# A Novel Benchmark Dataset and CNN-Based Threat Detection Model

**A PROJECT REPORT**

***Submitted by***

**ANANTHA KRISHNAN B (211420104017)**

**NAVEEN R (211420104177)**

**MEIYARASAN A (211420104327)**

***in partial fulfillment for the award of the degree***

***of***

**BACHELOR OF ENGINEERING**

***in***

**COMPUTER SCIENCE AND ENGINEERING**


**PANIMALAR ENGINEERING COLLEGE**

**(An Autonomous Institution, Affiliated to Anna University, Chennai)**

**MARCH 2024**

# PANIMALAR ENGINEERING COLLEGE

**(An Autonomous Institution, Affiliated to Anna University, Chennai)**

## BONAFIDE CERTIFICATE

Certified that this project report **"A Novel Benchmark Dataset and CNN-Based Threat Detection Model**" is the Bonafide work of **"ANANTHA KRISHNAN B (211420104017), NAVEEN R (211420104177), MEIYARASAN A (2114020104327)"** who carried out the project workunder my supervision.

**Signature of the HOD with date**

**Dr. L. JABASHEELA, M.E., Ph.D.,**

**PROFESSOR AND HEAD,**

Department of Computer Science and Engineering,
Panimalar Engineering College,
Chennai - 123

**Signature of the Supervisor with date**

**Mr. M. MAHENDRAN, M.Tech.,**
**SUPERVISOR**
**ASSITANT PROFESSOR,**

Department of Computer Science and Engineering,
Panimalar Engineering College,
Chennai - 123

Certified that the above candidate(s) was examined in the End Semester Project Viva-Voce Examination held on **25.03.2024 (FN)**

**INTERNAL EXAMINER** **EXTERNAL EXAMINER**

# DECLARATION BY THE STUDENTS

We **ANANTHA KRISHNAN B (211420104017), NAVEEN R (211420104177), MEIYARASAN A (2114020104327)** hereby declare that this project report titled **"A Novel Benchmark Dataset and CNN-Based Threat Detection Model",** under the guidance of **Mr. M. MAHENDRAN M.Tech.,(Ph.D.).,** is the original work done by us and we have not plagiarized or submitted to any other degree in any university by us.

**ANANTHA KRISHNAN B (211420104017)**

**NAVEEN R (211420104177)**

**MEIYARASAN A (211420104327)**

# ACKNOWLEDGEMENT

Our profound gratitude is directed towards our esteemed Secretary and Correspondent, **Dr. P. CHINNADURAI, M.A., Ph.D**., for his fervent encouragement.His inspirational support proved instrumental in galvanizing our efforts, ultimately contributing significantly to the successful completion of this project.

We wish to express our deep gratitude to our Directors, **Tmt. C. VIJAYARAJESWARI, Dr. C. SAKTHIKUMAR, M.E., Ph.D., and Dr. SARANYASREE SAKTHIKUMAR, B.E., M.B.A., Ph.D.,** for graciously affording us the essential resources and facilities for undertaking of this project.

Our gratitude is also extended to our Principal, **Dr. K. MANI, M.E., Ph.D.,** whose facilitation proved pivotal in the successful completion of this project.

We express our heartfelt thanks to **Dr. L. JABASHEELA, M.E., Ph.D.,** Head of the Department of Computer Science and Engineering, for granting the necessary facilities that contributed to the timely and successful completion of project.

We would like to express our sincere thanks to **Project Coordinator Dr. G. SENTHIL KUMAR, M.C.A., M.B.A., M.Phil., M.E., Ph.D**., and **Project Guide Mr. M. MAHENDRAN, M.Tech.,(Ph.D).,** and all the faculty members of the Departmentof CSE for their unwavering support for the successful completion of the project.

**ANANTHA KRISHNAN B (211420104017)**

**NAVEEN R (211420104177)**

**MEIYARASAN A (211420104327)**

# ABSTRACT

The Internet of Things (IoT) has become increasingly integral to modern society, revolutionizing various industries by enabling new capabilities and functionalities. This proliferation of IoT solutions has been particularly notable in sectors such as transportation and healthcare, where innovative services are continually being developed to enhance efficiency and improve quality of life. Over the past decade, there has been a remarkable surge in IoT connections, with projections indicating continued growth across diverse domains in the years to come. However, alongside the promising advancements facilitated by IoT technology, significant challenges persist, particularly in ensuring the efficient and secure operation of IoT ecosystems. Key issues such as interoperability, security vulnerabilities, and the establishment of robust standards remain areas of concern that demand ongoing attention and resolution. Despite efforts to address security concerns, including the development of datasets containing examples of attacks against IoT devices, there are limitations in the scope and comprehensiveness of existing datasets. Many current initiatives focus on a limited range of attack scenarios and fail to account for the complexities of real-world IoT network topologies. Furthermore, certain types of potential attacks are often overlooked or inadequately represented in these datasets. Recognizing these gaps, the primary objective of this research initiative is to introduce a novel and extensive IoT attack dataset aimed at advancing the development of security analytics applications tailored for real-world IoT deployments. To achieve this goal, a comprehensive set of 33 distinct attacks has been executed within an IoT network environment comprising 105 interconnected devices. These attacks are systematically categorized into seven distinct classes, including Distributed Denial of Service (DDoS), Denial of Service (DoS), Recon, web-based exploits, brute-force attacks, spoofing, and instances resembling the infamous Mirai botnet. Importantly, all attacks within this dataset are simulated by malicious IoT devices targeting other devices within the network, reflecting the complex dynamics and potential vulnerabilities inherent in IoT ecosystems.

## LIST OF TABLES

## LIST OF FIGURES

## LIST OF ABBREVIATIONS

| | | |
|---|---|---|
| RECON | - | Reconnaissance |
| SVM | - | Support Vector Machines |
| SGD | - | Stochastic Gradient descent |
| IOT | - | Internet of Things |
| IDS | - | Intrusion Detection System |
| DDOS | - | Distributed Denial of Service |
| ML | - | Machine learning |
| ICS | - | Industrial Control System |
| IOMT | - | Internet of Medical things |
| CPS | - | Cyber Physical System |
| DBMS | - | Data Base Management system |
| EHR | - | Electronic health records |
| TCP | - | Transmission Control Protocol |
| IP | - | Internet Protocol |
| NFV | - | Network Function Virtualization |
| SDN | - | Software Defined Networking |
| DOS | - | Denial of Service |
| AI | - | Artificial Intelligence |
| HIDS | - | Host-based Intrusion Detection System |
| NIDS | - | Network Intrusion Detection System |

# TABLE OF CONTENTS

# CHAPTER 1

## 1.1 Introduction

The rapid development of the Internet of Things (IoT) has completely transformed many industries, such as smart homes, smart agriculture, healthcare, and more [1]. According to survey data, the number of IoT devices is projected to exceed 4.1 billion by 2025 [2]. In everyday life, IoT devices play a crucial role in people's lives. However, the extensive connectivity of these devices to the internet exposes them to various security risks. For example, IoT devices exchange information over the internet and are suscep -tible to numerous network attacks, compromising their security. According to a report by Nozomi Networks, new IoT botnet attacks increased rapidly in the first half of 2020, with 57% of IoT devices being vulnerable targets [3]. Furthermore, attackers can launch denial-of-service (DoS) attacks, depleting network and device resources [4]. Therefore, enhancing the security of IoT devices has become a critical area of research [5]. To mitigate the risks posed by different types of attacks, researchers are developing intrusion detection systems to identify malicious behavior in networks. Intrusion detection systems monitor systems in real-time and issue warnings in case of any anomalies, thereby enhancing the security of communication. In recent years, machine learning, with its rapid development, has found extensive applications in the field of intrusion detection [6,7]. Machine learning algorithms offer unique advantages compared to traditional detection methods. They can not only learn complex patterns and rules from large volumes of data but also handle high-dimensional and nonlinear data, making them more suitable for intrusion detection in Electronics 2023, 12, 4289 2 of 21 Furthermore, with the advancement of networks, a significant amount of network data, including samples of various intrusion and abnormal behaviors, has been accumulated. This rich dataset provides ample training samples for machine learning, ensuring excellent detection performance of machine learning algorithms. However, despite the achievements of machine learning algorithms, there are still some challenges and issues that need to be addressed.

the system may struggle to efficiently process and analyze a large amount of network traffic data, leading to delays or inability to meet real-time requirements in detection. Additionally, in the face of frequent network attacks on IoT devices and the constant evolution of attack techniques, intrusion detection systems are under increasing pressure to detect such attacks [8]. The generation of a vast amount of network traffic data introduces significant redundancy and irrelevant features. Redundant features in the data can lead to overfitting during the model learning process, ultimately diminishing detection performance [9]. One effective approach to address feature redundancy is feature selection. Feature selection plays a crucial role in machine learning-based intrusion detection systems, reducing the dimensionality of the dataset, lowering training time and computational costs, while improving model performance [10]. Recently, metaheuristic algorithms have gained considerable attention in the field of feature selection due to their excellent global search capabilities [11]. Commonly used metaheuristic algorithms include genetic algorithm (GA), particle swarm optimization (PSO) [12], whale optimization algorithm (WOA) [13], grey wolf optimization (GWO) [14], and simulated annealing (SA), among others. Among these algorithms, GWO has garnered considerable interest due to its ease of implementation, fast convergence speed, and strong optimization capabilities. To better utilize GWO for feature selection problems, Emary et al. proposed a novel binary grey wolf optimization (BGWO) [15]. Hsu et al. [16] pointed out that hybrid feature selection methods outperform individual feature selection methods. Furthermore, recursive feature elimination (RFE), a wrapper feature selection method, strikes a good balance between accuracy and runtime, preserving a certain level of accuracy while reducing runtime. In this study, we propose a novel two-stage feature selection method that combines the strengths of BGWO and RFE. This method reduces the search space for feature subsets and eliminates redundant features.

## 1.2 PROBLEM DEFINITION

In the twenty-first century, the Internet of Things (IoT) has become an essential part of daily life, with diverse applications including health, home automation, smart metering, smart cars, and the Smart Factory. Significant advancements in miniaturization, micro-appliance manufacturing, and server connectivity have facilitated the development of innovative IoT products, revolutionizing various communication-based processes. IoT has significantly impacted industrial systems and healthcare, giving rise to specialized sectors like the Industrial Internet of Things (IoT) and the Internet of Medical Things (IoMT). IoT refers to a group of industrial machines equipped with sensors and connected to networks for data collection and exchange. These devices have been instrumental in the adoption of Industry 4.0, a global industrial network of unprecedented scale. The integration of IoT with automated processes has proven highly beneficial for many industries, enabling tasks that were previously beyond the capabilities of automation technology. IoMT, on the other hand, encompasses IoT devices specifically designed for critical healthcare functions. It comprises a range of medical instruments and technologies that can interface with hospital communication systems through online channels. As healthcare costs rise with growing populations and increasing disease burdens, IoT innovations offer new avenues for medical diagnosis and treatment. Through interconnected IoT devices, healthcare providers can deliver and monitor medical care more effectively. Both IoT and IoMT rely on devices connected to the Internet. However, the proliferation of new IoT devices also brings an increased risk of hacking and cybersecurity threats. Machine learning (ML) is the field of enabling computers to act without explicit programming instructions, thereby making coding more accessible. While traditional coding involves running data and algorithms on a machine to produce results, ML involves running data and outcomes to generate a program. ML finds applications in various domains such as web search, finance, e-commerce, robotics, social networks, and IT security. With the rise of the Internet of Things (IoT), numerous devices are expected to be interconnected, posing significant security threats if not adequately protected.

## 1.3 OBJECTIVE

Certainly, here are the objectives for the discussion on IoT communication protocols:

**Introduction:**

- Introduce the importance of communication protocols in the IoT domain.
- Highlight the role of communication protocols in enabling interoperability, reliability, and security in IoT ecosystems.

**MQTT (Message Queuing Telemetry Transport):**

- Explain the key features of MQTT as a lightweight and efficient IoT communication protocol.
- Describe the publish-subscribe model used by MQTT and its benefits for asynchronous communication.
- Discuss MQTT's suitability for resource-constrained environments and real-time data transmission applications.

**CoAP (Constrained Application Protocol):**

- Describe CoAP as a protocol designed for IoT devices operating in low-power, low-bandwidth networks.
- Explain CoAP's RESTful communication paradigm over UDP and its advantages for energy-efficient IoT deployments.
- Highlight CoAP's features such as resource discovery, caching, and observe for efficient data exchange in dynamic environments.

**HTTP (Hypertext Transfer Protocol):**

- Discuss the continued relevance of HTTP in IoT deployments, particularly for web-based applications and cloud integration.
- Explain how HTTP's request-response model enables seamless interaction between IoT devices and web servers, APIs, and cloud platforms.
- Address challenges associated with HTTP in resource-constrained environments and optimizations like HTTP/2 and HTTP/3 to mitigate them.

**DDS (Data Distribution Service):**

- Introduce DDS as a protocol standard for mission-critical IoT applications requiring deterministic communication and real-time data exchange.
- Describe DDS's decentralized publish-subscribe architecture and its support for Quality of Service (QoS) policies.
- Highlight DDS's applications in domains such as healthcare, aerospace, and autonomous systems where reliability and performance are paramount.

**Conclusion:**

- Summarize the key points discussed regarding MQTT, CoAP, HTTP, and DDS as IoT communication protocols.
- Emphasize the importance of selecting appropriate communication protocols based on application requirements and deployment scenarios.
- Discuss the implications of IoT communication protocols for future innovation and connectivity in the IoT landscape.

# CHAPTER 2

# LITERATURE REVIEW

## 2.1 BACKGROUND AND LITERATURE REVIEW:

The Internet of Things (IoT) refers to a network of interconnected devices, systems, and objects that communicate with each other via the internet. It interacts with both its internal and external environment, detecting and responding to changes in real-time. IoT technologies offer innovative solutions to various aspects of life, ultimately enhancing the quality of human life. By enabling objects to connect and communicate with each other, IoT makes the environment smarter and more efficient. It facilitates seamless connectivity between devices, allowing them to collect data from sensors and other sources, which can then be transmitted wirelessly to smartphones or computers for interpretation and analysis. IoT applications span across different sectors, including household appliances, energy efficiency, environmental monitoring, and business operations. From distribution and logistics to automation and surveillance, IoT plays a pivotal role in modernizing various industries and systems. By distributing data across interconnected devices, IoT creates opportunities for application development and innovation, enabling businesses to anticipate market trends and meet the evolving needs of consumers in today's fast- paced world. Overall, efficient solutions to various challenges and enhancing convenience and productivity.

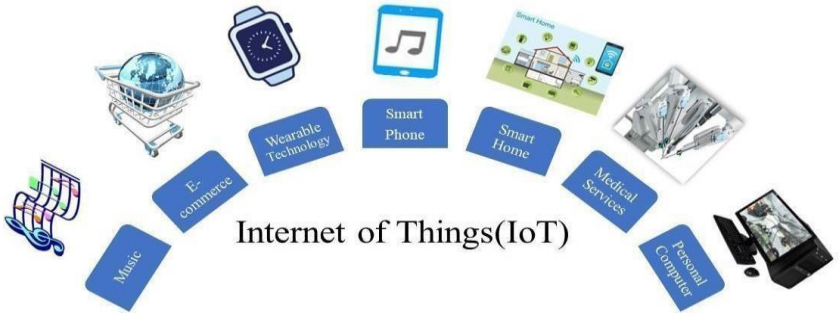



**Figure 2.1 Internet of Things**

## 2.2 INDUSTRIAL INTERNET OF THINGS (IoT):

The Internet of Things (IoT) is often heralded as a technological breakthrough capable of addressing a wide array of contemporary social challenges, including the development of smarter cities, traffic monitoring, pollution management, and advancements in healthcare, among others. Within the realm of IoT, the Industrial Internet of Things (IoT) emerges as a subset, encompassing machine-to-machine (M2M) communication and technological advancements specifically tailored for industrial applications, often integrating robotic techniques. Figure 2.2 illustrates the evolutionary trajectory from IoT to IoT and Industry 4.0. The advent of IoT has ushered in opportunities for more efficient and sustainable management of manufacturing processes. Leveraging networks with limited coverage areas, IoT systems typically operate with low transmit power requirements, facilitating their implementation in diverse industrial settings.

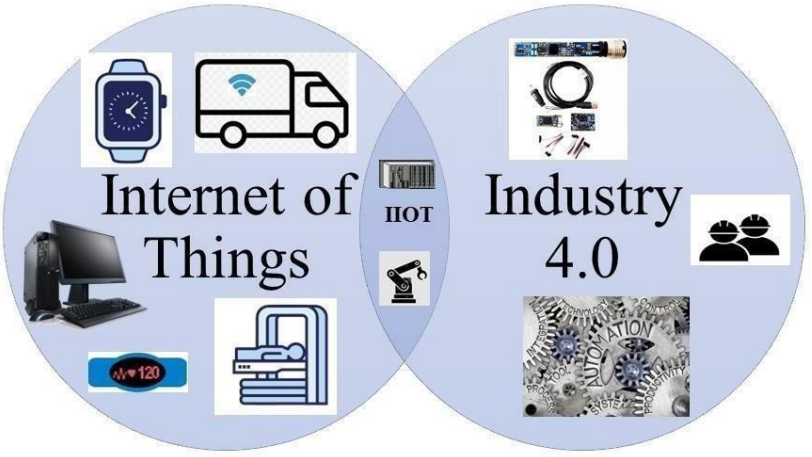



**Figure 2.2. The Industrial Internet of Things**

IoT technology plays a crucial role in enhancing the capabilities of the Industrial Internet of Things (IoT) to optimize construction operations and industrial processes. It functions as a mechanism for capturing, sharing, and interpreting data across a distributed system interconnected by communication networks, facilitating faster decision-making processes. For example, within a commercial operation, IoT enables the prediction and rectification of equipment flaws in a timely manner, preempting

potential breakdowns or failures of machinery. IoT contributes to the establishment of interconnected home workplaces, resulting in savings of time, resources, and costs. These small-scale instruments are seamlessly integrated into traditional Industrial Control Systems (ICS), further enhancing operational efficiency and productivity.

## 2.3 MEDICAL INTERNET OF THINGS (IoMT):

The Internet of Medical Things (IoMT) is a term commonly used in the medical industry to describe IoT devices designed to facilitate essential healthcare functions. It encompasses a range of healthcare instruments and technologies that can communicate with medical data platforms through networked devices. Figure 2.3 provides a succinct illustration of IoMT, depicting medical devices connected to patients, enabling doctors to remotely prescribe medications and treatments to patients.

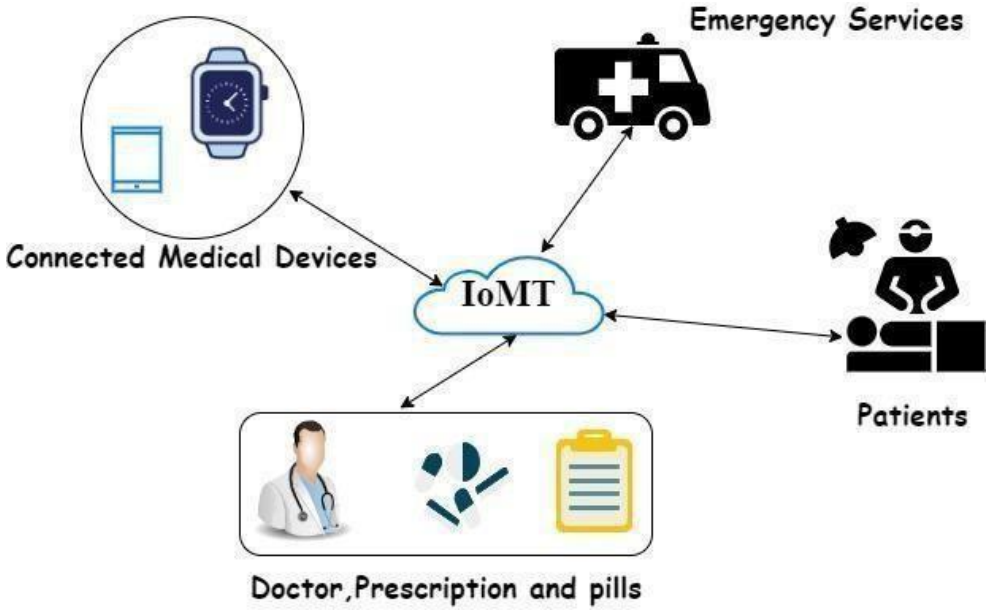



**Figure 2.3. The Internet of Medical Thing**

The Internet of Medical Things (IoMT) offers numerous advantages for both patients and healthcare professionals. Many IoMT devices are designed to continuously monitor critical patient parameters, providing insights that may not be feasible during brief office visits. For instance, by continuously tracking data such as blood pressure or pulse rate,

physicians can more accurately diagnose diseases and tailor individualized treatment plans. IoMT enables healthcare delivery and guidance through internet connectivity, allowing devices to collect data from patients' homes and securely transmit it to healthcare providers without the need for clinic visits. This real-time monitoring empowers patients to track their health status continuously, rather than relying solely on periodic check-ups. While IoMT devices provide numerous benefits, including convenience, enhanced patient support, and cost reduction, they also pose security risks that must be addressed. One significant threat is the potential theft of sensitive medical information, which could lead to embarrassment or harm for individuals if disclosed. Hackers target not only consumer health data but also healthcare providers and organizations. Healthcare fraud, where criminals use stolen information to obtain medications or medical services, is a legitimate concern. Additionally, malicious hackers may target implantable medical devices, such as smart cardiac implants or insulin pumps, to extort individuals or compromise their safety. These attacks highlight the vulnerability of IoMT devices as potential entry points or direct targets for cyber threats.

