INSTITUTO
UNIVERSITÁRIO
DE LISBOA

Federated Learning for mHealth: an exploration

Guilherme Santos Fernandes Carvalho

Master's in Computer Science and Engineering

Supervisors:
Doctor Ana Maria de Almeida, Associate Professor,
Iscte – Instituto Universitário de Lisboa

Doctor José Miguel de Oliveira Monteiro Sales Dias, Full Professor,
Iscte – Instituto Universitário de Lisboa

October, 2024

TECHNOLOGY
AND ARCHITECTURE

Department of Information Science and Technology

Federated Learning for mHealth: an exploration

Guilherme Santos Fernandes Carvalho

Master's in Computer Science and Engineering

Supervisors:
Doctor Ana Maria de Almeida, Associate Professor,
Iscte – Instituto Universitário de Lisboa

Doctor José Miguel de Oliveira Monteiro Sales Dias, Full Professor,
Iscte – Instituto Universitário de Lisboa

October, 2024

Acknowledgment

I would like to express my heartfelt appreciation to my supervisors, Professor Ana Maria de Almeida and Professor José Miguel Dias, for their invaluable guidance, support and encouragement throughout the course of this dissertation. Their expertise and insights have been instrumental in shaping my work and I am deeply grateful for the opportunity they provided me to be a part of the AIMHealth project.

I extend my sincere gratitude to Otávio, Susana and Professor Mauricio for their insights, experience and unwavering support throughout the study. Their feedback and perspectives greatly enriched this work and helped me to approach the research with a more comprehensive perspective.

To my family, thank you for your unwavering encouragement, patience and belief. A special thanks to my parents, who were not only a constant source of support but also never hesitated to give me the reality checks I needed, pushing me to be the best version of myself. Your support has been my anchor throughout this journey, providing strength and stability through every challenging moment.

To my friends, thank you for being there every step of the way. From keeping me company in the library and making those long study sessions feel less tedious, to sharing laughs during coffee breaks and moments that helped me recharge, you made this journey so much more enjoyable.

Finally, I am deeply grateful to everyone who offered their support during this chapter of my life. Thank you!

Resumo

A monitorização remota de pacientes emergiu como uma solução valiosa no contexto da Covid-19, possibilitando serviços de saúde mais acessíveis e abrangentes. Através da recolha de dados fisiológicos e informações de saúde, a tecnologia mHealth pode ser utilizada para acompanhar pacientes com doenças crónicas, detetar anomalias e prever eventos de saúde. Esta dissertação teve como objetivo desenvolver uma aplicação de mHealth baseada em Inteligência Artificial, denominada AIMHealth, que é capaz de suportar Aprendizagem Federada em tempo real, visando a deteção de anomalias relacionadas à Covid-19 em dados de frequência cardíaca em repouso. A implementação da Aprendizagem Federada permite a monitorização de saúde descentralizada, garantindo a privacidade dos dados dos utilizadores. Este trabalho explorou o uso de modelos autoencoder e testou várias estratégias para aprimorar a precisão na identificação de anomalias, juntamente com adaptações para preparar a aplicação para uso no mundo real. O desenvolvimento do AIMHealth representa tanto um avanço quanto um desafio na saúde digital, oferecendo uma abordagem promissora para o monitoramento remoto de pacientes e a identificação de padrões de saúde relevantes.

Palavras-chave: Saúde Móvel, Aprendizagem Federada, Deteção de Anomalias, Covid-19, Monitorização Remota de Pacientes, Inteligência Artificial

Abstract

Remote monitoring has emerged as a valuable solution, particularly in the context of Covid-19, by providing accessible healthcare services to a broader population. Through the collection of physiological and health data, mHealth technology can monitor patients with chronic illnesses, detect anomalies and predict health events. This dissertation aimed to develop an AI-based mHealth application, AIMHealth, capable of supporting real-time Federated Learning (FL) to detect Covid-19 anomalies in rest heart rate data. By implementing FL, AIMHealth enables decentralized health monitoring while ensuring data privacy. This study explored the use of autoencoder models and tested various strategies to improve the accuracy of anomaly detection, alongside adaptations to prepare the application for real-world use. The development of AIMHealth represents both an advancement and a challenge in digital health, offering a promising approach for remote patient monitoring and the identification of relevant health patterns.

Keywords: Mobile Health, Federated Learning, Anomaly Detection, Covid-19, Remote Patient Monitoring, Artificial Intelligence

Contents

List of Figures

List of Tables

Acronyms

AI: Artificial Intelligence
COVID-19: virus SARS-COV 2
ECG: Electrocardiograms
FedAdam: Adaptive Federated Optimization using Adam
FedAvg: Federated Averaging strategy
FL: Federated Learning
GDPR: General Data Protection Regulation
IoMT: Internet of Medical Things
IoT: Internet of Things
LSTM: Long Short-Term Memory
MAE: Mean Absolute Error
mHealth: mobile health
ML: Machine Learning
MSE: Mean Squared Error
PRISMA: Preferred Reporting Items for Systematic Reviews and Meta-Analyses
RHR: Resting Heart Rate
RQ: Research Question
STD: Standard Deviation

CHAPTER 1

Introduction

1.1. Context

The global pandemic triggered in 2019 by the virus SARS-COV 2 (COVID-19) and declared by the World Health Organization on March 11, 2020 [**1**], has exposed inherent weaknesses in health systems worldwide. From the lack of infrastructure to respond to the increased need for care, to the scarcity of resources and inadequacy in responding to emergencies [**2**]. Faced with the challenges of the pandemic, such as social distancing and movement restrictions, it was important to optimize medical care by expanding digital medicine, telemedicine and mobile health (mHealth) applications, ensuring safe and efficient healthcare delivery without exposing individuals to contagions [**3**].

The crisis has highlighted the immediate impacts on managing infectious diseases and the need for more effective approaches to the prevention and management of chronic conditions [**4**]. As the population ages, complex challenges arise, from the increase in these chronic conditions to the growing pressure on health systems [**5**].

In this scenario, digital solutions can facilitate and optimize the response of individuals and health systems, enabling remote diagnosis and monitoring. mHealth has emerged as an accessible and effective solution for enhancing public health [**6**]. mHealth applications have been evolving, providing health monitoring, promotion, awareness and supporting decision-making [**7**].

As Internet of Things (IoT) technology advances, smartphones and wearables expand the potential of digital health, making it possible to collect and continuously track vital indicators such as respiration rate, heart rate, body temperature and blood oxygen saturation. This technology provides real-time information about the user's health because these are the first signs that show the user may be ill [**8**]. Through remote monitoring with mHealth applications, it is possible to promote health in a proactive, preventive and personalized way, identifying potential problems early on and enabling more effective interventions, reducing the need for urgent care and consequently reducing the number of hospital admissions [**9**].

Advancements in Artificial Intelligence (AI) and Machine Learning (ML) facilitate faster and more accurate diagnoses while enhancing the ability to predict diseases before the appearance of symptoms. However, there is an urgent need to adopt distributed AI approaches to enable scalable and privacy preserving intelligent healthcare applications at the network's edge [**10**].

In this scenario, implementing Federated Learning (FL) in mHealth applications guarantees greater privacy of user data, promoting a personalized approach to healthcare [**10**].

FL is a decentralized machine learning approach in which models are trained locally on user devices and only model updates, not raw data, are shared with a central server [**10**]. This method ensures that sensitive health information remains on the user's device, improving data privacy and security. This approach not only enhances crisis response capabilities but also establishes a framework for a health system that is centered around the patient, adaptable to individual needs and focused on prevention. As healthcare management advances, it is crucial to investigate innovative solutions like FL to safeguard the privacy and security of sensitive health information.

This dissertation aims to contribute to developing a mHealth application, AIMHealth, enhanced by FL. The AIMHealth application is part of the AIMHealth Secure Platform to be developed as part of the AIMHealth - Mobile Applications Based on AI for Public Health Response project, funded by the Foundation for Science and Technology (FCT) [**11**], coordinated by the Center for Research in Science, Technology and Architecture (ISTAR-IUL), under grant number DSAIPA/AI/0122/2020.

1.2. Motivation

The motivation behind this research is the growing need for efficient and affordable solutions for remote disease monitoring [**12**]. The proposed application, based on mHealth and powered by FL, delivers real-time anomaly detection and personalized patient monitoring, all while ensuring privacy and security.

By employing advanced ML algorithms, mHealth must be able to analyze health data on an ongoing basis, identifying patterns that deviate from the norm and warning of possible deviations. This personalized approach, combined with the privacy protected by FL, not only anticipates specific patient needs but also promotes a rapid and preventive response to changes in health indicators.

The focus on FL in the application's proposal ensures the privacy of user data and establishes a relationship of trust between technology and healthcare [**13**] ensuring data confidentiality. The app paves the way for a safer and more collaborative partnership between patients and healthcare professionals, promoting a patient-centred approach.

The collaborative framework of AIMHealth, involving institutions like ISTAR-IUL, CIS-IUL and others brings together expertise from various fields to create a robust response system for public health emergencies. This dissertation aims to contribute to the AIMHealth project, which is being spearheaded by the Centro de Investigação em Ciências da Informação, Tecnologias e Arquitetura (ISTAR-IUL) in collaboration with the Centro de Investigação e de Intervenção Social (CIS-IUL), both part of Iscte – Instituto Universitário de Lisboa. Additionally, the project includes contributions from the Instituto de Telecomunicações, the Associação para Investigação e Desenvolvimento da Faculdade de Medicina da Universidade de Lisboa (AIDFM) and the Centro Cardiovascular da Universidade de Lisboa (CCUL).

The AIMHealth project amplifies the capabilities of mHealth applications by providing a sophisticated, secure and user-centred platform demonstrating how technology can be

utilized to improve public health outcomes, in a world where the frequency and impact of health emergencies are increasing [**14**]. The integration of AI and blockchain technologies within this project sets a new standard for privacy and security in health applications, enabling more proactive and predictive health management strategies.

1.3. Objectives and Research questions

The main challenge resides in the limited availability of advanced mHealth applications that are proficient in systematically collecting health data, facilitating continuous monitoring, providing accurate diagnoses and delivering personalized health services, all that while ensuring the privacy and security of user data. In the current scenario, numerous mHealth solutions struggle to effectively integrate these elements, leading to significant gaps in healthcare monitoring and delivery.

FL presents a compelling solution to this challenge by allowing AI models to be trained directly on users' devices. This approach optimizes data collection efficiency while prioritizing privacy and confidentiality concerns.

Considering the challenges presented in the research, this dissertation aims to address the issues raised by the design and implementation of an artificial intelligence-based mHealth application - the AIMHealth app. This application intends to validate the integration of FL for the identification of anomalies in the user's health through continuous and personalized monitoring.

The main objectives of this dissertation are as follows:

- To design and implement an AI-driven mHealth application capable of integrating Federated Learning for real-time health data processing while enhancing privacy and security.
- To evaluate the ability of Federated Learning to identify patterns and anomalies specifically associated with COVID-19 in users' physiological data.

To achieve these objectives and guide the development of this dissertation, the following research questions were defined:

RQ1: How can we optimally prepare the AIMHealth application to use Federated Learning automatically?

RQ2: Can we identify patterns or anomalies in the user's physiological data that allow the detection of COVID-19 using Federated Learning within the context of the AIMHealth app?

1.4. Document Structure

This thesis is structured into five chapters, summarized as follows:

(1) **Introduction** – This chapter introduces the context, motivation, objectives and research questions of the dissertation.

(2) **Literature Review** – In this chapter, a comprehensive and systematic review of the literature is presented, following the PRISMA methodology. This review provides the theoretical foundation for the research.
(3) **Development and Optimization of the AIMHealth Application** – This chapter addresses the first Research Question (RQ) by discussing the initial state of the AIMHealth application and detailing the modifications made to prepare the application to use FL automatically.
(4) **Federated Learning an exploration** – This chapter responds to the RQ2, focusing on the implementation of FL techniques to identify patterns or anomalies in the user's physiological data that allow the detection of COVID-19
(5) **Conclusions** – The final chapter presents the conclusions drawn from the research, discusses the limitations encountered and suggests directions for future work.

CHAPTER 2

Literature Review

This section will explore FL techniques and data processing in mHealth, exploring the application of ML methods and real-time signal processing for analyzing medical data. Techniques aimed at the early detection of anomalous patterns, such as changes in heart rhythms, which can indicate potential health problems in users in mHealth environments, will be discussed.

Additionally, will be explained the approach used to search for related work and get into the requisite background knowledge essential for a comprehensive understanding of the dissertation. Towards the conclusion of this section, I will provide a concise summary, highlighting the identified literature gaps and emphasizing how this dissertation contributes to advancing the research field in mHealth, FL and anomaly detection.

2.1. Systematic Review

Following the Preferred Reporting Items for Systematic Reviews and Meta-Analyses (PRISMA) methodology [1], the search for papers related to the dissertation had to be carried out in several repositories. The repositories used were the IEEE Xplore Digital Library[2], Web of Science database[3] and Scopus database[4].

The searches were carried out using a specific keyword query, which was applied to the titles and abstracts of articles within the repositories. This search was carried out in October 2023 and refined to return only articles written in English and published between 2017 and 2023, either freely accessible for Portuguese Academia.

Google Scholar was also used as an auxiliary tool when necessary to find works that complemented those found in the other repositories.

The search query used in the articles repositories is as follows:

("Health" AND "Federated Learning" AND ("Detection" OR "phone"))

A total of 326 articles were obtained from the respective repositories. After eliminating duplicates, a total of 216 articles were maintained. The next step was to screen the title, abstracts, introduction and keywords to select the articles for further review. In some cases, when the information was not considered sufficient to make a decision, the conclusion or the complete document was read.

[1]PRISMA: `http://www.prisma-statement.org/`
[2]IEEE Xplore: `https://ieeexplore.ieee.org/Xplore/home.jsp`
[3]Web of Science: `https://www.webofscience.com/wos/woscc/basic-search`
[4]Scopus: `https://www.scopus.com/home.uri?zone=header&origin=`

Following the PRISMA methodology, 31 documents were collected after the removal of duplicates and screening titles, abstracts, introductions and keywords. The excluded documents were not aligned with the context of the study.

2.2. Related Work

In recent years, there has been a significant increase in the number of articles exploring how FL works in various areas of healthcare, with the 216 articles obtained clearly illustrating this trend, as shown in Figure 2.1. Among them, article [**10**] stands out for offering a detailed survey on FL in healthcare which is relevant for understanding its principles, limitations, advantages, implementation requirements in real-world scenarios and respective applications.

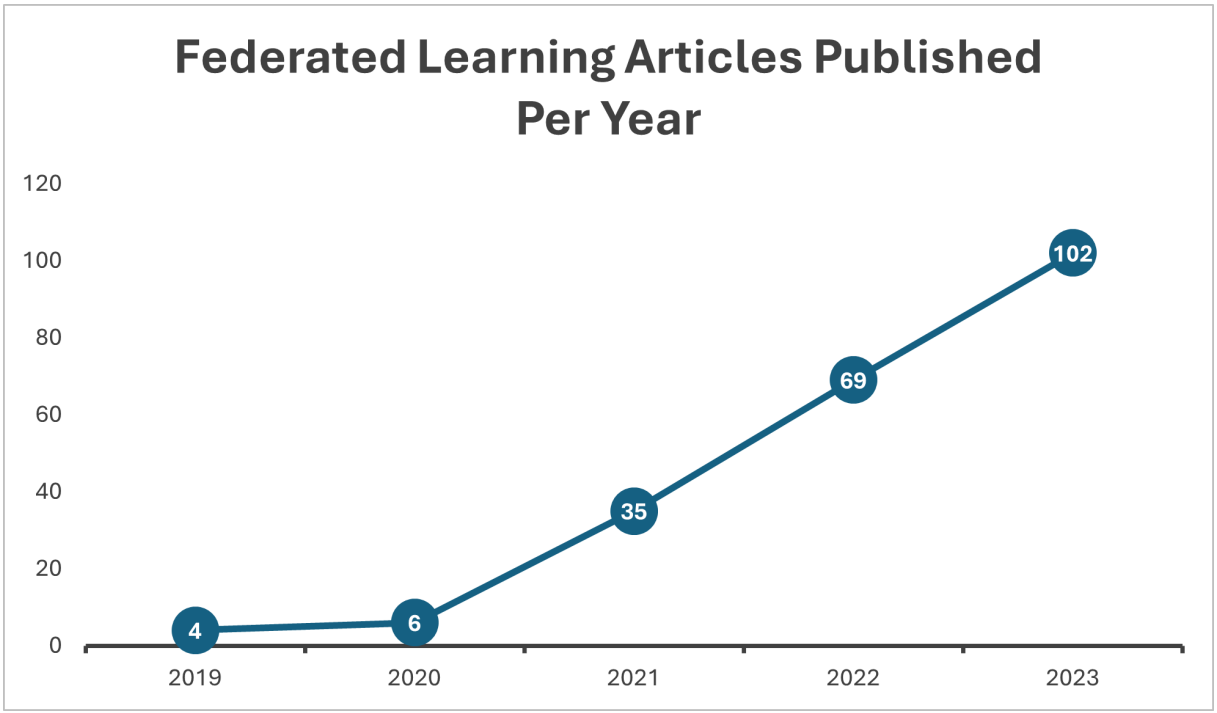


FIGURE 2.1. Number of Federated Learning Articles Published Per Year

2.2.1. Federated Learning in Mental Health

The literature review highlights nine articles that focus on the application of FL in mental health contexts. These studies provide valuable insights into how FL can be used in the detection and monitoring of conditions such as depression, stress and loneliness. The authors of [**15**] address the application of a FL framework in Mental Health Monitoring Systems to safeguard user data privacy, reduce network usage and improve performance through mobile applications.

Other approaches include the model proposed by the authors of [**16**] which monitors depression in individuals by analyzing data collected from their keyboards while writing on social networks. The authors of [**17**] also proposed a mHealth application to detect depression using a smartphone, mentioning the stages and limitations of their solution.

The authors of [**18**] propose a solution using multi-source mobile data, comparing the results of using independent and identically distributed data and non-independent and non-identically distributed data to detect depression.

Stress detection using wearable devices has been explored in several recent studies, as highlighted in [**19**] which presents FL as a solution to privacy challenges in collecting and training robust models, although it guarantees greater privacy, it got inferior results to individual and centralized learning. Similarly, the authors of [**20**] used wearable devices to monitor stress through heart data and achieved promising results in terms of privacy and performance. The authors of [**21**] obtained positive accuracies when using classifications of Electrodermal activity to detect stress, guaranteeing the privacy of the patient's data.

Emotion detection has also been a subject of research, with the authors of [**22**] presenting a method that uses FL for real-time classification of emotional state from multimodal data streams collected by wearable devices. Authors in [**23**] developed a model for detecting loneliness using smartphone sensors, which despite improving privacy and showing promising performance, concerning centralized learning has lower performance which may be due to data heterogeneity.

2.2.2. Federated Learning in Cardiovascular and Neurological Health

An analysis of studies focusing on the use of FL in cardiovascular and neurological health reveals significant advances. These articles explore how FL can be applied to the detection and monitoring of cardiac conditions, as well as in neurological contexts, offering a promising vision for improving diagnoses and treatments.

The authors in [**24**] focus on the detection of cardiac arrhythmias using Electrocardiograms (ECG) on ultra-edge nodes using an Async-FL model, improving accuracy and reducing network overhead compared to synchronous FL. In turn, the authors of [**25**] and [**26**] detect anomalies in ECG, using Autoencoders and Explainable AI to explain why the algorithm has detected a particular anomaly. The authors in [**27**] propose using FL to train distributed ML models on local devices, aiming to detect heart disease in real-time through multi-label classification, to continuously monitor patients' vital signs and predict possible adverse events.

In neurological contexts, the authors of [**28**] propose an FL framework for detecting epileptic seizures on mobile devices using Deep Neural Networks to extract ECG features. Additionally, researchers in [**29**] explore how FL can be utilized to monitor "Freezing of Gait" in real-time for patients with Parkinson's disease.

2.2.3. Federated Learning in COVID-19

The literature has explored various strategies to address the COVID-19 pandemic through FL, resulting in a wide variety of articles dedicated to this topic. For instance, the article [**30**] provides a comprehensive review of FL approaches to COVID-19 detection, highlighting the importance of preserving the security and privacy of patient data.

The authors of the article [**31**] proposed FedLoss for detecting COVID-19 on mobile devices. This approach effectively addresses class imbalance challenges by integrating predictive losses, demonstrating increased efficiency, particularly in scenarios with limited COVID-19 cases. In their study, factors such as breathing, coughing and voice were taken into account during the disease identification process. In article [**32**], the authors proposed the Genetic Clustered Federated Learning algorithm, which utilizes genetic algorithms for COVID-19 detection. This algorithm aims to optimize hyperparameters in a personalized way for different clusters of devices, using a federated approach.

In response to the pandemic's mask-wearing mandates, a facial mask detection system was developed as described in article [**33**]. This system utilizes a FL approach with asynchronous weight updates, to improve data privacy and reduce the centralized computational load, achieving competitive results in terms of performance and communication efficiency.

2.2.4. Federated Learning in Security and Privacy

The implementation of secure and intelligent healthcare systems has been explored in several articles. Article [**34**] discusses various privacy-preserving techniques for these FL-based systems, including differential privacy, incentive mechanisms, blockchain and the use of digital twins. The authors in [**35**] present the implementation and evaluation of FL and Differential Privacy techniques in mHealth systems, which when simulating an external attack, the results indicated that the combination of these two significantly reduces the attacker's ability to infer users' private information.

