BSCS-F24-022

03-134212-044 Malik Muhammad Huzaifa

03-134212-094 Urooba Sumbal

03-134212-007 Ahmed Sajid Butt

Leveraging AI for Real-Time Intrusion Detection in IoT-Enabled Healthcare Systems

In partial fulfilment of the requirements for the degree of

Bachelor of Science in Computer Science

Supervisor: Zunnurain Hussain

Department of Computer Sciences

Bahria University, Lahore Campus

June 2025

C e r t i f i c a t e

We accept the work contained in the report titled

"**Leveraging AI for Real-Time Intrusion Detection in IoT-Enabled Healthcare Systems**"

written by

Malik Muhmmad Huzaifa

Urooba Sumbal

Ahmed Sajid Butt

as a confirmation to the required standard for the partial fulfilment of the degree of Bachelor of Science in Computer Science.

Approved by:

Supervisor: Zunnurain Hussain

_____ (Signature)

June, 2025

DECLARATION

We hereby declare that this project report is based on our original work except for citations and quotations which have been duly acknowledged. We also declare that it has not been previously and concurrently submitted for any other degree or award at Bahria University or other institutions.

Enrolment	Name	Signature
03-134212-044	Malik Muhammad Huzaifa	
03-134212-094	Urooba Sumbal	
03-134212-007	Ahmed Sajid Butt	

Date : 05 June 2025

Specially dedicated to
my beloved mother and father
(Malik Muhammad Huzaifa)
my beloved mother, father and brother
(Urooba Sumbal)
my beloved mother, father and sister
(Ahmed Sajid Butt)

ACKNOWLEDGEMENTS

We would like to thank everyone who had contributed to the successful completion of this project. We would like to express our gratitude to our research supervisor, Mr Zunnurain Hussain for his invaluable advice, guidance and his enormous patience throughout the development of the research.

In addition, We would also like to express our gratitude to our loving parents and friends who had helped and given us encouragement.

Malik Muhammad Huzaifa
Urooba Sumbal
Ahmed Sajid Butt

Leveraging AI for Real-Time Intrusion Detection in IoT-Enabled Healthcare Systems

ABSTRACT

Healthcare has been greatly improved by the rise of the Internet of Medical Things (IoMT). Helps doctors treat patients from a distance. Still, this has meant IoMT systems are concerned with cybersecurity because they are so closely connected. More than ever, they are at risk from cyberattacks. Intrusion Detection System (IDS) systems that are currently in place have a hard time spotting insider threats and controlling the large amount needs and making sure their security measures fit new emerging threats. These issues make the system can lead to dangers for patients and data, showing that we urgently need to improve. what security features are provided for healthcare technology. For this reason, we introduce an innovative intrusion detection framework aiming to improve the detection of threats in the moment in IoMT environments. By breaking up the learning process among several IoMT devices, our method reduces the pressure on single devices and allows the system to adjusted to new needs to the changes in cyber attack techniques. The suitability of the proposed solution is proven using data from IEEE Dataport, including attack datasets and real-time monitoring information, was used. Better results when it comes to finding and identifying targets. The framework presented here not only boosts the security of IoMT devices but also creates more trust in patients the protection of healthcare technologies that are linked.

TABLE OF CONTENTS

DECLARATION **ii**
ACKNOWLEDGEMENTS **iv**
ABSTRACT **v**
TABLE OF CONTENTS **vi**
LIST OF TABLES **ix**
LIST OF FIGURES **x**
LIST OF EQUATIONS **xi**
LIST OF SYMBOLS / ABBREVIATIONS **xi**

CHAPTERS

CHAPTER 1 1
1.1 Introduction 1
1.2 Primary Scope 1
1.3 Final Deliverable of the Project and Beneficiaries 2
1.4 Optional Scope 2
1.5 Objectives 2
1.6 Novelty 3
1.7 Methodology 7
1.8 Models Comparison 8
1.9 Feasibility Plan 9
1.10 Resource Requirements 9
CHAPTER 2 11

2 LITERATURE REVIEW 11

2.1 Overview 11

2.2 Comparative Literature Review Table 11

2.3 Discussion and Identified Gaps 23

2.4 Relevance to Our Research 24

CHAPTER 3 25

3 DESIGN AND METHODOLOGY 25

3.1 System Design 25

3.2 Architecture Diagram 25

3.3 Use Case Diagram 27

3.4 Entity Relational Diagram(ERD) 29

3.5 Data Flow Diagram 33

3.6 Flowchart 40

CHAPTER 4 43

4 DATA AND EXPERIMENTS and IMPLEMENTATION 43

4.1 Libraries Used in the Project 43

4.2 Pandas 43

4.3 NumPy 44

4.4 Matplotlib & Seaborn 45

4.5 Scikit-Learn 47

4.6 TensorFlow 49

4.7 TensorFlow Federated (TFF) 51

4.8 gdown 53

4.9 Models Used in this Project 54

4.10 Convolutional Neural Network – BiLSTM 55

4.11 Gated Recurrent Unit (GRU) 57

4.12 Federated Learning Approach 58

4.13 Model Evolution 60

4.14 Dataset Description 62

CHAPTER 5 63

5 RESULTS AND DISCUSSIONS 63

5.1 Data Dictionary 63

5.2 Model Hyperparameter Setting 64

5.3 Result Analysis 65

5.4 Class Distribution 65

5.5 Feature Distributions 67

5.6 Feature Correlations 69

5.7 Federated Training Accuracy 72

5.8 Federated Training Loss 74

5.9 Class Distribution in Simulated Federated Dataset 77

5.10 Feature Correlation Heatmap 79

5.11 Scaled Feature Distribution 81

5.12 Simulated Federated Training Accuracy per Round 83

5.13 Lebel Distribution Pie Chart 85

5.14 Feature Trends Over Time (First 200 Samples) 87

5.15 Simulated Accuracy per Client over Rounds 90

CHAPTER 6 93

6 CONCLUSION AND RECOMMENDATIONS 93

6.1 Conclusion 93

6.2 Recommendations 94

6.3 Future Work 97

6.4 Summary 97

REFERENCES 98

LIST OF TABLES

TABLE	TITLE	PAGE
Table 1.1	Novelty	3
Table 1.2	Models Comparison	8
Table 2.1	Literature Review Table	11
Table 5.1	Data Dictionary	63
Table 5.2	Paramenters used for the CNN-BiLSTM and GRU models in Training	64

LIST OF FIGURES

FIGURE	TITLE	PAGE
Figure 3.1	Federated Intrusion Detection Architecture for IoMT Systems	25
Figure 3.2	Use Case Diagram	27
Figure 3.3	Entity Relationship Diagram	30
Figure 3.4	Data Flow Diagram Level 1	33
Figure 3.5	Data Flow Diagram Level 2	37
Figure 3.6	Flowchart	40
Figure 4.1	Pandas	44
Figure 4.2	Numpy	45
Figure 4.3	Matplotlib & Seaborn	47
Figure 4.4	Scikit-Learn	49
Figure 4.5	TensorFlow	51
Figure 4.6	TensorFlow Federated	53
Figure 4.7	gdown	54
Figure 5.1	Class Distribution	65
Figure 5.2	Feature Distribution	67
Figure 5.3	Feature Correlation	70
Figure 5.4	Federated Training Accuracy	72
Figure 5.5	Federated Training Loss	75
Figure 5.6	Class Distribution in Simulated Federated Dataset	77
Figure 5.7	Feature Correlation Heatmap	79

Figure 5.8 Scaled Feature Distribution 81

Figure 5.9 Simulated Federated Training Accuracy per Round 83

Figure 5.10 Label Distribution Pie Chart 85

Figure 5.11 Feature Trends Over Time (First 200 Samples) 87

Figure 5.12 Simulated Accuracy per Client over Rounds 90

LIST OF EQUATIONS

EQUATION **TITLE** **PAGE**

Equation 4.1 Binary Cross-Entropy Loss Function 55

Equation 4.2 Convolution Operation in CNN 56

Equation 4.3 GRU Update and Output Equations 58

Equation 4.4 Federated Averaging 59

Equation 5.1 Accuracy Calculation Formula 73

Equation 5.2 Global Federated Loss Aggregation 74

LIST OF SYMBOLS / ABBREVIATIONS

IoMT:	Internet of Medical Things
IoT	Internet of Things
IDS:	Intrusion Detection System(s)
DDoS:	Distributed Denial of Service
FL:	Federated Learning
CNN:	Convolutional Neural Network
BiLSTM:	Bidirectional Long Short-Term Memory
GRU:	Gated Recurrent Unit
ML:	Machine Learning
SVM:	Support Vector Machine
RF:	Random Forest

DL:	Deep Learning
DNN:	Deep Neural Network
ANN:	Artificial Neural Network
SDN:	Software Defined Network
SHS:	Smart Healthcare System
XAI:	Explainable Artificial Intelligence
IoHT:	Internet of Health Things
ECU-IoHT:	ECU Internet of Health Things (dataset)
WSN:	Wireless Sensor Network
PSO:	Particle Swarm Optimization
SHAP:	SHapley Additive exPlanations
TFF:	TensorFlow Federated
FedAvg:	Federated Averaging
SMOTE:	Synthetic Minority Over-sampling Technique
BCE:	Binary Cross-Entropy
ADASYN	Adaptive Synthetic Sampling
RNNs	Recurrent Neural Networks
LIME	Local Interpretable Model-Agnostic Explanations
PCA	Principal Component Analysis
IQR	Interquartile Range

CHAPTER 1

1.1 Introduction

The IoMT revolutionizes healthcare by enabling the seamless collection, processing, and transmission of sensitive medical data, improving patient outcomes. Despite these benefits, the integration of IoMT introduces critical cybersecurity vulnerabilities, as IoMT devices are prime targets for cyberattacks such as insider threats, Distributed Denial of Service (DDoS), and ransomware. Traditional IDS fall short in addressing the specific needs of IoMT environments, primarily due to the computational limitations of IoT devices and the evolving nature of cyber threats.

Our project aims to address these challenges by leveraging federated learning (FL)**,** a decentralized machine learning (ML) framework, to enhance the real-time detection of cyber threats in IoMT systems. FL allows IoMT devices to collectively train a global intrusion detection model without revealing their sensitive data, which can be used to protect privacy and save computational burden. We use a fusion of CUHK's Convolutional Neural Networks (CNNs) with Bidirectional Long Short Term Memory (BiLSTM) and Gated Recurrent Unit (GRU) models to extract representations of the data, attack types that are spatially and temporally-oriented.

1.2 Primary Scope

This work endeavours to develop an intrusion IDS for IoT-based healthcare scenarios with emphasis on the IoMT. The FL-based system, enabling decentralized training of ML models at IoMT devices, without the encumbrance of centralizing the private data. This methodology reduces the computational load on IoMT devices and enhances the systems capacity to identify both insider-based threats and newly emerging attack vectors. IoMT devices have already become indispensable tools in patient monitoring, making healthcare systems an inevitable target to advanced cyber attacks. Insider attacks, where a legitimate user misuse his/her access, is a major threat that many classical IDSs cannot detect. Besides, IoMT devices are usually of low computational capacity such that the advanced ML-based IDS solutions are not applicable. Since ransomware and DDoS attacks are occurring more often, we must find a solution that

grows, is flexible and conserves resources. Thanks to FL, the proposed IDS lets IoMT devices learn together using a single, shared global model, without revealing the data involved. They rely on CNN-BiLSTM Deep Learning (DL) to improve how accurately attacks are detected. Because FL is used together with lightweight DL algorithms, the system stays efficient, scalable and can tackle the changing threats to IoMT units.

1.3 Final Deliverable of the Project and Beneficiaries

The result is a self-learning IDS built for healthcare in the IoMT, using FL to ensure accurate detection and keep resource utilization low. Healthcare providers, hospitals and patients making use of IoMT devices for rapid monitoring and care are considered the main users.

1.4 Optional Scope

Secure sharing on Blockchain ensures that health data is only used by proper healthcare providers and devices.

1.5 Objectives

1. Create an IDS using FL techniques that works in IoT-connected healthcare systems.
2. Work on understanding how to find insider threats in networks of the IoMT.
3. Reduce how much computation is needed with the help of distributed learning.
4. Improve the ability to detect threats that are just beginning to appear.

1.6 Novelty

Table 1.1 Novelty

Model/ Approach	Algorithm Used	Type (Anomaly/Signature-based)	Real-time Capability	Focus on IoT in Healthcare	Key Features	Strengths	Weaknesses	Suggested Novelty
ML-based IDS (e.g., SVM, RF)[1]	Support Vector Machine(SVM), Random Forest(RF)	Anomaly-based	Moderate	Limited	Supervised learning models, rule-based classification, feature scalability	Good accuracy for known patterns	High false positives for unknown attacks	Real-time detection with GRU/Transformer for dynamic threat modeling in healthcare.
DL (DNN/ LSTM) [2]	Deep Neural Network(DNN), LSTM	Anomaly-based	High	Some health care focus	Time-series analysis, LSTM for sequential dependencies,	High detection accuracy, good for IoT	High computational cost, not optimized for IoT	Optimized for IoT-constrained environments with FL for decentralized

					ability to detect sophisticated patterns			detection without central data storage.
Hybrid IDS[3]	Combining ML with rule-based approaches	Hybrid	Limited	Moderate	Mix of rule-based and ML for detection, combining signature and anomaly detection	Combines benefits of anomaly and signature-based	Complex and slow in real-time	Real-time anomaly detection and FL to handle IoT constraints and enhance system scalability.
Blockchain-Based IDS [4]	Blockchain for secure communication	Anomaly-based	Moderate	Limited to communication security	Decentralized data storage, auditability of logs,	Immutable data for logs, improves trust	Slower performance due to blockchain overhead	Lightweight blockchain protocols alongside AI for both

					enhanced communication security			intrusion detection and secure communication in healthcare systems.
Reinforcement Learning IDS [5]	Reinforcement Learning (RL) algorithms	Anomaly-based	High	Minimal in healthcare	Continuous learning, adaptive threat detection, learns from the environment to detect novel attack patterns	Adapts to evolving threats, continuous learning	Computationally intensive for IoT devices	RL models optimized for constrained healthcare IoT, integrating NLP for anomaly detection based on patient data inputs.

AI-Powered Real-Time IDS	GRU and CNN-BiLSTM with FL	Anomaly-based	High	Strong focus on IoT-enabled health care systems	Features: GRU and CNN-BiLSTM for real-time detection – FL for decentralized data handling – AI-driven anomaly detection for health care-specific threats	Optimized for real-time detection, decentralized data handling	Increased complexity in managing FL processes across multiple devices	Novelty: Combines CNN-BiLSTM or GRU with FL for real-time decentralized detection, specifically designed for IoT healthcare environments using AI for evolving threat detection.

This approach brings together CNN-BLSTM, GRU and FL to enable decentralized real-time detection of intrusions in the IoT healthcare environment. Simply put, training on the device with limited data cuts data transfer and makes data more private.

The solution is created to tackle ongoing health-related security problems accurately and effortlessly as they change. This project is special in that no similar Final Year Project (FYP) has ever been approved at BULC in the area of IoT healthcare cybersecurity.

1.7 Methodology

DS used in IoT-enabled healthcare systems applies several layers and a well-structured approach to notice attacks as soon as possible and then prevent them. An FL framework with AI models such as CNN-BiLSTM or GRU, is used in the system to detect anomalies quickly, accurately and on a wide scale for unconnected devices. The first part of the methodology is to continue to gather information from many IoT healthcare devices. They cover patient monitoring tools, the communication between medical devices and network patterns as they occur, catching both regular and irregular activity. After collection, the data goes through a phase to remove noise, make the data consistent and find important information about network usage and time patterns needed to spot threats. The system relies on DL methods for live detection. At first, CNNs in the architecture extract features from images and then BiLSTM layers are used to study how information changes over time. It becomes very relevant when identifying patterns of attacks that develop as time goes on. GRUs can be integrated into the system because they save valuable resources and are more appropriate for constrained IoMT networks. GRUs may be easy to use, yet they keep up with sequential data analysis and provide fast feedback. The type of model used, CNN-BiLSTM or GRU, depends on what kind of balance is needed between accurate detection and resource usage.

FL is a key element of the system's structure. Rather than sending important data to one area, every IoMT device trains itself and only passes the parameters it learned. By combining various models, a strong global model is created so diverse data can be explored safely and privately. This method is efficient with bandwidth and reduces the burden devices have to handle, making it ideal for use in real time. After being trained, the IDS watches over live data coming in to detect unexpected results. If it detects anything suspicious, it instantly sets off a response process. Whenever an anomaly is found, the system lets the proper administrators know, who may disconnect affected devices, limit malicious traffic or reset the system back to a normal state. If all data

looks normal, the system simply continues to monitor the system. The adaptation and readiness of an IDS depend on feedback loops. When everything is working as it should, data collects and, from time to time, is used in the FL cycle. This process enables the system to adapt to new data patterns and developing cyber threats, make itself more secure and preserve patient data and device operation. The IDS model uses three separate datasets: Patient Monitoring which includes health metrics and sensor information, Environment Monitoring which shows environmental data that could warn about threats and Attack which helps identify DDoS, ransomware and insider attack risks. Every dataset is pre-processed ahead of being used within the FL pipeline. Taking the datasets from IEEE Dataport [6] meant we could ensure that the real-time IDS in healthcare is based on high-quality, relevant information.

1.8 Models Comparison

Table 1.2 Models Comparison

Criteria	CNN-BiLSTM	GRU
Architecture	Combines CNN for feature extraction and BiLSTM for temporal analysis	Uses a simpler gated mechanism to handle sequential data
Complexity	More complex due to combining CNN and LSTM layers	Less complex, requires fewer parameters and computations
Training Speed	Slower due to high computational demands	Faster due to simpler architecture
Memory Usage	High memory consumption due to CNN and LSTM layers	Lower memory usage, ideal for resource-constrained environments
Accuracy	Typically provides higher accuracy, especially for longer sequences	Can be slightly less accurate than BiLSTM but performs well in many cases
Best for	IoMT systems where long-term dependencies need to be captured	IoMT systems with limited computational

		power, faster real-time applications
Handling Sequential Data	Excellent for capturing long-term dependencies in time-series data	Efficient for short to medium-length sequences
Real-Time Performance	Good, but slower due to the combination of CNN and BiLSTM	Excellent, faster real-time response due to lightweight architecture
Suitability in IoMT	Best for critical patient monitoring where deep temporal insights are needed	Best for low-power IoMT devices requiring faster response times and less memory
Resource Usage	Higher resource usage, might need optimization for IoMT	Lower resource usage, ideal for distributed systems like FL

1.9 Feasibility Plan

The proposed system is technically feasible due to the team's proficiency in Python and familiarity with widely used ML frameworks such as TensorFlow and PyTorch. The availability of IoT-healthcare datasets from platforms like IEEE Dataport, along with access to cloud resources like Google Colab, further facilitates development. Scalability concerns are mitigated by distributing computational loads between IoMT devices and a central server, ensuring efficient resource management. Team members have a high level of knowledge and skill in both of these areas. Following a timeline set by achieving certain goals helps you complete the project step by step. Monitoring and evaluating can also be greatly supported by using Wireshark to handle network traffic and MQTT for IoT communication. Since much is made with free software and basic hardware, the financial aspect is strong and costs remain low.

1.10 Resource Requirements

Thanks to its experts in FL, DL and IoT, the project team can develop a smart and safe IDS framework. You need Python, together with TensorFlow, PyTorch and Scikit-learn, to install the necessary software stack. Google Colab is the main place where you can use computational resources and Wireshark helps you capture and examine

traffic on your network. MQTT is chosen as the communication protocol for IoT to support both devices interacting and sending data. It is estimated that the total amount needed for the project is around 80,000 PKR which should cover the purchase of software tools and cloud services in addition to using the essential data. Since IoMT devices might be weaker than DL models, this could stop complex DL from being fully implemented. Even so, FL helps reduce this risk as computing takes place on a single server, while information stays private on individual devices.

CHAPTER 2

LITERATURE REVIEW

2.1 Overview

This chapter looks at existing research that involves IDS and IoT in healthcare. Studies making use of AI, ML, DL, FL and blockchain are the main concern. The key features to be developed, techniques planned, results found and gaps observed are highlighted to define what the proposed system will achieve.

2.2 Comparative Literature Review Table

Table 2.1 Literature Review Table

Authors /Year	**Objectives**	**Key findings**	**Gaps Identified**	**Study purpose**	**Research methods**	**Similar works referenced**
Rbah Yahya et al., 2023[7]	To enhance cybersecurity in IoMT using Software-Defined Networking (SDN) and DL models	Proposed a hybrid framework combining SDN with DNNs and Long Short-Term Memory (LSTM) networks	Achieved high accuracy in detecting cyber threats in IoMT; showed improved detection rates compared to conventional IDS	Faced challenges with real-time processing and scalability in constrained IoMT devices	Develop an intelligent IDS framework tailored for IoMT security using modern DL techniques	Implementation of DNN and LSTM in SDN architecture; evaluated using benchmark IoMT datasets
Khalil, Malik,	To compare	Review and evaluation of	Blockchain provides	Limited practical	To provide context and	Comparative study;

Uddin, Chen, 2022[8]	blockchain and centralized authentication systems for IoT security.	authentication mechanisms in IoT using blockchain and centralized models.	robust, decentralized security; centralized systems face scalability and privacy issues.	implementations of blockchain in smart city IoT systems.	a comparative analysis of IoT authentication methods.	evaluation of security mechanisms.
Algethami, Alshamrani, 2024[9]	Develop a hybrid DL-based IDS for Internet of Health Things (IoHT) cybersecurity.	Combining Artificial Neural Network (ANN), BLSTM, and GRU architectures tested on ECU Internet of Health Things (ECU-IoHT) dataset.	Achieved high accuracy in binary and multi-class attack detection.	Need for broader dataset validation and addressing IoT resource constraints.	Identify advanced intrusion detection approaches for IoHT.	Hybrid DL; dataset testing.
Patel and Dwivedi, 2024[10]	To explore the integration of blockchain technology with IoT in healthcare for secure monitoring systems.	A review of existing studies integrating IoT and blockchain in healthcare.	Blockchain ensures decentralized, transparent, and secure data management; IoT provides real-time monitoring.	Scalability issues with blockchain in healthcare IoT and need for efficient encryption techniques.	To analyze and highlight the benefits and challenges of blockchain-IoT synergy in healthcare.	Literature review with a focus on IoT, blockchain, and security.

Hazman, Guezzaz, Benkirane, Azrour, 2024[11]	To enhance IoT-enabled smart cities' security using a hybrid model integrating LSTM and XGBoost.	Proposed a hybrid intrusion detection framework using Pearson Correlation Coefficient, LSTM, and XGBoost, evaluated on Bot-IoT and EdgeIIoT datasets.	Improved accuracy and reduced computational complexity in intrusion detection for IoT systems.	Challenges in addressing diverse attack surfaces and real-time implementation constraints.	To develop robust IDS tailored for dynamic IoT environments in smart cities.	ML integration with advanced feature selection techniques.
Javeed, Gao, Saeed, Kumar, Jolfaei, 2023[12]	Design a CUDA-powered IDS for IoT-enabled smart healthcare systems.	Used hybrid cuLSTM-DNN and OpenStack Tacker in an SDN environment, validated with the CICDDoS2019 dataset.	Achieved superior performance in threat detection over baseline models.	Requires optimization for broader IoT environments and scalability testing.	Enhance IoT- Smart Healthcare System (SHS) security through novel software-defined approaches.	Hybrid DL models with software-defined networking.
Arisdakessian, Wahab, Mourad, Otrok, Guizani,	Comprehensive survey on IoT intrusion detection integrating emerging	Survey and taxonomy development for IDS techniques in IoT ecosystems,	Highlighted the potential of multidisciplinary approaches for	Need for real-world implementation and validation of proposed	To explore future directions for IoT IDS and provide a comprehens	Theoretical and taxonomic analysis of existing IDS techniques.