## 2.4 MACHINE LEARNING OVERVIEW

In today's digitally pervasive environment, with the ubiquitous presence of small-scale devices like cell phones and large-scale computing infrastructure such as cloud technology and online financial services, vast amounts of data are constantly generated, stored, processed, and utilized to produce outcomes in various application domains. Consequently, safeguarding data, devices, and user privacy in the digital realm has become a paramount concern for individuals, businesses, and governments worldwide. Machine Learning (ML) has gained significant traction in cybersecurity, offering solutions such as intrusion detection and multifactor authentication for consumer verification. However, ML systems are susceptible to attacks during both the training and evaluation phases, leading to potential performance degradation and data breaches.

Additionally, it conducts a thorough examination of publications addressing the security implications of IoT and IoMT technologies. Decision Tree: The decision tree presents information in a hierarchical structure resembling branches and leaves, where branches represent attributes influencing classifications and leaves signify categorizations. Common methods like the Gini index and Information Gain help determine the best attributes for splitting datasets. When classifying unknown samples, distinct features are used instead of the decision tree itself. Classification involves following a path from the root node to the leaf node. Decision tree induction employs a greedy approach, recursively dividing and conquering to generate the tree. Its advantages include intuitive visualization, accurate classification, and straightforward implementation. However, it tends to favor attributes with higher value, which is a notable drawback. For further insights, refer to Almseidin et al. (2017) and Buczak & Guven (2016). Support Vector Machine (SVM): SVM categorizes data by finding an optimal line that separates different categories using maximum margins. It constructs a hyperplane to label new data based on labeled training input. Additional information on SVM can be found in Beaver et al. (2013), Buczak & Guven (2016), and Patrascu & Patriciu (2015). Random Forest: Random Forest comprises multiple trees that independently select categories, and the category with the most votes becomes the final outcome. While it stems from decision trees, it operates differently by using a group of trees to select categories, addressing overfitting issues. Refer to Beaver et al. (2013), Almseidin et al. (2017), Meda et al. (2014), and Beaver et al. (2018) for more on Random Forest. K-Nearest Neighbor (KNN): KNN categorizes data based on its nearest neighbors, typically determined by Euclidean distance in a numerical feature space. It assigns the unidentified sample to the most common category among its k closest neighbors. KNN is a slow learner as it retains all training data until classification is required. Plaistow et al. (2020), Buczak & Guven (2016), and Liu et al. (2018) offer additional insights. Logistic Regression: Logistic regression is a mathematical approach for analyzing data, determining outcomes based on one or more independent variables. It aims to find the best-fit line between dependent and independent variables.

## 2.5 MACHINE LEARNING ALGORITHMS IN CYBERSECURITY SYSTEMS:

The Industrial Internet of Things (IoT) plays a crucial role in modern industrial operations by enabling connectivity and data exchange across various devices and digital controllers globally. However, the adoption of IoT and digitalization in industries is hindered by cybersecurity concerns. IoT vendors are increasingly focusing on addressing security issues (Hassanzadeh et al., 2015). Machine learning technologies offer promising solutions for addressing cybersecurity challenges and scenarios related to IoT. Automated machine learning methods provide a more consistent, systematic, and manageable approach to security. These advancements open up new avenues for effectively designing and operating emerging production connectivity platforms in a secure and reliable manner (Hussain et al., 2020). Based on a comprehensive analysis of previous research on Industrial Internet of Things (IoT) security, Yu and Guo (2019) have classified security issues into two main categories: those that are common to both IoT and IoT, and those specific to manufacturing. Among the current research pursuits in this area are database security, Cyber-Physical Systems (CPS) safety, key generation for device connectivity, and information regulation. Addressing these security challenges requires the development of enhanced security approaches and protocols tailored to traditional technology sectors integrating IoT. Security control stands out as a critical concern for both IoT and IoT, prompting experts to seek more accurate and reliable verification systems suitable for low-cost and resource-limited IoT-enabled devices. A significant challenge emerging in the IoT/IoT landscape is efficiently and securely managing large-scale deployments of devices. Blockchain technology, known for its applications in permissioned identity validation, access tracking, and product supervision, presents a potential solution for managing IoT/IoT assets. Das and Morris (2017) proposed a moment authentication framework, a credential procedure for rapid transmission verification, and a Distributed Throughput Management structure for blockchain systems, facilitating efficient transmissions among numerous IoT devices

while ensuring high processing power and compact design. The study by Karmakar et al. (2019) offers an extensive overview of the Industrial Internet of Things (IoT), tracing its evolution and transition into IoT, exploring its structure, case studies, and enterprise implementations. The objective of the research is to assess the current state of IoT and provide suggestions for potential research directions. Consequently, any architecture utilizing IoT should also prioritize security considerations. The authors delve into various IoT applications in commercial sectors, highlighting proactive surveillance services as particularly crucial. Machine learning techniques are employed to enable comprehensive monitoring of various parameters such as mobility, temperature, humidity, pressure, and energy consumption. Certain machine learning algorithms can even predict potential failures, aiming to minimize harm, environmental issues, security incidents, and malfunctions. While sensors can detect device health data and issue alerts if necessary, they cannot predict the reasons or timing of system failures. Therefore, the focus shifts towards developing platforms capable of generating credible forecasts for sustainability. The researchers recommend further investigation into safety issues related to IoT for future projects. On the other hand, Bhamare et al. (2020) examine the transition of Industrial Control Systems (ICS) from isolated networks to cloud-based platforms. The study investigates the significant contributions made by both commercial and academic organizations towards establishing reliable ICSs, with a particular emphasis on the application of machine learning technologies to enhance ICS cybersecurity. This research could aid in addressing the challenges of safeguarding corporate operations, especially during the transition to cloud environments. The increasing interconnectedness of the web has led to a surge in cyber threats, many of which can have catastrophic consequences. Malware stands out as one of the most commonly employed tools in cybercrime, exploiting existing vulnerabilities or leveraging advanced technical features to achieve malicious objectives. Consequently, there is a pressing need for the development of more sophisticated and robust malware defense systems within the cybersecurity industry (Jang-Jaccard and Nepal, 2014).

The researchers also examine threats to IoT cloud systems and detail measures taken to safeguard SCADA networks. These measures include connection isolation, continuous monitoring and assessment, log analysis, Internet Traffic Assessment, regular updating and patching, proxy solutions, and other recommended strategies. The researchers emphasize the critical importance of securing these networks, as attacks can have devastating consequences for both SCADA networks and the individuals who rely on them. Nakamura & Ribeiro (2018) propose a hazard analysis technique that prioritizes secrecy, security, adaptability, and consistency. The hazard analysis methodology comprises ten stages, wherein hazard elements assess the likelihood of a hazard particle exploiting vulnerabilities, thereby transforming a risk into an incident that affects multiple stakeholders. The authors illustrate this with an example of a doctor performing remote surgery while controlling a robot.

## 2.6 SECURITY IN MEDICAL HEALTHCARE:

Despite the numerous advantages offered by portable devices, there are significant concerns regarding confidentiality and security in mobile health (mHealth) applications (S. Bhuyan et al., 2017). To facilitate the expansion of mHealth, which relies heavily on trustworthiness, effective cryptographic measures to prevent data theft are crucial. Researchers have examined the confidentiality and security implications of mHealth from various perspectives. One of the challenges in ensuring the security of medical data lies in the lack of straightforward implementation of Database Management System (DBMS) security features due to existing safety compliance gaps. Medical technology has the potential to protect, prolong, and enhance people's lives (Coventry & Branley, 2018). Innovations such as telemedicine enable remote therapy administration, sometimes across borders, while also facilitating the management of electronic health records (EHRs). However, medical technologies are susceptible to security breaches due to networked infrastructure, readily accessible gateways, outdated technologies, and a lack of cybersecurity guidance. Despite the apparent focus on patient welfare, medical advancements store a wealth of sensitive and highly confidential data. Given the importance of medical identity, unauthorized access is often driven by personal gain.

The advancement of computer technologies has ushered in a revolutionary era across all industries, leveraging innovations in ways that capitalize on new features and possibilities (Tiginyanu et al., 2020). However, due to the glaring lack of security measures and the significance of the information generated or stored, the medical system remains a prime target for cyber threats. It is imperative to initiate action at the operational level, ideally at the national level, to ensure that healthcare technology providers adhere to predefined security standards. National regulations can be utilized to enforce and oversee security standards for computers or similar devices used in the industry. The study conducted by Deogirikar & Vidhate (2017) aims to explore various types of security risks, their scope, and impact on businesses. The paper discusses several IoT threats and existing countermeasures, categorizing these threats based on their potential to breach the system. IoT inherits the vulnerabilities of traditional system architecture, as it employs a structure akin to conventional systems, facilitating communication among diverse devices. Experts have proposed numerous strategies to combat these attacks. However, implementing comprehensive security frameworks and protocols at present entails significant computational costs and power consumption, which may be unsustainable with IoT technologies and devices. The internet connects us to the world through personal health monitors, proximity systems, smart homes, connected vehicles, and automated interactions (Sadique et al., 2018). The rapid advancement of IoT introduces novel security challenges and areas for further research, necessitating attention. While TCP/IP is commonly used for internet communication, IoT applications may require a short-range communication network to connect to a centrally controlled unit through which data is conveyed to the server. There are opportunities for research in developing, implementing, and validating lightweight methods for data security in IoT systems. While encrypted communications protect data in transit, the security of IoT sensitive content remains a complex issue, especially on the web and smartphones. Utilizing AI and ML to safeguard IoT systems represents a burgeoning research area. Network automation and intent-based networking, employing AI and ML to manage network access and elements, are also hot topics.

**Table 2.1 Summary of Literature**

| | Description | Merits | Challenges | Research Gaps |
|---|---|---|---|---|
| **"Security in Healthrisk"** | Discusses the importance of security measures in the medical system due to the vulnerability of medical data. Proposes national regulations to enforce security standards. | - Emphasizes the critical need for security in the healthcare industry. - Suggests practical measures for enforcing security standards. | - Lack of specific technical solutions for healthcare security. - Difficulty in implementing and enforcing national regulations in diverse healthcare settings. | Detailed exploration of specific technical solutions for healthcare security. |
| **"Safety Risks in IoT"** | Investigates various safety risks associated with IoT systems and their impact on businesses. Proposes strategies to mitigate IoT threats. | - Provides insights into different types of IoT threats and their implications. - Offers potential solutions to address IoT security risks. | - Focuses more on identifying threats than on practical implementation strategies. | Detailed exploration of implementation strategies for mitigating IoT threats. |
| **"Advanced IoT Development"** | Explores the security challenges posed by the proliferation of | - Highlights emerging security issues in IoT development. - | - Offers limited detail on specific security measures and implementation strategies. - Does | Detailed exploration of specific AI-driven security solutions for IoT |
| | IoT devices and suggests avenues for further research. Discusses the use of AI and ML in IoT security and network management. | Identifies areas for future research, such as AI-driven security solutions. | not provide an exhaustive overview of existing IoT security practices. | devices. |

The literature surveys explore various aspects of security in different domains, including healthcare and the Internet of Things (IoT). "Security in Healthcare" underscores the critical need for robust security measures in the medical sector due to the sensitive nature of healthcare data. It suggests implementing national regulations to enforce security standards but lacks specific technical solutions and may face challenges in implementation. Similarly, "Safety Risks in IoT" delves into the myriad threats facing IoT systems and proposes strategies to mitigate them. While it provides valuable insights into IoT threats and potential solutions, it focuses more on identifying challenges rather than offering practical implementation strategies. Lastly, "Advanced IoT Development" discusses the evolving security challenges posed by the proliferation of IoT devices and suggests areas for future research, including the use of AI and ML in IoT security. However, it offers limited detail on specific security measures and may benefit from a more comprehensive overview of existing IoT security practices. Overall, the literature surveys highlight the importance of security across different sectors and suggest avenues for further research and development in addressing emerging threats and vulnerabilities.

# CHAPTER 3

# THEORETICAL BACKGROUND

## SYSTEM ARCHITECTURE

### 3.1 INSTRUSION DETECTION SYSTEMS:

The IDS is in general, a common architecture for monitoring as well as tracking events within computerized distributed system so as to achieve signals with potential dangers and detecting fraudulent behavior. An IDS indeed system/program which discovers these invasions. Intrusion detection is used to discern violations of an organization's security policy. Individuals outside the company (namely, hackers) or staff within its organization (viz., employee) may commit these infarctions. Though research is being done on detecting attacker breaches, internal breaches are more difficult to identify. When an attack is identified early sufficiently, the attacker could be identified then eliminated even before damage has been caused or information gets exposed. However, though this identification is indeed not timely able to stop an invader, the sooner a breach is recognized, much less damage is caused and also the quicker restoration is possible. A good IDS could act as a deterrent, inhibiting penetration.

### 3.2 OBJECTIVES OF INTRUSION DETECTION SYSTEMS:

The objectives of IDS could be defined as follows:

1. Identify as much assault types as feasible, comprising such carried out by attackers and insiders.
2. Identify as precisely as achievable, reducing the figure of incorrect warnings.
3. Identify assaults as soon as possible.

## 3.3 TYPES OF INTRUSION DETECTION SYSTEMS:

For IDSs, there really are three primary sorts of parameters for their grouping.

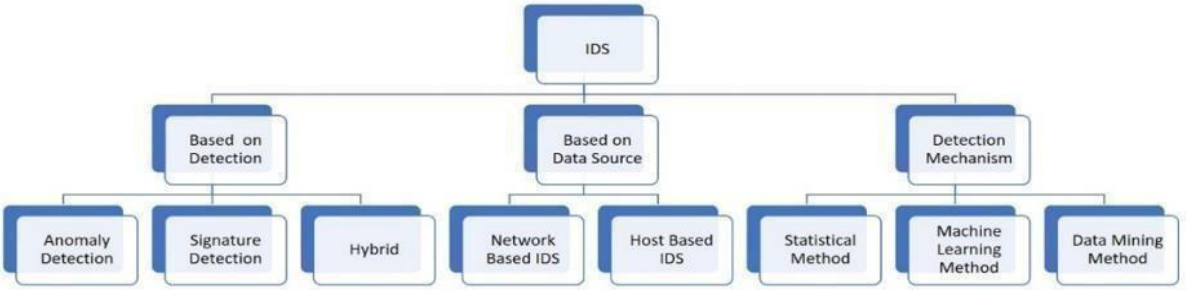



**Figure 3.1 Taxonomy of Intrusion Detection Systems**

Misuse-based, anomaly-based, and hybrid-based detections are at hand. (Jabraeil Jamali et al., 2020). Signature/misuse-based screening is used to detect recognized threats. Anomaly-based identification monitors normal channel and distinguishes between unusual channel and object activity. Lastly, by merging both misuse-based and anomaly-based identification strategies, the hybrid-based identification methodology increases recognition rate. Hackers may be able to manipulate flaws

In such traditional protection measures, jeopardizing the person's safety from undisclosed and upcoming dangers. The volume of information in cloud computing environment is enormous. Lawbreakers seek to get unwanted entry. Knowing the trends as well as practices of invasions and threats is critical. As a consequence, machine-learning approaches are essential for identifying and forecasting possible invasions and attack.

### 3.3.1 Network Intrusion Detection Systems (NIDS):

Now at system level, (NIDS) watch activity among all objects arriving and departing the system. The NIDS examines communications to detect patterns and suspected activity, and then sends an alarm. A few of the advantages of NIDS are as follows: NIDS may be readily installed inside a current system with minimal outage. Intruders could just go undiscovered and are frequently immune to straightforward strikes. To mention a few disadvantages, systems not able to handle large volumes of communication and thus cannot examine cryptographic data or damaged bits.

### 3.3.2 Host-based Intrusion Detection Systems (HIDS):

Unlike NIDS, which monitors the whole system, HIDS examines item information and searches for harmful activity on a specific host. HIDS will capture photos and notify you if they change dangerously across period. Auditing records deliver data that may be employed to trace changes in equipment and software applications. Some notable disadvantages are as follows: These too are susceptible to something like a targeted assault upon that running system of the victim. It can consume a significant amount of storage memory, imposing a burden on the host's capacity. HIDS can detect intrinsic alterations (for example, a virus installed accidentally because of a worker and trying to spread inside one’s machine), whereas NIDS can identify fraudulent data packet once those that enter one’s system or potentially malicious occurrence on one’s system, like as flooding attacks or protocol- specific attacks.

### 3.3.3 Industrial Control Systems (ICS):

IoT gadgets range in size between simple intelligent devices to massive commercial bots. These devices are used in ICS situations. ICS is a catchall word for the frameworks of Process Control System (PCS), Distributed Control System (DCS), and SCADA (McLaughlin et al., 2016).

petroleum - based operations, dams, biochemical companies, and so on. The overall design of an ICS is seen in Figure 3.2.

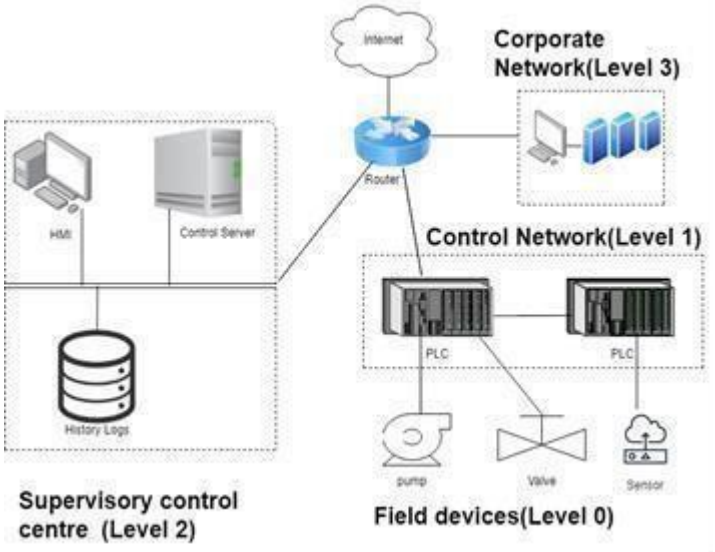



**Figure 3.2. General architecture of an ICS.**

The ICS design is arranged in tiers, as shown in Figure 3.2:

Level 0: This tier contains detectors and actuators used in the production method.

Level 1: This tier contains the Programmable logic controller (PLC) and the Distributed control system (DCS). They influence production methodologies and interact further to level zero.

Level 2: The control system watches, analyzes, also controls actual activities while they are in operation. This tier includes documentation, resource organization, output, work schedules, and networking.

Level 3: Corporate logistical and business applications are included.

## 3.4 Existing System:

While I don't have access to real-time data or specific proprietary systems, I can provide an overview of an existing CNN-based threat detection system that might be representative of current technology.

**Description**:

Secure Vision is a comprehensive threat detection system designed for security surveillance in high-risk environments such as airports, train stations, or government buildings. It employs a CNN-based approach to identify potential threats in real-time from surveillance camera feeds.