Articles [**36**] and [**37**] adopt blockchain technology and FL to improve the accuracy of disease prediction and intrusion detection in an Internet of Medical Things (IoMT) context.

Anomaly detection models for remote patient monitoring have been proposed by [**38**], which use FL, Long Memory Recurrent Neural Networks and digital twins to improve efficiency in anomaly detection. Additionally, the article [**39**] proposes an anomaly detection system for the diabetes management system using FL to preserve patient privacy, with experimental results showing effective detection of malicious events with low latency.

2.2.5. Other Federated Learning approaches

In addition to the approaches discussed above, there are other applications of FL, such as the study referenced as [**40**] in the field of Human Activity Recognition. This research emphasized aspects such as the strategic placement of sensors, improving the accuracy of FL models and optimizing the use of bandwidth. Furthermore, in the field of fetal health monitoring, the concept of Federated Data Quality was introduced in [**41**], with a focus on ensuring data integrity in distributed monitoring environments.

The authors in [**42**] address device selection in FL systems with the dual objective of maximizing the accuracy of the global model and minimizing energy consumption in edge devices. The study introduces FedSens, a framework that incorporates a deep

learning model based on extrinsic-intrinsic reinforcement. Additionally, the article [**43**] details AsyncFedKD, a pioneering approach that integrates knowledge distillation with asynchronous training, aimed at effective anomaly detection in FL systems.

Finally, article [**44**] investigates user assignment and resource management strategies in Hierarchical FL in distributed environments. This study presents a thorough comparison of the performance of the proposed scheme, called the Efficient Assignment and Resource Allocation Algorithm, highlighting its efficiencies in minimizing communication overhead and improving model accuracy in a variety of scenarios and data sets.

2.2.6. Summary

This review has provided valuable insights into the application of FL and anomaly detection within healthcare. Autoencoder models have proven to be highly effective in capturing temporal patterns and optimizing training efficiency for anomaly detection. Several strategies for FL were explored, establishing them as viable options for mHealth applications. The data to be utilized for the detection of COVID-19 will be heart rate, as supported by the findings in article [**45**], which confirms that heart rate can be effectively used to detect anomalies associated with COVID-19. The study demonstrated that 63% of COVID-19 cases could have been identified in real time before symptom onset through a system based on significant elevations in Resting Heart Rate (RHR) relative to individual baseline levels. These insights will guide the methodological approaches in Section 4.

Although several mobile healthcare applications employ FL to detect abnormal health conditions, there is a lack of specific solutions for remote monitoring of patients with multiple health conditions in real-time. This gap highlights the need for a mobile application, based on AI and FL, that not only efficiently manages the complexity of multiple health conditions but also ensures the privacy and security of users' data.

CHAPTER 3

Development and Optimization of the AIMHealth Application

3.1. Introduction

This chapter addresses the RQ1. It provides a comprehensive overview of the AIMHealth application's current state, highlights the contributions made to enhance the user experience and discusses the necessary changes implemented to achieve the objectives.

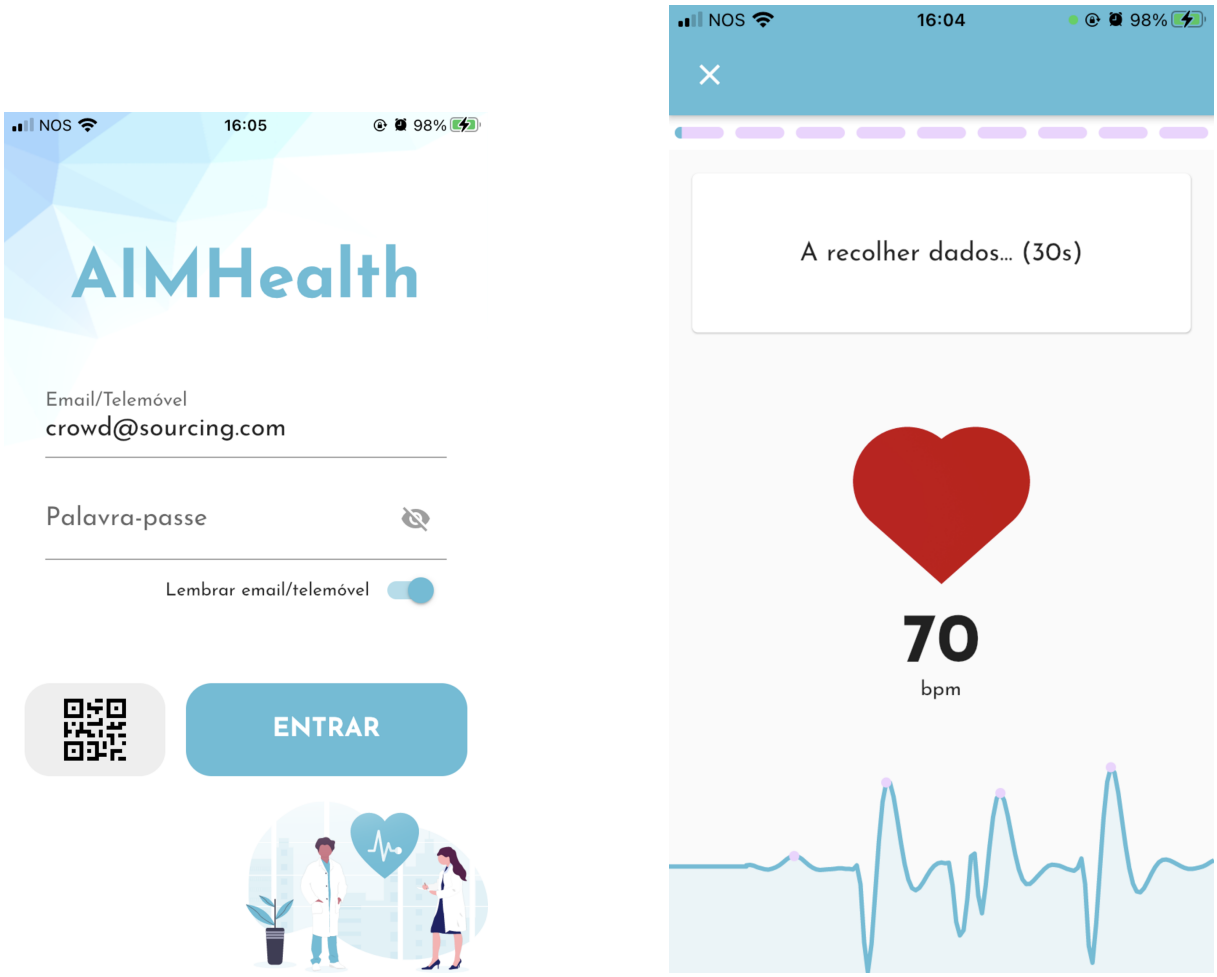
3.2. Current Application Overview

The AIMHealth application was developed as a mobile solution for health monitoring on Android, designed to collect and manage physiological data from users. Built using the Flutter framework [**46**], the app offers a user-friendly interface for inputting and monitoring health parameters. As shown in Figure 3.1, the screenshots provide a visual representation of the application's login page and the heart rate data collection process.

The AIMHealth app was designed to enable users to input both cardiac and respiratory data, including voice, cough, heart rate and blood oxygen saturation, developed to support COVID-19 patients. The app incorporated two key protocols for cardiac assessment: one using camera-based photoplethysmography to measure heart rate and blood oxygen levels over 30 seconds and another relying on data collected from a device for transmission of similar physiological data. Additionally, two respiratory protocols were included, one capturing a short audio clip of the patient's cough and the other recording the patient's voice while saying '33', both aimed at evaluating respiratory function. All collected data was securely stored to ensure patient privacy and data protection.

A significant contribution to the improvement of the AIMHealth application was made by [**47**]. Focused on enhancing the user experience through a user-centered design approach, conducting in-depth user research, including interviews with potential users, to understand their needs and challenges. Based on this, were developed user personas and created both low and high-fidelity prototypes, resulting in a more accessible and intuitive interface, particularly for older adults with chronic health conditions. Additionally, introduced features like medication management and appointment scheduling, making the app more functional and user-friendly.

As previously mentioned, the protocols within the application were all manual, requiring users to input data themselves. This manual approach proved insufficient in achieving one of the project's primary objectives: the automatic identification of diseases, such as COVID-19, through anomaly detection. While initially useful, manual data collection did not meet the requirements for continuous and automated monitoring, which is essential for the accurate and timely detection of real-time anomalies. The reliance on users to



(A) Login page (B) Collecting heart rate data page

FIGURE 3.1. Screenshots of the AIMHealth app

manually input data compromised both the consistency and frequency of data collection, placing an additional burden on the patient. This could lead to missing data and, ultimately, reduce the effectiveness and reliability of the system. Given these limitations, it became necessary to enhance the application to enable automated monitoring without the need for constant user intervention.

3.3. Enhancements for Automation

To achieve the goal of continuous monitoring, several changes needed to be implemented in the AIMHealth application. The app required integration with wearable devices to collect real-time data and the ability to run in the background to ensure continuous operation and storage for the data collected from these devices. These modifications are detailed below:

3.3.1. Integration with Wearable Devices

To enable the application to retrieve health data in real-time, a device capable of continuously monitoring health signals was needed. Wearable devices, such as smartwatches, are ideal for this purpose because they are equipped with integrated sensors and allow

for an automated collection of data like: heart rate and steps, without requiring manual input from the user.

To integrate the wearable devices with the AIMHealth app, the Health library[1] was selected, as it simplifies the connection with Google Fit[2], which was chosen for its platform that supports a wide range of wearables and health tracking devices. Google Fit provides a unified interface that simplifies the integration of diverse health data. This integration allows the application to continuously and seamlessly retrieve health data from wearable devices in the background. To ensure accurate data collection, communication between the AIMHealth app and Google Fit was configured to automatically and precisely transfer health data to the app without requiring user intervention.

To use this feature, users must grant permission for the app to access their Google Fit data. Additionally, Google Fit itself requires permission from the phone's health application to access specific health metrics, ensuring a secure flow of data. Once these permissions are granted, health data is automatically and accurately transferred from the wearable devices to the application, without requiring any manual input from the user.

3.3.2. Background Monitoring

To implement background data collection, the Flutter Workmanager library[3] was used to ensure that the application continued to monitor the user's health even when it was not actively in use. This library allows for the execution of periodic and continuous tasks, ensuring that monitoring remains uninterrupted, even when the application is running in the background or the mobile device is in a resting state. Additionally, WorkManager is responsible for collecting the most recent health data from wearable devices, ensuring that the application always has up-to-date information for analysis.

However, WorkManager has a limitation where the minimum interval for background requests is set at 15 minutes. While there are potential solutions to overcome this limitation, it is not necessary at this stage because all available data within this 15-minute interval will be collected and transmitted, ensuring that no data is lost. Given this constraint, the 15-minute interval was chosen as the smallest available option, which aligns reasonably well with the dataset's one-hour intervals used in the following section 4.3. Should future implementations require more frequent data collection, this setting will need to be revised to ensure the system continues to capture data effectively.

3.3.3. Storage of Health Data

The security and privacy of users' health data were considered during the development of new features, although it is important to note that this implementation was part of a limited proof-of-concept. Basic symmetric encryption techniques were implemented following the retrieval of data from Google Fit. However, given the proof-of-concept nature of the project, these security measures were not intended for long-term use in a

[1]Health library: `https://pub.dev/packages/health`
[2]Google Fit: `https://www.google.com/fit/`
[3]Flutter Workmanager library: `https://pub.dev/packages/workmanager`

production environment and should be regarded as preliminary steps toward more robust data protection.

For managing user authentication and storing data securely, Appwrite[4] was employed. This platform facilitated basic encryption and storage management, ensuring a certain level of privacy and data integrity. However, the data collected during this proof-of-concept was not stored permanently and will not be used beyond the scope of the project. Future iterations of the application will require more advanced encryption standards, full compliance with General Data Protection Regulation (GDPR) requirements, and more rigorous privacy protections to ensure that health data is adequately safeguarded in a real-world environment.

3.4. Impact of Implemented Changes

The implemented changes made the AIMHealth application transition from a manual data entry system to an automated real-time health monitoring app. The integration of wearable devices and incorporating libraries such as Health Connect and WorkManager have enabled continuous data collection without the need for active user involvement. This transformation has reduced the burden on users, as they no longer need to manually input health data, which not only improves the user quality of experience but also ensures a more consistent and accurate data flow for analysis.

These enhancements establish a fundamental basis for the integration of FL. By automating the collection of physiological data from wearable devices, AIMHealth can continuously gather and process large volumes of decentralized health data. This data can be used to train distributed ML models without needing to share sensitive information, making the application well-prepared to use FL for real-time anomaly detection and health condition monitoring.

[4]Appwrite: `https://appwrite.io`

CHAPTER 4

Federated Learning an exploration

4.1. Introduction

To address the initial RQ2, the original plan was to conduct tests with real participants using the AIMHealth app. This approach would allow for real-world insights into the application's functionality by capturing live user data and directly implementing the FL model within its environment. It would allow for in-depth observation of how users interact with the system, monitoring their physiological data in real time and uncovering potential challenges in data collection and processing that may only happen in a live scenario. However, achieving this presented three major issues. First, it proved difficult to recruit a sufficient number of participants who owned a smartwatch to collect the necessary data. Second, even among those who could participate, there was no warranty that they would contractCOVID-19 or another illness during the testing period, which posed a significant issue for anomaly detection tests. Third, the application is not yet fully compliant with security standards, thus it does not ensure the needed safeguards to comply with the GDPR. Because of the last two issues that stand out as severe limitations, it was decided to perform a FL simulation instead. The Stanford dataset was used as a practical solution to address the test scenario to respond to RQ2 and progress without the constraints of participant recruitment, the timing of COVID-19 infections and the incomplete security infrastructure of the application.

This chapter will outline the steps involved in implementing and testing the FL models within the AIMHealth app to address RQ2. The chapter is organized as illustrated in the flowchart 4.1, where

- **Dataset:** Describes the dataset used for the experiments.
- **Data Preprocessing:** Outline the steps for preparing the dataset: extract resting heart rates, apply data smoothing techniques, assign labels, normalize the data and develop time series sequences.
- **Models:** Introduces the Long Short-Term Memory (LSTM) and Convolutional Autoencoder models used for anomaly detection.
- **Methodology:** Explores the FL approach, covering anomaly detection, threshold calculation, the implementation of FL and potential improvements to enhance the model's performance.
- **Results:** Provides a comprehensive analysis and discussion of the results, including evaluation of loss functions, performance of autoencoder models, anomaly detection, threshold methods and challenges like overfitting, subset to train data and clustering.

Additionally, it compares these findings with those of centralized and individualized learning models and seeks to identify patterns in anomaly detection.

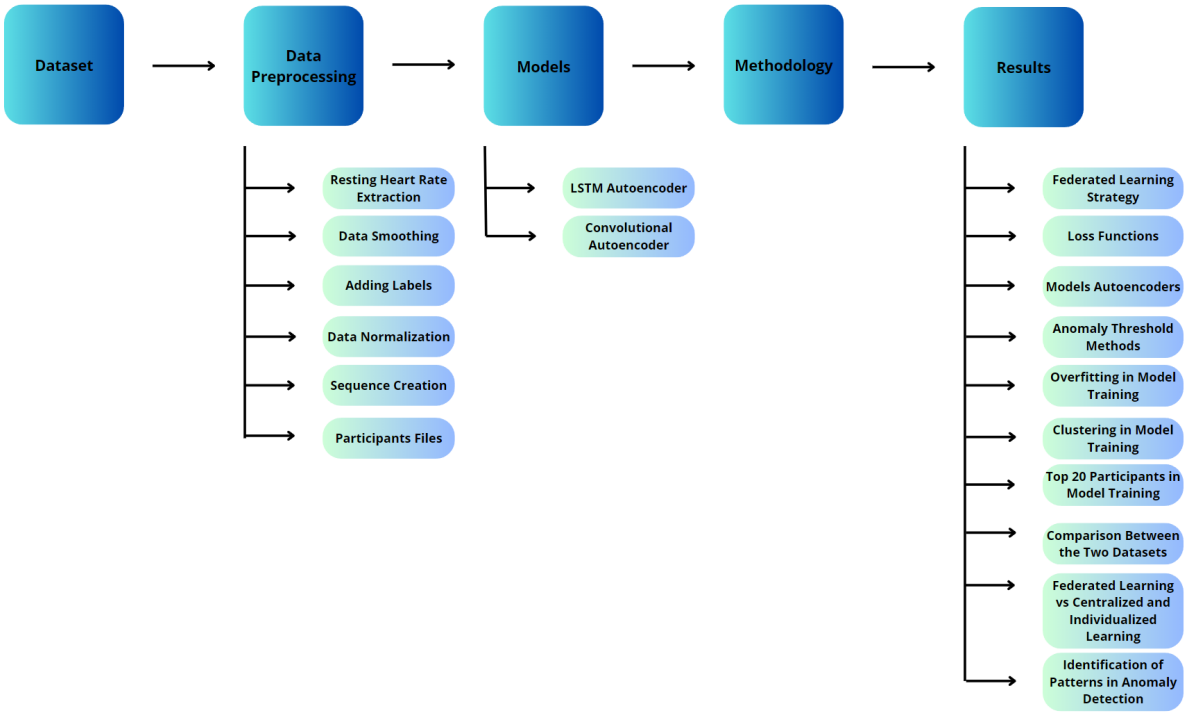


FIGURE 4.1. Flowchart illustrating the steps to address RQ2.

4.2. Dataset

The dataset chosen for this study is sourced from the Stanford COVID-19 Wearable Study - Phase 2 [**48**], from now on referred to as the Stanford dataset. The dataset has data collected from various wearable devices, such as FitBit and Apple Watch. It offers a range of metrics, including heart rate, steps taken per minute, symptom onset dates, COVID-19 test dates and other relevant health information. These metrics provide sufficient data for the main objective of this study, which is to identify anomalies that could suggest the presence of COVID-19 through variations in resting heart rate.

In total, the Stanford dataset comprises 2124 data points, including data from healthy participants who have not contracted any diseases, as well as 84 individuals diagnosed with COVID-19. Although the dataset contains a broad spectrum of information, this study will focus specifically on three datasets for each participant.

The first dataset maps heart rate measurements to specific dates to track the participants' heart rate trends over time. The second dataset pairs the dates with the number of steps to calculate the RHR. The third dataset provides information on symptom onset dates along with COVID-19 test results, which can help identify potential health anomalies.

A more detailed analysis of the dataset reveals that data from FitBit devices contain significantly more daily heart rate measurements, with an average of 9950 measurements

per day, compared to non-FitBit devices, which have an average of 380 measurements per day. This discrepancy is evident from the differences in measurement density, as illustrated in 4.2, which shows that a participant with a non-FitBit device has a lower measurement density compared to 4.3, a participant using a FitBit device.

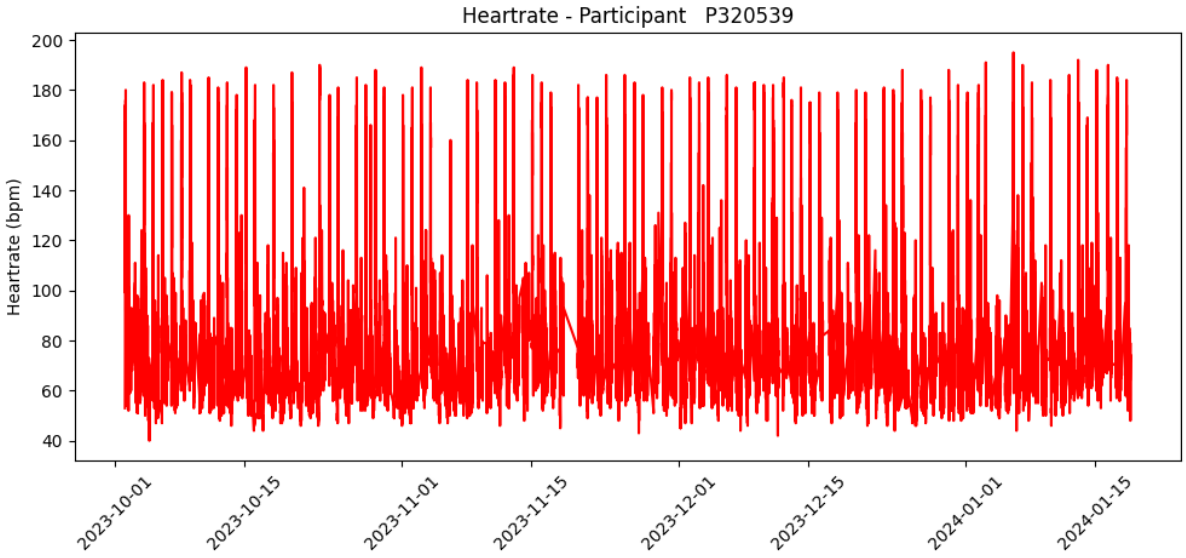


FIGURE 4.2. Heart rate data of Participant P320539 was recorded using a non-FitBit device.