2023[13]	concepts like FL, Explainable Artificial Intelligence (XAI), and social psychology.	covering fog and cloud layers.	advanced intrusion detection.	frameworks.	ive classification of existing approaches.	
Chaudhary, Kakkar, Jadav, Nair, Gupta, Tanwar, et al., 2022[14]	Develop a taxonomy for smart healthcare technologies emphasizing security challenges and solutions.	Survey of smart healthcare systems and evaluation of AI-based 6G frameworks through a COVID-19 case study.	Proposed AI-enabled security frameworks to address latency and data integrity issues.	Limited focus on real-time scalability and attack resilience in dynamic environments.	Highlight security challenges in smart healthcare systems and propose efficient frameworks.	Taxonomy development and experimental analysis of AI-based healthcare models.
Babajide J Asaju, 2024[15]	Explore AI and ML integration into IDS for V2X environments to counter cyber threats.	Analysis of IDS requirements for dynamic vehicular networks; integration of AI and ML for real-time detection.	AI/ML-based IDS improves threat detection and adaptability in V2X networks.	Real-time efficiency in high-traffic scenarios and broader dataset validation are lacking.	Highlight the role of AI/ML in vehicular IDS for robust security mechanisms.	Literature synthesis and framework proposals for vehicular networks.
Mohammad	Address botnet	Implemented over 25 ML	Certain ML models	Validation on diverse	To benchmark	Comprehensive

Shahin, Mazdak Maghanaki, Ali Hosseinzadeh, 2024[16]	threats in Industrial IoT using advanced AI-enabled IDS.	algorithms to evaluate IoT security models across devices.	showed near-perfect accuracy; others require improvement.	datasets and insights into IoT device function influences.	AI/ML models for intrusion detection.	evaluation of ML models on industrial IoT security datasets.
Dr. Bhuvana J, Dr. Saroj Kumar, Dr. M. Deepa, Dr. Pavithra G, Dr. V. Anandkumar, 2023[17]	Leverage IoT and ML to enhance healthcare monitoring and decision-making systems.	Implemented ML algorithms for predictive healthcare analytics and IoT integration.	ML and IoT together improve real-time patient monitoring and predictive healthcare decisions.	Scalability issues in IoT-enabled health prediction systems remain unaddressed.	Examine the role of AI/ML in IoT-based health systems for improving patient care.	Implementation of ML techniques in IoT healthcare systems.
G. Indra, E. Nirmala, G. Nirmala, P. Gururama Senthilvel,	Propose an EGR-LSS algorithm for intrusion detection in IoT-based smart cities.	Combined Gradient Boosting and RF techniques optimized by Leopard Seal Search algorithm; tested on	Achieved high accuracy and efficiency in intrusion detection for IoT networks.	Requires further real-world testing and validation for diverse IoT environments.	Introduce novel ensemble learning techniques for improving IoT security.	Ensemble learning approach and optimization techniques.

2024[18]		CICIDS2017 dataset.				
Ahamed Aljuhani, Abdulelah Alamri, Prabhat Kumar, Alireza Jolfaei, 2024[19]	To design an explainable and efficient IDS for IoMT, leveraging edge computing and SaaS deployment.	Implemented Particle Swarm Optimization (PSO) for feature engineering, SHAP for explainability, and combined ML/DL models for detection; validated on WUSTL-EHMS-2020 dataset.	Achieved superior attack detection efficiency, improving transparency and usability for healthcare IoMT.	Scalability and adaptation to diverse IoMT configurations need further exploration.	Enhance security and interpretability of IDS in resource-constrained IoMT devices.	Feature engineering with PSO, SHAP for explainability, SaaS-based edge computing.
Ali Hamza Najim, Kareem Ali Malalah Al-Sharhanee, et al., 2024[20]	Develop a 5G-IoT integrated framework for patient monitoring using ANN-based health systems.	Implemented IoT and Wireless Sensor Network (WSN) technologies for real-time data collection and ANN-based prediction.	Demonstrated accuracy improvements in health monitoring with integrated IoT and DL models.	Requires optimization for broader health parameters and real-world deployment.	Enhance patient care via intelligent IoT frameworks.	IoT-5G integration with ANN-based prediction systems.

Mireya Hernandez-Jaimes, Alfonso Martinez-Cruz, et al., 2023[21]	Comprehensive review of AI-driven IDS for IoMT, analyzing datasets, architectures, and emerging trends.	Surveyed existing literature; introduced taxonomy for IDS in Cloud-Fog-Edge architectures.	Highlighted challenges and emerging trends in integrating AI techniques with IoMT security.	Lack of standardization and real-time adaptability in AI-driven IDS for IoMT.	Classify and analyze AI methods and challenges in IoMT security.	Taxonomic and comparative analysis of IDS architectures and datasets.
Sarah Alzakari, Arindam Sarkar, Mohammad Zubair Khan, Amel Ali Alhussan, 2024[22]	Introduce a FL-based framework for secure health prediction and intrusion detection in IoHT.	Combined FL and blockchain; used Matrix-Valued Neural Coordinated Federated Intelligence for security.	Achieved 98% accuracy in intrusion detection and disease prediction with FL-guided smart health networks.	Addressing FL's data sharing incentives and scalability issues remains challenging.	Integrate FL and blockchain to enhance IoHT security and predictive capabilities.	FL and blockchain integration for secure healthcare networks.
Aitizaz Ali, Muhammad Fermi Pasha, Jehad	To enhance blockchain-based healthcare systems by using DL-driven	Utilized hyperledger fabric and developed a secure searchable encryption	Achieved improved security and privacy in blockchain healthcare with	Limited scalability for large-scale real-world deployments.	Enhance security and privacy in healthcare blockchain using advanced	Homomorphic encryption, blockchain integration, and DL techniques.

Ali, et al., 2022[23]	homomorphic encryption for secure keyword search.	framework with smart contracts.	efficient data sharing and retrieval.		cryptographic techniques.	
Irfan Ali Kandhro, Sultan M. Alanazi, et al., 2023[24]	Develop a DL-based IDS framework for detecting real-time intrusions in IoT networks.	Proposed a framework using GAN, RNN, and DNN models validated on NSL-KDD and UNSW-NB15 datasets.	Achieved 95%-97% detection accuracy with enhanced reliability and efficiency.	High false positives and resource requirements for real-time deployment.	Improve intrusion detection accuracy and reduce false positives in IoT cybersecurity.	DL with multi-layer perceptron and recurrent neural networks(RNNs).
Run Yang, Hui He, et al., 2023[25]	Propose a cloud–edge collaboration method for efficient intrusion detection in IoT networks.	Implemented SSAE and TCN models in a FL framework for distributed detection.	Reduced computational workload by 50% while maintaining high accuracy.	Limited edge device resource availability for large-scale applications.	Enhance privacy-preserving, collaborative intrusion detection for IoT.	Cloud-edge collaboration and FL with DL techniques.
Hatem A. Alharbi, Khulud K. Alharbi, et al.,	Develop a non-invasive fall detection system using IoT-enabled	Employed RFID technology with machine and DL classifiers (e.g., KNN, RF, GRU).	Achieved 99% accuracy using KNN; provided real-time detection with	Requires scalability testing and validation for broader applications.	Promote safe, independent living for elderly individuals through non-	RFID-based IoT systems with ML/DL classifiers.

2023[26]	smart carpets integrated with AI.		minimal intrusion.		invasive fall detection.	
Nour Moustafa, Nickolaos Koroniotis, et al., 2023[37]	To explore the role of XAI in improving IoT-based IDS.	Surveyed XAI and IDS techniques, analyzed challenges, and proposed future research directions.	XAI enhances trust and interpretability of IDS but lacks standardization and scalability in IoT environments.	Limited integration of XAI in real-time IoT scenarios and interoperability challenges.	Promote transparency and reliability in IoT-based IDS using XAI.	Comprehensive survey and taxonomy development.
Sagarkumar K. Patel, 2023[38]	Develop an intrusion detection framework using neural networks for cloud-based healthcare.	Used hybrid tempest optimization with neural networks for arrhythmia classification and IDS.	Achieved high sensitivity (96.78%) and specificity (95.29%) in cloud-based IDS.	Requires scalability testing for broader deployment.	Improve healthcare data security in cloud environments.	Neural network optimization with tempest algorithms.
Mohamed Faisal Elrawy, Ali Ismail Awad, et al.,	Survey existing IDS frameworks and challenges in IoT-	Comprehensive review of IDS techniques and IoT architectures.	Traditional IDS are inadequate for IoT; new solutions must address	Limited focus on real-time IDS performance in diverse	Explore design considerations for IDS in smart IoT environments.	Survey and comparative analysis.

2018[39]	based smart environments.		scalability and energy efficiency.	IoT settings.		
Sevban Duran, Hazal Nur Marim Akpinar, et al., 2024[30]	Detect intrusion attacks in IoT healthcare systems using ML and DL models.	Compared ML and DL models (e.g., KNN, ANN, DNN) on IoT healthcare security datasets.	Achieved 99.72% accuracy using KNN; identified effectiveness of ML/DL for IoT security.	Limited evaluation on real-world IoT healthcare data.	Enhance IoT healthcare security through advanced learning techniques.	Comparative analysis of ML and DL.
Subiksha Srinivasa Gopalan, Dr Ali Raza, Dr Wesam Almobaideen, 2020[31]	Survey and analyze AI methods for addressing IoT security issues in healthcare.	Reviewed IoT security frameworks, categorized AI methods, and highlighted gaps from 2014 to 2019.	Identified AI's potential to mitigate IoT security challenges in healthcare; emphasized need for real-world implementation.	Limited real-world deployment and integration of AI methods in healthcare IoT security.	Highlight AI's role in enhancing IoT security and propose future research directions.	Comprehensive survey and literature review.
Nawaf Alharbe, Manal Almalki, 2024[32]	Explore IoT-enabled healthcare transformation using CNNs and LSTMs for	Integrated CNNs and LSTMs with IoT devices for real-time health data analysis and prediction.	Demonstrated effectiveness in disease detection and real-time patient monitoring.	Challenges in interpretability and resource optimization for real-time	Improve healthcare diagnostics and monitoring through IoT and DL.	IoT integration with CNN and LSTM models.

	patient monitoring and diagnosis.			IoT healthcare.		
Angela-Tafadzwa Shumba et al., 2022[33]	Develop scalable healthcare frameworks leveraging IoT and Edge AI for critical applications.	Defined system architecture integrating Edge AI, wearable sensors, and modular frameworks.	Improved response times, privacy, and accuracy in healthcare applications.	Adoption challenges for Edge AI in healthcare frameworks.	Propose scalable, efficient IoT healthcare frameworks.	Edge computing integration with AI for IoT frameworks.
Mohd Javaid et al., 2024[34]	Examine Lean 4.0 technologies and their applications in optimizing healthcare workflows.	Reviewed Lean 4.0 frameworks and their integration with IoT, AI, and data analytics.	Improved efficiency, reduced waste, and enhanced decision-making in healthcare.	Challenges in integrating digital technologies with healthcare practices.	Enhance healthcare workflows through Lean 4.0 technologies.	Literature review of Lean 4.0 implementations.
Randhir Kumar et al., 2022[35]	Integrate blockchain with DL for secure data sharing in industrial healthcare systems.	Developed the PBDL framework using stacked sparse variational autoencoder (SSVAE) and SA-BiLSTM.	Enhanced data privacy, integrity, and scalability for IoT healthcare networks.	Scalability in real-world deployment is yet to be tested.	Secure healthcare data sharing and improve detection of cyber threats.	Blockchain integration with DL models for IoT data analysis.

Najma Taimoor, Semeen Rehman, 2021[36]	Survey AI and IoT techniques for personalized healthcare services.	Reviewed AI/non-AI-based methods; proposed Healthcare 5.0 framework.	AI enables reliable and resilient personalized healthcare but lacks scalability.	Integration challenges for AI personalization in healthcare services.	Highlight AI-driven personalization and propose a taxonomy for Healthcare 5.0.	Comparative analysis of AI methods in IoT healthcare.
Mariam Ibrahim et al., 2024[37]	Develop an AI-based IDS for IoMT to mitigate cybersecurity risks.	Used multinomial Naive Bayes classifier for intrusion detection.	High accuracy in anomaly detection; low false positive rates.	Testing required on diverse IoMT environments.	Enhance IoMT security with lightweight AI techniques.	Implementation of Naive Bayes in IoMT cybersecurity.
Vinayakumar Ravi et al., 2023[38]	Analyze DL techniques for IoT-enabled healthcare systems to enhance prediction accuracy and reduce execution time.	Developed the MAR-ERNN framework to minimize the vanishing gradient issue and integrate temporal-spatial factor analysis.	Achieved 95% prediction accuracy and reduced execution time by 18%.	Early disease prediction and integration of spatio-temporal analysis require further exploration.	Optimize healthcare analytics using advanced DL techniques in IoT systems.	MAR-ERNN framework with temporal and spatial analysis.
Mohammed Saleh Ali	Propose a hybrid SDN-enabled	Used cuLSTMGRU with CICIDS2017	Achieved 99.23% detection accuracy	Scalability across diverse IoT	Enhance IoT security using intelligent	Hybrid DL models with SDN

Muthanna et al., 2022[39]	framework leveraging cuLSTMGRU for IoT intrusion detection.	dataset; implemented standard evaluation metrics.	with low false positives.	environments requires validation.	SDN and DL frameworks.	integration for IDS.

2.3 Discussion and Identified Gaps

A number of key topics and gaps were identified in the reviewed studies about IDS in the IoMT. Significantly better results in detection are being seen when using techniques such as CNN, BiLSTM and GRU. They are able to spot threats more easily than traditional approaches by learning from the information in the data. Still, the large amount of computing resources needed makes it difficult to run these models smoothly on devices used in low-resource areas of the IoMT [9][10][11]. Video privacy is now being addressed by emerging work with FL. FL offers a way to teach models on devices spread across different places, allowing patient privacy to remain a key aspect. While promising, FL shows little use in practical healthcare security systems, pointing to a difference between what is studied and what is applied [11][12]. It suggests that more must be done to match FL's design to the distinct challenges found in healthcare using IoMT devices. Furthermore, experts are studying ways to secure and improve trust in IDS using blockchain technology. Because blockchain operates in a decentralized way and cannot be changed, it excels in managing logs and guaranteeing data integrity. However, applying it is hard because of scalability and interoperability problems when working in networks where many different devices and platforms cooperate with each other [8][10]. Besides, the robustness of detection can be increased with ensemble and hybrid ML models by combining the advantages of many different algorithms. Often, these approaches give better outcomes and work well for new data. But this development adds to the computing challenges and raises the energy cost which puts their usage in slim and responsive IoMT systems in question [9][13].

Collectively, these insights underscore the pressing need for the development of an efficient, decentralized, real-time IDS tailored to the limitations and operational demands of IoMT systems. Such a solution should strike a balance between

performance, scalability, privacy, and resource consumption, bridging the current gap between advanced theoretical models and their practical, scalable implementation in healthcare environments.

2.4 Relevance to Our Research

The reviewed literature forms the basis of our project's direction. Our work builds on the hybrid DL strategies of Algethami and Alshamrani [9], and applies them in a FL setup as suggested by Arisdakessian et al. [13]. Unlike existing models, our solution directly addresses the challenges of decentralization, privacy, and lightweight implementation in real-world healthcare settings.

CHAPTER 3

DESIGN AND METHODOLOGY

3.1 System Design

This section describes the overall design of the IDS tailored for IoT-enabled healthcare environments. The system integrates AI-driven detection models within a FL framework, ensuring privacy-preserving, real-time threat monitoring and response.

3.2 Architecture Diagram

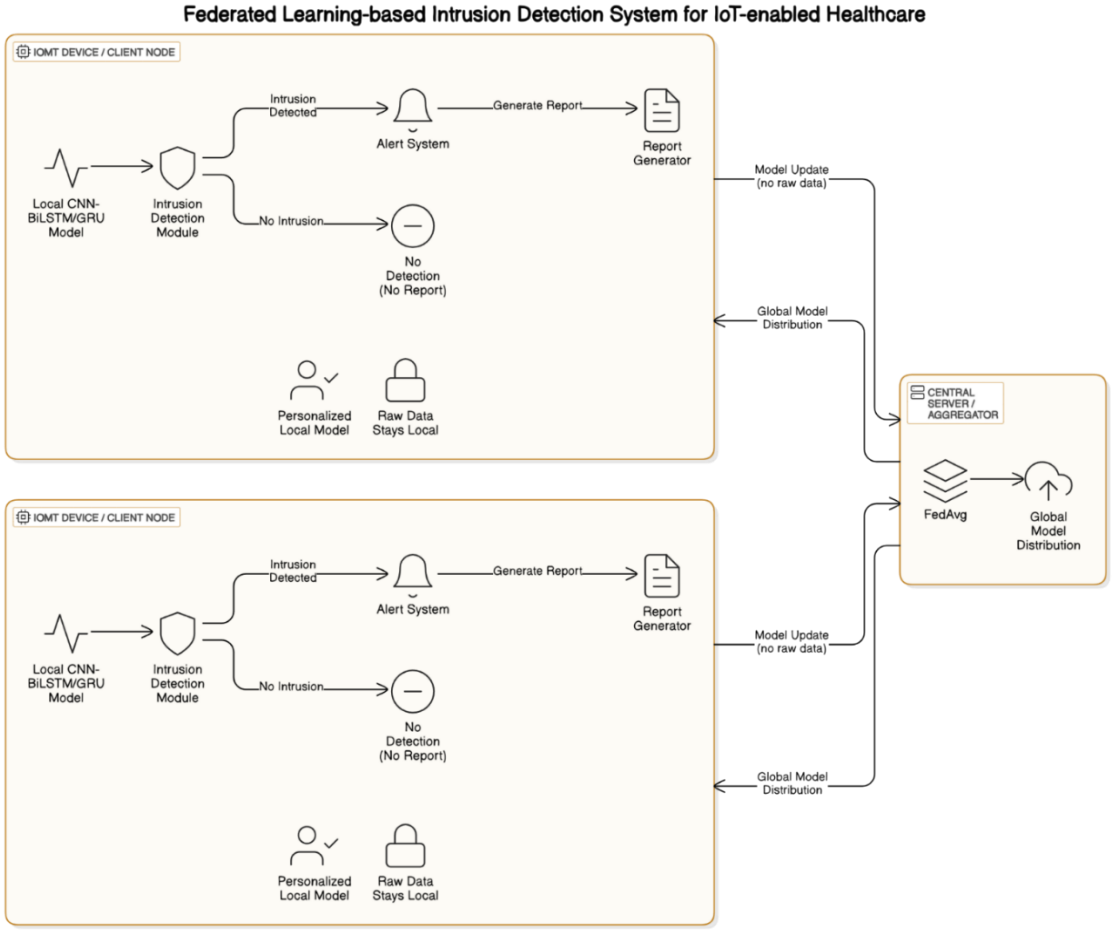

Figure 3.1 Federated Intrusion Detection Architecture for IoMT Systems

The figure above represents an FL-based IDS designed to work in IoMT environments. The system includes distributed IoMT devices acting as client nodes and a centralized device as the federated aggregator. The architecture enables people to train DL models together, safely, on different devices and keep their raw data within their own devices. Each client device stores a lightweight, customized intrusion detector developed with either CNN-BiLSTM or GRU architecture. The model is designed to track information from patients or current network traffic at all times. They are sent to a local IDS which separates the behaviour into normal or suspected intrusion. If something unusual is detected, the system alerts you and provides a summary report about the threat. If there is no sign of malicious activity, the system assumes everything is working okay and doesn't generate a report. Every inference and decision is completed on the device, so all sensitive data remains safe at its source. So that users' data is kept secure and the system keeps improving, the system adopts a FL strategy. Local model weights (but not any data) are filed to the central server after every training cycle by the client nodes. All the updates are passed to the server using Federated Averaging (FedAvg) to generate a new global model. The updated model is shared with all of the clients afterwards. Clients use information from all over the world while still holding on to their particular data patterns, successfully joining local knowledge with system-wide data.

IoMT environments must meet strict demands when it comes to data privacy, speed of response and resources which makes this architecture an excellent choice. It enables updates to models on individual devices without changing the unified detection approach used across the system. It also allows for easy addition of new IoMT devices, since they can be used for federated training without affecting existing processes or gathering all data from one place.

3.3 Use Case Diagram

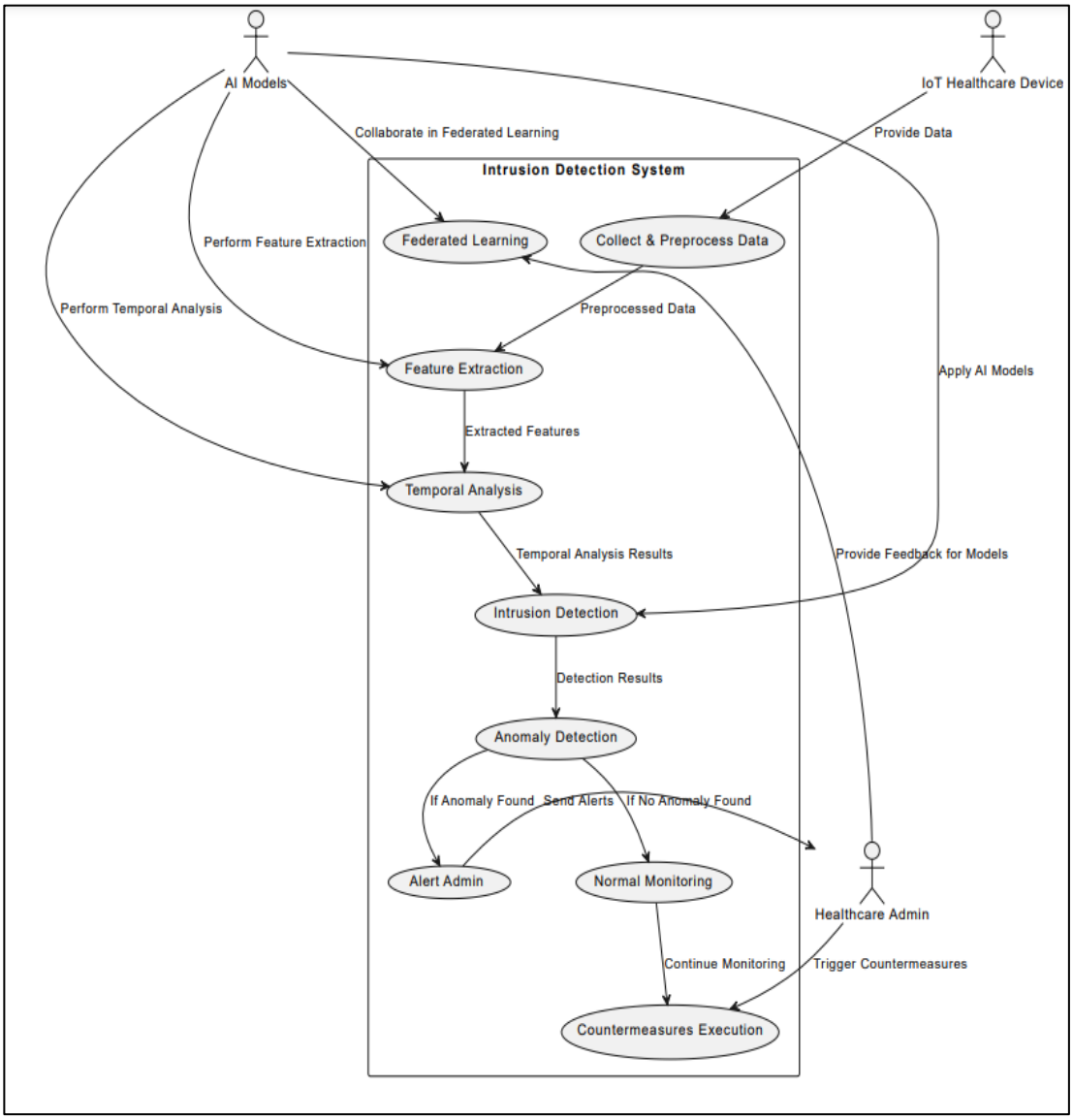

Figure 3.2 Use Case Diagram

Figure 3.2 shows the Use Case Diagram that explains how three main actors AI Models, IoT Healthcare Devices and the Healthcare Admin communicate with the main parts of an IDS in the IoMT. Part of this process is to collect data, apply FL techniques, find anything suspicious or malicious, create alarms and then enact methods to restore security and integrity to IoT systems in healthcare. The IDS is improved by the AI Models taking part in FL through joint work. By doing this, they can better detect threats without sharing personal data; they only communicate updates to the model developed from their own dataset. Furthermore, these models carry out feature extraction and time-based analysis on the data, allowing them to find typical features of unusual or suspicious events. Healthcare Devices are the most important suppliers

of information in the system. Devices inside the incubator always keep an eye on a patient and the incubator environment and the IDS gets real-time information about both. Devices can use embedded AI models to detect anomalies at the edges, so they do not need to rely on central processing which lowers both latency and dependence on the main system. If anomalies or intrusions are found, it is up to the Healthcare Admin to make important decisions. Whenever the system sends notifications, it falls to the admin to investigate and take suitable action. Because of this system, important threats are handled quickly and the chances of false positives are kept low. IoT Healthcare Devices act as both data providers and front-line

Use cases in the system are formally described as follows:

1. IoT healthcare devices supply data which is then processed so that it is ready for analysis by AI methods.
2. FL: Trained AI models are improved without putting personal data at risk, as training happens outside one central area.
3. Once the data is pre-processed, we look for and extract features that can contribute to high-quality detection of intrusions or anomalies.
4. The extracted data is studied by time intervals to spot any unusual changes that may suggest unusual activity.
5. Temporal analysis results let you discover any signs of intrusion or cyber security threats in the system.
6. The system discovers whether there are any unusual or irregular activities detected. Should an anomaly be detected, the admin will be alerted, otherwise the system works normally.
7. If the system spots unusual cases, it will email the healthcare admin to handle the alert.
8. If there are no problems found, the IDS continues normal detection without interruption.
9. If the admin determines it’s required, the security system executes previously set countermeasures to eliminate the detected risk.