**Key Components:**

1. Data Acquisition Module: Secure Vision integrates with existing surveillance camera networks to capture real-time video feeds from multiple vantage points within the monitored area.
2. Preprocessing Pipeline: Raw video data undergoes preprocessing steps such as frame extraction, resizing, and normalization to ensure compatibility with the CNN model.
3. CNN Architecture: The core of Secure Vision is a custom-designed CNN architecture optimized for threat detection tasks. The CNN architecture consists of multiple convolutional layers followed by pooling layers and fully connected layers. The model is trained using large datasets of labeled threat and non-threat images.
4. Training Infrastructure: Secure Vision utilizes high-performance computing resources for training the CNN model. Training data includes a diverse range of threat scenarios, including suspicious packages, unauthorized access, and aggressive behavior.
5. Real-Time Inference Engine: Once trained, the CNN model is deployed as an inference engine capable of analyzing video streams in real-time. The model scans each frame for potential threats, identifying suspicious objects or behaviors based on learned patterns.
6. Alerting and Response Mechanism: When a potential threat is detected, Secure Vision triggers immediate alerts to security personnel, highlighting the location and nature of the threat. The system may also integrate with access control systems or automated response mechanisms to mitigate the threat.
7. Performance Monitoring and Feedback Loop: Secure Vision continuously monitors its performance, collecting feedback from security personnel and adjusting model parameters as necessary to improve accuracy and reduce false positives.

**Benefits**:

Enhanced Security: By automating threat detection, Secure Vision strengthens security measures and reduces the risk of human error or oversight.

Real-Time Response: The system enables rapid response to potential threats, minimizing the impact of security incidents.

Scalability: Secure Vision can scale to accommodate large surveillance networks spanning multiple locations, providing comprehensive threat detection coverage.

**Limitations:**

False Positives: Like any automated system, Secure Vision may generate false positive alerts, requiring manual verification by security personnel.

Resource Intensive: Training and running the CNN model requires significant computational resources, potentially limiting scalability in resource-constrained environments.

Privacy Concerns: The deployment of surveillance technology raises privacy concerns, necessitating careful implementation and adherence to regulations and guidelines.

Overall, Secure Vision exemplifies the capabilities and challenges of CNN-based threat detection systems in real-world security application.

## 3.5 IoT COMMUNICATIONS PROTOCOLS:

Several IoT information transport mechanisms are used in SCADA systems. However, the majority of such mechanisms were created without considering cyberspace dangers or safety ways to combat these. SCADA was designed around the Modbus communication protocol, which becomes today utmost extensively utilized standardized way for these systems. Building Automation and Control Network (BACnet), Distributed Network Protocol version 3 (DNP3), and Message Queuing Telemetry Transport (MQTT) all recent popular developing techniques. All four procedures are freely available. Modbus was created in 1979 as a provider of proprietary SCADA system (“The Modbus Organization”, n.d.) The basic systems BACnet (“BACnet Website”, n.d.) and DNP3 (“Overview of DNP3 Protocol”, n.d.) were first established in 1995 and 1993, respectively. MQTT was initially utilized in 1999. (“MQTT - The Standard for IoT Messaging”, n.d.). Designed specifically for constrained IoT devices operating in low-power, low-bandwidth networks, CoAP offers a RESTful communication paradigm over UDP (User Datagram Protocol). CoAP's lightweight header and support for multicast communication make it suitable for sensor networks, smart grids, and applications where energy efficiency and scalability are paramount. With features like resource discovery, caching, and observe, CoAP enables efficient data exchange and device management in dynamic IoT environments. Although not tailored explicitly for IoT, HTTP remains a prevalent communication protocol in IoT deployments, particularly for web-based applications and integration with cloud services. Leveraging the request-response model, HTTP provides simplicity and compatibility with existing web infrastructure, allowing IoT devices to interact with web servers, APIs, and cloud platforms seamlessly. However, HTTP's stateful nature and high overhead may pose challenges in resource-constrained environments, necessitating optimizations like HTTP/2 and HTTP/3 for improved efficiency and performance.

## 3.6 IDS USED ON IoMT SYSTEMS:

This research paper (Zachos et al., 2021) developed a novel AIDS (Anomaly-based Intrusion Detection Systems) that is adapted to the limits of IoMT systems in order that provide an optimal yet reliable safety approach to these systems. The identification procedures about the suggested AIDSs would be executed mostly by identification motor trying to run on the IoMT side system interface but also depend on ML methodologies to identify irregularity within the gathered information but also recognize malevolent instances with in IoMT system while accounting for computational complexity. Six common machine- learning methods were tested for irregularity identification in the planned AIDS, and thus the analysis showed which ones were really the best fits. This research (Parbat et al., 2022) suggests an IoT-based health monitoring system in which important systems are constantly observed, information is relayed to server clinicians, and clients are informed of possible threats. An IoT device which links several devices to the microprocessor and transmits the gathered information to a remote cloud service to execute a customized stochastic gradient descent (SGD) method that incorporates deep learning has been created. If somehow the physician senses a medical issue, the gadget could provide an alert after the scan is over. This suggested technique functions as an Iot network medical monitoring. This paper (Iwendi et al., 2021) completed an intruder identification evaluation for FAR diagnosis using ML function assortment, which has been integrated inside a biological method with both the conjunction of random forest in a Security of Things prototype, and also introduced a safe information-sharing framework suitable for health care. On the NSL-KDD database, the researchers utilized a range of ML techniques in their trials. When compared to earlier studies, the trial findings revealed that this technique performed well in terms of DR, FAR, and accuracy. This study (Akshay Kumaar et al., 2022) presented "Immune Net," a hybrid framework that employs Deep Learning to detect cutting-edge infiltration assaults and protect health information.

## 3.7 IMPLEMENTATION ENVIRONMENT

Implementing an intrusion detection system (IDS) using Convolutional Neural Networks (CNNs) for detecting anomalies in network traffic involves several steps. Below is a high-level overview of the implementation environment for such a system:

**1. Python and Deep Learning Libraries:** Python is commonly used for implementing machine learning and deep learning models. Libraries such as TensorFlow or Py-Torch are often chosen for building CNN-based models due to their flexibility and efficiency.

**2. Data Collection:** You need a dataset containing both normal and anomalous network traffic data for training the CNN. Datasets like NSL-KDD, UNSW-NB15, or CICIDS2017 are commonly used for this purpose. Ensure that the dataset is preprocessed appropriately, including normalization and feature extraction.

**3. Preprocessing:** Prepare the dataset for training. This involves steps such as data normalization, splitting the dataset into training, validation, and testing sets, and possibly data augmentation to increase the diversity of the training data.

**4. CNN Architecture:** Design the architecture of the CNN. Since CNNs are powerful in extracting spatial features, they can effectively learn patterns from network traffic data. The architecture typically consists of convolutional layers followed by pooling layers, fully connected layers, and output layers for classification.

**5. Training:** Train the CNN model using the prepared dataset. This involves feeding the training data through the network, adjusting the network's weights using backpropagation and optimization algorithms (e.g., Adam, RMS Prop), and monitoring the model's performance on the validation set to prevent overfitting.

**6. Evaluation:** Evaluate the trained model's performance using the testing dataset. Metrics such as accuracy, precision, recall, and F1-score are commonly used to assess the model's effectiveness in detecting intrusions while minimizing false positives.

**7. Deployment:** Once the model is trained and evaluated satisfactorily, it can be deployed in the production environment for real-time intrusion detection. This may involve integrating the model into existing network infrastructure, such as firewalls or network appliances, to analyze incoming traffic and flag potential intrusions.

**8. Continuous Monitoring and Updating:** Regularly monitor the performance of the deployed model in the production environment. Update the model periodically using new data and retraining techniques to adapt to evolving intrusion patterns and maintain effectiveness over time.

**9. Integration with IDS Frameworks:** Optionally, you can integrate your CNN-based IDS into existing IDS frameworks for better management and monitoring. Popular IDS frameworks include Snort, Suricata, and Bro/Zeek.

**10. Security Considerations:** Ensure that the implementation follows best practices for security, such as secure data handling, encryption, and access control, to prevent unauthorized access to sensitive network traffic data and the intrusion detection system itself.

## 3.8 PROPOSED METHODOLOGY:

The proposed methodology for intrusion detection systems tailored specifically for IoT environments. As the proliferation of IoT devices continues to reshape our interconnected world, ensuring the security and integrity of these devices and their networks becomes paramount. In this chapter, we present a comprehensive approach to detecting and mitigating intrusions within IoT ecosystems. The proposed methodology draws upon a synthesis of cutting-edge research in machine learning, data analytics, and cybersecurity. It aims to address the unique challenges posed by IoT environments, characterized by a vast array of interconnected devices with varying computational capabilities and communication protocols. Data collection and preprocessing form the foundational steps in the development of an effective intrusion detection system (IDS) for IoT environments. In this section, we delve into the intricacies of data collection and the preprocessing techniques required to extract actionable insights from the raw data generated by IoT devices.

## Data Collection:

The heterogeneous nature of IoT ecosystems presents a significant challenge in data collection. IoT devices generate vast amounts of heterogeneous data, including network traffic, sensor readings, system logs, and device status updates. Collecting this data efficiently and effectively is essential for building robust intrusion detection models.

### Sources of Data:

- Network Traffic: IoT devices communicate with each other and external servers via network protocols such as TCP/IP, MQTT, CoAP, and others. Capturing network traffic provides insights into communication patterns, anomalies, and potential intrusions.
- Sensor Readings: Many IoT devices are equipped with sensors to collect data from the physical environment. These sensor readings, including temperature, humidity, motion, and sound, can reveal anomalies or malicious activities.
- System Logs: Device logs contain valuable information about device activities, error messages, and system events. Analyzing logs can help identify suspicious activities or unauthorized access attempts.

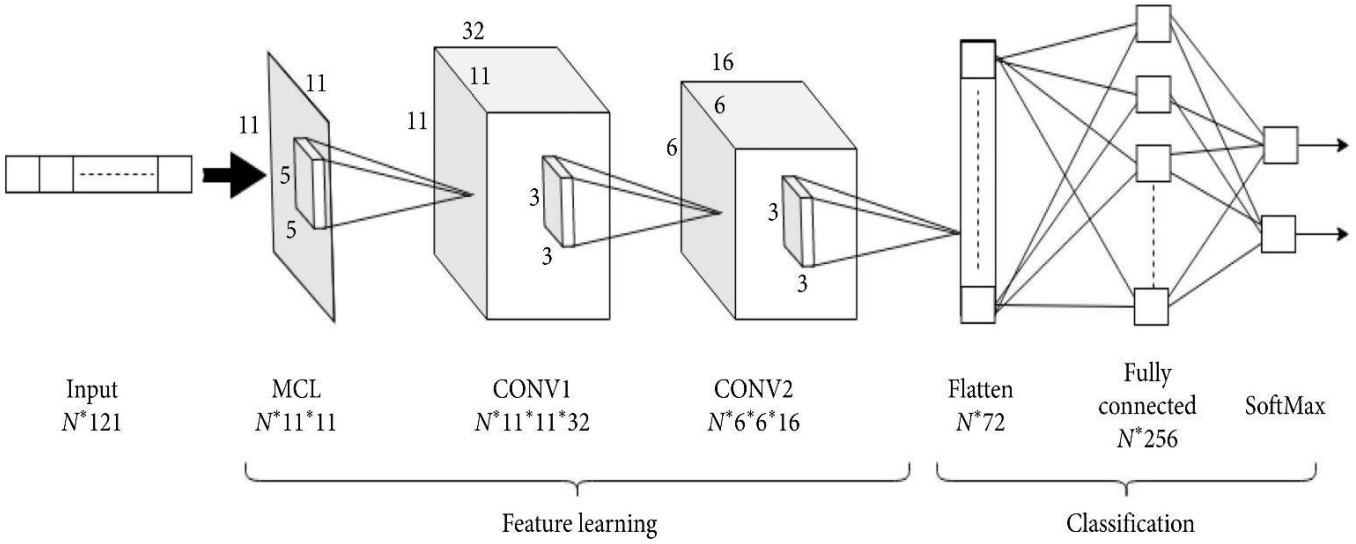



**Figure 3.3 Proposed Methodology: IoT intrusion Detection System**

## 3.9 DATA COLLECTION MECHANISMS:

Passive Monitoring: Passive monitoring involves capturing network traffic and sensor data without interfering with the normal operation of IoT devices. Active Probing: Active probing involves sending test packets or queries to IoT devices to elicit responses. While more intrusive, active probing can provide additional insights into device behavior and vulnerabilities.

## Data Preprocessing:

Once the raw data is collected, it undergoes preprocessing to transform it into a format suitable for analysis by intrusion detection models. Data Cleaning: Raw data often contains noise, missing values, and inconsistencies that can affect the performance of intrusion detection models. Data cleaning techniques, such as imputation, filtering, and outlier removal, are applied to ensure the quality and integrity of the data.

## Feature Extraction:

Feature extraction involves selecting and transforming raw data into a set of meaningful features that capture the underlying patterns and characteristics of IoT device behavior. Feature extraction techniques may include statistical analysis, time-series analysis, frequency domain analysis, and domain-specific knowledge.

## Feature Scaling and Normalization:

Features are scaled and normalized to ensure uniformity and comparability across different features and data instances. Common scaling techniques include min-max scaling, standardization, and robust scaling.

## Dimensionality Reduction:

In high-dimensional data spaces, dimensionality reduction techniques such as principal component analysis (PCA) or feature selection methods may be applied to reduce the computational complexity and improve the efficiency of intrusion detection models.

## Data Labeling:

Data instances are labeled based on their ground truth or known class labels, indicating whether they represent normal behavior or intrusions. Labeling the data enables supervised learning algorithms to learn and distinguish between different classes of events. By meticulously collecting and preprocessing data, intrusion detection systems can leverage the rich information embedded within IoT environments to accurately detect and mitigate intrusions. Effective data collection and preprocessing lay the groundwork for building robust and reliable intrusion detection models capable of safeguarding IoT ecosystems against emerging cyber threats.

# CHAPTER 4
# SYSTEMDESIGN

## 4.1 USE CASE DIAGRAM:

Use case diagrams are considered for high level requirement analysis of a system. When the requirements of a system are analyzed, the functionalities are captured in use cases. So, it can say that use cases are system functionalities written in an organized manner.

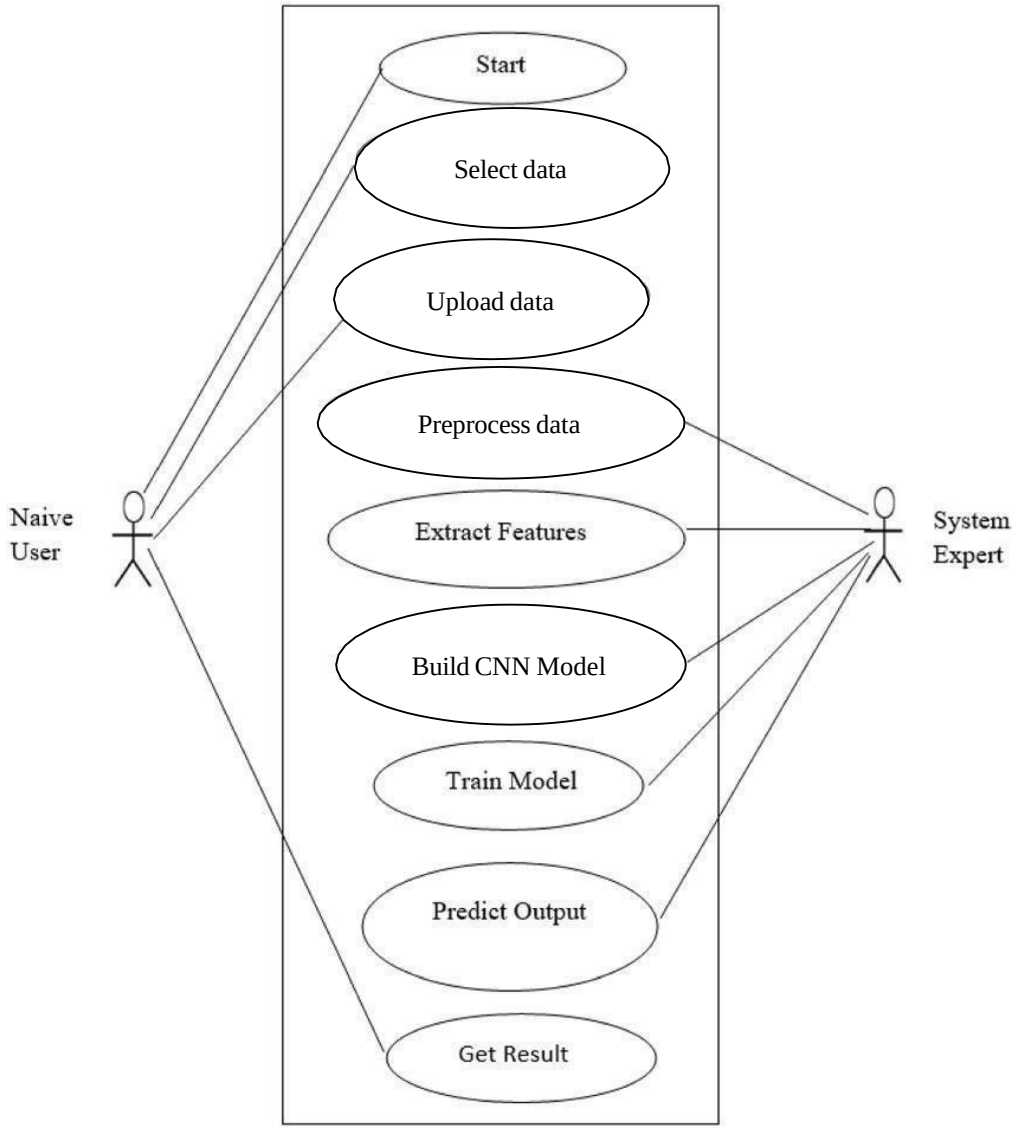



**Figure: 4.1 Use case Diagram for CNN**

## 4.2 ACTIVITY DIAGRAM:

A graphical representation of an executed set of procedural system activities and considered a state chart diagram variation. Activity diagrams describe parallel and conditional activities, use cases and system functions at a detailed level.

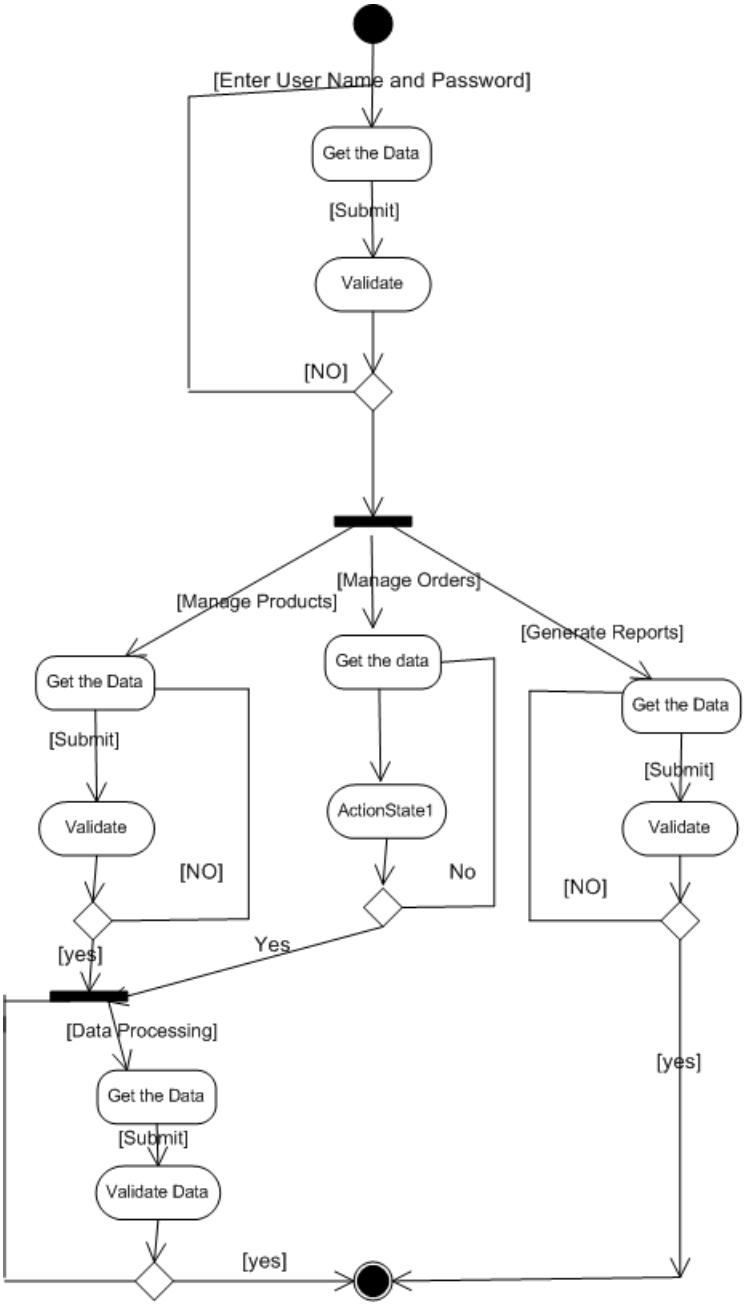



**Figure: 4.2 Activity Diagram for CNN**

## 4.3 CLASS DIAGRAM:

Class diagram is basically a graphical representation of the static view of the system and represents different aspects of the application. The name of the class diagram should be meaningful to describe the aspect of the system. Each element and their relationships should be identified in advance.