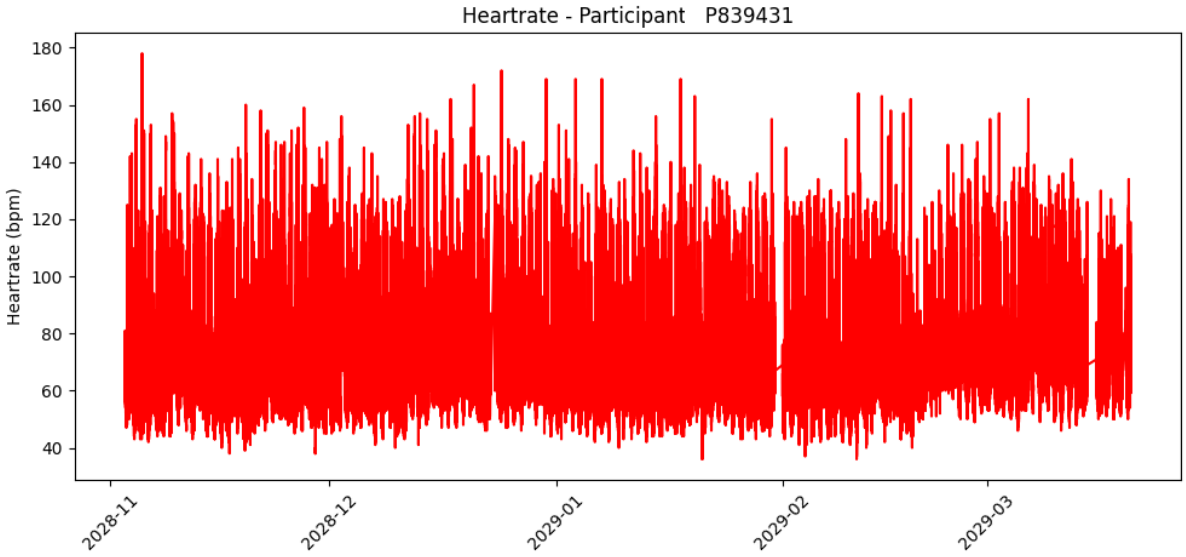


FIGURE 4.3. Heart Rate of Participant P839431 was recorded using a fitbit device.

The distribution of days with and without measurements for each participant is shown in 4.4. This figure illustrates the presence of gaps in the data, where several days lack measurements. Together, these subsets (days with and without measurements) define the

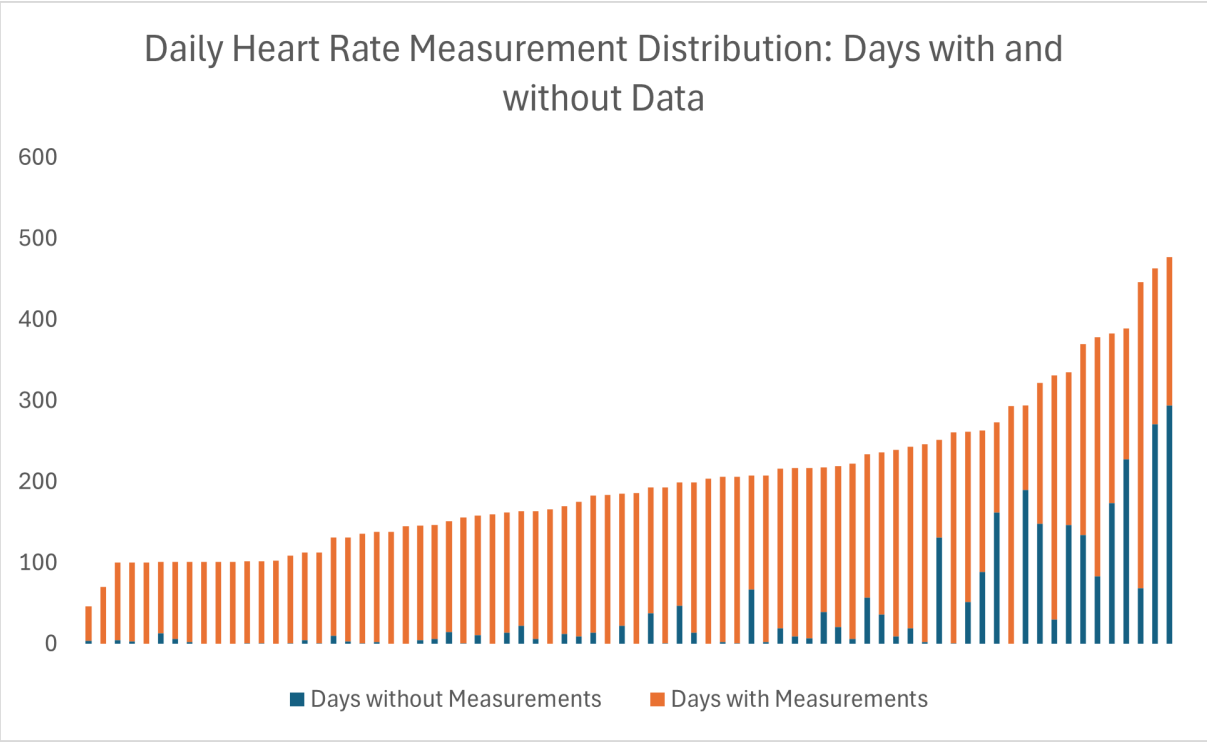


FIGURE 4.4. Distribution of days with and without heart rate measurements

full measurement period for each participant, which spans an average of 164 days, less than six months. These gaps are particularly evident in cases like that of Participant P885171, who shows periods of sparse measurements, as illustrated in 4.5.

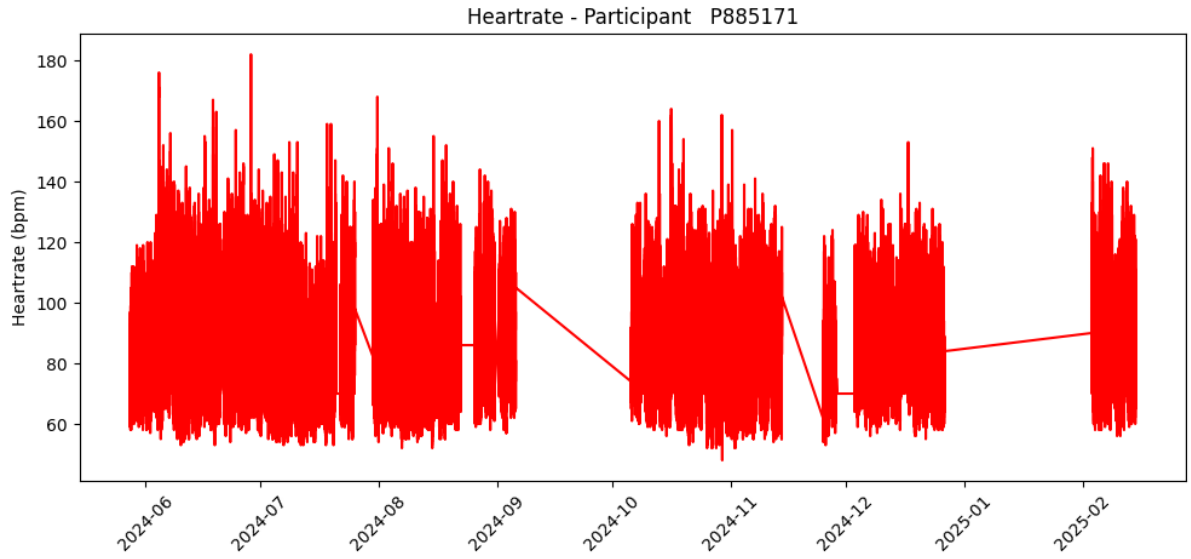


FIGURE 4.5. Heart Rate of Participant P885171.

Another conducted analysis focused on the number of measurements per day. It was observed that there were several days where the number of measurements fell below 50% of the participant's daily average. In some cases, the measurements decreased to less

than 15% of the typical daily values for certain participants. This indicates significant inconsistencies in data collection on those days.

To ensure rigorous data anonymization, participants have been assigned unique numeric identifiers and all the dates in the dataset are fictitious. This approach guarantees that the data cannot be traced back to individual participants.

4.3. Data preprocessing

To ensure the data's quality and suitability for ML applications a comprehensive preprocessing process was applied. In this section, two different datasets from the Stanford COVID-19 Wearable Study - Phase 2 will be created.

The two datasets consist of time series data of RHR, which will be analyzed to identify anomalies and patterns associated with COVID-19. The difference between the datasets is that the first dataset, referred to as the Covid dataset, includes only participants who tested positive for COVID-19, while the second dataset, referred to as the Healthy dataset, includes the same participants as the first dataset, along with an equal number of healthy individuals. These balanced samples allow for a more accurate comparative analysis, providing a closer approximation to a real-world scenario to explore differences in heart rate patterns between infected and non-infected individuals.

The subsequent subsections will detail the specific steps taken during the data processing phase, as seen in Figure 4.6.

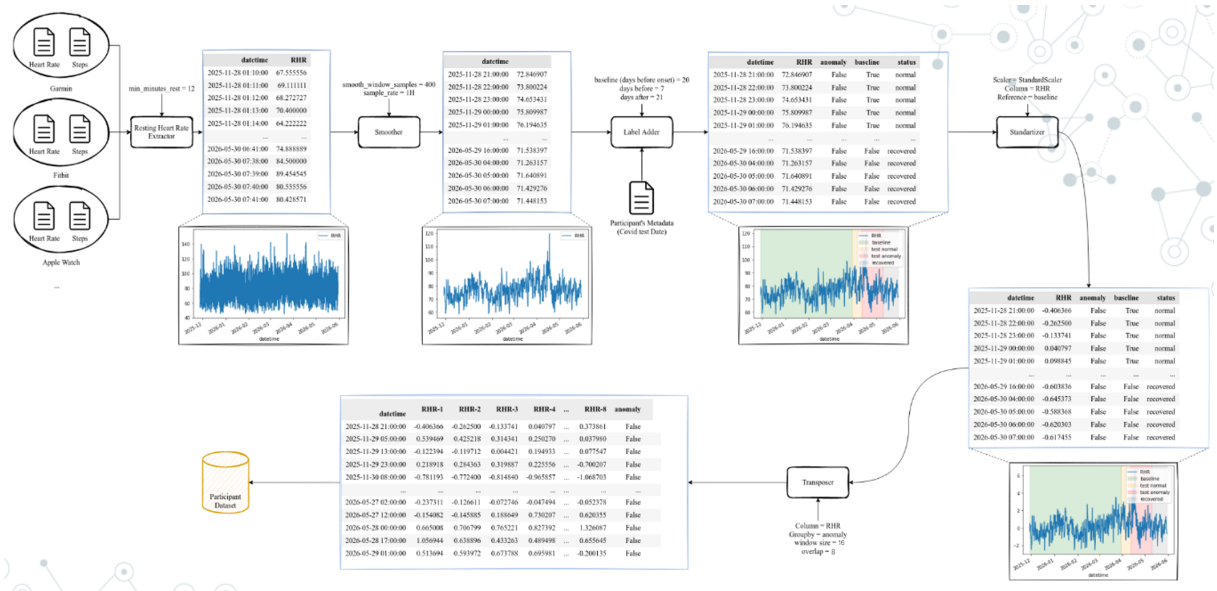


FIGURE 4.6. Data processing for resting heart rate analysis. Adapted from SSLTools[1].

4.3.1. Resting Heart Rate Extraction

In order to effectively identify potential anomalies that may be associated with the presence of COVID-19, the analytical process initiates with a comprehensive examination

[1]SSLTools: `https://otavioon.github.io/ssl_tools/tutorials.html`

of heart rate and step count data collected from participants. This information is systematically organized into two datasets: one dataset records the date along with the corresponding heart rate for each individual, while the other dataset documents the date alongside the step count for each participant.

These two datasets are then combined to create a new dataset that includes the date and the RHR, calculated based on periods when the participant remained inactive. To determine when a participant's heart rate can be classified as resting, the parameter min_minutes_rest is used. This parameter indicates that a participant's heart rate will be classified as resting if they remain inactive, with a step count of zero, for a continuous period of min_minutes_rest minutes.

The min_minutes_rest value has been established at 12 minutes, based on physiological research indicating that a continuous rest duration of at least 10 minutes is generally sufficient for the heart rate to stabilize at a resting state, as demonstrated in study [**49**].

4.3.2. Data Smoothing

After extracting the resting heart rate, a data smoothing process is applied to enhance the quality of the data for further analysis. The main goal of applying smoothing techniques is to reduce noise in the RHR data, as this noise can hide significant patterns and trends that are essential for accurate anomaly detection. Smoothing helps to minimize minor variations caused by external factors or measurement errors that do not reflect meaningful changes in the user's condition. By reducing these unwanted fluctuations, the underlying trends become more evident, ensuring that long-term patterns are more visible.

This process utilizes two parameters: smooth_window_sample, which defines the number of samples used for smoothing the data and sample_rate, which determines the frequency at which the data is resampled. The smoothing process applies a moving average filter to the data using the window size specified by smooth_window_sample. After smoothing, the data is downsampled according to the defined sampling rate, producing a new data frame that includes the date, time, and RHR at the desired intervals. In this study, a smooth_window_sample of 400 was selected and the data was resampled at an interval of one hour, resulting in a sample_rate of 1 hour.

4.3.3. Adding Labels

In this subsection, labels were assigned to the data frame in order to categorize the data according to various periods in relation to the onset of symptoms.

For participants who have contracted COVID-19, the parameter baseline_days defines the number of days before symptom onset that will be considered as the baseline period is 21 days. A new column called "baseline" is added to the data frame, marking entries as True for dates that fall within the designated baseline period and False otherwise. This period reflects the participant's usual state before the onset of any symptoms. If a participant does not have sufficient data before symptom onset to meet the training requirement of at least 20 time series data points, their data is excluded from the analysis.

Additionally, participants without available symptom onset dates were also removed from the study, as it was not possible to establish a reference point for the baseline and anomaly period. These issues affected certain participants, making it impossible to use their data in the training process. Consequently, their data was excluded from the study.

The before_onset and after_onset parameters define the number of days before and after symptom onset, respectively, that will be considered as the anomalous period, the chosen period was 7 days before and 21 days after symptom onset, based on findings from previous research [**45**]. A new column called "anomaly" is added, indicating True if the date falls within this abnormal period to identify possible deviations from the baseline that might indicate the presence of COVID-19 and False otherwise.

In addition to these columns, a new column is included, which provides a descriptive label for the condition of the user on that specific date:

- **normal:** The date falls outside the anomalous period, specifically more than 7 days before symptom onset, representing the participant's healthy condition.
- **before onset:** The date is within the 7 days leading up to symptom onset, considered part of the anomalous period.
- **onset:** The exact day symptoms begin.
- **after onset:** The date falls within the period from symptom onset up to 21 days after, during which the participant is still considered in an anomalous period.
- **recovered:** The date falls after the 21 days following symptom onset, indicating that the participant is expected to have returned to normal health conditions.

For participants who did not contract COVID-19, the labelling process takes on a different approach. In this case, there are no anomalies present, so all data points are classified as normal. The dataset for these individuals is divided, with 70% of the data used as baseline data for training and 30% set for testing purposes.

4.3.4. Data Normalization

Once the labels have been applied, the data is normalized using the Z-normalization technique for standardizing the heart rate data, facilitating meaningful comparisons across various periods and participants. The formula for Z-normalization is as follows:

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

Where x represents the individual heart rate measurement, μ symbolizes the mean of the heart rate values during the baseline period and σ represents the standard deviation of the heart rate measurements during the same period. It is important to emphasize that both the mean and standard deviation are calculated solely based on the data from the baseline period. These baseline values are then applied to normalize the entire dataset, including periods before and after symptom onset.

The purpose of applying Z-normalization is to standardize the data, allowing the heart rate values to be compared on the same scale. By normalizing the data relative to the

baseline period, the model can more effectively detect deviations or anomalies that may indicate the presence of COVID-19.

4.3.5. Time Series Sequence Development

The final step in the data preparation process is the development of time series sequences. This process involves reorganizing the data by grouping consecutive rows and transforming them into columns, effectively creating new temporal features. The primary input parameters for this step are window_size and overlap, which determines the length of each sequence and the extent of overlap between consecutive sequences.

For instance, when utilizing a dataset containing 100 samples, with a window_size of 10 and an overlap of 0, the resulting process will generate 10 sequences, each containing 10 samples (10 rows with 10 columns). Within each sequence, the samples are represented as sequentially numbered columns, such as RHR-0, RHR-1, ..., RHR-9. The first sequence comprises the first 10 samples, the second sequence comprises the next 10 samples, and so on.

In this study, a window_size of 16 and an overlap of 8 were chosen. This configuration indicates that each sequence encompasses 16 samples, but subsequent sequences start 8 samples after the previous one, resulting in a 50% overlap. This overlapping strategy enhances the model's ability to capture temporal dependencies in the data.

A consideration in this process is the preservation of label integrity, sequences drawn from anomaly periods are kept distinct from those drawn from non-anomaly periods to ensure that sequences marked as anomalous contain only data points from the anomaly period, preventing any mixing of labels that could confound the model's learning process.

4.3.6. Participants Files

After completing the preprocessing phase, visual representations of participants data will be examined. Each participant has an associated data frame with a unique participant_id to identify each individual.

Figure 4.7 illustrates the resting heart rate of participant 723961, who was diagnosed with COVID-19, along with the respective labels assigned during the preprocessing phase.

The following figure 4.8 shows the data of participant 741238, a healthy individual who did not contract COVID-19, with the corresponding labels assigned during the preprocessing phase.

4.4. Models

This study aims to assess two models for detecting anomalies in time series data: LSTM and Convolutional Autoencoders, to determine the most suitable approach. By comparing these distinct models, the goal is to uncover the most effective method for identifying anomalies in this scenario.

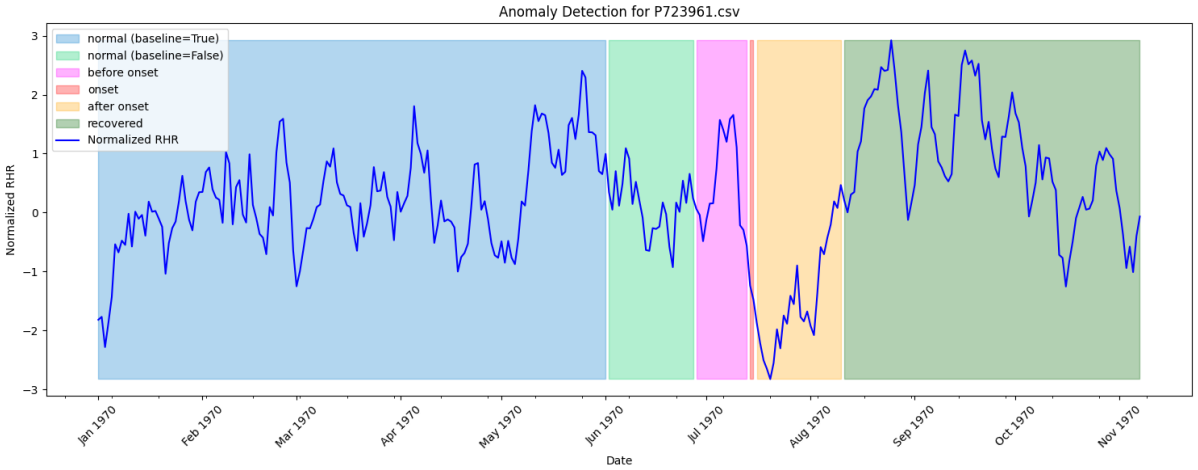


FIGURE 4.7. Resting Heart Rate of Participant 723961 with assigned labels.

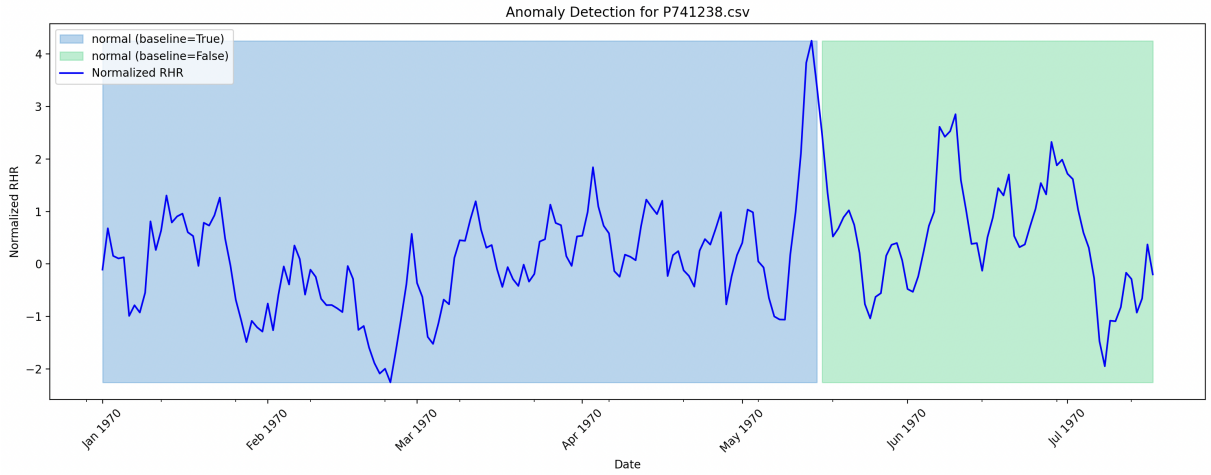


FIGURE 4.8. Resting Heart Rate of Participant 741238 with assigned labels.

