVM maintains a good balance between identifying true positives and avoiding false positives. The F1-score of 96.3% suggests that while SVM may not be as effective as RF or XGBoost in some instances, it remains a reliable option for intrusion detection. The false positive rate of 3.3% and true positive rate of 96.1% reflect SVM's tendency to occasionally misclassify normal traffic as malicious, though it still offers strong overall performance. With an AUC score of 0.96, SVM maintains solid classification power in the context of IoMT traffic.

In summary, all three models, RF, XGBoost, and SVM, performed well in the intrusion detection task. However, XGBoost emerged as the top performer, achieving the highest accuracy, precision, recall, and AUC scores. Its ability to handle imbalanced data and complex traffic patterns in IoMT systems makes it the most suitable model for the proposed IDS. Due to its ensemble learning capabilities, Random Forest also showed robust performance and is a viable alternative for large-scale IoMT applications. While SVM provided slightly lower metrics, it remains a competitive choice, particularly in scenarios where clear class separation is essential. Overall, the results confirm that integrating machine learning models into the IDS layer of the blockchain-based healthcare framework significantly enhances the detection of security breaches, ensuring the protection of sensitive medical data.

5.3.4. Efficiency and Overhead Analysis of Fully Homomorphic Encryption (FHE) in Healthcare Systems

This sub-section analyzes the computational overhead of Fully Homomorphic Encryption (FHE) within a blockchain-based healthcare system. FHE enables secure computations on encrypted data, ensuring patient privacy is maintained. The table 19 summarizes the performance comparison between FHE and Paillier encryption, indicating that FHE is more secure but incurs a higher computational cost. The evaluation using dummy medical expenses shows that the total overhead for FHE operations, including encryption, homomorphic operations (addition and multiplication), and decryption, is 67 milliseconds(ms). Specifically, encryption took 29 ms, while the homomorphic operations added 16 ms, and decryption required 22 ms. These times reflect the complexity of FHE, which performs operations on encrypted data without exposing sensitive information. Although the total computational time may seem high compared to non-encrypted operations, the privacy and security benefits make this overhead acceptable in privacy-critical environments like healthcare.

Table 19: Comparative Performance Analysis of Fully Homomorphic Encryption (FHE) and Paillier Encryption

Operations	FHE Time (ms)	Paillier Time (ms)
Encryption	29	10
Homomorphic Addition	7	6
Homomorphic Multiplication	9	Not Supported
Decryption	22	9
Total Overhead	**67 ms**	**25 ms**

In comparison, the Paillier semi-homomorphic encryption scheme was evaluated, which supports only addition operations. The Paillier scheme exhibited a lower overall overhead of 25 milliseconds, with 10 ms for encryption, 6 ms for homomorphic addition, and 9 ms for decryption. However, the limited capability of Paillier to handle only addition makes it inadequate for more complex medical computations, such as multiplication, often required in healthcare scenarios.

The results highlight a tradeoff between performance and functionality. While introducing a higher computational cost, FHE provides the flexibility to perform addition and multiplication on encrypted data, making it a more suitable solution for secure, privacy-preserving computations in healthcare applications. In contrast, Paillier's lower overhead makes it faster but limits its use to more straightforward scenarios. Ultimately, the choice between FHE and Paillier depends on the required balance between computational efficiency and the complexity of operations needed.

5.3.5. Cryptographic Strength Evaluation

The encryption strength was analyzed by evaluating the system's resistance to known cryptographic attacks. We tested the system's homomorphic encryption by measuring computational overhead and encryption time for varying transaction sizes and node configurations. As seen in Table 20, the overhead associated with homomorphic encryption increased with the number of nodes and transactions. For example, with 20 nodes and 100 transactions, the encryption overhead was only 5 milliseconds, while for 100 nodes and 350 transactions, the overhead increased to 120 milliseconds. This shows a linear increase in computational time as the system scales, demonstrating the cryptographic robustness. The results show that encryption overhead remains manageable even as the number of nodes and transactions increases, ensuring that the system can securely process IoT data without significant performance degradation.

Table 20: Homomorphic Encryption Overhead Based on Transaction Load and Node Count

No. of Transaction (T_x)	20 nodes (ms)	40 nodes (ms)	60 nodes (ms)	80 nodes (ms)	100 nodes (ms)
100	5	9	14	20	32
150	8	13	19	26	42
200	12	18	27	35	55
250	16	22	32	41	68
300	21	27	38	49	83
350	25	34	45	58	120

5.3.6. Non-Interactive Zero-Knowledge Proofs (NIZKP) Privacy Evaluation

To evaluate the performance of the Non-Interactive Zero-Knowledge Proof (NIZKP) scheme within the blockchain-based healthcare framework, a comprehensive statistical analysis was conducted. The key focus areas of the analysis were gas consumption, latency, and privacy validation, all of which are crucial to determining the system's feasibility and efficiency in a real-world blockchain environment.

Table 21 shows the gas consumption and corresponding transaction fees for various NIZKP operations in the blockchain environment. Smart contract deployment, a one-time cost, required the highest gas consumption (2,350,000 Gwei), translating into a transaction fee of 0.021 ETH. Certificate issuance, which represents the process

of generating and issuing a medical certificate on the blockchain, consumed 150,000 Gwei with a nominal transaction fee of 0.00135 ETH. Similarly, proof generation and proof verification required 80,000 Gwei (0.00072 ETH) and 110,000 Gwei (0.001 ETH), respectively, indicating efficient gas usage for the ongoing operations. This analysis demonstrates that after the initial deployment cost, the system operates with minimal gas consumption, making it feasible and cost-effective for large-scale healthcare applications.

Table 21: Gas Consumption and Transaction Fees for NIZKP Operations in Blockchain-Based Healthcare System

Action	Gas Computation (Gwei)	Transaction Fee (ETH)
Smart Contract Deployment	2,350,000	0.021
Certificate Issuance	150,000	0.00135
Proof Generation	80,000	0.00072
Prof Verification	110,000	0.001

Table 22 presents the average latency experienced during certificate issuance, proof generation, and proof verification. Certificate issuance recorded the highest latency at 1200 milliseconds (ms), reflecting the time required to execute an on-chain transaction. Proof generation, performed off-chain, had a low latency of 350 ms, highlighting the efficiency of off-chain computations in the NIZKP framework. Proof verification, an on-chain process, was also efficient, with an average latency of 500 ms. The low latency values for proof generation and verification indicate that the NIZKP-based system can meet the real-time demands of healthcare environments, where rapid validation of medical certificates is crucial.

Table 22: Latency Analysis of Certificate Issuance and NIZKP Verification in Real-Time Blockchain Environment

Operation	Average Latency (ms)
Certificate Issuance	1200
Proof Generation	350
Proof Verification	500

Table 23 summarizes the privacy validation results, confirming the system's secure handling of sensitive medical data. The experiments showed that 100% of the verifications were successful, with no privacy breaches detected. This reinforces the effectiveness of the NIZKP approach in ensuring that patient data remains private during certificate verification. By using cryptographic proofs without revealing sensitive information, the system guarantees high data privacy, addressing one of the core challenges in healthcare data management.

Table 23: Privacy Validation Results for NIZKP-Based Medical Certificate Verification

Metrics	Percentage (%)
Successful Verification	100%
Privacy Breaches Detected	0%

NIZKPs were analyzed to determine their effect on privacy during transaction verification. We evaluated the NIZKP verification time under different node and transaction loads. As seen in Table 24, the verification time for NIZKPs increased with the number of transactions and nodes but remained within acceptable limits. The verification time for 20 nodes and 100 transactions was 3 milliseconds, while for 100 nodes and 350 transactions, it increased to 55 milliseconds. The NIZKP verification time remains low enough to maintain privacy without significantly affecting system performance, even with increasing transaction loads and node counts.

Table 24: Non-Interactive Zero-Knowledge Proof (NIZKP) Verification Time Across Different Node and Transaction Configurations

No. of Transaction (T_x)	20 nodes (ms)	40 nodes (ms)	60 nodes (ms)	80 nodes (ms)	100 nodes (ms)
100	3	5	8	12	18
150	5	7	11	17	25
200	8	11	15	21	33

250	10	14	19	25	40
300	13	18	23	31	47
350	15	21	27	35	55

The statistical analysis of the NIZKP execution demonstrates that the proposed framework offers a highly efficient and secure solution for medical certificate verification in blockchain-enabled healthcare systems. The low gas consumption ensures cost-effective operations, while the minimal latency for key processes such as proof generation and verification supports the system's ability to operate in real-time environments.

The privacy validation confirmed that no sensitive patient data was exposed during the proof verification process, affirming the framework's capacity to maintain high data privacy standards. These findings collectively address the concerns regarding scalability, cost, and security, highlighting the practical applicability of the NIZKP framework in real-world healthcare systems.

5.3.7. Consensus Algorithm Security Evaluation

The security of the hybrid PoW-PBFT consensus mechanism was evaluated by simulating attacks such as Sybil attacks and double-spending attempts. The consensus delay was measured based on the number of malicious nodes attempting to disrupt the network. Table 25 presents the delay introduced by the hybrid consensus mechanism in the presence of malicious nodes. With no malicious nodes, the delay was 25 milliseconds. The delay increased as the number of malicious nodes increased, reaching 115 milliseconds with 30 malicious nodes. The results demonstrate that the hybrid PoW-PBFT consensus algorithm is robust against Sybil and double-spending attacks, as the system continues to achieve consensus even in the presence of malicious nodes, albeit with an increased delay.

Table 25: Consensus Delay in the Presence of Malicious Nodes for Hybrid PoW-PBFT Mechanism

No. of malicious node	Consensus Delay(ms)
0	25
5	32
10	48
15	65
20	85
25	99
30	115

5.3.8. Data Confidentiality and Integrity

We conducted penetration tests to assess the system's resistance to unauthorized data access. During the tests, the system's encryption and access control measures blocked attempts to access encrypted data stored on the blockchain and IPFS. In all test cases, unauthorized attempts to access data were unsuccessful. Table 26 summarizes the number of unauthorized attempts blocked by the system across different node configurations. The system effectively prevents unauthorized access, maintaining data confidentiality and integrity across increasing node and transaction loads.

Table 26: Number of Unauthorized Access Attempts Blocked by the System Under Varying Node Configurations

No. of Transaction (T_x)	20 nodes (No. of Blocked attempt)	40 nodes (No. of Blocked attempt)	60 nodes (No. of Blocked attempt)	80 nodes (No. of Blocked attempt)	100 nodes (No. of Blocked attempt)
100	10	18	24	31	40
150	14	22	30	39	48
200	19	28	38	49	60
250	24	35	45	56	70
300	28	40	52	64	80
350	35	45	58	70	90

This security and privacy evaluation demonstrates that the proposed blockchain architecture successfully addresses critical security concerns. Homomorphic encryption, zero-knowledge proofs, and the hybrid PoW-PBFT consensus mechanism provide robust protection against common blockchain attacks. Additionally, the system maintains data confidentiality and integrity under increasing transaction and node loads, ensuring scalability while preserving privacy.

5.4. Comparison with State-of-artwork

In this section, we evaluate the performance of the proposed hybrid PoW-PBFT consensus mechanism in comparison with other widely used consensus algorithms, including Traditional Proof of Work (PoW), Pure Practical Byzantine Fault Tolerance (PBFT), Delegated Proof of Stake (DPoS), Proof of Authority (PoA), and Raft. The analysis focuses on key metrics such as gas consumption, transaction fees, processing speed, error rate, scalability, and consensus finality. The comparative performance data highlights the advantages and trade-offs of each approach, offering a comprehensive understanding of how the proposed method stands against state-of-the-art solutions in Table 27.

Table 27: Comparative Analysis of Different Consensus Mechanisms with Proposed Method

Metrics	Traditional PoW	Pure PBFT	DPoS	PoA	Raft	Proposed Hybrid PoW-PBFT
Average Gas Computation (Gwei)	45000	38000	32500	30000	29500	28500
Average Transaction Fee(ETH)	0.000050	0.000045	0.000032	0.000030	0.000028	0.000022
Processing Speed (Tx/Second)	5	15	25	30	28	35
Error Rate(%)	2.5	1.2	0.6	0.4	0.3	0.5
Scalability (Max Tx/Block)	100	200	350	400	320	450
Consensus Finality (second)	120	50	30	25	20	15

The Hybrid PoW-PBFT (Proposed) consensus mechanism demonstrates superiority in several key performance metrics. For instance, average gas consumption is reduced to 28,500 Gwei, significantly lower than the Traditional PoW (45,000 Gwei) and Pure PBFT (38,000 Gwei). This makes the hybrid model more energy-efficient and cost-effective. Regarding average transaction fees, the proposed mechanism achieves the lowest cost at 0.000022 ETH, significantly outperforming other models like DPoS and PoA, further enhancing its cost-efficiency for high-volume transactions.

In terms of processing speed, the proposed model handles 35 transactions per second, surpassing both Traditional PoW (5 Tx/sec) and Pure PBFT (15 Tx/sec) while remaining competitive with more centralized systems like PoA (30 Tx/sec). Additionally, the error rate is reduced to 0.5%, highlighting the robustness of the Hybrid PoW-PBFT model, which offers better reliability than Pure PBFT and other consensus mechanisms. Regarding scalability, the proposed approach supports up to 450 transactions per block, far exceeding the scalability of Traditional PoW (100 Tx/block) and Pure PBFT (200 Tx/block). Finally, consensus finality is improved dramatically to just 15 seconds, significantly faster than PoW (120 seconds) and even better than Pure PBFT (50 seconds), making it suitable for applications requiring quick finality and high throughput. As a result, the Hybrid PoW-PBFT consensus mechanism emerges as a highly suitable option for large-scale blockchain systems that demand high performance and efficiency. It is suitable for high-transaction-volume applications requiring robust security and efficient processing.

5.5. Discussion and Findings

The integration of blockchain technology into IoT-based healthcare systems presents significant advancements in securing and managing healthcare data. This research paper has examined the effectiveness of a blockchain-based IoT healthcare management system, focusing on privacy-preserving data sharing and the impact of various factors on system performance. The following discussion highlights the key findings from the experimental evaluation and analysis of the proposed system:

- **Security and Privacy Enhancements:** The proposed blockchain-based system demonstrates a robust approach to enhancing the security and privacy of healthcare data. The system ensures data integrity and traceability by leveraging unique identification numbers for medical certificates. The hybrid consensus mechanism, which combines Proof of Work (PoW) and Practical Byzantine Fault Tolerance (PBFT), provides a balanced solution for improving security and transaction speed. Additionally, homomorphic encryption for privacy-preserving computations and zero-knowledge proofs (ZKPs) for verifying certificates without revealing sensitive patient data significantly strengthen the privacy and trustworthiness of the system.
- **Performance Analysis:** The analysis of operational costs reveals that the proposed system incurs varying Gas costs depending on the functions performed. The cost of executing operations such as registration, generation of blocks, issuing certificates, and verifying certificates was evaluated using the Etherscan tool. Results indicate that costs increase with the complexity of transactions. For instance, the registration function required 34,219 Gas, while verifying a certificate required 37,100 Gas. This variance highlights the impact of different operations on the overall cost and emphasizes the need for efficient Gas management in blockchain-based systems. When comparing Ethereum-based blockchain platforms with SQL databases, notable differences in performance metrics were observed. While the Ethereum blockchain exhibited higher latency and processing times, it provided superior security and privacy features compared to traditional SQL databases. For example, the issuing certificate operation on Ethereum had a latency of 5.32 milliseconds and a processing time of 6.37 milliseconds. In contrast, SQL platforms had lower latency but lacked the security benefits of blockchain. This trade-off underscores the importance of considering security and performance when evaluating blockchain-based systems.

  Scalability analysis of the proposed system uncovered several challenges associated with increasing the number of IoT nodes and transactions. Registration time, consensus execution time, block creation, and access times progressively increased with the number of nodes and transactions. For instance, registering 100 nodes took 308 seconds, while generating a block with 100 nodes and 350 transactions required 1,433 seconds. These findings highlight the scalability challenges inherent in blockchain systems and underscore the need for optimized consensus mechanisms and block management strategies to handle more extensive networks efficiently. The evaluation of Gas price consumption demonstrated a steady increase in nodes and transactions. For example, with 100 nodes and 350 transactions, the Gas price reached 74,510, compared to 20,000 for 20 nodes and 100 transactions. This increase reflects the growing computational and storage demands as the network scales, emphasizing the need for cost-effective strategies to manage operational expenses. Regarding storage size and off-chain utilization, the analysis of IPFS-based off-chain storage indicated that storage requirements increase with the number of transactions. For instance, with 350 transactions, the storage size was 39 KB, compared to 9 KB for 100 transactions. This growing storage requirement highlights the importance of efficient data management strategies to accommodate larger transaction volumes without compromising system performance.
- **Latency and Throughput:** The evaluation of latency and throughput revealed that the Ethereum blockchain-based system generally had higher latency and lower throughput than SQL databases. For example, the latency for verifying certificates on Ethereum was 7.12 milliseconds, compared to 2.63 milliseconds on SQL platforms. Throughput was also lower on Ethereum, with 6.38 Kbps compared to 10.57 Kbps on SQL platforms. These performance differences highlight the trade-offs between blockchain's security benefits and traditional database performance metrics.

In conclusion, integrating blockchain technology into IoT-based healthcare systems offers significant security, privacy, and data integrity advantages. The proposed system demonstrates a promising approach to addressing privacy

concerns and managing healthcare data effectively. The findings from this study provide valuable insights for developing more efficient and secure blockchain-based healthcare solutions in the future.

6. Conclusion and Future Work

6.1. Conclusion

This paper presents a novel blockchain-based IoT application to improve medical certificate management's security, privacy, and efficiency in healthcare systems. The proposed solution uses unique identification numbers for medical certificates. It incorporates a hybrid consensus mechanism combining Proof of Work (PoW) and Practical Byzantine Fault Tolerance (PBFT) for enhanced security and transaction speed. Privacy is safeguarded through homomorphic encryption and Non-Interactive Zero-Knowledge Proofs (NIZKPs), which enable secure medical certificate verification without revealing sensitive data. Integrating the Interplanetary File System (IPFS) also ensures scalable and efficient data storage. An Intrusion Detection System (IDS) has been added to monitor IoT traffic and detect security threats. Experimental results demonstrate the system's robustness, achieving an accuracy of 98.1% for the IDS. At the same time, blockchain evaluation metrics show low latency, high throughput, and enhanced security, positioning this solution as a transformative approach to secure healthcare management.

6.2. Limitation

The application faces high energy consumption and computational costs due to the PoW component. Advanced cryptography adds overhead, and IPFS may struggle with data retrieval and reliability issues.

6.3. Potential Industrial Applications

The proposed system offers significant industrial benefits across various sectors, particularly in healthcare:

- **Healthcare Providers**: Simplifies the management and verification of medical certificates, improving operational efficiency by reducing manual processes and paperwork.
- **Insurance Companies**: Enhances the validation of medical claims and records, minimizing fraud and reducing administrative errors using secure blockchain-backed certificates.
- **Medical Institutions**: Provides an efficient platform to manage large volumes of medical records, ensuring the security and privacy of sensitive patient information through advanced encryption and IDS monitoring.

6.4. Future Work

Future work will explore more efficient consensus algorithms, optimize smart contracts, improve IPFS integration, and conduct real-world testing to enhance scalability and performance.

6.5. Societal Applications

This system holds great potential for improving the security, privacy, and trust in healthcare operations:

- **Data Security and Privacy**: Ensures patient data remains confidential, addressing data breaches and unauthorized access concerns. Patients can have peace of mind knowing their medical records are secure.
- **Trust in Healthcare**: By leveraging transparent and secure certificate management, this system reduces fraudulent activities in medical certificate handling and increases trust among patients, healthcare providers, and insurers.
- **Wider Technological Adoption**: The adoption of blockchain in healthcare can inspire its broader integration across other sectors, enhancing overall data security and setting a standard for secure information sharing in various industries.

Chapter 7 Comparative Analysis with State-of-the-Art Intrusion Detection Systems

1. Introduction

With the expansion of Internet technology applications, the transformation of traditional industrial manufacturing systems, and the advancement of 5G communication technology, the Industrial Internet of Things (IIoT) is advancing towards grander scale and more profound network interconnections, playing an increasingly pivotal role in modern industrial reform [1]. As a bridge between traditional industrial systems and the new Internet industry [2], IIoT provides essential support for modernizing traditional industries. IIoT comprises physical service systems and digital devices such as sensors and actuators, managing data generated during industrial production to facilitate intelligent production and management, as shown in Figure 1. During the COVID-19 pandemic, IIoT proved vital in maintaining efficient production management and continuity amid isolation and economic turmoil [3-4].

However, the vast volumes of data stored in IIoT systems elevate the risk of cyber-attacks, posing significant threats to industrial management and production [5]. As IIoT grows, more industries integrate into its network, resulting in increasingly complex structures and heightened demands for robust defense capabilities [6]. Consequently, IIoT security has become more urgent, driving research on intrusion detection technology to the forefront. Intrusion detection, a proactive security technology, automatically detects and reports abnormal traffic affecting IIoT network security, including internal attacks, external attacks, and misoperations [7]. Upon detecting abnormal traffic, intrusion detection technology swiftly identifies the type of anomaly and implements defensive measures to block it before compromising the network system [8]. Thus, intrusion detection is crucial for protecting IIoT network security, ensuring data integrity, reliability, and normal industrial production [9-10].

Figure 64: Architecture of Industrial Internet of Things (IIoT)

Current IIoT intrusion detection technology faces two key challenges [11]: extracting and distinguishing network behavior. The Internet of Things (IoT) plays a significant role in daily life, connecting various devices, from smart

home appliances to industrial control systems [12]. However, the widespread use of IoT has led to a surge in botnet attacks, including denial of service (DoS), distributed denial of service (DDoS), reconnaissance, theft, Mirai, and Gafgyt, which threaten IoT device security and privacy [13] IoT networks are particularly vulnerable to botnet attacks due to weak security protocols and inadequately protected devices.

Botnet Intrusion Detection Systems (IDSs) have been developed to combat these threats. These systems detect botnets, block communication between compromised devices and command and control (C&C) servers, and alert network administrators [14]. Deploying botnet IDSs enhances IoT network security and protects against botnets and other cyber-attacks. Therefore, designing efficient IDSs capable of detecting and preventing attacks on low-power IoT devices is crucial. IDSs maintain IoT network security and integrity by identifying and neutralizing harmful network packets.

However, traditional IDSs, which often rely on data mining, fuzzy techniques, heuristics, or complex machine learning methods, typically need more accuracy and energy efficiency. This is a significant concern for IoT networks, consisting of numerous interconnected devices requiring efficient power usage. Addressing these challenges requires accurate, high-performing IDSs that are energy-efficient and capable of detecting a wide range of threats. Developing IDSs involves using feature-selection algorithms to identify the most relevant and efficient features from datasets and training models with these selected features. Detection accuracy increases by focusing on important features and eliminating irrelevant ones, and packet processing time decreases.

