


الجمهورية الجزائرية الديمقراطية الشعبية
وزارة التعليم العالي والبحث العلمي
جامعة قاصدي مرباح ورقلة
كلية تكنولوجيات الإعلام والإتصال
قسم الإعلام الآلي وتكنولوجيات الإعلام

Republic of Algeria Democratic and People's
Ministry of Higher Education and Scientific Research
University of KASDI Merbah – Ouargla
Faculty of New Technologies of Information and Communication
Science Computer of Department and Information Technologies

Professional Master Thesis

Field: Mathematics and Computer Science
Sector: Computer Science
Specialty: Network Administration and Security

Theme

Federated & Centralized Deep Learning for Cyberattack Detection in IoMT

Presented by: Dadda Safa and Badjadi Katrennada

Publicly discussed: 10/06/2025

Jury members:

Dr. BENBEZIANE Mohammed	President	UKM Ouargla
Dr. BENBEZIANE Mohammed	Supervisor	UKM Ouargla
Dr. BENBEZIANE Mohammed	Examiner	UKM Ouargla

Academic year: 2024/2025

Acknowledgments

First and foremost, we extend our sincere thanks to Allah Almighty for His guidance and care. He alone is the source of success and blessings in every step we take. We also express our deepest gratitude to our dear parents and families, who have been our first supporters in this academic journey, with their sacrifices and invaluable guidance. We would like to express our profound thanks and appreciation to our esteemed supervisor, Dr. Benbezziane Mohammed, for his wise advice, continuous support, and great care, which played a significant role in our success and the completion of this work. We also thank the distinguished members of the jury for their time and careful attention to our work, as well as our dear professors who contributed to enriching our knowledge and guiding us. Our deep gratitude goes to everyone who played a role in supporting and assisting us throughout this academic journey. Finally, we offer a special thanks to our families and friends, who stood by us at every moment, supporting us with their love and encouragement, and were a key reason for our perseverance. Thank you all from the bottom of our hearts.

Abstract

As technology continues to evolve, one of the most promising recent advancements is the Internet of Things (IoT) and its applications in various fields, especially healthcare. This development has helped formulate the concept known as the Internet of Medical Things (IoMT), which has tremendous possibilities in improving healthcare services, especially in remote patient monitoring, real-time vital signs tracking, and automated medical decision processes. This study focuses on the impact of artificial intelligence, particularly deep learning models, in increasing the security of IoMT environments, which face the brunt of serious cyber threats. Attack detection was modeled with four different deep learning models: two centralized (CL-CNN and CL-LSTM) and two federated (FL-CNN and FL-LSTM) using the CIC-BCCC-NRC-IoMT-2024 dataset. Evaluation results demonstrate that federated learning achieves commendable results while maintaining data confidentiality, thereby serving as a powerful method for protecting intelligent medical systems.

Keywords: Federated learning, IoMT Security, Deep learning, Centralized Learning, CNN, LSTM, Cyberattack.

Résumé

Avec l'évolution continue de la technologie, l'une des avancées les plus prometteuses récentes est l'Internet des objets (IoT) et ses applications dans divers domaines, en particulier la santé. Ce développement a permis de formuler le concept d'Internet des objets médicaux (IoMT), qui offre de grandes possibilités pour améliorer les services de santé, notamment dans la surveillance des patients à distance, le suivi en temps réel des signes vitaux, et les processus de décision médicale automatisés. Cette étude se concentre sur l'impact de l'intelligence artificielle, en particulier des modèles d'apprentissage profond, dans le renforcement de la sécurité des environnements IoMT, confrontés à de graves cybermenaces. La détection des attaques a été modélisée à l'aide de quatre modèles d'apprentissage profond : deux centralisés (CL-CNN et CL-LSTM) et deux fédérés (FL-CNN et FL-LSTM), en utilisant le jeu de données CIC-BCCC-NRC-IoMT-2024. Les résultats d'évaluation montrent que l'apprentissage fédéré obtient des résultats remarquables tout en préservant la confidentialité des données, constituant ainsi une méthode puissante pour protéger les systèmes médicaux intelligents.

Mots-clés : Apprentissage fédéré, Sécurité de l'Internet des objets médicaux, Apprentissage profond, Apprentissage centralisé, CNN, LSTM, Cyberattaque.

ملخص

مع استمرار تطور التكنولوجيا، يُعد إنترنت الأشياء من أبرز الابتكارات الحديثة، وخاصةً تطبيقاته في مختلفمع استمرار تطور التكنولوجيا، يُعد إنترنت الأشياء من أبرز الابتكارات الحديثة، وخاصةً تطبيقاته في مختلف المجالات، وخصوصًا في مجال الرعاية الصحية. وقد ساهم هذا التطور في صياغة مفهوم يُعرف بإنترنت الأشياء الطبية، الذي يحمل إمكانيات هائلة لتحسين خدمات الرعاية الصحية، خاصة في مجالات مراقبة المرضى عن بُعد، وتتبع العلامات الحيوية في الوقت الفعلي، والقرارات الطبية المؤتمتة. تركز هذه الدراسة على تأثير الذكاء الاصطناعي، وبشكل خاص نماذج التعلم العميق، في تعزيز أمان بيئات ،(IoMT) التي تواجه تهديدات إلكترونية خطيرة. تم نمذجة اكتشاف الهجمات باستخدام أربعة نماذج تعلم عميق: اثنان مركزيان واثنان فيدراليان، باستخدام مجموعة بيانات .CIC-BCCC-NRC-IoMT-2024 تُظهر نتائج التقييم أن التعلم الفيدرالي يحقق نتائج متميزة مع الحفاظ على سرية البيانات، مما يجعله وسيلة قوية لحماية الأنظمة الطبية الذكية.

الكلمات المفتاحية: التعلم الفيدرالي، أمان إنترنت الأشياء الطبية، التعلم العميق، التعلم المركزي، الشبكات العصبية التلافيفية، الشبكات العصبية طويلة وقصيرة المدى، الهجمات السيبرانية.

المحتويات

List of Figures

List of Tables

Acronyms and Abbreviations

IoT	Internet of Things
IoMT	Internet of Medical Things
IMDs	Implantable medical devices
RPMs	Remote Patient Monitoring Systems
HNDs	Hospital Networked Devices
EHRs	Electronic Health Records
SMPC	Secure Multi-Party Computation
IDS	Intrusion Detection Systems
DDoS	Distributed Denial of Service
IA	Intelligence Artificielle
ML	Machine Learning
DL	Deep Learning
SGD	Stochastic Gradient Descent
ANN	Artificial Neural Network
CNN	Convolutional Neural Network
DNN	Deep Neural Network
RNNs	Recurrent Neural Networks
ReLU	Rectified Linear Unit
LSTM	Long Short-Term Memory
FL	Federated Learning
FedProx	Federated Proximal
FedAvg	Federated Averaging
HFL	Horizontal Federated Learning
VFL	Vertical Federated Learning
FTL	Federated Transfer Learning
IID	Independent and Identically Distributed
SMOTE	Synthetic Minority Over-sampling Technique
PCA	Principal Component Analysis

General Introduction

With the rapid advancement of digital technologies, the **Internet of Things (IoT)** has emerged as a transformative paradigm that is reshaping various sectors, especially healthcare. The integration of IoT in medical environments has led to the rise of the **Internet of Medical Things (IoMT)**, which plays a vital role in enhancing healthcare services by enabling remote patient monitoring, personalized treatment, real-time health tracking, and smart medical facilities. However, this increasing reliance on interconnected medical devices has expanded the **attack surface**, exposing these systems to a variety of **cybersecurity threats** that could compromise patient safety and data privacy.

Problem Statement :

Despite its great advantages, IoMT faces **growing cybersecurity challenges**, including **Denial of Service (DoS/DDoS) attacks, spoofing, eavesdropping, and exploitation of communication protocols** such as MQTT. These threats pose significant risks, especially given the sensitive nature of medical data. Therefore, it has become crucial to implement **intelligent mechanisms capable of early detection and accurate classification of cyberattacks**, ensuring the security and trustworthiness of healthcare systems.

Proposed Solution :

To address these challenges, this research proposes the use of **Artificial Intelligence (AI), particularly Deep Learning (DL)**, as a powerful approach to detect and classify cyberattacks in IoMT environments. The proposed framework combines both **Centralized Learning (CL)** and **Federated Learning (FL)** to evaluate the effectiveness of different learning paradigms in safeguarding medical IoT networks.

Applied Work :

The practical part of this research involves the development of **four deep learning models**:

- Two centralized models (**CL-CNN** and **CL-LSTM**)
- Two federated models (**FL-CNN** and **FL-LSTM**)

These models were trained and evaluated using the **CIC-BCCC-NRC-IoMT-2024 Dataset**, which contains real-world cyberattack scenarios targeting IoMT environments. A comprehensive comparison was then conducted to assess the performance of each model in terms of **accuracy, detection rate, false positive rate, and computational efficiency**. This comparison aims to highlight the most effective approach for securing IoMT systems against emerging threats.

Document Plan :

This thesis is structured in **four chapters** as follows:

- **Chapter I: Overview on the Internet of Medical Things (IoMT)**
  This chapter covers the fundamentals of IoT and IoMT, including their architecture, components, and healthcare applications. It discusses IoMT security challenges and analyzes cybersecurity threats, focusing on their impact on confidentiality, integrity, and availability.

- **Chapter II: Theoretical Background**
  This chapter explores the role of AI, particularly Deep Learning, in securing IoMT systems. It details the CL-CNN and CL-LSTM models used for detecting and classifying cyberattacks.

- **Chapter III: Federated Learning survey**
  This chapter introduces Federated Learning as an effective solution for preserving data privacy in IoMT systems. It explores how this decentralized learning approach can be integrated with deep learning models specifically FL-CNN and FL-LSTM to enable secure and privacy-aware cyber defense. The chapter also discusses how these models operate, along with the associated challenges and limitations.

- **Chapter IV: mplementation and Experimental Results**
  This chapter presents the development and evaluation of four models (CL-CNN, CL-LSTM, FL-CNN, and FL-LSTM) using the CIC-BCCC-NRC-IoMT-2024 Dataset. It compares their performance based on accuracy and other metrics, concluding with their effectiveness in securing IoMT environments.

Chapter 1

Overview on the Internet of Medical Things (IoMT)

1.1 Introduction

The online arena has become central to different sectors and is advancing at a faster pace than ever due to the acceleration in technology in the recent years. Internet of Things, or the IoT, is one of the most powerful inventions that has newly emerged and transformed the humanity on a radically human-centric manner. It allows for the interlinking and transmission of information between multitudes of devices, making data collection, analysis, and improvement possible, IoT allows for the automation of industries, savings in costs and improvement in performance.

Healthcare is one sector that IoT enormously impacts. Many furthers specialized sub concepts have emerged, such as the Internet of Medical Things (IoMT) that incorporate smart medical devices and systems to facilitate quality monitoring and management of patients' care services.

This chapter provides an abstract introduction IoT with IoMT i.e the technical and theoretical aspects which relate to those modern concepts. The chapter covers he topic of IoT in detail, its challenges, applications in healthcare, related architecture, and describes the problem statement of this research study bleeds into the role that Artificial Intelligence (AI) plays in augmenting and supporting the IoMT systems.

1.2 Definition of the Internet of Things (IoT)

Although the term Internet of Things (IoT) is ubiquitous in academic, industrial, and technological communities, there is no single, widely accepted definition. This discrepancy is a result of the interdisciplinary nature of IoT and its extremely broad range of applications from home automation and health care systems to industrial control and transportation. However, there is a shared concept to which the majority of definitions gravitate: IoT is a network of connected physical things that are embedded with sensors, software, and communication technologies to collect, transmit, and process data through the internet with minimal or no direct human involvement.

The Internet of Things (IoT), incorporates intelligent, interconnected machines and sensors that can, without needing brute force human labor, exchange information among themselves. It builds on older technologies like RFID and smart sensors as well as communication frameworks in order to facilitate new opportunities for immediate application evaluation.[1]

Similarly, another source defines : "Internet of Things (IoT) is a network of smart things that share information over the internet. The smart things are used to deploy in a different environment to capture the information, and some events are triggered The applications of IoT is a smart city, smart home, Intelligent transportation system, agriculture, hospital, supply chain system, earthquake detection, a smart grid system." [2]

1.3 Components of the Internet of Things (IoT)

While some authors have described the architecture of the Internet of Things (IoT) using a three-layer model which includes the Perception Layer, Network Layer, and Application Layer [3] , other more detailed models described it using a five-layer architecture. These models also add a Processing Layer and Business Layer, arguing more layers improve system control, understanding, and security management over data [4] .

In this research, a four-layer model is used which adds a Processing Layer to the three foundational layers. This selection was made considering the importance of the Processing Layer within the Internet of Medical Things (IoMT) framework, the layer responsible for data analysis, cyber threat detection, and mitigation. This choice is consistent with the focus of this study, which seeks to establish stronger controls on the security of transmitted data in such sensitive environments. The following figure illustrates the architecture of the Internet of Things (IoT) and its corresponding layers :

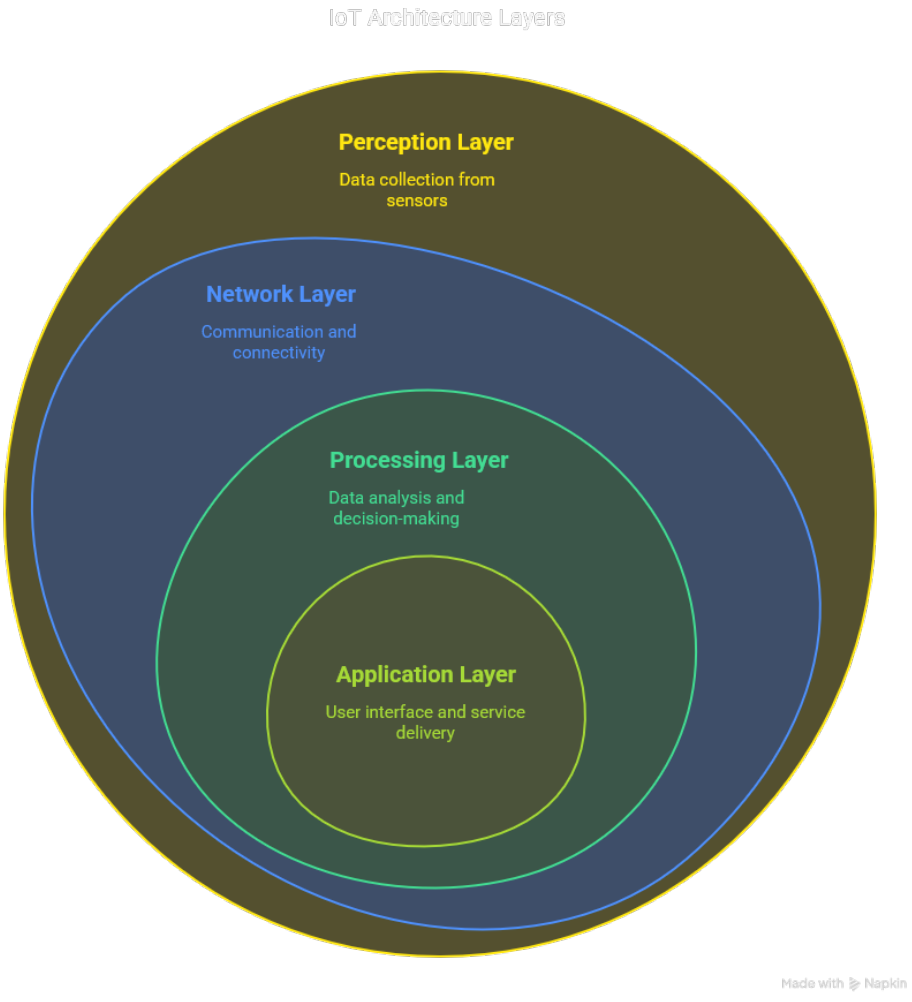

Figure 1.1: Architecture of the Internet of Things (IoT)

1. **Perception Layer:** The perception layer is supposed to gather information from sensors technologies. These normally include wireless sensor networks, video cameras, radio frequency identification, a near field communication device, Global Positioning Systems, or any other technologies.[5]

2. **The Network Layer:** The network layer enables communication between smart devices, servers, and network infrastructure. It also plays a key role in handling and transferring sensor-generated information. [3]

3. **The Processing Layer:** is the heart of the IoT architecture. It collects information from the above layer, stores it, processes it, and carries out analysis. In fact, big data analytics, cloud computing, and database technologies are utilized in providing superior services that sustain the entire IoT system.[3]

4. **The Application Layer:** As the closest interface to the users, the application layer has been one of the recognized layers of the IoT architecture. It provides necessary components to users for control, supervision, and management of IoT services. It has also empowered users to not only analyze the current status of the devices which are connected, but to also forecast future behaviors using analytics.[6]

Having examined the fundamental structure of the Internet of Things, it is equally important to appreciate its expanding influence globally. The importance of IoT goes beyond the framework and how smart solutions are provided by integrating both physical and digital systems. From healthcare, agriculture, IoT technologies, and smart cities, modern life is shaped by these technologies, enabling timely decisions and actions.

In the health care sector, remote patient monitoring is possible with wearable devices enabling timely medical interventions. In agriculture, smart sensors gauge weather conditions alongside soil moisture to enhance irrigation and crop yield Integrating an Internet of Things infrastructure into sensitive and critical areas of work has raised a new set of security concerns. The very nature of IoT devices such as their interconnection, low computational power capabilities, and wide-range deployment in real-life settings make them vulnerable to multiple security threats including denial of service (DoS), data spoofing and interception, unauthorized access, and control over the system.

1.4 Security in IoT

Major sectors have adopted IoT systems at a remarkable speed which has in turn raised serious concerns of security. Cyber threats tend to target IoT devices as they capture, store, and transmit sensitive information. Issues related to the security of an IoT are both technical and operational and are multifaceted in nature. Some essential security issues in IoT include unauthorized access, data theft, malware infections, denial of service attacks, and privacy violations.

Perhaps the most notable challenge is the low processing power, low available memory, and short battery life of most IoT devices, making strong security difficult to achieve. With no one unified security protocol applicable to all IoT devices and platforms, the exposure to these vulnerabilities increases. In places like in the healthcare sector, where devices can be life-critical, these factors are even more concerning.[7]

1.5 The Internet of Medical Things (IoMT)

The interconnection of internet-connected medical devices, including wearable health monitoring, in-hospital equipment, and home healthcare equipment, is the focus of the Internet of Medical Things (IoMT), a niche application of the general Internet of Things (IoT).Security and privacy are of great concern since such devices acquire very personal health data, including blood pressure, body temperature, and heart rate [8]

Apart from this, other literature defines IoMT as the result of the convergence of IoT technology and medical devices to provide real-time monitoring and control for health-care professionals. All medical devices are made to be easily connected under the new paradigm to allow faster and less expensive healthcare services[9].

In the same context, another source defines IoMT as "A subset of the Internet of Things (IoT), the Internet of Healthcare Things is the convergence and integration of sensor data collected by medical devices and mobile technologies, as applied to healthcare. Devices linked to cloud platforms on which captured data is stored and analyzed has come to be known as the Internet of Medical Things . " [10]

The following figure shows how the Internet of Medical Things (IOMT) connects patients , medical applications, and healthcare providers:

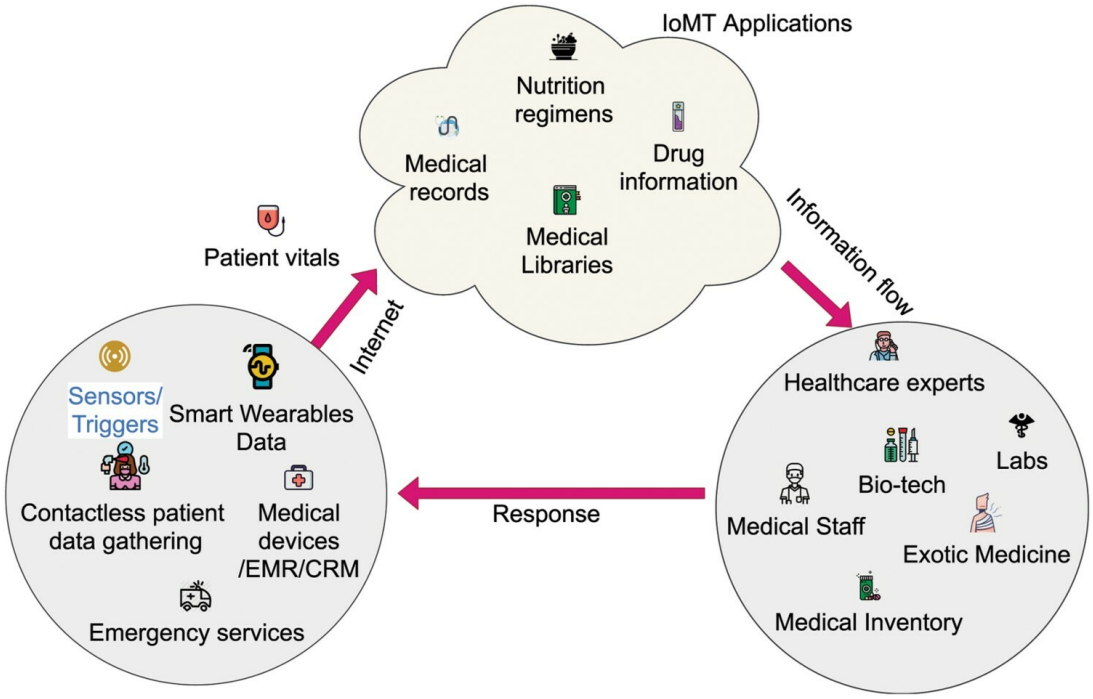

Figure 1.2: Internet of Medical Things (IoMT)

1.6 Architecture of the Internet of Medical Things (IoMT)

The Internet of Medical Things (IoMT) works on integrated layered architecture where each layer has its own specific role towards smart and secure healthcare services. A wide understanding of this architecture is important too because this study aims to identify security vulnerabilities that may threaten such an environment. Some other studies [11], [12],[13] proved that normal IoMT architectures typically consist of these four basic layers. It usually starts with the Perception Layer. This layer forms the vision of the system, where information is gathered from actuality using medical sensors like blood pressure monitors, heart rate monitors, and devices for glucose measurement as well as temperature and oxygen level monitoring [13] . Technologies on this layer used for capturing vital signs accurately include RFID (radio frequency identification), Bluetooth, and GPRS (General Packet Radio Service) technology. Additionally, this layer is nearest to physical reality. Data will then be forwarded to the network layer which will further relay such information from end devices that are central servers or processing centers via 4G/5G networks, Wi-Fi, NB-IOT, and Zig Bee networks. This layer secures data and routes it in a proper way. Then comes the transport layer which manages the organization of the dialogue between devices and its recipient using TCP/IP or UDP protocols, or otherwise the MQTT protocol which plays a very significant role in the Internet of Things. This is intended for resource-constrained devices such as sensors and wearable medical devices since it is based on a publish-subscribe model (where a sender publishes messages and data to the subscriber recipient is received through an intermediary known as ”broker” dependent on either local networks or Internet to relay information). In addition, this layer makes sure that data delivery will be completed successfully even under improper conditions of communications. Finally, data will be available at the Application layer, which represents an interface with users, physicians, or medical systems. This includes mobile health monitoring applications as well as remote diagnostic systems and health data platforms in hospitals. At this layer, the data is analyzed, visualized, and used to make immediate medical decisions.The following figure illustrates the architecture of the Internet of Medical Things (IoMT):

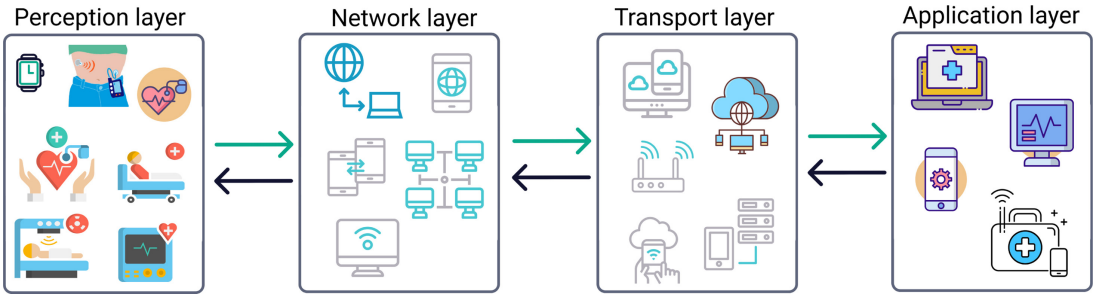

Figure 1.3: Architecture of the Internet of Medical Things (IoMT) [13]

1.7 Types of IoMT Devices:

An Internet of Medical Things (IOMT) systems comprise a range of automated devices that are enfocused on health data collection,patient tracking and supporting medical decisions.these devices can futher be classified as [14]:

1.7.1 Implantable Medical Devices :

Implantable medical devices, or IMDs, are surgically implanted into a patient's body to help treat certain diseases, track essential bodily systems, or deliver medication to specific areas of the body.They include cardiovascular, neurological, orthopedic, hearing, and medication delivery devices, and their types vary according to their intended use. Usually made of biocompatible materials and driven by internal power sources, these devices need to be continuously monitored to guarantee their safety.