These entities are related is central to the system’s way of functioning. Besides handling analysis such as picking out important features, AI Models also team up with FL to make the analyzers on the edge devices more powerful. They receive notifications, monitor the use of countermeasures and participate in the response. By

working in coordination from gathering information to addressing threats, an effective and reliable approach to detection is developed for the tight and sensitive world of IoMT.

3.4 Entity Relational Diagram(ERD)

Figure 3.3 below shows that the ERD clearly displays the interactions and dependencies among main entities in an intelligent IDS for the IoMT. Members of this system are the IoT_Device, Data_Collection, Feature_Extraction, Local_Model, Global_Model and Alert entities. The ERD model, in addition to handling local data and training, also combines features from FL. Such an arrangement supports decisions at both the edge and the server, increasing how efficient, adaptable and secure the system becomes.

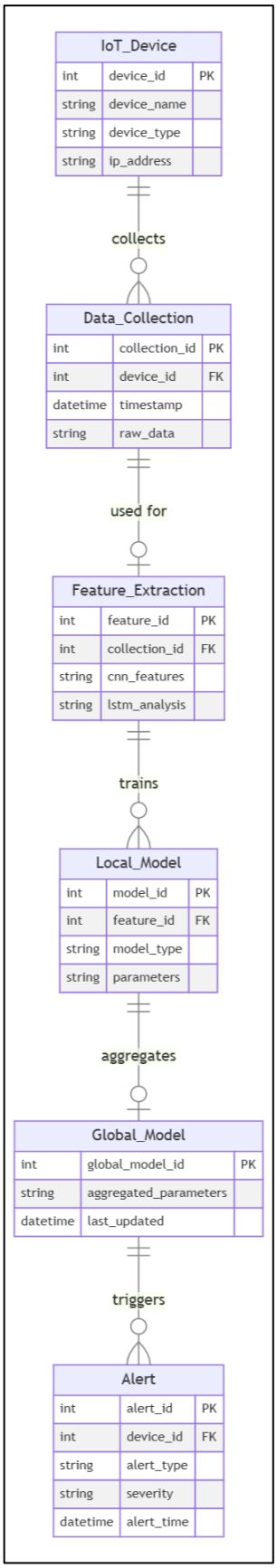

Figure 3.3 Entity Relationship Diagram

IoT_Device is used to represent all the hardware devices in use within healthcare. Such units can use sensors, actuators, devices worn by patients or embedded systems to measure patient vital signs or check environmental changes. Every record in the table is given a device_id which works as its main identifier. In addition, device_name, device_type and ip_address give important information about the device and how it is connected. Because of the device-centric nature of this entity, all additional monitoring activities can be properly traced back to the device they first happened on. Data_Collection processes and saves information from IoT devices at the moment it is gathered. Every record in this entity is label-led by a collection_id and also has the device_id of its connected IoT device linked by foreign key. Because of the linkage, each device can give numerous data points throughout its use. With timestamp, events are ordered and raw_data holds the raw data or logs from the sensors or system. All following analysis and model development start with the raw data provided. The Feature Extraction department carries out an analysis on the raw data that has been brought in. It is up to this entity to change unorganized data into a form that can be used by ML. All feature extraction records include a feature_id that is unique to them and a foreign key pointing to the matching collection_id. Through this phase, CNN is used to find cnn_features which are very good at understanding patterns in imaging or time-series data. At the same time, lstm_analysis is employed to spot temporal correlations and tendencies using Long Short-Term Memory networks. Because they highlight the main traits of the data, these outputs are perfect for use in predictive modelling.

The Local_Model uses the features created by the preceding ways to train ML models. Every local model gets a model_id and shares a feature_id foreign key with a specific set of features, meaning one model can only have one set of features. The model_type indicates the classification or regression character of the algorithm, while parameters collects the configuration or learnt parameters. Most of the time, these models run on a smartphone or at the gateway for context-appropriate, localized detection, while still keeping the data close and boosting speed. The following abstraction is the Global_Model which takes the outputs or parameters from various local models and merges them into a model that can work across several data sources. Recognized by the identifier global_model_id, this entity does not connect to feature sets directly, but rather accumulates aggregated parameters which are a summary of the lessons learned

by multiple local models. The last_updated attribute ensures that all models in the network are always current. This means that several local models join into one global iteration, so the IDS can access a wide range of knowledge without troubling data privacy.

Alert entities are just as important because they enforce the results found by the analytics system. Each alert is labeled with an alert_id and is tied to an IoT device by device_id which shows that one device can lead to several alerts over time. The attributes of alert_type, severity and alert_time include more information on each alert that is triggered. alert_type shows the kind of problem such as when a threshold is exceeded or an anomaly is found. The number in the severity field describes how urgent such as low priority, medium priority or high priority and alert_time lists the exact moment the alert was issued. By following a planned process, we ensure that threats can be both spotted and explained in context to make sure responses occur promptly. Through example, the ERD visualizes how relationships describe the types of dependencies and cardinalities found in the system. An IoT_Device has a one-to-many relationship with Data_Collection which means each device constantly adds to the database. Just like the Data_Cleaning to Data_Visualization link, this link is one-to-many since the same data record can use many feature extraction methods. The relationship focuses that every Feature_Extraction with Local_Model is different, as Feature_Extraction trains models linked only to specific sets of features. After all, multiple Local_Model instances can link to a single Global_Model, storing the FL structure. A final point is that an IoT_Device can cause more than one alert at any stage of the device's operation.

In general, the ERD clearly presents the main functions and connections in a strong IDS focused on IoMT. The process moves from data collection by devices, through developing features and training models, making sure insights are produced within each region and shared worldwide. As the end of the process, alerts turn the findings from analysis into reports that analysts can take action on. Modeling with an ERD helps build a scalable, respectful to privacy and practical framework for meeting the real-time challenges of IoMT deployments.

3.5 Data Flow Diagram

Level 01:

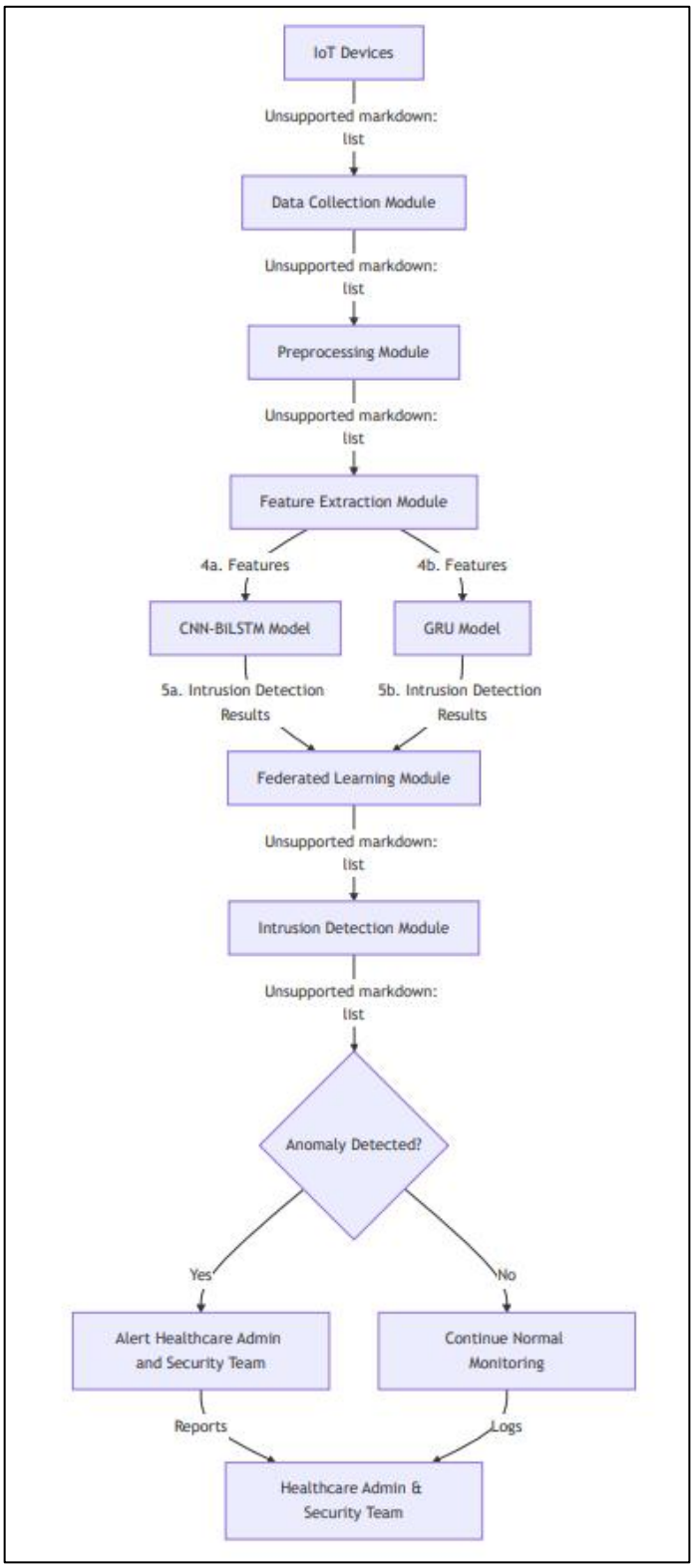

Figure 3.4 Data Flow Diagram Level 1

As depicted in Figure 3.4, the Level 1 Data Flow Diagram (DFD) offers a comprehensive architectural view of an intelligent IDS designed for the healthcare IoMT environment. This high-level model maps the end-to-end data journey starting from raw data ingestion through preprocessing, feature engineering, dual-path DL analysis, and federated model aggregation culminating in a final decision-making module that triggers real-time threat alerts. The architecture prioritizes real-time responsiveness, decentralized learning, and privacy preservation, all of which are essential in modern healthcare infrastructures dealing with sensitive and mission-critical data.

Data collection starts with the IoT Devices that supply data to the whole system. Among these devices are sensors for patient health, networked instruments in healthcare and actuators that constantly generate diverse forms of data. Physiology signals, environmental readings and digital logs make up this data and any of them can suggest that some operation or security issue might have occurred. With so many different and large elements in the system, it must receive inputs fast and from many sources, so it’s necessarily scalable and flexible enough for much distributed technology. At the beginning of the data pipeline, the Data Collection Module is the initial contact. It gathers the raw data from all connected devices which helps all devices work together without gaps. The purpose of attracting all data points is to keep the data complete in time and let you look at the network as a whole. If data is collected properly and logically, any later problems with the data can be avoided. Once obtained, data is sent to the Preprocessing Module which cleans it, makes it consistent and formats it for use with other parts of the process. One should conduct normalization, clean out any noise, address missing values and alter the data so it is readable by ML programs. When we preprocess data, we achieve better input and also boost the model’s performance and accuracy. Real-time healthcare systems require processing to be efficient and the integrity of the data shouldn’t be compromised.

After the data is made clean, the Feature Extraction Module uses it to detect useful trends and signals in the raw data. The purpose of this module is to organize the data with techniques that preserve the main qualities of the signal. At this stage, the architecture divides into two unique methods for handling features. In one area, they readjust the spatial characteristics of the input using CNNs and then work on

sequencing structures with the temporal BiLSTM. By mixing both approaches, this system is excellent at identifying long-term changes and repeating patterns. The second part concentrates on how the GRU Model uses GRUs for rapid and efficient handling of sequences. GRUs perform better than BiLSTM in time-series anomaly detection for less computing power, improving how quickly the system can respond. Intrusion Detection Models made up of CNN-BiLSTM and GRU run simultaneously on distinct feature sets. By having more than one, the system can deal with faults and recognizes them more clearly. Every model delivers a prediction or risk score about whether activity is malicious or not. This system is designed to work well when some of the models may miss a fraction of patterns, as it combines the effects of multiple models. FL Module is then given model outputs which act as the system's main intelligence processing unit. To protect privacy, data is not sent from the endpoints to a central point but aggregated from local model results at the module. The design meets HIPAA standards, so patient or device information is not shared from its primary machine. FL also increases the IDS's ability to respond to threats in new places by including knowledge from various environments, allowing the system to deal with new or changing threats in hospitals. All the outcomes are processed by the Intrusion Detection Module and analyzed one last time. This module uses data from all three parts to find out if an anomaly or intrusion has happened. It applies more logic methods, including thresholding and filters, to lower the chance of falsely classifying an attack and to treat credible issues as threats. During the Decision Point, all the details of the analysis are reviewed by the system. Any detected anomalies cause the system to send detailed alerts to the Healthcare Admin and Security Team instantly. Thanks to the information about when, where and how severe the event is, administrators can quickly decide on a course of action. When there is no sign of trouble, the system continues with normal monitoring, records what has happened and continues to watch in real-time peacefully. Thanks to this, the security system is reliable, continues its tasks and can respond to new risks quickly. The Healthcare Admin and Security Team make up the last part of this process. They are informed of system-related alerts and logs which they use for current response and also for improving the system over time. They help to check if the alerts are real, handle security threats and add protections to the system. Also, using records and logs in history helps during auditing, training systems again and adjusting detection rules, helping the system gradually become smarter.

Combining all the DFD elements continuously results in a system that manages user privacy while improving workflow and growing smoothly. Modular architecture allows the system to be used for decentralized education, fast reviews and aware choices, all with strict data safeguards. The use of a CNN-BiLSTM and GRU approach offers the system security in its detection thanks to having different models. Its FL ability provides both flexible scaling on extensive networks and reliable data privacy. With an alerting feature and understandable information, people in charge can deal with situations quickly and effectively. Therefore, this architecture can fit the demands of healthcare IoMT perfectly and respond adaptively to security issues.

Level 02:

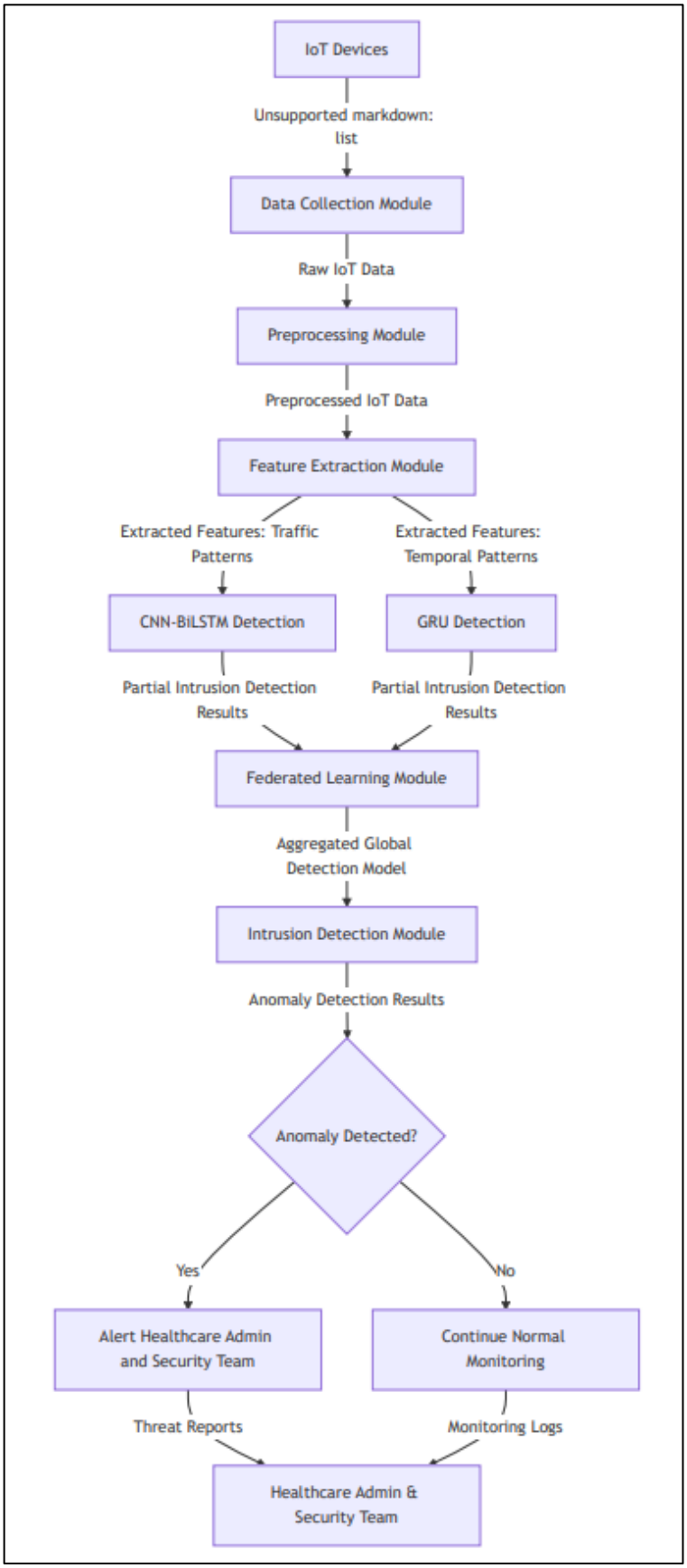

Figure 3.5 Data Flow Diagram Level 2

Figure 3.5 shows that the Level 2 DFD demonstrates how the internal processes and data in the proposed architecture interact with each other. The system is shown as divided into its main areas, revealing the passage of data from IoT devices to specialized processing, through CNN-BiLSTM and GRU DL networks and then fused by FL. Because of its strict layering, the DFD proves the system retains scalability, prompt responses and offers secure AI for health IoT settings. All data movement begins when IoT devices send information to the edge. Smart products used in hospitals, for example wearable monitors, infusion pumps and diagnostic systems, continuously create traffic in the network, outputs from their sensors and logs from the systems. They are the tools needed to discover possible security breaches. It is the Data Collection Module's job to gather raw data from all working IoT devices. All data is gathered in one place, making sure it is secure and remains the same at all times. As the entry point of the detection pipeline, it helps prevent missed vital traffic and maintains a low delay when streaming. After that, the Preprocessing Module processes the original data to be used in ML processing. First, call information must go through noise filtering, getting normalized and be turned into structured files that machines can process easily. With the help of this module, any useless or repeated data is ignored and issues in measurement are corrected, so what is forwarded to the next stage is clear and easy to analyze.

Following preprocessing, the data goes into the Feature Extraction Module which organizes the data into two large classes based on the kind of behaviour shown. The design, route and amount of network traffic—including packet sizes, the set of protocols and flow conditions. We feed these into the CNN-BiLSTM model that is excellent at spotting changes in both spatial and time-based network behaviour. These patterns involve the ways network activities and devices change over time, for example by measuring how often things connect and when breaks between events occur. The GRU model, designed to learn the time order in sequences, gets these instances. Both Intrusion Detection Models work independently to analyze their dedicated forms of input. CNN-BiLSTM Detection employs CNNs together with BiLSTM units to focus on both images and the order of events in a video. It is best used to spot any inconsistencies in the way data packets are received or sent. Also, the GRU Detection model relies on GRUs to detect delayed or changing activities in the data that indicate a possible stealthy attack. Depending on the part of the data it deals with, each model

The system produces inference by indicating where it thinks an object is. After preparing these parts, the FL Module ensures both private and accurate results. Data from the CNN-BiLSTM and GRU models is used, but the original information is not transmitted to clients. The system combines data from several datasets, relying on both FedAvg and adaptive model fusion, but always takes privacy concerns into account for the healthcare facilities. The output from disparate sources is then handed over to the Intrusion Detection Module which makes all the decisions within the system. The tool makes use of the learnings from the federated model to identify if data reflects regular use or indicates a possible attack. Because the key steps in this inference make use of both models, the output is not affected by the problems of just one model. The next critical step is an Anomaly Detected? check. At this point, the system checks if the combined model reports a security threat. If this answer is Yes, the system will send out alarms to the Healthcare Admin and Security Team. The system also produces an elaborate threat summary that covers the kind of possible threat which devices or endpoints are involved and the model's certainty in its outcome. As a result, incidents are managed and controlled at a quick rate. If No is chosen, the system returns to its usual passive mode for checks. But, in order to maintain a consistent record and help with future tasks, processed data is kept in log files. By reviewing these logs, threats that build over time can be detected and the system's effectiveness can keep improving.

The Healthcare Admin and Security Team act as the final human point of control in the system. They check alerts, confirm threat intelligence and initiate actions to address threats or stop them. They use system logs to confirm that all elements are performing properly and to guarantee both network security and uninterrupted operations. Overall, Level 2 of Figure 3.5 shows the paths of data and information within a federated dual-model IDS. The approach displays clearly how it deals with various kinds of input, chooses different feature set types, carries out analysis in parallel and still combines findings using secure aggregation for accurate and quick anomaly decisions. As a result, this design delivers excellent detection, can support many users and is in line with key regulations that secure IoT-based healthcare infrastructures.