4.4.1. LSTM Autoencoder

The motivation behind selecting the LSTM Autoencoder is based on its ability to identify long-term dependencies in time series data for effective anomaly detection [**50**]. The autoencoder learns to reconstruct normal sequences, so any deviation from this pattern results in higher reconstruction errors, indicating potential anomalies. The architecture of the model can be viewed in Annex A.

4.4.2. Convolutional Autoencoder

The Convolutional Autoencoder was chosen due to its enhanced capacity for capturing local temporal dependencies within time series data [**51**]. Convolutional layers excel at capturing features from short-term patterns, significantly improving the ability to detect anomalies within localized sections of a time series. The architecture of the model can be viewed in Annex A.

4.5. Methodology

After acquiring the datasets and establishing the architecture of the models, the next step was the implementation of FL to train the models across distributed data sources. A visual representation of FL is provided in Figure 4.9.

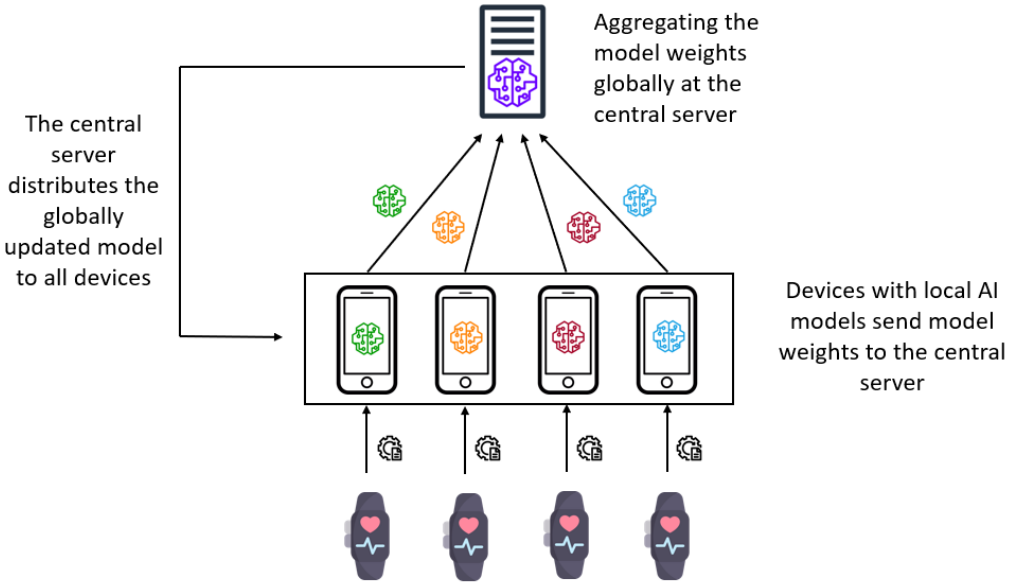


FIGURE 4.9. Diagram of a Federated Learning. Adapted from wikipedia[2].

The goal was to explore various FL strategies to identify the most effective approach for anomaly detection in time series data, specifically focusing on resting heart rate data.

The initial experiments utilized two distinct FL strategies: the Adaptive Federated Optimization using Adam (FedAdam) and the Federated Averaging strategy (FedAvg). The Adam optimizer is an adaptive learning rate optimization algorithm that adjusts the learning rate for each parameter individually. In contrast, FedAvg focuses on aggregating the gradients computed locally by each client through averaging, thereby creating a global model update that reflects the contributions of all clients involved.

Anomaly detection in this context was performed by calculating the reconstruction error using two primary loss functions: Mean Absolute Error (MAE) and Mean Squared Error (MSE). These loss functions measured the difference between the model's output and the actual input data, allowing the identification of anomalies. To determine whether a particular data point was classified as an anomaly, several threshold calculation methods were applied, including standard deviation, percentile based and variance based thresholds, which helped establish boundaries for anomaly classification.

To further improve the model's performance, three additional approaches were implemented:

[2]wikipedia: `https://en.wikipedia.org/wiki/Federated_learning`

(1) **Overfitting Approach:** The model was fine-tuned on each participant's local data to personalize the detection of anomalies, aiming to capture subtle individual patterns that may enhance sensitivity to deviations.

(2) **Clustering Approach:** Participants were grouped based on similarities in heart rate data, allowing the model to specialize within each cluster and potentially improve the accuracy of anomaly detection.

(3) **Top-20 Participants Approach:** The model was trained on the top 20 participants with the best individual performance, aiming to leverage their more reliable patterns to create a generalized model that performs effectively across the entire dataset.

These additional methods were explored as potential avenues to enhance the model's performance, with the goal of evaluating whether they could improve anomaly detection in distributed time series data.

4.5.1. Federated Learning Implementation

The FL process was implemented using the Flower framework[3], which facilitated the simulation of distributed learning across multiple clients. Each client represents a unique subset of the data, mimicking a real-world scenario where data is decentralized. The Flower framework was instrumental in coordinating communication between clients and managing the global model updates, allowing for effective scaling across distributed data sources.

Algorithm 1 illustrates a simplified pseudo-code representation of the overall FL process using the FedAvg strategy[4]

Algorithm 1 Flower Federated Averaging (FedAvg)

Input: Global model M, number of clients n_{clients}, number of rounds R, epochs per client E
Output: Final global model M

Initialize: Server initializes global model M
for each round $r \in [1, R]$ **do**
    Server broadcasts global model M to all clients
    for each client $i \in [1, n_{\text{clients}}]$ **in parallel do**
        Client i receives global model M from server
        Client i trains local model M_i on its own data D_i for E epochs
        Client i computes updated local weights W_i
        Client i sends updated weights W_i to the server
    end for
    Server aggregates the local weights to update the global model:

$$M \leftarrow \frac{1}{n_{\text{clients}}} \sum_{i=1}^{n_{\text{clients}}} W_i$$

end for
Return: Final global model M

[3]Flower framework: `https://flower.ai/`
[4]Flower FedAvg:`https://flower.ai/docs/framework/ref-api/flwr.server.strategy.FedAvg.html`

The FedAdam strategy[5] is employed to enhance the server-side aggregation by using the Adam optimizer, which updates the global model based on the gradients computed by the clients. This approach allows the model to incorporate the client-provided gradients in the optimization process. The corresponding pseudo-code for the FedAdam aggregation function is shown in Algorithm 2.

Algorithm 2 Flower Federated Adam (FedAdam)

Input: Number of clients n_{clients}, gradients G_i, learning rate η, momentum parameters β_1, β_2 and epsilon ϵ

Output: Final global model M

Initialize optimizer parameters: $m_t = 0$, $v_t = 0$
for each round $r \in [1, R]$ **do**
  for each client $i \in [1, n_{\text{clients}}]$ **in parallel do**
    Client i computes local gradients G_i
    Client i sends gradients G_i to the server
  end for
  Server updates the optimizer states:

$$m_t = \beta_1 \cdot m_t + (1 - \beta_1) \cdot \frac{1}{n_{\text{clients}}} \sum_{i=1}^{n_{\text{clients}}} G_i; \quad v_t = \beta_2 \cdot v_t + (1 - \beta_2) \cdot \frac{1}{n_{\text{clients}}} \sum_{i=1}^{n_{\text{clients}}} G_i^2$$

  Compute bias-corrected moment estimates: $\hat{m}_t = \frac{m_t}{1-\beta_1^t}; \quad \hat{v}_t = \frac{v_t}{1-\beta_2^t}$
  Update global model: $M \leftarrow M - \eta \cdot \frac{\hat{m}_t}{\sqrt{\hat{v}_t}+\epsilon}$
end for
Return: Final global model M

In the FL strategies, including FedAdam, the algorithm runs over 10 training rounds. During each round, all available clients participate in the training process, performing local updates on their respective data and sending their model updates to the server. The server aggregates these updates to refine the global model, simulating the FL process orchestrated by the Flower framework. The models used were LSTM and Convolutional Autoencoders, as detailed in Section 4.4.

4.5.2. Anomaly Detection and Threshold Calculation

Anomaly detection conducted in this study was based on the reconstruction error, with two primary loss functions being utilized: MAE and MSE. These loss functions measure the difference between the model's output and the actual input data, allowing for the identification of anomalies. The MAE calculates the mean of the absolute differences between the predicted and actual values, making it more robust to outliers. In contrast, the MSE computes the mean of the squared differences, which penalizes larger errors more significantly, making it more sensitive to significant deviations.

Several threshold calculation methods were tested to determine whether a particular data point was an anomaly. These methods define how the loss distribution is used to set a boundary for classifying anomalies. After calculating the reconstruction error, it is

[5]Flower FedAdam:`https://flower.ai/docs/framework/ref-api/flwr.server.strategy.FedAdam.html`

compared against a predetermined threshold. Should the error exceed this threshold, the data point is deemed anomalous. The threshold approaches implemented are described below:

4.5.2.1. *Standard Deviation Threshold*

This method calculates the threshold as one Standard Deviation (STD) above the mean of the loss distribution. Any reconstruction error exceeding this threshold is considered an anomaly. This approach assumes that most reconstruction errors are normally distributed around the mean and any significant deviation beyond one STD can be classified as anomalous.

4.5.2.2. *Percentile Threshold*

In this method, the threshold is set as a specific percentile of the loss distribution. This approach is useful for controlling the proportion of data points classified as anomalies. The percentile value determines the sensitivity of the anomaly detection higher percentiles result in fewer anomalies being detected, while lower percentiles increase sensitivity. In this study, the 75th percentile was selected to maintain a balance between detecting enough anomalies and reducing false positives.

4.5.2.3. *Variance Threshold*

This approach uses the variance of the loss distribution to determine the threshold. The threshold is set as the mean plus a multiple of the variance, where the factor is set to one by default. A higher factor results in a more permissive threshold, meaning fewer points will be classified as anomalies, while a lower factor makes the threshold more restrictive, identifying more anomalies.

4.5.3. Possible Improvements to Federated Learning

This section explores three techniques that could potentially enhance FL in this context: the use of overfitting, implementing clustering strategies and selecting the top 20 participants to optimize model training.

Overfitting Approach

Overfitting was introduced to explore whether a higher degree of personalization could improve the model's performance in detecting anomalies. After training the global model using federated data, an additional fine-tuning phase was performed for each participant individually. This fine-tuning consisted of training the model on each participant's local data for 10 epochs. The objective was to personalize the model for each participant, capturing unique patterns in their physiological data that may be essential for identifying deviations from their typical behavior.

This approach is based on the hypothesis that minor overfitting to local data could increase the model's sensitivity to individual variations, thereby potentially improving its ability to detect anomalies. This method was compared to a standard global model trained without overfitting to evaluate whether this individualized fine-tuning provided significant advantages in anomaly detection. The results of this comparative analysis

aimed to determine whether the enhanced sensitivity to personal data indeed translated into improved detection performance.

Clustering Approach

The second approach involved clustering participants based on similarities in their physiological data, particularly heart rate patterns. The hypothesis was that grouping participants with similar data characteristics could lead to more specialized models, potentially improving overall performance compared to a single global model trained on the entire dataset.

Clustering was conducted using the K-Means algorithm with the Dynamic Time Warping metric, which is particularly effective for time series data. By clustering clients with comparable heart rate rhythms, the model could be adapted to specific subgroups, potentially enhancing its capacity to detect anomalies within those clusters. The reasoning behind this approach is that by ensuring more homogeneous data within each cluster, the model can identify more significant patterns, resulting in enhanced accuracy in anomaly detection. Figures 4.10 and 4.11 illustrate the clusters for the Covid dataset and the healthy dataset with K=2, respectively.

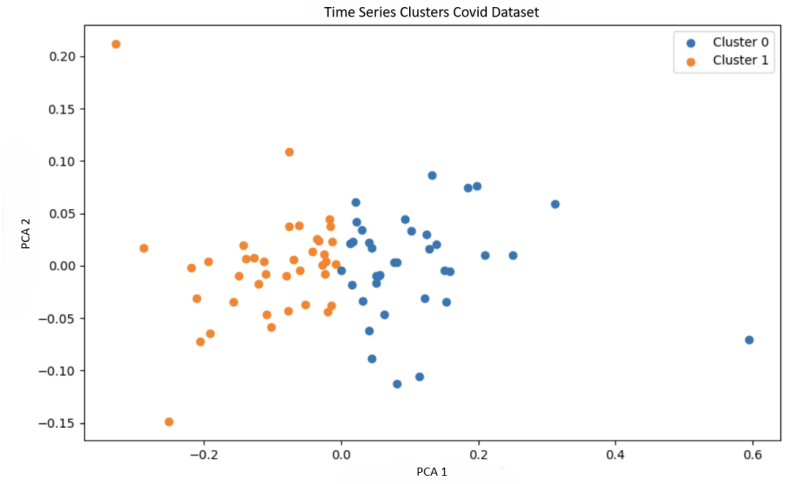


FIGURE 4.10. Covid dataset clusters representation, K=2.

Additionally, clustering with K=3 was performed, as shown in Figures 4.10 and 4.13. This difference from K=2 to K=3 was explored to assess whether increasing the number of clusters would capture more detailed subgroup patterns in the data, potentially enhancing the model's ability to detect anomalies by accounting for more nuanced variations in heart rate patterns. It is possible to observe that both data values are significantly different between both principal components. However, while the clustering was based on heart rate patterns, the specific features extracted and used to form the clusters were not explicitly defined, as the process was highly data-driven.

Further increases in K were not considered, as visual inspection of the clusters revealed no significant improvement in the separation between groups beyond k=3.

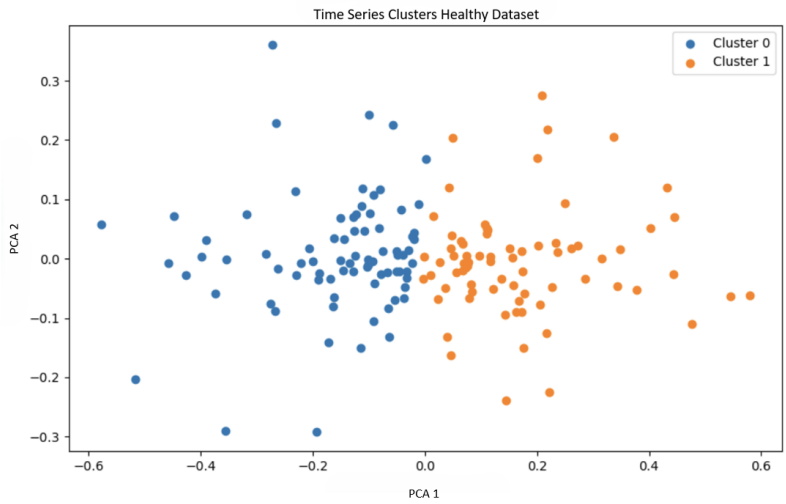


FIGURE 4.11. Healthy dataset clusters representation, K=2.

FIGURE 4.12. Covid dataset clusters representation, K=3

FIGURE 4.13. Healthy dataset clusters representation, K=3

Top 20 Participants Approach

The third approach focused on selecting the top 20 participants based on their individual performance during training. The goal was to assess whether using this high performing subset could lead to a generalized model that performs better across the entire dataset. The hypothesis posits that these top performers could more accurately reflect the underlying patterns in the data, resulting in a more robust and resilient global model.

The training of the model on the top 20 selected participants was subsequently tested on the entire dataset to evaluate its effectiveness. The rationale was that participants who exhibited better individual performance during training might contribute more reliable and meaningful patterns, enhancing the model's anomaly detection capabilities when applied to all participants. This method aimed to improve the model's generalization ability, making it more adaptable and resilient beyond the high-performing subset.

4.6. Results

This section presents and analyzes the results obtained in identifying anomalies related to COVID-19 using resting heart rate data. In total, predictions were made across 192 distinct scenarios, distributed between two datasets (Covid and Healthy). The analysis explored deeply into two FL strategies, FedAvg and FedAdam, while also examining the performance of two distinct loss functions: MAE and MSE. Additionally, three methods for determining anomaly thresholds were analyzed: 1STD, Percentile and Variance Threshold. Both LSTM and Convolutional autoencoder models were employed, carefully examining variations that addressed both overfitting and non overfitting scenarios. Furthermore, the clustering method and the examination of the top 20 participants were implemented across all these scenarios, restricted to the FedAvg strategy. To facilitate a clearer interpretation of the results, the analysis will focus primarily on the Covid dataset, while a distinct section will be dedicated to comparing the two datasets. The analysis follows a systematic structure to compare each approach in a stepwise manner:

(1) **Federated Learning Strategy**: A comparison of two FL strategies, FedAvg and FedAdam, to explore their impact on anomaly detection performance.
(2) **Loss Functions**: Evaluation of MAE and MSE, focusing on how each loss function influences sensitivity in detecting anomalies through threshold comparisons.
(3) **Models Autoencoders**: An assessment of two autoencoder architectures, LSTM and Convolutional, to identify which model better reconstructs time series data and effectively detects anomalies.
(4) **Anomaly Threshold Methods**: This analysis compares different threshold methods, including Threshold1STD, Percentile Threshold and Variance Threshold to evaluate how threshold selection influences anomaly detection outcomes.
(5) **Overfitting in Model Training**: This analysis examines the effect of overfitting on model performance. The performance of models subjected to overfitting is

compared to those trained without overfitting, too determine the implications of this technique.

(6) **Clustering in Model Training**: The use of clustering is explored to determine whether training models based on clusters of participants with similar characteristics improves overall performance compared to using a single global model.

(7) **Top 20 Participants in Model Training**: This analysis focuses on the top 20 participants with the most accurate predictions to evaluate whether training on this subset enhances the model's performance for all participants.

(8) **Comparison Between the Two Datasets**: A comparative analysis is conducted to examine how the Covid and Healthy datasets differ in terms of anomaly detection performance in healthy and unhealthy participants.

(9) **Federated Learning vs Centralized and Individualized Learning**: The results of the FL models are compared with those of centralized and individualized models. This comparison highlights the advantages and disadvantages of FL in relation to centralized and individualized approaches.

(10) **Identification of Patterns in Anomaly Detection**: This section tries to identify patterns in anomaly detection, examining whether consistent trends can be found across participants.

In this study, the following evaluation metrics were used to assess the performance of the models: recall, precision, balanced accuracy and specificity. Each metric provides valuable insights into different aspects of model performance, particularly in dealing with the imbalanced nature of the datasets. Below is a brief explanation of each metric and its corresponding formula, where TP stands for true positives, TN represents true negatives, FP denotes false positives, and FN refers to false negatives.

- **Recall**: Recall measures the ability of the model to correctly identify positive cases. It is the ratio of true positives to the sum of true positives and false negatives.

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

- **Precision**: Precision measures the proportion of true positive predictions among all positive predictions made by the model. It evaluates the ability of the model to avoid false positives.

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

- **Balanced Accuracy**: Balanced accuracy is the average of sensitivity and specificity and it is particularly useful for evaluating imbalanced datasets, as it considers both classes equally.

$$\text{Balanced Accuracy} = \frac{1}{2}\left(\frac{\text{TP}}{\text{TP} + \text{FN}} + \frac{\text{TN}}{\text{TN} + \text{FP}}\right)$$

- **Specificity**: Specificity measures the proportion of correctly identified negative cases. It is the ratio of true negatives to the sum of true negatives and false positives.

$$\text{Specificity} = \frac{\text{TN}}{\text{TN} + \text{FP}}$$

To statistically validate the performance differences across the various approaches, the Wilcoxon signed-rank test [**52**] was chosen for comparing paired results. This test was selected due to its suitability for paired data where normality is not assumed, making it appropriate for assessing differences in metrics across model configurations and learning strategies. In this context, the p-value derived from the Wilcoxon test indicates statistical significance: a p-value above 0.05 suggests no significant difference, implying results may be due to chance, while a p-value of 0.05 or below indicates statistically significant differences, suggesting the observed variations are meaningful. Balanced accuracy was prioritized as the primary metric in this analysis, given its utility for assessing performance in imbalanced datasets.

4.6.1. Federated Learning Strategy

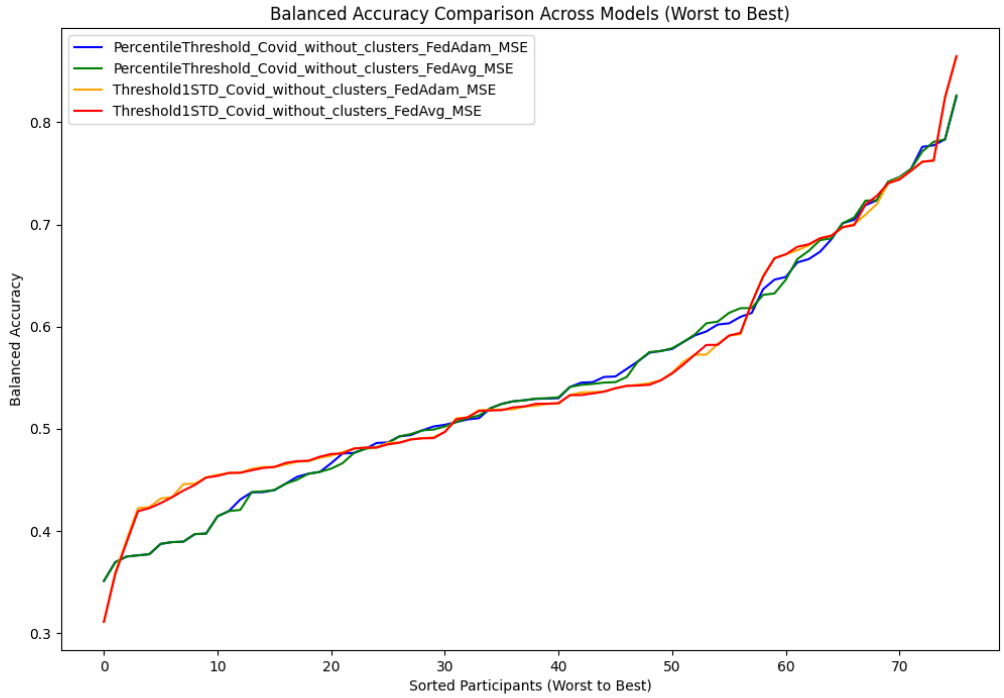


FIGURE 4.14. Balanced Accuracy Comparison of FedAvg and FedAdam on the Covid Dataset Using Convolutional Model and MSE Loss Function with Percentile and 1STD Thresholds.