1.7.2 Wearable Medical Devices :

Wearable Health Tech. The new sensors and wireless comms tricks used in things like smart fitness trackers, always-on glucose monitors, and wearable ECGs help with the auto checking and capturing of many health signs. People can now join in their own care with active real-time health watching. This upfront way towards healthcare helps in finding out issues early and stopping diseases along with other bad health problems.

1.7.3 Remote Patient Monitoring Systems:

RPM systems are designed to observe specific medical parameters of a patient and automatically capture relevant information. A number of sensors are used to collect information from the patient and transmit it to medical specialists for continuous monitoring and timely intervention. Of great significance is the fact that through so called RPM systems, patients are provided with secure mobile applications or web portals that allow them to view their health data, track their health status, access educational materials, and communicate with healthcare practitioners remotely. The operation of RPMs is based on wireless technologies such as Bluetooth, Wi-Fi, and mobile telephony.

1.7.4 Hospital Networked Devices:

HNDs are the tools and equipment that doctors use in hospitals and other clinical settings to diagnose, treat, and keep an eye on patients. Typical HNDs feature laboratory equipment for testing blood and other bodily fluids, as well as diagnostic and imaging equipment including X-ray machines, computed tomography (CT), magnetic resonance imaging (MRI), and ultrasound scanners. monitoring tools, such as pulse oximeters , ECG machines, and blood pressure monitors .Infusion pumps, which are utilized to provide drugs, liquids, and other materials straight into a patient's bloodstream .Equipment for rehabilitation, such as physical therapy, exercise, braces, splints, etc. Equipment for patient placement, such as hospital beds and stretchers.

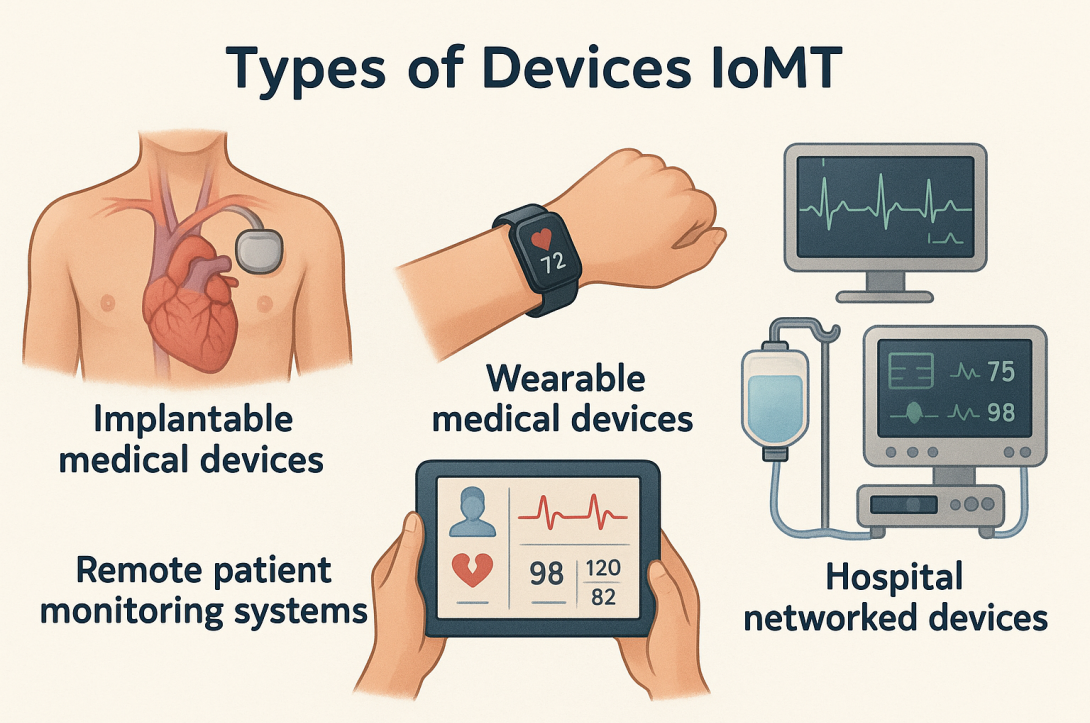

Figure 1.4: Types of Devices in IoMT

1.8 Applications of IoMT in Healthcare:

1.8.1 Smart Hospitals:

To enhance patient care and streamline healthcare operations, smart hospitals integrate IoMT equipment and technology. These can include remote monitoring tools, automated drug dispensers, and other technology that support medical professionals in providing more effective and efficient treatment.[15]

1.8.2 Remote Patient Monitoring:

Using medical equipment to keep an eye on patients away from conventional healthcare facilities, including their homes, is known as remote patient monitoring, or RPM. Blood pressure monitors, blood glucose meters, and other vital sign monitors that may transmit data to medical professionals for examination are examples of RPM devices.[15]

1.8.3 Medical Imaging:

X-rays and MRIs are two examples of medical imaging that can be enhanced by IoMT. By connecting to the internet, medical imaging equipment may send pictures to medical professionals for evaluation and diagnosis.[15]

1.8.4 Fitness Tracking:

The widespread use of smartphones has led to the development of many wearable devices equipped with sensors to monitor physical activities such as the number of steps taken, calories burned, and exercise duration.[16]

1.8.5 Hospital Management and Service Team:

Hospitals have numerous resources in the form of construction, capital, housing, equipment, infrastructure, data, and information systems. Compared to conventional approaches, IoMT Solutions IoMT solutions provide services at a higher effectiveness and efficiency regarding the management of the assets. Autonomous Robots, for example, perform a great number of routine tasks, including food, medication, and medical supplies delivery; custodial chores such as biological waste removal; and disinfection processes. This would also decrease the overall depletion of hospital resources which would cut down the cost of healthcare services ultimately.[16]

1.8.6 Doctors and Nurses :

Patients can access the Nao Robot, which explains illness risks and provides lifestyle suggestions [17] . Such interaction automates repetitive yet routine pre and post consultation tasks disengaging with robots augment the physician's role with advanced assistants. This shift increases the overall effectiveness and efficiency of medical services by allowing physicians to shift their focus towards more specialized work.[16]
The following figure shows applications of the Internet of Medical Things in Healthcare:

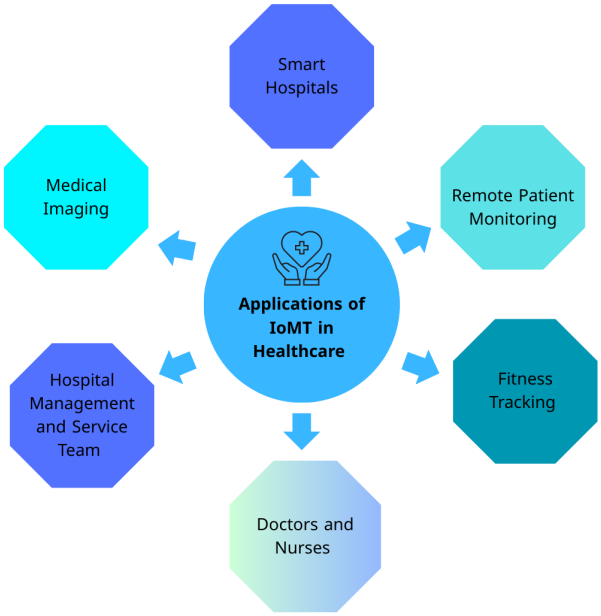

Figure 1.5: Applications of the Internet of Medical Things in Healthcare

1.9 Cybersecurity Threats in the Internet of Medical Things (IoMT)

Internet of Medical Things (IoMT) devices, such as implanted sensors and wearable technology, are essential components of the smart healthcare infrastructure, utilized to collect patients' vital data and enhance the quality of healthcare services. Despite their phenomenal effect, they also represent a fertile ground for cyber attacks that threaten patient privacy and physical well being. Research indicates that the heterogeneous and dynamic nature of IoMT devices increases the possibility of attacks that exploit vulnerabilities because these devices are often prone to malicious connection to open wireless networks and are located in many places such as homes and hospitals which greatly complicates their management and protection [18] [19] .
The most serious issues in the field of security for IoMT systems include the following:

- **Botnet attacks:** These can be used to hijack a medical device network and facilitate DDoS attacks.
- **Impersonation and identity theft:** These assist attackers to issue commands which cause harm to life support machines like drug infusion pumps and pacemakers.
- **Phishing:** These are among the irksome attacks directed at personnel for purposes of manipulating systems and granting unauthorized access.

What makes defending the IoMT environment more complex are the increasing cyber attacks and the very specific characteristics that IoMT has when compared to a traditional IoT environment. IoMT devices are sensitive in nature as they deal with life threatening information like a patient's health and medical history, making any form of breach ethically as well as legally alarming, especially when such data is often basis for necessary

clinical decisions.
Moreover, the dependence on some of these devices such as infusion pumps and cardiac monitors makes them vulnerable, meaning that any form of flaw leads to possible endangerment of a patient's life and critical health complications or death. Volume and computational constraints on most of the devices precludes the application of sophisticated security measures typical to advanced systems [19].
Aside from that, the fact that IoMT devices are deployed in homes, healthcare facilities, and transportation devices adds to the problem. This diversity creates and enforces difficulties in maintaining a single and effective security framework. It also complicates software updates, security response strategies across all network nodes, and security monitoring and coordination at all system nodes.
Considering the obstacles outlined above, the construction of new advanced level security measures to augment the integration of IoMT devices into an efficient, secure healthcare system has been marked as an urgent priority [18] [19]. Thus, the increasing use of innovations in artificial intelligence, particularly deep learning and federated learning, dictates a change in the design of intelligent security systems that are capable of evolving with new threats and provide anticipatory defense without degradation of the functioning of medical devices.

1.10 Cyber Threats in the Internet of Medical Things (IoMT) According to the CIC-BCCC-NRC-IoMT-2024 Dataset

The number of cyber threats targeting Internet of Medical Things (IoMT) devices is increasing along with their use in healthcare. These devices become susceptible to a variety of attacks that jeopardize patient privacy and system security, including DoS, DDoS, spoofing, MQTT-based, and reconnaissance assaults. For the Internet of Medical Things (IoMT) ecosystem to be properly secured, it is essential to comprehend various attack techniques:

1.10.1 Denial of Service (DoS) attacks:

Within the realm of digital threats, Denial of Service (DoS) attacks rank among the most dangerous, particularly while examining the risks posed to Internet of Things (IoT) systems; which rely heavily on availability and reliability. DoS attacks target restricting access to specific servers or devices leading to service interruption which affects the availability aspect of information security. Such attacks are problematic because they are relatively easy to execute [20] . In the case of medical IoT devices, the effects can be catastrophic by disabling controls or patients' available medical resources. Even more, attackers can prevent legitimate users from using healthcare applications [21] , which compromises essential elements of health care services with regards to patient data and information privacy. The following illustrates an figure 6 of a DOS attack.

Figure 1.6: A DoS Attack

1.10.2 Distributed Denial of Service (DDoS) attacks:

is a type of cyberattack assaults are those that are launched from several hacked nodes. The most popular DoS attack uses massive traffic flooding to use up bandwidth, network resources, target CPU time, etc. SYN flood, DNS flood, Ping flood, UDP flood, ICMP broadcast, and others are some of the most prevalent DoS attacks.[22] The following illustrates an figure 7 of a DDOS attack.

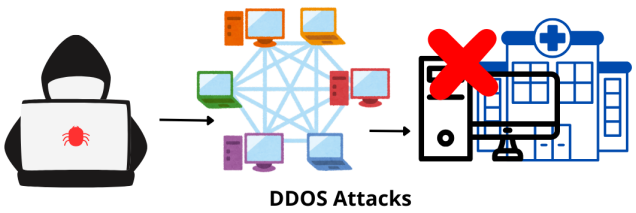

Figure 1.7: A DDoS Attack

The more prevalent forms of DoS and DDoS attacks focus on the TCP, UDP, and ICMP protocols. Such attacks involve the UDP Flood, TCP Flood, and ICMP Flood. These can be performed individually (DoS) or collectively with many devices (DDoS). The objective of such attacks is either to consume resources or bandwidth and interrupt services fully or partially.

- **UDP flood:** Here, a huge number of UDP packets are dispatched to either random or specific ports on the target. In this attack, arbitrary target ports are flooded with diverse UDP packets causing the host to constantly search for the application listening port and respond back with ICMP Destination Unreachable packets. As a result of this operation, target host resources become unreachable.[22]
- **TCP flood:** It floods the victim with a number of TCP connection requests using false source IP addresses and does not complete the three-way handshake communication, leading to accumulation of half-connections that eventually disrupt processing resources at the server and make it unable to respond to requests.[23]
- **ICMP Flood (Ping Flood):** It is a flooding attack. ICMP is short for Internet Control Message and is primarily used to query any server and report errors. The attacker sends a large number of ICMP Echo Request packets to the targeted system, which attempts to process each incoming ICMP request, causing it to consume CPU and network resources. [24]

1.10.3 Reconnaissance Attacks:

This form of attack is designed to gather details regarding the information technology infrastructure before executing an actual assault. These assaults can be categorized as pre-invasion reconnaissance operations, Employing mapping and scanning processes, these cyber attacks are capable of detecting connected devices, open ports, security gaps as well as ongoing operations [25]. An assailant can formulate a routing schematic of the intended

network and its corresponding IP address system, useful for subsequent strikes.[26]
The following are some techniques used by reconnaissance attacks to gather information :

- **Port Scanning:** discovers open ports on a computer by methodically scanning its ports, which is how all information enters and exits a computer. Attackers can determine which services are accessible and where an attack could occur by using port scanning. The fundamental idea behind port scanning is to extract and examine data from the opened port.[27]
- **Recon Ping Sweep:** gathering all active hosts' Internet Protocol (IP) addresses using the Packet INternet Grouper (PING) tool. This kind of attack looks for connected and active hosts by pinging every IP address in the network's address range.[25]
- **Recon OS Scan:** This attack probes the remote machine using programs like nmap [11]. could be identify the victim's protocol implementation and deduce the operating system it is running on by analyzing the victim's answers to both legitimate and invalid TCP packets. Such a method might assist the attacker in identifying victims running outdated OS versions where known vulnerabilities can be exploited, even if it might not provide definitive findings on the precise OS version (because several OS versions may share the same implementation of TCP).[25]
- **Recon Vulnerability Scan:** Looking at ports that respond across a range of addresses can give adversaries tremendous intelligence. The attacker's objectives, such as installing malware, accessing private information, or even causing the network to crash due to an overload of electronic traffic, can then be advanced by using that information to investigate known vulnerabilities linked to the responding services and applications. [28]

1.10.4 MQTT Attacks:

An application layer protocol called Message Queue Telemetry Transport (MQTT) facilitates communication via the transport layer protocol TCP/IP. The protocol's characteristics, including its lightweight nature, low bandwidth requirements, ease of implementation, and high degree of dependability, make it widely employed in IoT systems with limited resources. Its foundation is a publish/subscribe communication mechanism, in which the broker is in charge of informing MQTT clients of messages.[29] However, its simple design makes it vulnerable to several cyberattacks, especially in environments lacking strong security measures.

- **MQTT PUBLISH Attack:** In an MQTT Publish flood attack, whether it is a single-source DoS attack or a distributed denial of service (DDoS) attack from multiple compromised devices (botnets), an MQTT Publish flood occurs after an attacker connects to the MQTT broker during data exchange and sends huge Publish packets, causing the latter to disrupt the service and affect other subscribers in the system [30].
- **MQTT CONNECT Attack:** Sending a lot of request connect packets to overload the target during the MQTT broker connection procedure is known as a MQTT

CONNECT flood. The goal is to cause a MQTT broker's service to be interrupted, rendering the MQTT network unusable. The attack is carried out following a three-way handshake because MQTT is an application layer protocol.[30]

- **Malformed Attack:** Certain types of single-packet assaults referred to as malformed attacks, can target and exploit the MQTT protocol [31]. In order to cause exceptions inside the identified service, this kind of attack attempts to generate and deliver a large number of erroneous, corrupted packets to the broker [32].

1.10.5 SPOOFING Attacks:

Spoofing is defined as assuming someone's identity, be it a person or a computer, using deceptive information such as an IP address, URL, or email identity. Spoofing in computing can take many forms, however, each and every one involves deception or misrepresentation of information.[33] One of the spoofing attacks is also the Man-in-the-Middle(MITM) attack.

- **Man-in-the-Middle (MITM) attack:** Attacks by a man-in-the-middle existed long before computers were invented. A malicious postman who opens letters and reads or alters them before delivering them to their intended recipients encapsulates the fundamentals of MITM attacks. Most Internet applications try to provide services securely using encrypted links provided by SSL (Secure Sockets Layer) / TLS (Transport Layer Security) protocols at the application layer. Even though SSL/TLS form mutual trust relationships, it is almost always used in scenarios where only one party asserts the connection because of administrative overheads. This is an example of a gap that can be exploited to mount an attack. Employing multiple methods to intercept communication between two endpoints defines the man-in-the-middle attack. The assaulting side may seize control of a proxy by breaking their victims' line.[34] In the scope of the Internet of Medical Things (IoMT), a man-in-the-middle (MITM) attack occurs when an unduly censoring and filtering person extracts sensitive information to and from medical devices and the server. This poses a greater threat because vital information may be lost, altered, or stolen. The following figure illustrates the concept of a man-in-the-middle (MITM) attack, where an attacker eavesdrops and intercepts the communication between two parties without their knowledge:

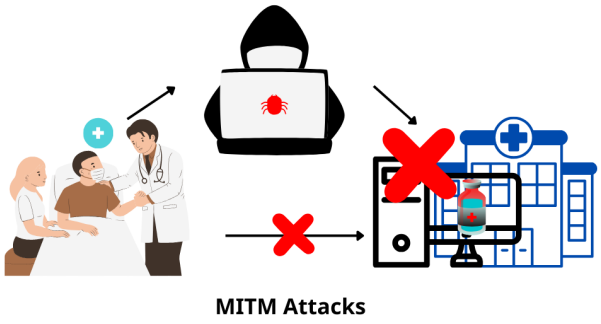

Figure 1.8: A MITM Attack

1.11 Basic Security Requirements for the Internet of Medical Things (IoMT)

According to several studies [11] [13] [35] [36], the CIA Triad is considered one of the fundamental principles underlying data protection in the digital world, particularly IOMT. This triangle forms the framework for ensuring information confidentiality, integrity, and availability. Each of these is described below:

- **Confidentiality:** Confidentiality protects sensitive and private patient information and the complete confidentiality of their treatment from unauthorized parties [13], as it can only be accessed by those who have the right to view it [35].
- **Integrity:** Protects data from unauthorized modifications or changes. [36] This means that medical information is not destroyed or deleted during storage or IOMT data transfer without proper authorization. [13]
- **Availability:**Ensures that legitimate users have access to the correct data, so that appropriate users or nodes can access medical resources and devices quickly, reliably, and at any time [11]. This is essential for immediate response to medical emergencies.