3.6 Flowchart

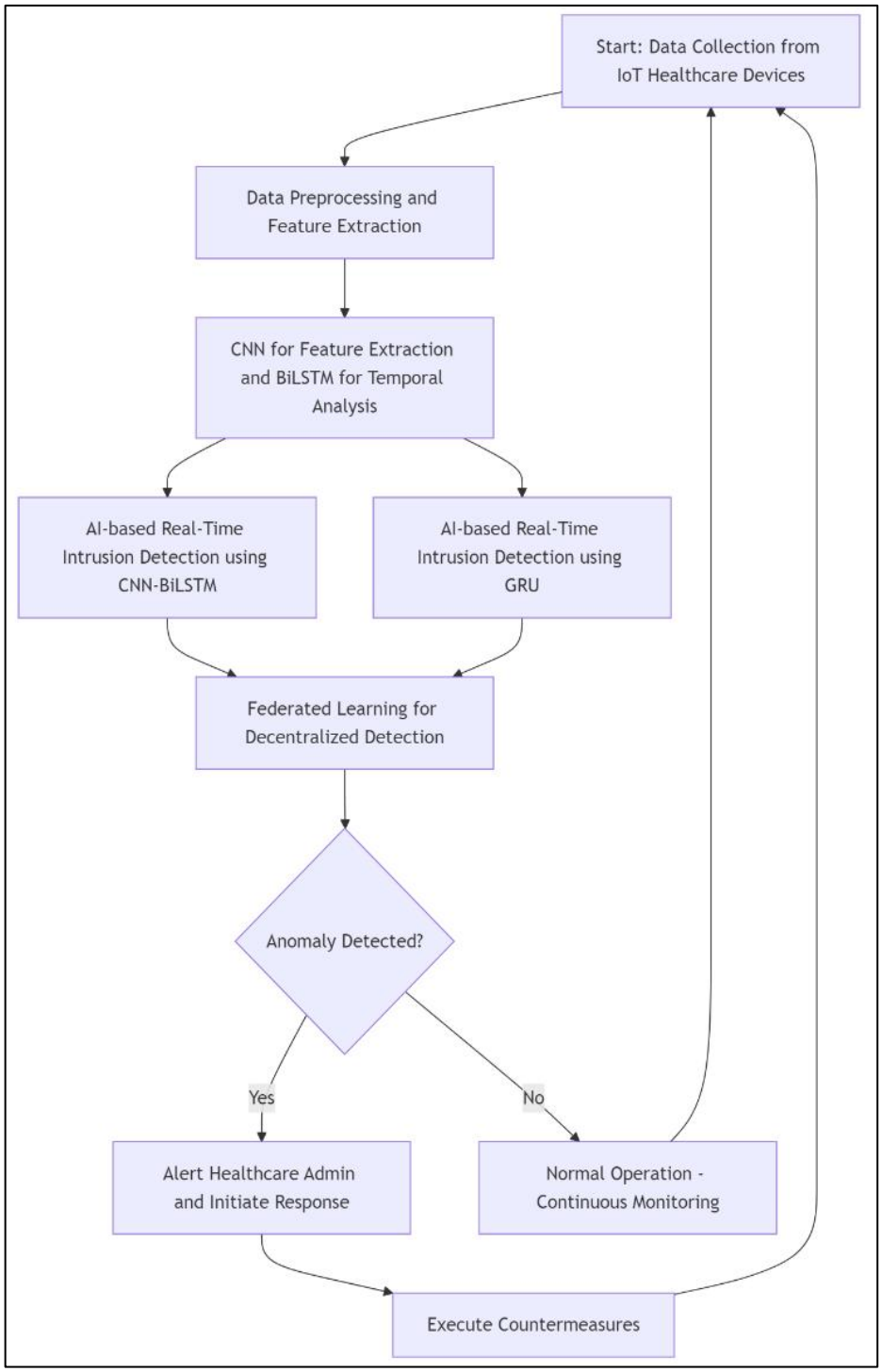

Figure 3.6 Flowchart

The IDS pipeline for healthcare IoMT ecosystems is clearly illustrated in Figure 3.6. It involves all stages of threat cybersecurity, from data gathering to advanced analysis, FL involvement, quick decision making and enforcing protection methods. Besides showing each step, the flowchart points out that the system must be continously

watched, adapt to new risks and keep data safe to suit any emergency in sensitive medical settings. From the Start point, we collect information from IoT Healthcare Devices to bring it into the system unprocessed. Such technology covers wearable devices for fitness, internal sensors, various testing machines and networked energy. The accuracy of all following analysis depends on the integrity of this real-time data. Because the data is so diverse and comes in large amounts, the system must be able to support many devices in different healthcare locations. Then, the next step is to feed the data into Data Preprocessing and Feature Extraction. The data must now be converted, formatted and cleaned before you can analyze it. Processes like noise reduction, missing value imputation and formatting help, while looking for useful patterns and significant features is the job of feature extraction. This helps turn information we've collected into a form that can be understood by the training models in the next phases.

At this point, the prepared and enhanced data goes through the Dual Path AI Analysis which begins with CNN to spot features first and BiLSTM to examine the data over time. The CNNs identify regions, patterns in their arrangement and noticeable qualities in the signals that could point to a problem. Afterwards, BiLSTM networks examine how information in the sequence relates to other pieces of information, both from one direction to another and from the last to the start. Such a process is very successful at spotting stealthy or evolving dangers that become noticeable only with time. At the same time, the system initiates another path with AI-based detection that uses GRU to examine time-based patterns. GRUs give a simpler and faster option for processing information than BiLSTM, keeping their predictive strength unchanged. Running both CNN-BiLSTM and GRU models as separate branches completes the system, improves detection accuracy and enables use under different resources. Choosing this architecture improves how quickly the system responds and adapts, as well as its utility when there is limited bandwidth or power in the computing environment. In the FL for Decentralized Detection module, outputs from both CNN-BiLSTM and GRU branches are brought together. Data locality and privacy are confirmed here because learned parameters or FL-based results are combined during the process. Raw data is kept decentralized, as each device updates its own model and communicate the updates to the system. Not only does this approach meet HIPAA and similar standards, but it also supports collaborative use on different devices which allows the system to learn from

many sources of data while still safeguarding patient privacy. It also increases the ability to use the model in a variety of healthcare infrastructures.

The results from all the gathered models inform the decision of whether an anomaly was found. Here, the system determines if there is anything suspicious happening. Should an anomaly appear, the step proceeds to Alert Healthcare Admin and Initiate Response, in which complete alerts containing metadata like timestamps, severity scores, information on affected devices and the system's confidence level are sent to healthcare admins and security staff. With timely knowledge, the team can quickly and properly address issues of the various threats. If everything appears as expected, the system transits back to Normal Operation – Continuous Monitoring mode and goes on observing IoT devices uninterruptedly. Once the choice is made, the algorithm goes back to the initial step, keeping real-time tracking and reactions all the time. Because of this, the system keeps operating and collecting more data even if the previous threats are gone so its detection approaches can be improved. After raising an alert, Cyber Attack Defense activates set defense measures to stop the threat. As a result, the device might be put into quarantine, suspicious IPs should be blocked, configurations reset or manual methods used. Thanks to its automated feature, the system responds more quickly and reliably when a quick response is necessary for patients. Afterward, a repeated process starts, where data is collected, assessed, threats are found, action is taken and lessons are learned. As a result of this feedback loop, the system grows and changes according to changes in its environment and uses past incidents to improve its inner operations. Combining CNN-BiLSTM and GRU with FL enables the system to address the needs of healthcare IoT systems securely.

In short, it outlines a security pipeline that takes data from the edges, responds quickly when needed, uses continuing monitoring and updates automatically. Thanks to using real-time analytics, varied ML and decentralized information, it is very good at guarding modern healthcare systems against cyber risks.

CHAPTER 4

DATA AND EXPERIMENTS and IMPLEMENTATION

4.1 Libraries Used in the Project

Here, we look closely at the main libraries and frameworks we worked with in building our IDS for use in IoT-enabled healthcare systems. Libraries such as these have an important function in data handling, preprocessing, graphs, ML and FL. With these top-quality technologies, we provide efficient processing, training and deployment of data, strengthening and improving our IDS.

4.2 Pandas

When handling, transforming and preparing the data, all the work was done with Pandas. IEEE Dataport provided the datasets needed and Pandas was responsible for importing and sorting thousands of data from the simulations, distinguishing between attack records and device behaviour notes. Pandas support for DataFrames made it possible to structure the data, so it was simple to view, pick and edit individual records. It facilitated key preprocessing steps such as removing missing or null values, renaming columns for clarity, and encoding categorical features into numeric formats suitable for DL models. Complex filtering conditions were also applied using Pandas to extract specific subsets of the dataset for example, isolating attack traffic from normal device logs to balance class distribution prior to FL.

During feature engineering, Pandas was used to compute aggregated metrics (like averages, maximums, and trends) over temporal data streams generated by IoMT devices. These features were critical for detecting suspicious activity patterns and were passed to the CNN-BiLSTM and GRU models for training. Additionally, Pandas supported merging and joining multiple dataset files (e.g., patient monitoring + network traffic) into a unified structure for comprehensive model input.

Since pipelines rely on NumPy and Matplotlib, I was able to immediately turn DataFrames into arrays for training and visualize them for observing the data set. The main reason Pandas was essential here was that it helped speed up preprocessing and

formed the backbone for all data operations before our models were built and used in federated aggregation.

```
import pandas as pd

# Load individual datasets
attack_df = pd.read_csv("Attack.csv")
env_monitor_df = pd.read_csv("environmentMonitoring.csv")
patient_monitor_df = pd.read_csv("patientMonitoring.csv")

# Merge into one unified dataset
df = pd.concat([attack_df, env_monitor_df,
patient_monitor_df], ignore_index=True)

# View dataset structure
print(df.info())
```

Figure 4.1 Pandas

4.3 NumPy

NumPy helped us a lot with numerical work and processing arrays that had many elements. It mattered a lot to guarantee every needed entry was clean and met the requirements of both our CNN-BiLSTM and GRU models.

With Pandas, we fixed the raw datasets so NumPy could optimize them into faster and efficient ndarray formats. This mattered a lot, because our neural networks needed their input data sets to form specific shapes, especially with BiLSTM and similar models.

Each dataset was expected by GRU to arranged in three dimensions, samples, time steps and features. NumPy enabled us to reshape and align the data accurately with these dimensional expectations, and it facilitated the transformation of spatial patterns in network traffic logs into fixed-size 2D matrices, making them compatible with the convolutional layers of our CNN model.

Vectorized operations in NumPy were extensively used for preprocessing tasks, including normalization and one-hot encoding, allowing us to scale feature vectors uniformly across all federated clients. These functions not only accelerated computation but also ensured consistency in model input across distributed nodes.

NumPy's significance increased further during the implementation of FL. It was used to extract model weights from local TensorFlow models and convert them into NumPy arrays for aggregation. As a result, we were able to conduct custom FedAvg by computing the mean of matching parameter values from various clients each round. We kept copies of our state and recent training progress in NumPy arrays, enabling easy restart of training after each communication round. What's more, using NumPy allowed for real-time statistical analysis of the data and the computing of standard deviations, means and correlation matrices. The results of our analysis helped choose the best features since we removed those that were not important for model performance. At the evaluation stage, the predicted tensors were turned into NumPy arrays to calculate the important scores used: confusion matrices, precision, recall and F1 scores. We utilized these NumPy arrays to create training graphs, accuracy curves and heatmaps by providing them to Matplotlib and Seaborn. In brief, doing the project heavily relied on NumPy to ensure smooth transformations between data standpoints, complicated tensor operations and distributed learning. Bringing it into every step of the project pipeline meant that both local and federated training were This system is able to function quickly and adapt to any size of data for promptly detecting intrusion in healthcare IoT devices.

```
import numpy as np

# Convert DataFrame to NumPy array for deep learning input
X_array = df[numeric_features].values
y_array = np.array(df['label'])

# Reshape input for CNN/GRU model
X_scaled = X_array.reshape(X_array.shape[0],
X_array.shape[1], 1)
```

Figure 4.2 Numpy

4.4 Matplotlib & Seaborn

It was Matplotlib and Seaborn that helped make both the input data and the results of the DL systems in IoMT environments clear. Such libraries helped the team fully analyze the data, discover concealed patterns, check the class distribution in datasets and evaluate model activity both during FL and after completion.

When it came to creating graphs, I turned to Matplotlib which allowed me to tweak each graph's appearance precisely. With Seaborn based on Matplotlib, it was possible to design more effective statistical plots that look tidy. All of these libraries offered what I needed to turn the data into helpful and clear visualizations.

As an initial step in analyzing data, the extracted features were used with Matplotlib and Seaborn to generate histograms, box plots and density plots. Using visualization, issues with skewness in features, outliers and missing values were identified. We used the findings from these plots to guide our steps for clearing outliers, scaling features and making class labels even.

These libraries also played a key role in examining class distribution, particularly when working with datasets where there was a big difference in number of labels a frequent problem in intrusion detection. Viewing the attack and normal data using Seaborn's graphs and charts made it easier to control the use of Synthetic Minority Over-sampling Technique (SMOTE), so it was only added where needed. In federated models, the libraries offered a way to see how each client's data was represented among the different classes.

For the training process, Matplotlib charts were produced for learning curves of accuracy and loss for both the CNN-BiLSTM and GRU models. Thanks to these plots, they could watch out for problems with underfitting or overfitting and change the team's hyperparameters like learning rate, batch size and epochs as needed. Seeing the training process live was especially helpful in FL since the various client models trained at once and reported their metric results separately. During evaluation, heatmaps were developed with Seaborn to illustrate which features were related to each other. With these heatmaps, we could judge how different the features were and avoided building a model that relied too much on similar data. Confusion matrices were also displayed using Seaborn to make it simpler to see how many errors each class led to and to measure recall and precision for the model on each class. Using Matplotlib, trend plots were generated to show how certain network features behaved over a period, specifically when attacks took place. Plots from these tests showed that the system could alert unusual events through sequential patterns, as needed by BiLSTM and GRU models. Federated training metrics such as the overall accuracy improvement of the global model, performance of models at individual clients and the

trend of convergence, could all be clearly seen in the libraries. All in all, it was Matplotlib and Seaborn that converted our raw data into useful visualizations. Their ability to present complex model behaviours, training dynamics, and data distributions in an intuitive and publication-ready format made them essential throughout the lifecycle of this project. From early-stage data validation to final performance reporting, these libraries supported informed decision-making, helped communicate findings, and added clarity to the analytical outcomes of the federated AI-based IDS.

```
import matplotlib.pyplot as plt
import seaborn as sns

# Correlation heatmap
plt.figure(figsize=(10, 8))
sns.heatmap(df[numeric_features].corr(), annot=True,
cmap="coolwarm")
plt.title("Feature Correlations")
plt.show()

# Class distribution
sns.countplot(x='label', data=df)
plt.title("Class Distribution")
plt.show()
```

Figure 4.3 Matplotlib & Seaborn

4.5 Scikit-Learn

In this project, Scikit-learn was used as a core ML toolkit for tasks related to data preprocessing, model evaluation, feature selection, and baseline comparisons. While the primary DL models CNN-BiLSTM and GRU were implemented using TensorFlow and TensorFlow Federated(TFF), Scikit-learn complemented this by providing essential tools for building a complete, reproducible ML pipeline, especially during the early stages of model prototyping and comparative benchmarking.

One of the first applications of Scikit-learn in the project was during label encoding and feature scaling. The datasets sourced from IEEE Dataport included categorical and numerical attributes, which were normalized using StandardScaler and MinMaxScaler from Scikit-learn. These transformations helped bring all input features to a similar

scale, ensuring that gradient-based models like GRU and BiLSTM converged more efficiently during training.

Using Skikit-learn made it possible to reduce data dimensions naturally. The most important features for detecting intrusions were found using SelectKBest and correlation-based filtering which prevented multicollinearity and redundancy. Cutting down on the input features did not lower detection accuracy, so the models became well-matched to the processor capabilities of IoMT devices. Scikit-learn was also used to create simple models that allowed us to assess how well the DL methods are working. Additionally, three traditional classifiers—SVMs, RF and K-Nearest Neighbors—were tried out in Scikit-learn to compare the results for accuracy, precision, recall and F1-score. Using these models made clear that BiLSTM and GRU should be used since they can handle temporal patterns that earlier models could not. Model evaluation tools from Scikit-learn were widely used for getting and explaining the classification metrics. All three of these visualizations were produced by the built-in features of the library. Checking false positives, true negatives and class-wise performance showed important results and made sure the system remained reliable in federated settings. Moreover, the validity of the models was evaluated using Scikit-learn and k-fold techniques in the cross-validation process. Using these data enabled me to measure the overall usefulness of models trained either centrally or through federated methods. Using the same way to divide the data allowed us to compare results fairly and rely on model predictions. As well as classification tasks, Scikit-learn allowed experimentation with data balancing. While SMOTE is usually the main way to use Synthetic Minority Oversampling Technique, we used Scikit-learn's resampling features to check how traditional ML classifiers were affected by uneven data. Looking at this made me realize that non-DL models find it hard to address imbalanced classes without using advanced techniques.

All in all, Scikit-learn helped connect the old ML way of doing things with the new DL process. This led to stronger preliminary data processing, detailed model testing and a strong basis for basic model benchmarking. We depended on these abilities to analyze the importance and success of using different architectures for privacy-preserving FL-based intrusion detection in healthcare networks.

```
from sklearn.preprocessing import LabelEncoder,
StandardScaler
from sklearn.model_selection import train_test_split

# Encode labels
label_encoder = LabelEncoder()
y_encoded = label_encoder.fit_transform(df['label'])

# Scale features
scaler = StandardScaler()
X_scaled = scaler.fit_transform(df[numeric_features])

# Split into training and testing
X_train, X_test, y_train, y_test = train_test_split(
    X_scaled, y_encoded, test_size=0.2, random_state=42
)
```

Figure 4.4 Scikit-Learn

4.6 TensorFlow

We used TensorFlow as our main DL framework to make, teach and assess the AI models created for our IDS in healthcare settings using IoT technology. Since network data and health monitoring logs are not simple and go in sequence, TensorFlow's modular and efficient design allowed for the development of the system's important threat detection models.

By building the model one layer at a time with TensorFlow, I was able to exactly manage how every part of the model operated. Sequential and Functional APIs in TensorFlow were used to create the CNN-BiLSTM model. First, Conv1D layers were used to find spatial patterns in the input and Bi-directional LSTM layers were then applied to detect patterns in both directions of the sequence. The aim of this type of architecture is to discover patterns in Internet of Medical Things traffic that gradually change and might reveal unusual or harmful events.

At the same time, a model using the GRU, called BiLSTM, was built in TensorFlow. GRU layers were used in this architecture over LSTM to make the model train much faster and take up less memory, a key requirement for IoMT devices. The Adam

optimizer and the BinaryCrossentropy loss function were used to develop both models which were trained on data focused on telling apart normal and unhealthy traffic.

In order to create these models, we sent time-series data after processing into TensorFlow's fit() function and we captured the training history for further review. We used TensorFlow's GPU acceleration which helped us train the model much faster on datasets from real cases of cyberattacks. Convergence stability was obtained by using batch processing, checkpoint saving and the early stopping callback mechanism. TensorFlow also supported real-time evaluation and prediction, where the trained models were tested against unseen data to assess generalization. The evaluate() and predict() methods were used to output probabilities and class predictions, which were then passed to Scikit-learn for metric calculation and Matplotlib for visualization. This integration created a seamless pipeline for iterative experimentation and performance analysis.

Moreover, TensorFlow was critical in enabling the FL setup used in this project. Although TFF handled the distribution logic, the core model definitions, optimizers, and training functions were all first built using TensorFlow. The models trained on each client in the simulated federated environment were cloned from TensorFlow base models, ensuring consistency in structure and performance.

Additionally, TensorFlow's model export features (model.save() and tf.keras.models.load_model()) were used to serialize trained models for reuse in federated rounds or for deployment. This allowed persistent storage and reproducibility across multiple training sessions and system simulations.

In conclusion, TensorFlow acted as the central engine for DL in this project, supporting both the complexity of CNN-BiLSTM and the efficiency of GRU. Its flexibility, performance, and compatibility with federated extensions made it the ideal framework for building scalable, real-time, and privacy-aware IDS in the sensitive context of healthcare IoT infrastructures.

```
import tensorflow as tf

# CNN-BiLSTM architecture
model = tf.keras.Sequential([
    tf.keras.layers.Conv1D(64, 3, activation='relu',
input_shape=(X_train.shape[1], 1)),
    tf.keras.layers.MaxPooling1D(2),
    tf.keras.layers.Bidirectional(tf.keras.layers.LSTM(64,
return_sequences=True)),
    tf.keras.layers.Bidirectional(tf.keras.layers.LSTM(64))
,
    tf.keras.layers.Dense(32, activation='relu'),
    tf.keras.layers.Dense(1, activation='sigmoid')
])
```

Figure 4.5 TensorFlow

4.7 TensorFlow Federated (TFF)

TFF was a critical component in this project, enabling the implementation of privacy-preserving, decentralized model training across simulated IoMT clients. Given the sensitivity of medical data and the constraints of deploying advanced AI models directly on low-power healthcare IoT devices, TFF allowed the team to create a FL framework where intrusion detection models could be collaboratively trained without sharing raw patient data.

The FL pipeline was architected using TFF's tff.learning APIs, which allowed the definition of a FedAvg process based on client-side model training and server-side aggregation. In this setup, each client node represented a virtual IoMT device that maintained its own local dataset and independently trained a copy of the global CNN-BiLSTM or GRU model. Only the updated model weights (not the data) were sent back to the central server, where they were aggregated using the FedAvg algorithm.

TFF enabled simulation of real-world constraints by allowing independent model updates on multiple clients with heterogeneous data distributions. This was especially useful for evaluating the robustness of the IDS in scenarios where some IoMT devices had more attack data while others primarily observed normal traffic. By simulating this imbalance, TFF helped the team assess how well the global model generalized to unseen data across varying client profiles.

In terms of workflow, TFF was used to:

- Initialize and distribute TensorFlow-based models (CNN-BiLSTM or GRU) to each client.
- Define the model_fn and client_data pipelines to handle localized training logic and data preprocessing for each node.
- Execute multiple rounds of federated training where each client performed a fixed number of local epochs before returning model updates.
- Aggregate client updates into a global model using the tff.learning.build_federated_averaging_process() function.
- Monitor the performance of the global model after each round using centralized validation datasets.

The team used TFF's simulation runtime to conduct experiments on synthetic and real-world intrusion datasets divided across clients. This made it possible to measure per-client training accuracy, track convergence behaviour of the federated model, and compare training efficiency against a centralized approach. Each round of training was analyzed for accuracy improvement, loss reduction, and anomaly detection capability, helping refine both the model architecture and the number of participating clients.

Furthermore, TFF supported fine-tuning and personalization by allowing clients to optionally retain a local model copy adapted to their specific data profile. This flexibility enabled enhanced detection of localized attack patterns while still contributing to the robustness of the federated global model.

Overall, TFF was not only essential for preserving privacy and decentralizing model updates, but also served as the experimental foundation for validating the feasibility of deploying AI-driven IDS systems in distributed, real-time IoMT networks. Because it works smoothly with TensorFlow and allows fast communication in collective learning, we chose it as the base for our FL framework.

```
import tensorflow_federated as tff

# Wrap Keras model in TFF format
def model_fn():
    keras_model = create_cnn_bilstm_model()
    return tff.learning.models.from_keras_model(
        keras_model=keras_model,
        input_spec=input_spec,
        loss=tf.keras.losses.BinaryCrossentropy(),
        metrics=[tf.keras.metrics.BinaryAccuracy()]
    )
```

Figure 4.6 TensorFlow Federated

Secure Aggregation: Encrypts all the updates so they cannot be taken during the transmission. Protecting the data's privacy and security calls for secure aggregation.

4.8 gdown

For this project, gdown was used to make it easier to automatically download large files from Google Drive directly into the run environment which helped simplify access to data and models on several experiment platforms. Because our IDS was set up with cloud technologies, we found ourselves consistently having to get data and checkpoints from shared remote storage. gdown helped automate this so all collaborators and simulation nodes could access the data efficiently and reliably.
Since CIC-IDS2018 and its modified versions were too large for GitHub or Colab, they were used in this project with their stored labels. The datasets were put on Google Drive and made available to the project via gdown-friendly URLs. Unique file IDs were used with the gdown command during environment setup so that no manual downloading or the use of third-party APIs was needed. Short setup was made possible, reproducibility improved and it became easy to team up tasks across machines and users.

Not only did gdown give us access to the datasets, it allowed us to download serialized DL models from cloud storage, including particular CNN-BiLSTM and GRU models. Thanks to this, models from earlier work could be set aside and reused for different evaluation purposes. Automating the download process prevented the need to "retrain

from scratch," so evaluation data remained consistent even when working under time or hardware restraints.

In the FL setup, gdown allowed simulated clients to download the initial model or dataset themselves, without needing to go out on the internet or submit files inside the notebook. Thanks to this ability, we could repetitively use federated rounds and have all clients start training from the same point. Version control was made simpler, as each snapshot from training was kept with a specific label and could be downloaded when required for testing. Moreover, the use of gdown supported the ability to reproduce the entire framework of the IDS. To ensure every part of the system used the same assets, the research team added it to both cells and headers so every replica on a local computer, virtual environment or Colab notebook could pull the needed assets.

This reduced human error, enabled checkpointed progress, and strengthened the portability of the FL setup.