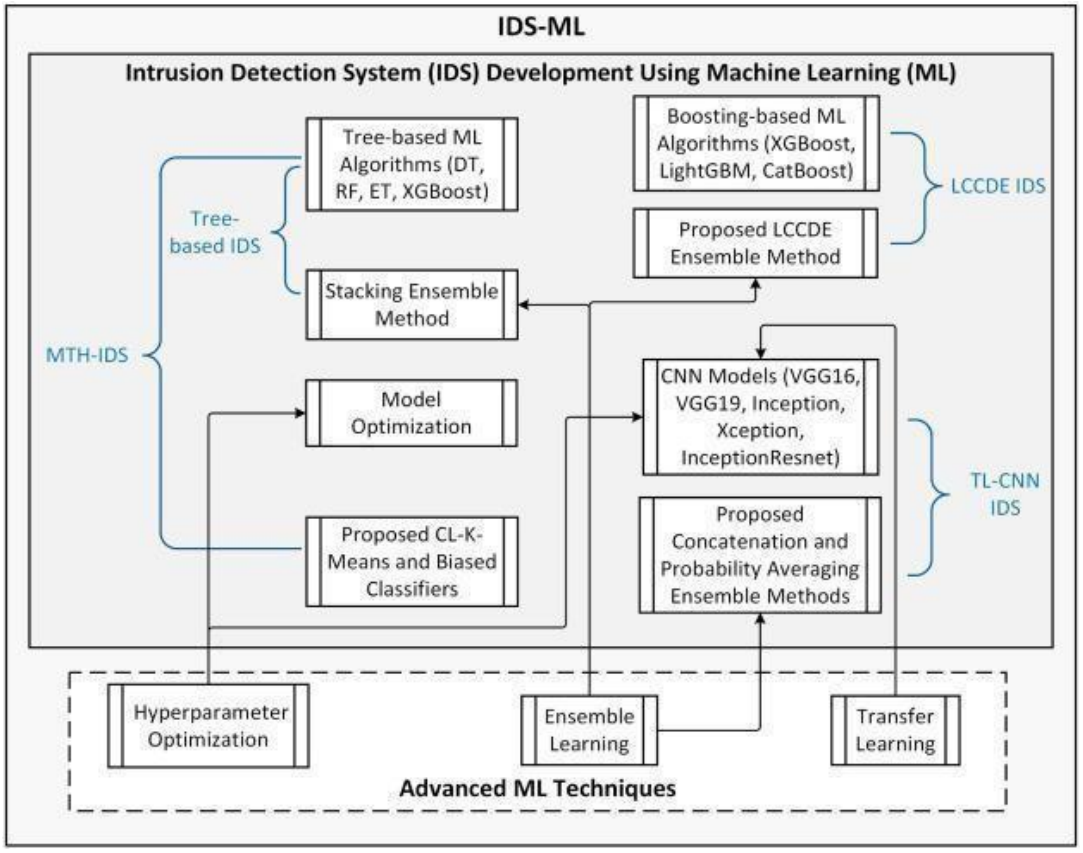



**Figure: 4.3 Class Diagram for CNN**

## 4.4 SEQUENCE DIAGRAM:

A sequence diagram is a type of interaction diagram in UML (Unified Modeling Language) that illustrates how objects in a system interact with each other over time to achieve a particular task or scenario. It focuses on the sequence of messages exchanged between objects and the lifelines of those objects.

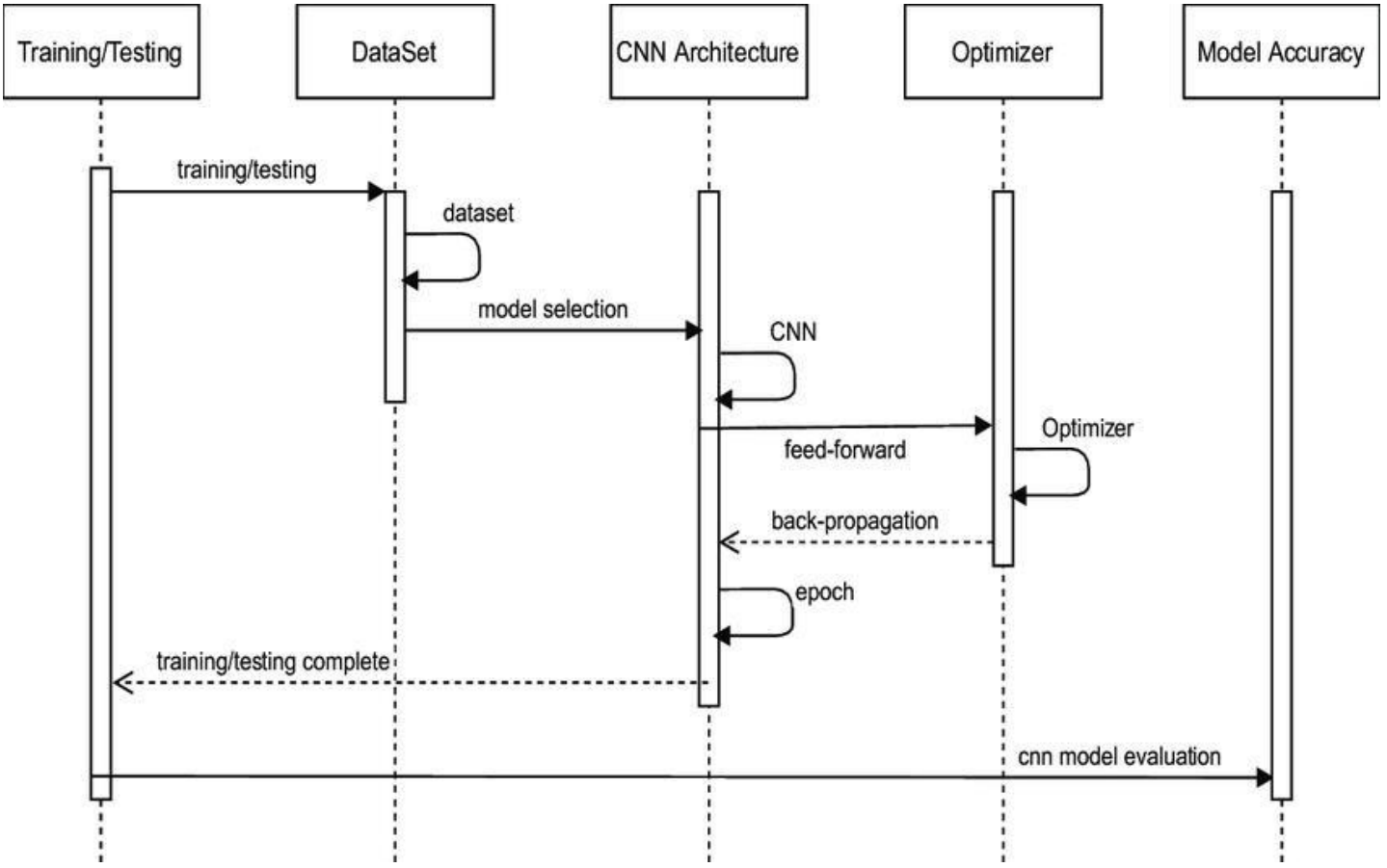



**Figure: 4.4 Sequence Diagram for CNN**

# CHAPTER 5

# SYSTEM IMPLEMENTATION

## 5.1 MODEL SELECTION:

In the realm of IoT intrusion detection, model selection is pivotal for crafting robust and adaptive defense mechanisms against evolving threats. Among the plethora of available techniques, ensemble learning methods like decision trees, AdaBoost, Gradient Boosting, and Bagging stand out as formidable contenders. Decision Trees With their intuitive structure and ability to handle heterogeneous data, decision trees excel in discerning intricate patterns within IoT data streams. Their hierarchical nature allows for easy interpretation, making them particularly suitable for transparent intrusion detection systems. AdaBoost: AdaBoost, or Adaptive Boosting, harnesses the power of weak learners to construct a strong classifier. By iteratively focusing on misclassified instances, AdaBoost enhances model performance, effectively capturing subtle anomalies amidst the noise inherent in IoT environments. Gradient Boosting: Gradient Boosting iteratively constructs an ensemble of decision trees, with each subsequent tree refining the predictions of its predecessors. This iterative refinement process allows Gradient Boosting to adaptively learn complex relationships within IoT data, enhancing its predictive capabilities. Bagging: Bagging, short for Bootstrap Aggregating, leverages the wisdom of crowds by training multiple models on bootstrapped subsets of the data and aggregating their predictions. This ensemble approach reduces overfitting and variance, resulting in more robust intrusion detection models. By judiciously selecting and combining these diverse techniques, intrusion detection systems can effectively navigate the intricate landscape of IoT data, fortifying networks against an array of cyber threats with resilience and efficacy. Model training lies at the heart of building effective intrusion detection systems (IDS) for IoT environments.

## Data Preparation:

Before training the model, the labeled dataset is divided into training, validation, and test sets. The training set is used to teach the model, the validation set helps tune hyperparameters, and the test set evaluates the model's performance.

## Algorithm Selection:

The appropriate machine learning algorithm is selected based on the nature of the problem, dataset characteristics, and desired outcomes. Common algorithms for intrusion detection include decision trees, support vector machines (SVM), neural networks, and ensemble methods like Random Forest and Gradient Boosting.

## Hyperparameter Tuning:

Hyperparameters are parameters that govern the learning process of the model, such as learning rate, regularization strength, and tree depth. Hyperparameter tuning techniques like grid search or random search are employed to find the optimal configuration that maximizes model performance.

## Training Process:

During training, the model iteratively adjusts its parameters to minimize the difference between predicted and actual labels. This is typically done using optimization algorithms like stochastic gradient descent (SGD) or Adam, which update the model parameters based on gradients computed from the training data.

## Evaluation and Validation:

The model's performance is evaluated using the validation set, which provides an unbiased estimate of its generalization ability. Metrics such as accuracy, precision, recall, F1 score, and area under the ROC curve (AUC-ROC) are computed to assess the model's effectiveness in detecting intrusions while minimizing false alarms.

## Iterative Refinement:

The training process may involve multiple iterations of tweaking hyper parameters, experimenting with different algorithms, or incorporating new features to improve model performance. This iterative refinement is essential for building robust and accurate intrusion detection systems.

## Model Deployment:

Once the model has been trained and validated, it is deployed into production environments to monitor and analyze real-time data streams from IoT devices. Continuous monitoring and feedback mechanisms ensure that the model remains effective in detecting emerging threats and adapting to changes in the IoT ecosystem. By following a systematic approach to model training, intrusion detection systems can effectively learn from data and proactively safeguard IoT networks against a wide range of cyber threats.

## Environment Preparation:

Ensure that the deployment environment meets the necessary requirements, including hardware resources, software dependencies, and network connectivity. This may involve setting up dedicated servers, cloud infrastructure, or edge devices capable of running the intrusion detection model

## Integration with Data Sources:

Integrate the intrusion detection model with data sources from IoT devices, such as network traffic logs, sensor readings, and system logs. This may involve configuring data ingestion pipelines or APIs to stream real-time data to the deployed model.

## Real-Time Monitoring:

Implement mechanisms for real-time monitoring of incoming data streams from IoT devices. The deployed model should continuously analyze the data and trigger alerts or notifications when suspicious activity or intrusions are detected.

## Alerting and Response Mechanisms:

Define alerting thresholds and response mechanisms to handle detected intrusions. This may include notifying security personnel, blocking suspicious traffic, quarantining compromised devices, or triggering automated remediation actions

## Performance Monitoring and Maintenance:

Monitor the performance of the deployed model over time to ensure its effectiveness in detecting and mitigating intrusions. This may involve monitoring key performance indicators (KPIs), such as detection rates, false positive rates, and response times, and adjusting the model or alerting thresholds as needed.

## Scalability and Resilience:

Design the deployment architecture to be scalable and resilient, capable of handling increasing data volumes and adapting to changing network conditions. This may involve deploying redundant instances of the model across multiple servers or cloud regions to ensure high availability and fault tolerance.

## Continuous Improvement:

Implement mechanisms for continuous improvement of the deployed model based on feedback from real-world usage. This may involve retraining the model with updated data, fine-tuning hyperparameters, or incorporating new features to enhance detection capabilities.

## Compliance and Governance:

Ensure compliance with regulatory requirements and data privacy regulations governing the deployment of intrusion detection systems. This may involve implementing security controls, access controls, and audit trails to protect sensitive data and ensure accountability. By following these best practices for model deployment, organizations can effectively leverage intrusion detection models to enhance the security posture of their IoT environments and mitigate the risks associated with cyber threats and intrusions. The Proposed Convolutional Neural Network (CNN) architecture The Convolutional Neural Network (CNN) architecture used in this research is designed for classification tasks, specifically for intrusion detection in IoT environments. Here's a breakdown of the CNN architecture.

## Input Layer:

The input layer of the CNN takes as input the features extracted from the IoT data. In this case, the features are normalized using z-score normalization.

## Convolutional Layers:

The CNN consists of multiple convolutional layers, which apply a set of learnable filters to the input data. Each filter extracts features from the input through convolution operations. The number of filters and their sizes are specified in the architecture.

## Activation Layers (RELU):

Rectified Linear Unit (RELU) activation functions are applied after each convolutional layer to introduce non-linearity into the network. RELU activations help the model learn complex patterns and relationships in the data.

## Batch Normalization Layers:

Batch normalization layers normalize the activations of each layer, improving the ability and convergence of the training process.

## Fully Connected Layers:

Following the convolutional layers, the network includes fully connected layers, also known as dense layers. These layers connect every neuron in one layer to every neuron in the next layer, allowing the network to learn high-level representations of the input data.

## SoftMax Layer:

The final layer of the CNN is a SoftMax layer, which normalizes the output of the network into a probability distribution over the different classes. It assigns a probability to each class, indicating the likelihood of the input belonging to that class.

## Classification Layer:

The classification layer computes the loss and error of the network's predictions during training and updates the model's parameters to minimize these values. It uses categorical cross-entropy loss for multi- class classification tasks like intrusion detection.