1.12 Threats to Confidentiality, Integrity, and Availability in Network Systems

Below is the impact of different types of attacks on the cybersecurity triangle (CIA) in IoMT environments:

- **DOS/DDOS:** Medical IoT systems are vulnerable to DOS/DDOS attacks, which overburden them with excessive traffic, preventing medical equipment or services from operating and making them unavailable to authorized users.[36] As a result, the main impact of this type of attack is availability.
- **Man-in-the-Middle (MITM) attacks:**These types of attacks enable unwanted third parties to access and change conversations between devices and servers; intruding confidentiality through snooping and integrity by changing data that is sent.[37]
- **Reconnaissance:** A study [38] showed that security weaknesses related to medical devices may result in the breach of confidential information concerning the devices and networks which is deemed an infringement of confidentiality in the cybersecurity triad. From this angle, one can deduce that reconnaissance attacks that seek to exploit security lapses for information collection pose a direct threat to confidentiality.

Conclusion

Smart and connected medical devices are revolutionizing healthcare through the Internet of Medical Things (IoMT). However, patient safety and data privacy can be undermined by dire cybersecurity threats such as DoS attacks, spoofing, and data breaches. Traditional security methods are no longer enough to address these challenges. For this reason, AI is coming to the forefront to detect and mitigate threats in real-time. In this chapter, we have focused on the anatomy of IoMT and its possible threats. The subsequent chapter will shift focus to the role of AI, particularly deep learning and federated learning, in enhancing IoMT security.

Chapter 2

Theoretical Background

2.1 Introduction artificial intelligence

AI refers to the science and engineering of making intelligent machines, through algorithms or a set of rules, which the machine follows to mimic human cognitive functions, such as learning and problem solving. AI systems have the potential to anticipate problems or deal with issues as they come up and, as such, operate in an intentional, intelligent and adaptive manner[39].

AI algorithms have significantly impacted the general performance of Intrusion Detection methods by enhancing security and preserving privacy. Because of their capability to effectively adapt to dynamic network environments, handle large volumes of data,and provide reliable outcomes.Machine Learning and Deep Learning represent a medium to provide a solution for anomaly detection systems. ML is a subfield of AI (figure 1) that automatically seeks to learn meaningful patterns or relationships from observations. On the other hand, DL is a subdomain of ML and the functioning of the human brain inspires it to process and learn data with several levels of abstraction[40].

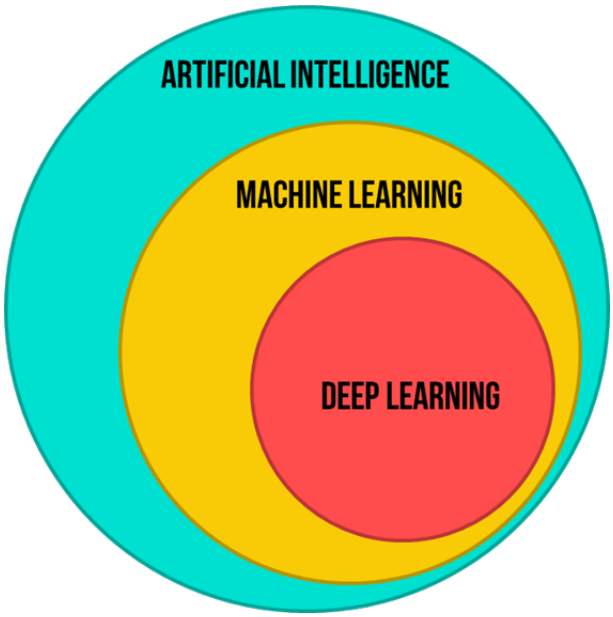

Figure 2.1: Hierarchy of AI

2.2 Deep learning

Deep learning approaches have grown dramatically in terms of performance in a wide range of applications considering security technologies, particularly, as an excellent solution for uncovering complex architecture in high dimensional data. Thus, DL techniques can play a key role in building intelligent data-driven systems according to today's needs, because of their excellent learning capabilities from historical data. Consequently, DL can change the world as well as humans' everyday life through its automation power and learning from experience. DL technology is therefore relevant to artificial intelligence.[41]

Deep learning (DL) plays a pivotal role in the Internet of Medical Things (IoMT) by enhancing diagnostic precision, enabling real time health monitoring, and facilitating

personalized treatment strategies. DL models are widely employed to analyze medical imaging and sensor data from IoMT devices to detect diseases, assess health risks, and automate clinical interventions such as insulin delivery. Moreover, IoMT technologies contribute significantly to elderly care, particularly in home environments, by integrating smart sensors that can identify intrusions, fires, water leaks, and other hazards—promptly triggering alerts to ensure swift and effective responses.[42]

The most important medical device and IoMT standard is IEEE 11073 , a standard for network communication between medical devices and healthcare IT systems. According to device deployment with DL, we match the above standards with DL applications to demonstrate how DL can be utilized in the interoperability area. Using wearable devices as an example, DL models such as CNN, RNN, and LSTM are used to detect abnormal activities or predict the potential risks. [42]

Despite its promising potential, the integration of deep learning (DL) in the Internet of Medical Things (IoMT) presents several new challenges. Key concerns include establishing robust data privacy standards , resolving compatibility issues among diverse systems , and addressing the limitations of central processing capabilities within edge devices [41]. Another critical challenge lies in ensuring that DL models are secure, consistently available, and adaptable to different clinical environments even as they undergo updates .

2.2.1 Convolutional Neural Network (CNN)

Convolutional Neural Networks (CNNs) are a specialized type of artificial neural network (ANN) designed to efficiently extract and learn hierarchical features from grid-structured data, making them particularly effective for visual inputs such as images and videos. Due to their strong capability in identifying spatial patterns, CNNs have become a cornerstone in computer vision tasks. A typical CNN architecture comprises several key components, including an input layer, one or more convolutional layers for feature extraction, pooling layers for dimensionality reduction, and fully connected layers that perform the final classification or prediction tasks.[43]

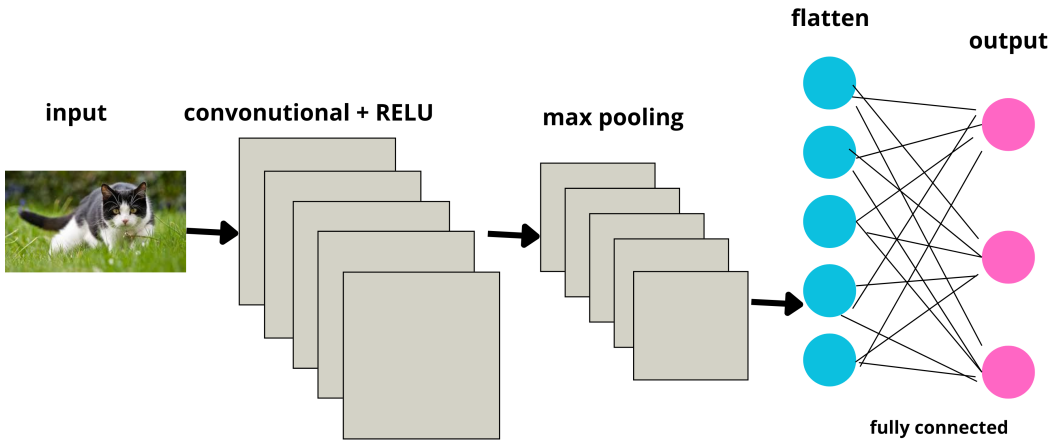

Figure 2.2: Typical CNN architecture showing convolution, pooling, flattening, and dense layers.

2.2.1.1 Input layer

The input layer is the acceptor of a CNN, in which raw information such as images or sequences are fed into the network. For image-based CNNs, inputs are usually 2D arrays (e.g., height × width × channels), depending on whether the image is grayscale (1 channel) or color (3 channels for RGB). It performs no computation; it merely presents the data in a structured format for processing.

2.2.1.2 Convolutional Networks

The convolutional layer is the fundamental component of a CNN. It employs a group of learnable filters (also known as kernels) which slide over the input data, calculating a dot product of the filter with local regions of the input. Every filter identifies some features such as edges, textures, or shapes. The output is a feature map which provides the spatial presence of these features. A number of different filters can be used to extract different features from the same input, resulting in multiple feature maps.

2.2.1.3 Activation Layer (ReLU)

After convolution, an activation function is used to introduce non-linearity to the model. The most commonly used activation function in CNNs is the Rectified Linear Unit (ReLU), which maps all negative values of the feature map to zero. This allows the network to learn complex patterns by clipping the linearity of the convolution operation. Leaky ReLU or ELU can also be used depending on the application.

2.2.1.4 Pooling Layer

Pooling layers reduce the spatial size of feature maps, which reduces parameters and computation needed, and helps to render the features translation invariant to small movements in the input as well. Max Pooling is most commonly used, with the maximum value in a given window (e.g., 2×2) of the feature map. Average Pooling is another, with the

mean of the values in the window. Pooling helps the network generalize better and also reduces overfitting.

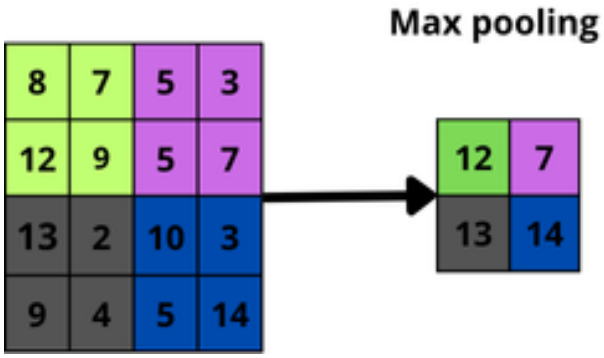

Figure 2.3: Pooling layer

2.2.1.5 Flatten layer

Before plugging the output into fully connected layers, multi-dimensional feature maps have to be flattened into 1D vector. This process is carried out by the flatten layer. It converts the 2D or 3D output of convolutional/pooling layers to a 1D vector to be used as an input by the dense layer.

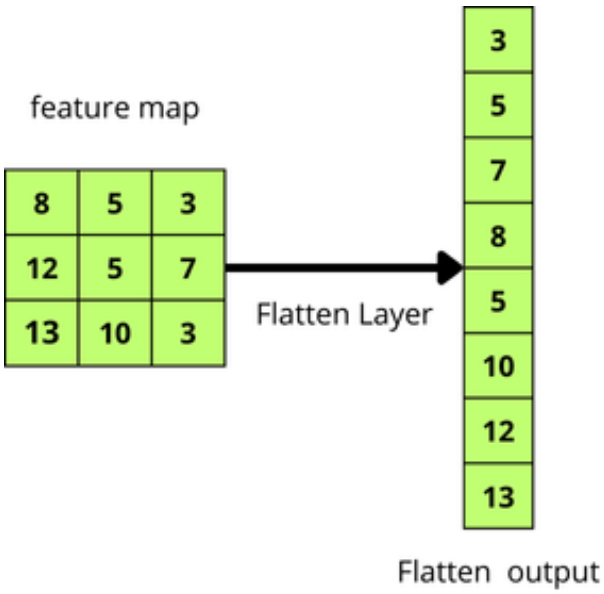

Figure 2.4: Flatten Layer

2.2.1.6 Fully Connected (Dense) Layer

Fully connected layers are traditional neural network layers where every neuron is connected to every neuron in the previous layer. These layers learn to combine the high-level features extracted by the convolutional and pooling layers into class scores or predictions. Dense layers are typically found toward the end of the CNN and are crucial for tasks like classification or regression

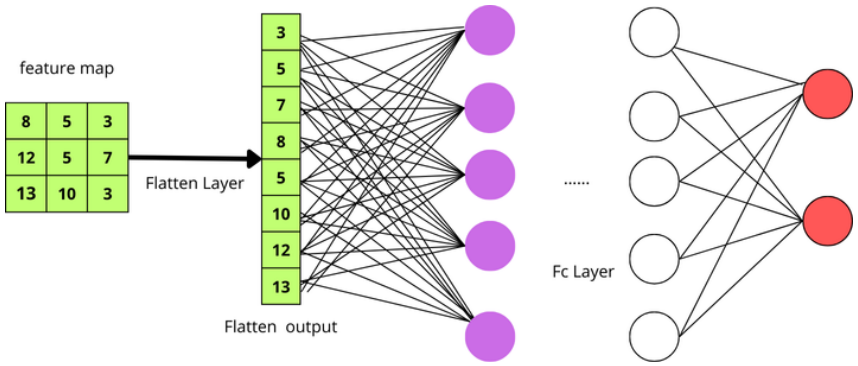

Figure 2.5: Fully Connected Layer

2.2.1.7 Output layer

The final layer of a CNN is the output layer, responsible for producing actual predictions. In classification, it is usually a fully connected softmax layer (for multi-class cases) or a sigmoid layer (for binary class cases). The output layer transforms the internal representation of the network into probabilities or output scores that correspond to target classes.

2.2.2 Recurrent Neural Networks

Recurrent Neural Networks (RNNs) are a type of neural network made to work with sequences, like sentences or time-series data. They keep a "memory" of what they have seen so far by passing a hidden state from one step to the next. This lets RNNs learn patterns that depend on earlier inputs, such as predicting the next word in a sentence or forecasting stock prices. However, basic RNNs can struggle to remember information from far back in the sequence, which led to improved versions like LSTM and GRU that handle long-term dependencies better.[44][45]

2.2.3 Long short-Term Memory

Long Short-Term Memory (LSTM) networks are one of the most effective solutions to the long-standing issue of exploding and vanishing gradients in recurrent neural networks (RNNs). Their introduction marked a significant advancement in the field of machine learning, enabling stable training over long sequences. LSTMs have been widely adopted in real-world applications, enhancing systems such as Google's speech recognition, Amazon Alexa's responses, and powering over 4 billion translations per day on Facebook as of 2017[46].

2.2.3.1 LSTM architecture

The basic computational unit in the hidden layer of a Long Short-Term Memory (LSTM) network is known as a memory block. Each memory block contains one or more memory cells, and is equipped with a pair of adaptive multiplicative gating units—the input gate and output gate. These gates regulate the flow of information into and out of the memory cells.

At the core of each memory cell lies a Constant Error Carousel (CEC), a self-connected linear unit whose activation is referred to as the cell state. The CEC is critical in addressing the vanishing gradient problem, as it allows error signals to persist over time without exploding or vanishing. In the absence of new inputs or error signals, the local error backflow through the CEC remains constant, thereby preserving long-term dependencies during training.

The CEC is shielded from unnecessary updates by the input and output gates. When these gates are closed, near-zero activations and irrelevant or noisy inputs are prevented from altering the cell state. Consequently, the cell's internal state remains stable and does not affect the rest of the network.[47]

This mechanism ensures that relevant information is retained over long sequences, enabling LSTM networks to model long-term dependencies effectively.

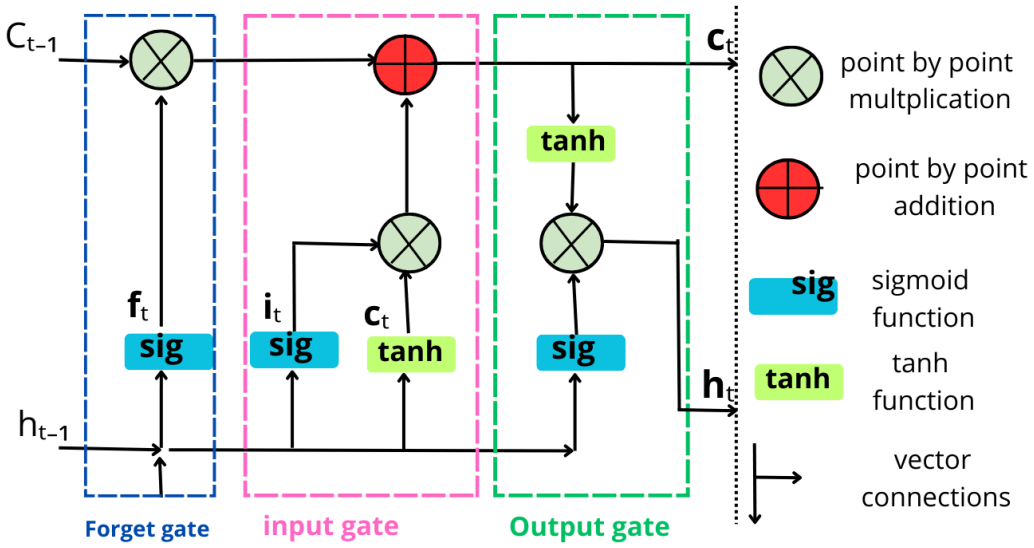

Figure 2.6: Long Short Term Memory.

2.2.3.2 Forget Gate in LSTM

The forget gate determines which information from the previous cell state C_{t-1} should be retained or discarded. It takes the current input $\mathbf{x}_t$ and the previous hidden state $\mathbf{h}_{t-1}$, applies learned weight matrices and bias, and passes the result through a sigmoid activation to produce a gating vector $\mathbf{f}_t \in [0,1]^d$:

$$\mathbf{f}_t = \sigma\big(\mathbf{W}_f \mathbf{x}_t + \mathbf{U}_f \mathbf{h}_{t-1} + \mathbf{b}_f\big), \tag{2.1}$$

where

- $\sigma(\cdot)$ is the sigmoid activation function,
- $\mathbf{W}_f$ and $\mathbf{U}_f$ are the weight matrices for the input and hidden state, respectively,
- $\mathbf{b}_f$ is the bias vector.

Each element of $\mathbf{f}_t$ is a value between 0 and 1. An element close to 0 means the corresponding component of the previous cell state C_{t-1} is largely forgotten, while an element close to 1 means it is largely retained. The updated cell state $\mathbf{C}_t$ incorporates this gating as follows:

$$\mathbf{C}_t = \mathbf{f}_t \odot \mathbf{C}_{t-1} + \cdots , \tag{2.2}$$

where $\odot$ denotes element-wise multiplication.[48]

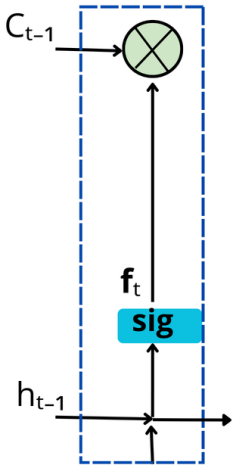

Figure 2.7: Forget Gate

2.2.3.3 Input Gate

The input gate controls which parts of the new information are written into the cell state. It first computes a gating vector $\mathbf{i}_t$ using a sigmoid activation, and then creates a candidate update vector $\tilde{\boldsymbol{C}}_t$ using a tanh activation. Finally, these are combined to determine what new information to add:

$$\mathbf{i}_t = \sigma\big(\mathbf{W}_i \mathbf{x}_t + \mathbf{U}_i \mathbf{h}_{t-1} + \mathbf{b}_i\big), \tag{2.3}$$

$$\tilde{\boldsymbol{C}}_t = \tanh\big(\mathbf{W}_c \mathbf{x}_t + \mathbf{U}_c \mathbf{h}_{t-1} + \mathbf{b}_c\big), \tag{2.4}$$

$$\mathbf{C}_t \leftarrow \mathbf{C}_t + \mathbf{i}_t \odot \tilde{\boldsymbol{C}}_t. \tag{2.5}$$

Here:

- $\sigma(\cdot)$ is the sigmoid activation function, filtering inputs between 0 and 1 [49].
- $\tanh(\cdot)$ produces candidate values between -1 and $+1$.
- $\mathbf{W}_i, \mathbf{U}_i, \mathbf{b}_i$ are the weights and bias for the input gate.
- $\mathbf{W}_c, \mathbf{U}_c, \mathbf{b}_c$ are the weights and bias for the candidate update.
- $\odot$ denotes element-wise multiplication.

.[48]

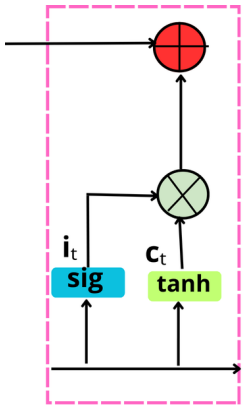

Figure 2.8: Input Gate

2.2.3.4 Output Gate

The output gate is responsible for determining which information from the current cell state should be output at each time step. To do this, the cell state is first passed through a tanh activation function to create a candidate output vector. Simultaneously, a sigmoid function processes the current input x_t and the previous hidden state h_{t-1}, generating a gating vector that decides what portion of the candidate output should be retained. Finally, these two vectors are element-wise multiplied to produce the final output h_t, which is then passed to the next LSTM cell and potentially to the output layer.

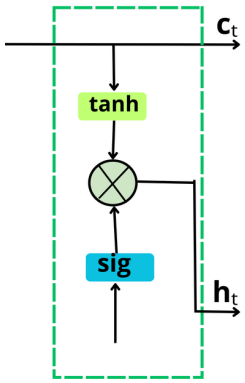

Figure 2.9: Output Gate

2.3 Evaluation Metric

Since the last decade, deep learning algorithms have been the number one choice for solving complex computer vision problems. Performance evaluation metrics play a critical role in assessing the effectiveness of a machine learning model. In classification tasks, particularly with imbalanced datasets, relying solely on accuracy can be deceptive. Metrics such as precision, recall, and the F1 score provide a more comprehensive understanding of model performance.[46]

Performance Evaluation Metrics

The experimental results of the proposed approaches are evaluated using four key performance metrics: **accuracy**, **precision**, **recall**, and **F1-score**. These metrics are computed based on the fundamental classification outcomes: **True Positive (TP)**, **True Negative (TN)**, **False Positive (FP)**, and **False Negative (FN)**. These terms are defined as follows:

- **True Positive (TP)**: The model correctly predicts a positive class.
- **True Negative (TN)**: The model correctly predicts a negative class.
- **False Positive (FP)**: The model incorrectly predicts a positive class when the actual class is negative.
- **False Negative (FN)**: The model incorrectly predicts a negative class when the actual class is positive.

These components form the basis for calculating the performance metrics, enabling a comprehensive evaluation of model effectiveness, particularly in the context of imbalanced datasets.