In this section, a comparison is made between two different FL strategies: FedAvg and FedAdam. Both strategies were evaluated across a comprehensive set of conditions: two models (LSTM and Convolutional), two loss functions (MAE and MSE) and the three

threshold methods. This evaluation resulted in a total of 12 outcomes for each strategy, providing an understanding of their performance across different scenarios.

Figure 4.14 provides an illustrative comparison between the FedAvg and FedAdam strategies on the Covid dataset, utilizing the convolutional model, MSE loss function and two threshold methods: Percentile Threshold and 1STD Threshold. As shown in the figure, both strategies display very similar performance under identical conditions. Due to this similarity, a comprehensive table with the mean relevant metrics for all outcomes under each strategy is presented in Table 4.1.

Strategy	**Balanced Accuracy**	**Recall**	**Precision**	**Specificity**
FedAdam	0.527588	0.278973	0.152640	0.776203
FedAvg	0.523890	0.287031	0.147107	0.760748

TABLE 4.1. Mean Performance Metrics Comparison for FedAvg and FedAdam Strategies.

The results indicate that FedAvg and FedAdam exhibit similar performance across all evaluated metrics, suggesting that, for this particular application, the simpler FedAvg approach may be preferable due to its ease of implementation and lack of significant trade-offs in performance. Consequently, in scenarios where efficiency is a priority, FedAvg may be the optimal choice, especially when computational simplicity and resource optimization are necessary.

4.6.2. Loss Functions

The next step involved comparing two loss functions widely used in anomaly detection problems: MAE and MSE. This comparison was conducted in the Covid dataset, implementing FedAvg as the FL strategy, for both LSTM and Convolutional models, across the three threshold methods.

Figure 4.15 presents an illustrative comparison between MSE and MAE loss functions, showing their performance on the Covid dataset using the Percentile and 1STD threshold methods, with the convolutional model and the FedAvg strategy. As observed in the figure, it is not clear to visually discern which loss function performs better under this scenario. Therefore, Table 4.2 provides an overview of the mean relevant metrics for all outcomes associated with each loss function for a clearer analysis.

Loss function	**Balanced Accuracy**	**Recall**	**Precision**	**Specificity**
MAE	0.525683	0.268226	0.146883	0.783139
MSE	0.526296	0.252847	0.150381	0.799744

TABLE 4.2. Mean Performance Metrics Comparison for MAE and MSE Loss Functions.

Figure 4.15 and Table 4.2 do not reveal substantial differences in performance between the MAE and MSE loss functions. Minor mean metric variations suggest a nearly identical

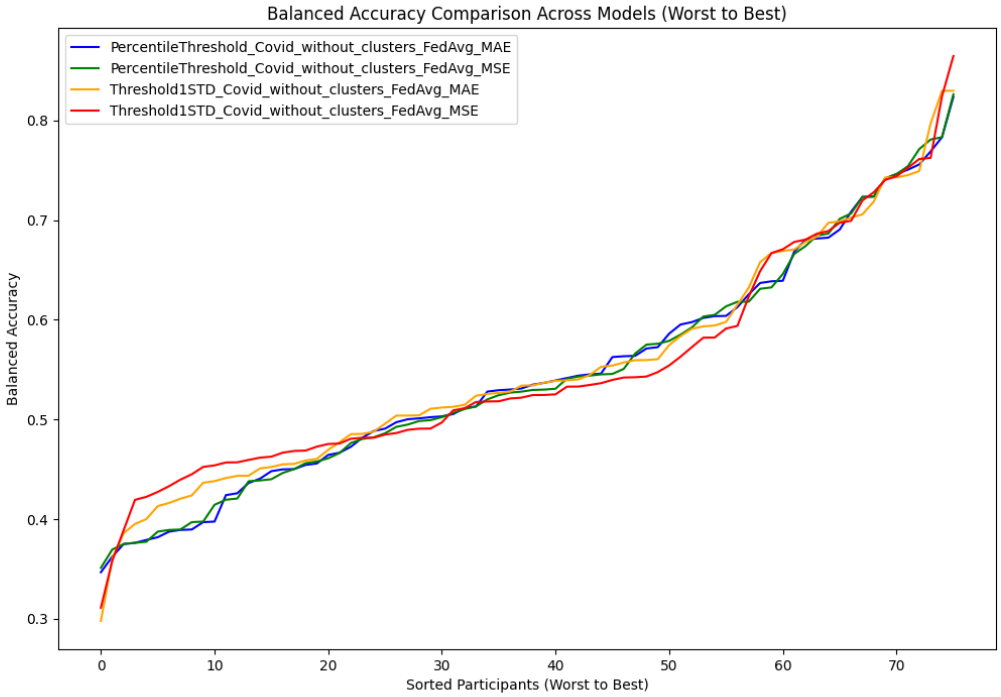


FIGURE 4.15. Balanced Accuracy Comparison of MAE and MSE Loss Functions on the Covid Dataset Using Convolutional Model, FedAvg Strategy, with Percentile and 1STD Thresholds.

performance across both loss functions, consistent with theoretical expectations given their closely related mathematical properties.

Given these minimal differences, it can be concluded that both loss functions perform similarly in this scenario. To streamline the analysis, only the MSE will be used in subsequent evaluations, facilitating comparisons across other parameters without compromising model effectiveness.

4.6.3. Models Autoencoders

The next phase involves conducting a comparative analysis between the two models employed in this study: LSTM and Convolutional Autoencoders. This comparison is conducted in the Covid dataset, using FedAvg as the FL strategy and MSE loss function across the three threshold methods.

Figure 4.16 illustrates the comparison between LSTM and Convolutional Autoencoder models, demonstrating their performance on the Covid dataset. This comparison evaluates both models using two threshold methods: Percentile Threshold and 1STD Threshold, under the FedAvg strategy and MSE loss function. As shown in the graph, the Convolutional model outperforms the LSTM model in terms of balanced accuracy. Table 4.3 offers a detailed overview of the mean relevant metrics for all outcomes associated with each model, facilitating a clearer and more comprehensive analysis.

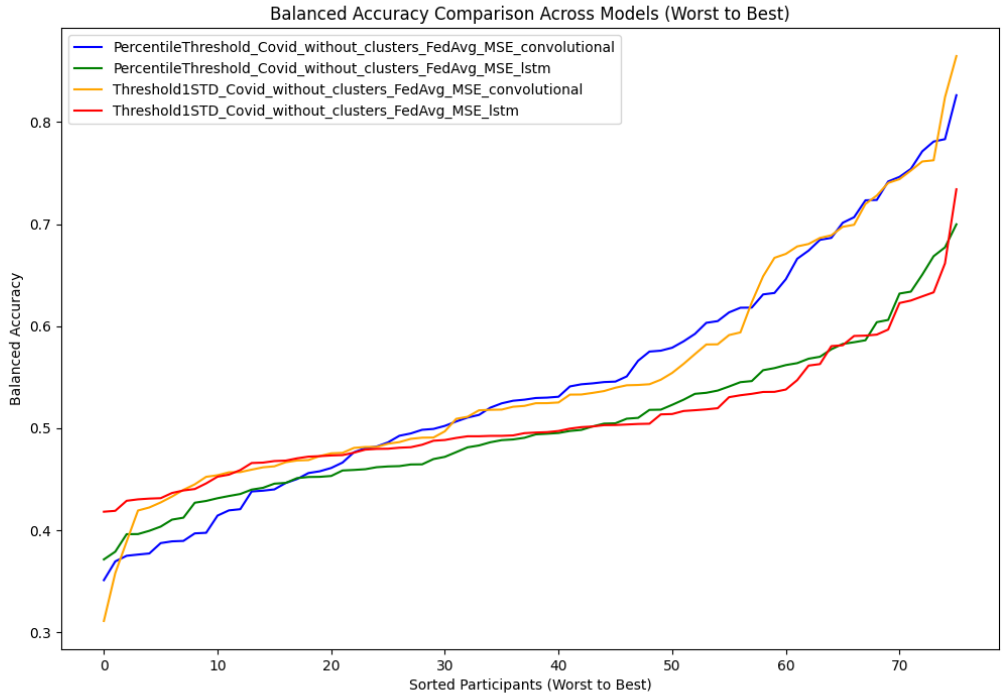


FIGURE 4.16. Balanced Accuracy Comparison of LSTM and Convolutional Models on the Covid Dataset Using FedAvg Strategy, MSE Loss Function, with Percentile and 1STD Thresholds.

Model	**Balanced Accuracy**	**Recall**	**Precision**	**Specificity**
LSTM	0.500586	0.261907	0.124098	0.739265
Convolutional	0.547194	0.312155	0.170116	0.782232

TABLE 4.3. Mean Performance Metrics Comparison for LSTM and Convolutional models

As shown in Figure 4.16, the Convolutional model demonstrates a clear advantage in performance when compared to the LSTM model. This is further supported by Table 4.3, which presents the performance metrics across all tested approaches, with the Convolutional model consistently outperforming the LSTM model across all evaluated metrics. Additionally, the Wilcoxon test results (p-value: 0.0008) confirm that this performance difference is statistically significant, indicating that the Convolutional model's superior results are unlikely due to random variation.

Another factor in the evaluation is the execution time required to train both models. The training times recorded in seconds for both models on the Covid and Healthy datasets are presented below in Table 4.4:

The results indicate that the Convolutional model consistently achieves shorter training times and higher performance metrics across all datasets and FL strategies. This makes the Convolutional model especially well suited for this particular scenario.

Dataset	LSTM (s)	Convolutional (s)
Covid FedAvg	1419.41	787.93
Covid FedAdam	2827.76	666.91
Healthy FedAvg	2155.89	2038.52
Healthy FedAdam	4181.12	1448.00

TABLE 4.4. Training Times for LSTM and Convolutional Models Across Different Datasets.

4.6.4. Anomaly Threshold Methods

This subsection presents a comparative analysis of three threshold methods: threshold1STD, Percentile threshold and Variance threshold. This comparison was conducted in the Covid dataset, using FedAvg as the FL strategy, MSE loss function and a convolutional model. The following results were observed.

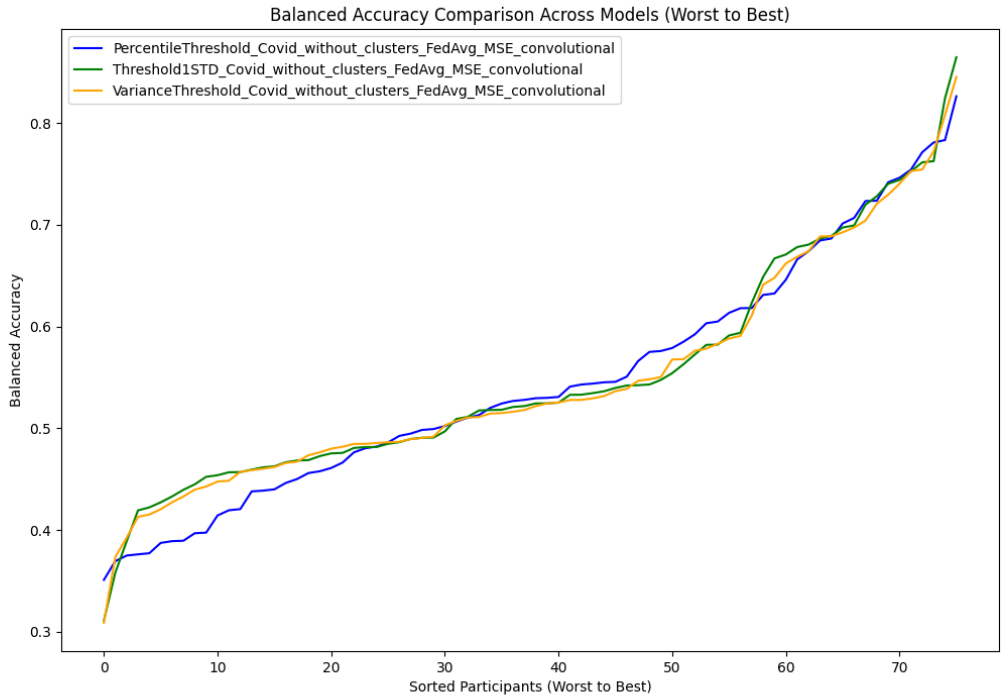


FIGURE 4.17. Balanced Accuracy Comparison of 1STD, Percentile and Variance Threshold Methods on the Covid Dataset Using the Convolutional Model, FedAvg Strategy and MSE Loss Function.

Figure 4.17 illustrates the comparison between Threshold Methods: 1STD, Percentile and Variance, demonstrating their performance on the Covid dataset, using the convolutional model, under the FedAvg strategy and MSE loss function. As shown in the graph, it is difficult to visually distinguish the differences between the three threshold methods in terms of balanced accuracy. For a clearer and more comprehensive analysis, Table 4.5 provides an overview of the mean relevant metrics for all outcomes under each threshold.

Threshold	Balanced Accuracy	Recall	Precision	Specificity
1STD	0.549748	0.242503	0.18701	0.856993
Percentile	0.545166	0.363288	0.158159	0.727044
Variance	0.547404	0.251614	0.182354	0.843195

TABLE 4.5. Mean Performance Metrics Comparison for Threshold Methods.

As shown in Table 4.5, the 1STD threshold method demonstrates a slight advantage in balanced accuracy, precision and specificity compared to the other methods. The Percentile method, however, exhibits higher recall, indicating greater sensitivity in detecting positive cases, which can be beneficial for anomaly detection contexts. The Variance method, meanwhile, has slightly lower values than 1STD in terms of balanced accuracy, precision and specificity.

Due to the minimal performance differences between the 1STD and Variance thresholds, the analysis centered on comparing the 1STD and Percentile methods. A Wilcoxon test was conducted to assess the statistical significance of their differences, with a p-value of 0.1167, indicating that variations between the two methods are not statistically significant.

Given the study's emphasis on anomaly detection, the Percentile threshold will be used in subsequent analyses for its slight recall advantage.

4.6.5. Overfitting in Model Training

The investigation into the overfitting strategy aimed to assess whether fine-tuning the global model on each user's data could lead to improved anomaly detection performance. The expectation was that personalizing the model for each user would improve its ability to identify anomalies by more effectively capturing unique user specific patterns.

As illustrated in Figure 4.18, the application of the FedAvg strategy on the Covid dataset, using the MSE loss function and convolutional model with Percentile and 1STD Threshold Methods, did not produce any performance improvements with overfitting. The results were virtually identical from those obtained in the absence of overfitting. Despite the additional fine-tuning process, which involved 10 epochs, the performance metrics remained unchanged. This suggests that the global model had already sufficiently generalized to account for individual variations or that the fine-tuning process did not add meaningful adjustments to capture the unique patterns of each user.

These results appeared unusual as they produced identical values across all metrics. Upon further investigation, it was discovered that there were minor variations in model behavior, including slight adjustments in threshold selection and loss values, but the overall values remained similar. For example, the value of the threshold difference between the overfitted and non overfitted approaches was less than 0.001. This minimal difference explains why the anomaly predictions were identical, leading to the same performance metrics across both approaches.

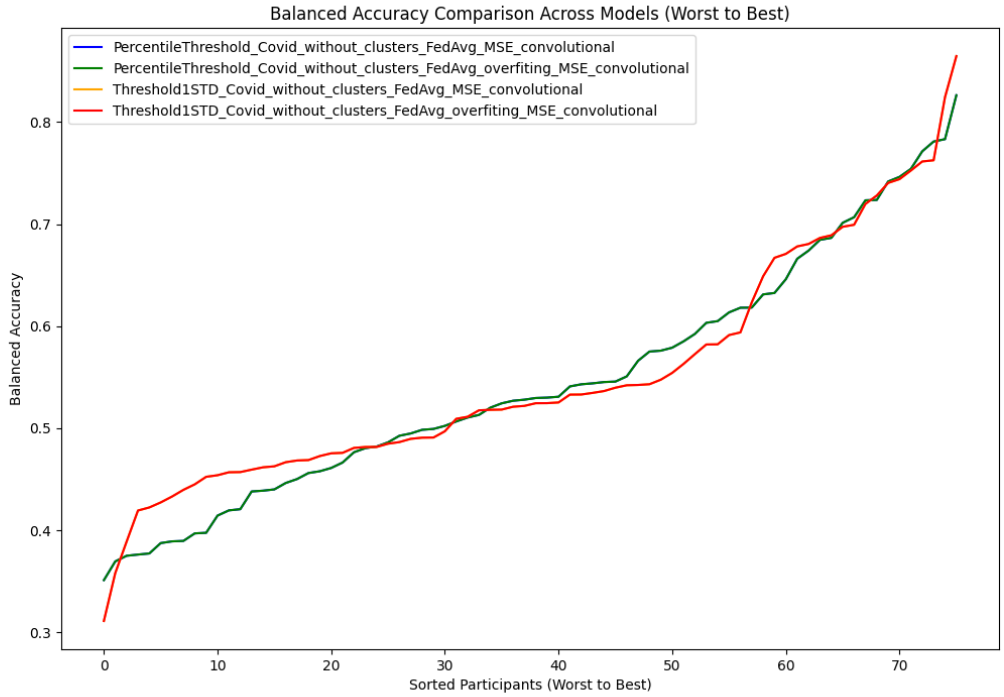


FIGURE 4.18. Comparison of Convolutional Model Performance With and Without Overfitting on the Covid Dataset Using FedAvg Strategy, MSE Loss, with Percentile and 1STD Threshold Methods.

The identical predictions of anomalies and the matching results across all metrics suggest that the overfitting process did not contribute significantly to the model's ability to detect anomalies.

Given that the convolutional model showed no notable differences, the next phase involved assessing the overfitting strategy using the LSTM model to determine if similar patterns occurred. Figure 4.19 showcases the results derived from the Covid dataset using the FedAvg strategy, the MSE loss function and the Percentile threshold method with the LSTM model.

The results reveal a slight improvement when overfitting is applied. The LSTM model benefits more from the overfitting process, as the difference is noticeable in this case. This improvement is consistent across other performance metrics, as further illustrated in Table 4.6.

Overfitting	**Balanced Accuracy**	**Recall**	**Precision**	**Specificity**
TRUE	0.518291	0.29447	0.134687	0.742112
FALSE	0.502032	0.260968	0.124514	0.743096

TABLE 4.6. Mean Performance Metrics Comparison With and Without Overfitting.

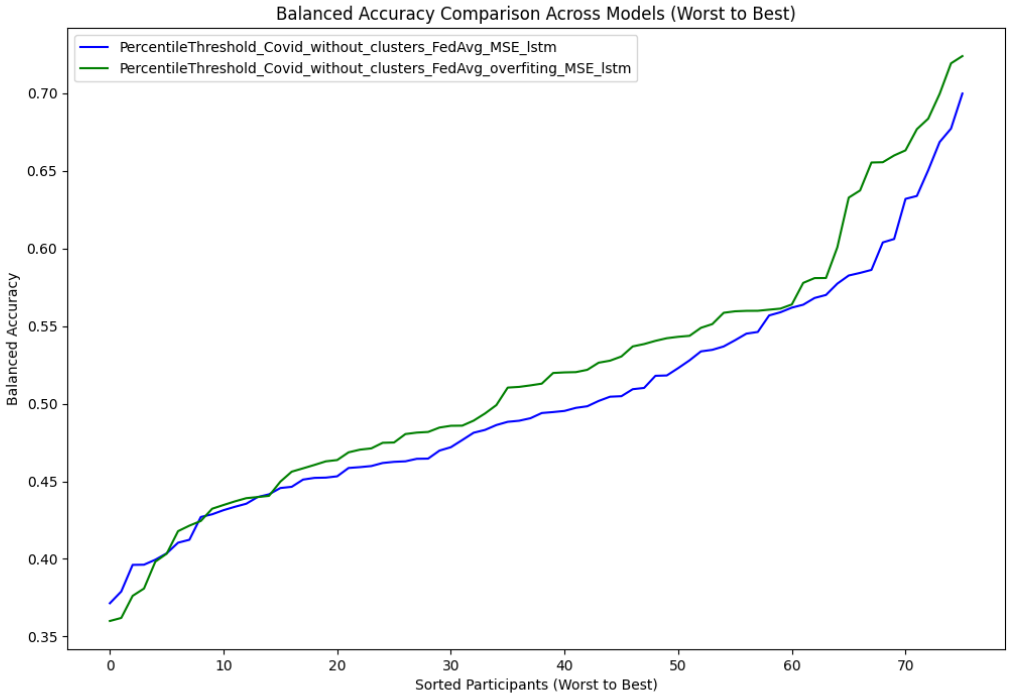


FIGURE 4.19. Comparison of LSTM Model Performance With and Without Overfitting on the Covid Dataset Using FedAvg Strategy, MSE Loss and Percentile Threshold.