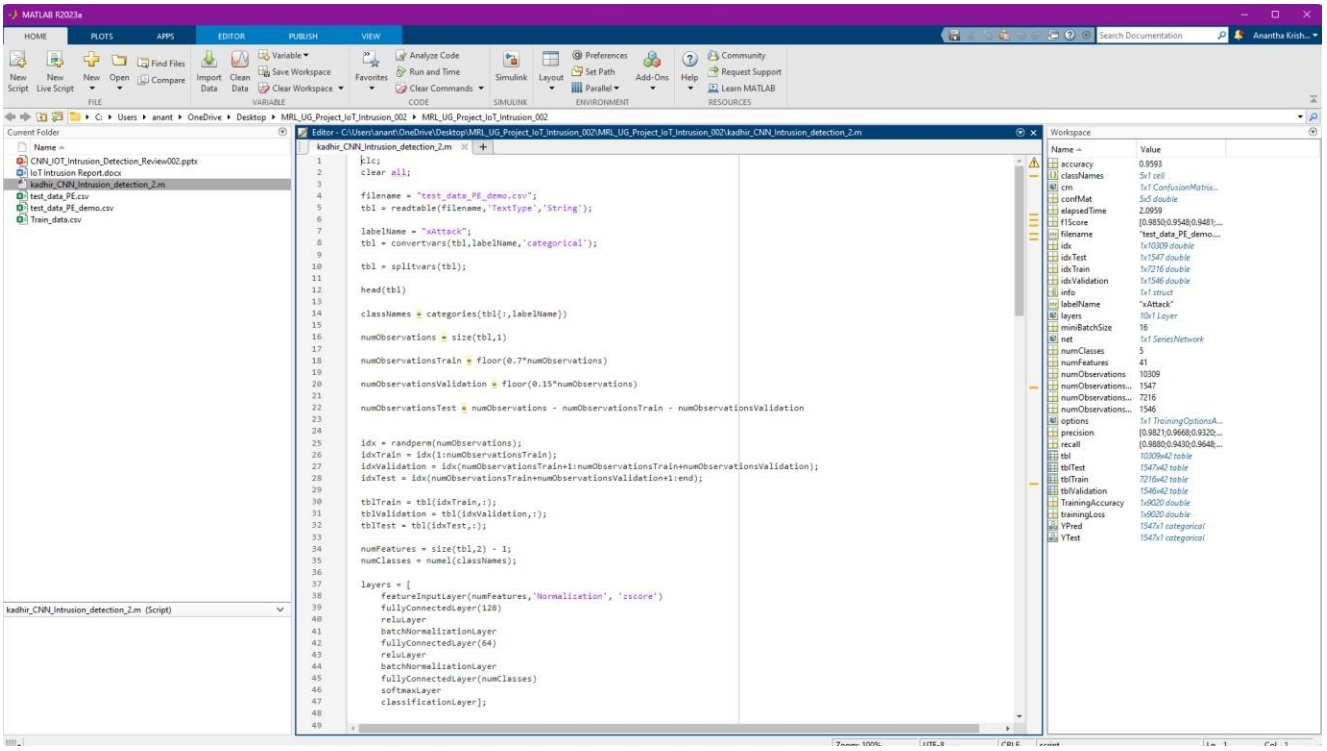



**Figure 5.1 Simulation started**

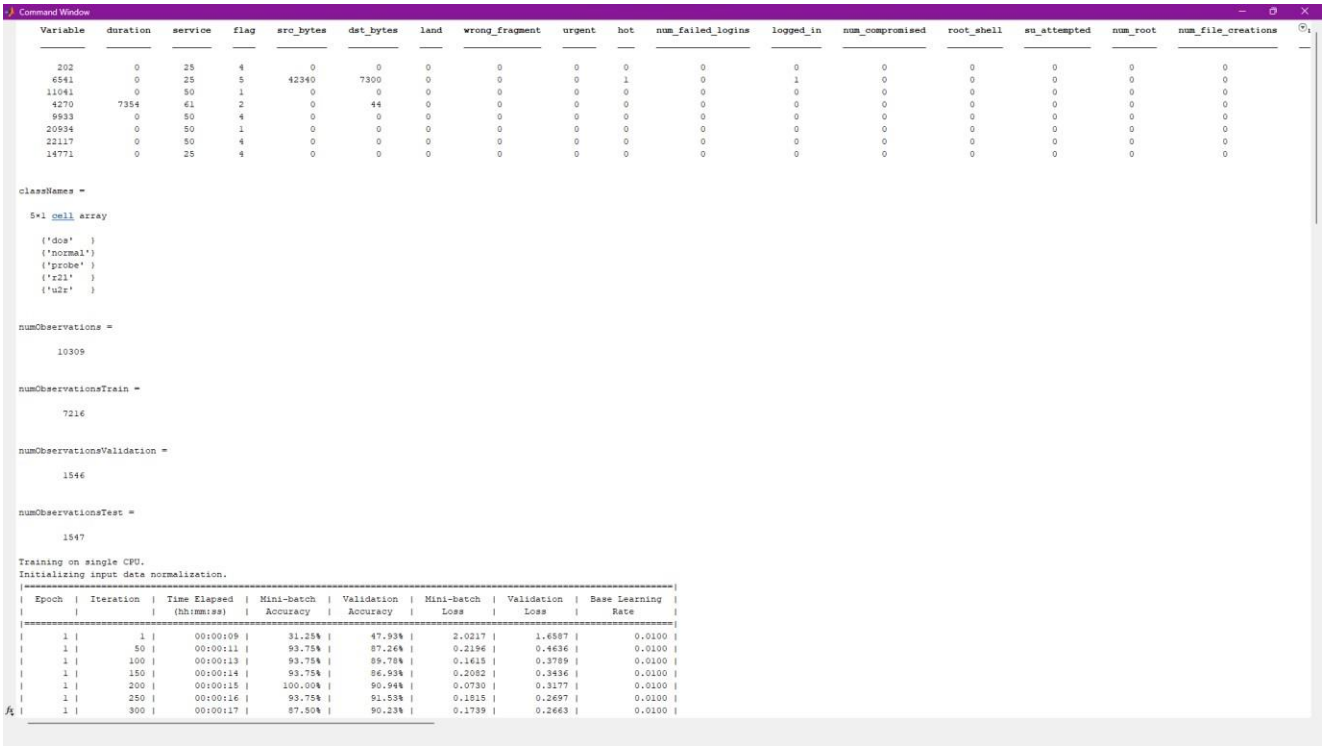



**Figure 5.2 Iteration started**

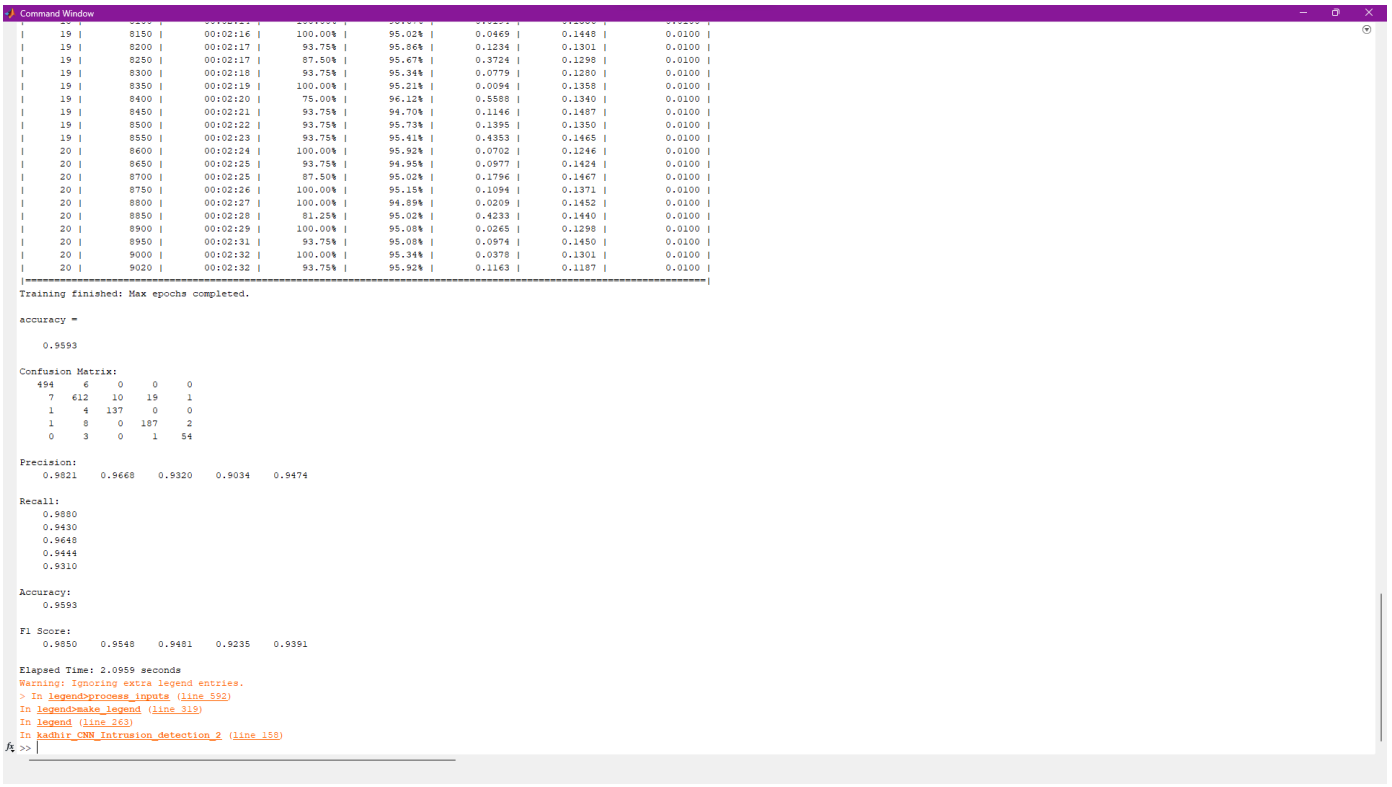



**Figure 5.3 Iteration Executed**

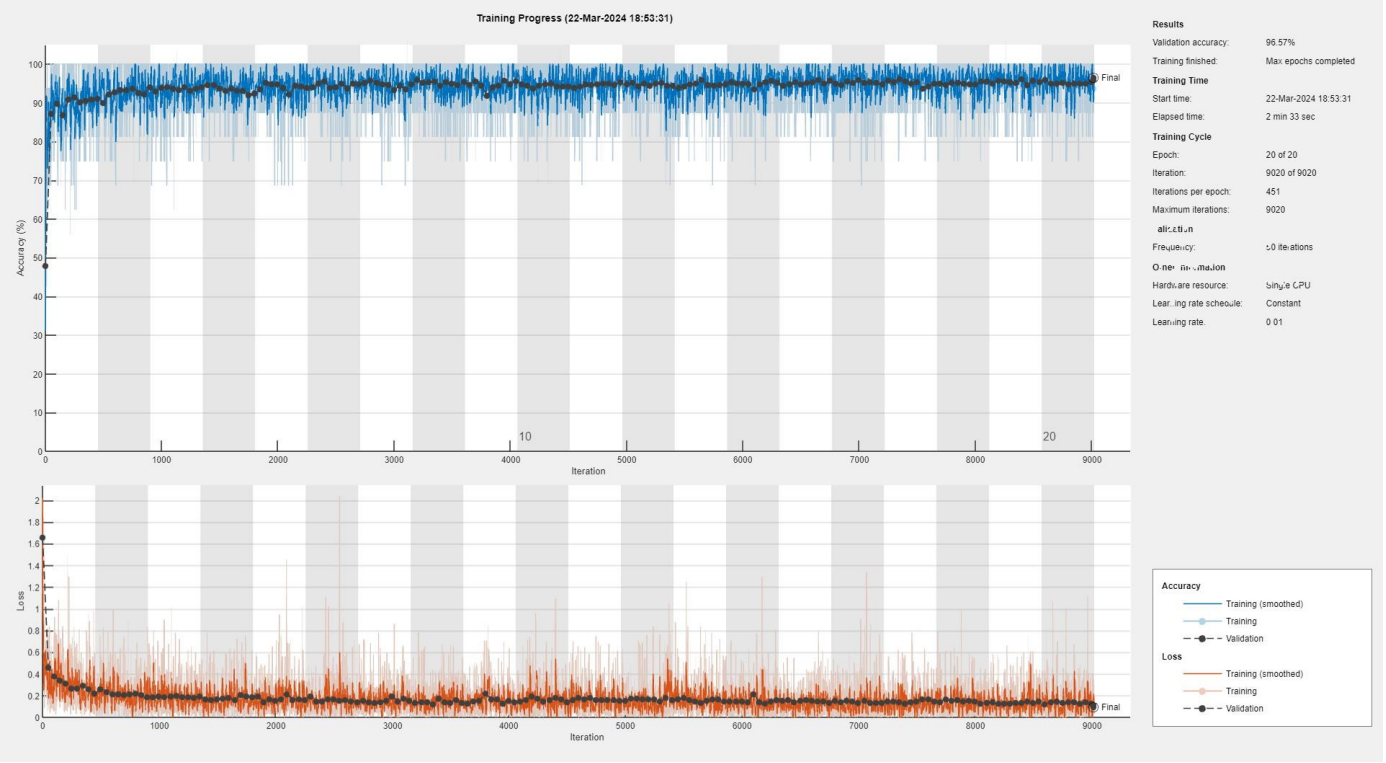
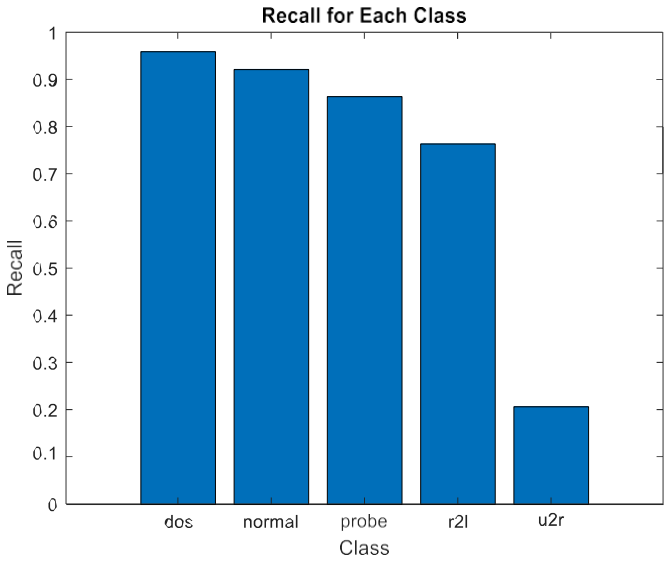



**Figure 5.4 Output of CNN Model**

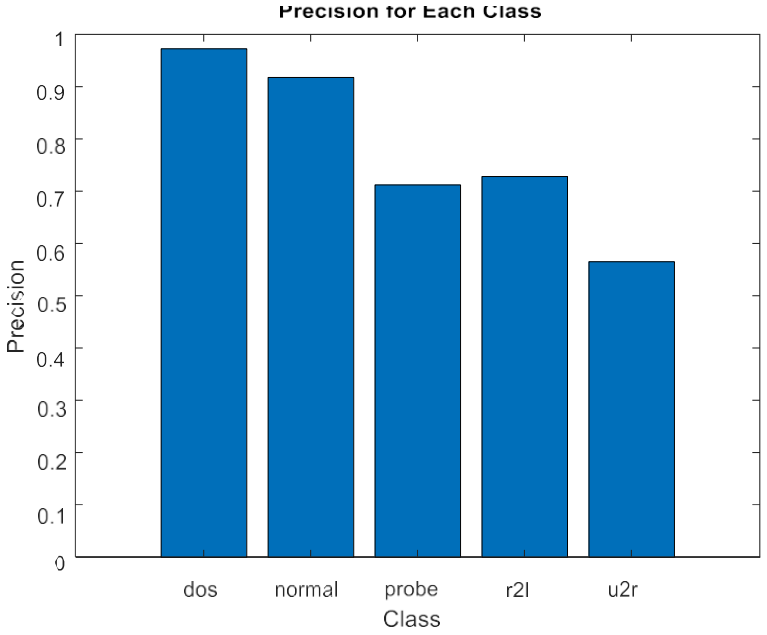
# CHAPTER 6
# RESULTS&DISCUSSION

## 6.1 Classifier Model

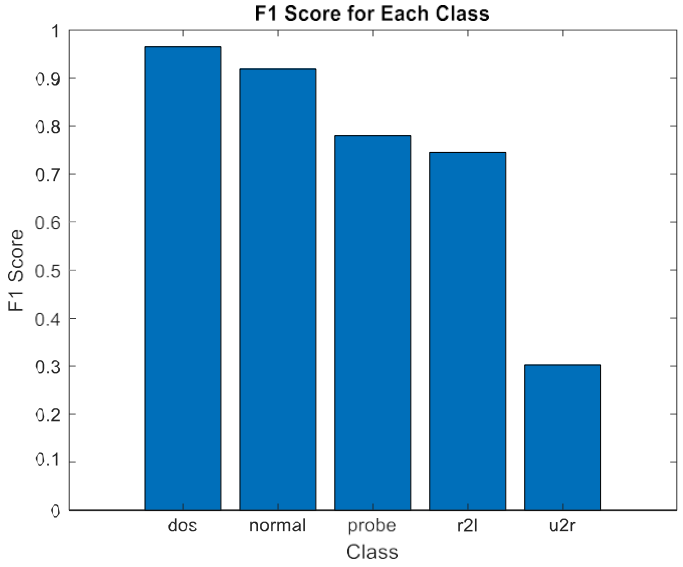
Chapter 6 delves into the culmination of our endeavors, presenting the results and delving into comprehensive discussions derived from the implemented intrusion detection system in IoT environments. Throughout this journey, we have meticulously crafted and fine-tuned our model, leveraging state-of-the-art techniques in machine learning and deep learning to safeguard IoT ecosystems against cyber threats. In this chapter, we embark on a detailed exploration of the outcomes obtained from our model deployment and evaluation. Through rigorous experimentation and analysis, we scrutinize the efficacy and performance of our intrusion detection system in classifying instances of normal behavior and various types of intrusions. We delve into the intricacies of the evaluation metrics, unraveling the model's accuracy, precision, recall, F1 score, and area under the ROC curve (AUC-ROC) to discern its strengths and limitations. Furthermore, this chapter serves as a platform for comprehensive discussions and interpretations of the results garnered.




**Recall for Each Intrusion Class**



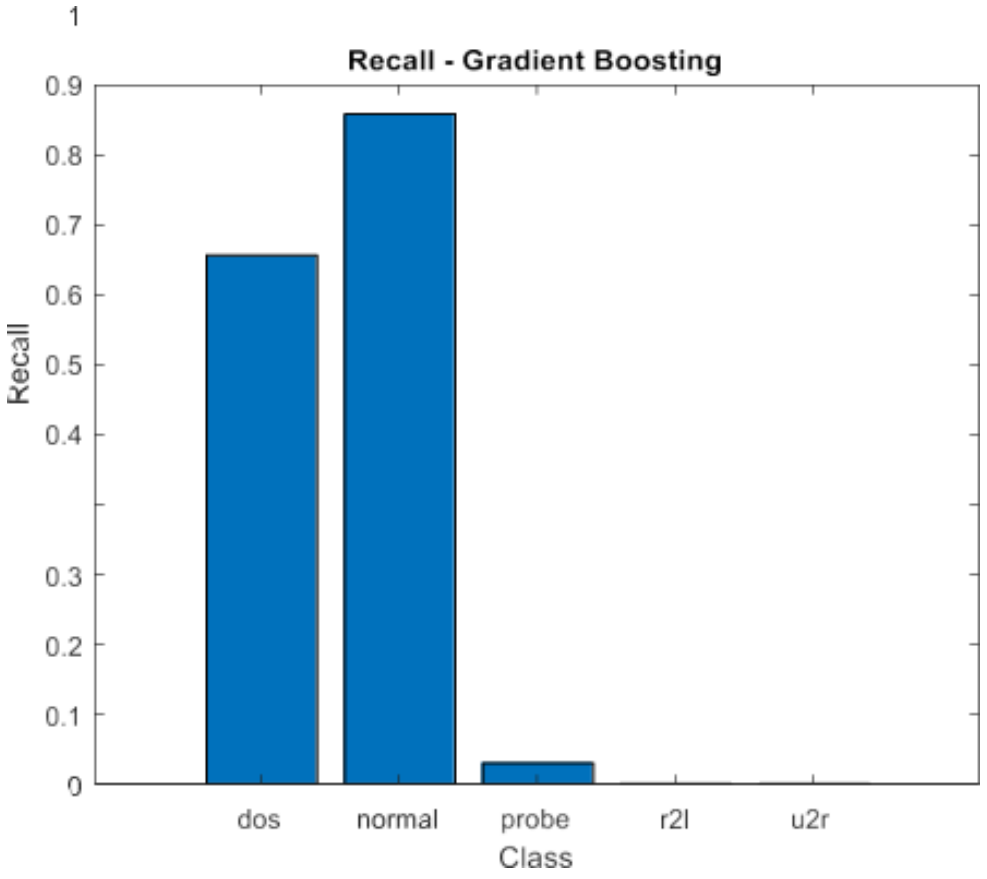

**Precision for Each Intrusion Class**




**F1 Score for Each Intrusion Class**

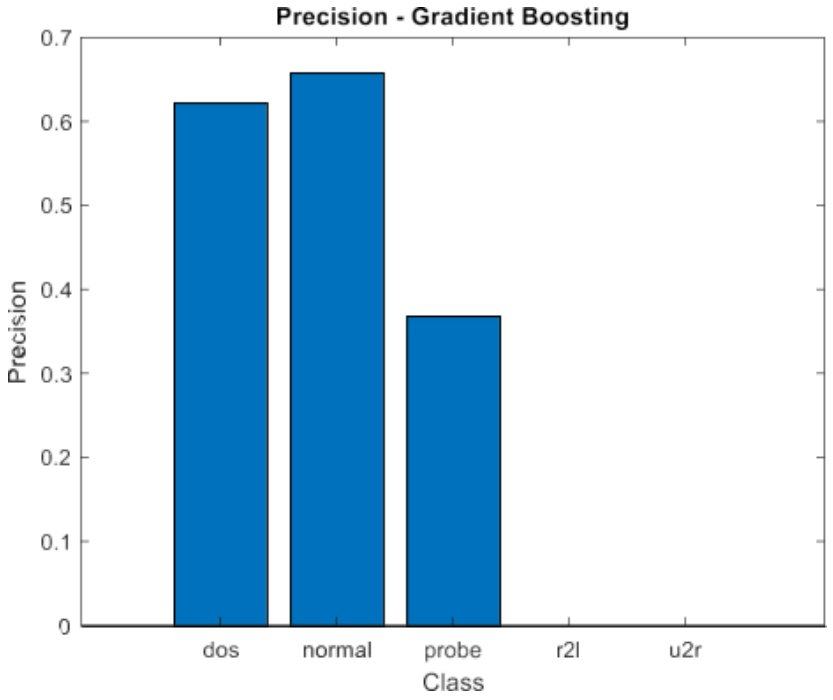
The decision trees classifier, the recall metric for each intrusion class in the dataset. The bar chart provides insights into the ability of the intrusion detection system to correctly identify instances of each class. Class 3 exhibits the highest recall at 0.8642, indicating a strong ability to detect instances of this intrusion type, while Class 5 demonstrates the lowest recall at 0.2071, suggesting challenges in accurately identifying instances of this class. The precision metric for each intrusion class in the dataset.

classified instances, while Class 5 exhibits the lowest precision at 0.5655, suggesting a higher rate of false positives. The F1 score metric for each intrusion class in the dataset. The bar chart provides an assessment of the balance between precision and recall for each class. Class 1 achieves the highest F1 score at 0.9663, indicating a strong balance between precision and recall, while Class 5 obtains the lowest F1 score at 0.3031, suggesting challenges in achieving a balance between precision and recall for this class.



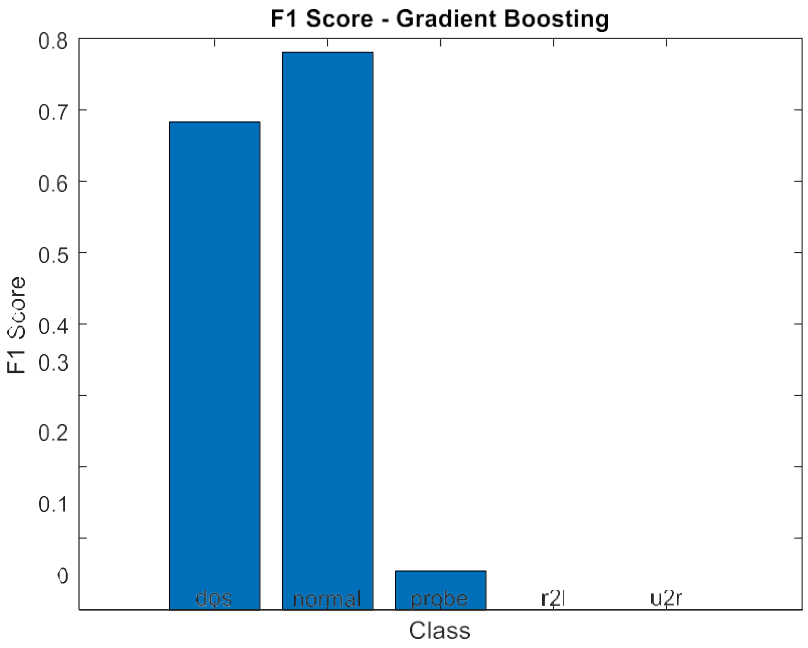

**Recall for Each Intrusion Class using Ada-Boost-Based Classifier Model**

The recall metric for each intrusion class in the dataset using an AdaBoost-based classifier model for IoT intrusion detection. The bar chart provides insights into the ability of the classifier to correctly identify instances of each class. Class 2 exhibits the highest recall at 0.9587, indicating a strong ability to detect instances of this intrusion type, while Classes 3, 4, and 5 demonstrate low recall rates, suggesting challenges in accurately identifying instances of these classes.