Accuracy

is defined as the proportion of correctly predicted samples relative to the total number of evaluation samples. It reflects the heuristic that a more effective model should yield a higher number of correct predictions. However, this assumption can lead to the so called "accuracy paradox," where a model that simply predicts the majority class without learning meaningful patterns may achieve high accuracy, thus overestimating its actual generalization capability.[46]

The formula for calculating accuracy is as follows:

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN} \tag{2.6}$$

Precision

Measures the accuracy of positive predictions. It answers the question, "Of all the items the model labeled as positive, how many were actually positive?"

$$\text{Precision} = \frac{TP}{TP + FP} \tag{2.7}$$

F1-Score

In many cases, data scientists and machine learning engineers aim to optimize both precision and recall simultaneously. However, improving one often comes at the expense of the other. To address this trade-off, the F1 score is used a metric that represents the harmonic mean of precision and recall. It provides a single score that balances the two, especially useful in scenarios involving imbalanced datasets.[50]

The formula for the F1 score is given by:

$$\text{F1 Score} = 2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}} \tag{2.8}$$

A higher F1 score indicates greater predictive power of the classification model. An F1 score approaching 1 signifies near-perfect precision and recall, reflecting a highly effective model. Conversely, an F1 score close to 0 suggests poor performance, indicating that the model struggles to accurately identify positive instances or maintain balance between precision and recall.

Recall

Recall, also known as the true positive rate (TPR), is the percentage of data samples that a machine learning model correctly identifies as belonging to a class of interest the "positive class" out of the total samples for that class.

$$\text{Recall} = \frac{TP}{TP + FN} \tag{2.9}$$

2.4 conclusion

In this chapter, we reviewed the foundational principles of artificial intelligence and deep learning, exploring how convolutional neural networks excel at extracting spatial hierarchies from data and how long short-term memory models capture temporal dependencies in sequential tasks. We saw that CNNs have revolutionized image based applications through layered feature learning, while LSTMs provide robust solutions for time series forecasting, language modeling, and beyond. Together, these architectures illustrate the power and versatility of modern neural networks in solving complex, high dimensional problems. In the next chapter, we will build on these insights by introducing federated learning a decentralized paradigm that promises to extend AI's reach into privacy sensitive and resource constrained environments by enabling collaborative model training without sharing raw data.

Chapter 3

Federated Learning survey

3.1 Introduction

In today's world of big data, the biggest concern is not how much data we have, but how to keep it safe and private. People worry a lot about their personal information being leaked. Laws like GDPR in Europe and China's CyberSecurity Law now give people more control over their data. For example, companies must ask for permission before using someone's data to train AI models, and users can ask to have their data deleted.[51] However, these laws also make it harder to collect data for training machine learning (ML) models in the traditional, centralized way. In centralized training, all data is sent to one server, which can lead to privacy risks. Federated Learning (FL) is a new way to train machine learning models using data from many different sources without moving the data from where it is stored. Instead of sending data to one place, FL allows each data owner to train the model locally and only share the learned updates.[52]

3.2 Definition

Federated Learning (FL) helps different groups build one shared AI model without moving their private data to a central location. Instead, the data stays where it is, keeping it safe.

At first, FL used a method called distributed SGD to train models on many devices. But this needed each device to send its model updates often to a central server, which caused privacy and communication problems.[53] In 2016, Google improved this by allowing devices to train models locally, without sending raw data to a server. Since then, FL has grown with many new methods and systems to support it.[54]

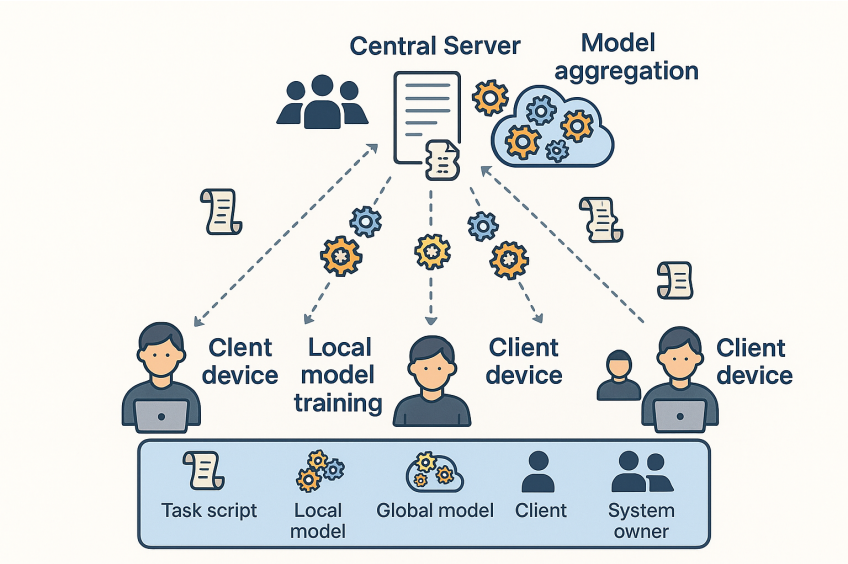

Figure 3.1: federated learning architecture.

3.3 Core Challenges

We next describe three of the core challenges associated with solving the distributed optimization problem. These challenges make the federated setting distinct from other classical problems, such as distributed learning in data center settings or traditional private data analyses. [55]

3.3.1 Privacy Concerns

Federated learning improves privacy by sending model updates (like gradients) instead of raw data, but even those updates can leak sensitive information to the server or outsiders. Techniques such as secure multiparty computation and differential privacy can strengthen privacy, but they usually slow down training or weaken the final model's accuracy. Finding the right balance between keeping data safe and keeping the system fast and accurate remains a major challenge for private federated learning.[56]

3.3.2 Increased Latency

In conventional machine learning algorithms, data from all devices / sensors have to be aggregated in the central cloud for training the machine learning algorithms. This incurs substantial communication cost.[57]

3.3.3 Statistical Heterogeneity

In federated learning, the data on each device is often very different from others for example, people use different words when typing on their phones. Also, some devices have more data than others. This makes it hard to train one global model for everyone. To solve this, researchers use methods such as multitask learning and metalearning, which allow each device to have its own personalized model.[58] These approaches help deal with data differences between devices and improve the overall learning process.

3.4 Application of federated learning

Healthcare

Electronic Health Records (EHRs) are a key source of data for machine learning in healthcare. But if a machine learning model is trained with data from only one hospital, its predictions might be biased. To make the model better and more accurate, it needs more data, which means hospitals would need to share patient records.[59] Since this data is very sensitive, sharing it is not always possible. Federated Learning solves this problem by letting hospitals build a shared model without having to share patient data.

Internet of Things (IoT)

Federated Learning is increasingly used in the IoT sector to enable privacy preserving machine learning across devices without transferring sensitive data to a central server. It has key applications in smart transportation, smart cities, and smart grids helping to improve traffic systems, urban infrastructure, and energy management by training models directly on local data.[53] This decentralized approach ensures data privacy while still enabling powerful AI solutions, making FL a valuable tool as the use of connected devices grows.

Finance

in finance a serious problem called multi-party borrowing happens when someone takes a loan from one bank just to pay back another. This kind of behavior can harm the whole financial system if many people do it. To detect these users without sharing full customer lists between banks A and B, federated learning can be used. With FL, each bank encrypts its user data and only the overlapping users (those borrowing from both banks) are revealed after decryption without exposing the rest of the customers[60]

3.5 Mechanism of federated learning

3.5.1 model initialization

The central server begins by selecting a baseline AI model such as a neural network or linear regression—and chooses a group of participating devices.[61]

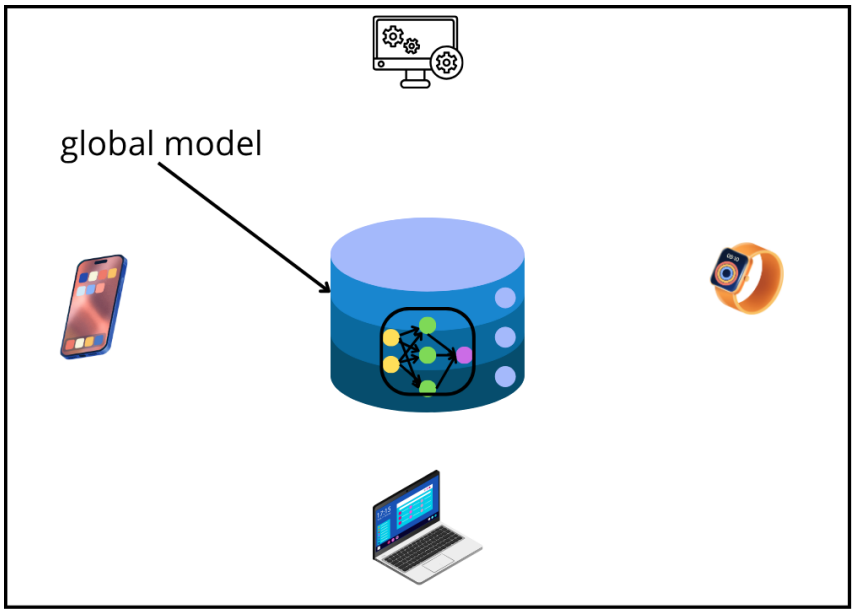

Figure 3.2: Model initialisation

3.5.2 Model Broadcast

Once the server has a new global model, it distributes its parameters back to a selected subset of client nodes such as smartphones or organizational servers—so everyone begins their next round of local training from the same starting point. Rather than sending to every connected device, only selected devices are chosen each round.[62]

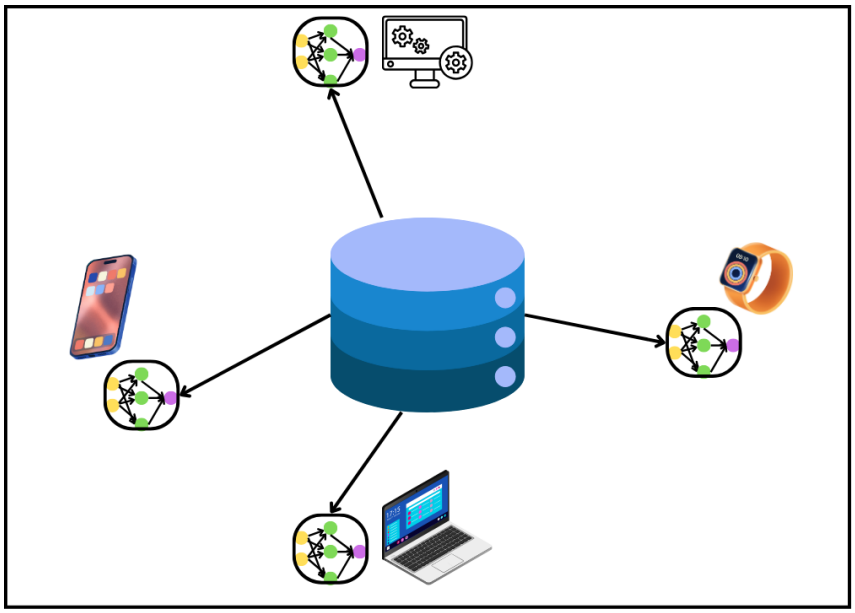

Figure 3.3: model distribution

3.5.3 local model evaluation and training

each client trains a local model when it gets instructions from the central server. This training follows specific settings like how many times to train (epochs) and how fast to

learn (learning rate). Normally, only the model weights are sent between the server and the clients, but in this setup, the training settings (hyperparameters) are also shared.[63]

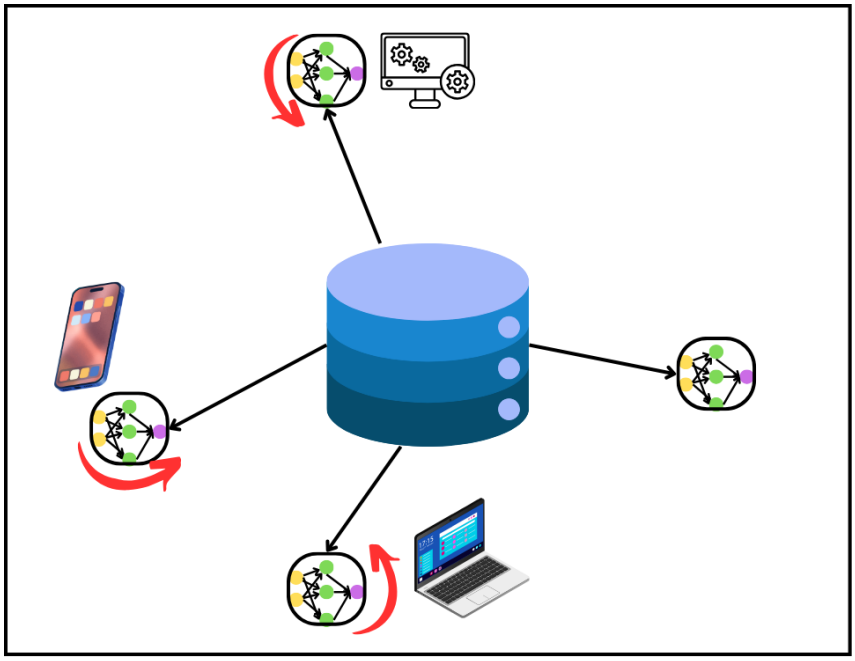

Figure 3.4: Local model taining

3.5.4 Model Update Upload

Each client checks how well its local model performs. If it meets the required performance, the client sends the model to the central server. This step is important for tasks like choosing the best clients, rewarding contributions, or grouping clients. After training, the local model data (like weights or gradients) is uploaded to the central server. But sending this data can be expensive if there are many clients or limited internet speed. To reduce this cost, a message compressor can be used. Tools like Google Sketched Update and IBM PruneFL help reduce the amount of data sent.[63]

3.5.5 Model Aggregation

The server merges all client updates into one global model using Federated Averaging (FedAvg), which computes a weighted average of each client's model based on its number of training samples. This ensures that every data point, regardless of its source, has the same influence, so a client with 10 samples does not outweigh one with 100.[62]

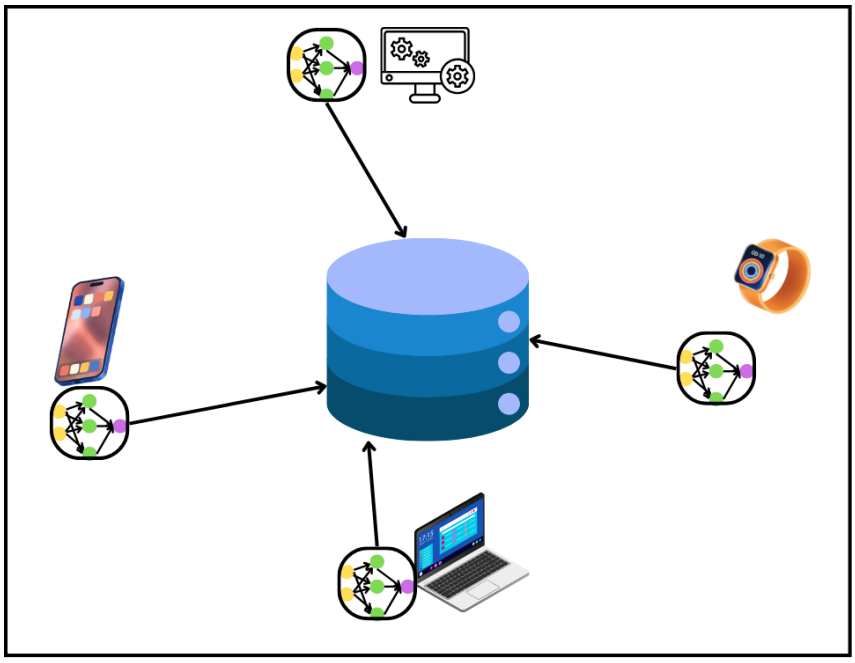

Figure 3.5: model Aggregation

3.5.6 repeat steps

During each training round, the selected client devices only train their local models for a short time. As a result, after aggregation, the global model has only been slightly updated using the data from participating clients. To build a well-performing model that generalizes across all client data, this process must be repeated over many rounds.

3.6 Algorithm of federated learning

In federated learning, aggregation algorithms are key to merging updates from different clients into one global model without sharing raw data. They help the model improve while protecting each client's privacy.

3.6.1 Fedaverage

Federated Averaging (FedAvg) is the most commonly used strategy in federated learning. In this method, each client trains a model using its own local data, and only the model updates not the raw data are shared with a central server. The server then averages these updates to create a new global model and sends it back to the clients. This process continues until the model becomes accurate enough. FedAvg is known for protecting data privacy, reducing communication costs, and being highly scalable. It works well even with a large number of clients, who can join or leave the training process at any time, making it a practical and environmentally friendly solution for decentralized machine learning.[53]

3.6.2 FedProx

FedProx is an improved version of FedAvg that helps handle differences in client data, known as statistical heterogeneity. It works by adding a regularization term to each client's

training process, which helps keep the local model closer to the global model. This small adjustment, called re-parameterization, leads to more stable and accurate training. In practice, FedProx has shown better and more reliable convergence compared to FedAvg. Theoretically, it also ensures that each device can successfully contribute to the training, even when the data is very different across devices. Overall, FedProx offers a more robust and stable alternative to FedAvg.[53]

3.6.3 FedSGD

FedSGD (Federated Stochastic Gradient Descent) is a method used with neural networks that works by averaging gradients from multiple devices. It can also include differential privacy (DP) to protect user data. Compared to training a model on just one device, FedSGD often gives better accuracy and strikes a good balance between privacy and performance by sharing only part of the gradients.[64]

3.7 Data partition

3.7.1 IID

In simple terms, the IID (independent and identically distributed) assumption means that a group of random variables are not only independent from each other, but also come from the same probability distribution. This idea is very important in probability and statistics because it makes many calculations and theories possible, like how we estimate averages or use Bayes' rule. The IID concept is also used in many fields such as machine learning, data science, information theory, and engineering. It plays a big role in how we analyze data, build models, and understand uncertainty, making it a core idea behind much of modern science and technology.[65]

3.7.2 NO-IID

In centralized learning, non-IID data refers to situations where the data does not follow a uniform distribution. This can happen due to class imbalances, changes in data over time, or biases during data collection. While these issues can be challenging, they are often addressed using techniques like data augmentation, resampling, or careful train-test splitting.[66] However, in federated learning, the problem of non-IID data is more complex. Each client or participant may have their own unique and uneven data distribution, and even the data within a single client can be unbalanced or biased. This creates two layers of non-IID challenges: differences within each client's data (intra-client non-IID) and differences across clients (inter-client non-IID). Because federated learning does not allow direct data sharing due to privacy concerns, traditional centralized methods are not effective for handling these issues, making non-IID data a significant challenge in FL.[67]

3.8 Classification of Federated Learning

3.8.1 Vertical Federated Learning

is a type of federated learning used when different parties (e.g., organizations or institutions) have data about the same users but with different types of features. In other

words, the data is split vertically, based on features rather than users. The participating parties share common sample IDs (such as customer IDs) but each holds different attributes (such as one party having financial data and another having browsing behavior). These parties work together to train a machine learning model without sharing their raw data, preserving privacy while making use of their combined feature sets.[53]

3.8.2 Horizontal Federated Learning

Horizontal Federated Learning (HFL) is used when different clients or institutions have datasets with the same types of features but different samples or users. This means the data is split horizontally, based on users rather than features. Each client trains a local model on its own data using the same machine learning algorithm, and then only shares the model updates (not the raw data) with a central server. The server then aggregates these updates to build a global model. For example, in smart healthcare, HFL can be used for speech disorder diagnosis, where users speak the same sentence (same features) in different voices (different data samples). Their devices train local models, and the server combines the updates to improve a shared speech recognition model all while keeping each user's voice data private.[68]

3.8.3 Federated Transfer Learning

Federated Transfer Learning (FTL) is used when different clients have both different users (segment spaces) and different types of features (feature spaces) making their datasets very different from one another. FTL uses transfer learning techniques to align and convert different feature spaces into a common form so that clients can still train models collaboratively. During training, privacy is preserved using encryption while exchanging gradients between clients and the server. In smart healthcare, FTL allows hospitals from different countries—with different patient types and medical practices—to collaborate on building a shared AI model. This helps improve disease diagnosis by learning from a wider variety of data, even though the data formats and patient populations are different.[68]

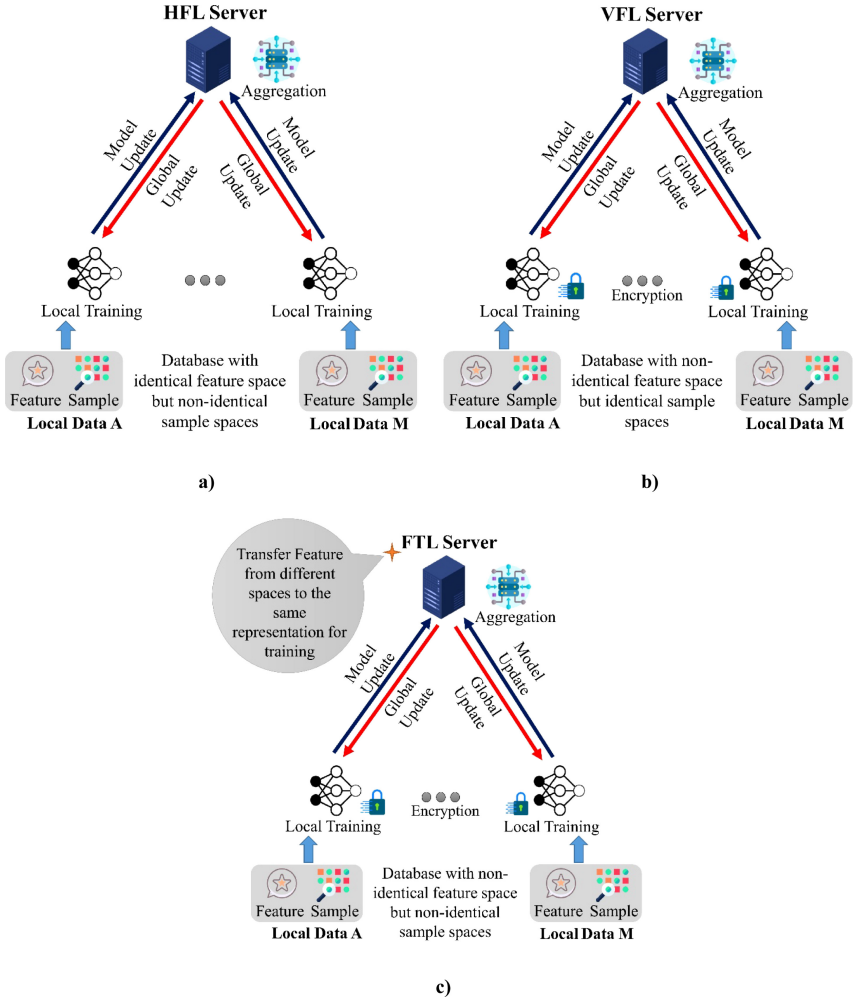

Figure 3.6: VFL/HFL/FTL[68]

3.9 Federated Learning for Smart Healthcare

The healthcare sector has long struggled with protecting sensitive patient information, facing threats like unauthorized access to records (which stronger access controls, authentication, and audit logs can prevent), insider misuse of data (which targeted training, strict access policies, and monitoring can curb), and external breaches or cyberattacks (which encryption, robust network security, and continuous monitoring can defend against). While these issues and their fixes are well understood, the rise of advanced tools like artificial intelligence introduces new, subtle privacy risks AI models can inadvertently reveal details about the data they were trained on. Because health data are governed by strict regulations, it's crucial for researchers developing AI solutions in health care to deeply understand these technical challenges and follow clear, domain-specific privacy guidelines.[69]

3.9.1 Limitations of Current Smart Healthcare Systems

- **Privacy Risks:** Centralized AI requires sharing raw patient data with clouds, exposing it to hackers or unauthorized insiders.