Feature selection plays a crucial role in developing lightweight IDSs [15]. Model-based feature selectors, such as importance- and correlation-coefficient methods and forward- and backward-sequential approaches, are commonly used due to their high performance and low false positive rates [16]. These approaches utilize regression algorithms, such as linear, lasso, logistic, or ridge, to calculate importance coefficients by analyzing the relationship between input features and output labels [17-18]. Based on these coefficients, feature-selection algorithms determine the most relevant and efficient features, enhancing IDS performance by eliminating irrelevant and inefficient ones.

1.1. Motivation

The rapid development of IoT network technology introduces vulnerabilities in data transmissions due to insecure network connections, demanding robust protection against unauthorized access, malicious activities, and potential security threats. Network intrusions jeopardize user data security and disrupt functionality, while sophisticated cyberattacks threaten data confidentiality, integrity, and availability. Current intrusion detection methods often need more accuracy and incur high computational costs, leading to suboptimal performance in detecting various attacks. These challenges drive us to innovate in intrusion detection and prevention. By implementing a deep learning-based approach, we aim to develop a robust and flexible model for early detection of network intrusions. Although advancements exist, current models need help with timely detection, scalability, and reliance on low-dimensional security data. To address these limitations, we propose an Efficient Feature Selection-based Intrusion Detection System using Artificial Intelligence (AI)-based Models to enhance security and effectively prevent network attacks.

The key contributions of this paper are as follows:

- We deployed 10 state-of-the-art Artificial Intelligence (AI)-based Intrusion Detection Models in an Industrial IoT environment.
- We utilized advanced wrapper-based feature selection methods, including forward-based, backward-based, and recursive feature elimination methods, to optimize feature selection, reduce computation time, and enhance the accuracy of intrusion detection in IoT networks.
- We rigorously tested the 10 AI-based Intrusion Detection Models using two well-known publicly available IoT/IIoT datasets, N_BaIoT and Edge-IIoT 2022.
- We conducted an in-depth performance analysis of all 10 models across both datasets, focusing on critical metrics such as accuracy, recall, F1-score, precision, G-mean, and specificity.
- We provided detailed insights into the effectiveness of each feature selection techniques methods, highlighting their impact on the models' performance.

This research focuses on applying AI-driven intrusion detection in Industrial IoT environments. By leveraging cutting-edge feature selection methods and performance evaluation matrices, the study aims to develop high-performing, energy-efficient IDSs capable of defending against a wide range of cyber threats. The context of this research is the growing complexity and interconnectedness of IIoT networks, which necessitate robust and scalable security solutions.

1.2. Paper Structure

This research paper is structured as follows: **Section 2** delves into the foundational aspects of intrusion detection, exploring its various types, security and privacy issues, and the different cyber-attacks prevalent in IoT and IIoT. **Section 3** conducts an extensive literature review, critically examining existing techniques for detecting cyber-attacks and identifying these methods' significant challenges. **Section 4** details the problem statement, dataset description, preprocessing techniques, feature selection methods, and the AI-based models employed for intrusion detection. **Section 5** outlines the experimental design, presents an in-depth analysis of the results, and compares the performance of our AI-based models against current state-of-the-art approaches. Finally, **Section 6** concludes the paper on the study findings and suggests potential directions for future research.

2. Background and Related Work

This section establishes the groundwork for comprehending IIoT networks and their associated Intrusion detection strategies.

2.1. Intrusion Detection System (IDS)

Intrusion Detection Systems (IDS) are essential components in cybersecurity. They serve as vigilant guardians that monitor network or system activities to identify and respond to potential security threats. IDS is critical in protecting organizations' digital assets by detecting malicious activities, unauthorized access attempts, and policy violations. One of the primary types of IDS is the signature-based IDS. This system compares observed events with predefined signatures or patterns of known threats. When a match is found between the observed activity and a signature in the database, the IDS generates an alert to notify system administrators or security personnel. Signature-based IDS excel at detecting well-known attacks that have identifiable patterns or signatures, making them an effective defense against known threats. In contrast, anomaly-based IDS take a different approach to threat detection, rather than relying on predefined signatures, anomaly-based IDS establishes a baseline of normal network or system behavior by analyzing historical data. These systems continuously monitor network traffic or system activities and flag any deviations from the established baseline as potential security threats. Anomaly-based IDS are handy for detecting new or unknown threats that do not have existing signatures, making them valuable tools for identifying unusual or suspicious behavior that may indicate a security breach [19]. Figure 2 shows the Intrusion Detection System mechanism and its type.

Organizations can establish a comprehensive defense strategy addressing cyber threats using signature-based and anomaly-based IDS. Signature-based IDS protects against known attacks, while anomaly-based IDS offers a proactive defense against emerging threats and zero-day vulnerabilities. Together, these IDS work synergistically to enhance an organization's overall security, helping to mitigate risks and protect critical assets from cyber threats in an ever-evolving digital landscape. The core functionalities of an Intrusion Detection System (IDS), extensively studied, encompass a series of crucial steps designed to safeguard networks and systems from potential security threats, with its components elaborated upon to detail these protective measures [19] as follows:

- **Network Monitoring:** This component involves continuously monitoring network traffic to capture packets containing vital network-related information. The IDS can identify threats and security vulnerabilities by analyzing network packets and flows.
- **Data Collection Techniques:** The IDS employs various data collection techniques to gather information about target systems and network activities. These may involve using network commands, tools like "Wireshark" for packet sniffing, or querying domain details using tools such as nslookup.

- **Packet Analysis:** In this stage, the IDS scans network packets to uncover potential security threats, such as unauthorized access attempts, data breaches, or malware injections. The IDS can detect anomalies indicative of malicious activities by analyzing packet contents.
- **Signature Identification and Storage:** Following packet analysis, the IDS identifies attack patterns or signatures of known threats. These signatures are stored in a centralized database, enabling the IDS to efficiently recognize and respond to similar attack patterns in the future.
- **Alert Generation Mechanisms:** When an attack pattern is detected, the IDS promptly generates alerts or alarms to notify security administrators. These alerts provide critical information about the nature of the threat, facilitating rapid response and mitigation efforts.

Despite its capability to scrutinize network packet contents for detecting and quantifying attacks, Intrusion Detection Systems (IDS) exhibit several limitations:

- IDS cannot preemptively block or prevent identified attacks solely based on pattern recognition or signature matching from a database. Integration with additional security mechanisms, such as Intrusion Prevention Systems, is necessary to execute blocking actions.
- While IDS conducts thorough network analysis and monitors network activity, it needs the capability to take immediate action upon attack detection. Consequently, continuous intervention by a security officer or administrator is required to respond effectively to identified threats.
- IDS exhibits inefficiency in processing encrypted network packets, necessitating specialized networking tools for examination. This may render system resources vulnerable until intrusion detection occurs.
- The prevalence of false positives generated by IDS significantly impacts system efficiency and reliability.
- Regularly updating the attack signature database is essential to ensure the IDS remains effective against evolving threats.
- As identified in prior research, IDS vulnerabilities extend to protocol-based attacks.

2.2. Industrial Internet of Things (IIoT)

The Industrial Internet of Things (IIoT) represents a pivotal evolution in industrial processes, blending traditional manufacturing with cutting-edge digital technologies to optimize efficiency, productivity, and connectivity in Industry 5.0. Unlike its predecessors, IIoT integrates intelligent sensors, devices, and machinery into interconnected networks, enabling real-time data collection, analysis, and decision-making [20]. This interconnectedness facilitates seamless communication between machines, systems, and humans, unlocking new avenues for automation, predictive maintenance, and resource optimization in industrial settings.

However, adopting IIoT introduces new cybersecurity challenges, particularly concerning intrusion detection and prevention. As industrial systems become increasingly interconnected and digitized, they become more susceptible to cyber threats, including unauthorized access, data breaches, and system tampering [21]. Cyber-attacks targeting IIoT infrastructure can have severe consequences, ranging from operational disruptions and production downtime to compromised safety and financial losses.

It is crucial to delve into the specific cybersecurity implications of IIoT and explore effective intrusion detection mechanisms tailored to industrial environments. This involves investigating advanced anomaly detection techniques, machine learning algorithms, and AI-driven solutions capable of identifying and mitigating cyber threats in real-time. Additionally, understanding the unique characteristics of IIoT networks, such as legacy systems integration, resource constraints, and critical infrastructure dependencies, is essential for designing robust intrusion detection systems capable of safeguarding Industry 5.0 ecosystems against evolving cyber threats.

Security and privacy issues in IoT (Internet of Things) and IIoT (Industrial Internet of Things) environments are multifaceted and pose significant challenges to data and systems integrity, confidentiality, and availability [21]. Some key issues include:

- **Device Vulnerabilities:** IoT and IIoT devices often have limited computing resources and may lack robust security features, making them vulnerable to attacks such as malware infections, firmware exploits, and physical tampering.
- **Data Security:** Data transmitted between IoT/IIoT devices and backend systems may be susceptible to interception and tampering, raising concerns about data confidentiality and integrity. Additionally, data stored on devices or in the cloud may be at risk of unauthorized access or data breaches.
- **Network Security:** Inadequately secured communication channels between IoT/IIoT devices and backend systems can be exploited by attackers to intercept data, launch man-in-the-middle attacks, or disrupt communication through denial-of-service (DoS) attacks.
- **Privacy Concerns:** IoT and IIoT devices often collect vast amounts of data about users, their behaviors, and their environments. However, this data's indiscriminate collection and sharing raise significant privacy concerns, especially regarding personally identifiable information (PII) and sensitive data.
- **Supply Chain Risks:** The global nature of IoT/IIoT supply chains introduces security risks, including counterfeit components, insecure firmware/software updates, and vulnerabilities introduced during manufacturing or distribution processes.
- **Regulatory Compliance:** Compliance with data protection regulations such as the General Data Protection Regulation (GDPR) and industry-specific standards (e.g., NIST, ISO 27001) is a significant challenge for IoT/IIoT deployments, especially considering the diverse regulatory landscape across different regions and industries.
- **Lifecycle Management:** Effective management of IoT/IIoT device lifecycles, including provisioning, monitoring, patching, and decommissioning, is critical for maintaining security posture and mitigating risks associated with outdated or unsupported devices.
- **Interoperability Challenges:** Integrating diverse IoT/IIoT devices and systems from different vendors often leads to interoperability challenges, which can introduce security vulnerabilities and complicate security management and monitoring efforts.

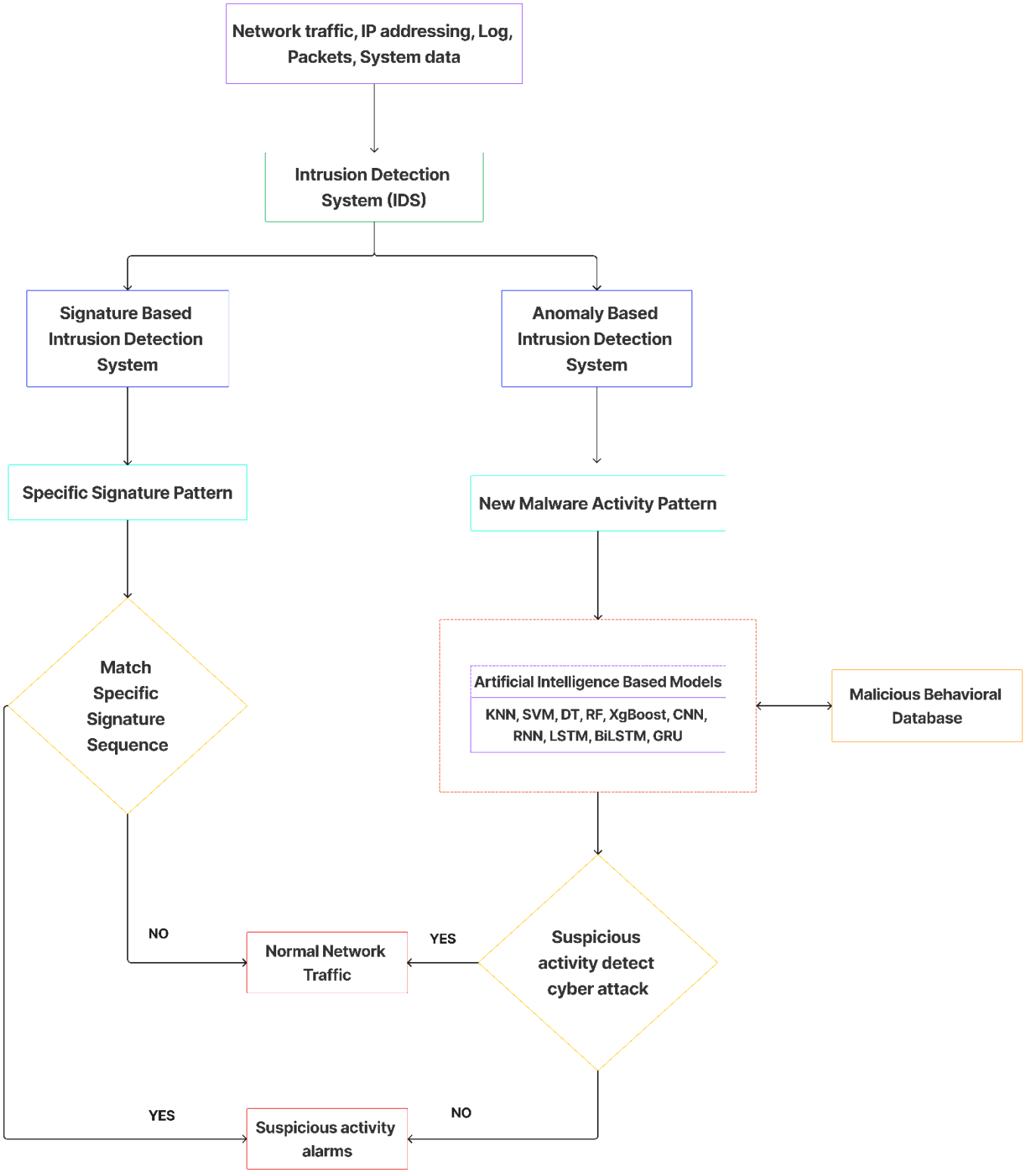


Figure 65: Intrusion Detection System mechanism and its type

Table 1 provides a comprehensive overview of the significant security and privacy issues prevalent in IoT/IIoT networks, detailing their impacts on the CIA triad, the affected layers of the IoT/IIoT architecture, and the recommended countermeasures to mitigate these risks. Addressing these security and privacy issues requires an advanced approach encompassing technical and regulatory measures. This includes implementing robust security protocols, encryption mechanisms, access controls, and intrusion detection systems and enhancing the culture of security awareness and compliance within systems.

Table 38: Security and Privacy Issues in IoT/IIoT Networks: Impact, Affected Layers, and Countermeasures

Security and Privacy Issues	CIA triad	Impact on confidentiality	Impact on Integrity	Impact on availability	Layered of IoT/IIoT architecture	Countermeasures
Device Vulnerabilities	Vulnerability to malware, firmware exploits, physical tampering	✓	✓	✓	Device/Endpoint Layer	Regular firmware updates, security patches, and device hardening
Data Security	Data breaches, unauthorized access, data tampering	✓	✓	✓	Communication Layer	End-to-end encryption, data anonymization, access controls
Network Security	Data interception, man-in-the-middle attacks, denial-of-service	✓	✓	✓	Network Layer	Secure communication protocols, network segmentation
Privacy Concerns	Unauthorized data collection, privacy violations	✓	✗	✗	Application Layer	Data minimization, user consent, privacy-enhancing technologies
Supply Chain Risks	Counterfeit components, insecure updates, vulnerabilities	✓	✓	✓	Device/Endpoint Layer	Supply chain audits, vendor risk assessments
Regulatory Compliance	Non-compliance fines, legal liabilities	✓	✓	✓	Data Layer	Compliance frameworks, data governance policies
Lifecycle Management	Security risks from outdated devices, unpatched vulnerabilities	✓	✓	✓	Device/Endpoint Layer	Device monitoring, patch management, end-of-life policies
Interoperability Challenges	Security vulnerabilities, integration issues	✓	✓	✓	Communication Layer	Standards compliance, interoperability testing

2.3. Type of Cyber-attack on IoT/IIoT Environment

The security of IoT/IIoT environments is paramount due to the increasing prevalence and sophistication of cyber-attacks, which target different layers of the architecture and pose significant threats to confidentiality, integrity, and availability, as represented in Table 2.

Table 39: Types of Attacks in IoT/IIoT Architecture: Description, Countermeasures, and Impact on CIA

Types of attacks	Layer in IoT /IIoT architecture	Description	Countermeasures	Impact on CIA
Denial of Service (DoS)	Network Layer	Overloads system, disrupting service	Implement network traffic filtering and rate-limiting	C: ✗ I: ✗ A: ✓
Man-in-the-Middle (MitM)	Communication Layer	Intercepts and alters communication	Use encryption and authentication protocols	C: ✓ I: ✓ A: ✗
Phishing	Application Layer	Deceives users into sharing sensitive information	User awareness training and email filtering	C: ✓ I: ✗ A: ✗
Malware	Device/Endpoint Layer	Infects devices, compromising security	Install antivirus software and regular updates	C: ✓ I: ✓ A: ✓
Distributed Denial of Service (DDoS)	Network Layer	Floods network with traffic, causing system failure	Deploy DDoS protection services and load balancers	C: ✗ I: ✗ A: ✓
Insider Threat	Data Layer	Malicious actions by authorized users	Implement role-based access control and monitoring	C: ✓ I: ✓ A: ✓
Brute Force Attack	Authentication Layer	Repeated login attempts to guess credentials	Enforce strong password policies and account lockout	C: ✗ I: ✗ A: ✓
Firmware Exploitation	Device/Endpoint Layer	Exploits vulnerabilities in device firmware	Regular firmware updates and vulnerability scanning	C: ✓ I: ✓ A: ✓

Replay Attack	Communication Layer	Replays previously captured data to gain unauthorized access	Implement timestamping and secure communication protocols	C: ✓ I: ✓ A: ✗
SQL Injection	Data Layer	Injects malicious SQL commands into databases	Use parameterized queries and input validation	C: ✓ I: ✓ A: ✗

2.4. Literature Review

This section investigates the utilization of Machine Learning (ML) and Deep Learning (DL) methodologies in previous research for designing Intrusion Detection Systems (IDSs), which play a critical role in protecting computer networks against Cyber threats.

2.4.1. Machine Learning for IDS in IIoT

Intrusion Detection Systems (IDS) in IoT environments have witnessed a surge of research endeavors aimed at fortifying the security posture of interconnected devices. This literature review critically examines pivotal contributions in this domain, elucidating the evolving landscape of IDS for IoT and highlighting innovative approaches to address its inherent challenges, as shown in Table 3.

Talukder et al. [22] introduced MLSTL-WSN, a novel IDS leveraging machine learning (ML) techniques in Wireless Sensor Networks (WSNs). By employing SMOTE Tomek to address class imbalance, their methodology demonstrates enhanced detection accuracy and robustness, addressing a crucial concern in IoT deployments. Alqahtani et al. [23] explored cyber intrusion detection utilizing machine learning classification techniques. While not explicitly IoT-focused, their insights into machine learning algorithms' efficacy lay foundational groundwork for IDS in IoT ecosystems, underscoring the importance of leveraging advanced computational methods for threat detection. Meryem and Ouahidi [24] proposed a hybrid IDS integrating machine learning algorithms, catering to the intricacies of modern cyber threats. Their approach showcases the synergistic potential of combining multiple detection mechanisms, vital for combating sophisticated intrusion attempts targeting IoT infrastructures. Asif et al. [25] devised a MapReduce-based intelligent model for intrusion detection, leveraging machine learning in IoT environments. Their work exemplifies the integration of distributed computing paradigms with machine learning techniques to address scalability challenges in large-scale IoT deployments. Gad et al. [26] delved into IDS for Vehicular Ad Hoc Networks (VANETs), employing machine learning on the ToN-IoT dataset. Their research underscores the importance of tailored intrusion detection mechanisms for specific IoT applications, emphasizing the need for context-aware security solutions.

Bangui et al. [27] proposed a hybrid machine-learning model for intrusion detection in VANETs, highlighting the significance of adaptability and resilience in vehicular IoT environments. Their approach showcases the efficacy of combining diverse machine-learning techniques to enhance detection accuracy amidst dynamic network conditions. Alhajjar et al. [28] investigated adversarial machine learning in network IDS, shedding light on the emerging threat landscape of sophisticated attacks. Their research underscores the importance of incorporating adversarial robustness into IDS frameworks to mitigate evolving cyber threats targeting IoT infrastructures. Sarhan et al. [29] focused on feature extraction for machine learning-based IDS in IoT networks, addressing the challenge of extracting relevant features from heterogeneous IoT data sources. Their work lays the groundwork for developing context-aware intrusion detection mechanisms tailored to IoT environments.

Liu et al. [30] proposed an intrusion detection approach for imbalanced network traffic, utilizing machine learning and deep learning techniques. Their research emphasizes the importance of addressing the class imbalance in IoT datasets to prevent detection biases and ensure comprehensive threat coverage. Singh et al. [31] introduced AutoML-ID, an automated machine-learning model for intrusion detection in Wireless Sensor Networks (WSNs). Their methodology streamlines the model development process, offering a scalable solution for deploying IDS in resource-constrained IoT environments. Zou et al. [32] presented HC-DTTSVM, a novel intrusion detection method based on decision tree twin support vector machine and hierarchical clustering. Their approach showcases the potential of hybrid machine learning techniques in enhancing detection accuracy and scalability in IoT environments. Louk and Tama [33] proposed Dual-IDS, a bagging-based gradient-boosting decision tree model for network anomaly intrusion detection. Their research underscores the importance of ensemble learning techniques in enhancing detection robustness and resilience against evolving cyber threats. Mohiuddin et al. [34] explored hybridized meta-heuristic techniques for intrusion detection, integrating the Weighted XGBoost Classifier. Their approach demonstrates the

efficacy of meta-heuristic optimization in enhancing the performance of machine learning-based IDS in IoT environments. Zouhri et al. [35] evaluated the impact of filter-based feature selection in intrusion detection systems, highlighting the importance of feature engineering in enhancing detection accuracy and reducing computational overhead in IoT deployments. Amaouche et al. [36] proposed IDS-XGbFS, an intelligent intrusion detection system utilizing XGBoost with a recent feature selection for VANET safety. Their methodology showcases the integration of advanced machine learning algorithms with feature selection techniques tailored to IoT-specific applications.