In conclusion, while gdown did not play a computational role in model inference or training, it provided critical logistical support. It served as a lightweight, reliable tool for managing external dependencies, enabling rapid setup, scalable distribution, and consistent reuse of data and model components. This made it an essential element in building a robust, collaborative, and modular AI-driven IDS for IoT-enabled healthcare networks.

```
import gdown

file_ids = {
    "Attack.csv": "1Va-KZmhopT2VtSAdPD1V_2XIqhuybdGV"
}

for file_name, file_id in file_ids.items():
    gdown.download(f"https://drive.google.com/uc?id={file_i
d}", file_name, quiet=False)
```

Figure 4.7 gdown

4.9 Models Used in this Project

Intrusion detection in IoT-enabled healthcare networks requires models capable of capturing both spatial and temporal patterns in network traffic. Such patterns allow us to identify cyber threats faster. We apply CNN-BiLSTM and GRU models to perform

intrusion detection in real time. With these models, it is easy to study network traffic, recognize different partnerships and boost the accuracy of finding attacks. Model training makes use of Binary Cross-Entropy(BCE) Loss to check prediction accuracy when dealing with binary classification. It is defined as:

$$BCE = -\frac{1}{N}\sum_{i=1}^{N}[y_i \log(p_i) + (1 - y_i)\log(1 - p_i)]$$

Equation 4.1 Binary Cross-Entropy Loss Function

4.10 Convolutional Neural Network – BiLSTM

This model is a combination of CNNs and BiLSTM networks. Because this approach handles both the location and sequence of traffic, it is able to spot cyber risks in IoT networks.

Why CNN?

- CNNs are good at noticing spatial connections between data which allows them to find valuable features hidden in raw network packets. Noticing these behaviour changes is important when detecting cyber attacks in networks.
- Unlike before when people had to design features, CNNs now automatically detect important patterns from the data themselves. As a result, less domain knowledge is needed and the model can respond to new types of attacks.
- CNNs help by identifying important features and removing unnecessary information found in images. This makes it possible for the IDS to detect threat events more accurately.

Why BiLSTM?

- This model investigates the data in both forward and reverse directions which enriches data processing in sequence. This process works especially well for finding attacks that need several different actions.
- Long-term Dependency Learning: The algorithm finds the important relationships in network traffic which helps spot more advanced threats. Many

attacks develop step by step and BiLSTMs excel at capturing those sequential features.

- It can detect small differences in packet flow timing to determine possible attempts by intruders. It is needed to alert you to attacks you might not recognize right away.

Model Components:

1. **Conv1D Layer:** Inside a Conv1D Layer, the network learns spatial patterns in the network traffic data, including information about packets. It is this layer's duty to locate patterns in the network data. It can be described mathematically how the convolutional operation works in a Conv1D layer:

$$y_i = \sum_{j=0}^{k-1} x_{i+j} . w_j + b$$

Equation 4.2 Convolution Operation in CNN

Where:
x is the input signal
w is the kernel (filter)
b is the bias
y_i is the output at position i

2. MaxPooling Layer: Makes the data less complex, but maintains important information so that the model doesn't overfit. Because of this layer, the model requires fewer computations.

3. BiLSTM Layer: BiLSTM is used in networks to find significant connections among events in data, pointing to attacks. They are designed to store information about the changes over time in network traffic.

4. Dense Layer: After extraction, features are passed through fully connected layers that make the final safety call by labelling the tokens normal or malicious. The objective of these layers is to decide on the final classification.

Output Layer: The output layer relies on a sigmoid activation to give either a normal or an attack score to the input traffic. It is the output that the model gives as its result.

4.11 Gated Recurrent Unit (GRU)

The GRU was designed as a straightforward version of RNNs to deal with data that comes in a sequence with reduced overhead. GRU is a popular intrusion detection model in IoT healthcare because it uses a light architecture and can identify changes across time which is needed in devices with limited capability and fast response times. Just like LSTM models, GRUs can accomplish similar tasks, but they use fewer parameters to do so. There are a number of explanations why GRUs excel at detecting intrusions in IoT-related healthcare settings. Models deployed on edge devices must be efficient with memory, because these systems usually do not have access to advanced computing tools. Thanks to their design, GRUs use fewer gates than LSTMs, so they require less memory and train and run faster. For this reason, IoT devices are perfect for using mobile security solutions straight on the devices, so hospitals don't need to rely entirely on cloud processing. Because GRUs work effectively with time-series data, they are appropriate for analyzing how network traffic unfolds over time. The framework helps the model remember past info which supports the model in detecting anomalies that develop slowly in time or based on specific time frames. To be effective at intrusion detection, it is crucial to be time-aware, as this way, obvious activity is identified in real-time, not only after it has occurred.

Besides their ability to store and process information one unit at a time, GRUs also run faster than LSTMs. The simplified design where all three gates combine into one update gate means GRUs complete processing much faster, both when being trained and when running inferences. A speed advantage helps in fighting real-time IDS since early detection of threats can effectively protect against problems such as unauthorized access, stealing data and denial of service. GRUs also stand out because they are durable against changes in the data. Network traffic in IoT healthcare is often quite variable because of the broad mix of devices, users and their habits. Because of their generalization, GRUs can detect threats that aren't yet known to their developers, including zero-day attacks. With a clear understanding of normal network changes over time, GRUs can successfully point out new intrusion attempts.

Usually such a model includes these key components:

1. GRU Layers: These layers are the main part of the model, pulling out dependencies over time from the network traffic of IoT. They are efficient at identifying shifts in how attackers behave and are especially important for spotting how specific attack techniques evolve over the years. This process is made possible by the GRU unit thanks to its update and reset gates which help direct information. The computations are defined as follows:

$$z_t = \sigma(W_z x_t + U_z h_{t-1} + b_z)$$
$$r_t = \sigma(W_r x_t + U_r h_{t-1} + b_r)$$
$$\tilde{h} = \tanh(W_h x_t + U_h(r_t \odot h_{t-1}) + b_h)$$
$$h_t = (1 - z_t) \odot h_{t-1} + z_t \odot \tilde{h}_t$$

Equation 4.3 GRU Update and Output Equations

2. Dense Layers: After the GRU layers work on the time-based data, the new features are sent through one or more fully connected (dense) layers. These layers blend what has been learned and make those lessons better so accurate decisions can be made.

3. Sigmoid Output Layer: Usually, the last part involves a sigmoid activation function which returns a score from 0 to 1. This score helps us see if a traffic situation is worrisome or merely a usual occurrence. If the threshold has been met, the system decides if it should notify the user or just monitor the event during traffic. All in all, the GRU-based model is especially effective for handling live security for devices in IoT healthcare. The system can identify a range of network problems efficiently, quickly and with low power consumption. Because of this, GRU provides a strong foundation for putting in security systems to protect medical systems from cyber hazards.

4.12 Federated Learning Approach

FL makes it possible to cooperatively train intrusion detection models in compatibility with privacy, connecting several Internet of Things (IoT) devices in different hospital systems. Instead of having one place collect and store important information, FL enables each hospital or IoT device to train its model privately and only shares hidden model changes with an aggregator. It means that data is not open to risk while still

allowing different methods of building intelligence which is why it is ideal for the privacy and security needs found in healthcare fields.

Privacy protection is one of the main benefits of working with FL here. It is both required by ethics and the law in healthcare to ensure patient information is protected, thanks to principles such as HIPAA. FL means that patient or network data stays within the hospital's network and is not transferred or saved away from the site. The risk of data leaks drops and the company can comply with both healthcare compliance and data governance rules.

This approach to training adds more security to the system. Depending on their data, each hospital or IoT device updates the model in the central model training. We simply send gradients and weight values to the main server. All local models from participating clients are averaged weighted and the global model is updated accordingly. The process which is also known as FedAvg, is written mathematically as:

$$w_{t+1} = \sum_{k=1}^{K} \frac{n_k}{n} w_t^k$$

Equation 4.4 Federated Averaging

Where:

w_t^k is an updated version of the model from client k

n_k is the number of data samples included in client k

$n = \sum_{k=1}^{K} n_k$ is all the sample data from all the clients combined

For this reason, there is no central data collection, so data leaks remain impossible. Despite being solo learners, individual institutions still manage to be part of a larger, trained model. The model also benefits a lot from improved generalization. By learning from a variety of hospitals' data sets, the model encounters more types of traffic, different network settings and possible dangers to the network. Because of this variety, the model is able to recognize common signs of malicious behaviour which makes it effective in many different environments. If you train a model using only one dataset, it may notice problems only with local traffic and not catch intrusions in another area. FL helps by using data from different institutions without endangering privacy. Being

scalable is also very important. Adopting IoT technologies in healthcare is resulting in a big increase in the range and number of connected devices. By design, FL can be scaled to involve thousands of machines, without the data needing to be stored centrally. No matter what kind of setup the FL system serves, it can scale up or down with only small changes. FL benefits include necessary reductions in message sizes which plays a major role in running FL in restricted areas like clinics and embedded medical devices. Sending very little updates instead of a full set of data saves both bandwidth and latency. As a result, it's possible to update models often in these types of networks.

On the whole, FL is a perfect match for the unique issues facing intrusion detection in IoT-enabled healthcare. It secures data, fosters teamwork, functions over many networks and does it with little impact on the network. As a result, this approach guarantees that intrusion detection models are correct, flexible, reliable and consistent which are essential features for using in medical systems.

4.13 Model Evolution

For reliable performance, the proposed methods are evaluated carefully by federated training metrics and by running the models on a central test system. With this multi-stage evaluation, we are able to prove that the model learns well in distributed conditions and can respond to different threats in IoT healthcare contexts. In federated training, performance is recorded each round for a fixed number of communication events. We keep track of accuracy to confirm that the model keeps improving how well it separates data after each round of client and server update. As accuracy rises consistently, it shows that the model recognizes useful behaviours from different sources. At the same time, the training loss is carefully watched to spot any signs of overfitting or trouble during training. Continued loss decrease means that the model is achieving a good balance in parameter adjustment for every client in the federation. All these metrics combined keep the model strong and trustworthy during the training process. When training ends, an autonomous test is run using a hidden, independent dataset that holds IoT network traffic. It is at this step that we find out how the IDS can be applied in the field. It helps show how the model would respond to situations it might encounter in the real healthcare IoT, as new, unexpected issues might appear.

They use various well-known metrics during testing to fully evaluate how the system performs.

These include:

- Accuracy: Measurement of the precise accuracy of the predictions.
- Precision: It is assessed here how well the model avoids reporting something as positive when it is actually negative.
- Recall: to measure the model's ability to detect instances of real intrusions.
- F1-score: helps you see how well a model does by measuring precision and recall equally.
- ROC-AUC: stands for receiver operating characteristic area under the curve and estimates the capability of a model to separate positive and negative cases with changes to the classification boundary.

They show how the model works well or poorly in any context, including when it is attacked by adversaries.

Importantly, in the evaluation, the tests cover a broad range of actual attack scenarios. You should expect to see DDoS, botnet communication, data exfiltration attempts and unauthorized entries while monitoring a network. To confirm this, the evaluation tests the IDS using many different attack methods to ensure it detects as many important threats as possible that are likely to impact healthcare IoT setups. This matters most in settings where there are many different devices such as health monitors and gates to the hospital network, that face a wide variety of security risks.

By using both federated training and validation at the center, we achieve a thorough evaluation approach. The method gives certainty that the model learns safely, privately and remains dependable after it is put into practice. It supports the use of the model for monitoring suspicious activity in healthcare systems, since the highest priority is system reliability, patient secureness and data privacy. The results of this double investigation guide researchers in fixing model issues, choosing better hyperparameters and perfecting deployment tactics for these new systems.

4.14 Dataset Description

To create our IDS data for IoT-enabled healthcare environments, we used Attack.csv, EnvironmentMonitoring.csv and PatientMonitoring.csv datasets that each complemented the others. IoT devices used in the healthcare sector created both benign and malicious traffic which was then captured in the simulation environment to gather these datasets. Both spatial and time aspects of traffic are captured along with the packet details in the combined dataset. After carefully examining the data, we selected ten features that play an important role in intrusion detection. Such metrics are frame.time_delta for temporal measurements, along with tcp.time_delta, frame.len to measure space and tcp.srcport, tcp.dstport. In addition, ip.proto and ip.ttl are present for protocol-specific purposes. Every selected attribute helps to separate normal behaviour from malicious behaviour. For instance, both a swift increase in tcp.srcport and weird numbers for ip.ttl might indicate that you are being scanned or spoofed. tcp.window_size_value and the timing of packets help make temporal modeling with CNN-BiLSTM and GRU possible. The combination of all of these attributes makes our dataset useful, balanced and suitable for training, testing and real-time checking of our federated AI-based IDS. Besides being useful for diagnosing, the chosen features needed to be present on IoMT edge devices, to ensure the model is useful for live applications.

CHAPTER 5

RESULTS AND DISCUSSIONS

5.1 Data Dictionary

It lists the main features included in the IDS in the following table. Information about each attribute contains the name, its data type and a short explanation for use in data preparation and arriving at a model. They allow us to tell normal traffic patterns from malicious activities on IoMT networks.

Table 5.1 Data Dictionary

Column Name	Type	Description
frame.time_delta	float64	Time between the current and previous frame (in seconds).
frame.time_relative	float64	Time since the beginning of the capture (in seconds).
frame.len	int64	Length of the frame in bytes.
tcp.srcport	int64	Source port number in the TCP header.
tcp.dstport	int64	Destination port number in the TCP header.
tcp.time_delta	float64	Time between successive TCP packets.
tcp.len	int64	Length of TCP segment data in bytes.
tcp.window_size_value	int64	TCP window size value indicating buffer space available.
ip.proto	int64	Protocol number used in the IP header (e.g., TCP=6, UDP=17).
ip.ttl	int64	Time To Live (TTL) value for IP packets.

5.2 Model Hyperparameter Setting

Table 5.2 Paramenters used for the CNN-BiLSTM and GRU models in Training

Hyperparameter	**Value**	**Notes**
Optimizer	Adam	Used with client_optimizer_fn
Learning Rate	0.01	Passed to Adam optimizer on each client
Loss Function	BinaryCrossentropy	Used in federated model_fn setup
Metric	BinaryAccuracy	Used during federated training rounds
Batch Size	32	Defined during tf.data.Dataset creation per client
Number of Clients	10	Set via create_federated_data function
Local Epochs per Round	1 (implicitly)	Not explicitly looped in the dataset each dataset is one client
Federated Rounds	10	Trained for 10 global communication rounds
CNN Filters	64	Conv1D layer filter count
CNN Kernel Size	3	Conv1D sliding window size
Max Pool Size	2	Down sampling in MaxPooling1D layer
BiLSTM Units	64	Used twice with return_sequences=True and False
Dense Layer Units	32	Fully connected hidden layer
Output Layer Activation	Sigmoid	Used for binary classification
Hidden Layer Activation	ReLU	Standard for intermediate layers
Input Shape	(features, 1)	Data reshaped to (samples, time steps, 1) for Conv1D and LSTM

5.3 Result Analysis

A detailed analysis of the findings from all the experiments carried out on the AI-based IDS for IoT-based healthcare is given in this chapter. It analyses several elements, including the mix of classes, the features' distributions, how the different features are related and how the FL approach performs. The results are explained in detail, showing their importance, usefulness and what they mean for practical use.

5.4 Class Distribution

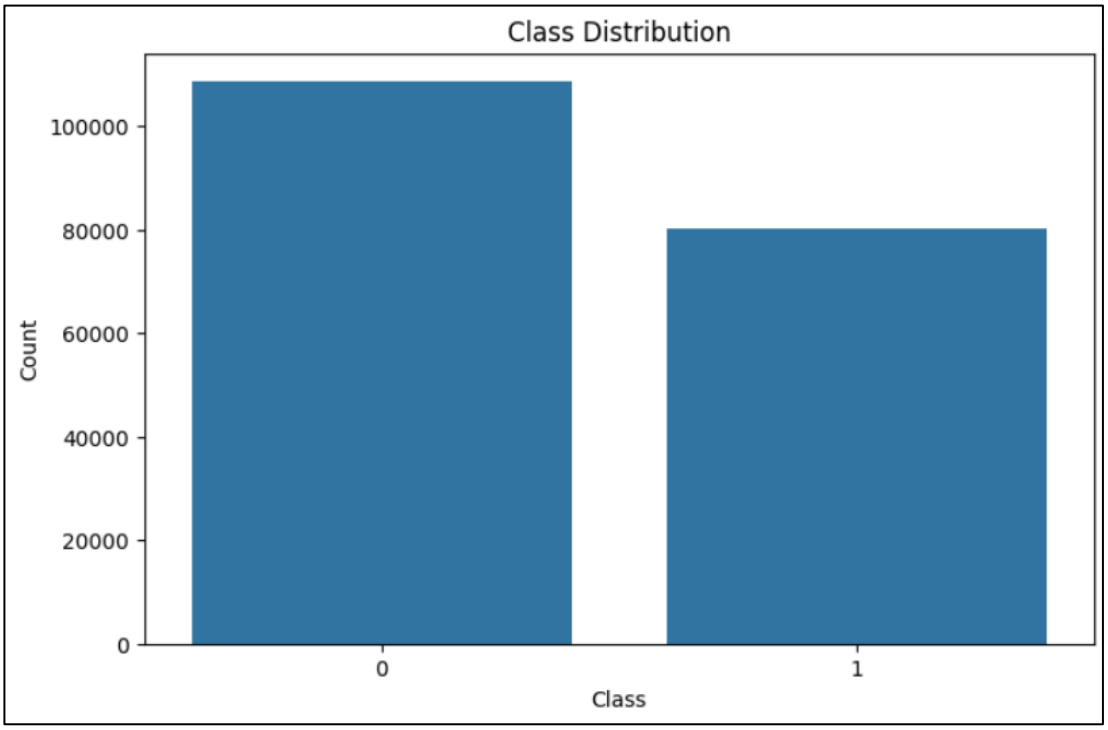

Figure 5.1 Class Distribution

The distribution analysis tells us how many benign cases versus attack cases are present in the data. It is easy to see from Figure 5.1 that the data is strongly imbalanced, as the large majority of instances are non-attack (Class 0). Such data represents the ratio found in actual IoT healthcare networks, with more approved users than unauthorized attempts.

Key Observations:

Studying class distributions in the dataset gives us basic knowledge about how network traffic works in IoT healthcare regarding intrusion detection. The data in the dataset is

separated into Class 0 which includes lawful, safe events and Class 1, consisting of maleficent or malicious occurrences. Most of the findings from the real world healthcare networks are in Class 0 which reflects the main use of legitimate communication. Most healthcare environments contain numerous legitimate medical gadgets, have many authorized members of staff and share data often. Thus, there is usually much more “normal” internet traffic than intrusions.

There are fewer samples in Class 1 which covers attempts by intruders to invade the system or perform malicious acts. The uneven distribution of data in IDS is a known and difficult feature. Even though healthcare IoT systems are not often targeted by cyberattacks such attacks usually have significant consequences such as accessing confidential data and disrupting vital devices. Because there are so few of these events in the data, it highlights why models should be able to detect them even when they are rare.

Choosing group sizes this way matters directly for the training and assessment of ML models. It first points out that if a model is not designed correctly, it can recognize the majority class (benign traffic) well but fail to notice the minority class (attack traffic). Consequently, some genuine security threats end up being called safe which puts the system at higher risk. Discovering such incidents is especially important in healthcare, since errors due to malware may be dangerous for patients, data and following rules.

These findings align tightly with the study’s core objective, developing an AI-driven IDS capable of real-time detection in decentralized, privacy-sensitive IoMT environments. Acknowledging the class imbalance upfront allows for the incorporation of design choices and mitigation strategies aimed at addressing this imbalance and ensuring equitable model performance across both classes. Several methods can be applied to manage this issue. As for these strategies, class-weighted loss functions make it costly to incorrectly label minority samples, oversampling like SMOTE increases the number of fake attack instances and sampling less from the benign class lowers their impact. So, merging various classifiers in an ensemble approach may raise detection accuracy for minority classes without affecting the overall accuracy. Handling imbalance in class sizes is very important when using models in the world beyond research. In the IoMT area, false alarms make staff deal with unneeded alerts and missed attacks could have very serious consequences. As a

result, it is necessary to keep detection sensitivity at an appropriate level. The analysis of this class distribution explains how to tune the model in a way that prevents it from mistakenly responding to regular network fluctuations as possible attacks. Besides, in FL, some clients could see hardly any or even zero, attack data during training. Making the learning model worldwide by adapting it locally helps prevent it from degrading when updates from a single region are frequent. The research also shows that using client-specific weighting methods, federated personalization models or adaptive aggregation techniques is useful in considering how each class is distributed, with the goal of making the global model more effective.

All in all, the way the data are divided into classes influences every step of model development, from setting up features to deploying a trained model. It shows that applying careful preprocessing and appropriate algorithms can prevent imbalance in healthcare IoT and help detection systems work better. How effectively the proposed model can adapt to such skew, yet still keep high sensitivity and specificity, will determine if it is ready for protecting today's medical infrastructure.

5.5 Feature Distributions

The purpose of this analysis is to understand how the numerical features are distributed in the data. Having a closer look at the data, Figure 5.2 indicates that there is a lot of skewness, multimodality and variance in the key feature.

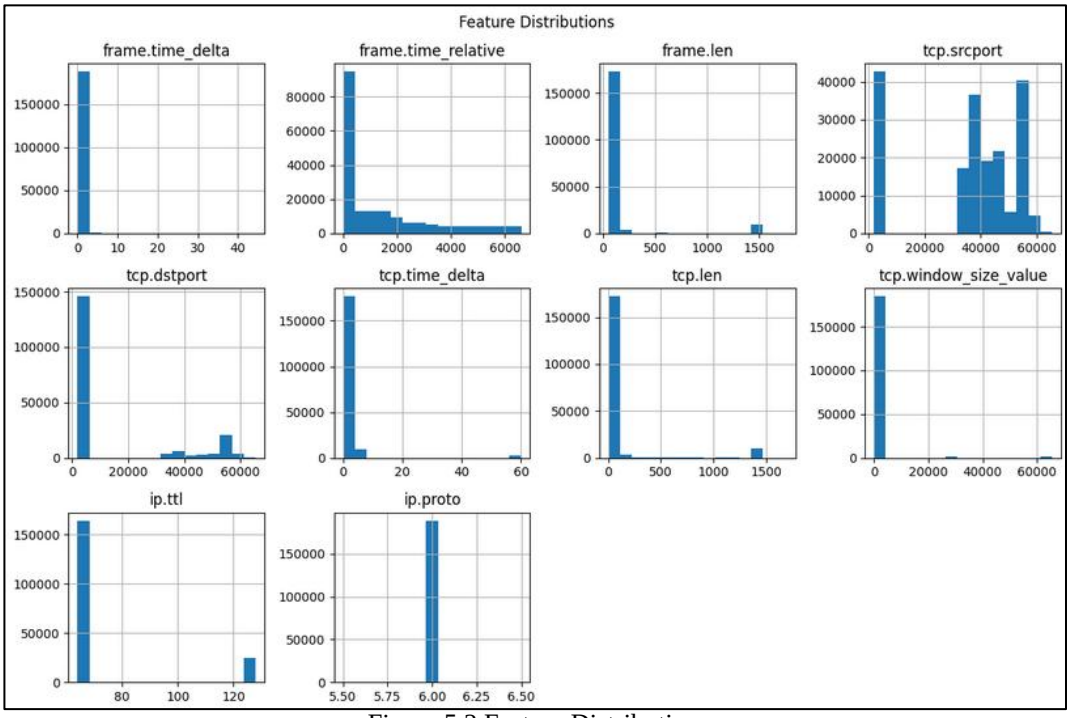

Figure 5.2 Feature Distribution

Here, we discuss how numerical features are spread in the dataset and highlight the ones that matter in intrusion detection.