- **Data Scarcity:** Individual medical sites often lack enough local data for accurate model training, leading to slow manual analysis.
- **Poor Model Performance:** Small or imbalanced datasets even with GAN-based augmentation still yield low accuracy.
- **High Communication Costs:** Transmitting large medical files (images, audio) to the cloud consumes bandwidth, increases latency, and drains device batteries.[70]

3.9.2 Benefits of Federated Learning in Smart Healthcare

- **Enhanced Privacy:** Patient data stays on local devices; only model updates (gradients) are shared.
- **Balanced Accuracy & Utility:** Minor accuracy loss vs. centralized models, but with strong generalization and privacy gains.
- **Reduced Communication Overhead:** Lightweight updates cut network congestion, latency, and power use.
- **Scalability:** Easily extends to millions of devices without centralized data storage.[70]

3.10 Conclusion

This chapter introduced federated learning as a way to train models across distributed, privacy-sensitive devices without sharing raw data. We covered FL's main architectures (horizontal, vertical, transfer), core algorithms (e.g., FedAvg), and challenges around communication, heterogeneity, and privacy. We also showed how FL overcomes traditional healthcare AI limitations by keeping data local and reducing network costs. In the next chapter, we will present our experimental setup datasets, federated CNN/LSTM models, and evaluation metrics and report results on accuracy, communication efficiency, and privacy–utility trade-offs in realistic IoMT scenarios.

Chapter 4

Implementation and Experimental Results

4.1 Introduction

In this chapter, the practical application of the proposed deep learning models is described and the experimental evaluation conducted to assess their effectiveness in classifying network attacks using the CIC-BCCC-NRC-IoMT-2024 Dataset. Initially, the chapter presents the dataset and explains its preprocessing methodologies to make it suitable for the training phase. Then the chapter outlines the experimental setup, including the hardware and software specifications used. Both centralized (local) learning and federated learning models are implemented and compared in terms of architecture and performance. The results of each model are presented and analyzed, with a final comparison between centralized and federated approaches to highlight the strengths and limitations of each. The chapter concludes with a discussion of the outcomes and a summary of key findings.

4.2 Implementation

4.2.1 Environments and Libraries

4.2.1.1 Programming Language

- **Python:** Python is a high-level, interpreted, object-oriented programming language with dynamic semantics. It is easy to learn and has a simple syntax, which improves code readabilit . Python includes advanced built-in data structures and supports dynamic typing and binding, making it ideal for rapid application development and as a scripting language to integrate different components. It also supports modules and packages, promoting code organization and reuse.[71]

4.2.1.2 Development Environment

- **Google Colaboratory (Colab):** Colab is a hosted Jupyter Notebook service that offers free access to computer resources, such as GPUs and TPUs. Colab is particularly well-suited to data science, teaching, and machine learning.[72]
- **Kaggle Notebooks:** is one of the largest communities around the globe that centers on data science and machine learning. Its framework and materials allow users to develop models and analyze data which can be shared with experts globally, allowing idea exchange on a professional level. Moreover, it provides users with personalized training lessons, contests, collaborative Jupyter Notebooks, and other big datasets.[73] Kaggle's programmatic environment enables and free GP GPU access, which is ideal for training more powerful Federated Learning (FL) models. FL models have been successfully executed on Kaggle.

4.2.1.3 Libraries: A variety of libraries were used to implement the models and facilitate data processing, training, and evaluation. These libraries include:

- **NumPy:** is the main Python library for scientific computing. This Python library offers a number of derived objects (such as masked arrays and matrices) and a

wide range of routines for fast array operations, such as mathematical, logical, discrete Fourier transforms, basic linear algebra, basic statistical operations, random simulation, and much more.[74]

- **Pandas:** Built on top of the Python programming language, pandas is an open source data analysis and manipulation tool that is quick, strong, adaptable, and simple to use. handles data in an easily manipulable DataFrame format.[75]
- **Matplotlib:** A complete library for producing static, animated, and interactive visualizations .used to create superior graphs and charts that facilitate graphical analysis.[76]
- **PyTorch:** is a tensor library optimized for deep learning with CPUs and GPUs.[77] It is the basic framework used in this work to build deep models such as LSTM and CNN.
- **Flower:** it is a user-friendly, open source federated learning framework [78],For creating federated learning applications. LSTM and CNN-based FL models were implemented using Flower, which offers a flexible environment for decentralized training distribution and coordination across numerous clients while protecting data privacy.

The specifications of each cloud environment used during the experiments are summarized in the table below:

Table 4.1: Table of specifications of cloud environment used

Platform	Processor	GPU Used	RAM	Usage Context
Google Colaboratory	Intel Xeon	Yes (GPU T4 in LSTM Model) and (CPU in CNN Model)	12.72 GB	Centralized Learning (LC)
Kaggle Notebooks	Intel Xeon	Yes (Tesla T4)	13 GB	Federated Learning (FL) with Flower

4.3 The Experimental Dataset

4.3.1 Dataset Description

1 - CIC-BCCC-NRC-IoMT-2024 Dataset Description: The Canadian Institute for Cybersecurity (CIC) "is a comprehensive multidisciplinary training, research and development, and entrepreneurial unit that draws on the expertise of researchers in the social sciences, business, computer science, engineering, law and science. Based at the University of New Brunswick in Fredericton, the institution is the first of its kind to bring together researchers and practitioners from across the academic spectrum to share innovative ideas, create disruptive technology and carry out groundbreaking research into the most pressing cybersecurity challenges of our time." [79]

As part of its mission to enhance cybersecurity research, The National Research Council (NRC) of Canada, the Behavioural Centric Cybersecurity Centre (BCCC), and the Canadian Institute for Cybersecurity (CIC) collaborated to create dataset CIC-BCCC-NRC-IoMT-2024. It serves as a valuable resource for researchers seeking to enhance cybersecurity in IoMT environments.

2 - Datasets Developed by CIC: The CIC has helped create many datasets in the field of cybersecurity research and digital threat monitoring, which include:

Table 4.2: Datasets Developed by CIC for Cybersecurity Research

Dataset Name	**Description**
CIC-IDS 2017	includes information on network traffic and several attack types, including botnet, brute force, and DDoS assaults.
CIC-IDS 2018	An enhanced version of CIC-IDS 2017 that simulates a wider range of assaults in more realistic network settings.
CIC-ToN-IoT	Designed for studying security threats in Internet of Things (IoT) networks, including industrial IoT data.
CIC-IoT2022	IoT-based traffic
CIC-MalMem-2022	Focuses on malware detection using memory analysis techniques.
CIC-DDoS 2019	focuses on examining the various forms of DDoS attacks and how they affect network performance.
CIC-BCCC-NRC-IoMT 2024	Developed collaboratively by the Canadian Institute for Cybersecurity (CIC), the Behavioural Centric Cybersecurity Centre (BCCC), and the National Research Council (NRC) of Canada. This dataset focuses on cyber threats in the Internet of Medical Things (IoMT)

3 -Structure and Collection Environment of CIC-BCCC-NRC-IoMT-2024 Dataset

A dedicated IoMT testbed was established comprising:

- Real medical devices such as infusion pumps, patient monitors, smart ventilators, and wearable health trackers.
- Network infrastructure including managed switches, a gateway router, and a passive network tap for traffic capture.
- A workstation hosting attack-generation tools and packet-capture software (e.g., Wireshark, tcpdump).

- All devices run manufacturer-provided native firmware images (no emulation).

The **CIC-BCCC-NRC-IoMT-2024** dataset was generated using this realistic testbed environment. It is a specialized subset of the broader **CIC-BCCC-NRC TabularIoTAttack-2024** dataset, aimed at supporting cybersecurity research in Internet of Medical Things (IoMT) environments.

This dataset contains tabular CSV data of engineered network features extracted from traffic captured in the testbed, labeled with five main attack categories including DoS, DDoS, reconnaissance, MQTT attacks, spoofing plus benign traffic. The data is highly realistic, gathered from real medical devices under multiple simulated attack scenarios, and is suitable for training and evaluating AI-driven cybersecurity models in both centralized and federated learning contexts.

Attack Categories in the Dataset: The attacks are classified into 5 main categories as follows:

Table 4.3: Main Attack Categories in the CIC-BCCC-NRC-IoMT-2024 Dataset

Category	**Description**
DDoS Attacks	Distributed Denial of Service attacks
DoS Attacks	Denial of Service attacks
Reconnaissance Attacks	Scanning and information-gathering attacks
MQTT Attacks	Attacks targeting the MQTT protocol used in medical IoT devices
Spoofing Attacks	Identity spoofing attacks such as MITM (Man-In-The-Middle)

Dataset File and Directory Structure: Available Files and Their Sizes : The dataset includes multiple files, each representing different cyberattack scenarios and normal traffic:

Table 4.4: Dataset File and Directory Structure with Corresponding Sizes

File Name	**Size**
Benign Traffic.csv	21M
DDoS ICMP Flood.csv	1.6M
DDoS UDP Flood.csv	1.6M
DoS ICMP Flood.csv	1.4M
DoS TCP Flood.csv	1.0G
DoS UDP Flood.csv	1.9M
MITM ARP Spoofing.csv	690K
MQTT DDoS Publish Flood.csv	192M
MQTT DoS Connect Flood.csv	3.3M
MQTT DoS Publish Flood.csv	687K

MQTT Malformed.csv	1.2M
Recon OS Scan.csv	36K
Recon Ping Sweep.csv	47K
Recon Port Scan.csv	204M
Recon Vulnerability Scan.csv	3.6M

4.3.2 Preprocessing Dataset

Data preprocessing is defined as one of the most data mining processes that entails data preparation and data transformation to a proper format to the mining process. Data preprocessing is meant to reduce the size of the data, recognize the relationship between data, normalize data.[10] This portion explains data cleaning, managing incomplete data, handling outliers, dataset class balance, and dimensionality reduction using Principal Component Analysis (PCA).

4.2.1.1 4.3.2.1 Preprocessing Steps in Data Analysis

1. Data Cleaning:

Data cleaning is an initial step in data preprocessing methods which is utilized to identify the missing values, smooth noise data, detect outliers and rectify inconsistent. These dirty data will have effects on mining procedure and resulted in unreliable and poor output. Thus, it is necessary for some data-cleaning routines to be applied.[80] Where the actions listed below were taken:

- **Removing Unnecessary Columns:** Columns that do not contribute to the classification of attacks such as Flow ID, SRC IP, Dst IP, Timestamp, and Attack Name were removed due to redundancy. This step enhances the efficiency of the model.

- **Encoding Categorical Variables:** The Attack-Type column was transformed into numerical values using label encoding, facilitating the model's ability to process categorical data.

Encoded	Original
0	Benign Traffic
1	DDoS ICMP Flood
2	DDoS UDP Flood
3	DoS ICMP Flood
4	DoS TCP Flood
5	DoS UDP Flood
6	MITM ARP Spoofing
7	MQTT DDoS Publish Flood
8	MQTT DoS Connect Flood
9	MQTT DoS Publish Flood
10	MQTT Malformed
11	Recon OS Scan
12	Recon Ping Sweep
13	Recon Port Scan
14	Recon Vulnerability Scan

Figure 4.1: Encoding of categorical variables

• **Handling Missing Data:** To minimize the impact of outliers, the median was used to replace missing values in numerical attributes instead of the mean.

• **For categorical attributes** , the most frequent value (mode) was employed to fill in missing data, ensuring consistency across the dataset.

2. Handling Outliers

Using the Interquartile Range (IQR) method: By splitting a data set into quartiles, IQR is a tool for measuring variability. After sorting the data in ascending order, we divide it into four equal halves. The dataset is divided into four equal portions by the values Q1 (25th percentile), Q2 (50th percentile or median), and Q3 (75th percentile).[81] :

- The first (Q1) and third quartiles (Q3) were calculated, and the interquartile range was determined as $IQR = Q3 - Q1$.
- Outlier data points are those that are above $Q3 + 1.5 \times IQR$ or below $Q1 - 1.5 \times IQR$.

3. Data Balancing

As models tend to prefer the majority classes, it poses a serious problem during a classification task. This is referred to as data imbalance when " Data is said to be imbalanced if at least one of the target variable values has a significantly smaller number of instances when compared to the other values. " [82] .SMOTE, which stands for Synthetic Minority Over-sampling Technique, has been proposed as a solution.

• **SMOTE:** "is a pioneer oversampling method in the research community for imbalanced classification. The basic idea of SMOTE is oversampled by creating a synthetic instance in feature space formed by the instance and its K-nearest neighbors due to the ability to avoid overfitting and assist the classifier in finding decision boundaries between classes."[83]

Why SMOTE?

- Avoids overfitting: Synthetic samples are generated with respect to known patterns rather than simple duplication.
- Improves model precision: A balanced dataset ensures the model learns to recognize each type of attack correctly.
- Some attack classes had very few samples, and it was difficult for the model to accurately label them. SMOTE seeks to improve the generalization capability of the model, By creating fresh synthetic samples that provide the learning algorithm more information, and, as a result, greatly improve the predicted performance for the underrepresented class.[84]

Steps in Applying SMOTE: The SMOTE implementation process goes through several basic steps, according to these studies. [85] [86]These can be summarized as follows:

- Identify the imbalance: In the dataset, some categories contained more than 2 million samples, while another category contained less than 100. This led to a significant and clear imbalance in the categories of the dataset.
- Focus on the Minority Class: SMOTE specifically generates new data points for the minority class rather than the majority. For each instance in the minority class, it identifies a group of k-nearest neighbors from the same class, using a distance metric such as Euclidean distance in the feature space.
- SMOTE creates new synthetic samples instead of repetition by: Computing the difference between the original sample and one of its nearest neighbors, and multiplying by a random number between 0 and 1. Adding this result to the original sample provides a new point on the line between the original sample and the neighbor.
- Such iterations are performed for all underrepresented class samples until the desired level of imbalance between classes in the model. This is because the model has to be optimized for better ability to learn from Rare groups.
- Finally, the distribution is re-evaluated after the balance to ensure fair representation for all groups.

Class Distribution Before and After Balancing:

- Number of samples after cleaning: 3,247,350.
- Number of features after cleaning: 81.

Table 4.5: Class distribution before and after balancing the dataset

Class	Class Distribution Before Balancing	Class Distribution After Balancing
0	32577	32577
1	2528	5000
2	2544	5000
3	2107	5000
4	2106902	200000
5	3084	5000
6	1053	3000
7	276151	200000
8	237974	200000
9	953	30000
10	2246	5000
11	85317	85317
12	71	15000
13	485522	200000
14	8321	15000

4. Dimensionality Reduction Using PCA

Why Reduce Dimensions? PCA transforms high-dimensional data into a lower-dimensional form while retaining as much variance as possible. Increased calculation time,Overfitting,storage space for massive data, and other practical issues are some of the issues that high-dimensional datasets provide for machine learning methods. Perhaps the greatest worry, though, is that prediction models are becoming less accurate. Models for machine learning and statistics that are trained on high-dimensional datasets frequently have poor generalization.[87] Principal Component Analysis (PCA) reduces the amount of datasets while assisting in the extraction of the most important characteristics.

4.1 Understanding Principal Component Analysis (PCA): "Principal Component Analysis (PCA) is a dimensionality reduction and machine learning method used to simplify a large data set into a smaller set while still maintaining significant patterns and trends".[88] In PCA, the first component is extracted to explain the largest share of total variance in the variables observed. This component is most likely to be correlated with more than one variable. The second component is extracted to explain as much of the remaining variance not explained by the first component as possible, because the two components are uncorrelated. The rest of the components follow the same principle, that each new component accounts for variance not accounted for by previous components. The analysis is carried out in such a way that the components appear completely independent and unrelated to each other.[89]

4.1.1 Step-by-Step Explanation of PCA

Step 1. Standardization: In PCA, standardization is performed to ensure that no variable with a high degree of variance dominates others, This leads to a more accurate

and balanced analysis,Below is the mathematical expression [90]:

$$Z = \frac{\text{value} - \text{mean}}{\text{standard deviation}}$$

Step 2. Computing the Covariance Matrix: In Principal Component Analysis, PCA, the covariance matrix is one of the tools that is crucial for studying the relationship of different variables in a dataset. After standardizing the data, the covariance matrix is calculated to find how the variables change together. If the covariance between two variables is positive, it shows that there is a direct relation or positive correlation. On the other hand, when it is negative, it shows that there is an inverse relation or negative correlation. For those cases in which the number is around zero, there is most likely no linear relation. As described above, The covariance matrix is a key tool for identifying patterns and extracting the directions with the highest variance, which helps in reducing the dimensions while preserving the most influential information in the data.[91] In other words, the sign of the covariance (positive or negative) represents the type of relationship between the two variables:

- Positive covariance: The variables tend to increase or decrease together (positive correlation).
- Negative covariance: One variable increases as the other decreases (negative correlation).
- Zero or near-zero covariance: No clear linear relationship.

Step 3. Calculating Eigenvalues and Eigenvectors: Eigenvalues and Eigenvectors When it comes to data, eigenvalues denote the volume of variance contained in each axis of the data. Therefore, they are arranged in descending order to pinpoint the axes containing the greatest variance. Eigenvectors represent new directions that capture the maximum variance in the data. Each eigenvalue has its corresponding eigenvector which determines the direction of maximum variance, thus aiding in directing focus within the transformation space that holds the principal variance.[92]

Step 4. Selecting the Number of Principal Components: A cumulative variance plot is used to choose the number of principal components in order to maintain a high percentage of variance (95%) for example. After calculating and sorting the eigenvectors, only the most important components are kept. This creates the feature vector matrix, which lowers the dimensionality of the data while retaining the greatest amount of information [92].

Step 5. Transforming the Data: Projecting the original dataset onto the axes of the chosen principal components is the last step in Principal Component Analysis (PCA). This dimensionality reduction is achieved by multiplying the transposed original data matrix by the matrix of selected eigenvectors, preserving as much of the data variance as possible Below is the mathematical expression [90] :

$$\text{Final Dataset} = W^T \cdot Z^T$$

Where:

$$W = \text{Matrix of principal component vectors (feature vectors)}$$
$$Z = \text{Standardized original dataset}$$
$$T = \text{Transpose operation}$$

4.1.2 Why PCA Was Used in This Research

In this particular study, PCA was employed for these reasons:

- To maintain model accuracy while reducing dataset size.
- Improving the performance of a deep learning model by increasing computational efficiency.
- Improving data quality by removing unimportant and redundant features.
- Improving generalization and decreasing overfitting.

Figure 4.4 illustrates the application of Principal Component Analysis (PCA) on the dataset used in this study :

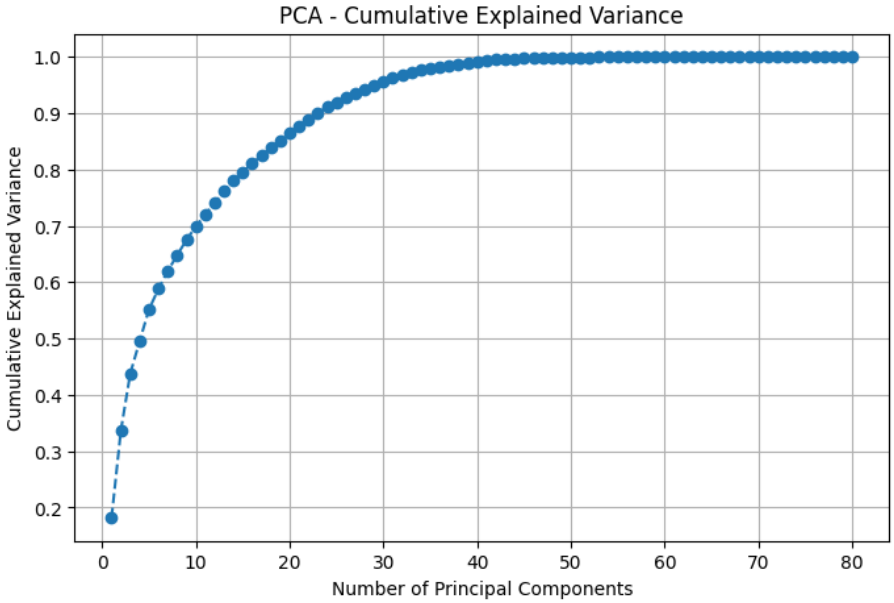

Figure 4.2: Application of Principal Component Analysis (PCA).

5. Data Normalization & Standardization:

Normalization As mentioned in [93], normalization refers to the procedure of converting digital data into comparable values. This can be done by implementing two primary techniques:

1. Min-Max Normalization: This method scales the data within a specified range [93]. In this method, a range of 0-1 was used to scale the data, ensuring comparability between features and addressing the large variation in values.