Table 40: A summary of Intrusion Detection System based on Machine Learning (ML) Techniques

References	Purpose	Methodology	Dataset used	Feature extraction technique	Result	Advantages	Disadvantages
[22]	Intrusion detection system for WSN	DT,RF, MLP, KNN, XGB, LGB	Wireless Sensor Network dataset	SMOTE_Tomek link	Acc= 99.70%	Addresses imbalanced data	Limited to WSNs, requires investigation on other network types
[23]	Intrusion detection system for Cyber security	Bayesian Network, NB, DT, RF, ANN	KDD-99	-	Acc= 94%	Compares multiple algorithms, offers flexibility	Relies on the unspecified dataset, limits the generalizability
[24]	Hybrid Intrusion detection system	KNN, NB, SVM, Logistic Regression	NSL-KDD	-	Acc= 98.77%	Lacks details on the specific hybrid approach	Requires more information on the hybrid method
[25]	Intrusion detection system for intelligent modeling	Map reduced-based intelligent model-IDS	Kaggle ML repository	-	Acc= 97.6%	Efficient for large datasets, scalable	Relies on unspecified dataset, limited details on the model
[26]	Intrusion Detection System for Vehicular Adhoc Networks	LR, NB, KNN, DT, Adaboost, Xgboost, RF, SVM	TON_IOT	Chi-Square and SMOTE	Acc= 99.1%	Focuses on VANETs, ToN-IoT specific	Limited applicability outside VANETs
[27]	Hybrid model Intrusion detection in VANET	SVM, Bayesian coresets, CNN, MLP, RF, Weighted-KNN	CIC-IDS-2017	Weighted clustering	Acc= 96.93%	Offers potentially better accuracy	Requires more information on the specific hybrid model
[28]	Network-based Intrusion detection system	Generative advertised network	NSL-KDD, UNSW-NB-15	PSO, GA	Acc= 99%	Improves IDS robustness against adversarial attacks	Enhances security, potentially computationally expensive
[29]	Intrusion detection system in IoT network	DFF, CNN, RNN, DT, LR, NB	UNSW-NB-15, TON-IoT, CIC-IDS-2018	PCA, AE, LDA	Acc= 96.11%	Improves intrusion detection accuracy in IoT	Addresses feature selection for IoT networks, limited details on specific techniques.
[30]	Intrusion detection system on network traffic-based	RF, SVM, Xgboost, LSTM, Alex-net, Mini-VGGnet, DSSTE	CIC-IDS-2018, NSL-KDD	Edited Nearest Neighbor	Acc= 96.99%	Effective for imbalanced network traffic intrusion detection	Handles imbalanced data and explores different techniques
[31]	Intrusion detection system using WSN	SVR, GPR, BDT, Ensemble regression, kernel regression, LR, BO	Synthetically generated simulated dataset	K-barriers	R=0.93	Achieves good accuracy with AutoML	Automates model selection reduces human effort
[32]	Intrusion detection system	HC-DTTWSVM	NSL_KDD, UNSW-NB - 15	Hierarchical clustering	Acc= 85.95%	Offers potentially better accuracy and reduced false positives	Relies on unspecified dataset, requires investigation on generalizability
[33]	Intrusion detection system	GBM, Light GBM, Catboost, Xgboost	NSL_KDD, UNSW-NB - 15, HIKARI-2021	-	Acc= 91.75%	Effective for anomaly detection, leverages ensemble learning	Relies on unspecified dataset, limits the generalizability

[34]	Intrusion detection system	xgboost	UNSW-NB-15, CIC-IDS-2018	Modified wrapper-based whale sine-cosine	Acc= 99%	Combines meta-heuristics for optimization and XGBoost for classification	Relies on unspecified dataset, limited details on meta-heuristics
[35]	Intrusion detection system	MLP, SVM, Xgboost, RF	CIC-IDS-2018, CIC-IDS-2017, TON-IoT	Relieff, Pearson correlation, mutual information, ANOVA, chi-square	Acc= 98%	Identifies the importance of feature selection, improves efficiency	Relies on unspecified IDS method and dataset, limited to filter-based selection
[36]	Intrusion detection system	xgboost	NSL-KDD, 5-routing metrics dataset	Boruta, ADASYN	Acc= 99%	Focuses on VANET security, leverages XGBoost, and feature selection	Limited applicability outside VANETs, relies on unspecified recent feature selection technique

2.4.2. Deep Learning for IDS in IIoT

In recent years, the proliferation of Internet of Things (IoT) devices has brought unprecedented challenges in ensuring the security and integrity of network infrastructures. Amidst these challenges, the development of robust intrusion detection systems (IDS) has emerged as a critical area of research. This literature review critically evaluates and synthesizes the critical contributions of IDS for IoT environments, focusing on novel deep-learning (DL) techniques and their applications, as shown in Table 4.

Maddu and Rao [37] introduced a pioneering approach to network intrusion detection and mitigation in Software-Defined Networking (SDN) using deep learning models. By harnessing the power of deep learning, their methodology showcases promising results in enhancing the security of SDN-based IoT networks. Similarly, Sharma et al. [38] proposed an anomaly-based IDS leveraging deep learning techniques tailored explicitly for IoT attacks. Their work underscores the significance of anomaly detection in fortifying IoT ecosystems against malicious activities. Song and Ma [39] extended the applicability of deep learning to edge-enabled IoT environments by introducing a federated attention neural network for intrusion detection. Their federated approach addresses decentralized IoT networks' inherent challenges, offering a scalable solution with improved detection accuracy. Nanjappan et al. [40] proposed DeepLG SecNet, a sophisticated IDS framework that integrates deep Long Short-Term Memory (LSTM) and Gated Recurrent Unit (GRU) architectures within a secure IoT network. This innovative amalgamation of deep learning and network security mechanisms demonstrates remarkable efficacy in detecting intrusions amidst the complexities of IoT environments. Devendiran and Turukmane [41] devised Dugat-LSTM, a novel IDS system harnessing chaotic optimization strategies in conjunction with deep learning. By integrating chaotic optimization into the learning process, their approach exhibits enhanced robustness and adaptability in detecting intrusions within IoT networks. Aljohani et al. [42] introduced a deep learning-based IDS tailored for Smart Grids, showcasing the applicability of AI-driven solutions in safeguarding critical infrastructure. Their work underscores the imperative of integrating advanced technologies to fortify the resilience of IoT-enabled systems against cyber threats. Sharma et al. [43] contributed to the advancement of explainable artificial intelligence (XAI) in intrusion detection for IoT networks. Their deep learning-based approach enhances detection accuracy and provides interpretability, which is crucial for understanding the rationale behind intrusion alerts. Kethineni and Pradeepini [44] proposed a hybrid deep-learning framework for intrusion detection in IoT-based smart farming. By amalgamating different deep learning architectures, their framework offers a comprehensive solution tailored to the unique characteristics of agricultural IoT deployments. Nandanwar and Katarya [20, 21] made significant strides in transfer learning and deep learning-enabled IDS for IoT environments. Their models demonstrate remarkable adaptability and performance in diverse IoT scenarios, underscoring the versatility of deep learning techniques in intrusion detection. Yin et al. [45] and Imran et al. [46] laid the groundwork for leveraging recurrent neural networks (RNNs) and deep learning in general for intrusion detection. Singh et al. [75] addresses the critical issue of class imbalance in activity recognition systems designed for multi-resident smart homes. Class imbalance, where frequent activities overshadow rarer ones, poses significant challenges to the accuracy and reliability of deep learning models, particularly Long Short-Term Memory (LSTM) and Bidirectional LSTM (BiLSTM) networks, which are adept at handling temporal data. The authors investigate various techniques to mitigate this issue, including data-level approaches such as oversampling and undersampling, and algorithm-level methods like cost-sensitive learning. Using the ARAS and CASAS-Kyoto datasets, the study evaluates

the effectiveness of these methods through metrics such as Exact Match Ratio (EMR), balanced accuracy, and F1-score. The results highlight the superior performance of cost-sensitive learning in most scenarios, although the impact of imbalance-handling techniques varied across datasets due to differences in their characteristics. Furthermore, the paper explores the role of explainable AI (XAI) in enhancing the transparency and trustworthiness of these models, particularly in high-stakes applications like elder care and health monitoring. The study concludes by emphasizing the importance of hybrid approaches and improved interpretability techniques to address class imbalance effectively while maintaining model accountability. This work significantly contributes to the advancement of explainable AI and the development of robust multi-resident activity recognition systems in smart home environments.
While their contributions predate some of the works mentioned above, they provide foundational insights into the efficacy of deep learning in addressing the evolving threat landscape of IoT environments.

Table 41: A summary of Intrusion Detection System based on Deep Learning(DL) Techniques

References	Purpose	Methodology	Dataset used	Feature extraction technique	Result	Advantages	Disadvantages
[37]	Network intrusion detection and mitigation in SDN	DCGAN	Edge-IIoT	Center-net based approach	Acc= 99.31%	Integrates mitigation with detection focuses on SDN	Relies on unspecified deep learning models and dataset
[38]	Anomaly-based Network IDS for IoT	DNN-GAN	UNSW-NB-15	Filter based method	Acc= 91%	Focuses on IoT security, leverages deep learning	Relies on unspecified deep learning techniques and dataset
[39]	Intrusion detection system for edge-enabled Internet of Things	FedACNN	UNSW-NB-15	-	Acc= 99%	Focuses on edge computing and federated learning	Relies on unspecified dataset, limited details on the approach
[40]	Enhanced IDS in IoT environment	DEepLG secnet	NSL-KDD, BOT-IoT	CNN	Acc= 98.92 %	Utilizes multiple deep learning techniques (LSTM, GRU)	Relies on unspecified dataset; details on secure network integration unclear
[41]	Network intrusion detection system	Dugat-LSTM	ToN-IoT, NSL-KDD	Chaotic optimization strategy	Acc= 99.65%	Integrates optimization for potentially better results	Relies on unspecified dataset, details on chaotic optimization strategy unclear
[42]	Cyber intrusion detection for smart grid	DLNN	13 bus system	-	Acc= 90%	Focuses on smart grid security, integrates mitigation	Relies on unspecified deep learning model and dataset
[43]	Intrusion detection system in IoT network	DNN+ explainable AI	NSL-KDD, UNSW-NB-15	Filter based approach	Acc= 99.2%	Focuses on interpretability for better understanding	Relies on unspecified deep learning model and dataset
[44]	Intrusion detection in IoT-based smart framing	BiGRU + CNN + attention mechanism	ToN-IoT, APA-DDoS	WHO algorithm	Acc= 99.71%	Focuses on a specific IoT application (smart farming)	Relies on unspecified deep learning models and dataset
[20]	Intrusion detection system in IoT environment	TL-biLstm	N_BaIoT	CNN	Acc= 99.52%	Utilizes transfer learning for potentially better performance	Relies on unspecified dataset, limited details on specific techniques
[21]	Intrusion detection system for industrial IoT environment	CNN-GRU	N_BaIoT	-	Acc= 99.75%	Focuses on industrial IoT security	Relies on unspecified deep learning model and dataset
[45]	Intrusion detection system	RNN	NSL-KDD	-	Acc= 99.81%	Pioneering work in deep learning IDS explores RNNs	Limited to older datasets (NSL-KDD), may not reflect current threats

[46]	Intelligent and efficient network intrusion detection system	Stacked non-symmetric deep autoencoder	KDD CUP-99	SVM	Acc= 99.65%	Efficient approach with good performance explores deep learning	Relies on a single dataset (UNSW-NB15), limits the generalizability

2.4.3. Feature selection based Method

Selecting the most relevant features in datasets is vital for developing IDSs that can effectively represent network traffic and detect potential threats. This importance is magnified in IoT networks, where the vast data volume and intricate architecture pose unique monitoring challenges [47,48]. Feature-selection methods for IDSs in IoT networks are categorized into three main types: wrapper, filter, and hybrid methods [49].

Wrapper methods evaluate feature subsets using a specific classifier, selecting the subset that delivers the best performance. For example, the forward selection algorithm starts with an empty feature set and incrementally adds features until reaching the desired number, while the backward elimination algorithm begins with a full feature set and removes features one by one [50]. In contrast, filter methods apply predefined criteria to select features independently of any classifier. Notable examples include the chi-square test, which measures feature independence from class labels, and mutual information, which assesses the dependency between features and class labels [51]. Hybrid methods combine the strengths of both wrapper and filter approaches. The fast correlation-based feature selection (FCBF) algorithm uses symmetrical uncertainty to identify relevant features, while the sequential forward floating selection (SFFS) algorithm employs a wrapper approach to select features based on classifier accuracy [52].

Extensive research has explored these feature-selection techniques for IDSs in IoT networks. Shafiq et al. [53] demonstrated that the forward selection algorithm, a wrapper method, outperformed filter methods on an IoT network traffic dataset. Louk et al. [54] found that the FCBF algorithm, a hybrid method, achieved the highest precision and the lowest false alarm rate in their evaluation of hybrid methods on an IoT network dataset. Feature selection is crucial for IDSs in IoT networks, with various proposed and evaluated methods, including wrapper, filter, and hybrid approaches. However, further research is needed to determine the most effective feature-selection methods for different datasets, as results can vary significantly across datasets [55].

2.5. Statistical Analysis of Publicly Available Datasets

In this section, we perform a detailed statistical analysis of various publicly available datasets, focusing on their characteristics, distribution, and suitability for enhancing intrusion detection systems in IoT environments, as represented in Table 5.

Table 42: Detailed Statistical Analysis of Benchmark Datasets for Evaluating Intrusion Detection Systems in IoT/IIoT

Datasets	Year	Size	No. of records	Labeled	Class distribution (Normal/Attack)	Attack type	Features	Protocol used	Data collection source	Nature of dataset	IoT-based
KDD-99	1999	18 MB	4898430	Labeled	60%/40%	4	41	TCP, UDP, ICMP	Simulated network traffic	Balanced	No
NSL-KDD	2009	22 MB	148517	Labeled	80%/20%	22	41	TCP, UDP, ICMP	Filtered KDD-99	Imbalanced	No
UNSW-NB-15	2015	100 GB	2540044	Labeled	83%/17%	9	49	TCP, UDP	Captured network traffic	Imbalanced	No
N-BaIoT	2018	3.3 GB	7062606	Labeled	87%/13%	10	115	TCP, UDP, HTTP	Industrial network traffic	Imbalanced	Yes
BOT-IoT	2019	69.3 GB	72 million	Labeled	82%/18%	6	31	TCP, UDP, HTTP	Botnet traffic	Imbalanced	Yes

TON-IoT	2021	209.17 MB	408203	Labeled	77%/23%	9	43	TCP, UDP, HTTP	Real-world IoT traffic	Imbalanced	Yes
Edge_IIoT	2022	5.2 GB	1909671	Labeled	78%/22%	14	61	TCP, UDP, HTTP	Building automated network traffic	Imbalanced	Yes
CIC-IDS-2017	2017	50 GB	3 million	Labeled	80%/20%	14	77	TCP, UDP, ICMP	Real-world network traffic	Imbalanced	No
CIC-IDS-2018	2018	450 GB	16233002	Labeled	80%/20%	7	80	TCP, UDP, ICMP	Real-world network traffic	Imbalanced	No
CIC-IDS-2019	2019	Approx. 500 GB	80000000	Labeled	81%/19%	8	80+	TCP, UDP, ICMP	Real-world network traffic	Imbalanced	No

2.6. Research Gaps and Limitations

In addressing the current landscape of Intrusion Detection Systems (IDS) for Industrial IoT environments, several research gaps and limitations have been identified that impede the development of more robust and efficient security solutions as follows:

- **Systematic Dataset Unavailability:** The absence of up-to-date datasets reflecting modern network attacks hinders the development of efficient IDS models capable of detecting zero-day attacks.
- **Imbalanced Dataset Detection Accuracy:** Most IDS methodologies exhibit lower detection accuracy for rare attack types due to imbalanced datasets, necessitating the development of balanced and comprehensive datasets.
- **Real-World Environment Performance:** The need for real-world testing for IDS methodologies questions their effectiveness outside controlled lab environments, highlighting the need for real-time validation.
- **Resource-Intensive Models:** Complex IDS models consume significant processing time and computing resources, underscoring the need for efficient feature selection algorithms to reduce overhead and improve performance.
- **Lightweight IDS for IoT:** Developing lightweight IDS models for resource-constrained IoT sensor nodes remains challenging, requiring models that balance computational efficiency and high intrusion detection rates.

3. Methodology

This section provides a comprehensive overview of the proposed methodology, detailing each critical step in developing our AI-based intrusion detection system. The proposed flow of Artificial Intelligence-based Intrusion Detection System models on different Wrapper-based Feature Selection Methods is shown in Figure 3. We begin by discussing the problem statement, identifying the challenges and objectives of enhancing intrusion detection in IoT/IIoT environments. Following this, we present the datasets utilized in this study, namely the N-BaIoT and Edge-IIoT-2022 datasets, elaborating on their characteristics and relevance. The subsequent data preprocessing step is explained, which includes handling missing values, normalization, and data transformation to ensure the datasets are suitable for model training. We then describe the feature selection methods employed, such as Forward Selection, Backward Selection, and Recursive Feature Elimination (RFE), to identify the most significant features contributing to accurate intrusion detection. Finally, we delve into the AI-based models implemented, detailing their architectures, training processes, and evaluation metrics to demonstrate their efficacy in detecting and mitigating cyber-attacks.

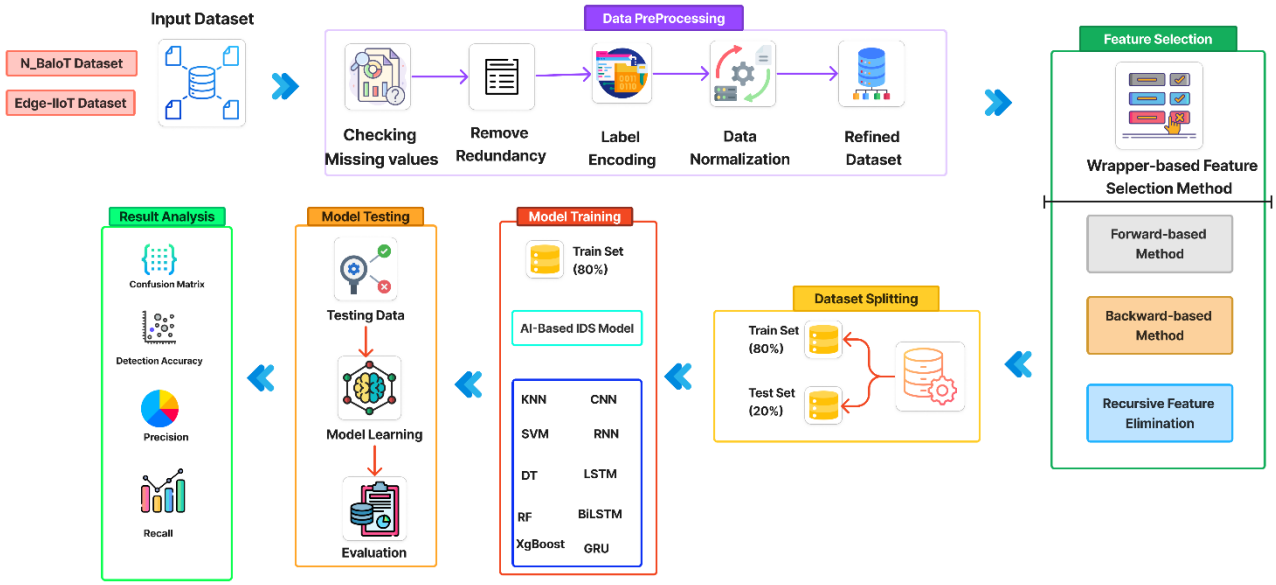


Figure 66: Proposed Flow of Artificial Intelligence-based Intrusion Detection System models on different Wrapper-based Feature Selection Methods

3.1. Problem Statement

IoT security poses significant network threats, particularly in identifying and neutralizing malicious activities. Rapid and accurate detection of unauthorized or irregular network traffic is crucial for mitigating potential intrusions or attacks. Advanced Intrusion Detection Systems (IDS) must efficiently distinguish between benign and malicious behaviors to ensure immediate detection and intervention, especially in fast-paced, resource-constrained wireless environments. Unique threats such as Flooding, Injection, and Impersonation attacks require tailored detection and countermeasure strategies. This study aims to enhance IDS technologies, strengthening wireless network defenses against sophisticated threats. We address the challenge of high-dimensional data in IoT environments, where irrelevant features can degrade IDS accuracy and performance. We aim to optimize feature sets and improve IDS effectiveness by employing feature selection techniques with AI-based models. The N_BaIoT and Edge-IIoT-2022 datasets, which capture current cyber-attacks, serve as the basis for our evaluation. Our approach seeks to refine IDS capabilities, ensuring robust and precise threat detection in IoT networks.

3.2. Dataset Description

We evaluated the model using two publicly available datasets: N_BaIoT and Edge-IIoT-2022.

3.2.1. N_BaIoT dataset

In this study, we used the N_BaIoT dataset, a publicly accessible compilation of network traffic data sourced from nine distinct IoT devices, as detailed in Table 6. This dataset, encompassing 115 features, has been pivotal in prior research on botnet detection in IoT/IIoT environments [56].

Table 43: Device Inventory name and Classification in the N-BaIoT Dataset

Device Name	Device Type
Danmini	Doorbell
Ennio	

Ecobee	Thermostat
Phiplips B120N/10	Baby Monitor
Provision PT-737E	Security Camera
Provision PT-838	
SimpleHome XCS7-1002-WHT	Security Camera
SimpleHome XCS7-1003-WHT	
Samsung SNH 1011 N	Webcam

The N_BaIoT dataset captures ten distinct attack classes, including the notorious BASHLITE and Mirai botnet attacks, alongside benign traffic. BASHLITE attacks encompass malicious activities such as scanning for vulnerable devices, sending spam data, UDP flooding, and combined spam and connection attempts to specified IP addresses and ports. Similarly, Mirai attacks involve automated vulnerability scans, ACK flooding, SYN flooding, UDP flooding, and optimized UDP flooding designed for higher packet rates, as shown in Table 7.