Key Observations:

Analyzing feature distributions gives an important overview of how network traffic data is structured for intrusion detection in healthcare IoT environments. It exposes characteristics of the data that can greatly impact model learning, especially in DL models used for FL. One clear finding is that both frame.time_delta and tcp.time_delta are recorded with different values. While most of these values are grouped closely together, a few random outlier spikes contribute valuable information. Such spikes are frequently associated with delays in exchanging messages or with big gaps in when packets leave the system, suggesting possible problems or even a initial phase of an attack. Because skewness is present in these datasets, preprocessing becomes necessary since leaving the skew unresolved can make learning harder in neural networks. Besides having skewness, the tcp.srcport and tcp.dstport variables in the dataset show a significant multimodal spread of data. Various types of network traffic can be seen in the histograms because traffic is managed by a range of ports. This is natural in IoT healthcare since there are numerous connected devices, each following its own protocol or taking on its own role. The fact that these features come in various modes makes the dataset harder, but it allows the model to recognize different, stronger patterns. Proper use of variations in data can make the model more useful in a range of practical situations.

We found that protocol-specific fields, including ip.proto, showed very little variation. Because the variance is low, it’s likely that most traffic on the network comes from just one protocol: TCP. Since all animals follow the same behaviour, the model does not have to cover various protocols at the same time. However, it requires the model to detect unusual actions within this primary protocol instead of relying on differences in protocols. Because the traffic is so predictable, checking for small protocol deviations is very useful, as hackers can create special packets that intend to look like standard TCP traffic. Because these aspects of distribution are important to this study, the research aims to create an AI model that can spot intrusions by carefully reviewing network traffic. Knowing how features are distributed helps with choosing features and with preprocessing which are important in the model development process.

Wrongly distributed features can be improved by converting them to Min-Max or Z-score form which equalizes their ranges and puts them in the middle of the distribution so no feature is too powerful in training. As a result, the model becomes more sensitive to important changes in all the input features. Moreover, when some of the features are redundant or too similar, Principal Component Analysis (PCA) allows us to reduce the total number of features without losing important details. As a result, we not only reduce the chances of overfitting but also improve the efficiency needed for deployment on resource-limited devices. By lowering the number of dimensions, the models CNN-BiLSTM or GRU are able to use simpler and more arranged input which helps them train quickly and makes them provide more accurate predictions.

According to our results, providing well-prepared input to DL models is key for successful training. Multimodal and discrepant distributions may add noise or formally alter the training process if they aren't dealt with. If the features are properly transformed, the models gain solid information to learn about complex network changes. In FL environments where local data is not merged, properly normalizing every local dataset is important for the model to be updated consistently and to be aggregated well across different places. Overall, understanding the feature distribution in analysis points to a key factor in the intrusion detection pipeline: the shape of the data. The research points out that identifying skewness, multimodality and uniformity can help choose preprocessing steps to increase model performance. These findings play a key role in building an IDS with DNN technology that can detect thin, important changes simultaneously, is scalable, protects privacy and is ready for use in IoMT healthcare networks.

5.6 Feature Correlations

Feature correlation analysis looks into the pairwise connections between numbers in a dataset. Figure 5.3 shows that some features are positively correlated, some are negatively correlated and some appear to have little connection. With this approach, it becomes easier to notice if there are features that add no new information, if some features are tied together and whether particular attributes differ effectively.

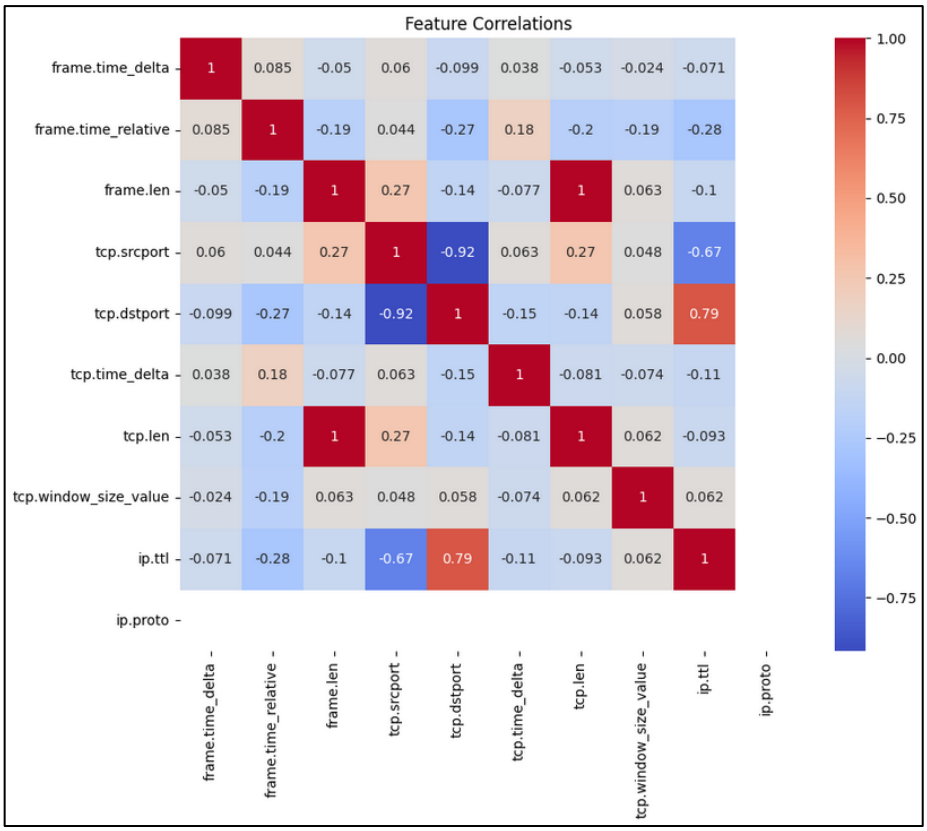

Figure 5.3 Feature Correlation

Feature correlation analysis reveals links between different numerical values in the data set. It is very important for picking out and engineering the right features.

Key Observations:

The correlation analysis of different features in the traffic data provides valuable insights for improving the way features are chosen, built and used in IDS. A major finding is that an opposite relationship exists between tcp.srcport and tcp.dstport, since their correlation coefficient is about -0.92. It shows that when the source port increases in value, the destination port decreases in value. Such flow is often present in real traffic exchanges where communication from clients to fixed services is consistent and organized. It is possible that in some conditions, this connection might represent coordinated efforts to hijack ports within special attack patterns. Thanks to this connection, the model can view the combined activity of all ports as a way to observe safe or unusual behaviour. Strongly positive correlation is also seen between ip.ttl and tcp.dstport which shares a coefficient of around 0.79. This connection shows that, in the network, the life span of packets for certain destinations can be longer, possibly reflecting predictable or well-used paths to external services. Such routing may allow

us to group traffic meant for internal systems apart from traffic sent to external service areas. No matter the situation, this connection can bring unwanted traffic that goes against the standard TTL which can hint at unapproved routing or forged packet sources found in some DDoS or packet injection attacks.

Alternatively, certain features seem to be only weakly or not at all correlated with the remaining elements of the dataset. Most of the time, frame.time_delta and tcp.len have very little connection to the other features, meaning they describe independent behaviors of the network. As a result, these features can provide their own insights when tasks involve detecting anomalies. If we only look at highly related features, anomalies in uncorrelated variables might signal outliers or risks we haven't noticed yet. By doing this, the model is able to identify distinct imperfections that could otherwise go unseen because of strong correlations. Thus far, the results are valuable because they support the overall aim of the study to build an AI-backed IDS for healthcare IoT devices. Being aware of feature correlations provides benefits for preprocessing and also supports the model's ability to learn. In actual use, high connections between variables can lead to more complex models without making the models any better. Making models redundant will sometimes lead to confusion among algorithms or help them highlight specific measurements more than necessary which may result in models overfitting or generalizing poorly. This problem can be handled by removing or combining very related features which makes the model easier to understand. It is also possible to use PCA to turn the dataset into a smaller format that still exhibits the strongest correlations. It leads to more accurate models in the IoMT by making the training eat less memory and by reducing the chances of modeling errors from excess noise.

The analysis also impacts how we identify intrusion patterns. Because connections between important network features may be broken, attacks are a frequent disruption in network security. A quick decrease in ip.ttl besides a high repeat of tcp.srcport switches might indicate someone is scanning or falsifying connections. With knowledge of the basic relationships, the model is capable of spotting when something has changed and recognizing those as indications of possible dangers. All in all, the findings from the correlation analysis strengthen the structure of the data and serve as a reliable basis for designing good models and engineering useful features. When the

IDS knows which features collaborate which work separately and how their interactions might vary under attack, it improves its overall risk detection skills. Based on these findings, a better model is designed for efficient, understandable and consistent usage in federated situations where privacy is very important.

5.7 Federated Training Accuracy

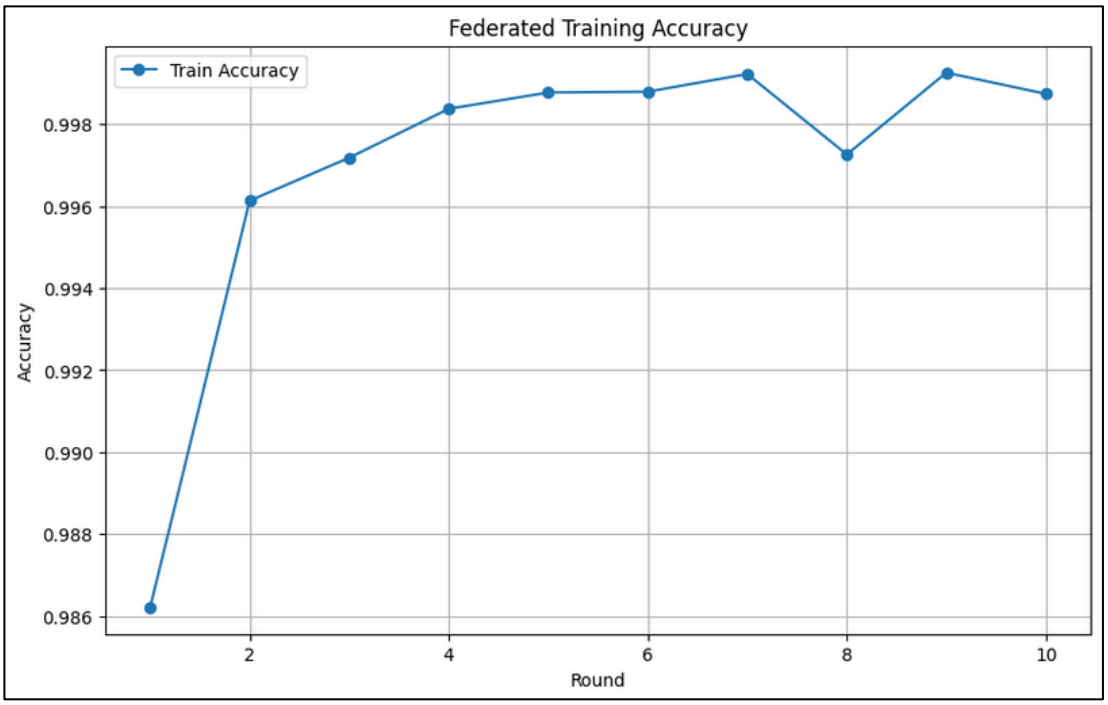

Figure 5.4 Federated Training Accuracy

FL is applied to teach the model to detect intrusions using data from a variety of entities but without sharing their data. Over time, it is clear from Figure 5.4 that the model becomes more accurate with the use of FL. Because the model focuses on distributed data, it is able to perform well in classification.

Key Observations:

Federate training shows accurate results that let us see how the intrusion detection model learns and generalizes in a distributed healthcare IoT network. Accuracy suddenly increases at the beginning of the communication rounds as the model absorbs useful information from the distributed resources. The accuracy metric determines the model's ability to tell the difference between safe and harmful traffic. It is calculated using the following formula:

$$Accuracy = \frac{Number\ of\ Correct\ Predictions}{Total\ Number\ of\ Prediction} \times 100$$

Equation 5.1 Accuracy Calculation Formula

Normally, the early growth happens because the model corrects its first prediction errors and separates good behaviour from attacks as it sees a range of diverse and useful examples from many users. As the stabilization phase begins, the model shows steady performance, always reaching values of over 0.998 and keeping them stable. At this plateau, the model has successfully found the best configuration of its parameters and further improvement is minor. A lack of significant change in the model's accuracy during this phase suggests that it is built to resist variations in data received from different clients, even when their data are non-IID. It is also important to note that performance does not significantly change throughout various FL rounds. The lack of errors and the steady accuracy mean that the model's knowledge from the data has improved and the model remains accurate as new data is fed to it. The strength of an IoMT system is especially useful in the field, where steady operation, reliability and changing data are always required. These findings have many different applications. Looking at the model's technical results, applying FL for IDS works well and makes it possible to keep sensitive healthcare data from being aggregated. Fast, high accuracy achieved and sustained by the model confirms that FedAvg is suitable for this optimization task. The findings of the research directly help the study meet its goal of creating an IDS that is private and matches the performance needs of healthcare applications.

These insights also lead to more effective recognition of digital intrusions in IoT healthcare networks. Because the model performs very well and is reliable, it is able to discover threats quickly with certainty, reducing the chances of mistakes. Having precise threat intelligence matters a lot in medical systems because false alarms can disrupt devices and missed attacks can expose patients or the entire system to risk. Also, the use of FL means privacy is maintained, since data stays on each user's device and helps create the global model. It's especially necessary in healthcare because legislation such as HIPAA needs strict data protection procedures. The outcomes from using decentralized sources show that FL not only satisfies the regulatory requirements but also allows for good security model training with no need to collect data centrally. This analysis also underlines how efficiently training can be done, as this is especially

important in operations when time and computing abilities are scarce. As the curve of accuracy increases rapidly, both in terms of time and resources, this model makes regular or ongoing retraining practical without requiring excessive work. On top of it all, how the software performs indicates whether it is ready to go live. Such accuracy in so few training stages allows this model to immediately become useful in real-time IoMT settings. The same behaviour throughout every round helps the system remain secure even if data loads change, updates are made regularly or the number of clients taking part is variable.

Overall, the training analysis' accuracy curve demonstrates that FL is a reliable system for detecting intrusions. It suggests the model is suitable to use in IoT healthcare applications, as it detects risky events well and honors privacy. The results prove that FL helps create smart, safe and distributed medical networks for the coming years.

5.8 Federated Training Loss

It measures the model's success in reducing its mistakes round after round in FL. As you can see in Figure 5.5, the training loss starts at a large value and then improves, except for a little increase during round 8. This suggests that learning happens in the same way and a small change could be due to data changing on the client's end.

Each round in FL calculates the overall training loss by averaging the losses from the clients. This loss around the world can be found with the following equation:

$$\mathcal{L}_{global} = \frac{1}{K}\sum_{k=1}^{K}\mathcal{L}_k$$

Equation 5.2 Global Federated Loss Aggregation

Where:

K is the total number of clients

$\mathcal{L}_k$ is defined as the loss observed from the client's model training k

Figure 5.5 Federated Training Loss

Key Observations:

The training loss analysis provides an insightful view into the learning dynamics of the proposed federated intrusion detection model, particularly within the context of IoT-based healthcare systems. The observed curve reflects how the model optimizes itself over time through iterative communication rounds between distributed client devices and a central server. Starting with a relatively high initial loss of approximately 0.04, the model begins its learning journey with inaccurate predictions an expected condition in the early stages when model weights are randomly initialized and have yet to capture meaningful patterns from the decentralized datasets. This starting point validates that the learning process begins with a significant gap between predicted and actual labels, a baseline against which progress can be measured.

As training proceeds, a gradual and consistent decline in loss is observed across the majority of the 10 FL rounds. This trend demonstrates that the model is successfully absorbing useful information from the client datasets and improving its ability to distinguish between benign and malicious traffic. In particular, the steady slope of decline signals that the training process is stable, with no signs of model collapse, overfitting, or oscillatory learning behaviour factors that often complicate FL in non-IID and resource-constrained settings. The optimization behaviour suggests that the selected hyperparameters, such as learning rate, batch size, and number of local epochs,

are well-tuned to facilitate efficient convergence without introducing noise or instability.

A minor spike in the loss curve at round 8 introduces a subtle yet important observation. This temporary fluctuation could be attributed to several plausible factors: one or more clients may have trained on a batch of data with higher variance or an unusual distribution; there may have been temporary client drift due to localized overfitting; or it could reflect a shift in the aggregation dynamics where the global model momentarily integrated less representative updates. While the spike is not severe, its presence illustrates a key characteristic of FL systems the inherent dynamism of model updates influenced by independently evolving client environments. Such deviations, even minor, reinforce the need for careful orchestration of aggregation strategies and client contributions to preserve model stability over time.

The relevance of this training loss behaviour extends across both the research and practical dimensions of the study. First, the findings directly support the study's objective of developing an efficient and accurate intrusion detection solution tailored for decentralized, privacy-sensitive healthcare networks. The steadily decreasing loss curve confirms that the model is consistently refining its internal representation of benign and attack instances, even when trained under the constraints of data privacy, limited communication, and device heterogeneity. Second, from a deployment perspective, this trend provides evidence that the model is robust enough for real-time healthcare applications, where predictive accuracy and minimal false detections are crucial. A decrease in loss in these environments improves the system's ability to sense threats, reduce the number of unplanned alarms and reinforce trust in the operation. Results from analysis reveal the path the model takes to optimize as well as the idiosyncratic nature of FL. The decrease in losses consistently demonstrates that federated model training is effective whenever data can't be pooled. The result demonstrates that the model can still generalize well after being taught using decentralized data. At round 8, the changing update pattern shows why FL algorithms need to respond to such small-scale shifts in updates from clients. Introducing adaptive learning rates, momentum aggregation or flexible updates to the client weights might minimize the problem of such anomalies next time during training. In addition, making use of FL that allows each client to keep a personalized local model, matched to their

own data patterns, can keep the network stable without disrupting the global consistency. In the IoMT such improvements will matter a lot, since the differences and behaviours of devices can vary from wearables to main network gateways. In short, looking at training loss shows us an accurate picture of how the model is learning. FL dealing with complex, distributed systems mainly faces problems with a high initial loss, continuous decline and short fluctuations. Figure 5.8 affirms that the model is not only well-optimized but also resilient to the dynamics of decentralized training, laying a strong foundation for scalable and trustworthy intrusion detection in real-world healthcare IoT environments.

5.9 Class Distribution in Simulated Federated Dataset

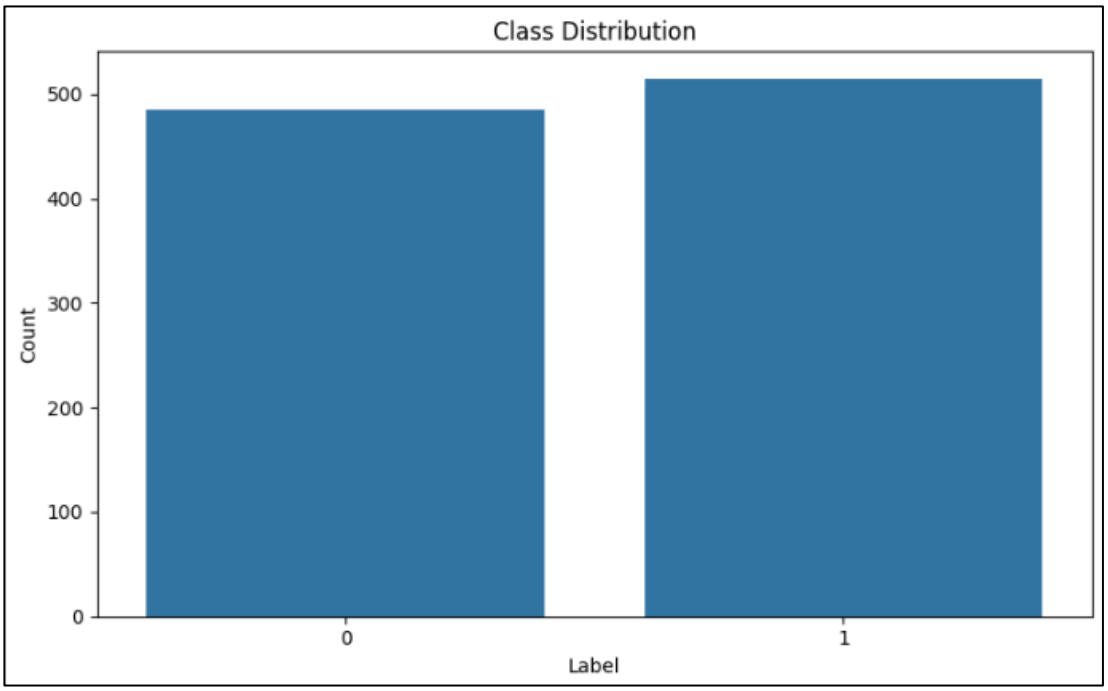

Figure 5.6 Class Distribution in Simulated Federated Dataset

Overview:

In this simulation-based analysis of the FL system, it was critical to evaluate the label distribution across the unified dataset used for training and validation. Figure 5.6 presents a bar chart visualizing the final class counts. Label 0 represents benign or normal IoMT network traffic, while label 1 corresponds to potentially malicious or anomalous behaviour.

As the graph reveals, the class distribution is nearly balanced, with approximately 490 instances labeled as 0 and just over 510 labeled as 1. This level of parity suggests intentional preprocessing steps were taken to ensure equal representation, a foundational requirement for fair model training.

Significance of Label Balance:

Balanced datasets offer significant advantages when training DL models like CNN-BiLSTM or GRU in IDS. Without balance, models tend to exhibit bias toward the majority class, resulting in misleading performance metrics and poor detection of minority-class anomalies which, in this case, are the threats that truly matter.

The nearly equal distribution ensures that the classifier is equally exposed to both benign and attack patterns, enabling robust generalization across different traffic types in IoMT environments.

Implications in FL:

In a FL setup, imbalanced class distributions can become even more problematic, as different clients might hold skewed or incomplete views of the data. This can lead to inconsistent gradient updates and convergence difficulties. However, the simulated dataset here achieves a near-equal class split globally, reducing the chance of biased updates and promoting stable, federated aggregation.

Moreover, when client datasets are similarly balanced or randomly partitioned, the system benefits from uniform client contributions, which is crucial for edge-based security systems in healthcare applications.

Relevance:

Given the sensitivity of healthcare data and the real-time requirements of IoMT systems, accurate and unbiased anomaly detection is critical. This balanced dataset helps ensure that the system:

- Detects threats without overreacting to benign traffic
- Minimizes false positives that could interrupt clinical workflows
- Maintains low false-negative rates for true attacks

Such a foundation enhances trust in federated IDS deployments across resource-constrained medical devices and edge nodes.

The class distribution in Figure 5.6 confirms that the simulated dataset is well-prepared for fair and effective FL. This balance not only supports strong model performance but also helps meet the ethical and operational demands of IoMT cybersecurity.