2. Z-score Normalization: This method uses the mean and standard deviation of the data [93]. In this method, features are transformed to have a zero mean and unit variance which helps ensure comparability across different features.

4.3 Compilation Process :

The following functions were used during the compilation of both LSTM and CNN models in the centralized learning setup:

4.3.1 Activation Functions Used :

• **ReLU (Rectified Linear Unit) :** With clean deep learning, the ReLU function also serves as an effective activation function that is straightforward to apply , they mitigate the vanishing gradient problem within non-negative bounds, setting zero as a lower bound while preserving positive values. They can be illustrated mathematically in the following way [94]:

$$f(x) = \max(0, x) = \begin{cases} x, & \text{if } x \geq 0 \\ 0, & \text{if } x < 0 \end{cases}$$

The following figure illustrates a graphical representation of the ReLU function:

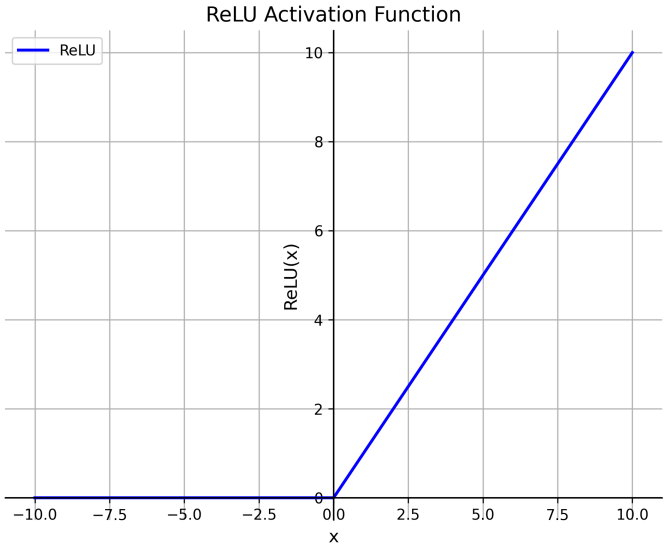

Figure 4.3: ReLU Activation Function

• **Softmax function:** In neural networks, the Softmax function is used as another activation function. It changes a vector of real numbers into a probability distribution. The values produced are between 0 and 1, and at the same time, the total is 1. The following formula will calculate Softmax function [94]:

$$f(x_i) = \frac{e^{x_i}}{\sum_j e^{x_j}}$$

4.3.2 Loss Functions Used :

Loss Function – Cross-Entropy Loss: As with other loss functions, Cross Entropy Loss is favorable in deep learning, particularly in cases dealing with multilabel classification problems. This function assesses the difference between a true class distribution and what the model predicts, measuring the accuracy of predicted probabilities with regard to the class [95]. and the categorical cross entropy loss (LCCE) is defined as [96]:

$$L_{\text{CCE}}(y, p) = -\frac{1}{N} \sum_{i=1}^{N} \sum_{c=1}^{C} y_{i,c} \log(p_{i,c})$$

where:

- N represents the total number of data samples.
- C stands for the number of possible classes.
- $y_{i,c}$ is 1 if the correct class for sample i is class c, and 0 otherwise.
- $p_{i,c}$ is the model's predicted probability that sample i belongs to class c.

4.3.3 Optimization Algorithm Used :

Adam Optimizer: It is one of the most effective optimization algorithms in neural networks training. Adam combines RMSProp and Momentum via moving averages for both gradients and quadratic gradients. Adam automatically adjusts the learning rate for each parameter individually,which enables lower cost computations and better efficiency of memory. It surpasses the results achieved through conventional stochastic gradient descent algorithms with remarkable ease [97].

4.3.4 Centralized Deep Learning Models:

4.3.4.1 Convolutional Neural Network (CNN) Model Architecture Description

In this work, we implemented a Convolutional Neural Network (CNN) model within a centralized learning approach to classify network intrusions based on the CIC-BCCC-NRC-IoMT-2024 Dataset. The architecture was designed to extract spatial features from the input data through convolutional operations, followed by pooling and fully connected layers to perform classification.
The CNN architecture consists of the following layers:

- Input Layer: The model takes as input the features presented in 1D arrays for sequential (1D) convolution. The input size is equal to the number of features of the model after preprocessing.
- First Convolutional Layer: Applies 32 1D filters of kernel size 3. This layer is followed by a ReLU activation function to introduce non-linearity.
- Max-Pooling Layer: Applies 2 pooling window to the output from first convolution to reduce the dimensionality.

- Second Convolutional Layer: 1D Convolution with 64 filters of size 3 followed by ReLU activation, then a max-pooling layer.
- Flatten Layer: The output from the last pooling layer is turned into a 1D array to be input into the Fully Connected Layer.
- Fully Connected Layer (FC1): A dense layer of 64 neurons is added, and subsequently a dropout layer of 0.5 rate is also applied to reduce overfitting.
- Output Layer (FC2): This last layer has as many neurons as there are target classes in the dataset, making it a fully connected layer. This layer predicts the class probabilities through inference using a softmax activation.

PyTorch was used to implement the model, and when a GPU was available, it was used for training. With a batch size of 64, it was optimized with a suitable optimizer (like Adam) and a categorical cross-entropy loss function. Eighty percent of the dataset was used to train the model, and another ten percent was used for testing and validation.

4.3.4.2 The Long Short-Term Memory (LSTM) Model Architecture Description

In this work, a neural network based on a Bidirectional Long Short-Term Memory (BiLSTM) architecture within the Centralized Learning (CL) framework was used to develop a multi-class classification model. This model's objective is to identify and categorize different kinds of cyberattacks in an Internet of Medical Things (IoMT) setting. The PyTorch library was used to implement the model.
The BiLSTM model consists of the following components:

- Input Layer: The preprocessed input features were reshaped to the LSTM input format of [batch_size, sequence_length, features].
- First LSTM Layer: A 128-unit-per-hidden-layer bidirectional LSTM was created: nn.LSTM(input_size=1, hidden_size=128, bidirectional=True) The dimensionality of the input features was 1. The output size of this layer will be a feature representation of size 256 after considering both directions.
- Second LSTM Layer: The second layer consists of a bidirectional LSTM with 64 hidden units in each direction. It further analyzes the temporal features and decreases the output to 128.
- Fully Connected Layers: The last dense layers process the output from the LSTM layers. The output from the LSTM pass through:Linear(128, 32), follows by RELU Function.
  The output is produced from a dense layer using: Linear(32, num_classes) to supply the class logits.
  For every two layers, a dropout with 0.2 is applied in order to prevent overfitting.
- Activation Function: ReLU was introduced to add non-linearity between fully connected layers.
- Output Layer: These logits are supplied for each class. During inference these probabilities are computed using softmax.

The model was trained with 80% of the dataset, validating with 10% and testing with the last 10%. The training carried out was set to a maximum of 50 epochs, 64 in batches. Overfitting was prevented with Early Stopping after 5 epochs of no improvement, checkpoints of the model were saved automatically during training for evaluation on the test set later.

Rationale for Using BiLSTM: The model selected was Bidirectional LSTM as it has the strength to learn contextual information from both past and future time steps. This type of perspective is important in identifying complex patterns in IoMT traffic since the series of network events depends on packets arriving before and after in the queue. BiLSTM, unlike conventional or one directional models, has increased sensitivity to time differences, thus making it optimal for the medical IoT cyberattack classification problem.

4.3.4.3 Performances Models:

CNN Models Performance:

Table 4.6 presents the classification report for the CNN model, detailing precision, recall, F1-score, and support for each class.

Table 4.6: Classification report for the CL CNN model.

Class	Precision	Recall	F1-score	Support
0	0.9982	1.0000	0.9991	1636
1	0.3393	0.2901	0.3128	262
2	0.3769	0.2076	0.2678	236
3	0.3068	0.3105	0.3086	248
4	0.9999	0.9988	0.9993	9857
5	0.5040	0.2593	0.3424	243
6	0.5373	0.2416	0.3333	149
7	0.9948	0.9975	0.9962	10034
8	0.9874	0.9956	0.9915	9837
9	0.8246	0.9637	0.8888	1517
10	0.8821	0.6300	0.7350	273
11	0.6066	0.1914	0.2909	4149
12	0.8766	0.9866	0.9284	749
13	0.7468	0.9531	0.8374	10037
14	0.7862	0.8228	0.8040	773
Accuracy			**0.8991**	50000
Macro Avg	0.7178	0.6566	0.6690	50000
Weighted Avg	0.8882	0.8991	0.8813	50000

Result Analysis:

The CNN model performed well in classifying network attacks, achieving an overall accuracy of approximately 89.9%.It achieved particular success in detecting attacks that were heavily represented in the data, such as categories 0, 4, 7, and 8(Benign Traffic ,DOS TCP Flood , MQTT DDOS Publish Flood And MQTT DOS Connecte Flood),

exceeding precision and recall metrics of 99%.It's evident the model is capable of recurring pattern detection or at the very least, clear pattern detection. On the other hand, some classes were not accurately classified such as class 1, 2, 5, 6 and 11(DDOS ICMP Flood , DDOS UDP Flood , DOS UDP Flood , MITM ARP Spoofing and Recon OS Scan) where recall and precision numbers dropped significantly with some being almost half proving the model struggles to adapt to various classes. The model's weighted F1 mean was approximately 88% with the macro mean standing at roughly 67%, This shows a discrepancy in the model's performance compared to less well-represented categories .
Overall, the CNN model performed very well in classifying attacks that appear heavily in the data, but it struggles to distinguish between some classes with less distribution or that may have similar patterns.

LSTM Models Performance:

Table 4.7: Classification report for the CL LSTM model.

Class	**Precision**	**Recall**	**F1-score**	**Support**
0	1.00	1.00	1.00	3224
1	0.49	0.67	0.56	488
2	0.63	0.38	0.47	457
3	0.47	0.60	0.53	478
4	1.00	1.00	1.00	19831
5	0.61	0.53	0.57	484
6	0.63	0.35	0.45	305
7	1.00	1.00	1.00	20174
8	1.00	0.99	1.00	20086
9	0.98	1.00	0.99	3008
10	0.82	0.87	0.84	497
11	0.90	0.94	0.92	8669
12	0.93	1.00	0.97	1535
13	0.98	0.95	0.97	19866
14	0.98	0.97	0.98	1488
Accuracy			**0.97**	100590
Macro Avg	0.83	0.82	0.82	100590
Weighted Avg	0.97	0.97	0.97	100590

Result Analysis:

The evaluation of classification performance of the LSTM model was done using precision, recall, and F1-score for every class. Evaluation on the model's test dataset yielded an accuracy of 97%, indicating a high level of generalization capability. Moreover, the model reached perfect precision, recall, and F1 score (1.00)for several well-represented classes such as class 0(Benign Traffic), class 4 (DOS TCP Flood), class 7(MQTT DDOS Publish Flood), and class 8(MQTT DOS Connecte Flood), suggesting that these attack types were consistently and correctly classified . Classes 9, 12, 13 and 14(MQTT DOS Publish Flood , Recon Ping Sweep , Recon Port Scan and Recon Vulnerability Scan) likewise received remarkable scores (above 0.97 in all metrics),reflecting excellent detection performance.
In contrast, a larger spread of underrepresented or more complex attack types performed remarkably lower. Take for example, class 2 (precision: 0.63, recall: 0.38) and class 6 (precision: 0.63, recall: 0.35). These two showed significant showed noticeable difficulty, with relatively low recall scores, , meaning that these classes had a larger number of false negatives. Class 3 also faced difficulty with a 0.60 recall, which indicates that a number of cases of this attack were incorrectly classified.

despite these difficulties in a few classes,The weighted average F1-score of 0.97 and macro average F1-score of 0.82 show that the LSTM model performs well across both balanced and imbalanced class distributions, This makes it a highly reliable model for multi-class attack detection in IoT or network environments.

4.3.4.4 Comparing CL Models (LSTM vs CNN) :

Table 4.8: Comparison Between LSTM and CNN Models.

Model	Accuracy	Precision	Recall	F1-Score	Training Time
CNN (CL)	89.91%	88.82%	89.91%	88.13%	1h
BiLSTM (CL)	97.00%	97.00%	97.00%	97.00%	1h45m

A comparative analysis was conducted to evaluate the performance of Centralized Learning (CL)models for detecting and classifying cyberattacks in Internet of Medical Things (IoMT) systems using the CIC-BCCC-NRC-IoMT-2024 Dataset. Through the analysis of the precise evaluation measures of each model, it was observed that the LSTM (CL) model had better performance compared to the CNN (CL) model in terms of accuracy and overall performance. The LSTM (CL) model achieved accuracy of 97.00%, F1 score of 97.00%, recall of 97.00%, and precision of 97.00%, indicating great and balanced performance in accurately classifying attacks. In contrast, the CNN (CL) model achieved accuracy of 89.91%, F1 score of 88.13%, recall of 89.91%, and precision of 88.82%, indicating good but worse performance than that of the LSTM model. These results exhibit LSTM (CL)'s superior ability to learn temporal patterns and thus its efficacy to identify cyberattacks in IoMT data compared to CNN (CL). This comparison is in line with the research objective to determine the superiority of the LSTM model in Centralized Learning environments to protect IoMT.

Training and Validation Accuracy & Loss Curve:

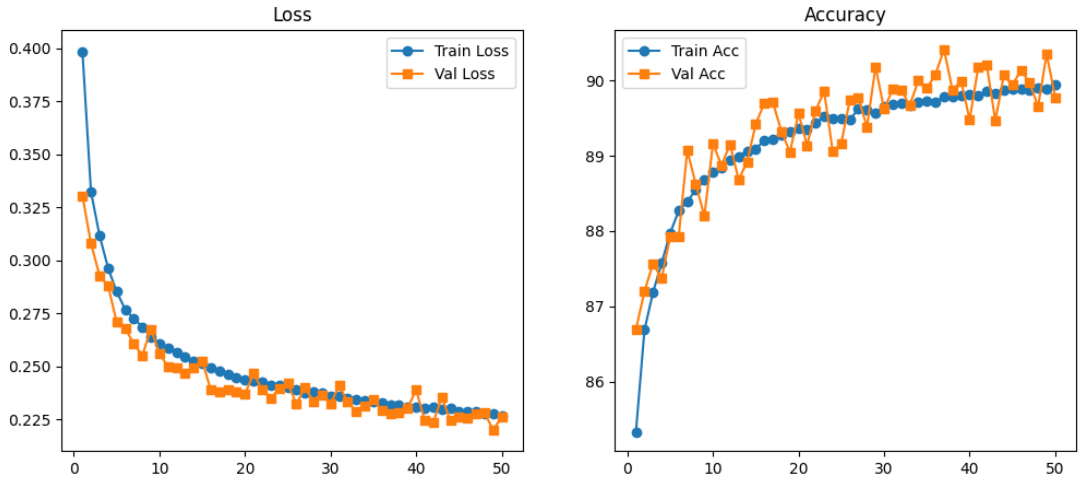

Figure 4.4: Training and Validation Accuracy & Loss Curve CNN CL .

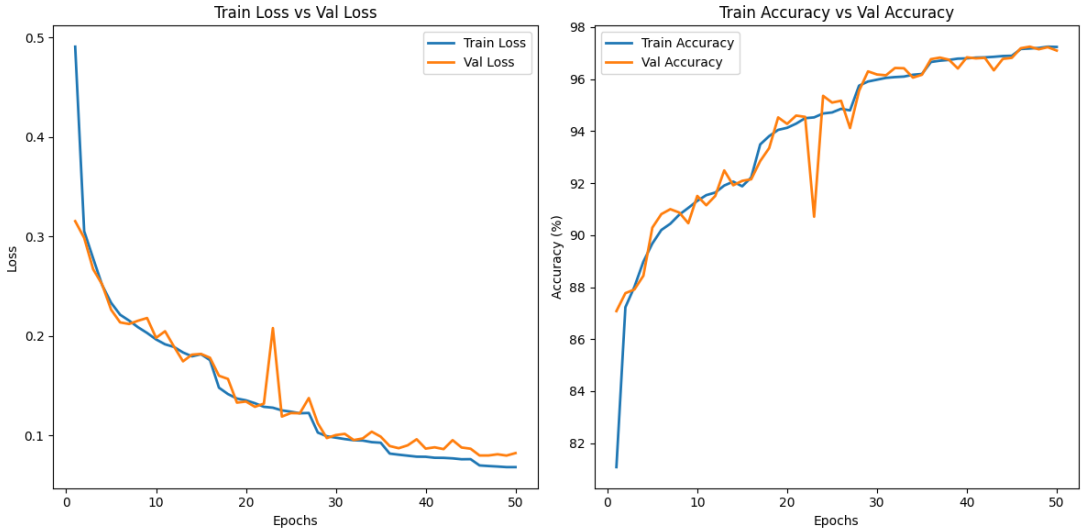

Figure 4.5: Training and Validation Accuracy & Loss Curve LSTM CL .

The training and validation curves show that the LSTM model performs significantly better than the CNN model in terms of accuracy and generalization ability. After 50 training iterations, LSTM had a remarkable training accuracy of 97.24% and validation accuracy of 97.10%, a training loss of 0.0683, and a low validation loss of 0.0824. By comparison, the CNN achieved a training accuracy of 89.94% in addition to validation accuracy of 89.78%, with a training loss that was 0.2268 and 0.2261 for validation loss. These results suggest that LSTM learns better and is more effective at generalization due to the small gap between training and validation metrics.
It can also be observed from LSTM loss curves that they exhibit a steady and smooth decline which indicates stable learning and a continuous decrease in errors. For CNN, there is an initial sharp decline in the first few stages followed by reaching high levels of loss, indicating a decreasing potential for improvement with further training. In general Overall, in modeling serial Internet of Things data, LSTM outperformes other models in both robustness and stability, making it the go to model for the task.

Confusion Matrix:

"is a performance measurement for the machine learning classification problems where the output can be two or more classes. It is a table with combinations of predicted and actual values" [98].

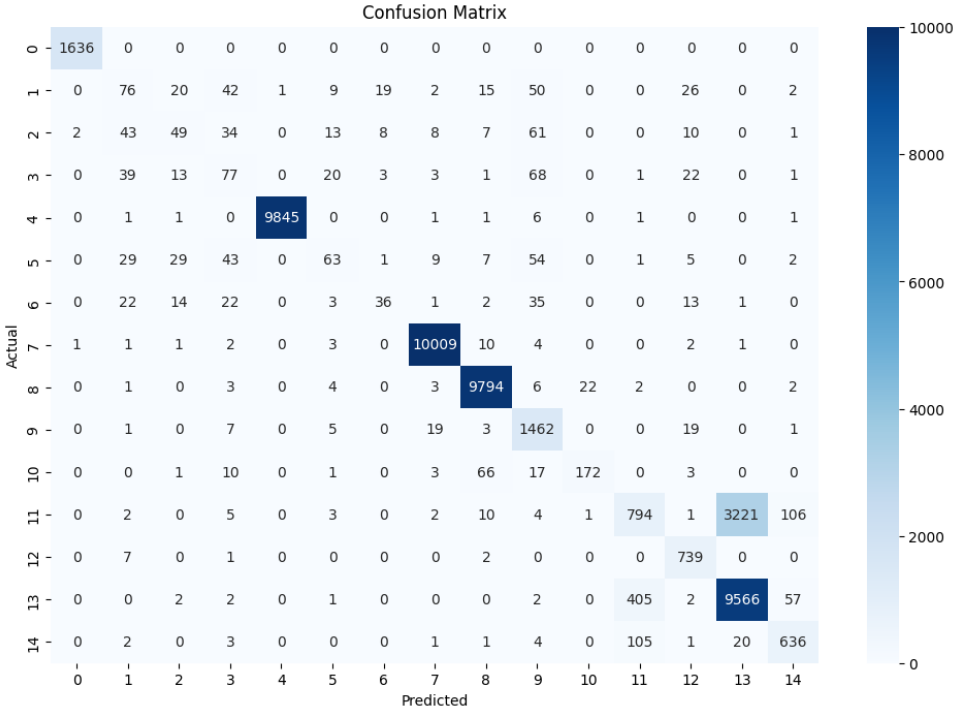

Figure 4.6: Confusion matrix (CNN CL).

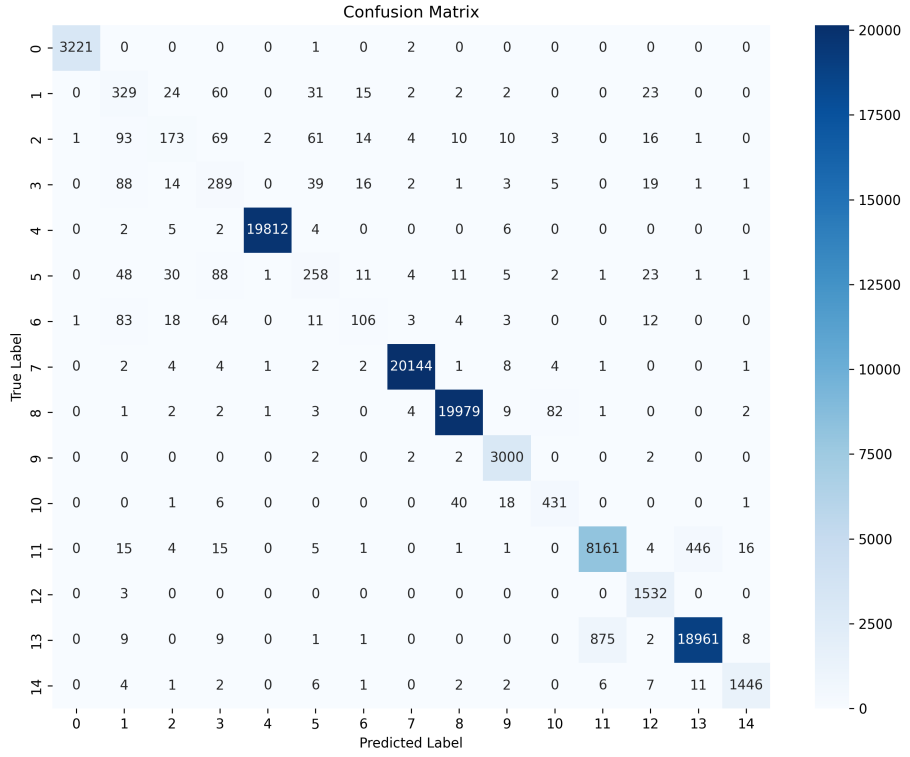

Figure 4.7: Fig.4.confusio n Matrix (LSTM CL).