Table 44: Distribution and Characteristics of Attack Classes in the N-BaIoT Dataset

Target Class	Attack Type	Description	Count
Benign	Benign	Unharmful network data	49548
gafgyt_combo	BASHLITE	Combines spam data and connection opening	59718
gafgyt_junk		Sending spam data to device	29068
gafgyt_scan		Network scan for vulnerable device	29849
gafgyt_udp		Flood targeted devices with the UDP packets	105874
mirai_ack	MIRAI	Flood targeted devices with the ACK packets	102195
mirai_scan		Automatic scan for vulnerable devices	107685
mirai_syn		Flood targeted devices with the SYN packets	122573
mirai_udp		Flood targeted devices with the UDP packets	2376655
mirai_udp_plain		Optimized UDP flooding for higher packets per second	81982

3.2.2. Edge-IIoT dataset

In this study, we used a second dataset named Edge-IIoT-2022 dataset [57], an extensive simulation encompassing 14 distinct cyberattacks categorized into five primary types: Denial of Service (DoS)/Distributed Denial of Service (DDoS), Information Gathering, Man in the Middle (MITM), Injection, and Malware attacks, as detailed in Table 8. We framed our intrusion detection systems (IDSs) within a multi-class classification context, differentiating among 15 classes: 14 representing specific attack types and one for normal traffic. Each data point in the Edge-IIoT-2022 dataset is characterized by a 61-feature vector, including 43 numeric features and string and nominal attributes. The dataset includes two pivotal label features, Attack_label and Attack_type, which indicate whether a data point is an attack and specify the type of attack, respectively. These labels are essential for classifying and detecting intrusions in AI-based models.

Table 45: Distribution and Characteristics of Attack Classes in the Edge-IIoT Dataset

Target Class	Attack Type	Description	Count
TCP SYN Flood	DDOS	Initiates numerous TCP handshake requests to deplete server resources, leading to unresponsiveness.	50062
UDP Flood		Sends a large volume of UDP packets to the server, overwhelming its capacity to process legitimate requests.	121567
HTTP Flood		Overloads the server with a high volume of HTTP queries, causing it to slow down or crash	48544
ICMP Flood		Floods the server with a high volume of ICMP (ping) requests, consuming its bandwidth and resources.	67939
Port scanning		Scans connected IoT devices to identify open ports that may be exploited for further attacks.	19977
OS Fingerprinting	Information gathering	Identifies the target operating system by analyzing responses to known probes and vulnerabilities.	853
Vulnerability Scanning		Detects potential security weaknesses in applications and networks to exploit in future attacks.	50026

DNS and APR Spoofing	MITM	Alters DNS and ARP tables to redirect traffic or intercept communication between devices.	358
Cross-site Scripting (XSS)	Injection	Injects malicious scripts into web applications, which execute in the user's browser.	15066
SQL Injection		Exploits vulnerabilities in database applications by inserting malicious SQL statements.	50829
Uploading		Attacks web applications that allow file uploads, often leading to the execution of malicious code.	36807
Backdoor	Malware	Establishes unauthorized remote access to an IoT device, allowing attackers to control it remotely.	24226
Password Cracking		Employs brute-force techniques to guess passwords or cryptographic keys, gaining unauthorized access.	49933
Ransomware		Encrypts files or locks IoT devices, demanding a ransom for their release.	9689

Our research aims to elevate the precision and efficiency of intrusion detection systems by focusing on these attack types and leveraging the extensive data provided by the N_BaIoT and Edge-IIoT-2022 datasets. This highlights the critical importance of selecting relevant features and deploying advanced AI-based models to optimize IDS performance, particularly in detecting and mitigating sophisticated IoT network threats.

3.3. Data Preparation

Data preparation is crucial in machine learning and deep learning. It involves cleaning and organizing data to enhance learning and model accuracy. Our research employed a two-step approach for data preparation, encompassing Data Pre-processing and Data Normalization techniques, as detailed in Algorithm 1.

3.3.1. Data Pre-Processing

In the data pre-processing stage, we transformed categorical features with nominal values into numerical values using label encoding, ensuring compatibility with the neural network's input requirements. We also removed irrelevant features such as date, time, and timestamp columns, which did not significantly contribute to output predictions.

3.3.2. Oversampling Minority Classes

In our experiments, we applied the Synthetic Minority Oversampling Technique (SMOTE) to generate synthetic samples for minority attack classes. This ensured a balanced dataset, particularly for rare attack types such as reconnaissance and backdoor attacks in the N-BaIoT and Edge-IIoT 2022 datasets. By addressing class imbalance during data preprocessing, we provided a more equitable distribution of instances, enabling our AI-based models to better learn patterns from underrepresented classes. To complement oversampling, we selectively reduced the size of majority classes where necessary, ensuring the overall dataset size remained manageable while maintaining a balanced representation of attack and benign classes.

3.3.3. Data Normalization

We applied data normalization using the min-max scaling technique to address feature imbalance, where some attributes had higher values than others and skewed model performance. This method maps the data to a range between 0.0 and 1.0 while preserving the data's inherent distribution [58]. The min-max scaling formula is mathematically expressed as follows:

$$y = \frac{X - X_{min}}{X_{max} - X_{min}}$$

Where $x\ and\ y$ represent the original and normalized values, respectively, and X_{min} and X_{max} are the feature's minimum and maximum values. This normalization step ensured balanced feature representation, enhancing our AI-based models performance and accuracy.

Algorithm 1: Min–Max Scaling

Input:
- $X(x1, ..., xi)$, where $1 < i < I$, denotes a dataset containing I attributes.

Output:
- $Xtrans(xtrans1, ..., xtransn)$, denotes the transformed dataset with scaled attributes.

1. for k from 1 to I, do
2. if (xi is a nominal attribute), then
3. Initialize and instantiate a LabelEncoder object to encode nominal attributes.
4. Fit the LabelEncoder to the xi values and transform them into numerical labels.
5. Store the transformed attributes in a new Pandas series.
6. Apply Min-Max Scaling to the transformed attributes:
7. $xtransi = (xi - \min(xi)) / (\max(xi) - \min(xi))$
8. else
9. Apply Min-Max Scaling directly to the numerical attribute:
10. $xtransi = (xi - \min(xi)) / (\max(xi) - \min(xi))$
11. end if
12. end for

3.4. Feature Selection

The objective of feature selection is to identify a representative subset of attributes from the original dataset, ensuring that the selected features are highly relevant to the prediction task. Modern intrusion detection datasets often contain numerous redundant and irrelevant attributes, diminishing the effectiveness of ML and DL algorithms and leading to uninterpretable results. Thus, the initial step in this study involves reducing dimensionality and selecting a pertinent feature subset from the dataset. We employ wrapper-based feature selection methods to optimize the selection process's efficiency and enhance classification accuracy. This approach centers on evaluating the relevance and redundancy of the selected features and navigating the search space to find the optimal solution.

3.4.1. Forward Selection wrapper-based method

Forward selection is a wrapper method for feature selection in machine learning. This method adds features to the model iteratively based on their impact on model performance. It starts with an empty set of features and gradually adds the most informative features one at a time. The key idea behind forward selection is to gradually build up a set of features that collectively optimize the model's performance. Algorithm 2 outlines the feature selection process utilizing the Forward Selection wrapper-based method for AI-based Models. This iterative approach systematically enhances model performance by sequentially incorporating features based on their impact on the chosen evaluation metric, optimizing predictive accuracy and interpretability.

Algorithm 2: Forward Selection wrapper-based method

Input:
- X: Feature matrix
- y: Target vector
- Model: Machine learning model
- Performance Metric: Metric used to evaluate model performance
- NumFeatures: Desired number of features to select [40 top features]

Output:
- selected_features: List of selected features

```
1. selected_features = empty list
2. best_performance = 0
3. remaining_features = list of all features in X
4. while remaining_features is not empty:
    a. max_performance_increase = 0
    b. for each feature in remaining_features:
        i. Add the current feature to selected_features.
        ii. Train the model using selected_features and evaluate its performance.
        iii. Calculate the performance increase by comparing it to best_performance.
        iv. If the performance increase is greater than max_performance_increase:
            - Update max_performance_increase
            - Update best_feature as the current feature
    c. If max_performance_increase > 0:
        - Add best_feature to selected_features
        - Remove best_feature from remaining_features
        - Update best_performance
    d. Else:
        - Break the loop
5. Return selected_features
```

3.4.2. Backward selection wrapper-based method

Backward elimination is another wrapper method for feature selection where features are removed from the model iteratively based on their impact on model performance. It starts with all features included and gradually eliminates the least informative features. Backward elimination aims to simplify the model by removing features that contribute the least to its predictive power. Algorithm 3 delineates the systematic feature selection process employing the backward-eliminating wrapper-based method for AI-based Models. This iterative technique iterates over the feature set, systematically removing attributes to enhance model performance based on the specified evaluation metric. This algorithm optimizes the efficiency and interpretability of the AI-based Models by strategically eliminating features that contribute minimally to predictive accuracy.

Algorithm 3: Backward Selection wrapper-based method

Input:
- X: Feature matrix
- y: Target vector
- Model: Machine learning model
- Performance Metric: Metric used to evaluate model performance
- NumFeatures: Desired number of features to select [40 top features]

Output:
- selected_features: List of selected features

1. selected_features = list of all features in X
2. best_performance = Evaluate model performance using selected_features
3. while selected_features is not empty:
    a. min_performance_drop = infinity
    b. for each feature in selected_features:
        i. Remove the current feature from selected_features
        ii. Train the model using selected_features and evaluate its performance.
        iii. Calculate the performance drop by comparing it to best_performance.
        iv. If the performance drop is less than min_performance_drop:
            - Update min_performance_drop
            - Update worst_feature as the current feature
    c. If min_performance_drop < infinity:
        - Remove worst_feature from selected_features
        - Update best_performance
    d. Else:
        - Break the loop
4. Return selected_features

3.4.3. Recursive Feature Elimination (RFE) method

Recursive Feature Elimination is a wrapper method where features are recursively removed from the model based on their importance ranking. It starts with all features included and gradually eliminates the least essential features according to a predetermined ranking criterion. RFE iteratively prunes the feature set by eliminating features with the lowest importance ranking, aiming to retain the most informative subset of features. Algorithm 4 outlines the Recursive Feature Elimination (RFE) method, a recursive wrapper-based approach designed for AI-based Models to iteratively refine feature sets by eliminating the least informative attributes. By iteratively assessing feature importance and selectively pruning the feature space, RFE streamlines the model's complexity while preserving predictive accuracy. This algorithm is instrumental in enhancing the efficiency and interpretability of AI-based Models by identifying and retaining the most relevant subset of features for optimal performance. The use of wrapper-based feature selection methods, particularly Recursive Feature Elimination (RFE), helped optimize feature subsets that were most relevant to distinguishing between attack and benign instances. By iteratively eliminating less significant features, RFE reduced the impact of irrelevant or redundant features that could exacerbate class imbalance issues.

Algorithm 4: Recursive Feature Elimination (RFE) method

Input:
- X: Feature matrix
- y: Target vector
- Model: Machine learning model

- Performance Metric: Metric used to evaluate model performance
- NumFeatures: Desired number of features to select [40 top features]

Output:
- selected_features: List of selected features

1. selected_features = list of all features in X
2. while number of selected_features > NumFeatures:
    a. Train the model using selected_features
    b. Calculate feature importance scores
    c. Identify the least important feature
    d. Remove the least important feature from selected_features
3. Return selected_features

3.5. Artificial Intelligence (AI) Based Model

In this sub-section, we introduced Artificial Intelligence (AI) based models for Intrusion Detection systems that have emerged as powerful techniques in exploring advanced security measures for Industrial IoT environments, offering innovative approaches to intrusion detection. Table 9 provides a detailed analysis of various AI-based models used for IDS, highlighting their key concepts, advantages, and disadvantages.

3.5.1. K-Nearest Neighbors (KNN)

The k-nearest neighbor (k-NN) algorithm is a simple yet powerful method for classifying objects based on their similarity to other objects in a dataset. It operates by examining the most similar examples in a dataset to determine the classification of a new object [59]. In intrusion detection, k-NN can determine whether new network activity is regular or malicious. The algorithm finds the closest examples in the dataset to the new activity based on specific features such as the number of connections or data transferred. It then predicts the new activity category by looking at the majority category of these closest examples. For instance, if most similar activities are labeled malicious, the new activity will also be classified as malicious; if most are normal, it will be classified as normal. The similarity between activities is measured using the Euclidean distance [60], which helps to identify the nearest neighbors. This straightforward approach makes k-NN an effective model for intrusion detection by leveraging the patterns and labels from known activities to classify new ones, as represented in Figure 4.

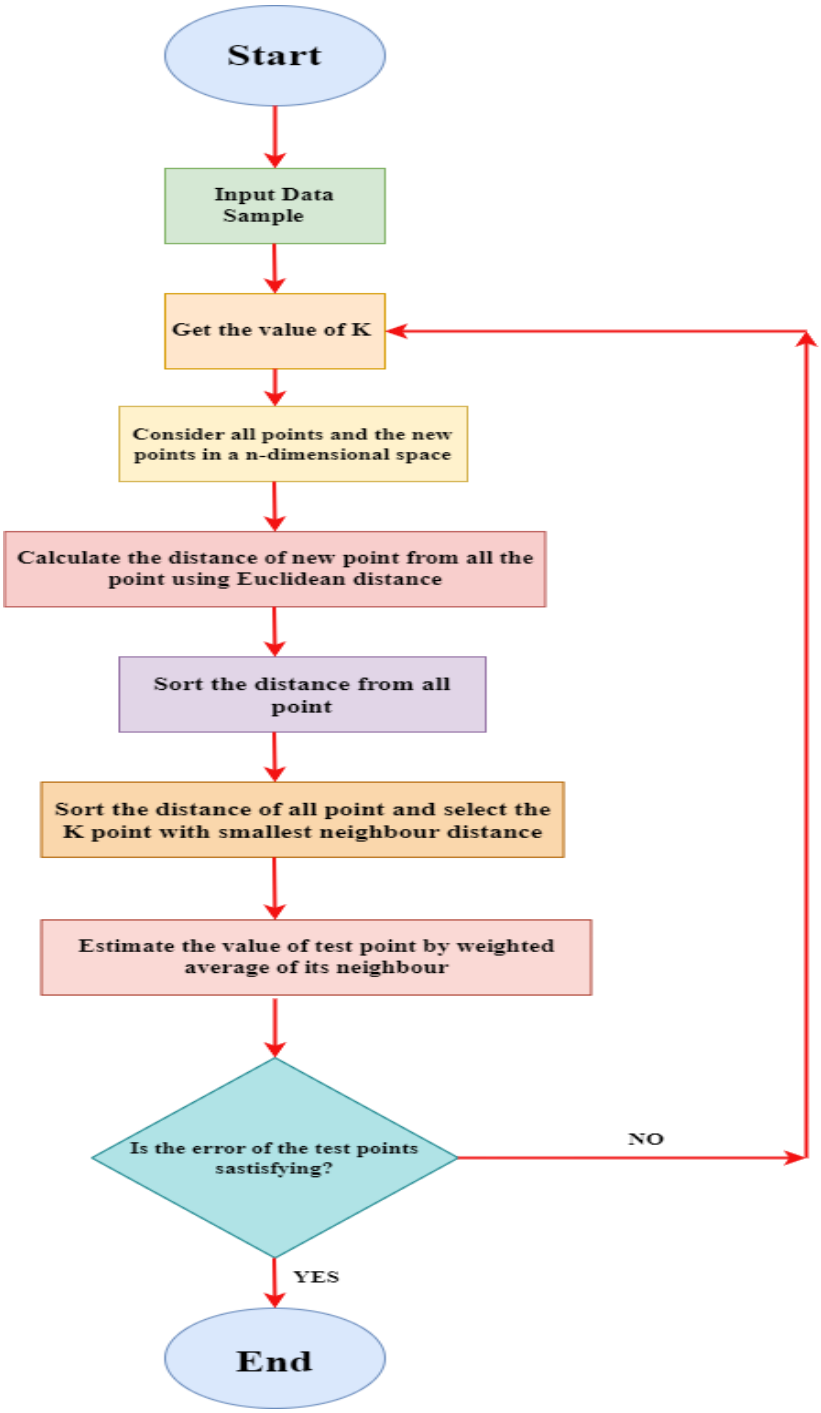


Figure 67: Flow of KNN

3.5.2. Support Vector Machine (SVM)

A support vector machine (SVM) is a supervised machine learning technique based on statistical learning theory. SVM classifies data by determining a set of support vectors, specific members of the labeled training data. The primary objective of SVM is to find an optimal hyperplane that can classify new data points accurately. A linear SVM acts as a binary classifier, separating multi-dimensional data by creating hyperplanes using each class's nearest training data

points. It maximizes the margin between these classes [61]. Consequently, SVM relies on a subset of the training data, known as support vectors, to perform classification. The Workflow of the SVM model for intrusion detection is shown in Figure 5.

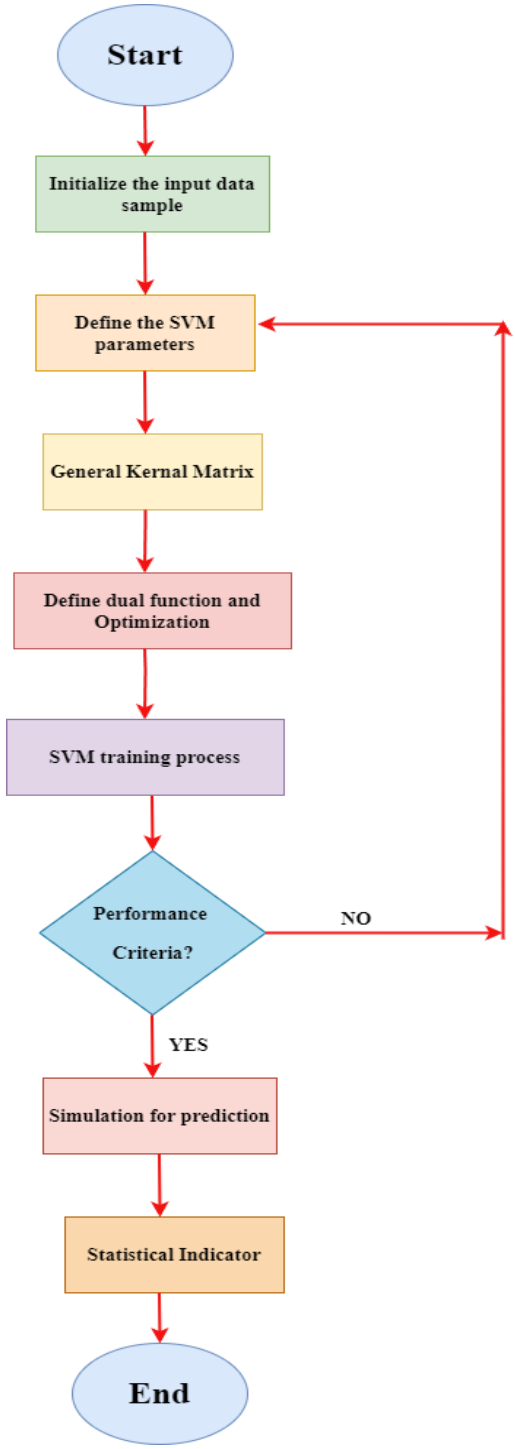


Figure 68: Flow of SVM

3.5.3. Decision Tree (DT)

The decision tree (DT) classification method uses an averaging approach by combining multiple models within an ensemble. This technique, known as "bagging" (Bootstrap Aggregating), reduces overfitting by aggregating and bootstrapping decision trees [62]. To evaluate the splitting of nodes, DT uses impurity measures such as the Gini Impurity (GI) and Information Gain (IG). Impurity measures assess the similarity of labels at a node and guide the division of DT nodes [63]. The Gini Impurity or Gini Index (GI) seeks to minimize impurity by identifying differences in the probability distributions of the target attribute's values. Information Gain (IG), on the other hand, measures the reduction in entropy, aiming to split nodes to achieve the highest information gain.

An essential parameter in decision trees is the maximum depth (max-depth), which indicates the extent to which nodes can be expanded. A deeper tree, resulting from more splits, tends to capture more data features, thus influencing the model's detection accuracy. Careful tuning of max-depth is essential to balance complexity and performance. The Workflow of the Decision Tree model for intrusion detection is shown in Figure 6.

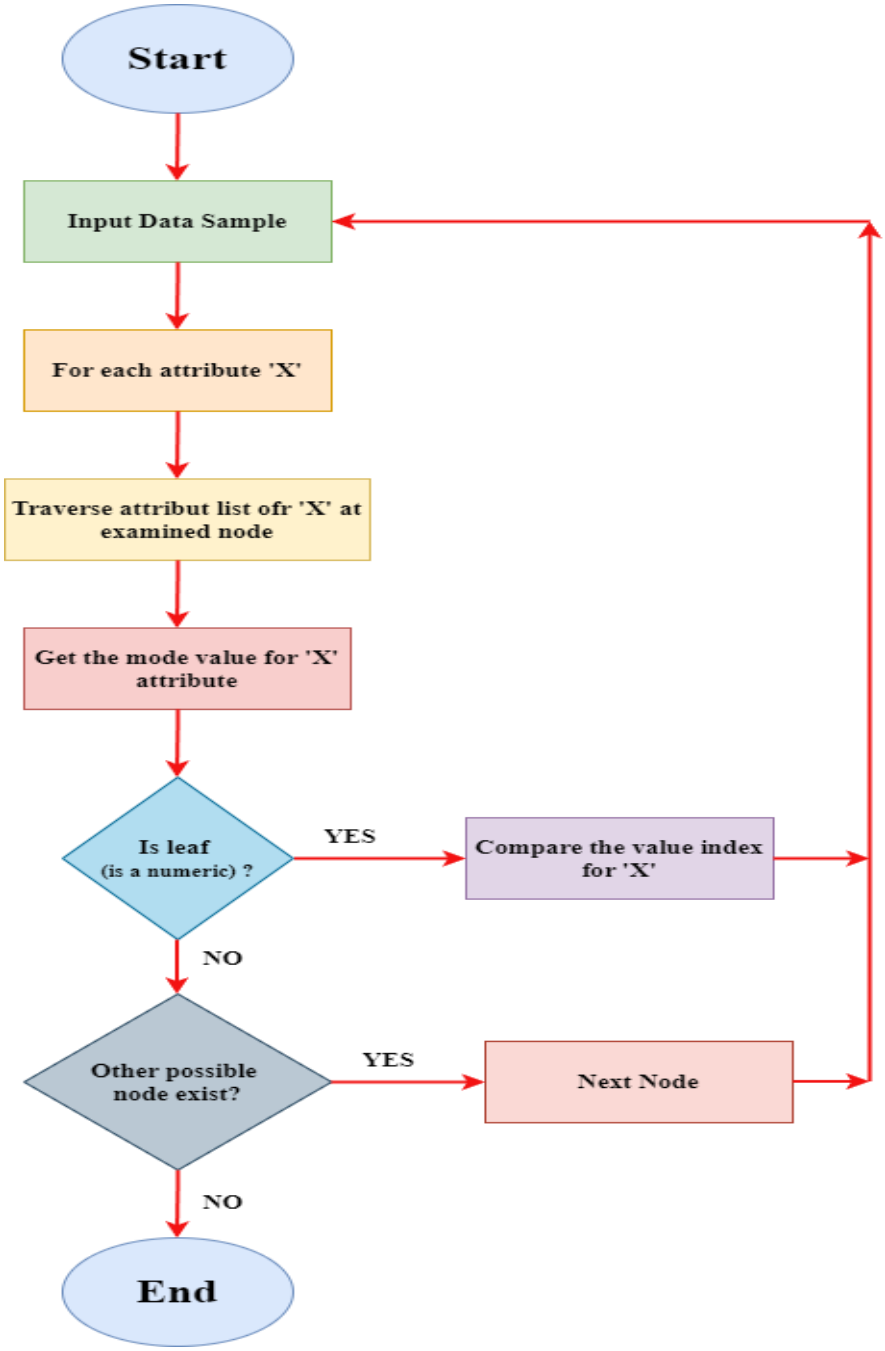


Figure 69: Flow of Decision Tree

3.5.4. Random Forest (RF)

The Random Forest (RF) is an ensemble prediction technique known for its effectiveness in various classification and regression problems [64]. RF makes final predictions by aggregating the outputs of multiple decision trees. This technique benefits from the random selection of data nodes when constructing each decision tree, enhancing the classifier's overall performance.