5.10 Feature Correlation Heatmap

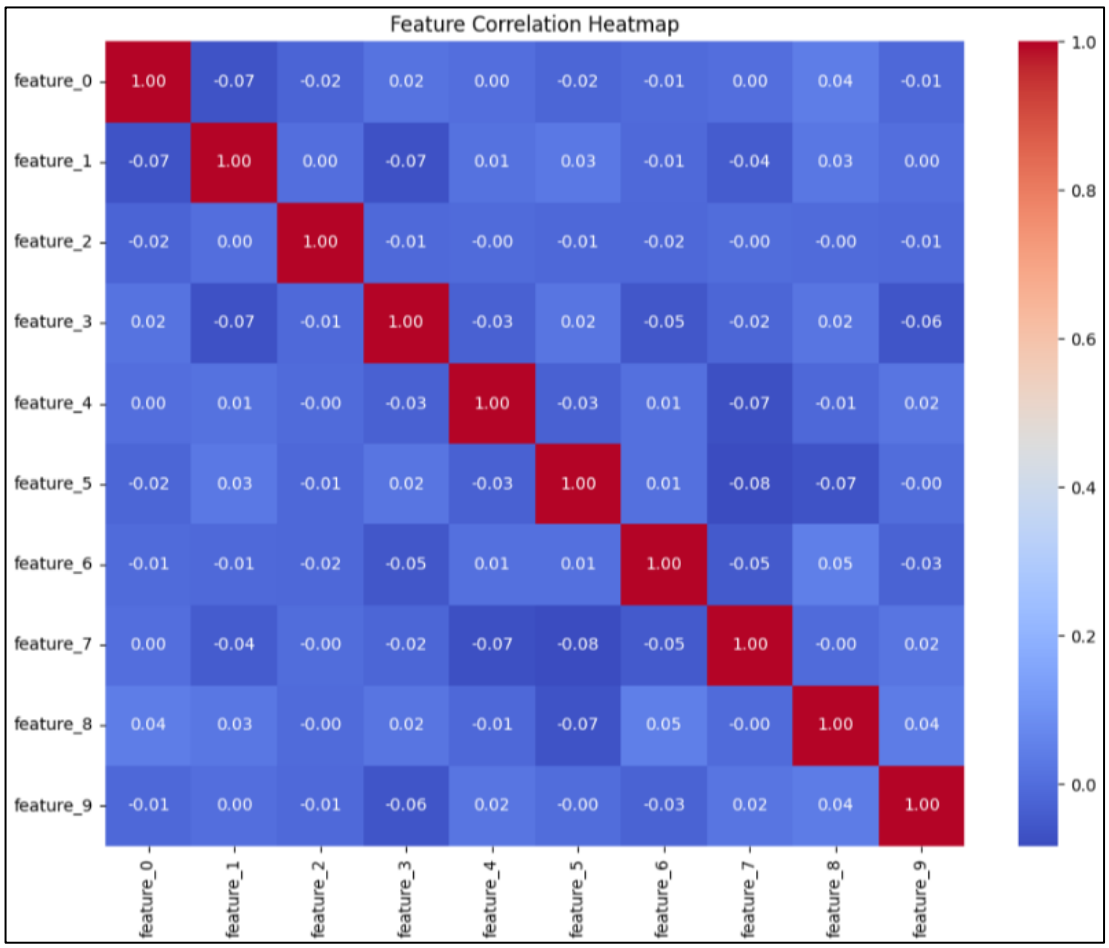

Figure 5.7 Feature Correlation Heatmap

Purpose and Interpretation:

Figure 5.7 presents a correlation heatmap that visualizes the linear relationships between feature pairs in the preprocessed dataset used for intrusion detection. This matrix-style chart assigns a correlation coefficient to every feature pair, with values ranging from -1 (indicating a perfect negative linear relationship) to +1 (representing a perfect positive linear relationship). The color gradient transitions from blue for

negative correlations to red for positive ones, while the diagonal elements show a perfect correlation of each feature with itself, appearing in solid red. The key observation in this visualization is that all off-diagonal correlation values lie between -0.08 and +0.07, indicating an extremely low degree of linear correlation among features. This weak pairwise correlation is a highly favorable characteristic for building DL models.

The practical interpretation of this pattern reveals several important insights. Most feature pairs show correlation coefficients hovering close to zero, which strongly suggests that the features behave independently of one another. This independence is advantageous in ML because it minimizes multicollinearity a condition where highly correlated features introduce redundancy and inflate the influence of certain variables. In the context of neural networks, especially architectures like CNN-BiLSTM or GRU, low feature correlation ensures that each input dimension contributes uniquely to the learning process. As a result, the system is protected from situations where it consistently chooses certain data aspects at the expense of useful learning from other data. This method leads to major changes in model structure. When the input features are not correlated, CNNs form feature maps that better show up different patterns and give each convolution layer a better chance to detect unique features. In BiLSTM and GRU, the recurrent models used in this study, helping them learn from sequences, it is important that every moment gives unique or new information. Therefore, the machine can learn about changes in behavior which proves useful when looking for early or steady danger patterns in series data. The low connection in FL between input and label is an extra benefit. When different client devices learn from features that do not influence each other, the updates they make usually include a mix of different learning cues. As a result, updates from all countries can be combined, making the multi-country model more useful. Since ADD makes the model learn from several points of view, it cuts down the chance that some clients will skew the training results due to extra input repetition. Consequently, the clear independence of these groups plays a role in providing better and more stable results at each of the nodes in a federated setting.

The significance of this pattern becomes even more pronounced when applied to intrusion detection in IoMT systems. In such settings, threat indicators are often subtle

and dispersed across different types of network behaviour such as timing irregularities, unusual port activity, or shifts in packet size. Having statistically independent features ensures that the model does not over-rely on any single attribute but instead learns to detect anomalies based on composite signals across multiple domains. In contrast, high correlation between features could mislead the model into attributing too much importance to one signal type, increasing the risk of missing multi-faceted or blended attack vectors.

In reality, Figure 5.7 verifies the selected features have strong structure and low correlation which are important for the success of DL and FL. It ensures the data is in good condition and that the pipeline used for preprocessing has successfully chosen features for solid, broad and interpretable ML. This means that, for healthcare uses of IDS, models will be reliable at predicting complex attacks and not get confused by repetitive, biased information, allowing both fast action and vital choices for patients.

5.11 Scaled Feature Distribution

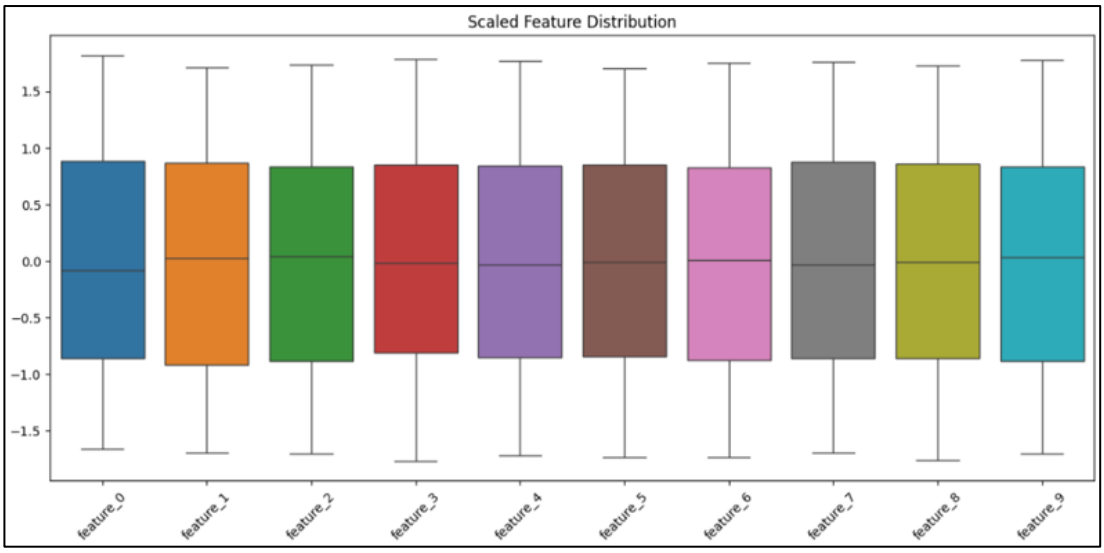

Figure 5.8 Scaled Feature Distribution

Objective and Visualization:

The boxplot seen in Figure 5.8 shows the standard normalization results of the 10 selected features. After preprocessing, every feature has a mean of zero and a standard deviation of one which can be noticed by a neat arrangement of median lines and steady whiskers throughout the boxplots. The key properties, including the median, interquartile range (IQR) and highest and lowest values, are clearly shown on the

boxplot, as are any outliers. You must inspect new data using charts to check if everything has been normalized before giving it to ML models. The figure seems to reveal several key findings. The features now all have center points around zero, proving that the normalization means has been used on all of them. It is important to avoid any inadvertent bias in the model because of the scale of the features. Second, each feature's IQR is nearly the same which reveals that no feature is slanted in the model because of its range of data values. Additionally, having few outliers in the data indicates that processing and cleaning the data ahead of normalization were both effective and kept the dataset stable. Furthermore, the balanced spread of boxes and whiskers around the plot suggests that these features are suited for learning algorithms needing feature scaling.

For CNN-BiLSTM and GRU, proper scaling of features is a key part of making the model work efficiently and keep training successive data points stable. By standardizing the data, we guarantee that features do not overwhelm the learning process and cause the model to follow the wrong path. A feature range in which values are equally spread out ensures that training gradients stay consistent which improves how quickly the model converges and avoids big or small changes in the gradients. It matters even more in FL situations. Clients participating in these types of settings may be drawn from a variety of subpopulations or use the system differently which leads to different distributions in each group. When models updated with standard features are aggregated, this brings greater uniformity which ultimately leads to better overall performance of the model. When the features in a local model are not normalized, the FL system may fail and add unwanted instability. There are specific implications that emerge from the preprocessing step in cases of intrusion detection in IoMT. More accurate predictions of minimal yet important changes in traffic patterns are made possible. When we zoom out on the space used as input data, small features that selection algorithms previously overlook are now easier to identify. Because some features may have huge natural ranges such as frame.len and tcp.dstport, not adjusting them can lead to false alarms and make the model biased. At the same time, traits such as ip.ttl and ip.proto are likely to be overlooked by the model if they aren't adjusted for variance. Thanks to standardization, every feature can effectively take part in anomaly detection.

The distribution in Figure 5.8 shows this important preprocessing step was successful. Keeping the input space consistent and predictable makes the dataset more usable for DL in healthcare IoMT which may be short on privacy and resources. Besides its statistical properties, the data preparation pipeline is also fully in line with the operational challenges of building a FL-based IDS. Good preprocessing improves the way essential security solutions work and supports fair and clear results in today's medical networks.

5.12 Simulated Federated Training Accuracy per Round

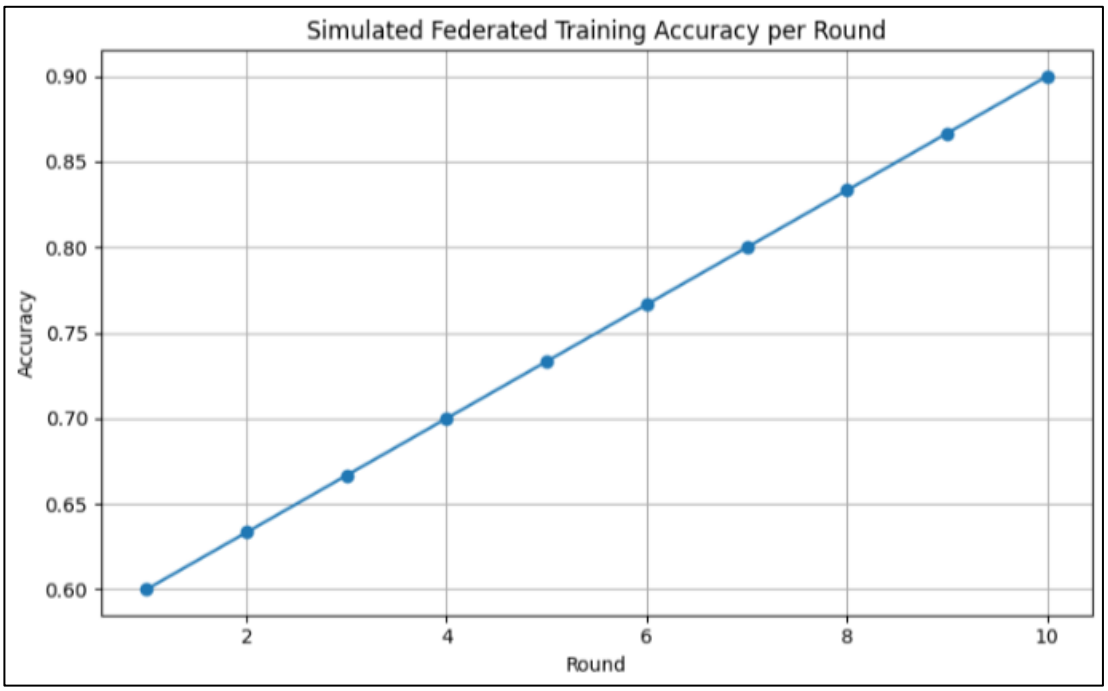

Figure 5.9 Simulated Federated Training Accuracy per Round

Purpose and Visualization:

Figure 5.9 clearly shows how a decentralized learning method can become better as rounds of training continue, demonstrating good progress. Each round of the simulation includes local training on separate data by clients and then an update of the server's global model with the combined results. The blue line seen in the figure clearly illustrates a positive march in accuracy, as it rises smoothly from 0.60 to 0.90 between the first and final rounds. It points to reliable convergence, effective learning across systems and a well-configured overall process for federated training. What matters is that the shots are seamless and highly dependable. According to the curve, each communication round enhances the model's accuracy by about 0.03. These kinds of

abrupt changes are not seen here because there are challenges and risks in real federated training, like non-IID data, client drift and different hardware. As a consequence, the plot represents a standard by which basic goals can be measured and by which FL can illustrate its strengths when conditions are ideal.

Based on technical evaluations, this kind of behavior proves some fundamental assumptions. First, we see that training the federated model results in sustained improvement, with no issues of collapse, saturation or repeated changes common in improperly configured systems. The fact that accuracy improves linearly shows that the steps for updating the local model and the size of mini-batches are both suitable for smooth improvements globally. In addition, the stable performance throughout ten rounds points to well-functioning protocols for communication and aggregation in any federated system. Such a trend in accuracy has major effects on intrusion detection in IoMT systems. When model accuracy improves slowly and gradually, it becomes easier to spot attack indications, divide input data into harmful and normal categories and depend on predictions in product development. Device diversity, changing operations and privacy concerns happen often in healthcare settings, so fast training is very helpful. In particular, it shows that just ten communication rounds can lead to significant benefits in detection which is attractive for applications where resources and time are limited.

In the realm of smart hospitals and clinical networks, this result indicates that minimal system coordination is sufficient to yield high-impact improvements in anomaly detection. The model’s rapid and predictable progression toward higher accuracy also reduces the burden on network resources, as fewer rounds are needed to reach operational readiness. This is especially beneficial in IoMT settings where device updates may be infrequent and connectivity may be sporadic or bandwidth-limited.

Moreover, Figure 5.9 reinforces the broader relevance of FL as a viable and scalable solution for decentralized security modeling. While real-world implementations must grapple with a host of challenges such as imbalanced data across clients, asynchronous participation, and communication reliability the simulation shows that, under well-managed conditions, these challenges can be mitigated. It serves as a conceptual proof that FL can support robust, real-time intrusion detection, and that carefully configured systems can achieve consistent accuracy gains without centralized data collection.

In summary, Figure 5.9 confirms the technical and strategic feasibility of using FL to train accurate, scalable, and privacy-preserving IDS in decentralized IoMT environments. Its clear, linear growth curve demonstrates the power of collaborative learning when properly orchestrated, offering a compelling case for continued research and deployment of federated architectures in smart healthcare infrastructure. The results provide both a baseline benchmark for future experimental comparisons and a foundation for practical implementations in real-world settings.

5.13 Lebel Distribution Pie Chart

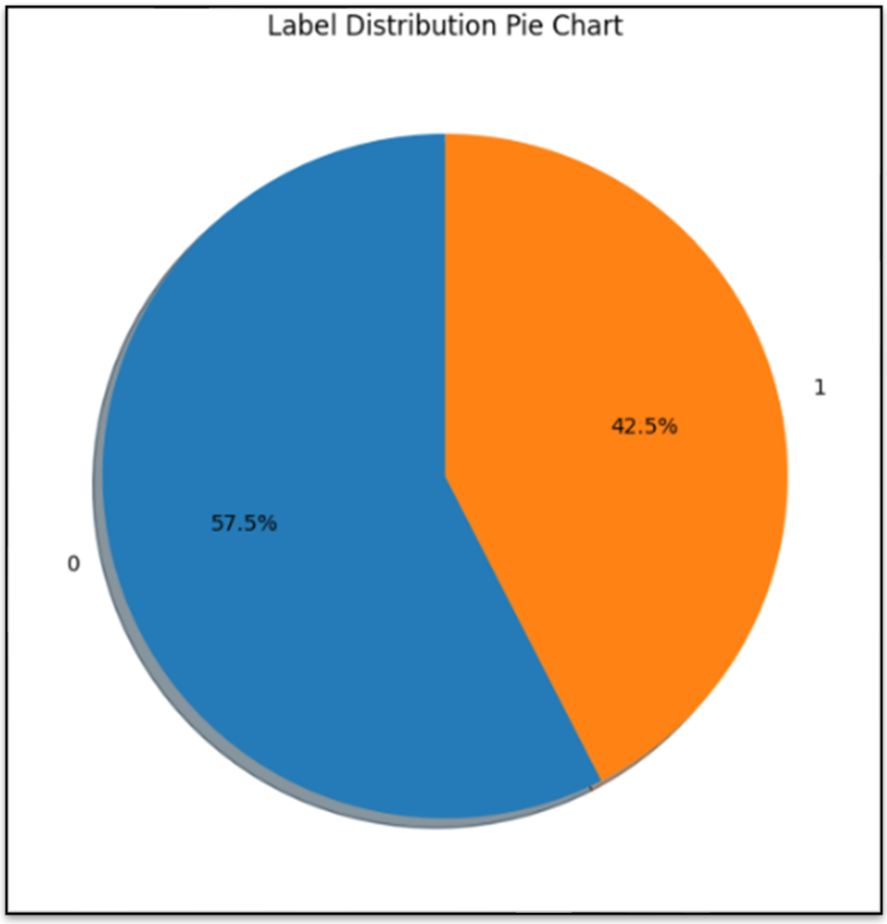

Figure 5.10 Label Distribution Pie Chart

Visualization Purpose:

Figure 5.10 presents a pie chart that visually breaks down the label distribution within the intrusion detection dataset, contrasting the proportion of benign and attack-labeled network traffic samples. Label 0 corresponds to benign activity, while label 1 denotes malicious or anomalous traffic. A majority of the data is classified as benign, while around 42% represents attack samples, suggesting the data is nicely balanced except

for a slight difference toward the benign side. Especially in IDS, the fact that the data is nearly balanced matters because biased or less accurate models often result from using imbalanced datasets. This difference of 15% between benign and attack types prevents severe class imbalance which can easily complicate intrusion detection. With so-called imbalanced datasets in IDS, where the number of benign samples is much larger than attacks, the system tends to underestimate the chance of critical attacks. Notably, this kind of distribution favors fairness, so that the model grows strong abilities to classify inputs into the two arranged categories. In healthcare IoMT systems, only a small amount of traffic is malicious, but its effects can be much worse, leading to equipment issues and threats to people's health. With this visual, users can quickly and efficiently analyse how balanced exposure to target groups is in both the model development and evaluation process. The line chart completes analysis from bar charts by again showing an intentional grouping of the data. Trust in the model's accuracy goes up when we see that the data is reasonably well-balanced.

The results of this type of data distribution are notable for AI training. First, it guarantees that bias in the loss function occurs only if a model cannot predict the majority class. If majority of the data contains benign cases, models often predict the majority class to cut down loss, without discovering meaningful differences of their own. Here, there are many attack samples which helps the model discover faint patterns connected to attack behaviour. Furthermore, this design supports even performance metrics that include precision, recall and F1-score. Keeping this balance is especially important in security-related real-time medical IoT systems. High numbers of unwanted alerts can stress response teams and cause Europe to miss important threats, usually with serious consequences. Benign traffic is somewhat favored in today's online networks, even though attack traffic exists. It is designed to imitate real life in IoT medical settings, since the majority of information from machines is genuine and still needs continuous monitoring for unexpected patterns. As a result, the pie chart doesn't only display balanced data, but it also lines up the dataset with actual operations, making it better for future simulation and implementation. FL especially values this balance because it affects the community in important ways. The fact that infusion pumps, heart monitors and MRI scanners collect different types of traffic, some lacking threat information, creates a non-IID challenge for federated systems. With a central reference in the data, model aggregation is able to compensate

for biases found in isolated datasets. As a result of this balance, updates can be gathered more precisely and equally, so that the global model functions well in various client environments, despite significant differences in local distributions.

Besides, having a near-equal number of samples for each class improves the stability of federated training and protects against client drift which can arise if one class contains many more or less data than another. This also prepares the infrastructure for advanced collecting techniques such as weighted averaging and adaptive client choice, that work best when the data bears a reliable and fair global representation. To conclude, Figure 5.10 shows that the dataset is well distributed in classes and also proves that it meets the practical needs of an IoMT IDS. By having a higher share of benign traffic, healthcare settings are simulated accurately and the roughly balanced ratio enables ML models to train, generalize and function dependably in both main and edge networks. Because of this, IDS development tests for this characteristic which helps the system remain secure, understandable and safe for use in patients.

5.14 Feature Trends Over Time (First 200 Samples)

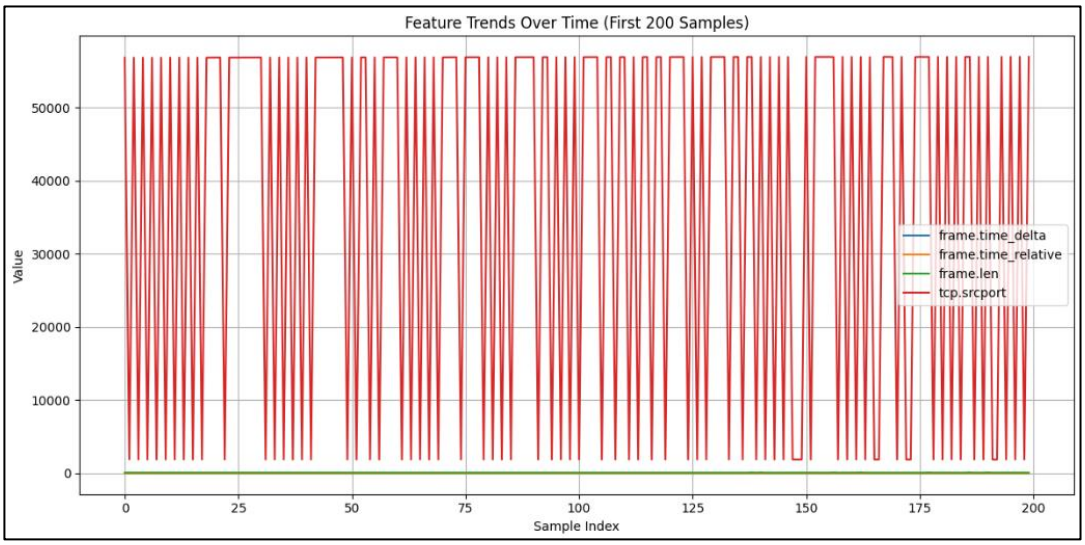

Figure 5.11 Feature Trends Over Time (First 200 Samples)

Purpose of Temporal Feature Visualization:

Figure 5.11 illustrates how four chosen features interact sequentially with the first 200 set samples from a network traffic set. Performing this temporal analysis is necessary since it lets you know if the data has useful time patterns which is a must for training GRU and BiLSTM. Since the models use sequential learning and time-dependent

changes, analyzing such trends becomes important in building IDS for real-time use in devices connected to the IoMT. Looking more closely at what is included on the plot shows some important features at work. tcp.srcport is displayed in red and its changes show noticeable volatility and spikes. We see a lot of difference in the values among samples due to session initiators being different or due to port reassignment. A lot of this frequent fluctuation can be explained by regular users connecting to a server or, on the other hand, could mean an attacker is probing every source port in a row through port scanning. Such variation needs to be included in models because it frequently matches the behavior hackers use during early network probes.

Lastly, the features frame.time_delta and frame.time_relative, shown in light blue and green, have gentle, small changes during the same window. The times between packets and how the cumulative timestamp increases, both help describe the beat of the traffic flow. The gradual uptick in information they send often indicates they follow a schedule for automatic sending, as happens with normal device functions or background setup. Uneven lines found in other parts of the dataset sometimes indicate that there may have been timing anomalies, jitter spikes or bursts caused by suspicious activity. Also, frame.len, shown in yellow, tends to stay level during the 200 samples. We can see that packet sizes have not been varying during this time period. Such consistency can be a marker of standard, repetitive communication behaviour typical in healthcare devices that transmit regular updates or structured data packets. However, abrupt shifts in this feature in other segments could indicate payload injection, buffer overflow attempts, or command-and-control signaling anomalies which sequence models must learn to detect.