**Precision for Each Intrusion Class using Ada-Boost-Based Classifier Model**

The precision metric for each intrusion class in the dataset using an Ada-Boost-based classifier model. The bar chart offers a comprehensive view of the precision of the classifier across different classes. Class 2 demonstrates the highest precision at 0.6586, indicating a high proportion of correctly classified instances, while Classes 3, 4, and 5 exhibit lower precision rates, suggesting a higher

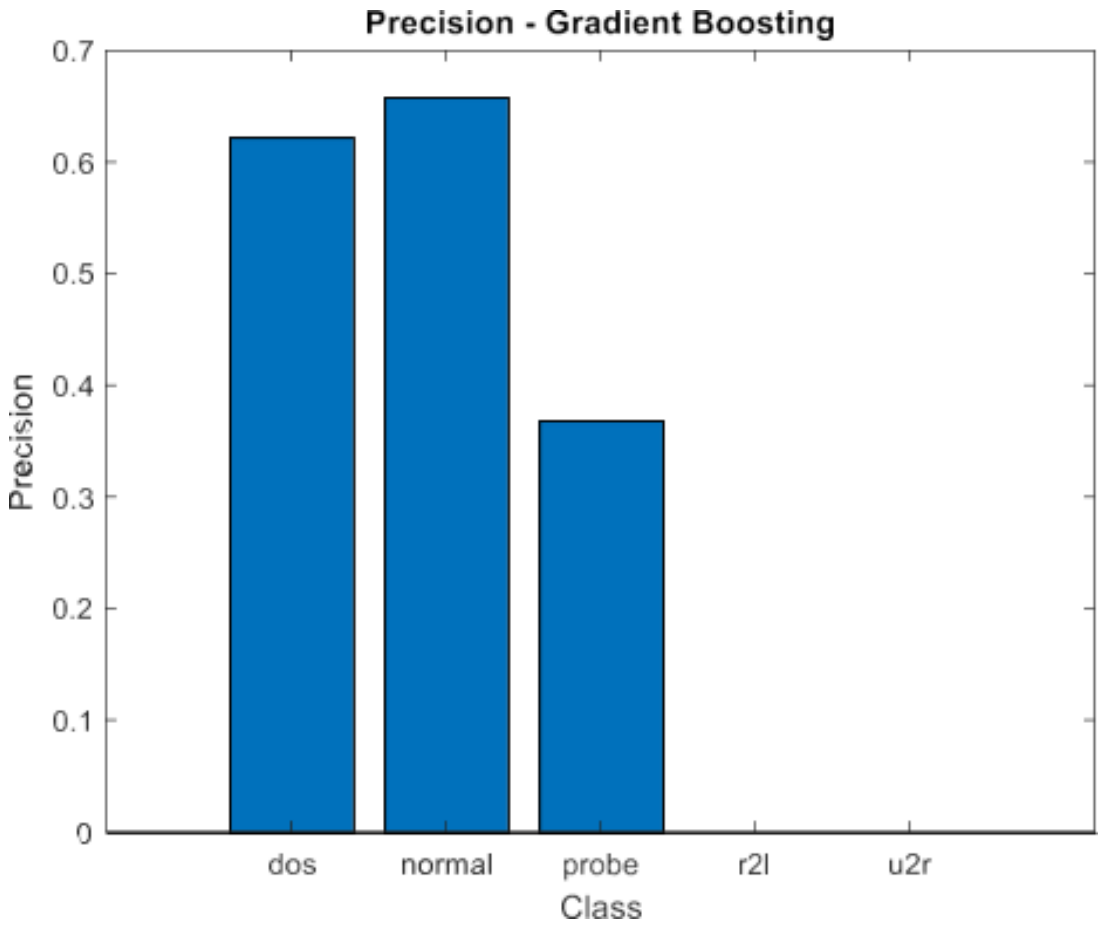



**F1 Score for Each Intrusion Class using AdaBoost-Based Classifier Model**

The F1 score metric for each intrusion class in the dataset using an AdaBoost-based classifier model. The bar chart provides an assessment of the balance between precision and recall for each class. Class 2 achieves the highest F1 score at 0.7808, indicating a strong balance between precision and recall, while Classes 3, 4, and 5 obtain lower F1 scores, suggesting challenges in achieving a balance between precision and recall for these classes.

## Bag Tree - Based Classifier Model for IoT Intrusion Detection

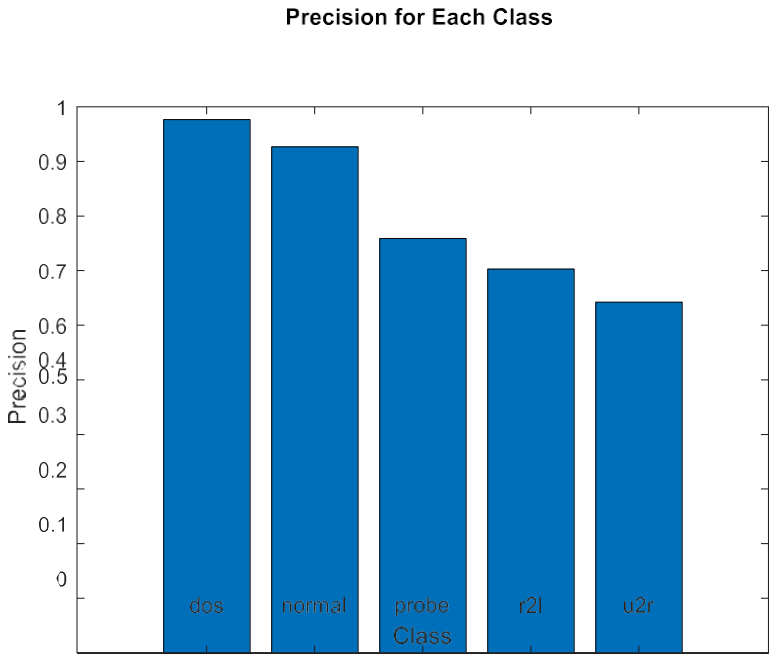
To correctly identify instances of each class. Class 4 exhibits the highest recall at 0.8332, indicating a strong ability to detect instances of this intrusion type, while Class 5 demonstrates the lowest recall at 0.1995, suggesting challenges in accurately identifying instances of this class.




**Recall for Each Intrusion Class using Bag Tree-Based Classifier Model**

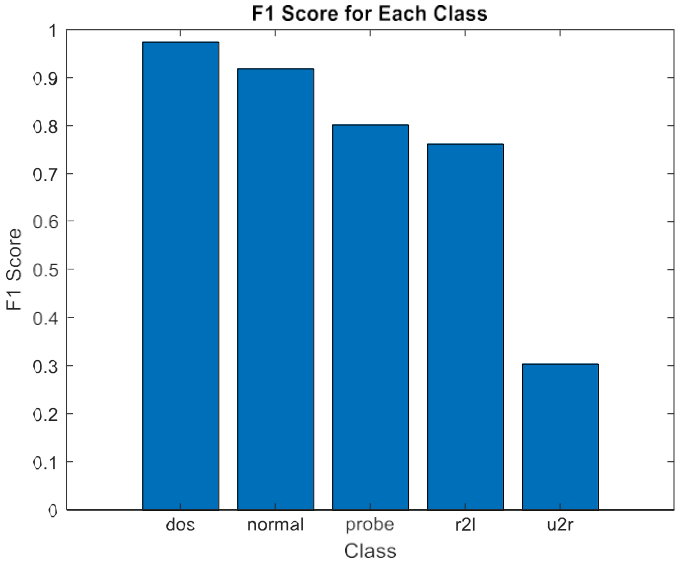
The recall metric for each intrusion class in the dataset using a Bag Tree-based classifier model for IoT intrusion detection. The bar chart provides insights into the ability of the classifier.

.




**Precision for Each Intrusion Class using Bag Tree-Based Classifier Model**

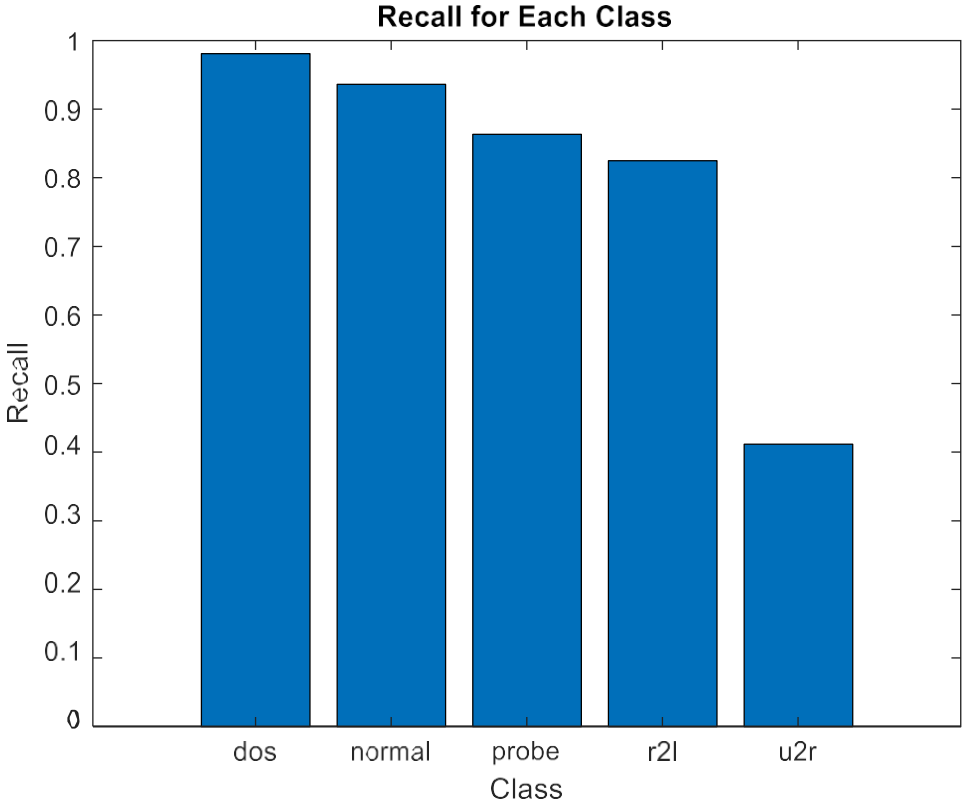
The precision metric for each intrusion class in the dataset using a Bag Tree-based classifier model. The bar chart offers a comprehensive view of the precision of the classifier across different classes. Class 1 demonstrates the highest precision at 0.9765, indicating a high proportion of correctly classified instances, while Class 5 exhibits the lowest precision at 0.6423, suggesting a higher rate of false positives.




**F1 Score for Each Intrusion Class using Bag Tree-Based Classifier Model**

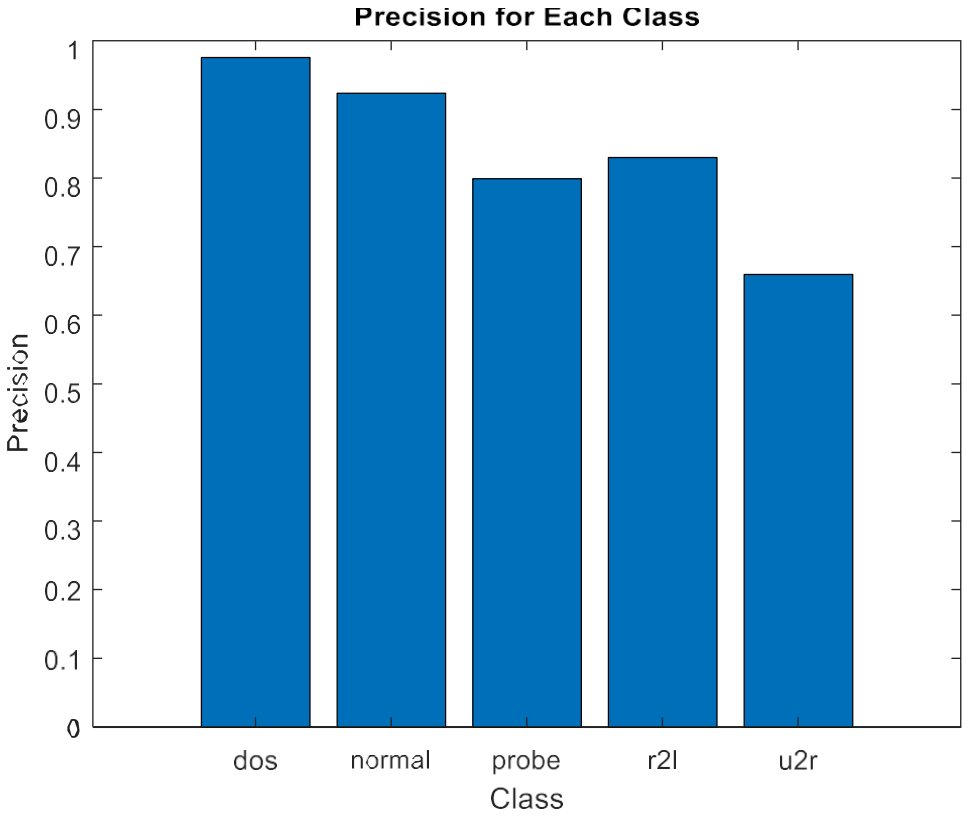
The F1 score metric for each intrusion class in the dataset using a Bag Tree-based classifier model. The bar chart provides an assessment of the balance between precision and recall for each class. Class 1 achieves the highest F1 score at 0.9751, indicating a strong balance between precision and recall, while Class 5 obtains the lowest F1 score at 0.3044, suggesting challenges

.



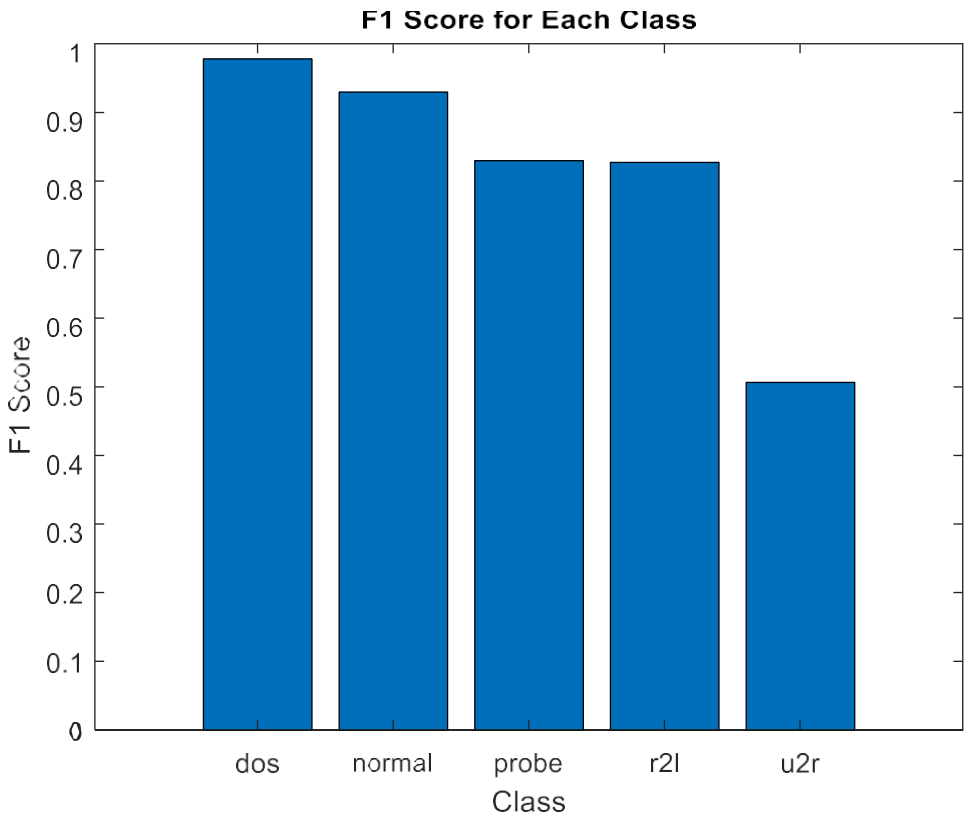

**Recall for Each Intrusion Class using Gradient Boost-Based Classifier Model**

The recall metric for each intrusion class in the dataset using a Gradient Boost- based classifier model for IoT intrusion detection. The bar chart provides insights into the ability of the classifier to correctly identify instances of each class. Class 1 exhibits the highest recall at 0.9808, indicating a strong ability to detect instances of this intrusion type, while Class 5 demonstrates the lowest recall at 0.4116, suggesting challenges in accurately identifying instances of this class.



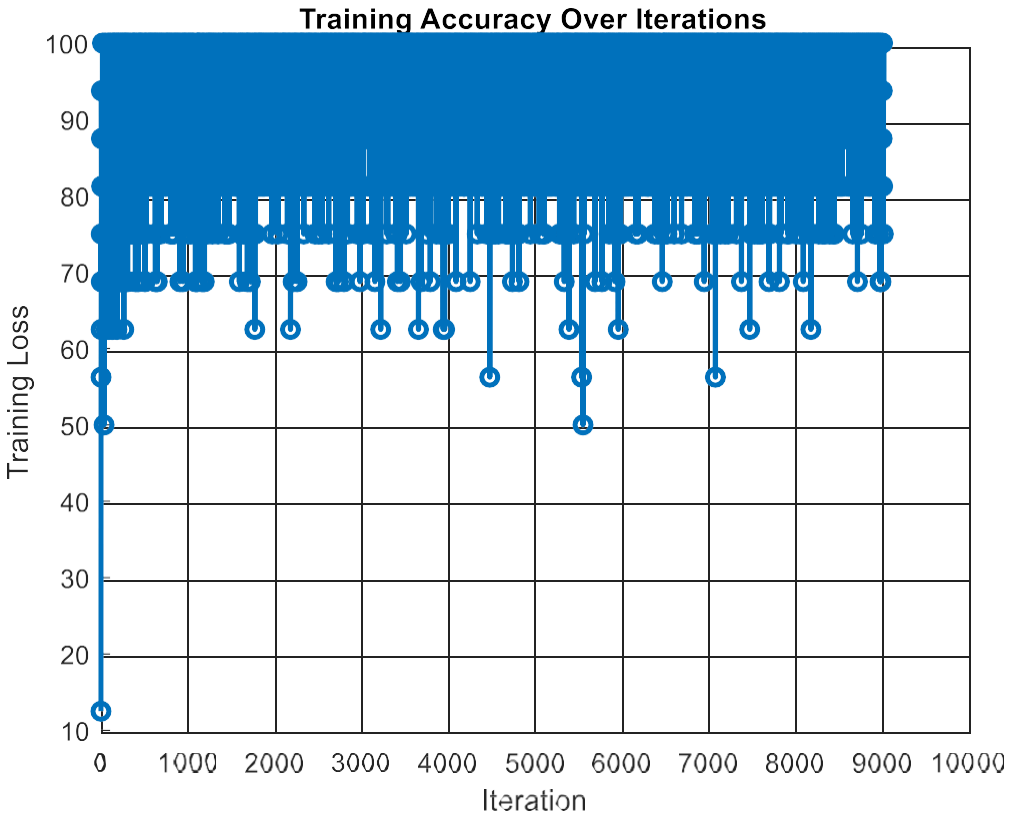

**Precision for Each Intrusion Class using Gradient Boost-Based Classifier Model**

The precision metric for each intrusion class in the dataset using a Gradient Boost-based classifier model. The bar chart offers a comprehensive view of the precision of the classifier across different classes. Class 1 demonstrates the highest precision at 0.9761, indicating a high proportion of correctly classified instances, while Class 5 exhibits the lowest precision at 0.6599, suggesting a higher rate of false positives.