The confusion matrix for each of these CL-based models is displayed in Fig 7 and 8 .The predicted classes are shown in the columns, while the actual classes are represented by the rows. Whereas the off-diagonal parts show wrong predictions, the diagonal elements show right predictions. a variety of metrics, including recall, precision, overall accuracy, and the F1-score, may be calculated using the confusion matrix. The following confusion matrices illustrate the performance of the CNN and LSTM models in attack classification with 97% accuracy and CNN with 89.91% accuracy.

The LSTM model is better than CNN at had greater True Positive values, meaning it was able to identify a greater number of attacks, which is also associated with the higher values along the primary diagonal of the matrice attesting to its ability to identify patterns and distinguish between nuanced forms of attacks. True Negatives (TN) also accounted for a high proportion, indicating that normal cases were correctly classified which makes the model dependable in sensitive contexts. On the other hand, the CNN model On the other hand, CNN performed less well than LSTM with a sparse matrix, indicating a higher number of false positives (FP), meaning fewer false alarms are avoided. LSTM also outperformed CNN in reducing false negatives (FN), meaning that attacks are classified as normal.

Although both models have high predictions (TP) and (TN), LSTM appears to be more balanced in predicting and classifying attacks in the context of IOMT.

4.3.5 Federated Learning Models

4.3.5.1 Federated learning strategy used

In federated learning models(CNN-LSTM) experiments use the same high-level Federated Averaging (FedAvg) approach: in each round, the server sends the current global model to all clients, each client updates it locally on its private data, then uploads only its model weights (not raw data). The server then averages those weights weighted by each client's dataset size to form a new global model, and the process repeats. This simple cycle of broadcast, local update, and weighted aggregation preserves data privacy while collaboratively improving the shared model.

4.3.5.2 Convolutional Neural Network (CNN) FL Model Training

In this experiment, we simulate a federated learning scenario in which three clients each train an identical 1D convolutional neural network on disjoint shards of a balanced, PCA-transformed network-attack dataset. Each client locally updates the global model for three epochs using Adam with weight decay and a decaying learning rate scheduler, then returns its updated weights and sample counts to the server. The server aggregates these updates via weighted averaging (FedAvg), updates the global model, and evaluates it centrally on a held-out validation set after each round. Over 30 rounds, this process allows the global model to converge improving both validation loss and accuracy while preserving data privacy by never transferring raw client data. Here's a concise summary of the key hyperparameters used in your federated CNN setup:

Hyperparameter	Value	Notes
Model Architecture		
Convolutional blocks	3	Each block: Conv1d → BatchNorm → ReLU → MaxPool1d
Conv1 out_channels	64	in_channels = 1, kernel_size = 3, padding = 1
Conv2 out_channels	128	same kernel & padding
Conv3 out_channels	256	same kernel & padding
Pooling kernel_size	2	halves sequence length each block
Dropout rate	0.20	applied after FC layers
FC layer sizes	128 → 64 → num_classes	two hidden layers before output layer
Training (Local)		
Batch size	64	for both local training and validation
Local epochs per client	3	number of epochs each client trains per round
Optimizer & Scheduler		
Optimizer	Adam	with lr = 0.001
Weight decay	1e-4	L2 regularization
LR scheduler	StepLR	step_size = 1, gamma = 0.9 (decay × 0.9 per local epoch)
Federated Settings		
Number of clients	3	each holds a disjoint data shard
Global rounds	30	total FedAvg aggregation cycles
Fraction fit/evaluate	1.0	all clients participate every round
Aggregation strategy	FedAvg	weighted by each client's sample count

Table 4.9: Hyperparameters for the Federated CNN Experiment

4.3.5.3 The Long Short-Term Memory (LSTM) FL Model Training

In lstm model we simulate a federated learning setup with three clients, each holding a disjoint shard of a standardized dataset . Data are first split 80/10/10 into train/val/test, then the training and validation portions are partitioned IID (or optionally non-IID via a Dirichlet distribution) across clients. Each client fine-tunes the shared bidirectional LSTM model (two stacked BiLSTM layers → dropout → FC layers) locally for one round of full-epoch training using Adam (lr=0.001), then returns updated weights to the server. The server aggregates via weighted FedAvg on both fit and evaluate steps, and after each global round (30 total), it centrally evaluates on the held-out test set, reporting a global test accuracy a table summarizing its key hyperparameters

Component	Hyperparameter	Value / Setting
Data Splits		
	Train/Val/Test ratios	0.8 /0.1/0.1
	Clients	3
	Partitioning	IID (default)
	Batch size	64
Model Architecture		
	Input shape	`(sequence_length=30, channels=1)`
	BiLSTM layers	1st: hidden=128, 2nd: hidden=64 (both bidir)
	Dropout	0.20
	FC layers	`128→32→num_classes`
	Activation	ReLU
Optimizer & Training		
	Optimizer	Adam
	Learning rate	0.001
	Loss	CrossEntropyLoss
	Local epochs per round	1 (full pass over local data)
Federated Settings		
	Global rounds	30
	Strategy	FedAvg (fraction_fit/evaluate=1.0)
	Aggregation weights	by number of local examples
Evaluation		
	Client eval frequency	every round
	Global test evaluation	via `evaluate_fn`, weighted-average metrics

Table 4.10: Summary of Hyperparameters for the Federated BiLSTM Experiment

4.3.5.4 Performances Models:

CNN Model Performance Table 4.11 shows the classification metrics for the CNN model, including precision, recall, F1-score, and support for each class.

Class	Precision	Recall	F1-score	Support
0	1.00	1.00	1.00	6647
1	0.42	0.30	0.35	1041
2	0.45	0.10	0.17	991
3	0.32	0.37	0.34	971
4	1.00	1.00	1.00	40093
5	0.49	0.20	0.29	938
6	0.57	0.13	0.21	600
7	1.00	1.00	1.00	40152
8	0.99	0.99	0.99	39787
9	0.78	0.98	0.87	5992
10	0.83	0.63	0.72	1013
11	0.65	0.39	0.49	17033
12	0.83	0.99	0.90	3067
13	0.79	0.90	0.84	39813
14	0.75	0.97	0.84	3041
Accuracy		0.91		201179
Macro Avg	0.72	0.66	0.67	201179
Weighted Avg	0.90	0.91	0.90	201179

Table 4.11: Classification report of the FL CNN model

Result Analysis: The CNN model does a great job separating the most common traffic types, such as benign traffic, TCP floods, and MQTT floods, with nearly perfect scores for these classes. It also detects port scans and ping sweeps very well. However, it struggles with some of the rarer attacks (for example, UDP floods, ARP spoofing, and ICMP floods), where it often misses many true cases or makes more mistakes. Overall accuracy is 91% , but because the model is much better on big classes than on small ones, the average precision and recall across all classes are lower.

LSTM Model Performance Table 4.12 shows the classification metrics for the FL LSTM model, including precision, recall, F1-score, and support for each class.

Class	Precision	Recall	F1-score	Support
0	1.00	1.00	1.00	3218
1	0.47	0.45	0.46	498
2	0.74	0.35	0.47	502
3	0.40	0.61	0.48	505
4	1.00	1.00	1.00	19947
5	0.63	0.54	0.58	490
6	0.68	0.40	0.50	317
7	1.00	1.00	1.00	19913
8	0.99	0.99	0.99	19960
9	0.94	0.99	0.96	3037
10	0.84	0.74	0.79	493
11	0.76	0.78	0.77	8531
12	0.88	1.00	0.93	1498
13	0.91	0.90	0.90	20136
14	0.94	0.98	0.96	1545
Accuracy		0.95		100590
Macro Avg	0.81	0.78	0.79	100590
Weighted Avg	0.95	0.95	0.94	100590

Table 4.12: Classification report of the model on test data

Result Analysis: The LSTM model achieves an overall accuracy of 95%, improving upon the CNN's 91%. It still classifies the largest classes benign traffic (class 0), DoS TCP Flood (4), MQTT DDoS Publish Flood (7), and MQTT DoS Connect Flood (8) perfectly or nearly so. High-volume categories like MQTT DoS Publish Flood (9) and Recon Port Scan (13) also see excellent F1-scores (0.96 and 0.90, respectively). Importantly, the LSTM handles previously challenging, lower-support attacks much better: for example, DDoS UDP Flood (class 2) jumps to 0.47 F1 (from 0.17), DoS UDP Flood (5) to 0.58, MITM ARP Spoofing (6) to 0.50, and Recon OS Scan (11) to 0.77. The macro-averaged precision, recall, and F1-score are around 0.80, reflecting more balanced performance across all 15 classes.

4.3.5.5 Comparing FL Models (LSTM vs CNN) :

A comparative study was carried out to assess how well different federated learning models detect and classify cyber-attacks in Internet of Medical Things (IoMT) environments using the CIC-BCCC-NRC-IoMT-2024 dataset. We trained both a convolutional neural network (CNN) and a bidirectional LSTM under identical federated averaging settings and evaluated them on the same test split of 100 590 samples across 15 attack categories.

The LSTM model outperforms the CNN in almost every respect. It achieves 95 % overall accuracy versus 91 % for the CNN, while both models maintain near-perfect scores on dominant classes such as benign traffic (class 0), TCP floods (class 4), and MQTT floods (classes 7 and 8). The real gains appear on minority classes: for DDoS UDP Flood (class 2) the LSTM's F1-score jumps from 0.17 to 0.47, for DoS UDP Flood (class 5) from 0.29 to 0.58, for MITM ARP Spoofing (class 6) from 0.21 to 0.50, and for OS Scan (class 11) from 0.49 to 0.77. These improvements reflect the LSTM's stronger ability to capture temporal dependencies and subtle sequential patterns in network flows that the CNN may overlook.

Training curves (see Figure[4.9],[4.8]) show that the LSTM's validation accuracy rises sharply in the early rounds stabilizing above 90 % by round 10 while its validation loss declines rapidly before plateauing, indicating robust convergence under federated learning.

Despite these advances, both models still exhibit imbalanced behavior: weighted averages remain high (≈ 0.95)(Table4.14) because large classes dominate, whereas macro-averaged precision, recall, and F1-score underscore that some rare attacks remain challenging. Overall, the bidirectional LSTM offers a more balanced and robust detection performance across all attack types, suggesting that sequence-aware architectures are particularly well suited for federated intrusion detection in IoMT systems.

Table 4.13: Comparison Between LSTM and CNN Models.

Model	**Accuracy**	**Precision**	**Recall**	**F1-Score**	**Training Time**
CNN (FL)	91.20%	90.00%	91.15%	90.45%	2h
BiLSTM (FL)	95.00%	95.01%	95.20%	94.35%	3h30m

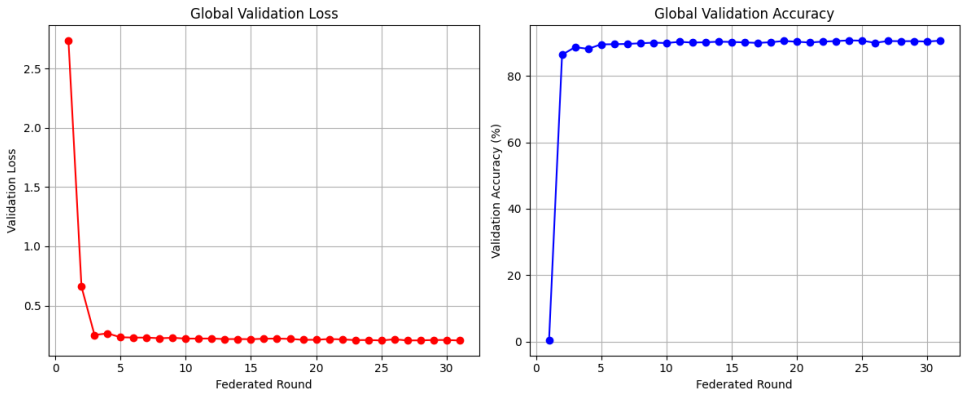

Figure 4.8: Validation Loss Accuracy (CNN FL).

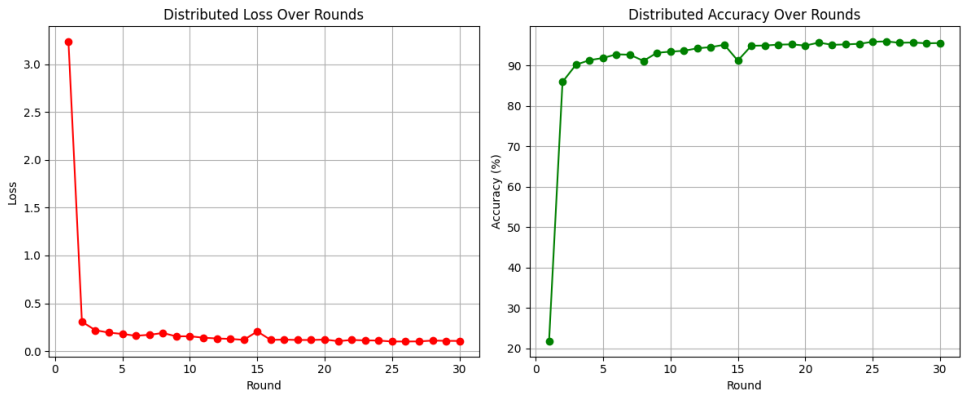

Figure 4.9: Validation Loss Accuracy (LSTM FL).

4.3.6 Why LSTM Outperforms CNN in Both Federated and Centralized IoMT Intrusion Detection

The LSTM (Long Short-Term Memory) model performs better than the CNN (Convolutional Neural Network) in this case mainly because of how it processes time-based data. In cybersecurity, especially with Internet of Medical Things (IoMT) systems, network traffic often follows a sequence for example, an attack might start with suspicious login attempts followed by data exfiltration. LSTM models are designed to remember these sequences over time. They have a special memory cell that helps them retain important information from earlier in the sequence while ignoring less useful parts. This is especially helpful when detecting subtle or rare attacks that unfold gradually.

On the other hand, CNNs are very good at identifying local patterns like recognizing shapes in an image or short bursts of activity but they don't naturally understand the order or timing of events. This limits their ability to detect attacks that depend on how network activity changes over time.

In the experiment, the LSTM showed much higher F1-scores than CNN on many minority classes, like DDoS UDP Flood, MITM ARP Spoofing, and OS Scans. These are more complex and less frequent types of attacks, where the temporal behavior matters more. LSTM could better detect these because it learned from the order of network packets and their timing patterns.

Furthermore, training curves showed that LSTM achieved higher accuracy and lower loss more quickly and consistently than CNN. This suggests that the LSTM not only learns better from sequential data but also generalizes well during federated training, even across different clients with varying data.

Overall, LSTM's ability to capture long-term dependencies and learn from the flow of events makes it especially suitable for intrusion detection in IoMT environments where data is temporal and attacks may evolve over time.

4.3.7 Comparing the Federated and Centralized LSTM Models

The centralized LSTM achieves slightly higher overall performance 97 % accuracy compared to 95 % for the federated LSTM—and scores a weighted F1 of 0.97 versus 0.94, thanks to perfect or near perfect classification of both major and many minor classes when trained on the full dataset. Its macro-averaged F1 of 0.82 also edges out the federated model's 0.79, reflecting a marginally better balance across all attack types. However, the federated LSTM comes very close in accuracy and F1 while preserving data privacy and handling IID splits across three clients. In federated training, the LSTM still boosts minority class F1-scores (e.g., from 0.17 to 0.47 for DDoS UDP Flood) and converges robustly, demonstrating that it can nearly match centralized results even when data remains decentralized.

Table 4.14: Comparison Between LSTM and CNN Models.

Model	**Accuracy**	**Precision**	**Recall**	**F1-Score**	**Training Time**
BiLSTM (CL)	97.00%	97.00%	97.00%	97.00%	1h45m
BiLSTM (FL)	95.00%	95.01%	95.20%	94.35%	3h30m

4.4 Results Discussion

The federated LSTM model demonstrated excellent capability in detecting a wide range of cyber-attacks within Internet of Medical Things (IoMT) systems. While the centralized LSTM model maintained a slight advantage due to having access to all training data at once, the federated version still achieved strong and competitive results without requiring any raw data to be shared among clients.

What makes the federated LSTM particularly effective is its ability to learn temporal patterns in local network traffic. Each participating client trains on its own environment, capturing the unique behavior and sequence of events specific to that location. These learned features are then shared in the form of model updates, not raw data, preserving privacy while enriching the global model with diverse knowledge.

The federated LSTM also promotes collaboration across hospitals and medical facilities by allowing them to collectively build a powerful intrusion detection model, even if they cannot share sensitive logs. This aligns perfectly with data protection regulations and helps institutions comply with legal and ethical standards like HIPAA and GDPR.

Moreover, the model adapts well to different environments. After global training, each site can fine-tune the LSTM locally to better fit its own traffic, creating a personalized detection system that still benefits from shared global knowledge. This flexibility and personalization are key advantages in real-world healthcare networks, where device behavior and attack types can vary widely.

In summary, the federated LSTM offers a privacy-preserving, collaborative, and robust solution for cyber-attack detection in IoMT systems. It provides strong security performance while ensuring sensitive patient and device data remains protected.

4.5 Conclusion

This chapter compared four models for detecting cyber-attacks in IoMT networks: centralized CNN, centralized LSTM, federated CNN, and federated LSTM. The LSTM models outperformed CNN models, especially in detecting time-based patterns and rare attack types. While the centralized LSTM was slightly more accurate, the federated LSTM offered strong performance while maintaining data privacy, making it the best choice for healthcare systems.

General Conclusion

The Internet of Medical Things (IoMT) is revolutionizing the healthcare industry by enabling real-time monitoring, diagnosis, and treatment through connected medical devices. While IoMT offers enhanced healthcare services, it also introduces new security vulnerabilities that could compromise patient safety and data privacy. This study presented a comprehensive overview of IoMT, its architecture, device types, applications, and the prevalent cybersecurity threats especially those highlighted in the CIC-BCCC-NRC-IoMT-2024 dataset.

To address these challenges, this work explored the use of Artificial Intelligence (AI) and Deep Learning (DL), particularly Convolutional Neural Networks (CNNs) and Long Short-Term Memory (LSTM) models, for intrusion detection in IoMT networks. Both centralized and federated learning (FL) approaches were implemented and evaluated. Experimental results demonstrated that LSTM models significantly outperformed CNN models in both centralized and federated environments, especially in detecting sequential patterns of attacks.

Furthermore, federated learning has proven to be a promising solution for preserving data privacy and enabling distributed training across multiple healthcare clients without compromising the performance of intrusion detection systems.

Future Works

Future Works

While this study demonstrates the effectiveness of deep learning, particularly LSTM models, in both centralized and federated learning settings for intrusion detection in the Internet of Medical Things (IoMT), several avenues remain for future exploration. One key direction is enhancing the robustness and scalability of federated learning frameworks to handle real-world non-IID data distributions, limited bandwidth, and resource-constrained IoMT devices. Future research could also integrate advanced privacy-preserving techniques such as differential privacy and secure multi-party computation to further safeguard sensitive medical data. Moreover, expanding the model's capability to detect zero-day attacks using continual or transfer learning methods could significantly improve security. Incorporating other deep learning architectures or hybrid models may also yield performance improvements. Lastly, testing the framework in real-time IoMT environments would provide valuable insights into its practical deployment and optimization.

Bibliography

[1] A. Al-Fuqaha et al. "Internet of Things: A Survey on Enabling Technologies, Protocols, and Applications." In: *IEEE Communications Surveys & Tutorials* 17.4 (2015), pp. 2347–2376. DOI: 10.1109/COMST.2015.2444095.

[2] B. K. Mohanta et al. "Survey on IoT security: Challenges and solution using machine learning, artificial intelligence and blockchain technology." In: *Internet of Things* 11 (2020), p. 100227. DOI: 10.1016/j.iot.2020.100227.

[3] Pallavi Sethi and Smruti R. Sarangi. "Internet of Things: Architectures, Protocols, and Applications." In: *Journal of Electrical and Computer Engineering* 2017 (2017), Article ID 9324035. DOI: 10.1155/2017/9324035.

[4] R. K. Patchmuthu, A. T. Wan, and W. S. Suhaili. "Exploring data security and privacy issues in Internet of Things based on five-layer architecture." In: *International Journal of Communication Networks and Information Security (IJCNIS)* 12.1 (2020), pp. 108–115. DOI: 10.17762/ijcnis.v12i1.4345.

[5] Ali Malkawi et al. "Design and applications of an IoT architecture for data-driven smart building operations and experimentation." In: *Energy and Buildings* 295 (2023), p. 113291. DOI: 10.1016/j.enbuild.2023.113291.

[6] P. P. Ray. "A survey on Internet of Things architectures." In: *Journal of King Saud University - Computer and Information Sciences* 30.3 (2018), pp. 291–319. DOI: 10.1016/j.jksuci.2016.10.003.

[7] L. Tawalbeh et al. "IoT Privacy and Security: Challenges and Solutions." In: *Applied Sciences* 10.12 (2020), p. 4102. DOI: 10.3390/app10124102.

[8] S. M. R. Islam et al. "The Internet of Things for Health Care: A Comprehensive Survey." In: *IEEE Access* 3 (2015), pp. 678–708. DOI: 10.1109/ACCESS.2015.2437951.

[9] S. Razdan and S. Sharma. "Internet of Medical Things (IoMT): Overview, Emerging Technologies, and Case Studies." In: *Journal of Medical Systems* 45.5 (2021), pp. 775–788. DOI: 10.1080/02564602.2021.1927863.

[10] IBM. *Medical Devices are Vital, but Vulnerable: Treat Infrastructure Risks to Safeguard Patient Care*. Tech. rep. International Business Machines Corporation, 2020. URL: https://www.ibm.com/thought-leadership/institute-business-value/report/medical-device-security.