The performance of RF primarily depends on two key hyperparameters: the total number of leaves and the number of trees. The decision tree divides the feature space into the regions for a given number of leaves. This feature space is then used to predict the final output of a decision tree, with the final predicted outcome determined by the majority vote of all trees. Thus, the optimal selection of the number of leaves and trees during the training phase is crucial for achieving better performance during evaluation. The Workflow of the Random Forest model for intrusion detection is shown in Figure 7. Careful tuning of these parameters is necessary, as excessively increasing their values can lead to higher computational complexity without significant gains in performance.

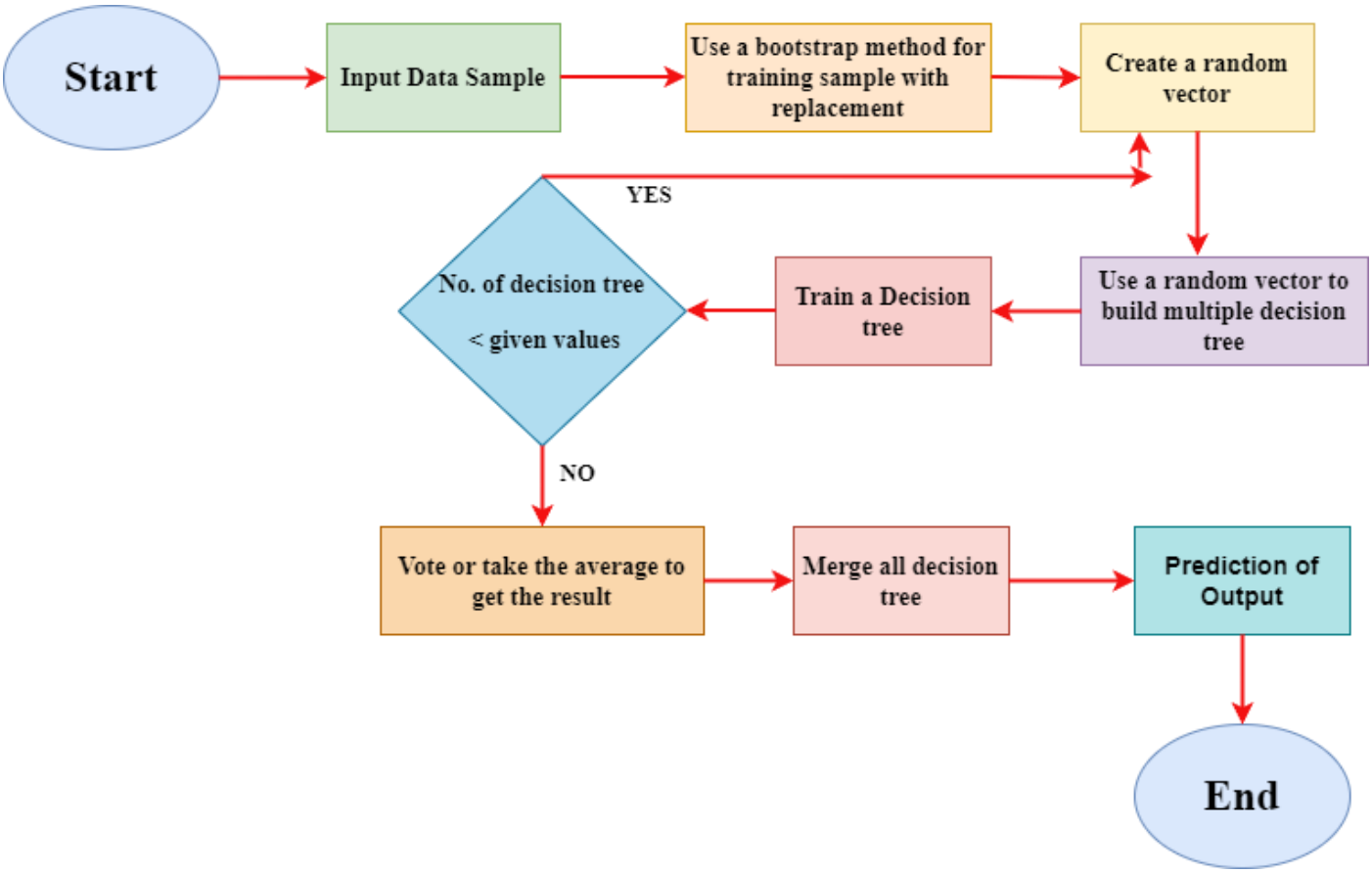


Figure 70: Flow of Random Forest

3.5.5. XGBoost

Extreme Gradient Boosting (XGBoost) is a powerful boosting technique that is part of the ensemble-based approach managed by the Distributed Machine Learning Community (DMLC) [65]. XGBoost is highly efficient, meticulously examining every bit of data value in the database. Before XGBoost, the random forest technique was commonly used, which involved providing the same data to multiple decision trees. Each decision tree was trained independently, and the overall accuracy was calculated by averaging the accuracies of all trees.

XGBoost, however, constructs decision trees sequentially, making it a sequential ensemble technique [66]. In this method, each data value in the database is assigned a weight, which determines its probability of being selected by a decision tree for further analysis. Initially, all data values have equal weights, which are then adjusted based on the analysis results, as shown in Figure 8. The outcomes of the initial data pass help create a new classification model that builds on previous results. This process continues iteratively until the final classifier is formed. Due to its depth and complexity, XGBoost produces results with low bias and high variance. In contrast, the random forest method typically

results in high bias and low variance because it better fits the training data. Careful tuning and understanding of XGBoost are essential for leveraging its capabilities effectively.

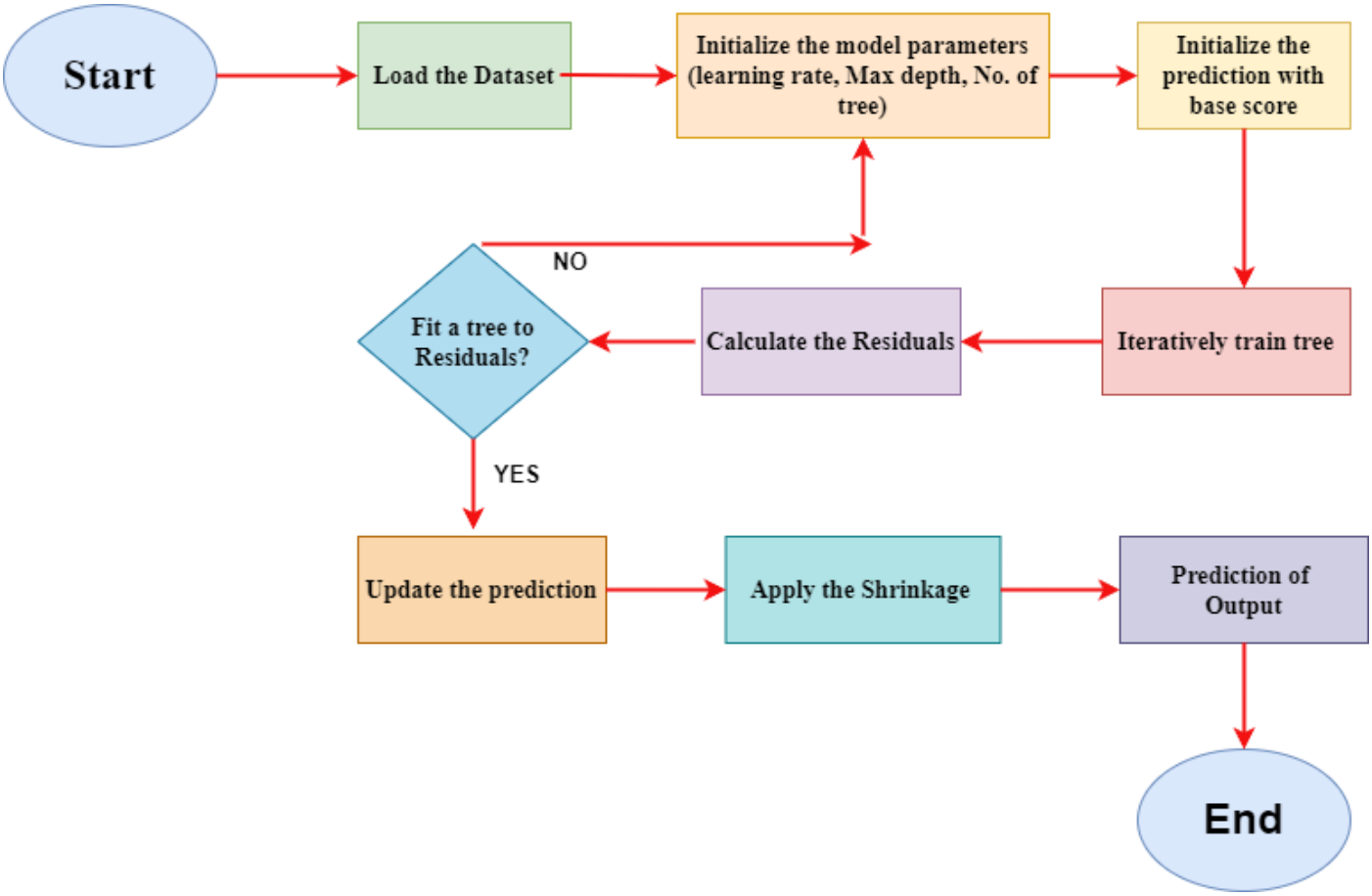


Figure 71: Flow of XGBoost

3.5.6. Convolutional Neural Network (CNN)

Convolutional Neural Networks (CNN) is a deep learning model primarily used for image data analysis. Though initially designed for image recognition tasks, CNNs can be adapted to process CSV data. By leveraging 1D convolutional layers, CNNs can effectively capture temporal patterns and dependencies in sequential CSV data [67]. This ability allows CNNs to extract relevant features from CSV datasets, enabling them to efficiently handle time series and other sequential data as represented in Figure 9.

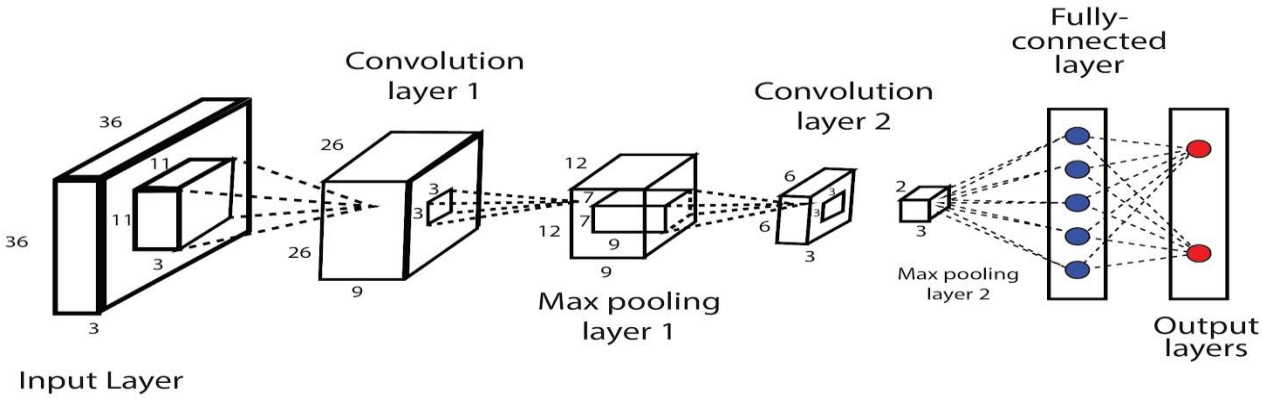


Figure 72: Convolutional Neural Network (CNN)

3.5.7. Recurrent Neural Network (RNN)

Compared to essential neural networks like Multilayer Perceptron (MLP), Recurrent Neural Networks (RNNs) offer more flexibility in processing information. Unlike MLPs, which process data in a single direction, RNNs can loop through different layers and temporarily store information for future use [68]. RNNs are classified as deep neural networks because they process information through multiple layers. As illustrated in Figure 10, unrolling a standard RNN reveals the depth of its structure [69]. Although RNNs are effective for various prediction tasks, they suffer from the vanishing gradient problem. This issue can hinder the training process, making capturing long-term dependencies in the data challenging.

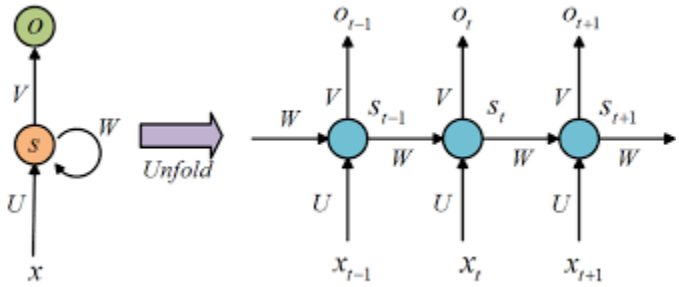


Figure 73: Recurrent Neural Network (RNN)

3.5.8. Long Short-Term Memory (LSTM)

Long Short-Term Memory (LSTM) networks, a variant of recurrent neural networks (RNNs), have become a popular choice for developing intrusion detection systems (IDS), particularly for network traffic analysis [70]. Network traffic data consists of sequential packets arriving over time. LSTM networks are specifically designed to handle such

sequential data, capturing long-term dependencies crucial for identifying malicious traffic patterns. Incorporating memory cells, LSTM networks can selectively remember or forget information, effectively addressing the vanishing gradient problem that traditional RNNs face [71]. This capability makes LSTM robust to real-world network traffic data irregularities, such as missing packets and noise. Their proficiency in managing sequential data, robustness to noisy and missing data, adaptability to different attacks, and high accuracy in detecting network intrusions while minimizing false positives make LSTM networks ideal for enhancing network security. The Systematic representation of the LSTM model is shown in Figure 11.

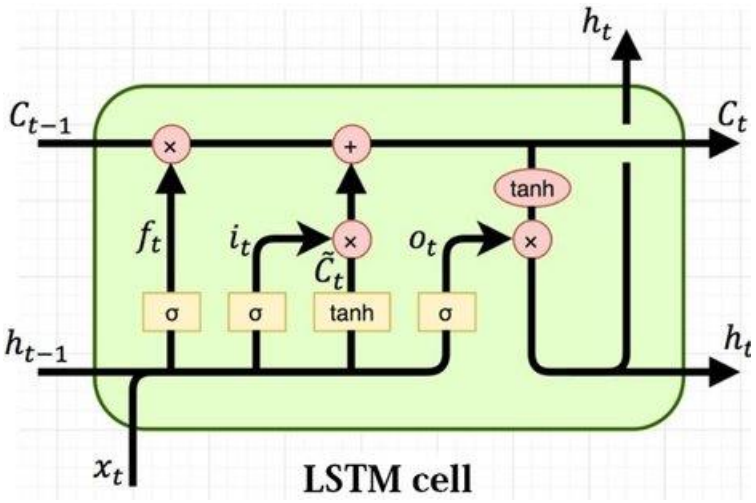


Figure 74: Long Short-Term Memory (LSTM)

3.5.9. Bidirectional- Long Short-Term Memory (Bi-LSTM)

A Recurrent Neural Network (RNN) is specifically designed for sequential data processing but often suffers from instability due to gradient vanishing and exploding issues. Hoch Reiter et al. [72] addressed these challenges by introducing Long Short-Term Memory (LSTM) networks, which incorporate gate mechanisms and memory units to effectively manage gradient behavior during training. Building on LSTM, Bidirectional LSTM (BiLSTM) enhances sequence learning by allowing information to flow in both forward and backward directions through two hidden states [73]. This bidirectional flow enables BiLSTM to capture context from past and future data points, thereby improving the network's ability to retain comprehensive information. In BiLSTM, the first layer's output sequence serves as the second layer's input, where the final output is a concatenation of the outputs from both forward and backward layers, resulting in a robust architecture that excels in learning and preserving sequential dependencies as shown in Figure 12.

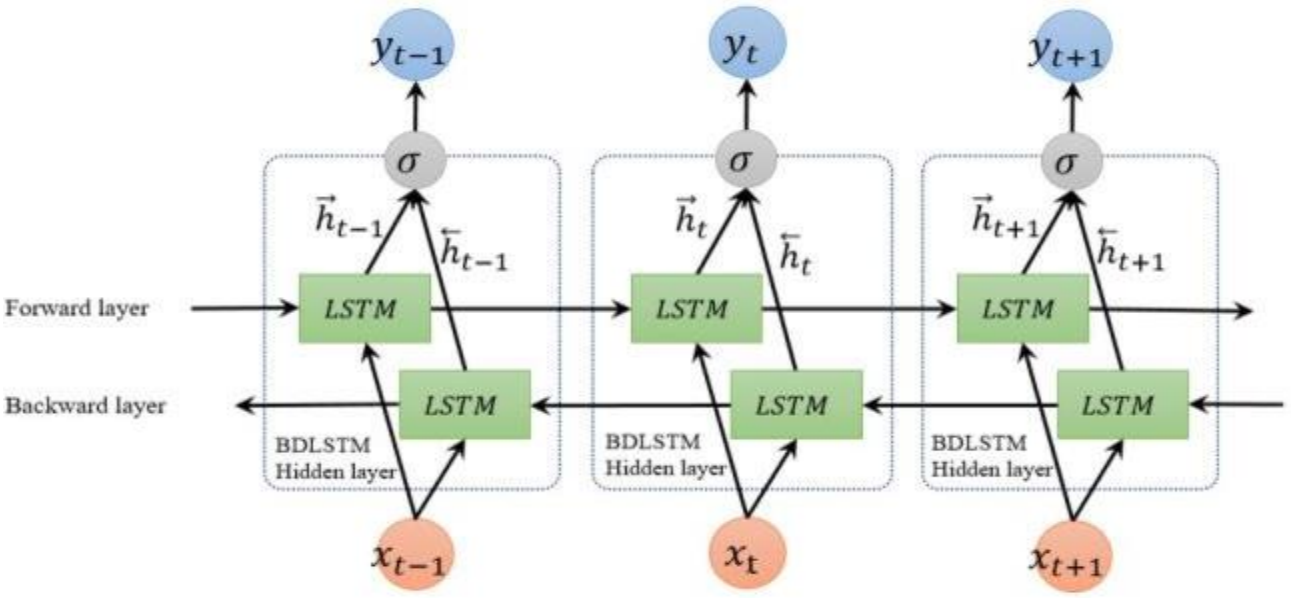


Figure 75: Bidirectional- Long Short-Term Memory (Bi-LSTM)

3.5.10. Gated Recurrent Unit (GRU)

The Gated Recurrent Unit (GRU) is a streamlined version of LSTM that reduces complexity by merging the forget and input gates into an update gate. GRU combines the hidden state and cell state, simplifying the architecture as shown in Figure 13. It retains important information while discarding irrelevant details, capturing temporal dependencies effectively [74]. The update gate determines how much past information to keep and new information to incorporate. By merging the hidden state and cell state, GRU eliminates the need for a separate cell state, reducing computational load. GRU offers a simplified yet efficient approach for sequential data modeling, reducing unnecessary complexities associated with LSTM networks.

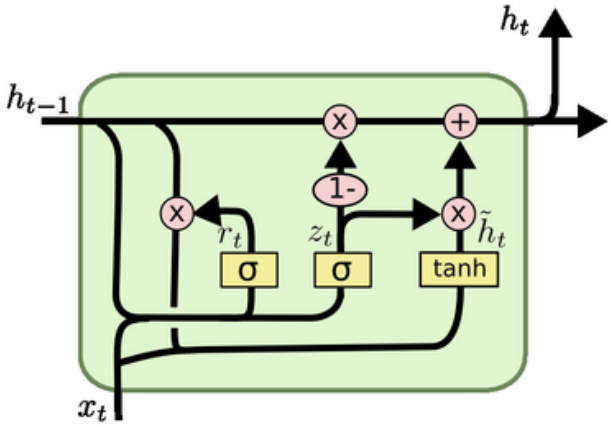


Figure 76: Gated Recurrent Unit (GRU)

Table 46: Analysis of Artificial Intelligence (AI)-Based Models for Intrusion Detection Systems (IDS)

Model	Definition	Key Concept	Advantage	Disadvantage
K-Nearest Neighbors (KNN)	KNN assigns a class label to an input data point based on the majority class among its K nearest neighbors.	Nearest neighbor search in feature space.	Simple, easy to implement, non-parametric.	Computationally expensive for large datasets.

Support Vector Machine (SVM)	SVM aims to find the hyperplane that best separates different classes in the feature space.	Maximizing margin between classes.	Effective in high-dimensional spaces, memory efficient.	Sensitive to the choice of kernel and regularization.
Decision Trees (DT)	Decision trees recursively split the feature space based on feature values to classify data points.	Splitting feature space to maximize information gain.	Easy to understand, interpretability.	Prone to overfitting can be unstable.
Random Forest (RF)	Random Forest aggregates predictions of multiple decision trees to make a final prediction.	Ensemble learning by combining multiple decision trees.	Reduction in overfitting, robust to noise.	Computationally expensive, black-box model.
XGBoost	XGBoost optimizes a differentiable loss function by adding weak learners sequentially to minimize the loss.	Gradient boosting with decision trees as base learners.	High predictive accuracy and handles missing values well.	Sensitive to hyperparameters, longer training time.
Convolutional Neural Network (CNN)	CNN applies convolutional filters to input data, followed by activation functions and pooling operations.	Hierarchical feature extraction through convolution and pooling.	Effective for image data, it automatically learns features.	Computationally expensive, requires large datasets.
Recurrent Neural Network (RNN)	RNN processes sequential data by maintaining an internal state or memory.	Capturing temporal dependencies in sequential data.	Suitable for sequential data, variable-length inputs.	Vanishing/exploding gradient problem, short-term memory.
Long Short-Term Memory (LSTM)	LSTM is designed to overcome the vanishing gradient problem and capture long-term dependencies in sequential data.	Gated mechanisms control the flow of information through the network.	Effective for long sequences, mitigates vanishing gradients.	Computationally expensive, longer training time.
Bidirectional LSTM (BI-LSTM)	BI-LSTM processes input sequences in both forward and backward directions to capture bidirectional dependencies.	Processing sequences bidirectionally for enhanced context.	Captures bidirectional information, improved performance.	Increased computational complexity and longer training time.
Gated Recurrent Unit (GRU)	GRU is a variation of the RNN architecture that simplifies the LSTM by combining the forget and input gates into a single update gate.	Simplified version of LSTM with fewer parameters.	Efficient memory usage is practical for sequential data.	It may not capture long-term dependencies or LSTM.