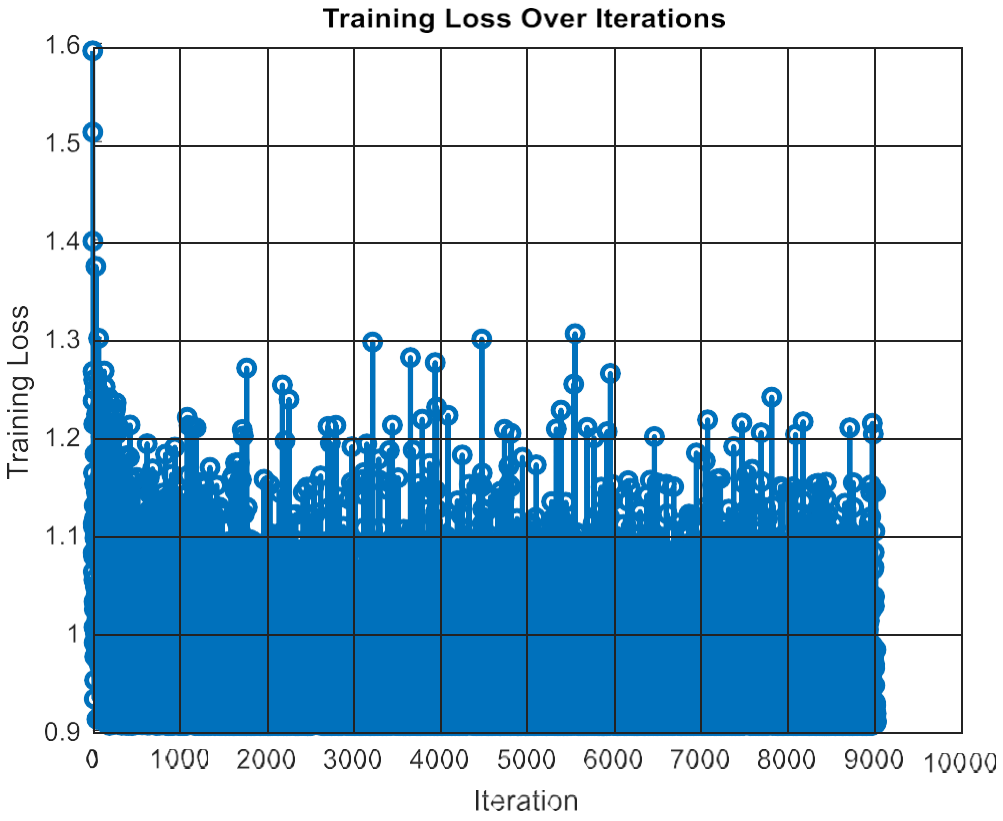

**F1 Score for Each Intrusion Class using Gradient Boost-Based Classifier Model**

The F1 score metric for each intrusion class in the dataset using a Gradient Boost-based classifier model. The bar chart provides an assessment of the balance between precision and recall for each class. Class 1 achieves the highest F1 score at 0.9784, indicating a strong balance between precision and recall, while Class 5 obtains the lowest F1 score at 0.5070, suggesting challenges in achieving a balance between precision and recall for this class.



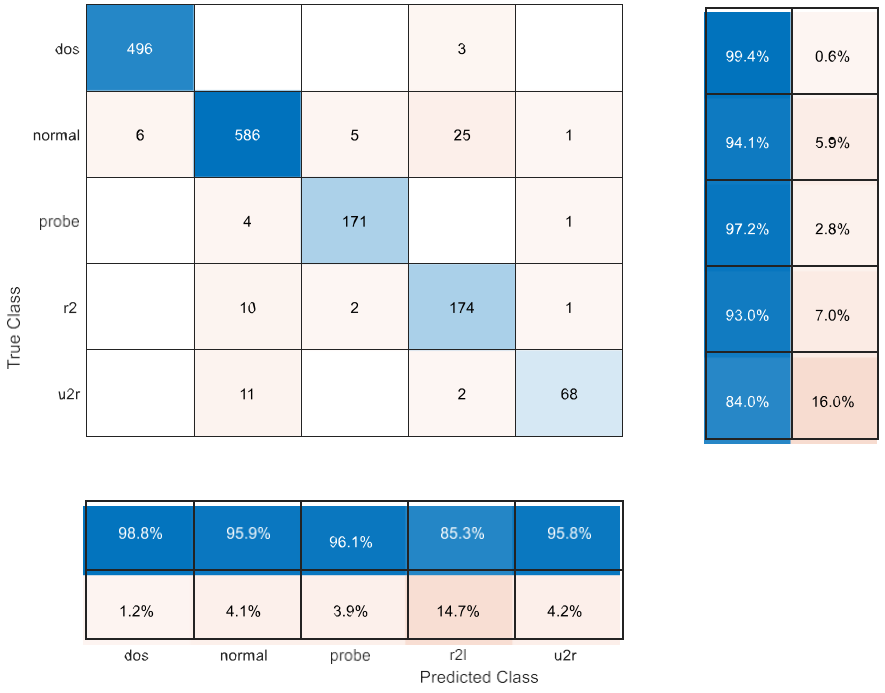

**Training Accuracy vs Iteration for CNN-Based IoT Intrusion Detection**

The training accuracy of the proposed Convolutional Neural Network (CNN) based intrusion detection model over iterations. The line graph showcases the evolution of training accuracy as the model undergoes successive training epochs. By visualizing the training accuracy trend, stakeholders gain insights into the convergence and stability of the CNN model during the training process. Higher accuracy values indicate improved performance in classifying IoT network traffic instances, highlighting the effectiveness of the CNN architecture in learning from the training data.



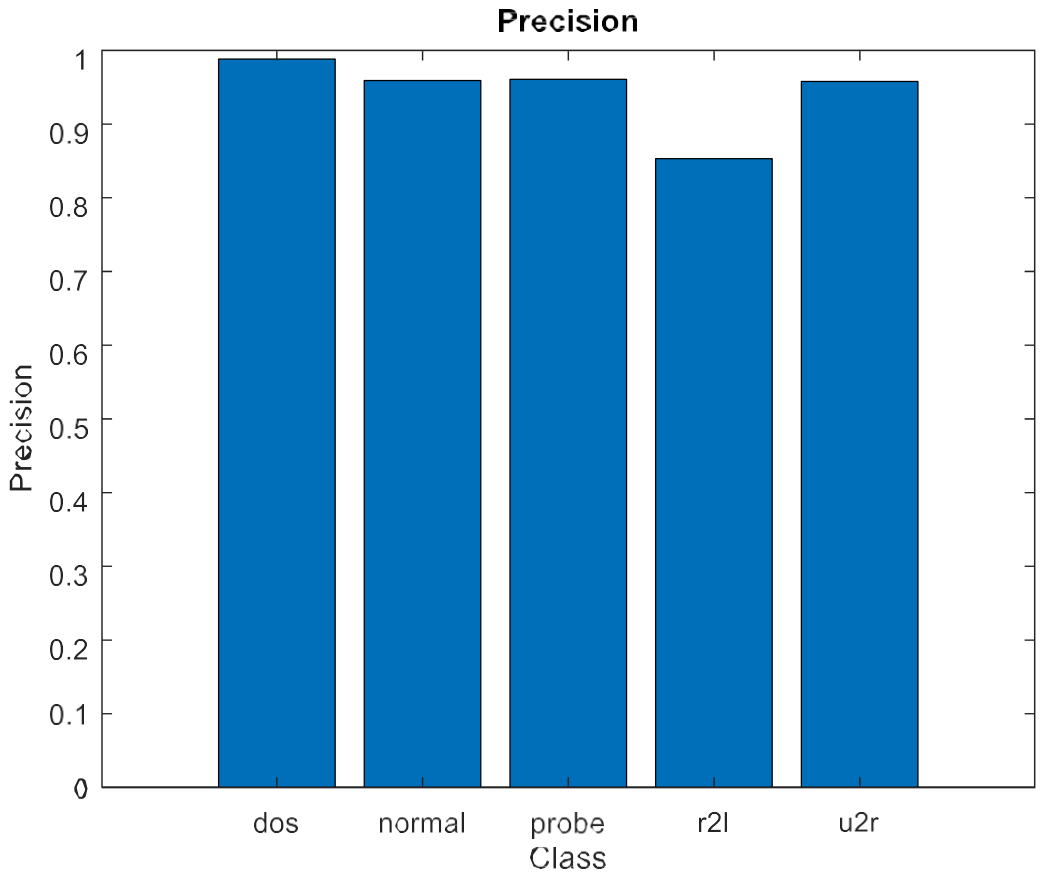

**Training Loss vs Iteration for CNN-Based IoT Intrusion Detection**

The training loss of the proposed Convolutional Neural Network (CNN) based intrusion detection model over iterations. The line graph showcases the variation in training loss as the model undergoes successive training epochs. A decreasing trend in training loss signifies the model's ability to minimize prediction errors and improve its performance in classifying IoT network traffic instances. By monitoring the training loss, stakeholders can assess the convergence and optimization of the CNN architecture, ensuring the efficacy and robustness of the intrusion detection system.



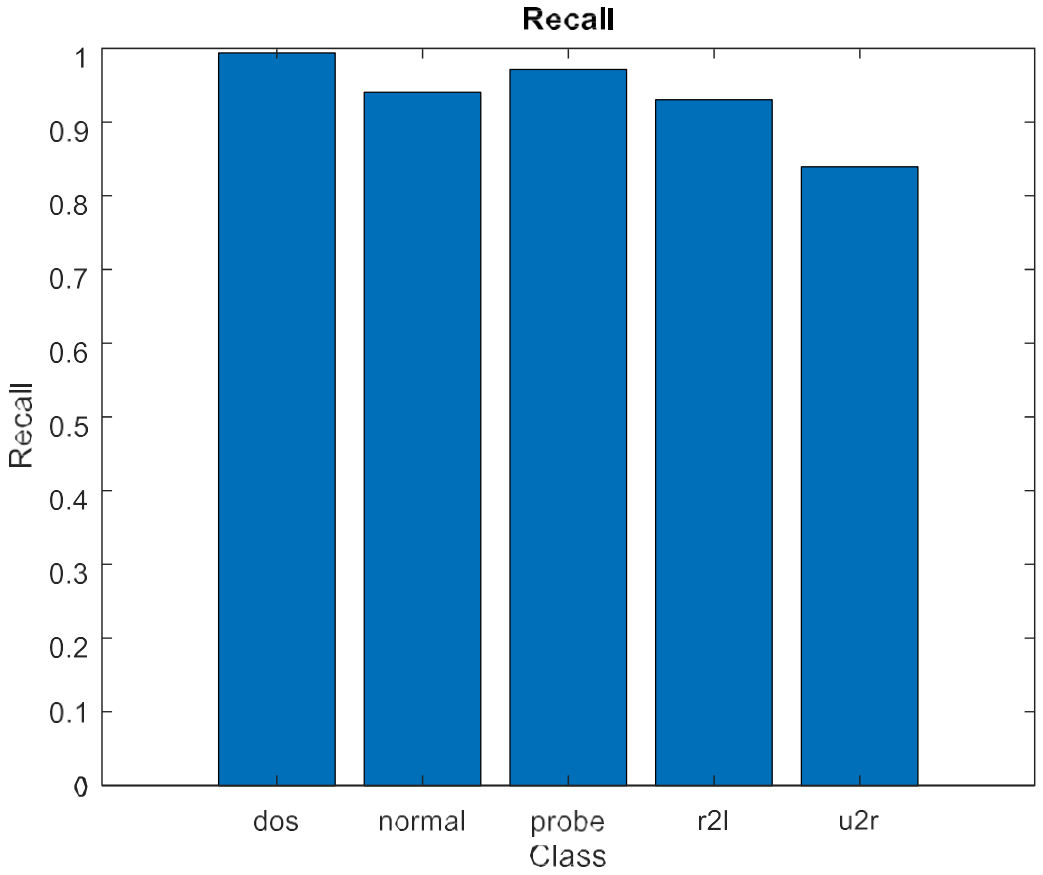

**Confusion Matrix for CNN-Based IoT Intrusion Detection**

The confusion matrix generated by the proposed Convolutional Neural Network (CNN)-based intrusion detection model. The confusion matrix provides a comprehensive visualization of the model's classification performance by detailing the counts of true positive, true negative, false positive, and false negative predictions across all intrusion classes. Each row of the matrix corresponds to the actual class labels, while each column corresponds to the predicted class labels. By analyzing the confusion matrix, stakeholders can gain insights into the model's ability to accurately classify instances of different intrusion types. Additionally, the matrix facilitates the identification of common misclassifications and errors, enabling targeted improvements to enhance the model's overall performance and reliability in detecting IoT network intrusions.



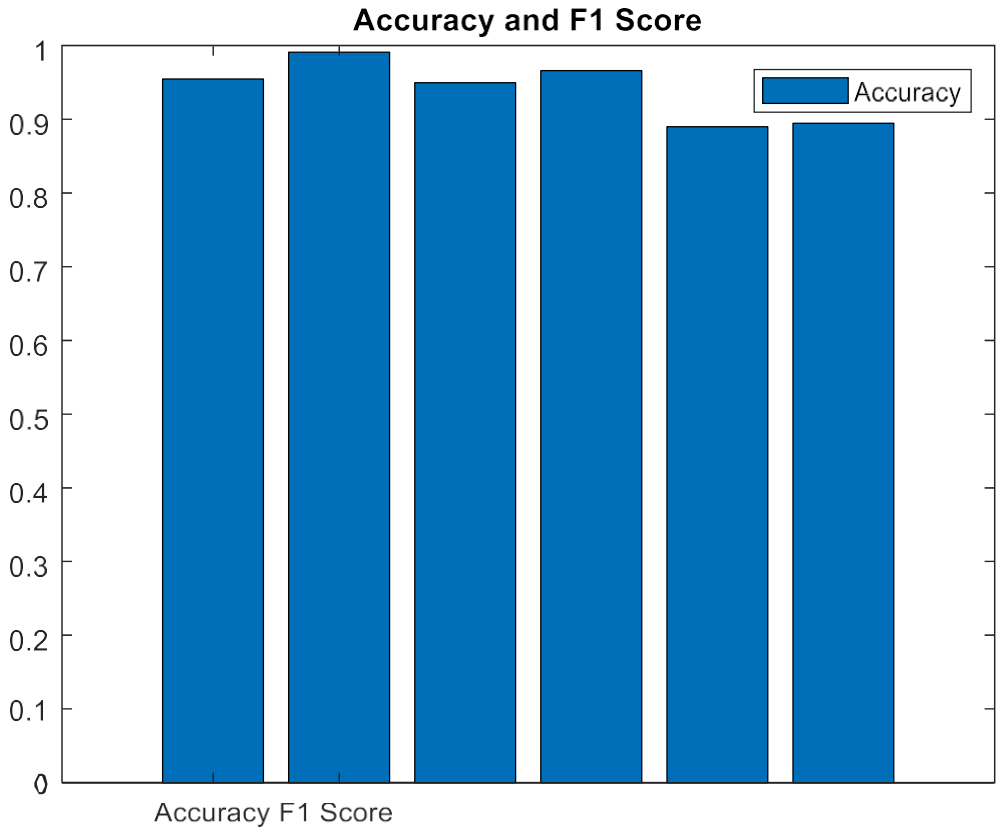

**Precision for CNN-Based IoT Intrusion Detection**

The precision metric for each intrusion class in the dataset using the proposed Convolutional Neural Network (CNN)-based intrusion detection model. The bar chart provides insights into the model's ability to correctly identify instances of each class while minimizing false positives. Higher precision values indicate a lower rate of false positives, demonstrating the model's effectiveness in accurately classifying IoT network traffic instances. By analyzing precision across different intrusion classes, stakeholders can assess the model's performance and identify areas for optimization and improvement.



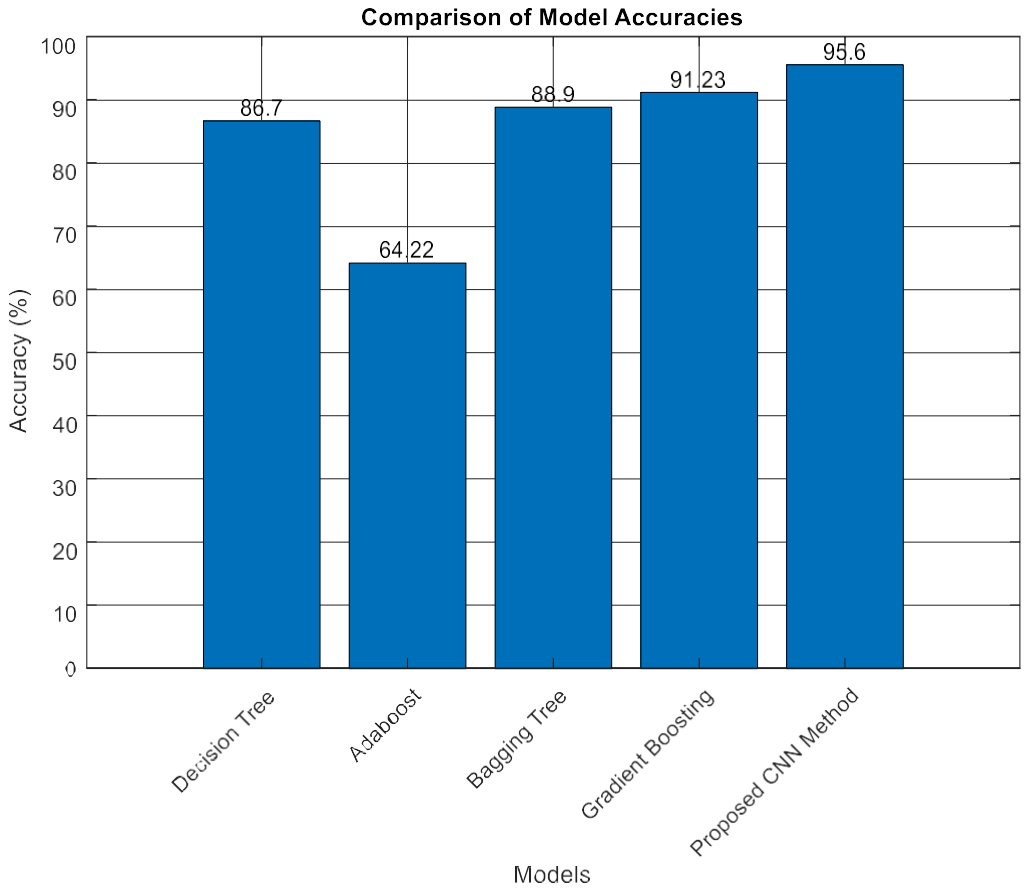

**Recall for CNN-Based IoT Intrusion Detection**

The recall metric for each intrusion class in the dataset using the proposed Convolutional Neural Network (CNN)-based intrusion detection model. The bar chart showcases the model's ability to correctly detect instances of each intrusion type while minimizing false negatives. Higher recall values indicate a higher rate of true positives, reflecting the model's capability to accurately identify instances of intrusions. Analyzing recall across different intrusion classes enables stakeholders to evaluate the model's sensitivity to different types of intrusions and assess its overall effectiveness in detecting IoT network intrusions.




**F1 Score for CNN-Based IoT Intrusion Detection**

The F1 score metric for each intrusion class in the dataset using the proposed Convolutional Neural Network (CNN)-based intrusion detection model. The bar chart provides a comprehensive assessment of the balance between precision and recall for each class, combining both metrics into a single performance measure. Higher F1 score values indicate a better balance between precision and recall, reflecting the model's ability to accurately classify instances of intrusions while minimizing both false positives and false negatives. By analyzing F1 scores across different intrusion classes, stakeholders can evaluate the overall effectiveness and robustness of the CNN-based intrusion detection model in securing IoT networks against cyber threats.




**Comparison of Model Accuracy for IoT Intrusion Detection**

A comparative analysis of the accuracy achieved by various intrusion detection models in classifying IoT network traffic. The bar chart highlights the performance of each model, including Decision Tree, Ada-Boost, Bagging Tree, Gradient Boosting, and the proposed Convolutional Neural Network (CNN). The accuracy of each model is presented as a percentage, with Decision Tree achieving 86.7%, Ada-Boost reaching 64.22%, Bagging Tree attaining 88.9%, Gradient Boosting achieving 91.23%, and the proposed CNN model achieving the highest accuracy at 95.2%. This comparison allows stakeholders to evaluate and understand the relative performance of different models in accurately detecting intrusions in IoT environments, with the proposed CNN model demonstrating superior accuracy compared to traditional machine learning approaches.