[11] M. Elhoseny et al. "Security and privacy issues in Medical Internet of Things: Overview, countermeasures, challenges and future directions." In: *Sustainability* 13.21 (2021), p. 11645. URL: https://www.mdpi.com/2071-1050/13/21/11645.

[12] Y. K. Saheed and M. O. Arowolo. "Efficient cyber attack detection on the Internet of Medical Things-smart environment based on deep recurrent neural network and machine learning algorithms." In: *IEEE Access* 9 (2021), pp. 161546–161554. DOI: 10.1109/ACCESS.2021.3128837. URL: https://ieeexplore.ieee.org/document/9617609.

[13] M. L. Hernandez-Jaimes et al. "Artificial intelligence for IoMT security: A review of intrusion detection systems, attacks, datasets and Cloud–Fog–Edge architectures." In: *Internet of Things* 22 (2023), p. 100887. DOI: 10.1016/j.iot.2023.100887.

[14] S. Messinis et al. "Enhancing Internet of Medical Things security with artificial intelligence: A comprehensive review." In: *Computers in Biology and Medicine* 170 (2024), p. 108036. DOI: 10.1016/j.compbiomed.2024.108036.

[15] S. R. Sindhuja, A. S. Kapse, and A. S. Kapse. "A Survey of Internet of Medical Things (IoMT) Applications, Architectures and Challenges in Smart Healthcare Systems." In: *ITM Web of Conferences*. Vol. 56. 2023, p. 05013. DOI: 10.1051/itmconf/20235605013.

[16] P. Matthew et al. "A review of the state of the art for the Internet of Medical Things." In: *Sci* 7.2 (2025). [Accessed: Apr. 24, 2025], p. 36. DOI: 10.3390/sci7020036. URL: https://doi.org/10.3390/sci7020036.

[17] S. A. El-Din, M. H. Selim, and N. S. Aref. "Medical Data Analysis Based on Nao Robot: An Automated Approach Towards Robotic Real-Time Interaction with Human Body." In: *Proceedings of the IEEE International Conference on Control System, Computing and Engineering (ICCSCE)* (Nov. 24–26, 2017). Penang, Malaysia: IEEE, 2017. DOI: 10.1109/ICCSCE.2017.8284386.

[18] M. K. Hasan et al. "A review on security threats, vulnerabilities, and counter measures of 5G enabled Internet-of-Medical-Things." In: *IET Communications* (2021). [Online]. Available: https://doi.org/10.1049/cmu2.12301.

[19] F. Kamalov et al. "Internet of Medical Things Privacy and Security: Challenges, Solutions, and Future Trends from a New Perspective." In: *Sustainability* 15.4 (Feb. 2023), p. 3317. DOI: 10.3390/su15043317.

[20] Nada Abughazaleh et al. "DoS Attacks in IoT Systems and Proposed Solutions." In: *International Journal of Computer Applications* 176.33 (2020). DOI: 10.5120/ijca2020920397.

[21] W. Yao et al. "Exploiting Ensemble Learning for Edge-assisted Anomaly Detection Scheme in e-healthcare System." In: *Proc. 2021 IEEE Global Communications Conference (GLOBECOM)*. Madrid, Spain, 2021, pp. 1–6. DOI: 10.1109/GLOBECOM46510.2021.9685745.

[22] K. Sonar and H. Upadhyay. "A survey: DDOS attack on Internet of Things." In: *Int. J. Eng. Res. Dev.* 10.11 (2014). [Online]. Available: https://www.academia.edu/10741009/A_Survey_DDOS_Attack_on_Internet_of_Things, [Accessed: Apr. 18, 2025], pp. 58–63.

[23] S. M. Specht and R. B. Lee. "Distributed Denial of Service: Taxonomies of Attacks, Tools, and Countermeasures." In: *Proc. 17th Int. Conf. Parallel and Distributed Computing Systems*. [Online]. Available: https://www.researchgate.net/publication/220922510_Distributed_Denial_of_Service_Taxonomies_of_Attacks_Tools_and_Countermeasures, [Accessed: Apr. 18, 2025]. San Francisco, CA, USA, 2004, pp. 543–550.

[24] Harshita. "Detection and Prevention of ICMP Flood DDOS Attack." In: *International Journal of New Technology and Research* 3.3 (Mar. 2017). Online. URL: https://www.neliti.com/publications/263333/detection-and-prevention-of-icmp-flood-ddos-attack (visited on 04/18/2025).

[25] M. M. Alani and E. Damiani. "XRecon: An explainable IoT reconnaissance attack detection system based on ensemble learning." In: *Sensors* 23.11 (2023), p. 5298. DOI: 10.3390/s23115298.

[26] M. Uma and G. Padmavathi. "A survey on various cyber attacks and their classification." In: *International Journal of Network Security (IJNS)* 15.5 (2013), pp. 390–396.

[27] H. P. Sanghvi and M. S. Dahiya. "Cyber reconnaissance: an alarm before cyber attack." In: *International Journal of Computer Applications* 63.6 (2013). https://citeseerx.ist.psu.edu/document?repid=rep1&type=pdf&doi=02468da6fb2cf4b0fee56bc43

[28] M. Fulford. *Reconnaissance Scans: What They Are, When to Stop Them, and When to Do Them*. https://www.cybermaxx.com/resources/reconnaissance-scans-what-they-are-when-to-stop-them-and-when-to-do-them/. Accessed Mar. 2. Mar. 2022.

[29] J. J. Anthraper and J. Kotak. "Security, Privacy and Forensic Concern of MQTT Protocol." In: *Int. Conf. Sustainable Computing in Science, Technology and Management (SUSCOM)*. [Online]. Available: https://dx.doi.org/10.2139/ssrn.3355193. Amity University Rajasthan, Jaipur, India, 2019.

[30] A. N. Aroon et al. "Detection of TCP and MQTT-Based DoS/DDoS Attacks on MUD IoT Networks." In: *Electronics* 14.8 (2025), p. 1653. DOI: 10.3390/electronics14081653.

[31] A. Al Hanif and M. Ilyas. "Effective Feature Engineering Framework for Securing MQTT Protocol in IoT Environments." In: *Sensors* 24.6 (Mar. 2024), p. 1782. DOI: 10.3390/s24061782.

[32] S. L. Qaddoori and Q. I. Ali. "An Efficient Security Model for Industrial Internet of Things (IIoT) System Based on Machine Learning Principles." In: *Rafidain Engineering Journal* 28 (2023), pp. 329–340. DOI: 10.33899/rengj.2022.134932.1191.

[33] P. R. Babu, L. Bhaskari, and C. Satyanarayana. "A Comprehensive Analysis of Spoofing." In: *International Journal of Advanced Computer Science and Applications* 1.6 (2011). DOI: 10.14569/IJACSA.2010.010623.

[34] Z. Cekerevac et al. "Internet of Things and the Man-in-the-Middle attacks – Security and economic risks." In: *MEST Journal* 5.2 (2017), pp. 15–25. DOI: 10.12709/mest.05.05.02.03.

[35] B. Lundgren and N. Möller. "Defining Information Security." In: *Science and Engineering Ethics* 25.2 (2019), pp. 419–441. DOI: 10.1007/s11948-017-9992-1.

[36] Osaro Mitchell and Christopher Osazuwa. "Confidentiality, Integrity, and Availability in Network Systems: A Review of Related Literature." In: *Confidentiality, Integrity, and Availability in Network Systems: A Review of Related Literature* 8.12 (Jan. 2024), p. 10. DOI: 10.5281/zenodo.10464076.

[37] O. Salem et al. "Man-in-the-Middle Attack Mitigation in Internet of Medical Things." In: *IEEE Transactions on Industrial Informatics* 18.3 (Mar. 2022), pp. 2053–2062. DOI: 10.1109/TII.2021.3089462.

[38] L. Bracciale, P. Loreti, and G. Bianchi. "Cybersecurity Vulnerability Analysis of Medical Devices Purchased by National Health Services." In: *Scientific Reports* 13.1 (Nov. 2023). Online. DOI: 10.1038/s41598-023-45927-1. URL: https://doi.org/10.1038/s41598-023-45927-1 (visited on 04/18/2025).

[39] Kirolos Eskandar. "Artificial Intelligence in Healthcare: Explore the Applications of AI in Various Medical Domains, Such as Medical Imaging, Diagnosis, Drug Discovery, and Patient Care." In: 1 (Dec. 2023), pp. 37–53. DOI: 10.5281/zenodo.12702083.

[40] Mousa Alalhareth and Sung-Chul Hong. "An Adaptive Intrusion Detection System in the Internet of Medical Things Using Fuzzy-Based Learning." In: *Sensors* 23.22 (2023). ISSN: 1424-8220. URL: https://www.mdpi.com/1424-8220/23/22/9247.

[41] John Mulo et al. "Navigating Challenges and Harnessing Opportunities: Deep Learning Applications in Internet of Medical Things." In: *Future Internet* 17.3 (2025), p. 107. ISSN: 1999-5903. URL: https://www.mdpi.com/1999-5903/17/3/107.

[42] Iqbal H. Sarker. "Deep Learning: A Comprehensive Overview on Techniques, Taxonomy, Applications and Research Directions." In: *SN Computer Science* 2 (Aug. 2021), pp. 1–20. DOI: 10.1007/s42979-021-00815-1.

[43] GeeksforGeeks Contributors. *Introduction to Convolutional Neural Network*. https://www.geeksforgeeks.org/introduction-convolution-neural-network/. Accessed: April 27, 2025. 2021.

[44] Jeffrey L. Elman. "Finding structure in time." In: *Cognitive Science* 14.2 (1990), pp. 179–211.

[45] Sepp Hochreiter and Jürgen Schmidhuber. "Long short-term memory." In: *Neural Computation* 9.8 (1997), pp. 1735–1780.

[46] Greg Van Houdt, Carlos Mosquera, and Gonzalo Nápoles Ruiz. "A review on the long short-term memory model." In: *Artificial Intelligence Review* 53 (2020), pp. 5929–5955.

[47] Sepp Hochreiter and Jürgen Schmidhuber. "Long Short-Term Memory." In: *Neural Computation* 9.8 (Nov. 1997), pp. 1735–1780. DOI: 10.1162/neco.1997.9.8.1735.

[48] GeeksforGeeks Contributors. *Deep Learning - Introduction to Long Short-Term Memory*. https://www.geeksforgeeks.org/deep-learning-introduction-to-long-short-term-memory/. Accessed: 2025-04-29. 2020.

[49] Gökhan Akar et al. "L2D2: A Novel LSTM Model for Multi-Class Intrusion Detection Systems in the Era of IoMT." In: *IEEE Access* 13 (2025), pp. 7002–7013. DOI: 10.1109/ACCESS.2025.3526883.

[50] Piyush Kashyap. *Understanding Precision, Recall, and F1 Score Metrics*. Accessed: 2025-04-30. 2024. URL: https://medium.com/@piyushkashyap045/understanding-precision-recall-and-f1-score-metrics-ea219b908093.

[51] Subrato Bharati et al. "Federated learning: Applications, challenges and future directions." In: *International Journal of Hybrid Intelligent Systems* 18.1-2 (2022), pp. 19–35. DOI: 10.3233/HIS-220006. arXiv: 2205.09513 [cs.LG].

[52] Leiyang Zhao and Jianjun Huang. "A distribution information sharing federated learning approach for medical image data." In: *Complex & Intelligent Systems* 9 (2023), pp. 5625–5636. DOI: 10.1007/s40747-023-01035-1.

[53] Betul Yurdem et al. "Federated learning: Overview, strategies, applications, tools and future directions." In: *Heliyon* 10.19 (2024), e38137. ISSN: 2405-8440. DOI: https://doi.org/10.1016/j.heliyon.2024.e38137. URL: https://www.sciencedirect.com/science/article/pii/S2405844024141680.

[54] H. Brendan McMahan and Daniel Ramage. *Federated Learning: Collaborative Machine Learning without Centralized Training Data*. https://research.google/blog/federated-learning-collaborative-machine-learning-without-centralized-training-data/. Google Research Blog. 2017. (Visited on 05/07/2025).

[55] Nguyen H. Tran et al. "Federated Learning over Wireless Networks: Optimization Model Design and Analysis." In: *IEEE INFOCOM 2019 - IEEE Conference on Computer Communications*. 2019, pp. 1387–1395. DOI: 10.1109/INFOCOM.2019.8737464.

[56] Tuo Zhang et al. *Federated Learning for Internet of Things: Applications, Challenges, and Opportunities*. 2022. arXiv: 2111.07494 [cs.LG]. URL: https://arxiv.org/abs/2111.07494.

[57] Nancy Victor et al. *Federated Learning for IoUT: Concepts, Applications, Challenges and Opportunities*. 2022. arXiv: 2207.13976 [cs.LG]. URL: https://arxiv.org/abs/2207.13976.

[58] Tian Li et al. "Federated Learning: Challenges, Methods, and Future Directions." In: *IEEE Signal Processing Magazine* 37.3 (May 2020), pp. 50–60. ISSN: 1558-0792. DOI: 10.1109/msp.2020.2975749. URL: http://dx.doi.org/10.1109/MSP.2020.2975749.

[59] Priyanka Mary Mammen. *Federated Learning: Opportunities and Challenges*. 2021. arXiv: 2101.05428 [cs.LG]. URL: https://arxiv.org/abs/2101.05428.

[60] Qiang Yang et al. *Federated Machine Learning: Concept and Applications*. 2019. arXiv: 1902.04885 [cs.AI]. URL: https://arxiv.org/abs/1902.04885.

[61] Manzoor Ahmed Khan et al. "A journey towards fully autonomous driving - fueled by a smart communication system." In: *Vehicular Communications* 36 (2022), p. 100476. ISSN: 2214-2096. DOI: https://doi.org/10.1016/j.vehcom.2022.100476. URL: https://www.sciencedirect.com/science/article/pii/S2214209622000237.

[62] *Flower Framework main*. https://flower.ai/docs/framework/tutorial-series-what-is-federated-learning.html. Accessed: May 06, 2024. 2024.

[63] Sin Kit Lo et al. *FLRA: A Reference Architecture for Federated Learning Systems*. 2021. arXiv: 2106.11570 [cs.LG]. URL: https://arxiv.org/abs/2106.11570.

[64] Divya Shenoy, Raghavendra Bhat, and K. Krishna Prakasha. "Exploring privacy mechanisms and metrics in federated learning." In: *Artificial Intelligence Review* 58 (2025), p. 223. DOI: 10.1007/s10462-025-11170-5. URL: https://doi.org/10.1007/s10462-025-11170-5.

[65] Longbing Cao. "Beyond i.i.d.: Non-IID Thinking, Informatics, and Learning." In: *IEEE Intelligent Systems* 37.4 (2022), pp. 5–17. DOI: 10.1109/MIS.2022.3194618.

[66] G. Varoquaux and O. Colliot. "Evaluating machine learning models and their diagnostic value." In: *Machine Learning for Brain Disorders*. Ed. by Editor Name. Publisher Location: Publisher Name, 2023, pp. 601–630.

[67] Daniel M. Jimenez G. et al. *Non-IID data in Federated Learning: A Survey with Taxonomy, Metrics, Methods, Frameworks and Future Directions*. 2024. arXiv: 2411.12377 [cs.LG]. URL: https://arxiv.org/abs/2411.12377.

[68] Sita Rani et al. "Federated learning for secure IoMT-applications in smart healthcare systems: A comprehensive review." In: *Knowledge-Based Systems* 274 (2023), p. 110658. ISSN: 0950-7051. DOI: https://doi.org/10.1016/j.knosys.2023.110658. URL: https://www.sciencedirect.com/science/article/pii/S0950705123004082.

[69] Sarthak Pati et al. "Privacy preservation for federated learning in health care." In: *Patterns* 5.7 (2024), p. 100974. ISSN: 2666-3899. DOI: https://doi.org/10.1016/j.patter.2024.100974. URL: https://www.sciencedirect.com/science/article/pii/S2666389924000825.

[70] Dinh C Nguyen et al. "Federated learning for smart healthcare: A survey." In: *ACM Computing Surveys (Csur)* 55.3 (2022), pp. 1–37.

[71] G. van Rossum and The Python Software Foundation. *Python Programming Language – Official Introduction*. https://www.python.org/doc/essays/blurb/. Accessed: May 2, 2025.

[72] Google. *Google Colaboratory*. https://colab.google/. Accessed: May 2, 2025.

[73] Kaggle. *Kaggle: Your Home for Data Science*. https://www.kaggle.com/. Accessed: May 2, 2025.

[74] NumPy Developers. *NumPy: the fundamental package for scientific computing with Python*. https://numpy.org/doc/stable/index.html. Accessed: May 2, 2025.

[75] The Pandas Development Team. *Pandas: Python Data Analysis Library*. https://pandas.pydata.org/. Accessed: May 2, 2025.

[76] The Matplotlib Development Team. *Matplotlib: Visualization with Python*. https://matplotlib.org/stable/index.html. Accessed: May 2, 2025.

[77] The PyTorch Development Team. *PyTorch Documentation*. https://pytorch.org/docs/stable/index.html. Accessed: May 2, 2025.

[78] The Flower Development Team. *Flower Framework Documentation*. https://flower.ai/docs/framework/index.html. Accessed: May 2, 2025.

[79] Canadian Institute for Cybersecurity. *IoMT Dataset 2024 Canadian Institute for Cybersecurity | UNB*. https://www.unb.ca/cic/datasets/iomt-2024.html. Accessed: Apr. 3, 2025.

[80] S. A. Alasadi and W. S. Bhaya. "Review of data preprocessing techniques in data mining." In: *Journal of Engineering and Applied Sciences* 12.16 (2017), pp. 4102–4107. DOI: 10.3923/jeasci.2017.4102.4107.

[81] GeeksforGeeks. *Interquartile Range to Detect Outliers in Data*. https://www.geeksforgeeks.org/interquartile-range-to-detect-outliers-in-data/. Accessed: Apr. 3, 2025.

[82] F. Thabtah et al. "Data imbalance in classification: Experimental evaluation." In: *Information Sciences* 513 (2019), pp. 429–441. DOI: 10.1016/j.ins.2019.11.004.

[83] G. A. Pradipta, R. Wardoyo, and A. Musdholifah. "SMOTE for handling imbalanced data problem: A review." In: *2021 Sixth International Conference on Informatics and Computing (ICIC)*. Jakarta, Indonesia, 2021. DOI: 10.1109/ICIC54025.2021.9632912.

[84] A. Fernández et al. "SMOTE for Learning from Imbalanced Data: Progress and Challenges, Marking the 15-year Anniversary." In: *Journal of Artificial Intelligence Research* 61 (2018), pp. 863–905. URL: https://www.jair.org/index.php/jair/article/view/11192.

[85] Analytics Vidhya. *Overcoming class imbalance using SMOTE techniques*. https://www.analyticsvidhya.com/blog/2020/10/overcoming-class-imbalance-using-smote-techniques/. Accessed: Apr. 8, 2025.

[86] C. Maklin. *Synthetic Minority Over-sampling TEchnique (SMOTE)*. [Accessed: Avril 8, 2025]. 2025. URL: https://medium.com/@corymaklin/synthetic-minority-over-sampling-technique-smote-7d419696b88c.

[87] J. Murel and E. Kavlakoglu. *What is dimensionality reduction?* [Accessed: Avril 8, 2025]. 2025. URL: https://www.ibm.com/think/topics/dimensionality-reduction.

[88] Built In. *Step-by-step explanation of Principal Component Analysis*. [Accessed: Avril 8, 2025]. 2025. URL: https://builtin.com/data-science/step-step-explanation-principal-component-analysis.

[89] L. C. Paul, A. A. Suman, and N. Sultan. "Methodological Analysis of Principal Component Analysis (PCA) Method." In: *International Journal of Computational Engineering & Management (IJCEM)* 16.2 (Mar. 2013). URL: http://www.ijcem.org.

[90] I. T. Jolliffe and J. Cadima. "Principal component analysis: a review and recent developments." In: *Philosophical Transactions of the Royal Society A* 374.2065 (2016), p. 20150202. DOI: 10.1098/rsta.2015.0202.

[91] M. H. Gordon et al. *Covariance Matrix Preparation for Quantum Principal Component Analysis*. arXiv preprint arXiv:2104.10815. 2021. URL: https://arxiv.org/abs/2104.10815.

[92] F. L. Gewers et al. "Principal component analysis: A natural approach to data exploration." In: *ACM Computing Surveys (CSUR)* 54.4 (2021), pp. 1–34. DOI: 10.1145/3447755.

[93] H. Henderi, T. Wahyuningsih, and E. Rahwanto. "Comparison of Min-Max normalization and Z-Score Normalization in the K-nearest neighbor (kNN) Algorithm to Test the Accuracy of Types of Breast Cancer." In: *International Journal of Informatics and Information Systems* 4.1 (2021), pp. 13–20. DOI: 10.47738/ijiis.v4i1.73.

[94] C. Nwankpa et al. *Activation functions: Comparison of trends in practice and research for deep learning*. arXiv preprint arXiv:1811.03378. 2018. URL: https://arxiv.org/abs/1811.03378.

[95] M. Yeung et al. "Unified focal loss: Generalising dice and cross entropy-based losses to handle class imbalanced medical image segmentation." In: *Computerized Medical Imaging and Graphics* 95 (2022), p. 102026. DOI: 10.1016/j.compmedimag.2021.102026.

[96] A. Z. E. Boukhamla, A. Semmadi, and T. Bahhou. "Federated Learning in Internet of Medical Things (IoMT) Healthcare Applications." MA thesis. University of Kasdi Merbah – Ouargla, Algeria, 2024. URL: https://dspace.univ-ouargla.dz/jspui/handle/123456789/37350.

[97] M. Reyad, A. M. Sarhan, and M. Arafa. "A modified Adam algorithm for deep neural network optimization." In: *Neural Computing and Applications* 35.23 (Apr. 2023), pp. 1–18. DOI: 10.1007/s00521-023-08568-z.

[98] *Metrics to Evaluate Your Classification Model to Take the Right Decisions*. [Accessed: Avril 10, 2025]. 2025. URL: https://www.analyticsvidhya.com/blog/2021/07/metrics-to-evaluate-your-classification-model-to-take-the-right-decisions/.