4. Experimental Setup and Result Analysis

This section delves into the experimental results, showcasing the efficacy of AI-based models in detecting anomalous behavior within IoT/IIoT environments. Utilizing the N_BaIoT and Edge-IIoT-2022 datasets, we rigorously trained and evaluated our models to ensure robust performance and accuracy.

4.1. Experimental Setup

We experimented with an ASUS-TUF Gaming F15 (FX506LHB) system featuring an Intel Core i5 10th Gen processor, 8GB RAM, 512GB ROM, and running on the Windows 11 operating system. The computer was outfitted with an NVIDIA GTX 1650 GDDR6 4GB graphics card, demonstrating satisfactory performance throughout the experiment. We thoroughly explored and analyzed the dataset by employing various data analysis frameworks such as Pandas, Numpy, Seaborn, Matplotlib, and Scikit-learn. The experiment accounted for the laptop's memory constraints, considering its limited 8GB RAM capacity.

4.2. Performance Evaluation Parameters

This section presents a thorough evaluation of our proposed model. To assess the approach's effectiveness, we conducted various analytical scenarios with diverse measurement parameters and detection times. The evaluation utilized several key metrics, detailed in Table 10. We used a comprehensive set of evaluation metrics, including precision, recall, F1-score, and G-mean, to ensure that model performance was assessed holistically. These metrics are particularly sensitive to imbalanced datasets and allowed us to evaluate the ability of our models to detect minority class instances accurately.

Table 47: Comprehensive Overview of Performance Evaluation Metrics for AI-based Model Evaluation

Parameters	Definition	Formula	Significance	Value Range
Accuracy	Proportion of correct predictions	$Accuracy = \frac{TP + TN}{TP + FP + TN + FN}$	Overall correctness of the model	0 to 1
Precision	Proportion of true positives out of all optimistic predictions	$Precision = \frac{TP}{TP + FP}$	Measure of model's ability to avoid false positives	0 to 1
Recall	Proportion of true positives out of all actual positives	$Recall = \frac{TP}{TP + FN}$	Measure of the model's ability to capture all positives	0 to 1
F1 score	The harmonic mean of precision and recall	$\text{F1Score} = 2 * \frac{\text{Precision*Recall}}{\text{Precision+Recall}}$	Balance between precision and recall	0 to 1
Specificity	Proportion of true negatives out of all actual negatives	$Specificity = \frac{TN}{TN + FP}$	Measure of model's ability to avoid false positives	0 to 1
G-mean	The geometric mean of TPR and TNR	sqrt(TPR * TNR)	Balanced measure considering both sensitivity and specificity	0 to 1

4.3. Result Analysis

In this section, we analyze the performance of AI-based models on the N-BaIoT and Edge-IIoT 2022 datasets. AI-based models exhibit considerable power for detection and solving complex problems, but their performance hinges on the quality and relevance of input features. In real-world scenarios, datasets often contain numerous less relevant or useful features. To address this, we employed wrapper-based feature selection techniques, including forward selection, backward selection, and Recursive Feature Elimination (RFE). We split each dataset into training and testing sets using an 80:20 ratio and identified the top 40 important features during the feature selection phase.

4.3.1. Comparative study on the N-BaIoT dataset

In this sub-section, we have analyzed the performance of the N-BaIoT dataset on three different wrapper-based techniques. Table 11 shows the performance of AI-based models on the N-BaIoT dataset using forward selection. Figure 14(a) shows the accuracy of the AI-based model. The GRU achieved 97.23% accuracy, the highest among all the models, followed by LSTM and BILSTM, having 96.56% and 95.26%, respectively.

Table 48: Comparative Analysis of AI-based Models Using Forward Feature Selection on the N-BaIoT Dataset

Model	Accuracy (%)	Precision (%)	Recall (%)	F1 score (%)	Specificity (%)	G mean (%)
K-Nearest Neighbors (KNN)	**77.56**	**76.23**	**76.12**	**76.17**	**76.23**	**76.17**
Support Vector Machine (SVM)	**81.32**	**82.36**	**81.47**	**81.91**	**82.36**	**81.91**
Decision Trees (DT)	**86.6**	**85.4**	**85.35**	**85.38**	**85.4**	**85.38**
Random Forest (RF)	**89.94**	**87.97**	**87.69**	**87.83**	**87.97**	**87.83**
XGBoost	**92.34**	**93.65**	**93.65**	**93.65**	**93.65**	**93.65**
Convolutional Neural Network (CNN)	**91.54**	**90.26**	**90.28**	**90.27**	**90.26**	**90.27**
Recurrent Neural Network (RNN)	**93.25**	**94.21**	**93.14**	**93.67**	**94.21**	**93.67**
Long Short-Term Memory (LSTM)	**96.56**	**95.92**	**95.87**	**95.89**	**95.92**	**95.89**
Bidirectional LSTM (BI-LSTM)	**95.26**	**95.14**	**95.21**	**95.17**	**95.14**	**95.17**
Gated Recurrent Unit (GRU)	**97.23**	**96.36**	**95.9**	**96.13**	**96.36**	**96.13**

The precision and recall score for GRU indicates the model's ability to correctly identify attacks while minimizing false positives, as shown in Figures 14(b) and 14(c). Figure 14(d) shows the F1-score of the AI-based models reflecting the balance between precision and recall, underscoring the model's effectiveness in detecting both attacks and correctly identifying benign instances. Figures 14(e) and 14(f) indicate the model's specificity and G-mean score, indicating capabilities to accurately classify normal instances, essential for minimizing the false alarm rate.

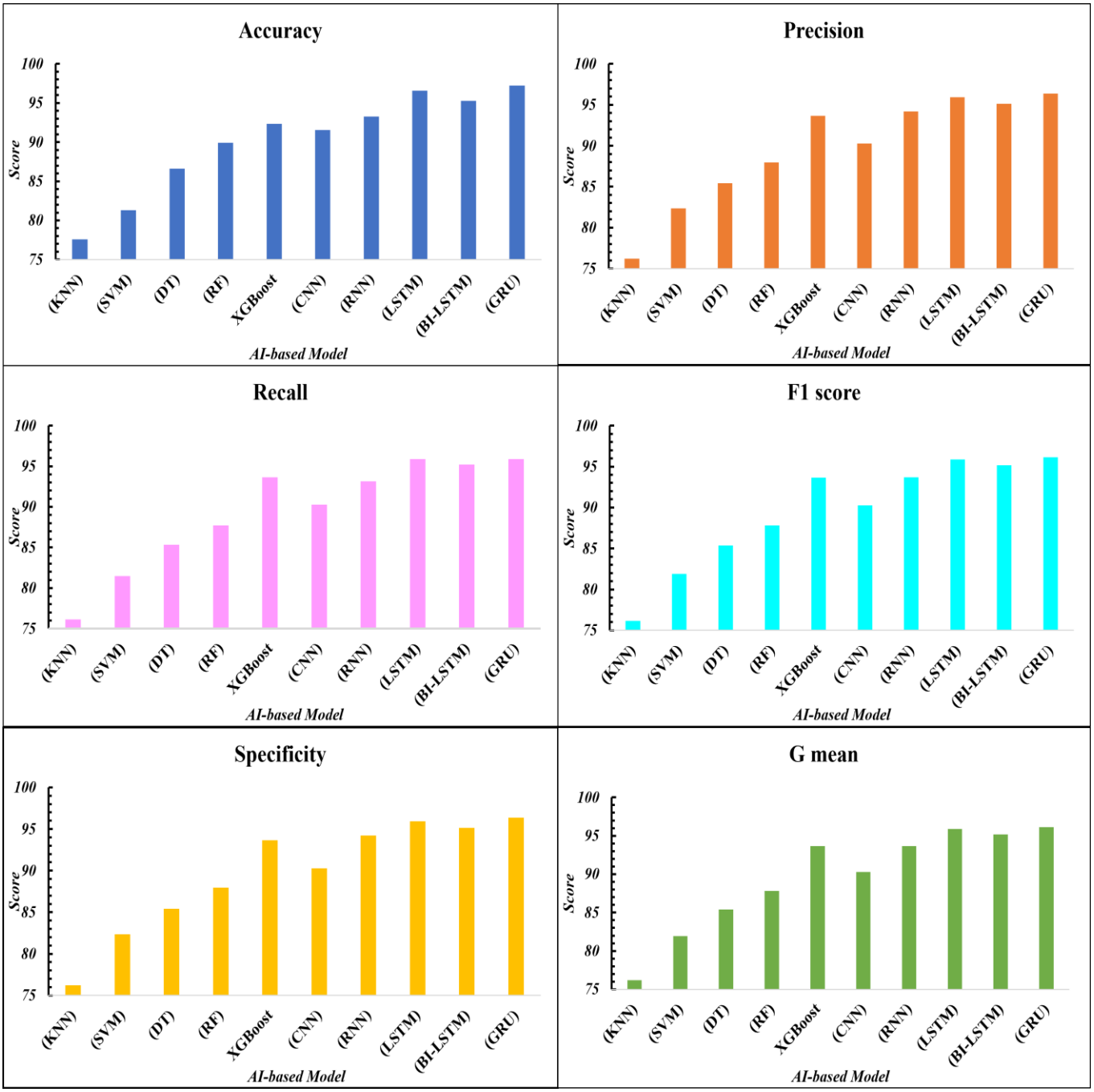


Figure 77: Performance of AI-based IDS models on Forward based Feature selection Method on N_BaIoT dataset

Table 12 shows the performance of AI-based models on the N-BaIoT dataset using the Backward selection-based wrapper method. The backward feature selection method showed varying impacts on the performance of different AI-based models. Models like GRU, LSTM, and BILSTM consistently maintained high accuracy and balanced performance across precision and recall, F1-score, specificity, and G-mean. These models benefit from eliminating the less relevant features, enhancing their ability to distinguish between normal and attack instances. On the other hand, models like KNN and SVM showed relatively lower accuracy and F1 scores, indicating that the elimination of features may have removed some relevant information needed for their classification. The graphical representation of accuracy, precision, recall, F1- score, specificity, and G-mean is shown in Figure 15 (a), (b), (c), (d), (e), and (f), respectively.

Table 49: Comparative Analysis of AI-based Models Using Backward Feature Selection on the N-BaIoT Dataset

Model	Accuracy (%)	Precision (%)	Recall (%)	F1 score (%)	Specificity (%)	G mean (%)
K-Nearest Neighbors (KNN)	**75.11**	**75.23**	**75.19**	**75.21**	**75.23**	**75.21**
Support Vector Machine (SVM)	**80.23**	**80.56**	**81.04**	**80.79**	**80.56**	**80.79**
Decision Trees (DT)	**84.2**	**83.4**	**84.14**	**83.77**	**83.4**	**83.77**
Random Forest (RF)	**87.63**	**88.65**	**87.97**	**88.31**	**88.65**	**88.31**
XGBoost	**89.87**	**88.73**	**88.8**	**88.77**	**88.73**	**88.77**
Convolutional Neural Network (CNN)	**89.21**	**88.75**	**87.75**	**88.25**	**88.75**	**88.25**
Recurrent Neural Network (RNN)	**91.12**	**90.21**	**90.23**	**90.22**	**90.21**	**90.22**
Long Short-Term Memory (LSTM)	**93.54**	**92.34**	**92.34**	**92.34**	**92.34**	**92.34**
Bidirectional LSTM (BI-LSTM)	**94.42**	**92.48**	**92.45**	**92.46**	**92.48**	**92.46**
Gated Recurrent Unit (GRU)	**95.53**	**93.65**	**92.78**	**93.21**	**93.65**	**93.21**

Table 13 shows the RFE method's performance on the N-BaIoT dataset. The results demonstrate that the RFE method has effectively improved the performance of all 10 AI-based models, as evidenced by the higher accuracy, precision, recall, F1-score, specificity, and G-mean compared to forward selection and backward selection wrapper methods. The RFE method effectively selects the most relevant features, enhancing the model's ability to detect and classify attacks. Figures 16 (a), (b), (c), (d), (e), and (f) show that the RFE method balances precision, recall, and overall accuracy.

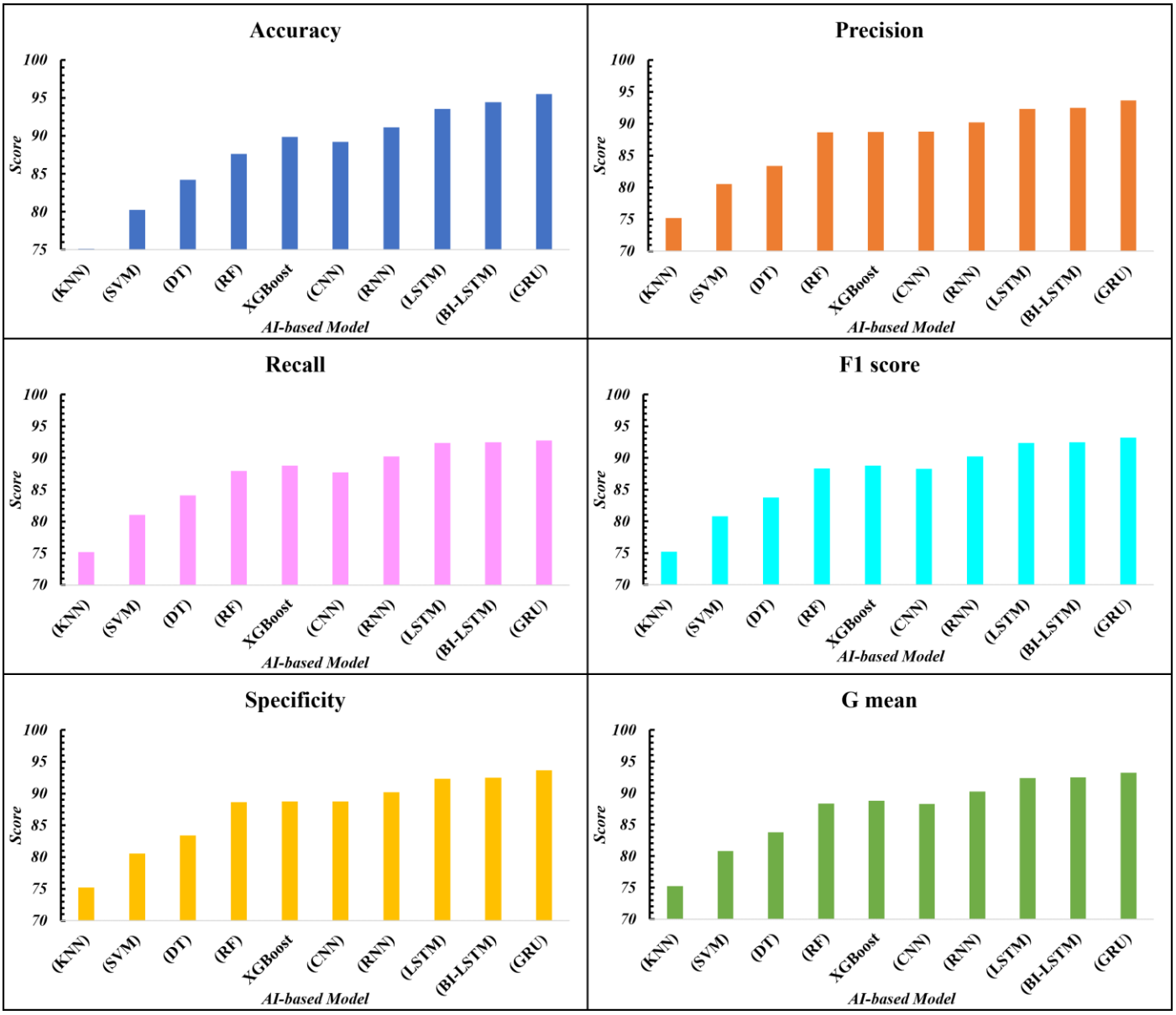


Figure 78: Performance of AI-based IDS models on Backward based Feature selection Method on N_BaIoT dataset

In a comprehensive comparison of feature selection methods applied to the N-BaIoT dataset, the Recursive Feature Elimination (RFE) method emerged as the most effective, consistently delivering superior performance across all models evaluated. The Gated Recurrent Unit (GRU) model, under RFE, achieved the highest metrics with an accuracy of 98.81%, precision of 97.93%, recall of 97.64%, and F1 score of 97.79%. This was followed by the Bidirectional Long Short-Term Memory (BI-LSTM) model, which also performed exceptionally well with an accuracy of 98.16%. In contrast, the forward selection-based wrapper method showed intermediate performance, with the GRU model again leading with an accuracy of 97.23%. The K-Nearest Neighbors (KNN) model was the lowest performer, with an accuracy of 77.56%.

Table 50: Comparative Analysis of AI-based Models Using Recursive Feature Elimination (RFE) on the N-BaIoT Dataset

Model	Accuracy (%)	Precision (%)	Recall (%)	F1 score (%)	Specificity (%)	G mean (%)
K-Nearest Neighbors (KNN)	**79.12**	**78.23**	**78.21**	**78.22**	**78.23**	**78.22**

Support Vector Machine (SVM)	**83.14**	**81.86**	**80.57**	**81.21**	**81.86**	**81.21**
Decision Trees (DT)	**87.36**	**85.62**	**85.61**	**85.61**	**85.62**	**85.61**
Random Forest (RF)	**91.34**	**90.27**	**90.04**	**90.15**	**90.27**	**90.15**
XGBoost	**94.27**	**92.81**	**92.37**	**92.59**	**92.81**	**92.59**
Convolutional Neural Network (CNN)	**93.19**	**92.52**	**92.49**	**92.51**	**92.52**	**92.51**
Recurrent Neural Network (RNN)	**96.72**	**94.78**	**94.7**	**94.74**	**94.78**	**94.74**
Long Short-Term Memory (LSTM)	**97.86**	**97.35**	**97.39**	**97.37**	**97.35**	**97.37**
Bidirectional LSTM (BI-LSTM)	**98.16**	**97.79**	**98.06**	**97.92**	**97.79**	**97.92**
Gated Recurrent Unit (GRU)	**98.81**	**97.93**	**97.64**	**97.79**	**97.93**	**97.79**

The backward selection-based wrapper method generally exhibited the lowest performance among the three techniques, with the GRU model achieving the highest accuracy of 95.53% and the KNN model showing the lowest accuracy of 75.11%. Comparing these methods in detail, RFE demonstrated significant improvements in accuracy over forward selection by 3.01% for KNN, 1.82% for Support Vector Machine (SVM), 0.76% for Decision Trees (DT), 1.4% for Random Forest (RF), 1.93% for XGBoost, 1.65% for Convolutional Neural Network (CNN), 3.47% for Recurrent Neural Network (RNN), 1.3% for Long Short-Term Memory (LSTM), 2.9% for BI-LSTM, and 1.58% for GRU. When compared to the backward selection, RFE's improvements in accuracy were even more pronounced, with increases of 5.34% for KNN, 3.64% for SVM, 3.16% for DT, 3.71% for RF, 4.4% for XGBoost, 4.46% for CNN, 5.6% for RNN, 4.32% for LSTM, 3.74% for BI-LSTM, and 3.28% for GRU.

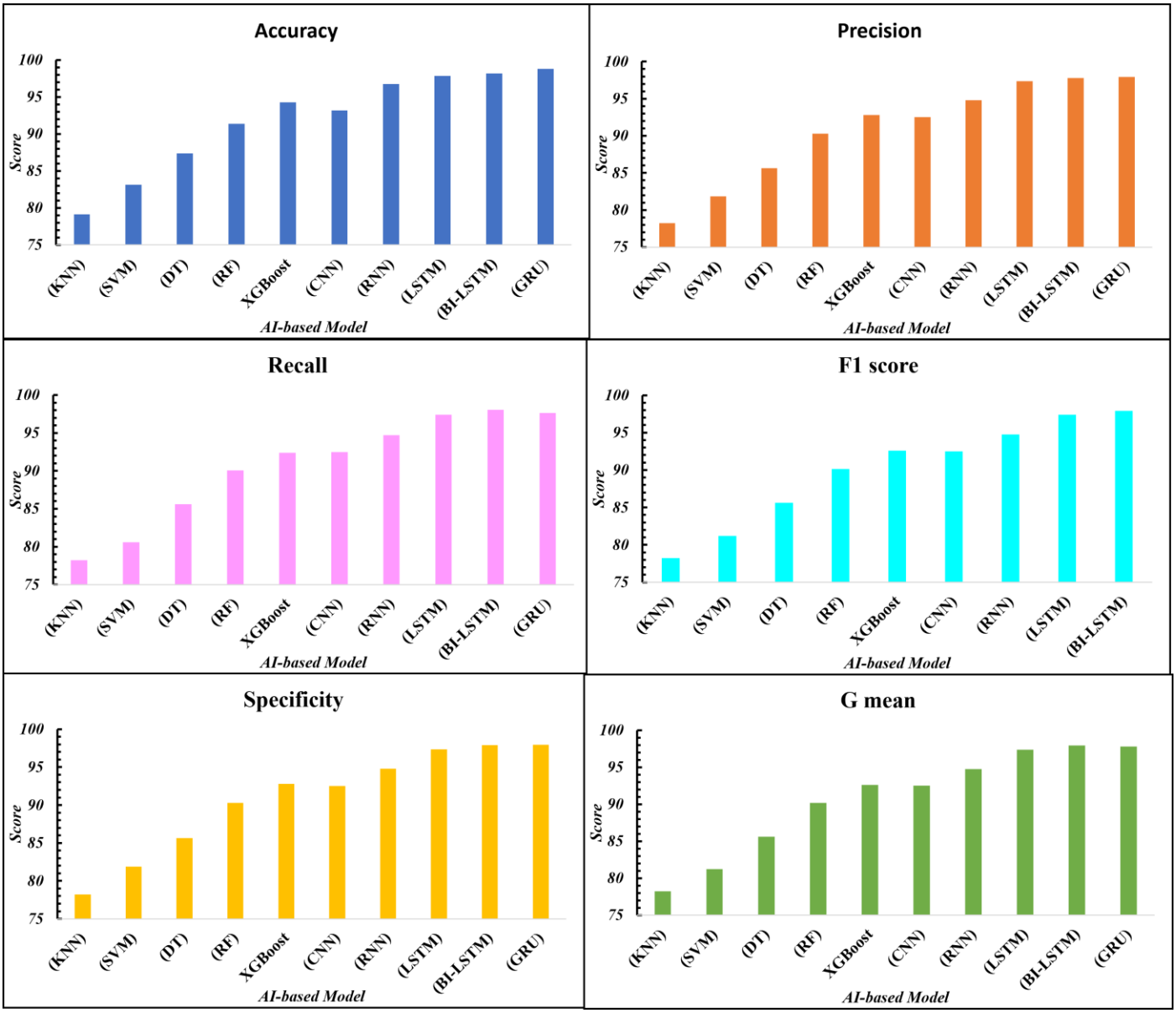


Figure 79: Performance of AI-based IDS models on Recursive Feature Elimination (RFE) selection Method on N_BaIoT dataset

These findings highlight that RFE not only optimizes model performance but does so consistently across all AI-based models, making it the most compelling feature selection technique for the N-BaIoT dataset. Forward selection provided intermediate improvements but needed to match the effectiveness of RFE, while backward selection consistently resulted in the lowest performance metrics. Overall, the results suggest that RFE is the superior method for enhancing the accuracy and reliability of intrusion detection systems within IoT environments.