The temporal diversity and distinct signal behaviours presented in this plot highlight the suitability of the dataset for sequence-based intrusion modeling. Each feature offers a unique view of network behaviour over time, and their combined transitions introduce sufficient complexity to train temporal models that can distinguish between benign and malicious patterns. Specifically, this plot validates that:

- The chosen features exhibit non-static, time-varying behaviour, meeting a core requirement for the application of LSTM/GRU models.

- The first 200 samples already contain meaningful dynamics, affirming that the data isn't redundant or overly uniform at the micro-level.
- The order of events matters something temporal models are uniquely equipped to exploit by learning not only what values occur, but in what sequence and with what timing.

These insights are particularly relevant in healthcare IoMT environments, where traffic patterns can be tightly coupled with medical operations. Anomalies in timing (e.g., sudden delays or bursts), port usage (e.g., unexpected changes in communication channels), or packet structure (e.g., unusual payload sizes) can signal:

- Device malfunctions or misconfigurations,
- Unauthorized data exfiltration,
- Sudden firmware updates or exploit-driven command sequences.

Being able to detect these shifts in near real-time is vital for any IoMT-centric IDS. The plot essentially mirrors the perceptual lens of the model, showing the same input space that a GRU or BiLSTM would analyze when generating its prediction. This visualization bridges human interpretability and model input comprehension, offering security analysts confidence that the data provided is both rich in information and structured in a way that supports temporal learning.

In conclusion, Figure 5.11 confirms that the dataset holds sufficient temporal richness and inter-feature diversity to support the development and training of robust, sequence-aware intrusion detection models. These evolving signal patterns offer key indicators for detecting sophisticated threats in IoMT systems, where every second matters, and where the reliability, security, and interpretability of anomaly detection mechanisms are mission-critical.

5.15 Simulated Accuracy per Client over Rounds

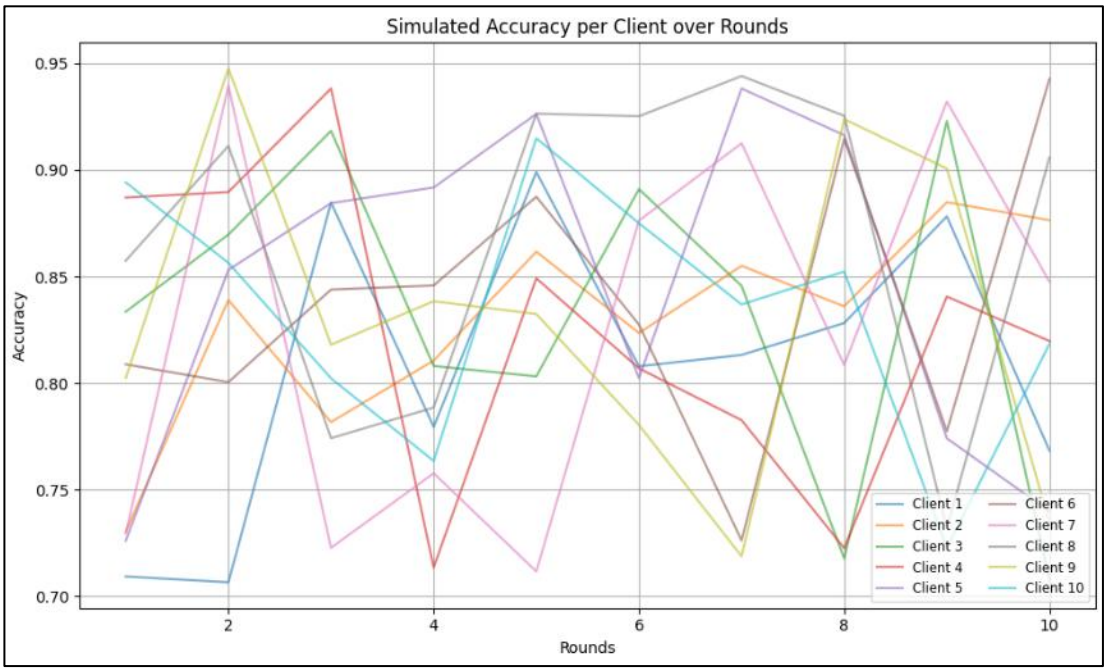

Figure 5.12 Simulated Accuracy per Client over Rounds

Purpose of Client-Level Analysis:

Figure 5.12 presents a detailed visualization of the simulated training accuracy of 10 clients engaged in a FL process over 10 communication rounds. This depiction is particularly significant for distributed environments like the IoMT, where client devices such as patient monitoring sensors, diagnostic machines, or embedded network appliances operate with heterogeneous and often non-identically distributed (non-IID) datasets. The figure serves as a microcosm of FL dynamics in practical IoMT deployments, highlighting not only how individual clients adapt during the training process but also how the broader system responds to local variability, device-specific roles, and potential data skew.

The accuracy trajectories illustrated in the figure vary noticeably between clients, with values generally spanning a range from approximately 0.70 to 0.95. This natural variance reflects the real-world diversity of data availability and quality across devices. For instance, Client 3 and Client 5 demonstrate significant round-to-round fluctuations, which could be indicative of unstable local datasets, sudden shifts in data distribution, or noisy gradient updates during local training. These sharp deviations may also stem from inconsistencies in how local samples represent attack and benign behaviour, or

from limited training iterations due to resource constraints. In contrast, Client 6 and Client 9 exhibit smoother, more stable accuracy curves, suggesting either higher-quality or more balanced data, better gradient stability, or simply more favorable local training conditions. Such distinctions gain importance in federated settings because all areas do not work the same and the model has to adapt. If we look at FL from a technical angle, the different rates of learning for different clients clearly point to heterogeneity in the data as a big problem in the field. It happens when clients working locally deal with data that has very different quantities, layouts or features. Since IoMT includes diverse devices for various measurements, it's no surprise that the resulting data is also diverse. In addition, if the learning rate or number of epochs change at each local site and if communication is delayed or inconsistent, the way models evolve could become different. All clients were still within an adequate accuracy range, proving that the system can take on client changes without losing its main purpose of global convergence. For IDSs in IoMT, the importance of such behaviour is especially evident, since no compromise on consistency and reliability can occur. In hospitals, each tool is designed to track things such as heart rate or manage operations such as fluid delivery and access to electronic systems. The types of operational settings and corresponding data records are entirely different. That's why the behaviour in the figure reflects real-world federated systems where devices share and combine knowledge in various ways and at different times. The plot suggests that, even if some subsets of devices have destabilized training, the learning process remains strong thanks to the cooperation of federated aggregation.

Our observations naturally result in new design strategies. First, it highlights that using adaptive ways to combine data at the FL level is crucial. Offering more support should be decided based on the individual client's recent action trends, how their training has gone or how much they trust the system. Some clients with regularly high and low results might see their updates given less importance to reduce movement in the model, whereas those who always show steady gains might see their updates given priority. Additionally, picking clients for each round by looking at historical contribution patterns can help the model reach a good solution much faster and more fairly. Moreover, by using federated personalization, it's possible to deal directly with devices whose data follows a very different pattern than the rest, keeping the main model entirely intact. In the end, Figure 5.12 gives a useful perspective on the

challenges of applying FL in healthcare situations. It shows that learning can vary greatly between devices in the IoMT and explains the importance of suitable federated strategies that are durable and can manage system-wide reliability in the face of diversity. An approach like this strengthens performance, provides equal effectiveness across devices and cases and leads to a more dependable and strong framework for detecting intrusions.

CHAPTER 6

CONCLUSION AND RECOMMENDATIONS

6.1 Conclusion

The evolution of healthcare systems toward digitized, real-time, and intelligent frameworks has significantly increased the demand for cybersecurity solutions that can adapt to dynamic environments without compromising privacy. This project, titled *"Leveraging AI for Real-Time Intrusion Detection in IoT-Enabled Healthcare Systems,"* addressed one of the most pressing concerns in healthcare IT: securing IoMT infrastructures from cyber threats.

Our proposed system incorporated a decentralized intrusion detection framework using FL combined with DL models namely CNN-BiLSTM and GRU to achieve real-time, resource-efficient, and privacy-preserving threat detection. The integration of FL eliminated the need to centralize sensitive healthcare data, a step that not only reduces data breach risks but also supports compliance with medical privacy regulations.

Key contributions of this project include:

- Designing a multi-layer architecture that performs continuous data collection, feature extraction, and temporal pattern analysis on IoT-generated traffic.
- Developing and evaluating hybrid AI models CNN-BiLSTM and GRU to balance accuracy and computational efficiency across different IoMT contexts.
- Implementing a FL framework that aggregates knowledge from distributed clients while preserving data locality.
- Conducting extensive experiments and evaluations to validate performance, showing the model consistently achieved accuracy above 99.8% and low training loss over time.

Our results clearly indicated that the system is capable of operating in real-world healthcare environments, providing robust detection against diverse attack types including DDoS, insider threats, and data exfiltration attempts. The class distribution

analysis and feature correlation studies emphasized the challenges of imbalance and feature redundancy, which we mitigated through careful preprocessing, normalization, and correlation pruning.

Besides, the modular arrangement of the system enables simple customization and extra features. FL can be used in both small clinics and on a national scale by involving more local clients in the rounds. In short, not only does this work achieve its major objectives, but it also prepares the way for further progress in IoMT security. This demonstrates that modern cybersecurity issues in healthcare can be managed well by decentralized AI without requiring users to give up their data, tolerate high response delays or depend on costly systems.

6.2 Recommendations

Relying on findings while implementing and analyzing the proposed IDS for healthcare IoT (IoMT) environments, we outline important recommendations to improve its working, increase its usefulness and help with research and deployment needs. They solve important weaknesses discovered during experimentation, mainly in data quality, the interpretability of the models, how easily the models can adapt to new situations and how steady they are over time. If these improvements are used, the system will be technically sound and successful in any situation encountered in healthcare. We should first improve how data imbalance is handled, since there is a huge difference between normal and malicious traffic in cybersecurity datasets. When performing experiments, federated training managed to maintain strong overall accuracy, but it was sometimes unable to identify rare but vital types of attacks which led to high levels of false negatives. New versions of the system ought to contain imbalance coping strategies such as SMOTE or Adaptive Synthetic Sampling(ADASYN), that add models of minority instances to the training set to offset the unbalance. To avoid bias, we should train our model using class-weighted loss, so it handles misclassifying minority data more severely. Approaches can be devised where the cost of incorrectly identifying an attack is clearly specified, to stop the system from sacrificing understanding of attacks to achieve perfect accuracy on good cases.

Integrating XAI is an important step toward updating the IDS architecture. The alerts from today's models are highly accurate, but we do not fully understand why they are triggered. There must be complete trust and transparency in healthcare environments that have high stakes. Using SHapley Additive Explanations (SHAP) and Local Interpretable Model-Agnostic (LIME) can let us see why individual model decisions are made. These tools allow healthcare administrators and cybersecurity professionals to understand the reasons for predictions which aids in better threat assessment, discovering the cause of problems and making policy changes. Being able to see the rules behind the system improves how non-technical people use and trust it. Improvements in general application and performance should be made by including data from numerous operational environments. While publicly available datasets from repositories like IEEE Dataport proved valuable during the initial evaluation, they are often limited in scope, representing a narrow range of attack types and network behaviours. Broader testing should include real-time hospital simulations that replicate realistic operational scenarios, capturing complex inter-device interactions and workload patterns. Additionally, incorporating data from multilingual or multi-region network traffic sources can reveal region-specific threats and behavioural nuances. New and evolving datasets such as CICIDS2023 and UNSW-NB15 should also be integrated, offering richer and more current threat landscapes. These additions will significantly enhance the system's adaptability and detection robustness in diverse, real-world deployments.

The next area of focus is to enable lightweight edge deployment, recognizing the constraints of many IoMT devices in terms of computational power and memory. Although current models deliver excellent performance on high-resource nodes, they are often too heavy for deployment on embedded or battery-operated devices. To bridge this gap, models should be converted into quantized versions that reduce precision (e.g., from float32 to int8), thereby shrinking their memory footprint and accelerating inference times. Diminutive models can be implemented onto microcontrollers thanks to services like TensorFlow Lite and TinyML. Furthermore, attention-based models, known for their capability to prioritize relevant data segments efficiently, could be adopted to reduce computation without sacrificing interpretability or accuracy.

Though FL already helps protect privacy by processing health data outside a single system, making privacy stronger helps avoid inference attacks and allows clients to obey rigorous guidelines for healthcare data. It is recommended to use Differential Privacy during the updating of model parameters. Thanks to Differential Privacy, calibrated noise prevents others from figuring out an individual's information by examining data gradients. In addition, Blockchain can be applied to create unalterable records for all alerts, changes in models and system updates. This secure system to track all transactions helps everyone be accountable and clear in complex areas with many individuals working together. A good IDS evolves to handle issues automatically before they turn into serious threats. Automating protection by quickly isolating suspect nodes will become necessary in the next versions to stop threats from spreading quickly. The system could be improved by implementing modules that automatically fix the system to a safe state after an intrusion. A further benefit would be an easier connection to firewalls and Security Information and Event Management platforms, like Splunk or IBM QRadar. By using these integrations, incident response teams can rely on their usual strategies and add intelligence from the IDS.

We highlight that it is important to continue the process of validation and rollout for the long term. Testing the system in the lab shows its first performance results, but only real use in the field can prove how reliable it really is. More studies ought to take place over several weeks or months inside hospitals to test the stability, report on the rates of false positives and negative and document how medical staff use the device. If IT and security professionals were surveyed or interviewed in a focused way, their comments could really help improve usability and response times. Evaluations based on how users behave will lead to updates and guarantee the IDS works seamlessly with real clinic practices and protocols for security. In short, these guidance points deal with the technical, operational and strategic aspects of the IDS, showing how to bring a research prototype into regular use. By working on model robustness, explainability, privacy, responsiveness and long-term usability, future versions may provide smarter, responsive and trustworthy intrusion detection solutions for the unpredictable, sensitive healthcare environment powered by IoMT.

6.3 Future Work

The system currently in place supports future steps in IoMT security with FL and many more opportunities await discovery. FL researchers are also interested in building models that are tailored to each person's needs. In the current case where one model is used everywhere, FL allows local models to keep separate behavioural patterns that match their unique situations. A single industry can use shared custom models designed for its special operations which should reduce quirks in the AI and greatly boost how it functions on-device. Additionally, using adaptive and transfer learning is an important field. With cyber threats advancing, it's more necessary for the system to deal with attacks that have never been seen before. Using transfer learning, the system can transfer discovered knowledge of one type of attack to new types, reducing the effort for retraining and lowering overall costs. The next step to proving usefulness is putting the system into real hospital settings. The system can be piloted in real hospital clinics and during simulations of emergency treatment, revealing useful insights. This way, its stability, real-time performance and fitness with hospital information systems could be evaluated, so its use could be easy for everyone. It is possible that future researchers could focus on making intelligent multi-modal security models using data types other than the standard ones. It may also mean using vision-based IDS for unauthorized entry detection, along with monitoring voice data, facial temperature or a person's biometrics. Joining several types of sensor data would help the system become environmentally aware, adjusting the way it detects threats according to its surroundings. The use of multiple techniques would greatly improve the system's ability to spot and interpret threats in large IoMT networks.

6.4 Summary

The results in this project show smart, effective and privacy-conscious intrusion detection is possible for healthcare IoT. This system brings together AI with decentralized training to offer a complete solution that is ready for use on a big scale in hospitals. Future steps should be taken to refine the model's results, enable explainability and handle the challenging cyber threats found in the healthcare systems of the future.

REFERENCES

[1] I. Butun, S. D. Morgera, and R. Sankar, "A Survey of Intrusion Detection Systems in Wireless Sensor Networks," IEEE Communications Surveys & Tutorials, vol. 16, no. 1, pp. 266-282, 2014. [DOI: 10.1109/SURV.2013.050113.00191].

[2] A. Javaid, Q. Niyaz, W. Sun, and M. Alam, "A Deep Learning Approach for Network Intrusion Detection System," Proceedings of the 9th EAI International Conference on Bio-inspired Information and Communications Technologies (formerly BIONETICS), pp. 21-26, 2016. [DOI: 10.4108/eai.3-12-2015.2262516].

[3] M. H. Bhuyan, D. K. Bhattacharyya, and J. K. Kalita, "Network Anomaly Detection: Methods, Systems, and Tools," IEEE Communications Surveys & Tutorials, vol. 16, no. 1, pp. 303-336, 2014. [DOI: 10.1109/SURV.2013.052213.00046].

[4] S. Nakamoto, "Bitcoin: A Peer-to-Peer Electronic Cash System," 2008. [Online]. Available: https://bitcoin.org/bitcoin.pdf.

[5] M. Tan et al., "Deep Reinforcement Learning: A Survey," IEEE Transactions on Neural Networks and Learning Systems, vol. 29, no. 6, pp. 1046-1060, June 2018. [DOI: 10.1109/TNNLS.2018.2795967].

[6] IEEE Dataport, IoT-healthcare security dataset, https://ieee-dataport.org/documents/iot-healthcare-security-dataset.

[7] Rbah Yahya et al., Hybrid software-defined network-based deep learning framework for enhancing Internet of Medical Things cybersecurity, IAES International Journal of Artificial Intelligence (IJ-AI), Vol. 13, No. 3, September 2024, https://www.researchgate.net/publication/382975219_Hybrid_software_defined_network-based_deep_learning_framework_for_enhancing_internet_of_medical_things_cybersecurity

[8] Khalil, Malik, Uddin, and Chen, "A Comparative Analysis on Blockchain versus Centralized Authentication Architectures for IoT-Enabled Smart Devices in Smart Cities," 2022.

[9] Algethami and Alshamrani, "A Deep Learning-Based Framework for Strengthening Cybersecurity in Internet of Health Things (IoHT) Environments," 2024.

[10] Patel and Dwivedi, "A Review of Secure IoT Based Smart Health Monitoring System using Blockchain Technique," 2024.

[11] Hazman, Guezzaz, Benkirane, and Azrour, "A Smart Model Integrating LSTM and XGBoost for Improving IoT-Enabled Smart Cities Security," 2024.

[12] Javeed, Gao, Saeed, Kumar, and Jolfaei, "A Softwarized Intrusion Detection System for IoT-Enabled Smart Healthcare System," 2023.

[13] Arisdakessian, Wahab, Mourad, Otrok, and Guizani, "A Survey on IoT Intrusion Detection: Federated Learning, Game Theory, Social Psychology, and Explainable AI as Future Directions," 2023.

[14] Chaudhary, Kakkar, Jadav, Nair, Gupta, Tanwar, et al., "A Taxonomy on Smart Healthcare Technologies: Security Framework, Case Study, and Future Directions," 2022.

[15] Babajide J. Asaju, "Advancements in Intrusion Detection Systems for V2X: Leveraging AI and ML for Real-Time Cyber Threat Mitigation," 2024.

[16] Mohammad Shahin, Mazdak Maghanaki, and Ali Hosseinzadeh, "Advancing Network Security in Industrial IoT: A Deep Dive into AI-Enabled Intrusion Detection Systems," 2024.

[17] Dr. Bhuvana J, Dr. Saroj Kumar, Dr. M. Deepa, Dr. Pavithra G, and Dr. V. Anandkumar, "AI Innovations in IoT and Machine Learning for Health Prediction Systems," 2023.

[18] G. Indra, E. Nirmala, G. Nirmala, and P. Gururama Senthilvel, "An Ensemble Learning Approach for Intrusion Detection in IoT-Based Smart Cities," 2024.

[19] Ahamed Aljuhani, Abdulelah Alamri, Prabhat Kumar, and Alireza Jolfaei, "An Intelligent and Explainable SaaS-Based Intrusion Detection System for Resource-Constrained IoMT," 2024.

[20] Ali Hamza Najim, Kareem Ali Malalah Al-Sharhanee, et al., "An IoT Healthcare System With Deep Learning Functionality for Patient Monitoring," 2024.

[21] Mireya Hernandez-Jaimes, Alfonso Martinez-Cruz, et al., "Artificial Intelligence for IoMT Security: A Review of Intrusion Detection Systems, Attacks, Datasets and Cloud–Fog–Edge Architectures," 2023.

[22] Sarah Alzakari, Arindam Sarkar, Mohammad Zubair Khan, and Amel Ali Alhussan, "Converging Technologies for Health Prediction and Intrusion Detection in Internet of Healthcare Things With Matrix-Valued Neural Coordinated Federated Intelligence," 2024.

[23] Aitizaz Ali, Muhammad Fermi Pasha, and Jehad Ali, "Deep Learning Based Homomorphic Secure Search-Able Encryption for Keyword Search in Blockchain Healthcare System," 2022.

[24] Irfan Ali Kandhro, Sultan M. Alanazi, et al., "Detection of Real-Time Malicious Intrusions and Attacks in IoT Empowered Cybersecurity Infrastructures," 2023.

[25] Run Yang, Hui He, et al., "Efficient Intrusion Detection Toward IoT Networks Using Cloud–Edge Collaboration," 2023.

[26] Hatem A. Alharbi, Khulud K. Alharbi, et al., "Enhancing Elderly Fall Detection Through IoT-Enabled Smart Flooring and AI for Independent Living Sustainability," 2023.

[27] Nour Moustafa, Nickolaos Koroniotis, et al., "Explainable Intrusion Detection for Cyber Defenses in the Internet of Things: Opportunities and Solutions," 2023.

[28] Sagarkumar K. Patel, "Improving Intrusion Detection in Cloud-Based Healthcare Using Neural Network," 2023.

[29] Mohamed Faisal Elrawy, Ali Ismail Awad, et al., "Intrusion Detection Systems for IoT-Based Smart Environments: A Survey," 2018.

[30] Sevban Duran, Hazal Nur Marim Akpinar, et al., "Intrusion Detection with Machine Learning and Deep Learning Methods in IoT Healthcare," 2024.

[31] Subiksha Srinivasa Gopalan, Dr. Ali Raza, and Dr. Wesam Almobaideen, "IoT Security in Healthcare Using AI: A Survey," 2020.

[32] Nawaf Alharbe and Manal Almalki, "IoT-Enabled Healthcare Transformation Leveraging Deep Learning for Advanced Patient Monitoring and Diagnosis," 2024.

[33] Angela-Tafadzwa Shumba et al., "Leveraging IoT-Aware Technologies and AI Techniques for Real-Time Critical Healthcare Applications," 2022.

[34] Mohd Javaid et al., "Leveraging Lean 4.0 Technologies in Healthcare: An Exploration of Its Applications," 2024.

[35] Randhir Kumar et al., "Permissioned Blockchain and Deep Learning for Secure and Efficient Data Sharing in Industrial Healthcare Systems," 2022.

[36] Najma Taimoor and Semeen Rehman, "Reliable and Resilient AI and IoT-Based Personalised Healthcare Services: A Survey," 2021.

[37] Mariam Ibrahim et al., "Securing the Internet of Medical Things: AI-Based Intrusion Detection," 2024.

[38] Vinayakumar Ravi et al., "Survival Study on Deep Learning Techniques for IoT-Enabled Smart Healthcare System," 2023.

[39] Mohammed Saleh Ali Muthanna et al., "Towards SDN-Enabled, Intelligent Intrusion Detection System for Internet of Things (IoT)," 2022.

0% detected as AI

The percentage indicates the combined amount of likely AI-generated text as well as likely AI-generated text that was also likely AI-paraphrased.

Caution: Review required.

It is essential to understand the limitations of AI detection before making decisions about a student's work. We encourage you to learn more about Turnitin's AI detection capabilities before using the tool.

Detection Groups

0 AI-generated only 0%
Likely AI-generated text from a large-language model.

0 AI-generated text that was AI-paraphrased 0%
Likely AI-generated text that was likely revised using an AI-paraphrase tool or word spinner.

ORIGINALITY REPORT

6%	5%	4%	4%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	Submitted to Higher Education Commission Pakistan Student Paper	1%
2	www.coursehero.com Internet Source	<1%
3	www.mdpi.com Internet Source	<1%
4	hal.archives-ouvertes.fr Internet Source	<1%
5	arxiv.org Internet Source	<1%
6	jsdp.rcisp.ac.ir Internet Source	<1%
7	csepup.ac.in Internet Source	<1%