The hypothesis that overfitting could improve model performance by capturing user specific patterns was partially supported by the results. For the convolutional model, no improvement was observed when overfitting was applied. Both the overfitted and non overfitted models had nearly identical performance metrics, suggesting that the global model was already generalized enough to account for individual variations, or that the 10 epochs of fine-tuning did not provide significant additional adjustments. This implies that overfitting may not be effective for this type of model, where further personalization did not lead to any discernible gains.

In contrast, applying the overfitting strategy to the LSTM model appeared to yield improvements across most evaluated metrics, suggesting that personalizing the global model allowed it to better capture user-specific patterns, potentially enhancing its anomaly detection capabilities. However, the Wilcoxon test results (p-value: 0.4654) indicate that these performance differences between the overfitted and non-overfitted LSTM models are not statistically significant. Therefore, while the metrics suggest some enhancement, the absence of statistical significance implies that the observed improvements may not be reliably superior. This outcome suggests that, although overfitting can align the LSTM model more closely with user-specific patterns, it does not result in a statistically meaningful improvement in model performance.

4.6.6. Clustering in Model Training

The clustering strategy was designed to explore whether grouping users with similar characteristics could improve the model's performance by allowing it to learn from more homogeneous subsets of data. The hypothesis was that models tailored to specific clusters would be better at detecting anomalies than a single global model.

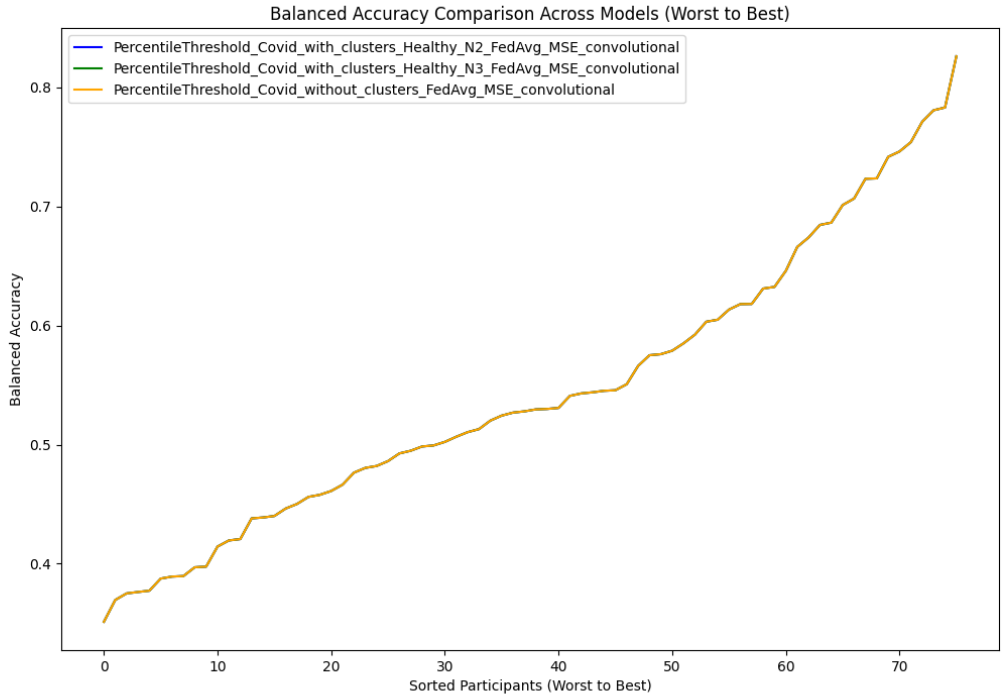


FIGURE 4.20. Comparison of Convolutional Model Balanced Accuracy With and Without Clustering on the Covid Dataset Using FedAvg Strategy, MSE Loss and Percentile Threshold. Clustering was applied with K = 2 and K = 3.

Figure 4.20 shows that, for the Covid dataset using the FedAvg strategy, the MSE loss function, the convolutional model and the Percentile threshold, no improvement was observed when clustering approaches were applied. The results appeared identical to those without clustering. While there were subtle variations in model behavior like in the overfitting approach to the convolutional model, such as slight differences in the selection of thresholds and minor adjustments in loss values, these did not translate into meaningful improvements in the overall performance metrics. Even when analyzed alongside the graphical representations, no significant differences were observed between the clustered and non clustered approaches, reinforcing that clustering had minimal impact on convolutional model performance in this context.

Since no significant differences were observed with the convolutional model, the next step was to test the clustering approach with the LSTM model to determine if a similar pattern happened. In the overfitting approach, some differences were noticeable when

using the LSTM model, suggesting potential benefits. The results, shown in Figure 4.21 for the Covid dataset using the FedAvg strategy, MSE loss function, LSTM model and Percentile threshold, indicate a slight improvement when clustering is applied. Specifically, N=3 clusters outperformed N=2 clusters, with both showing better results compared to the non clustered approach. This improvement is consistent across several performance metrics, as further detailed in Table 4.7.

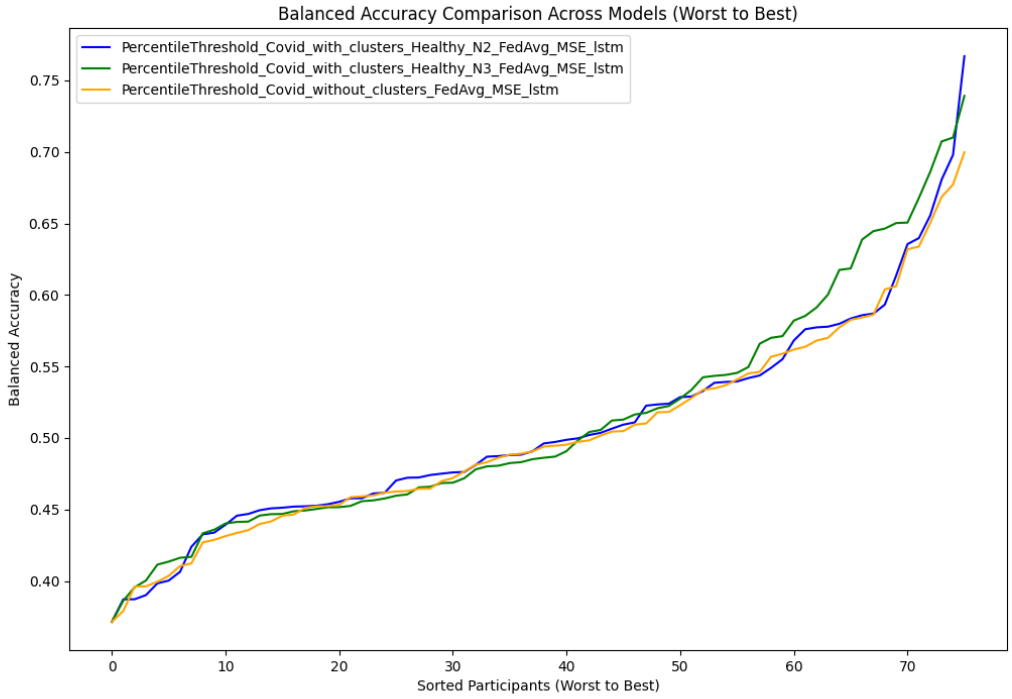


FIGURE 4.21. Comparison of LSTM Model Balanced Accuracy With and Without Clustering on the Covid Dataset Using FedAvg Strategy, MSE Loss and Percentile Threshold. Clustering was applied with K = 2 and K = 3.

Clusters	Balanced Accuracy	Recall	Precision	Specificity
N3	0.511462	0.279994	0.128742	0.742930
N2	0.506158	0.271080	0.125682	0.741236
FALSE	0.502032	0.260968	0.124514	0.743096

TABLE 4.7. Mean Performance Metrics Comparison with clustered and non-clustered approaches.

The clustering approach was intended to group participants based on the similarity of their heart rate patterns, with the hypothesis that this would lead to more specific models and better overall results. However, the results from the convolutional model showed no significant improvements, as the clustered and non clustered approaches had nearly identical performance across all metrics.

When the LSTM model was evaluated with the clustering approach, slight performance improvements were observed, with N=3 clusters outperforming N=2 clusters. This suggests that the LSTM model may be more responsive to clustering strategies, benefiting from subgroup-specific tuning that leverages participants with similar physiological characteristics. However, Wilcoxon signed-rank test results indicate that these improvements are not statistically significant: comparing the non-clustered model to N=3 yielded a p-value of 0.5844 and to N=2 yielded a p-value of 0.3132. Although modest, these findings imply that clustering may provide a slight advantage for LSTM models in anomaly detection tasks, yet this effect should be interpreted with caution given the statistical results.

4.6.7. Using Top 20 Participants in Model Training

In this section, the top 20 participants were selected based on their balanced accuracy performance on the Covid dataset. The selection process utilized the Percentile Threshold method, the MSE loss function and the FedAvg strategy. This subset of participants was then used to train a convolutional model with the same characteristics, which was subsequently tested on the entire dataset.

The thinking behind this strategy was that participants with the highest performance could provide more consistent and reliable patterns, potentially leading to a model that generalizes better across the full dataset. The hypothesis was that by focusing on these top performers, the model might capture more robust patterns, thereby enhancing the detection of anomalies when applied to all participants.

As shown in Figure 4.22, can be observed the performance of the top 20 participants subset compared to the full dataset, trained using the FedAvg strategy, the Percentile threshold, the MSE loss function and the convolutional model. The results are similar, with no significant changes observed between the two approaches.

Since no significant changes were observed in the comparison between the two approaches for the convolutional model, the same approaches were applied using the LSTM model. In Figure 4.23, we can observe the balanced accuracy performance of the LSTM model with two thresholds: 1STD and Percentile, using the MSE loss function and the FedAvg strategy for the Covid dataset. It appears that the model trained on the top 20 participants subset shows slightly better performance. However, the differences are subtle and hard to discern from the graph alone. Therefore, all relevant mean metrics are presented in Table 4.8 for a more comprehensive comparison, revealing a slight improvement in most metrics when comparing the model trained on the top 20 participants to the model trained on the full dataset.

This suggests that training the model on a carefully selected subset of high-performing participants may capture more consistent patterns, resulting in better anomaly detection across the dataset. While the improvement is modest, it indicates that focusing on specific, well-performing data points could potentially lead to enhanced model performance.

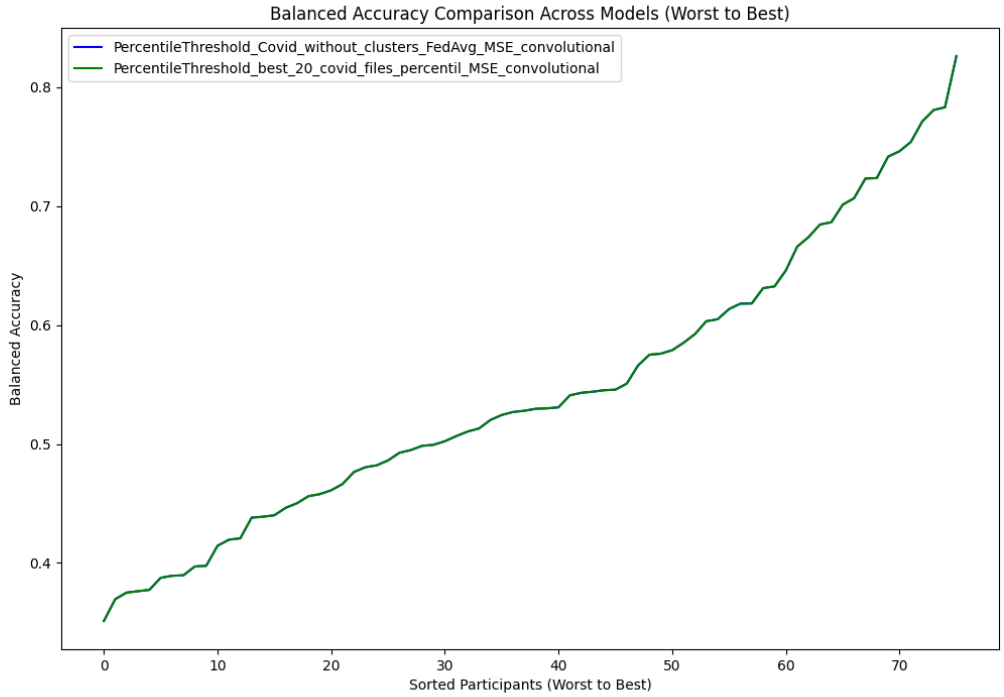


FIGURE 4.22. Comparison of Balanced Accuracy Between Top 20 Participants and Full Dataset Using Convolutional Model, FedAvg Strategy, MSE loss function and Percentile Threshold.

Model	Balanced Accuracy	Recall	Precision	Specificity
Percentil	0.502032	0.260968	0.124514	0.743096
Percentil_20	0.508584	0.277594	0.130263	0.739574
1STD	0.508236	0.144630	0.131841	0.871842
1STD_20	0.511916	0.149083	0.138832	0.874748

TABLE 4.8. Comparison of Mean Performance Metrics for Percentile and 1STD Thresholds Using Top 20 Participants vs. Full Dataset

This suggests that training the model on a carefully selected subset of high-performing participants may capture more consistent patterns, resulting in better anomaly detection across the dataset. While the improvement is modest, it indicates that focusing on specific, high-performing data points could potentially lead to enhanced model performance. To validate this difference, a Wilcoxon signed-rank test was conducted, yielding a p-value of 0.143, indicating that the observed improvement is not statistically significant. This suggests that while there may be a slight benefit to using top-performing participants, the effect should be interpreted with caution.

4.6.8. Federated Learning vs Centralized and Individualized Learning

This section aims to compare the best performing FL approach with both the centralized and individualized learning approaches. In the individualized approach, each model is

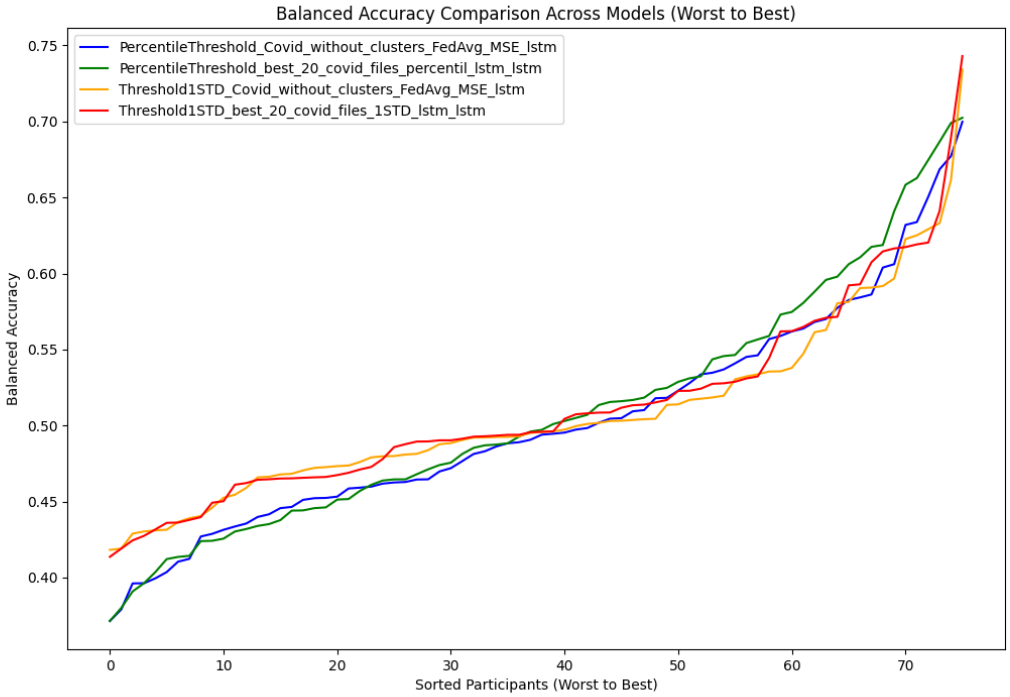


FIGURE 4.23. Comparison of Balanced Accuracy Between Top 20 Participants and Full Dataset Using LSTM Model, FedAvg Strategy, MSE loss function and Percentile Threshold.

trained exclusively on a single participant's data and tested on that same participant. This allows the model to focus on participant specific patterns, potentially capturing nuances unique to each individual. In the centralized approach, all participant data is pooled together to train a single global model, which is then tested on all participants, allowing the model to generalize across the entire dataset.

In Figure 4.24, the federated model using the FedAvg strategy, convolutional model, 1STD Threshold and MSE loss function is compared both the individualized and centralized models, which also use the convolutional model, 1STD Threshold and MSE loss function. Both approaches were applied to the Covid dataset and as can be observed the results are quite similar, with no significant difference in performance between the three approaches. The balanced accuracy, along with other relevant metrics, remained relatively unchanged across the two approaches, suggesting that the generalized model trained using FL did not substantially outperform or underperform the individualized models trained in a centralized manner.

Since no significant improvement was observed with the convolutional model, a comparison was conducted using the LSTM model to examine whether similar patterns would emerge. For this analysis, the federated model was configured with the FedAvg strategy, LSTM model, 1STD threshold and MSE loss function and then compared to both the individualized and centralized models with identical configurations.

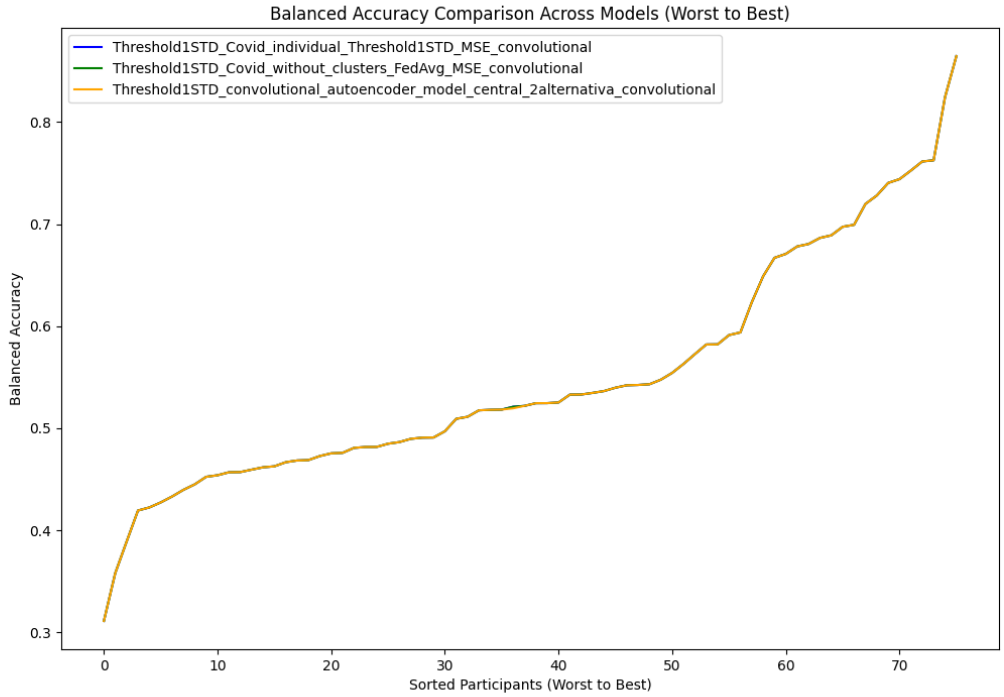


FIGURE 4.24. Comparison of Balanced Accuracy Among Federated, Centralized and Individualized Learning Approaches Using Convolutional Model, 1STD Threshold and MSE Loss Function.

In Figure 4.25, the balanced accuracy performance of the federated, centralized and individualized learning approaches is displayed, using the LSTM model along with the 1STD threshold and MSE loss function. As illustrated, the individualized model demonstrates superior performance in comparison to both the federated and centralized models, suggesting that it may be more effective in capturing participant-specific patterns and anomalies.

For a more comprehensive evaluation, Table 4.9 presents the mean performance metrics across the federated, centralized and individualized approaches. Consistently across all metrics, the individualized approach displays a performance gain, which suggests that training a model on a single participant's data may better capture unique individual characteristics and improve the model's accuracy in detecting anomalies.