4.3.2. Comparative study on the Edge-IIoT dataset

In this sub-section, we have analyzed the performance of the Edge-IIoT dataset on three different wrapper-based techniques. Table 14 shows the performance of AI-based models on the Edge-IIoT dataset using forward selection. The forward selection-based wrapper method significantly enhances the performance of AI-based models on the Edge-IIoT dataset by incrementally adding the most relevant features, resulting in improved metrics across accuracy, precision, recall, F1 score, specificity, and G mean, as shown in Figure 17 (a), (b), (c), (d), (e) and (f). Notably, the

Gated Recurrent Unit (GRU) model achieved the highest performance with an accuracy of 97.13%, precision of 96.24%, recall of 95.90%, and F1 score of 96.06%, demonstrating the method's efficacy in optimizing complex sequential models.

Table 51: Comparative Analysis of AI-based Models Using Forward Feature Selection on the Edge-IIoT Dataset

Model	Accuracy (%)	Precision (%)	Recall (%)	F1 score (%)	Specificity (%)	G mean (%)
K-Nearest Neighbors (KNN)	**76.14**	**75.62**	**75.42**	**75.51**	**75.62**	**75.51**
Support Vector Machine (SVM)	**78.49**	**77.84**	**78.10**	**77.96**	**77.84**	**77.96**
Decision Trees (DT)	**84.56**	**82.34**	**81.92**	**82.12**	**82.34**	**82.12**
Random Forest (RF)	**87.82**	**87.26**	**87.78**	**87.51**	**87.26**	**87.51**
XGBoost	**90.73**	**90.02**	**89.31**	**89.66**	**90.02**	**89.66**
Convolutional Neural Network (CNN)	**91.21**	**90.46**	**89.76**	**90.10**	**90.46**	**90.10**
Recurrent Neural Network (RNN)	**92.79**	**92.14**	**93.45**	**92.79**	**92.14**	**92.79**
Long Short-Term Memory (LSTM)	**94.53**	**93.94**	**94.06**	**93.99**	**93.94**	**93.99**
Bidirectional LSTM (BI-LSTM)	**95.68**	**95.34**	**95.14**	**95.23**	**95.34**	**95.23**
Gated Recurrent Unit (GRU)	**97.13**	**96.24**	**95.90**	**96.0.6**	**96.24**	**96.0.6**

Other models, such as the Bidirectional Long Short-Term Memory (BI-LSTM) and Long Short-Term Memory (LSTM), also showed significant improvements, indicating that forward selection effectively enhances their capability to handle temporal dependencies. In contrast, models like K-Nearest Neighbors (KNN) and Support Vector Machine (SVM) exhibited more modest gains, suggesting that these models may require different feature selection strategies or additional preprocessing to achieve optimal results. Overall, forward selection is a valuable technique for identifying key features that enhance the performance of AI models in intrusion detection systems within IoT environments.

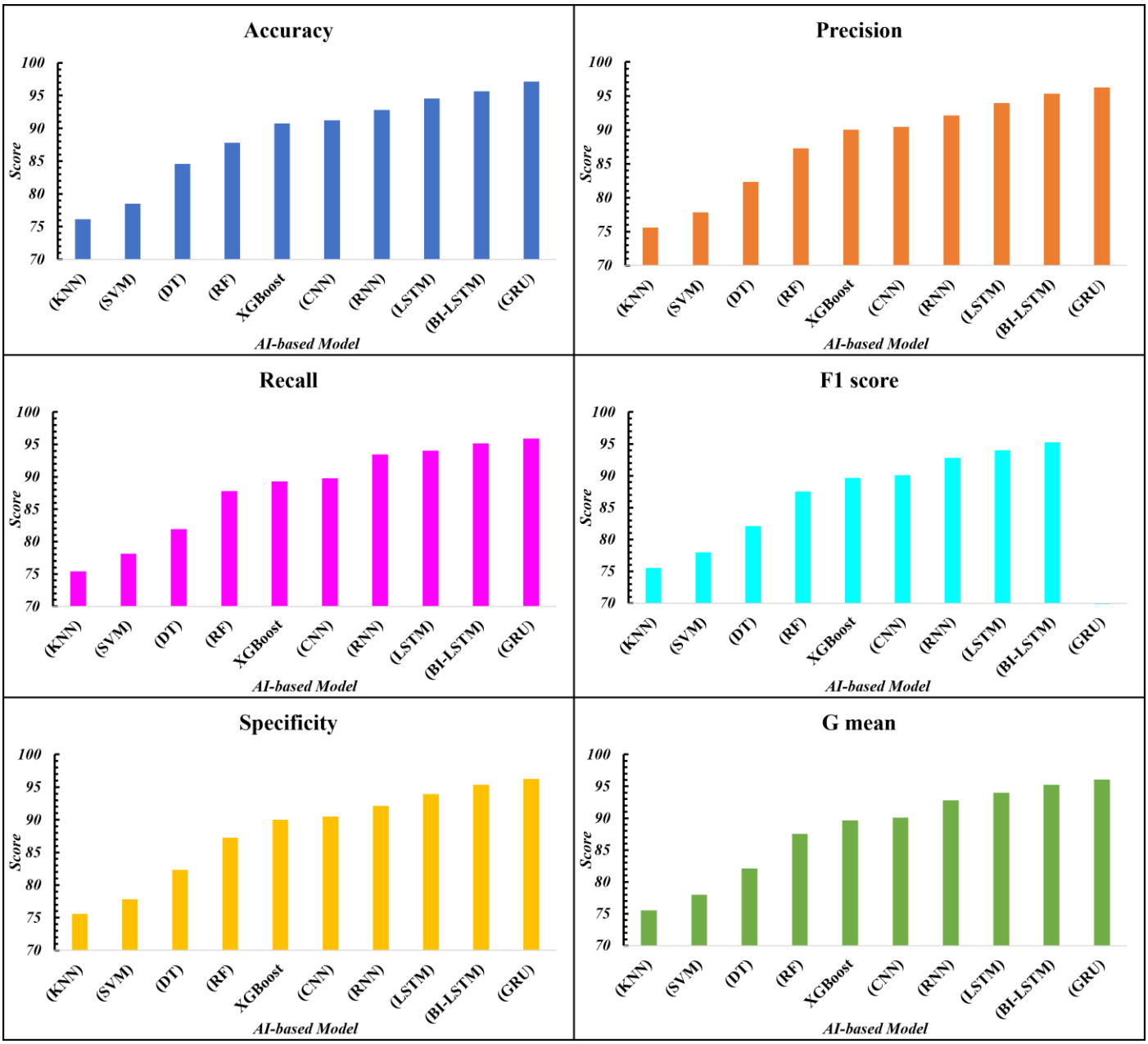


Figure 80: Performance of AI-based IDS models on Forward based Feature selection Method on Edge-IIoT 2022 dataset

Table 15 shows the performance of AI-based models on the Edge-IIoT dataset using backward selection. The backward selection-based wrapper method applied to the Edge-IIoT dataset refines model performance by systematically eliminating less relevant features, thus enhancing AI models' overall efficiency and accuracy, as shown in Figures 18 (a), (b), (c), (d), (e) and (f). The Gated Recurrent Unit (GRU) model achieved the highest metrics with an accuracy of 95.13%, precision of 94.14%, recall of 94.14%, and F1 score of 94.14%, demonstrating the method's capability to optimize complex models by focusing on the most pertinent features.

Table 52: Comparative Analysis of AI-based Models Using Backward Feature Selection on the Edge-IIoT Dataset

Model	Accuracy (%)	Precision (%)	Recall (%)	F1 score (%)	Specificity (%)	G mean (%)
K-Nearest Neighbors (KNN)	**74.83**	**75.12**	**74.12**	**74.61**	**75.12**	**74.61**

Support Vector Machine (SVM)	**76.36**	**75.47**	**75.47**	**75.47**	**75.47**	**75.47**
Decision Trees (DT)	**81.24**	**79.27**	**78.93**	**79.09**	**79.27**	**79.09**
Random Forest (RF)	**85.74**	**84.61**	**85.06**	**84.83**	**84.61**	**84.83**
XGBoost	**87.07**	**85.27**	**85.43**	**85.34**	**85.27**	**85.34**
Convolutional Neural Network (CNN)	**89.32**	**86.94**	**87.12**	**87.02**	**86.94**	**87.02**
Recurrent Neural Network (RNN)	**91.27**	**89.39**	**90.48**	**89.93**	**89.39**	**89.93**
Long Short-Term Memory (LSTM)	**93.47**	**91.24**	**91.24**	**91.24**	**91.24**	**91.24**
Bidirectional LSTM (BI-LSTM)	**94.09**	**94.10**	**94.06**	**94.07**	**94.10**	**94.07**
Gated Recurrent Unit (GRU)	**95.13**	**94.14**	**94.14**	**94.14**	**94.14**	**94.14**

Similarly, the Bidirectional Long Short-Term Memory (BI-LSTM) and Long Short-Term Memory (LSTM) models exhibited strong performance, indicating effective handling of temporal data dependencies. Conversely, simpler models like K-Nearest Neighbors (KNN) and Support Vector Machine (SVM) showed lower performance improvements, with KNN achieving the lowest accuracy at 74.83%, suggesting that these models might suffer from the removal of certain features critical for their classification tasks. Overall, the backward selection method effectively enhances model performance, particularly for advanced neural network models, by reducing dimensionality and improving feature relevance in intrusion detection systems within IoT environments.

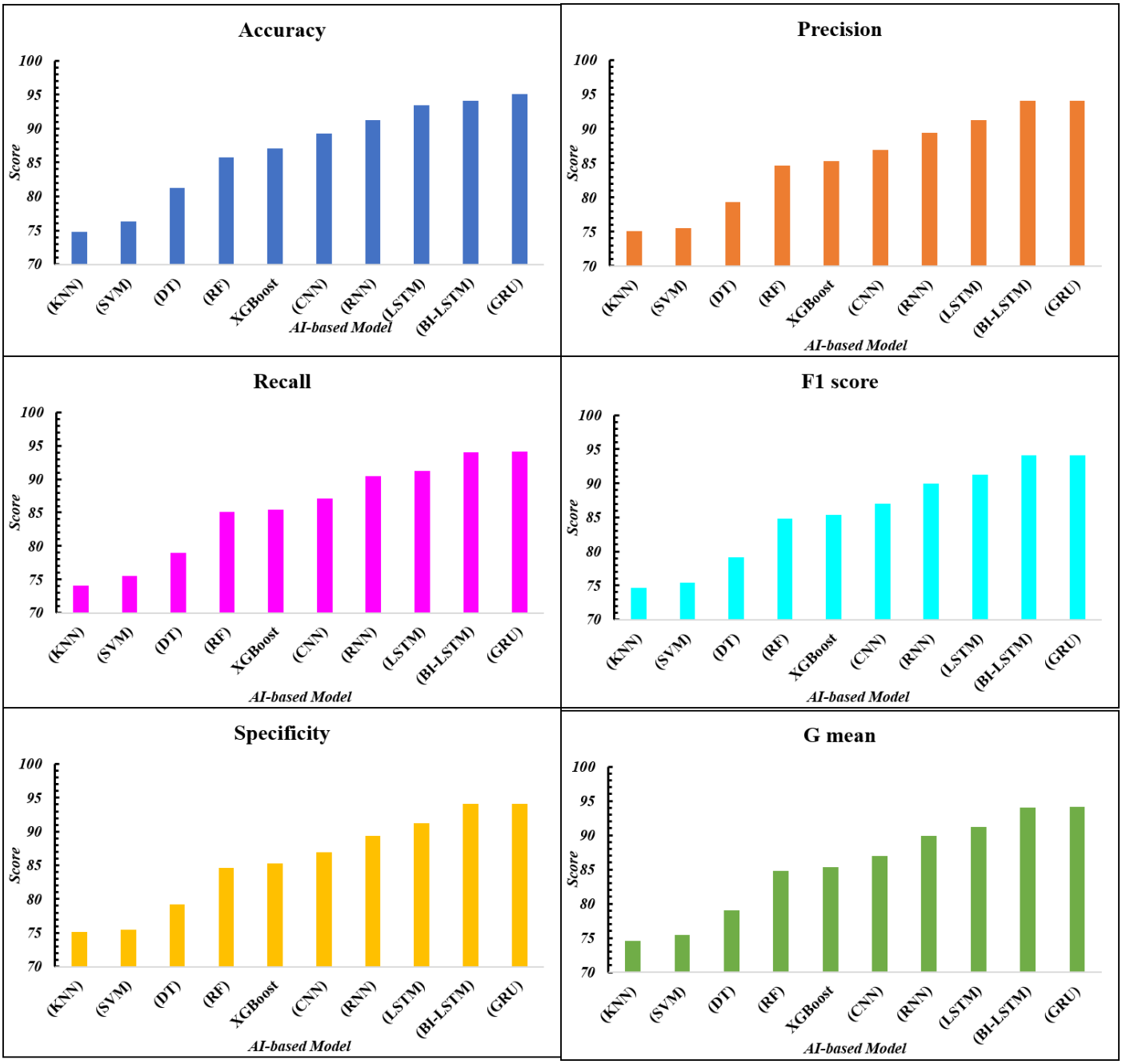


Figure 81: Performance of AI-based IDS models on Backward Feature selection Method on Edge-IIoT 2022 dataset

Table 16 shows the performance of AI-based models on the Edge-IIoT dataset using the Recursive Feature Elimination (RFE) method. The Recursive Feature Elimination (RFE) wrapper method markedly enhances the performance of AI models on the Edge-IIoT dataset by iteratively removing the least significant features, thereby refining the feature set for optimal classification. The Gated Recurrent Unit (GRU) model achieved the highest performance with an accuracy of 98.21%, precision of 96.36%, recall of 95.9%, and F1 score of 96.13%, underscoring the method's efficacy in improving model accuracy and robustness as shown in Figures 19 (a), (b), (c), (d), (e) and (f).

Table 53: Comparative Analysis of AI-based Models Using Recursive Feature Elimination (RFE) on the Edge-IIoT Dataset

Model	Accuracy (%)	Precision (%)	Recall (%)	F1 score (%)	Specificity (%)	G mean (%)
K-Nearest Neighbors (KNN)	**77.06**	**76.23**	**76.12**	**76.17**	**76.23**	**76.17**
Support Vector Machine (SVM)	**79.27**	**82.36**	**81.47**	**81.91**	**82.36**	**81.91**
Decision Trees (DT)	**86.03**	**85.4**	**85.35**	**85.38**	**85.4**	**85.38**
Random Forest (RF)	**88.94**	**87.97**	**87.69**	**87.83**	**87.97**	**87.83**
XGBoost	**91.42**	**93.65**	**93.65**	**93.65**	**93.65**	**93.65**
Convolutional Neural Network (CNN)	**92.03**	**90.26**	**90.28**	**90.27**	**90.26**	**90.27**
Recurrent Neural Network (RNN)	**93.65**	**94.21**	**93.14**	**93.67**	**94.21**	**93.67**
Long Short-Term Memory (LSTM)	**95.74**	**95.92**	**95.87**	**95.89**	**95.92**	**95.89**
Bidirectional LSTM (BI-LSTM)	**96.48**	**95.14**	**95.21**	**95.17**	**95.14**	**95.17**
Gated Recurrent Unit (GRU)	**98.21**	**96.36**	**95.9**	**96.13**	**96.36**	**96.13**

Similarly, the Bidirectional Long Short-Term Memory (BI-LSTM) and Long Short-Term Memory (LSTM) models demonstrated substantial improvements, achieving accuracies of 96.48% and 95.74%, respectively, indicating the method's effectiveness in enhancing the capabilities of sequential models. Other models, such as XGBoost and Convolutional Neural Network (CNN), also saw significant gains, with XGBoost achieving a precision of 93.65%. In contrast, simpler models like K-Nearest Neighbors (KNN) and Support Vector Machine (SVM) exhibited more modest improvements, highlighting that while RFE significantly boosts the performance of complex models, its impact is less pronounced on simpler models. Overall, RFE is a superior feature selection technique, significantly optimizing model performance across various AI models for intrusion detection in IoT environments.

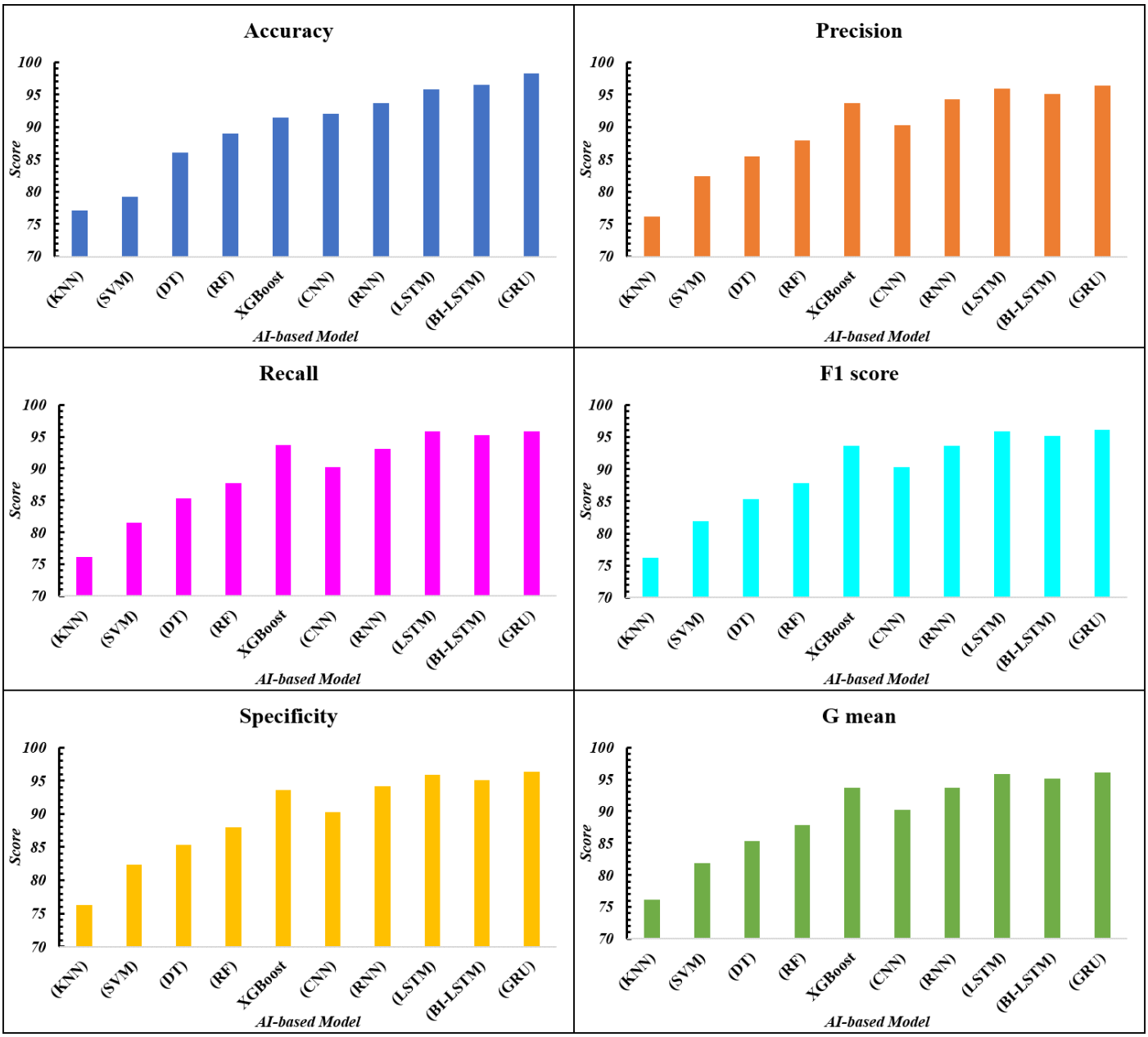


Figure 82: Performance of AI-based IDS models on Recursive Feature Elimination (RFE) selection Method on Edge-IIoT 2022 dataset

The comparative analysis of the feature selection techniques—forward selection, backward selection, and Recursive Feature Elimination (RFE) on the Edge-IIoT dataset reveals significant differences in model performance across various metrics: accuracy, precision, recall, F1 score, specificity, and G-mean. The RFE method consistently outperforms the forward and backward selection methods. For instance, in terms of accuracy, models using RFE show an improvement ratio ranging from 1.008 to 1.013 compared to forward selection and 1.030 to 1.050 compared to backward selection. Specifically, the K-Nearest Neighbors (KNN) model sees an accuracy increase of 1.2% with RFE over forward selection and 3% over backward selection. Similarly, the Support Vector Machine (SVM) model demonstrates a precision ratio increase of 5.8%, with RFE over forward selection and 9.1% over backward selection.

The Decision Trees (DT) model also shows a notable enhancement, with RFE improving recall by 4.2% over forward selection and 8.1% over backward selection. Random Forest (RF) models with RFE achieve a specificity ratio

improvement of 0.8% over forward selection and 4% over backward selection. The XGBoost model, mainly, sees a significant rise in precision, with RFE boosting it by 4% over forward selection and 9.8% over backward selection. Neural network models, such as the Convolutional Neural Network (CNN) and Long Short-Term Memory (LSTM), also benefit from RFE. The CNN model's F1 score with RFE is 0.2% higher than forward selection and 3.7% higher than backward selection, while the LSTM model shows an improvement in G-mean by 2% over forward selection and 5.1% over backward selection.