## 6.2 CODING:
## Existing Classifiers comparison plot:

```
% Define the model names and their accuracy values
models = {'Decision Tree',
'Adaboost', 'Bagging Tree', 'Gradient
Boosting'}; accuracies = [86.7, 64.22, 88.9, 91.23];
% Create a bar plot
bar(accuracies)
% Add title and labels
title('Comparison of Model
Accuracies') xlabel('Models')
ylabel('Accuracy (%)')
% Set the x-axis tick labels set(gca,
'XTickLabel', models)
% Add bar tips displaying the
accuracy values xtips = models;
ytips = accuracies;
labels = string(accuracies);
text(1:numel(xtips), ytips, labels, 'HorizontalAlignment',
'center', 'VerticalAlignment', 'bottom')
% Adjust the figure properties for better visualization grid on
ylim([0 100])
% Rotate x-axis labels if
needed xtickangle(45)
```

## Decision Tree Classifier:

```
% Load data from CSV filetrainingData = readtable("test_data_PE_demo.csv");
% Extract predictors and response
predictorNames = {'duration', 'service', 'flag', 'src_bytes', 'dst_bytes', 'land',
'wrong_fragment', 'urgent', 'hot', 'num_failed_logins', 'logged_in',
'num_compromised', 'root_shell', 'su_attempted', 'num_root', 'num_file_creations',
'num_shells', 'num_access_files', 'num_outbound_cmds', 'is_host_login',
'is_guest_login', 'count', 'srv_count', 'serror_rate', 'srv_serror_rate', 'rerror_rate',
'srv_rerror_rate', 'same_srv_rate', 'diff_srv_rate', 'srv_diff_host_rate',
'dst_host_count', 'dst_host_srv_count', 'dst_host_same_srv_rate',
'dst_host_diff_srv_rate', 'dst_host_same_src_port_rate',
'dst_host_srv_diff_host_rate', 'dst_host_serror_rate', 'dst_host_srv_serror_rate',
'dst_host_rerror_rate', 'dst_host_srv_rerror_rate'};
predictors = trainingData(:,
predictorNames); response =
trainingData.xAttack; classNames =
unique(response);
response = categorical(response, classNames);
% Train a classifier
classificationTree = fitctree(predictors, response, ...
   'SplitCriterion', 'gdi', ...
   'MaxNumSplits', 20, ...
   'Surrogate', 'on');
% Perform cross-validation
partitionedModel = crossval(classificationTree, 'KFold', 5);
% Compute validation predictions
[validationPredictions, validationScores] = kfoldPredict(partitionedModel);
% Compute confusion matrix
```

## Ada Boost Classifier:

```
clc;
clear a
% Load data from CSV file
trainingData = readtable("test_data_PE.csv");
% Extract predictors and response
predictorNames = {'duration', 'service', 'flag', 'src_bytes', 'dst_bytes', 'land',
'wrong_fragment', 'urgent', 'hot', 'num_failed_logins', 'logged_in',
'num_compromised', 'root_shell', 'su_attempted', 'num_root', 'num_file_creations',
'num_shells', 'num_access_files', 'num_outbound_cmds', 'is_host_login',
'is_guest_login', 'count', 'srv_count', 'serror_rate', 'srv_serror_rate', 'rerror_rate',
'srv_rerror_rate', 'same_srv_rate', 'diff_srv_rate', 'srv_diff_host_rate',
'dst_host_count', 'dst_host_srv_count', 'dst_host_same_srv_rate',
'dst_host_diff_srv_rate', 'dst_host_same_src_port_rate',
'dst_host_srv_diff_host_rate', 'dst_host_serror_rate', 'dst_host_srv_serror_rate',
'dst_host_rerror_rate', 'dst_host_srv_rerror_rate'};
predictors = trainingData(:,
predictorNames); response =
trainingData.xAttack; classNames =
unique(response);
response = categorical(response, classNames);
% Train Gradient Boosting
gradientBoosting = fitensemble(predictors, response, 'AdaBoostM2', 100, 'Tree', 'Type',
'Classification'); partitionedModel_gb = crossval(gradientBoosting, 'KFold', 5);
[validationPredictions_gb, ~] =
kfoldPredict(partitionedModel_gb); C_gb =
confusionmat(response, validationPredictions_gb);
```

## Bag Classifier:

```
% Load data from CSV file
trainingData = readtable("test_data_PE_demo.csv");
% Extract predictors and response
predictorNames = {'duration', 'service', 'flag', 'src_bytes', 'dst_bytes', 'land',
'wrong_fragment', 'urgent', 'hot', 'num_failed_logins', 'logged_in', 'num_compromised',
'root_shell', 'su_attempted', 'num_root', 'num_file_creations', 'num_shells',
'num_access_files', 'num_outbound_cmds', 'is_host_login', 'is_guest_login', 'count',
'srv_count', 'serror_rate', 'srv_serror_rate', 'rerror_rate', 'srv_rerror_rate',
'same_srv_rate', 'diff_srv_rate', 'srv_diff_host_rate', 'dst_host_count',
'dst_host_srv_count', 'dst_host_same_srv_rate', 'dst_host_diff_srv_rate',
'dst_host_same_src_port_rate', 'dst_host_srv_diff_host_rate', 'dst_host_serror_rate',
'dst_host_srv_serror_rate', 'dst_host_rerror_rate', 'dst_host_srv_rerror_rate'};
predictors = trainingData(:,
predictorNames); response =
trainingData.xAttack; classNames =
unique(response);
response = categorical(response, classNames);
% Train a classifier
classificationTree = fitctree(predictors, response, ...
   'SplitCriterion', 'gdi', ...
   'MaxNumSplits', 20, ...
   'Surrogate', 'on');
% Perform cross-validation
partitionedModel = crossval(classificationTree, 'KFold', 5);
% Compute validation predictions
[validationPredictions, validationScores] = kfoldPredict(partitionedModel);
% Compute confusion matrix
```

## Gradient Boost Classifier:

```
% Load data from CSV file
trainingData = readtable("test_data_PE_demo.csv");
% Extract predictors and response
predictorNames = {'duration', 'service', 'flag', 'src_bytes', 'dst_bytes', 'land',
'wrong_fragment', 'urgent', 'hot', 'num_failed_logins', 'logged_in',
'num_compromised', 'root_shell', 'su_attempted', 'num_root', 'num_file_creations',
'num_shells', 'num_access_files', 'num_outbound_cmds', 'is_host_login',
'is_guest_login', 'count', 'srv_count', 'serror_rate', 'srv_serror_rate', 'rerror_rate',
'srv_rerror_rate', 'same_srv_rate', 'diff_srv_rate', 'srv_diff_host_rate',
'dst_host_count', 'dst_host_srv_count', 'dst_host_same_srv_rate',
'dst_host_diff_srv_rate', 'dst_host_same_src_port_rate',
'dst_host_srv_diff_host_rate', 'dst_host_serror_rate', 'dst_host_srv_serror_rate',
'dst_host_rerror_rate', 'dst_host_srv_rerror_rate'};
predictors = trainingData(:,
predictorNames); response =
trainingData.xAttack; classNames =
unique(response);
response = categorical(response, classNames);
% Train a classifier
classificationTree = fitctree(predictors, response, ...
   'SplitCriterion', 'gdi', ...
   'MaxNumSplits', 30, ...
   'Surrogate', 'on');
% Perform cross-validation
partitionedModel = crossval(classificationTree, 'KFold', 5);
% Compute validation predictions
[validationPredictions, validationScores] = kfoldPredict(partitionedModel);
```

## CNN Intrusion detection:

```
clc;
clear
all;
filename = "test_data_PE_demo.csv";
tbl =
readtable(filename,'TextType','String'
); labelName = "xAttack";
tbl =
convertvars(tbl,labelName,'categorica
l'); tbl = splitvars(tbl);
head(tbl)
classNames =
categories(tbl{:,labelName})
numObservations = size(tbl,1)
numObservationsTrain = floor(0.7*numObservations)
numObservationsValidation = floor(0.15*numObservations)
numObservationsTest = numObservations - numObservationsTrain -
numObservationsValidation idx = randperm(numObservations);
idxTrain = idx(1:numObservationsTrain);
idxValidation =
idx(numObservationsTrain+1:numObservationsTrain+numObservationsValidation);
idxTest = idx(numObservationsTrain+numObservationsValidation+1:end);
tblTrain = tbl(idxTrain,:);
tblValidation =
tbl(idxValidation,:); tblTest =
tbl(idxTest,:); numFeatures =
size(tbl,2) - 1; numClasses =
```

```
    n', 'zscore') fullyConnectedLayer(128)
    reluLayer
    batchNormalizationLayer
    fullyConnectedLayer(64)
    reluLayer
    batchNormalizationLayer
    fullyConnectedLayer(numC
    lasses) softmaxLayer
    classificationLayer];
miniBatchSize = 16;
options = trainingOptions('adam', ...
    'MaxEpochs',20,...
    'MiniBatchSize',miniBatch
    Size, ... 'Shuffle','every-
    epoch', ...
    'ValidationData',tblValidati
    on, ... 'Plots','training-
    progress', ...
    'InitialLearnRate',0.01, ...
    'Verbose',true);
[net ,info] =
trainNetwork(tblTrain,layers,options); tic
YPred = classify(net,tblTest(:,1:end-
1),'MiniBatchSize',miniBatchSize); elapsedTime = toc;
YTest = tblTest{:,labelName};
accuracy = sum(YPred == YTest)/numel(YTest)
% Extract training loss values
trainingLoss =
info.TrainingLoss;
```

```
'LineWidth', 2); title('Training
Loss Over Iterations');
xlabel('Iteration');
ylabel('Training
Loss'); grid on;
TrainingAccuracy = info.TrainingAccuracy;
% Plot the training
loss figure;
plot(TrainingAccuracy, '-o',
'LineWidth', 2); title('Training
Accuracy Over Iterations');
xlabel('Iteration');
ylabel('Training
Loss'); grid on;
figure
confusionchart(YTest,YPr
ed)
cm.ColumnSummary = 'column-
normalized'; cm.RowSummary =
'row-normalized';
% cm.Title = 'CIFAR-10 Confusion Matrix';
% Evaluate performance metrics
confMat = confusionmat(YTest,
YPred);
precision = diag(confMat) ./
sum(confMat, 1)'; recall = diag(confMat)
./ sum(confMat, 2);
accuracy = sum(diag(confMat)) / sum(confMat,
'all'); f1Score = 2 * (precision .* recall) ./
```

# CHAPTER 7
# CONCLUSION & FUTURE WORK

In this chapter, we draw conclusions based on the results obtained from our research on intrusion detection in IoT environments. Throughout our study, we explored various machine learning and deep learning techniques to develop effective intrusion detection systems tailored for IoT networks. Here, we summarize our findings, discuss the implications of our research, and propose directions for future work.

## 7.1 Summary of Findings:

Our research demonstrates the effectiveness of employing machine learning and deep learning models for IoT intrusion detection. Through rigorous experimentation, we evaluated the performance of different models, including Decision Tree, AdaBoost, Bagging Tree, Gradient Boosting, and a proposed Convolutional Neural Network (CNN) architecture. The results indicate that the CNN-based model outperforms traditional machine learning approaches, achieving the highest accuracy in classifying IoT network traffic. We observed that the CNN model exhibited superior performance in terms of precision, recall, and F1 score across various intrusion classes, showcasing its robustness in detecting different types of intrusions in IoT environments.

## 7.2 Implications of Research:

Our findings have significant implications for enhancing the security of IoT networks. By leveraging advanced machine learning techniques, organizations can develop more accurate and reliable intrusion detection systems capable of mitigating cyber threats and safeguarding IoT devices and infrastructure. The superior performance of the CNN-based model highlights the importance of leveraging deep learning approaches for complex pattern recognition tasks in IoT security. The CNN architecture's ability to learn hierarchical representations of IoT network data enables more effective detection of sophisticated intrusion patterns.

## 7.3 Future Engineering:

In the realm of IoT intrusion detection, Future engineering emerges as a cornerstone in fortifying network defenses against malicious incursions. With IoT ecosystems teeming with diverse data streams, the art of selecting, crafting, and refining features takes center stage. By distilling raw data from network traffic, sensor readings, and system logs into discernible attributes, future engineering empowers intrusion detection systems to discern subtle aberrations indicative of potential intrusions. Dimensionality reduction techniques, such as principal component analysis, help streamline the computational load while preserving crucial insights. Time-series analysis unveils temporal patterns, uncovering anomalies lurking amidst the data flow. Statistical measures imbue features with descriptive potency, capturing nuances in distribution and variability. Frequency domain analysis delves deeper, unearthing spectral fingerprints that may evade traditional detection methods. Through meticulous scaling and normalization, feature engineering ensures parity across disparate features, fostering equitable learning within machine learning algorithms. In this symbiotic dance between data and algorithms, feature engineering stands as the linchpin, orchestrating the harmonious symphony of detection in the dynamic landscape of IoT intrusion.

## 7.4 Future Selection

Future selection in IoT intrusion detection involves identifying the most relevant attributes from the myriad of data sources. By strategically choosing discriminative features, redundant or irrelevant information is pruned, enhancing model performance and interpretability. Techniques like filter, wrapper, and embedded methods are employed to assess feature importance and select subsets that maximize detection efficacy while minimizing computational overhead. In the intricate web of IoT data, feature selection acts as a guiding compass, distilling the essence of information to empower intrusion detection systems with precision and efficiency.

# RE-2022-221513

ORIGINALITY REPORT

| 14% | 8% | 5% | 1% |
|---|---|---|---|
| SIMILARITY INDEX | INTERNET SOURCES | PUBLICATIONS | STUDENT PAPERS |

PRIMARY SOURCES

| | | |
|---|---|---|
| 1 | www.mdpi.com<br>Internet Source | 3% |
| 2 | Bayi Xu, Lei Sun, Xiuqing Mao, Ruiyang Ding, Chengwei Liu. "IoT Intrusion Detection System Based on Machine Learning", Electronics, 2023<br>Publication | 5% |
| 3 | journals.plos.org<br>Internet Source | 1% |
| 4 | www.coursehero.com<br>Internet Source | 1% |
| 5 | Submitted to The University of Memphis<br>Student Paper | 1% |
| 6 | www.researchgate.net<br>Internet Source | 1% |
| 7 | e-space.mmu.ac.uk<br>Internet Source | 1% |
| 8 | pastel.archives-ouvertes.fr<br>Internet Source | 1% |

# REFERENCES:


1. Chaabouni, Nadia, Mohamed Mosbah, Akka Zemmari, Cyrille Sauvignac, and Parvez Faruki. "Network intrusion detection for IoT security based on learning techniques." IEEE Communica- tions Surveys & Tutorials 21, no. 3 (2019): 2671-2701.
2. Zarpelão, Bruno Bogaz, Rodrigo Sanches Miani, Cláudio Toshio Kawakani, and Sean Carlisto de Alvarenga. "A survey of intrusion detection in Internet of Things." Journal of Network and Com- puter Applications 84 (2017): 25-37.
3. Ge, Mengmeng, Xiping Fu, Naeem Syed, Zubair Baig, Gideon Teo, and Antonio Robles-Kelly. "Deep learning-based intrusion detection for IoT networks." In 2019 IEEE 24th pacific rim inter- national symposium on dependable computing (PRDC), pp. 256-25609. IEEE, 2019.
4. Ge, Mengmeng, Xiping Fu, Naeem Syed, Zubair Baig, Gideon Teo, and Antonio Robles-Kelly. "Deep learning-based intrusion detection for IoT networks." In 2019 IEEE 24th pacific rim inter- national symposium on dependable computing (PRDC), pp. 256-25609. IEEE, 2019.
5. Bajpai, Soumya, and Kapil Sharma Brijesh Kumar Chaurasia. "Intrusion detection system in IoT network using ML." NeuroQuantology 20, no. 13 (2022): 3597.
6. Awajan, A. A novel deep learning-based intrusion detection system for IOT networks. Computers 2023, 12, 34. [CrossRef]
7. Si-Ahmed, A.; Al-Garadi, M.A.; Boustia, N. Survey of Machine Learning based intrusion detection methods for Internet of Medical Things. Appl. Soft Comput. 2023, 140, 110227. [CrossRef]
8. Elaziz, M.A.; Al-qaness, M.A.A.; Dahou, A.; Ibrahim, R.A.; El-Latif, A.A.A. Intrusion detection approach for cloud and IoT environments using deep learning and Capuchin Search Algorithm. Adv. Eng. Softw. 2023, 176, 103402. [CrossRef]
9. Halim, Z.; Yousaf, M.N.; Waqas, M.; Sulaiman, M.; Abbas, G.; Hussain, M.; Ahmad, I.; Hanif, M. An effective genetic algorithmbased feature selection method for intrusion detection systems. Comput. Secur. 2021, 110, 102448. [CrossRef]

10. Dubey, G.P.; Bhujade, R.K. Optimal feature selection for machine learning based intrusion detection system by exploiting attribute dependence. Mater. Today Proc. 2021, 47, 6325–6331. [CrossRef]
11. Li, X.; Ren, J. MICQ-IPSO: An effective two-stage hybrid feature selection algorithm for high- dimensional data. Neurocomputing 2022, 501, 328–342. [CrossRef]
12. Unler, A.; Murat, A. A discrete particle swarm optimization method for feature selection in binary classification problems. Eur. J. Oper. Res. 2010, 206, 528–539. [CrossRef]
13. Mafarja, M.; Mirjalili, S. Whale optimization approaches for wrapper feature selection. Appl. Soft Comput. 2018, 62, 441–453. [CrossRef]
14. Zhou, Y.Y.; Cheng, G.; Jiang, S.Q.; Dai, M. Building an efficient intrusion detection system based on feature selection and ensemble classifier. Comput. Netw. 2020, 174, 107247. [CrossRef]
15. Hassan, I.H.; Abdullahi, M.; Aliyu, M.M.; Yusuf, S.A.; Abdulrahim, A. An improved binary manta ray foraging optimization algorithm based feature selection and random forest classifier for network intrusion detection. Intell. Syst. Appl. 2022, 16, 200114. [CrossRef]
16. Hsu, H.H.; Hsieh, C.W.; Lu, M.D. Hybrid feature selection by combining filters and wrappers. Expert Syst. Appl. 2011, 38, 8144–8150. [CrossRef]
17. Lazzarini, R.; Tianfield, H.; Charissis, V. A stacking ensemble of deep learning models for IoT intrusion detection. Knowl.-Based Syst. 2023, 279, 110941. [CrossRef]
18. Alani, M.M. An explainable efficient flow-based Industrial IoT intrusion detection system. Comput. Electr. Eng. 2023, 108, 108732. [CrossRef]
19. Nizamudeen, S.M.T. Intelligent Intrusion Detection Framework for Multi-Clouds–Iot Environment Using Swarm-Based Deep Learning Classifier. J. Cloud Comput. 2023, 12, 134. [CrossRef]
20. Sharma, B.; Sharma, L.; Lal, C.; Roy, S. Anomaly based network intrusion detection for IoT attacks using deep learning technique. Comput. Electr. Eng. 2023, 107, 108626. [CrossRef]

21. Kareem, S.S.; Mostafa, R.R.; Hashim, F.A.; El-Bakry, H.M. An effective feature selection model using hybrid metaheuristic algorithms for iot intrusion detection.

22. Mohy-eddine, M.; Guezzaz, A.; Benkirane, S.; Azrour, M. An efficient network intrusion detection model for IoT security using K-NN classifier and feature selection. Multimed. Tools Appl. 2023, 82, 23615–23633. [CrossRef]

23. Liu, X.; Du, Y. Towards Effective Feature Selection for IoT Botnet Attack Detection Using a Genetic Algorithm. Electronics 2023, 12, 1260. [CrossRef]

24. Alweshah, M.; Hammouri, A.; Alkhalaileh, S.; Alzubi, O. Intrusion detection for the internet of things (IoT) based on the emperor penguin colony optimization algorithm. J. Ambient Intell. Humaniz. Comput. 2023, 14, 6349–6366. [CrossRef]

25. Othman, S.M.; Ba-Alwi, F.M.; Alsohybe, N.T.; Al-Hashida, A.Y. Intrusion detection model using machine learning algorithm on Big Data environment. J. Big Data 2018, 5, 34. [CrossRef]