Approach	**Balanced Accuracy**	**Recall**	**Precision**	**Specificity**
Federated	0.5082	0.1446	0.1318	0.8718
Central	0.5135	0.1761	0.1268	0.8510
Individual	0.5343	0.1609	0.2489	0.9076

TABLE 4.9. Comparison of Mean Performance Metrics for Federated, Central and Individual approaches

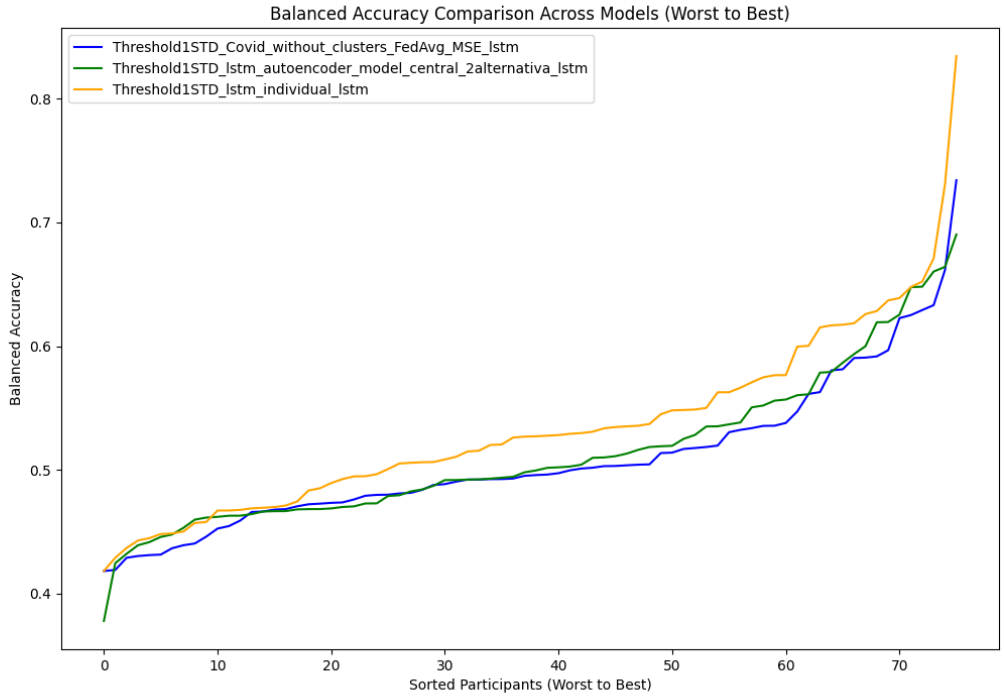


FIGURE 4.25. Comparison of Balanced Accuracy Among Federated, Centralized and Individualized Learning Approaches Using LSTM Model, 1STD Threshold and MSE Loss Function

The comparison between FL, centralized learning and individualized Learning revealed that, for the convolutional model, no significant performance differences were observed between the three approaches. However, when using the LSTM model, the individualized Learning approach demonstrated improvements in performance metrics, as seen in Table 4.9. This suggests that individualized learning, when applied to the LSTM model, may better capture participant specific patterns, leading to improved anomaly detection.

To statistically assess the significance of these differences, Wilcoxon signed-rank tests were conducted. The comparison between the federated and individualized approaches yielded a p-value of 0.0002, indicating a statistically significant advantage for the individualized approach. Additionally, comparing the individualized and centralized approaches resulted in a p-value of 0.00007, further confirming the statistical significance of the individualized approach. Meanwhile, the comparison between the federated and centralized approaches returned a p-value of 0.2504, suggesting minimal statistical differences between these two methods.

These findings suggest that individualized learning offers clear benefits for capturing unique participant patterns. However, the similar performance levels and statistical results between the federated and centralized approaches highlight the value of FL, especially in applications where data privacy and decentralized data processing are prioritized.

4.6.9. Comparison Between the Two Datasets

The analysis thus far has focused on variations within the Covid dataset. However, in real-world applications, healthy data would typically be far more prevalent than data from individuals with COVID-19. Consequently, it is essential to evaluate model performance under conditions that more closely align with such real-world distributions. In Figure 4.26, a comparison between the two datasets is presented in terms of balanced accuracy, utilizing the FedAvg strategy, MSE loss function and convolutional model with two threshold methods: 1STD and Percentile. Balanced accuracy was selected for this comparison, as recall cannot be calculated when there are no anomalies present in the healthy dataset. The results indicate that balanced accuracy is consistently higher for the Healthy dataset than for the Covid dataset, which may reflect the inherently more predictable nature of healthy data or the model's greater proficiency in identifying anomalies within unhealthy data or a combination of these factors.

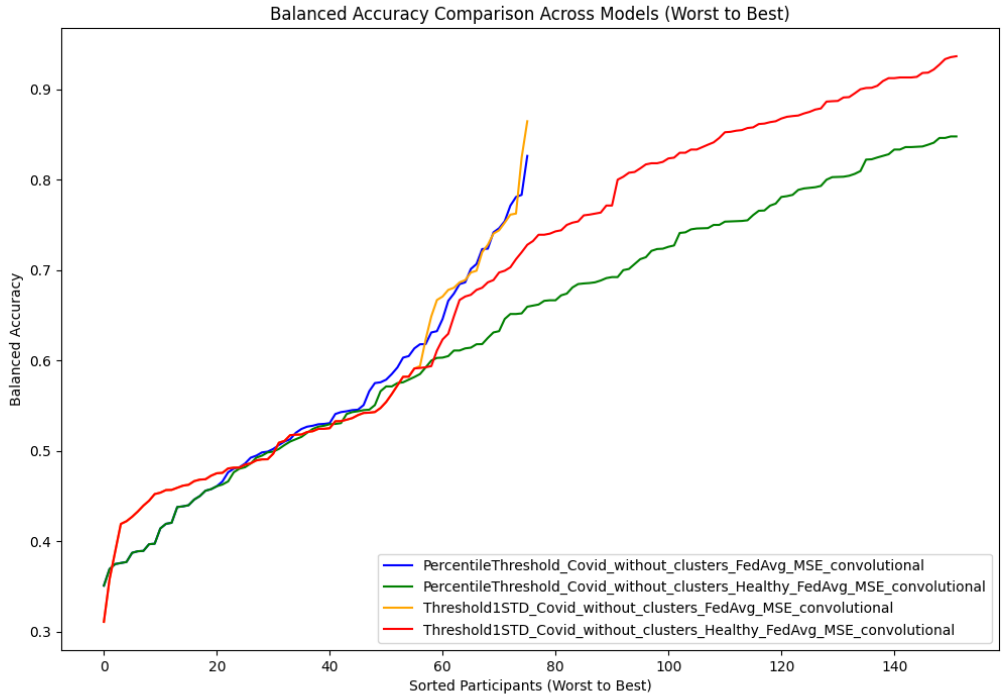


FIGURE 4.26. Balanced Accuracy Comparison Across Healthy and Covid Datasets Using the Convolutional Model, FedAvg Strategy, MSE Loss Function, with 1STD and Percentile Thresholds.

To further investigate whether the improved performance in the Healthy dataset is due to the inclusion of more accurately classified healthy data or by the effect of training on an expanded dataset (incorporating healthy participants), has improved the model's performance for unhealthy participants, a focused comparison was conducted. Specifically, only the unhealthy participants from the Healthy dataset were isolated and directly compared with those from the Covid dataset. As illustrated in Figure 4.27, this analysis

was performed using the FedAvg strategy, MSE loss function and convolutional model across two threshold methods: 1STD and Percentile. Results reveal that performance metrics for unhealthy participants in the Healthy dataset closely align with those in the Covid dataset. This indicates that the primary performance improvement comes from the inclusion of healthy data, which the model finds easier to classify. Additionally, it was observed that the 1STD threshold outperforms the Percentile threshold in the Healthy dataset.

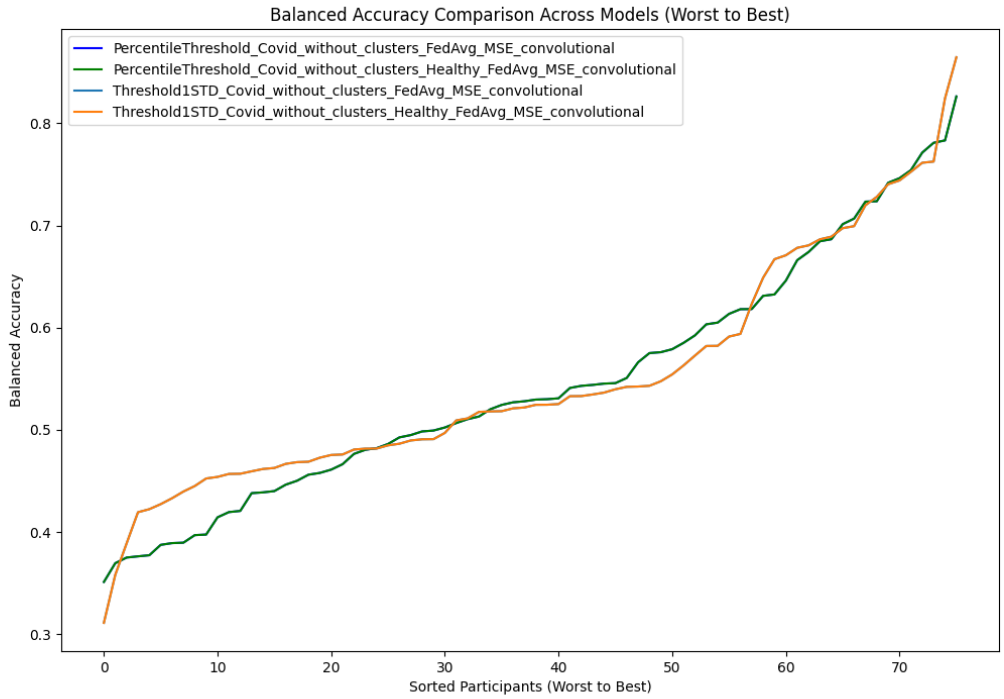


FIGURE 4.27. Balanced Accuracy Comparison for Unhealthy Participants in Healthy and Covid Datasets Using Convolutional Model, FedAvg Strategy, MSE Loss Function, with 1STD and Percentile Thresholds.

The findings were somewhat unexpected, as it was initially hypothesized that the model's performance in detecting COVID-19 anomalies might either improve with the introduction of additional healthy data, providing more baselines for detecting patterns or deteriorate due to potential over-generalization. However, no significant changes were observed. Further investigation revealed that while different thresholds were applied to individual participants, the variations were negligible (in most cases less than 0.001), leading to nearly identical results between the two datasets. This suggests that the increased volume of healthy data did not substantially influence the model's ability to detect anomalies in unhealthy individuals.

Despite the lack of improvement, there was also no decline in performance with the larger dataset. This indicates that in a real-world scenario, where healthy data is more

prevalent, the model can maintain consistent performance without being negatively affected by the increase in healthy data. The absence of personalization, which was initially thought to be a limiting factor, did not appear to significantly hinder the model's ability to generalize effectively.

Since no significant differences were observed when using the convolutional model to compare the performance across the two datasets, the LSTM model was employed to explore whether distinct patterns might emerge. Figure 4.28 presents four models, two trained on the Healthy dataset and two on the Covid dataset, all using the LSTM model, the MSE loss function, the FedAvg strategy and two threshold methods: Percentile and 1STD. The results indicate that the Healthy dataset achieves better balanced accuracy compared to the Covid dataset, with the 1STD threshold outperforming the Percentile threshold, similar to findings with the convolutional model.

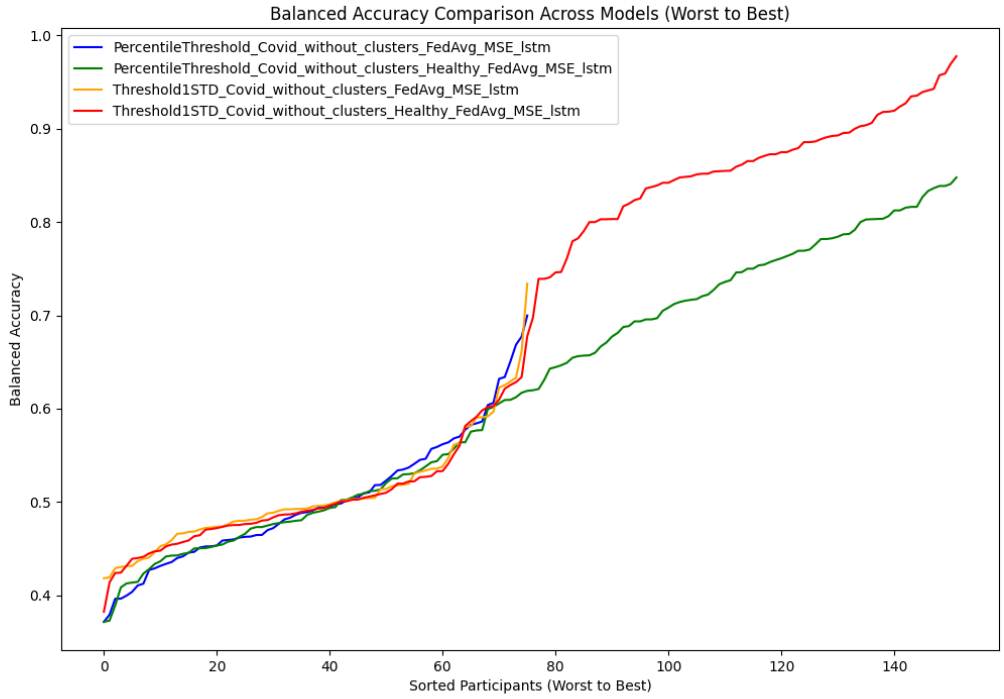


FIGURE 4.28. Balanced Accuracy Comparison Across Healthy and Covid Datasets Using the LSTM Model, FedAvg Strategy, MSE Loss Function, with 1STD and Percentile Thresholds.

To assess whether the performance would remain consistent for the unhealthy data across the two datasets, Figure 4.29 presents the same four models applied to the unhealthy data. Although some differences can be observed between the models trained on the Covid and Healthy datasets, the overall mean performance metrics appears similar. As further detailed in Table 4.10, the average results are nearly identical, suggesting that the generalization from the Healthy dataset did not significantly affect the model's ability to detect anomalies in the unhealthy data.

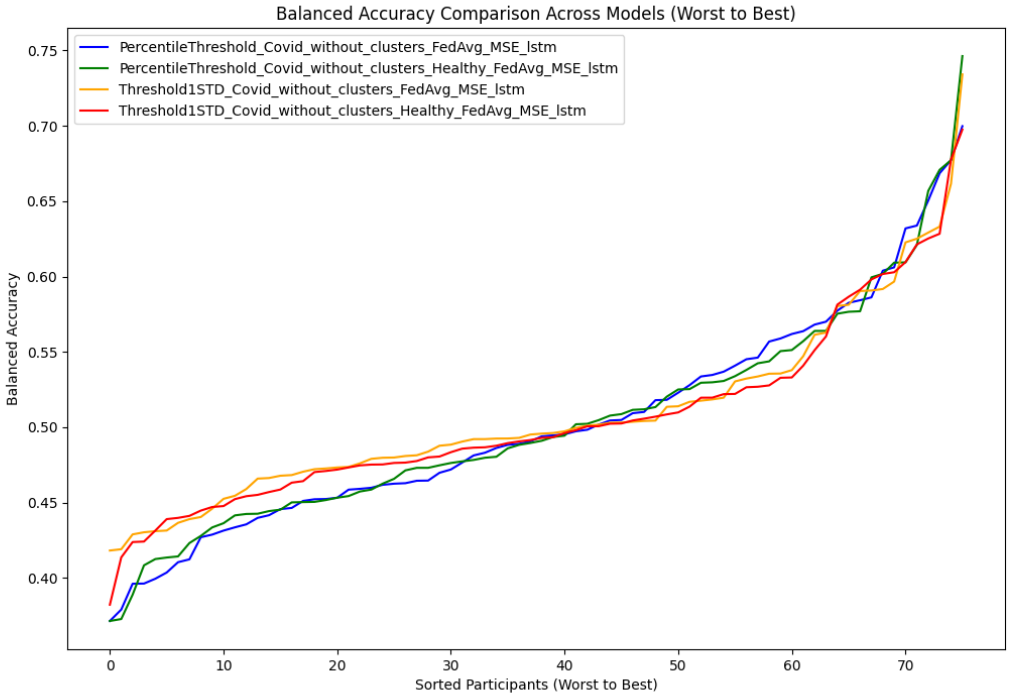


FIGURE 4.29. Balanced Accuracy Comparison for Unhealthy Participants in Healthy and Covid Datasets Using LSTM Model, FedAvg Strategy, MSE Loss Function, with 1STD and Percentile Thresholds.

Model	Balanced Accuracy	Recall	Precision	Specificity
Percentile_Covid	0.502032	0.260968	0.124514	0.743096
Percentile_Healthy	0.502489	0.258773	0.124319	0.746204
1STD_Covid	0.508236	0.144630	0.131841	0.871842
1STD_Healthy	0.505050	0.139433	0.121735	0.870668

TABLE 4.10. Comparison of Mean Performance Metrics for the unhealthy data in the Covid and Healthy datasets

Both models exhibited higher balanced accuracy on the Healthy dataset compared to the Covid dataset, likely due to an enhanced capacity for generalization when trained with healthy data. The results across both datasets suggest that the inclusion of healthy data did not negatively affect the model's anomaly detection capability. Although it was initially hypothesized that an increase in healthy data might either strengthen or weaken detection performance, the model demonstrated stable outcomes across varied scenarios, indicating an absence of over-generalization tendencies.

4.6.10. Identification of Patterns in Anomaly Detection

Following the analysis of the models and the examination of the results, additional efforts were directed towards identifying patterns in anomaly detection to better distinguish COVID-19. Given that the initial results did not meet expectations, the goal of this section

is to explore potential patterns that could improve the accuracy of identifying COVID-19 related anomalies. To address this, two hypotheses were formulated and tested:

(1) The first hypothesis proposes that a participant should only be flagged as having COVID-19 if the model detects a sequence of consecutive anomalies over N consecutive time series points.

(2) The second hypothesis suggests that within a sliding window of 10 anomaly predictions, at least N of those predictions must indicate an anomaly for COVID-19 to be flagged.

As indicated in the previous analysis, the Covid and Healthy datasets exhibited similar patterns when comparing the unhealthy data from both sources. Consequently, the Healthy dataset will be used for further analysis, allowing examination across both healthy and unhealthy data segments within the same dataset.

To rigorously test the formulated hypotheses, various values of N were selected. For the first hypothesis, which requires a sequence of consecutive anomalies to predict a COVID-19 event, values of N = 6, N = 7 and N = 8 were tested. For the second hypothesis, which requires at least N anomalies within a sliding window of 10 predictions for a COVID-19 flag, the same values of N were assessed.

For clarity in the analysis, the following labels were defined to evaluate these hypotheses in the graphs:

- **True_Covid**: The model detected COVID-19 for a participant across N time series, with all detections accurately corresponding to a true period of anomalous behavior. This confirms the prediction as accurate, with detections occurring exclusively during the anomalous period.
- **Covid**: The model detected COVID-19 for a participant across N time series, with a mix of correct and incorrect anomaly classifications. This indicates a pattern of covid detections across both anomalous and non-anomalous periods.
- **False_Nomal_Covid**: The model incorrectly flagged COVID-19 across N time series within a period classified as normal, where no actual anomalous behavior was present, resulting in an incorrect classification of the anomaly for the participant.
- **False_Recovered_Covid**: The model flagged COVID-19 across N time series during a period classified as recovered, leading to an incorrect classification of the anomaly for the participant.
- **Healthy**: The model did not detect any COVID-19 related anomaly across N time series, indicating that the data represented a healthy participant in a non-anomalous state.

To assess these hypotheses, the FedAvg model was applied to the Healthy dataset, using the 1STD threshold, convolutional model and MSE loss function. Figures 4.30 and 4.33 present the classification counts for unhealthy and healthy data, respectively, both considering the hypotheses and labels defined earlier.

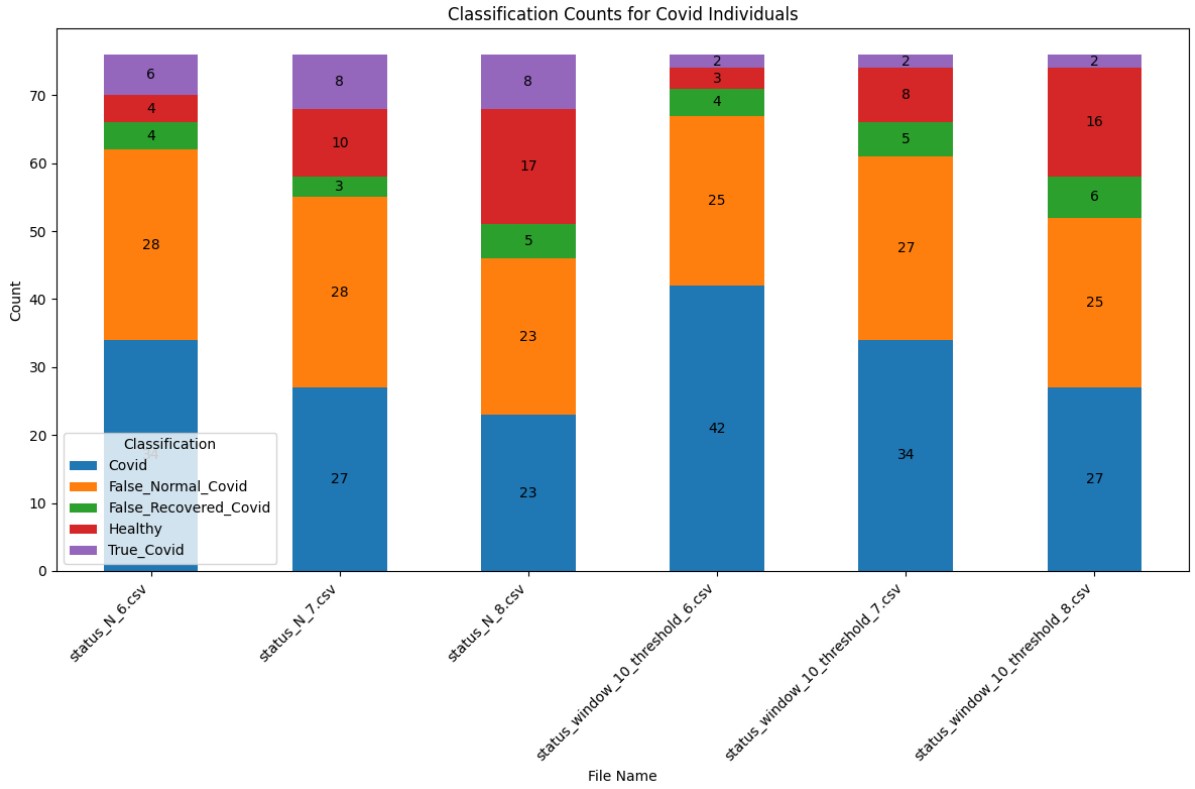


FIGURE 4.30. Classification counts for unhealthy data using the convolutional model, FedAvg strategy, MSE loss function and 1STD threshold.