The Bidirectional LSTM (BI-LSTM) and Gated Recurrent Unit (GRU) models demonstrate minor but consistent improvements in precision and recall with RFE. The BI-LSTM model's recall ratio is almost identical with RFE and forward selection but shows a 1.2% increase over backward selection. The GRU model achieves a 0.1% higher G-mean with RFE than forward selection and 2.1% higher than backward selection. These results underscore RFE's efficacy in refining feature sets, leading to enhanced model performance across different metrics. The iterative removal of the least essential features in RFE ensures a more impactful and efficient feature selection process, thus consistently yielding better results than forward and backward selection methods.

To tackle the challenges posed by imbalanced datasets, we implemented a combination of oversampling and undersampling techniques during the data preprocessing phase. Specifically, SMOTE was employed to generate synthetic samples for minority classes, effectively mitigating the bias towards majority classes. Additionally, the recursive feature elimination (RFE) method proved instrumental in refining the feature set, ensuring that only the most relevant features were used, which helped enhance the detection of rare attack types.

Our approach significant improvements in recall and F1-score metrics, particularly for minority classes, as evidenced in the results for the GRU and BI-LSTM models. For example, on the Edge-IIoT dataset, the GRU model achieved an F1-score of 96.06% with forward selection and 97.79% with RFE, demonstrating the effectiveness of our strategies in handling class imbalance. These findings align with existing literature on the benefits of oversampling and advanced feature selection techniques, reinforcing the robustness of our proposed framework.

4.4. Findings and Discussion

This section presents a detailed examination of how different wrapper-based feature selection techniques forward selection, backward selection, and Recursive Feature Elimination (RFE) influence the performance metrics of AI-based models on the N-BaIoT and Edge-IIoT 2022 datasets.

4.4.1. Findings on the N-BaIoT Dataset

i. Forward Selection

Forward selection significantly enhances the performance of AI models on the N-BaIoT dataset. The Gated Recurrent Unit (GRU) model demonstrated the highest accuracy at 97.23%, surpassing the Long Short-Term Memory (LSTM) and Bidirectional LSTM (BI-LSTM) models, which achieved 96.56% and 95.26%, respectively. These results underscore GRU's superior capability to identify and leverage relevant features to boost detection accuracy. Additionally, GRU maintained a balanced performance across precision (96.36%), recall (95.90%), and F1 score (96.13%), indicating its robustness in reducing false positives and accurately classifying benign instances.

ii. Backward Selection

Backward selection had varied impacts across different models. Complex models like GRU, LSTM, and BI-LSTM maintained high performance, with accuracies of 95.53%, 93.54%, and 94.42%, respectively. In contrast, simpler models such as K-Nearest Neighbors (KNN) and Support Vector Machine (SVM) experienced significant drops in accuracy, achieving only 75.11% and 80.23%. This suggests that backward selection effectively enhances complex models by eliminating irrelevant features but may inadvertently discard crucial features needed by simpler models, thereby diminishing their classification efficacy.

iii. Recursive Feature Elimination (RFE)

RFE emerged as the most compelling feature selection method on the N-BaIoT dataset, delivering superior performance across all evaluated models. The GRU model, under RFE, achieved remarkable metrics, including an accuracy of 98.81%, precision of 97.93%, recall of 97.64%, and F1 score of 97.79%. The BI-LSTM model was followed closely, with an accuracy of 98.16%. RFE's iterative process of removing the least significant features ensures the retention of only the most impactful ones, optimizing model performance more effectively than forward or backward selection.

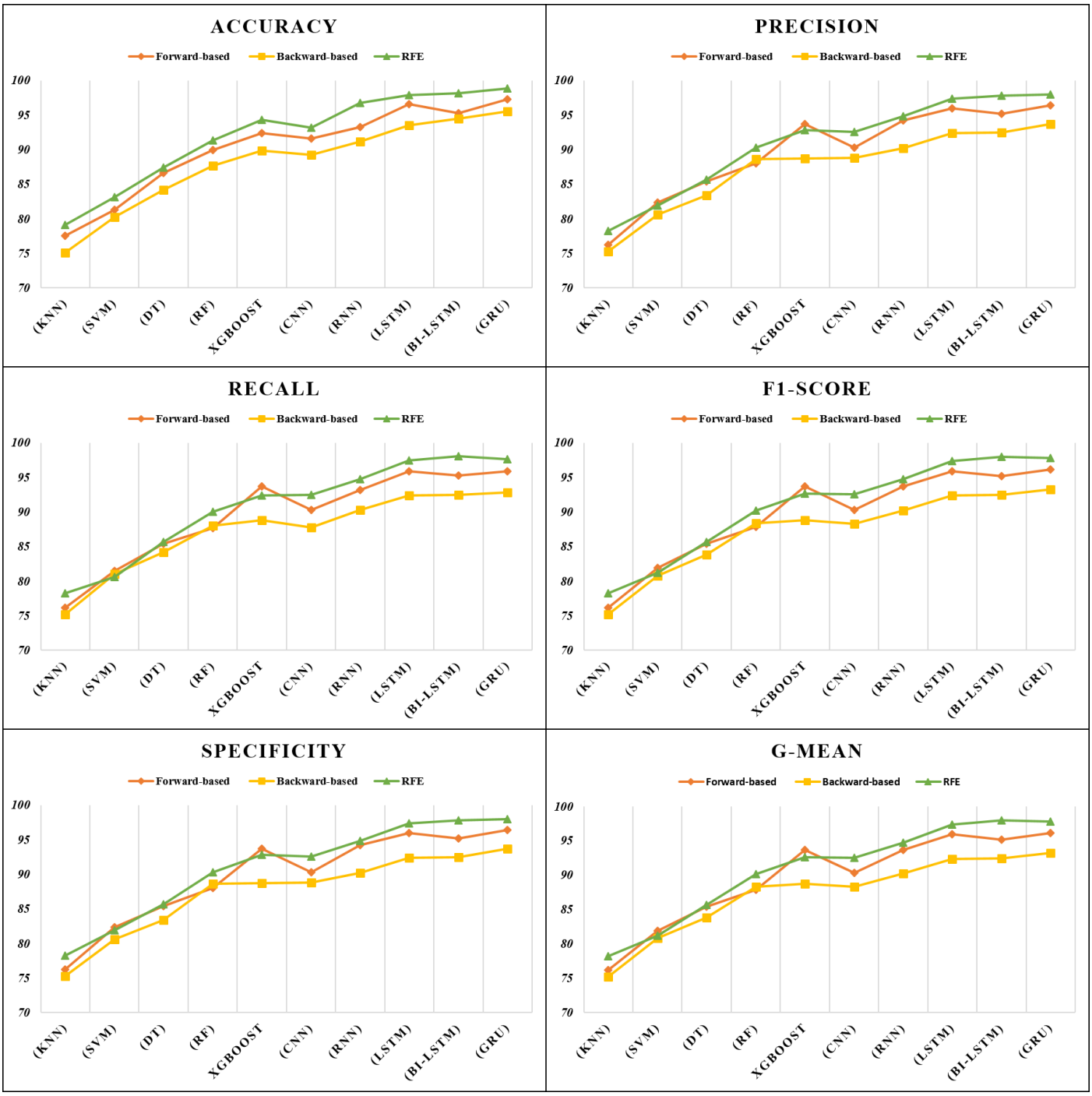


Figure 83: Comparison of AI-based IDS models on different Wrapper-based Feature selection methods using the N_BaIoT dataset.

4.4.2. Findings on the Edge-IIoT Dataset

i. Forward Selection

Forward selection substantially improved the performance of AI models on the Edge-IIoT dataset. The GRU model again led with an accuracy of 97.13%, precision, recall, and F1 score metrics of 96.24%, 95.90%, and 96.06%, respectively. This enhancement underscores the effectiveness of forward selection in boosting performance metrics across models, particularly those handling complex sequential data, such as LSTM and BI-LSTM, which also showed significant performance gains.

ii. Backward Selection

Backward selection refined model performance on the Edge-IIoT dataset by eliminating less relevant features. The GRU model achieved an accuracy of 95.13%, demonstrating the method's ability to optimize complex models. However, simpler models like KNN and SVM showed lower improvements, with KNN achieving an accuracy of 74.83%. This indicates that backward selection may be less effective for models requiring a broader feature set for optimal performance.

iii. Recursive Feature Elimination (RFE)

RFE produced the most pronounced improvements on the Edge-IIoT dataset. The GRU model achieved an impressive accuracy of 98.21%, further solidifying RFE's superiority. Other models, including BI-LSTM and LSTM, also saw substantial gains, achieving accuracies of 96.48% and 95.74%, respectively. RFE's ability to iteratively refine the feature set results in enhanced classification accuracy and robustness across various models.

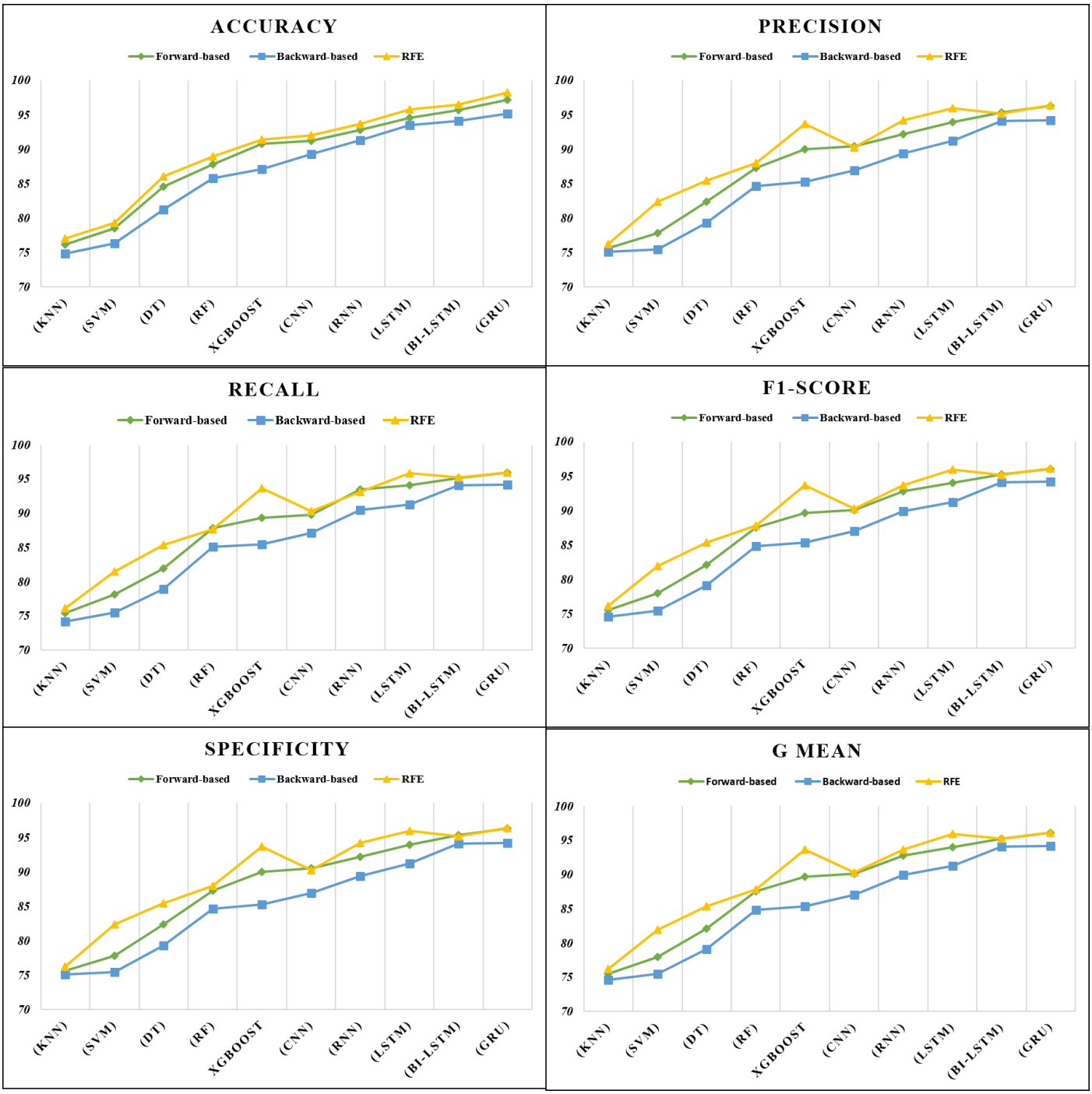


Figure 84: Comparison of AI-based IDS models on different Wrapper-based Feature selection methods using Edge-IIoT 2022 dataset

4.4.3. Overall Impact of Feature Selection Methods

Our comparative analysis underscores RFE as the superior feature selection technique in both datasets. On the N-BaIoT dataset, RFE improved model accuracy by up to 5.34% over forward selection and up to 5.6% over backward selection, as shown in Figure 20. Similarly, on the Edge-IIoT dataset, RFE outperformed forward selection by 1.008 to 1.013 times and backward selection by 1.030 to 1.050 times, as shown in Figure 21. These improvements were

particularly notable in complex models such as GRU, LSTM, and BI-LSTM, highlighting RFE's effectiveness in optimizing feature sets for advanced neural networks. In contrast, forward selection provided intermediate improvements, enhancing model performance but not matching RFE's effectiveness. Backward selection generally resulted in the lowest performance metrics, suggesting it may only be suitable for some model types, notably simpler ones.

In conclusion, RFE consistently outperformed forward and backward selection methods, making it the most compelling feature selection technique for enhancing the accuracy and reliability of AI-based intrusion detection systems in IoT environments. This comprehensive evaluation of feature selection methods provides valuable insights for researchers and practitioners aiming to optimize AI models for complex detection tasks.

5. Conclusion and Future Work

This section encapsulates the study findings and outlines the potential scope for future research.

5.1. Conclusion

This study thoroughly examined the efficacy of various wrapper-based feature selection methods—forward selection, backward selection, and Recursive Feature Elimination (RFE) when applied to ten state-of-the-art AI-based Intrusion Detection Systems (IDSs) in Industrial IoT (IIoT) environments. Our results, derived from testing on the N-BaIoT and Edge-IIoT 2022 datasets, consistently demonstrated that RFE significantly outperforms forward and backward selection methods. Specifically, RFE enhanced model performance by optimizing the feature sets, leading to higher accuracy, precision, recall, and F1 scores. The Gated Recurrent Unit (GRU) model exhibited the best performance, achieving remarkable accuracy and balanced metrics across both datasets. This highlights RFE's capacity to improve the robustness and reliability of complex AI models, particularly in scenarios requiring precise detection of cyber-attacks. Conversely, forward selection provided moderate improvements, while backward selection generally resulted in the lowest performance metrics, indicating its limited suitability for certain models, significantly simpler ones.

Our comprehensive analysis underscores the critical role of feature selection in developing efficient, high-performing IDSs for IIoT networks. By focusing on relevant features and eliminating redundant ones, RFE enhances detection accuracy and reduces computational overhead, making it the most effective method for optimizing AI-based IDSs in complex IIoT environments.

5.2. Future Work

Future research will extend this work by exploring several avenues to enhance further the effectiveness and applicability of AI-based IDSs in IIoT environments:

- ✓ **Integration of Real-Time Data:** Incorporating real-time data from live IIoT networks will provide a more dynamic and practical evaluation of IDS performance, ensuring the models can adapt to evolving threats and real-world conditions.
- ✓ **Hybrid Feature Selection Methods:** Investigating the combination of wrapper-based and filter-based feature selection methods may yield even more efficient and effective feature sets, potentially enhancing model performance beyond what RFE alone can achieve.
- ✓ **Deployment in Edge and Fog Computing Environments:** Evaluating the performance of AI-based IDSs in edge and fog computing scenarios will address the challenges of latency and resource constraints, making IDS deployment more practical for IIoT systems.
- ✓ **Enhancing Model Interpretability:** Developing techniques to improve the interpretability of AI models will facilitate their acceptance and trustworthiness, enabling better understanding and management of the IDSs by network administrators.

- ✓ **Exploring New AI Architectures:** Examining the potential of emerging AI architectures, such as Transformer models, may provide insights into further improving detection capabilities and computational efficiency.

- ✓ **Longitudinal Studies:** Conducting longitudinal studies to assess the long-term performance and adaptability of AI-based IDSs will ensure their sustainability and resilience against sophisticated, evolving cyber threats.

By addressing these future directions, we aim to develop more robust, adaptive, and efficient IDS solutions, thereby enhancing the overall security and resilience of IIoT networks against a wide array of cyber threats.

Chapter 8: Conclusion, Future Work and Societal Applications

This chapter summarizes the key findings of this research, highlighting its contributions to enhancing security and privacy in intrusion detection systems (IDS) using blockchain and AI-driven methodologies. Additionally, the chapter outlines the limitations of the study, proposes future research directions, and discusses the potential industrial and societal applications of the proposed framework.

8.1. Conclusion

The increasing integration of IoT in various domains has amplified the need for robust security mechanisms against cyber threats. This research proposed a hybrid blockchain-based intrusion detection system (IDS) leveraging AI-driven models to enhance network security. The proposed framework incorporated federated learning-based CNN-BiLSTM for anomaly detection, IBFT consensus for blockchain-based security, and Explainable AI (XAI) for model interpretability. The integration of Elliptic Curve Cryptography (ECC) and Zero-Knowledge Proofs (ZKP) ensured data confidentiality and privacy. Experimental evaluations demonstrated that the proposed IDS outperforms traditional security models in terms of detection accuracy, computational efficiency, and resilience to adversarial attacks. Comparative analysis against existing methodologies, including SAGBO-RSA, GBO-RSA, and ECC, highlighted the effectiveness of the proposed framework in reducing computational overhead while maintaining high security.

Despite its advantages, this study has certain limitations, including dependency on computational resources, scalability challenges in large-scale deployments, and potential latency introduced by blockchain operations. However, these challenges pave the way for future research directions aimed at optimizing security and efficiency. The insights gained from this research contribute to advancing IDS mechanisms, ensuring robust security for next-generation IoT environments.

8.2. Limitation of the study

Although the proposed framework significantly enhances IoT security, it has certain limitations that need further investigation. These limitations are primarily associated with computational demands, scalability, and blockchain integration complexity.

- The computational complexity of federated learning and deep learning-based IDS may require high-performance hardware, limiting its deployment in resource-constrained environments.
- The blockchain integration introduces latency, which may affect real-time intrusion detection efficiency in highly dynamic networks.
- The scalability of the proposed framework remains a challenge when applied to large-scale IoT infrastructures with heterogeneous devices and dynamic network conditions.
- The need for secure key management in ECC and ZKP implementations introduces an additional security overhead.
- Explainable AI (XAI) techniques require further refinement to provide more intuitive and human-understandable justifications for IDS decisions.

8.3. Potential Industrial Application

The proposed security framework has extensive industrial applications, addressing security, privacy, and intrusion detection challenges across multiple domains. Its integration of blockchain, AI, and cryptographic techniques ensures enhanced protection against cyber threats.

- **Smart Healthcare Systems:** Protects electronic health records (EHRs) and patient data from cyber threats by implementing blockchain-based access control and federated learning-powered anomaly detection to prevent unauthorized data modifications and breaches.
- **Autonomous Vehicles:** Enhances cybersecurity in autonomous vehicle networks by securing vehicle-to-vehicle (V2V) and vehicle-to-infrastructure (V2I) communications, reducing risks of cyber hijacking and unauthorized data manipulations.
- **Critical Infrastructure Protection:** Safeguards industrial control systems (ICS) and smart grid networks by detecting and mitigating cyber-physical attacks using AI-driven intrusion detection integrated with blockchain-based trust management mechanisms.
- **Financial Sector Security:** Prevents financial fraud, identity theft, and insider threats by leveraging federated learning-based anomaly detection and blockchain-based transaction security, ensuring transparency and secure digital payments.
- **Smart Cities and IoT Networks:** Enhances the security of interconnected urban infrastructure by deploying decentralized, AI-powered intrusion detection for smart traffic systems, surveillance networks, and public utilities.
- **Supply Chain Security:** Ensures data integrity and transparency in logistics and supply chain management by employing blockchain for secure transaction records and AI-driven anomaly detection to detect fraudulent activities.
- **Military and Defense Networks:** Strengthens cybersecurity in defense communication systems by using AI-driven threat intelligence and blockchain-based encrypted communication protocols to prevent cyber espionage and unauthorized data access.

8.4. Future Work

While this research provides a novel hybrid security framework, several areas require further exploration. Future work will focus on enhancing the proposed model's efficiency, scalability, and real-time adaptability.

- Lightweight AI Models: Develop resource-efficient AI-driven IDS models to support low-power IoT devices without compromising detection accuracy.
- Optimized Blockchain Consensus: Improve blockchain efficiency by designing a lightweight consensus mechanism to minimize computational overhead and transaction latency.
- Adaptive Intrusion Response: Implement self-healing mechanisms using reinforcement learning to autonomously mitigate detected threats in real-time.
- Cross-Domain Security Framework: Extend the proposed model to secure multi-cloud and edge computing environments with decentralized trust management.
- Enhanced XAI Techniques: Develop more interpretable AI models to improve transparency and trustworthiness in IDS decision-making processes.

8.5. Societal Applications

Beyond industrial implementations, the proposed framework significantly contributes to societal cybersecurity by addressing privacy concerns, securing critical data, and ensuring safer digital environments across various sectors.

- **Privacy Protection in Smart Homes**: Mitigates unauthorized access and cyber threats in IoT-enabled smart homes by deploying AI-driven intrusion detection and blockchain-based access control mechanisms, ensuring secure automation and data privacy.
- **Secure Public Health Data Management**: Safeguards sensitive healthcare records and epidemiological data from cyber breaches using cryptographic techniques, ensuring compliance with data protection regulations and enhancing trust in healthcare systems.

- **Digital Identity Protection**: Enhances security in digital identity management by using blockchain-based decentralized authentication, reducing risks of identity theft, data breaches, and unauthorized access to online services.
- **Cybersecurity Awareness and Education**: Supports cyber literacy initiatives by providing an explainable AI-based security model, helping individuals and organizations understand and mitigate cybersecurity threats effectively.
- **Emergency Communication Systems**: Strengthens disaster response networks by ensuring secure and resilient communication infrastructure using blockchain and AI-based anomaly detection for rapid and uninterrupted emergency response.
- **Ethical AI and Decision Transparency**: Promotes fairness and accountability in AI-driven security solutions by integrating explainable AI techniques, ensuring transparent decision-making processes in cybersecurity applications.
- **Child Online Safety**: Protects minors from cyber threats by implementing AI-driven monitoring systems in smart devices, detecting malicious activities, and preventing exposure to harmful digital content.