In Figure 4.30, it is evident that the model faces challenges in accurately identifying true COVID-19 patterns. While some data points are correctly flagged as related to COVID-19, a significant portion of the predictions are wrong, reducing the overall reliability of the model. Although the model can classify instances as COVID-19, it struggles with temporal precision, often identifying anomalies in periods that are not directly linked to the onset of COVID-19.

For the first hypothesis, which relies on consecutive anomaly sequences, the best value for N was found to be N=7. This value resulted in only 10 instances being classified as healthy in the unhealthy dataset, while 8 instances were correctly identified as COVID-19. This indicates that the number of healthy classifications within the unhealthy data is relatively small, which is promising. However, the model's inconsistency in pinpointing the exact onset of COVID-19 remains a concern. Although the model correctly identified 27 + 8 instances of COVID-19, in 27 of those cases, it also predicted COVID-19 during unrelated periods, thereby reducing its precision. This over classification suggests that the model is overly sensitive to any deviations in the data, potentially mistaking normal fluctuations for signs of infection.

When evaluating the second hypothesis, which employs a sliding window approach, the results were similarly inconclusive. The sliding window method did not offer a significant improvement in the detection of COVID-19 patterns. The optimal value for N in this hypothesis was found to be N=8, where 2 + 27 instances of COVID-19 were correctly

classified and only 2 of these were not identified in other parts of the dataset. This indicates that the sliding window approach also suffers from false positives, with the model continuing to detect anomalies in regions outside the expected periods. Additionally, 16 instances were incorrectly classified as healthy.

For a visual representation of how a participant is flagged as COVID-19 positive, Figure 4.31 illustrates an example of a True_Covid participant. In this case, while the participant exhibits some isolated anomalies outside the COVID onset period, the required N anomalies on both hypotheses were only met within the actual onset period. As a result, the participant was flagged as COVID-positive only within the onset phase, meeting the criteria set by both hypotheses and reinforcing the model's alignment with the expected classification during the infection period.

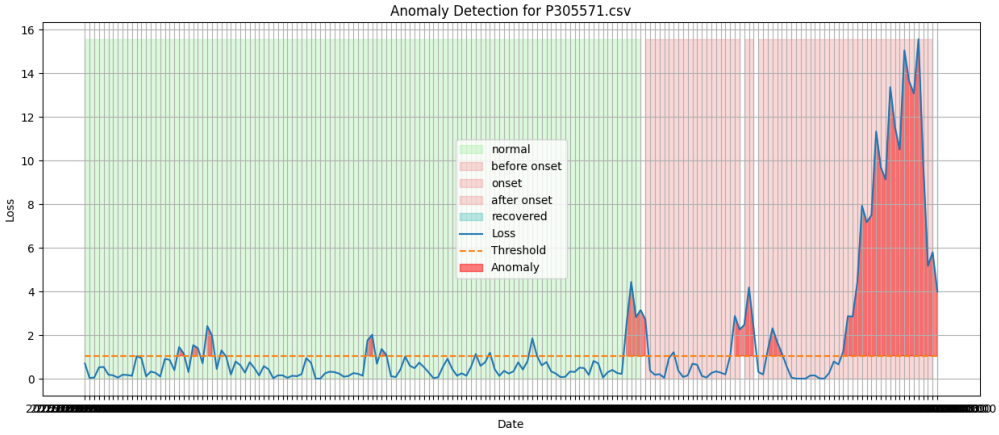


FIGURE 4.31. Detection of COVID-positive status within the correct onset period, label as a True_Covid participant.

For another representation, Figure 4.32 illustrates a case where the model's classification was less accurate. Unlike the previous example, where the COVID-positive status was flagged solely within the correct onset period, this participant was flagged as COVID-positive both within the correct timeframe and incorrectly during the recovery phase. This outcome reflects the model's tendency to detect anomalies outside the primary infection window, indicating a potential over-sensitivity to normal fluctuations in RHR data, leading to misclassifications beyond the expected period.

In Figure 4.33, it is evident that the model performs better at recognizing healthy data. A larger proportion of healthy data points is accurately classified under both hypotheses, demonstrating the model's greater proficiency in detecting healthy patterns compared to identifying COVID-19 anomalies. Of the 76 healthy participants in the Healthy dataset, 58 were accurately detected as healthy using the first hypothesis with N=7, resulting in an accuracy rate of approximately 76%. Under the second hypothesis with N=8, 61 participants were correctly identified as healthy, yielding an accuracy rate of approximately 80%.

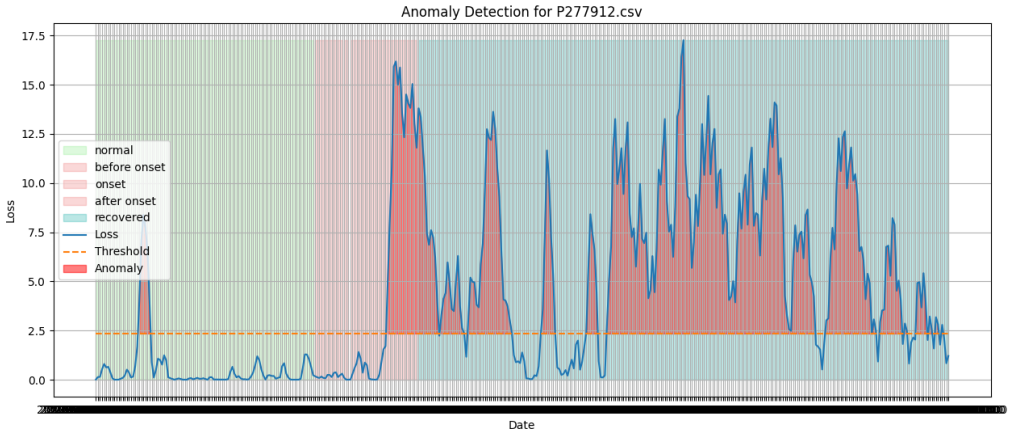


FIGURE 4.32. Detection of COVID-positive status within and outside the primary onset period, labeled as a COVID participant.

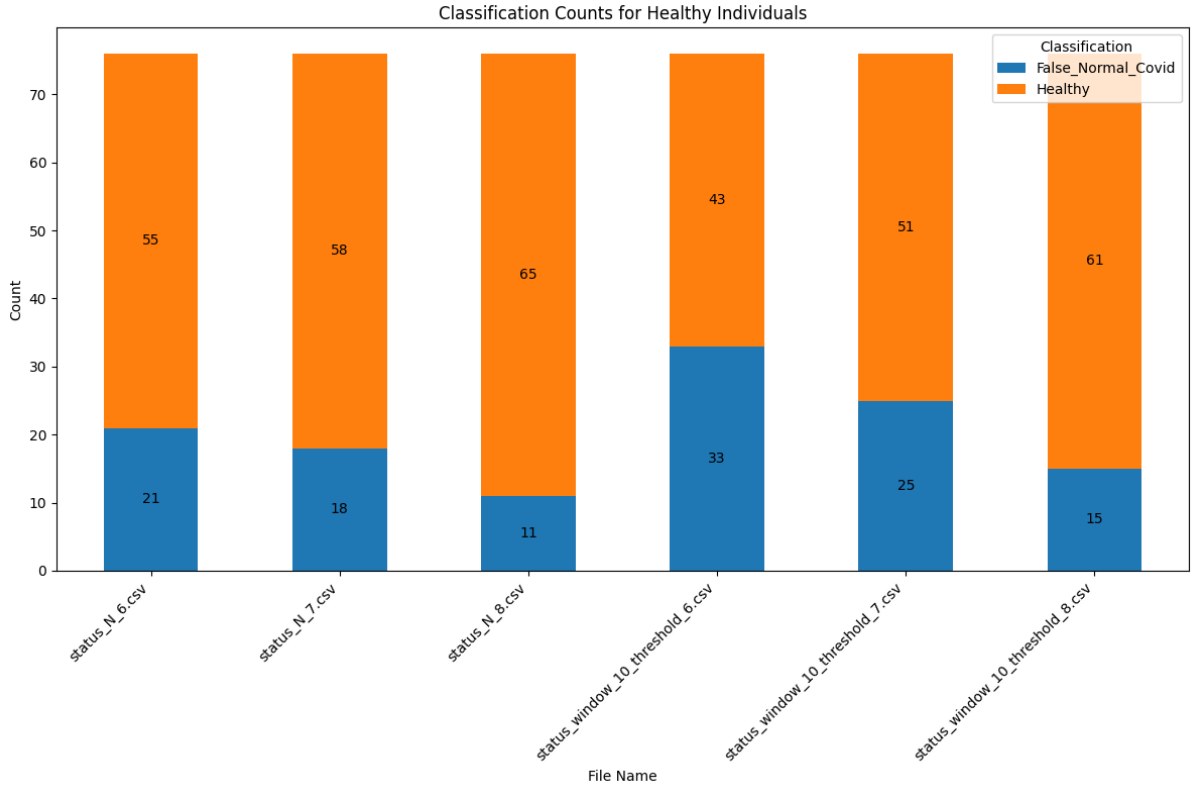


FIGURE 4.33. Classification counts for healthy data using the convolutional model, FedAvg strategy, MSE loss function and 1STD threshold.

The model shows stronger accuracy in identifying healthy data but struggles with precision in detecting COVID-19 related anomalies, often misclassifying periods before or after the actual infection as anomalous. In the Healthy dataset, accuracy rates for healthy classification reached approximately 76% and 80% under the two hypotheses, highlighting the model's proficiency with healthy participants. However, in the unhealthy participants, while some true COVID-19 instances were correctly identified, a considerable number of

false positives emerged, indicating the model's over-sensitivity to typical physiological fluctuations. This misalignment suggests that refining the model's temporal specificity and adjusting sensitivity thresholds could reduce false positives, ultimately enhancing its reliability in real-world anomaly detection scenarios.

CHAPTER 5

Conclusions

5.1. Discussion and Conclusions

This dissertation explored the integration of FL into the AIMHealth application, focusing on enhancing anomaly detection capabilities and possibly identifying patterns in physiological data for the detection of COVID-19. The study was guided by two main research questions:

(1) RQ1: How can we optimally prepare the AIMHealth application to use Federated Learning automatically?
(2) RQ2: Can we identify patterns or anomalies in the user's physiological data that allow the detection of COVID-19 using Federated Learning within the context of the AIMHealth app?

RQ1 was addressed by optimizing the AIMHealth application to support FL. The steps included the integration of wearable devices for continuous health monitoring, focusing on heart rate and step data collection via the Google Fit platform. These integrations allowed for decentralized data processing directly on user devices, enabling FL while ensuring data privacy. AIMHealth was configured to automate data collection and facilitate background processing without the need for constant user intervention, a feature needed for real-time health monitoring. This means that AIMHealth is now capable of seamlessly integrating FL into its architecture, making it well prepared for future real-time deployments.

RQ2 aimed to investigate whether patterns or anomalies related to COVID-19 could be effectively detected using FL. To address this, multiple models and techniques were applied. However, none of the approaches yielded notably strong results. The findings indicate that anomaly detection based solely on heart rate data did not achieve high accuracy for identifying COVID-19, suggesting that heart rate alone may be insufficient as a reliable metric for COVID-19 anomaly detection. This highlights the potential need to integrate additional physiological metrics for enhanced accuracy in future studies.

The model selection had the greatest impact on performance, with the convolutional autoencoder outperforming the LSTM model in terms of training time and overall efficiency, establishing it as the preferred model for this study. In contrast, choices in FL strategies, loss functions and thresholds had minimal effect on overall performance. Additionally, techniques such as overfitting, clustering and selecting the top 20 participants, though applied to enhance performance, yielded minimal improvements for the convolutional model, likely due to overfitting, data quality issues, or other factors limiting further performance gains.

For the LSTM model, while it was outperformed by the convolutional model, it did show improvements when personalization techniques like overfitting and clustering were applied. These enhancements suggest that the LSTM model may have a stronger capacity for capturing participant-specific patterns. Nevertheless, according to the Wilcoxon test, these improvements were not statistically significant. Despite these slight gains, the LSTM model did not surpass the overall performance metrics achieved by the convolutional model.

An analysis of the LSTM model revealed that individualized learning statistically outperformed both centralized and federated approaches, as confirmed by the Wilcoxon test. This suggests that individual models may capture participant-specific patterns more effectively, thereby enhancing anomaly detection accuracy. However, no statistically significant difference was observed between centralized and FL, indicating that, while centralized learning may offer a slight advantage over FL, the improvement is not statistically relevant. In contrast, for the convolutional model, no significant performance differences were found across centralized, individualized and federated approaches.

The analysis demonstrated limited success in identifying clear patterns for effective COVID-19 detection. Neither hypothesis provided robust results in accurately diagnosing the disease. While the model achieved over 75% accuracy in classifying healthy data, its precision in detecting COVID-19-related anomalies was less effective. The model frequently flagged periods around infection as anomalous, indicating an over-sensitivity to normal physiological variations. To address this, enhancing the model's temporal precision and incorporating adaptive sensitivity thresholds may reduce false positives, thereby improving reliability in real-world anomaly detection. Adapting thresholds to each participant's unique data profile and adjusting for individual fluctuations over time could help the model better differentiate between true anomalies and routine physiological changes, ultimately minimizing misclassifications.

The findings for RQ2 indicate that, although FL did not achieve high precision in detecting COVID-19 anomalies and patterns in heart rate data, its performance was comparable to that of centralized learning approaches. FL did not produce substantial improvements over centralized learning, nor did it perform notably worse. These results suggest that, for COVID-19 anomaly detection, FL may serve as a viable alternative to centralized learning, particularly when privacy, scalability and data decentralization are prioritized.

In conclusion, this dissertation explored the integration of FL in the AIMHealth application, enhancing its potential for privacy-preserving COVID-19 anomaly detection in resting heart rate data. While FL showed comparable performance to centralized learning, heart rate data alone proved insufficient for accurate COVID-19 detection. The study's findings underscore the need for a multimodal approach and further model refinement to improve reliability in real-world health monitoring scenarios.

5.2. Limitations

The dissertation encountered several limitations, primarily related to security standards, data availability and quality. The application's current security protocols were insufficient for large-scale deployment, especially regarding sensitive health data, making it non-compliant with regulations such as GDPR. This limitation prevented direct data collection from the app, as real-world testing could not proceed without addressing these security needs. Additionally, limited participant availability and the need for wearable devices posed further challenges. Consequently, an alternative dataset with wearable-derived data was used. Although this dataset provided a variety of metrics, including heart rate and steps, it contained substantial missing values that required imputation during preprocessing. While necessary to create a functional dataset, these adjustments likely introduced noise, potentially impacting the model's accuracy and limiting its ability to detect subtle COVID-19 patterns.

5.3. Future Work

In the future, efforts will focus on enhancing both the AIMHealth application and the FL approach to improve its applicability for real-world health monitoring and anomaly detection. Since heart rate data alone has shown limitations in detecting COVID-19 accurately, incorporating a multimodal health monitoring system that includes additional metrics, such as oxygen saturation, respiratory rate and body temperature, could improve detection accuracy by providing a more comprehensive view of users' health, potentially increasing sensitivity to COVID-19 anomalies.

To complement these improvements, further advances in FL are needed. Refining techniques such as adaptive thresholding that adjusts both between participants and over time for individual users could enhance anomaly detection, especially in complex, imbalanced datasets. Additionally, creating or sourcing a more robust dataset with minimal missing values and more balanced samples would help optimize model performance and enable more precise anomaly detection across diverse user profiles.

Preparing the AIMHealth application for real-world deployment will also require implementing FL in a real-time environment to validate simulation results. Expanding the app's functionality to support a broader range of health metrics will enable more comprehensive monitoring. Finally, ensuring robust data security, including updated encryption standards and GDPR compliance, is essential to protect user data as the application scales.

References

[1] Coronavirus disease (covid-19). World Health Organization. [Online]. Available: https://www.who.int/news-room/questions-and-answers/item/coronavirus-diseasecovid-19

[2] R. Filip, R. G. Puscaselu, L. Anchidin-Norocel, M. Dimian, and W. K. Savage, "Global challenges to public health care systems during the covid-19 pandemic: A review of pandemic measures and problems," *Journal of Personalized Medicine*, vol. 12, no. 8, p. 1295, 2022.

[3] G. B. Colbert, A. V. Venegas-Vera, and E. V. Lerma, "Utility of telemedicine in the COVID-19 era," *Reviews in Cardiovascular Medicine*, vol. 21, no. 4, p. 583, 2020.

[4] M. a. N. Saqib, S. Siddiqui, M. Qasim, M. Jamil, I. Rafique, U. A. Awan, H. Ahmad, and M. S. Afzal, "Effect of COVID-19 lockdown on patients with chronic diseases," *Diabetes & Metabolic Syndrome: Clinical Research and Reviews*, vol. 14, no. 6, pp. 1621–1623, 2020.

[5] K. Christensen, G. Doblhammer, R. Rau, and J. W. Vaupel, "Ageing populations: The challenges ahead," *The Lancet*, vol. 374, no. 9696, pp. 1196–1208, 2009.

[6] A. Asadzadeh and L. R. Kalankesh, "A scope of mobile health solutions in COVID-19 pandemics," *Informatics in Medicine Unlocked*, vol. 23, p. 100558, 2021.

[7] H. Abaza and M. Marschollek, "MHealth application areas and technology combinations," *Methods of Information in Medicine*, vol. 56, no. S 01, pp. e105–e122, 2017.

[8] M. Yogitha and K. Srinivas, "Using federated learning in anomaly detection and analytics on real-time streaming data of healthcare," *ACM Computing Surveys*, 2023.

[9] P. Ware, A. Shah, H. J. Ross, A. G. Logan, P. Segal, J. A. Cafazzo, K. Szacun-Shimizu, M. Resnick, T. Vattaparambil, and E. Seto, "Challenges of telemonitoring programs for complex chronic conditions: Randomized controlled trial with an embedded qualitative study," *Journal of Medical Internet Research*, vol. 24, no. 1, p. e31754, 2022.

[10] D. C. Nguyen, Q.-V. Pham, P. N. Pathirana, M. Ding, A. Seneviratne, Z. Lin, O. Dobre, and W.-J. Hwang, "Federated learning for smart healthcare: A survey," *ACM Comput. Surv.*, vol. 55, no. 3, feb 2022. [Online]. Available: https://doi.org/10.1145/3501296

[11] AIMHealth. (2021, May) Sample page. [Online]. Available: https://istar.iscte-iul.pt/aimhealth/

[12] K. Boikanyo, A. M. Zungeru, B. Sigweni, A. Yahya, and C. K. Lebekwe, "Remote patient monitoring systems: Applications, architecture, and challenges," *Scientific African*, vol. 20, p. e01638, 2023.

[13] (2023, February) Federated learning for privacy preservation in smart healthcare systems: A comprehensive survey. IEEE Journals & Magazine — IEEE Xplore. [Online]. Available: https://ieeexplore.ieee.org/document/9794622

[14] L. Yang, X. Fang, and J. Zhu, "Knowledge mapping analysis of public health emergency management research based on web of science," *Frontiers in Public Health*, vol. 10, p. 755201, 2022.

[15] B. Suruliraj and R. Orji, "Federated learning framework for mobile sensing apps in mental health," in *2022 IEEE 10th International Conference on Serious Games and Applications for Health(SeGAH)*, 2022, pp. 1–7.

[16] M. A. M. Pranto and N. Al Asad, "A comprehensive model to monitor mental health based on federated learning and deep learning," in *2021 IEEE International Conference on Signal Processing, Information, Communication Systems (SPICSCON)*, Dec 2021, pp. 18–21.

[17] N. Tabassum, M. Ahmed, N. J. Shorna, M. M. U. R. Sowad, and H. M. Z. Haque, "Depression detection through smartphone sensing: A federated learning approach," *International Journal of Interactive Mobile Technologies (iJIM)*, vol. 17, no. 01, pp. 40–56, 2023. [Online]. Available: https://doi.org/10.3991/ijim.v17i01.35131

[18] X. Xu, H. Peng, M. Z. A. Bhuiyan, Z. Hao, L. Liu, L. Sun, and L. He, "Privacy-preserving federated depression detection from multisource mobile health data," *IEEE Transactions on Industrial Informatics*, vol. 18, no. 7, pp. 4788–4797, 2022.

[19] M. A. Fauzi, B. Yang, and B. Blobel, "Comparative analysis between individual, centralized, and federated learning for smartwatch based stress detection," *Journal of Personalized Medicine*, vol. 12, no. 10, 2022. [Online]. Available: https://www.mdpi.com/2075-4426/12/10/1584

[20] Y. S. Can and C. Ersoy, "Privacy-preserving federated deep learning for wearable iot-based biomedical monitoring," *ACM Trans. Internet Technol.*, vol. 21, no. 1, jan 2021. [Online]. Available: https://doi.org/10.1145/3428152

[21] A. Almadhor, G. A. Sampedro, M. Abisado, S. Abbas, Y.-J. Kim, M. A. Khan, J. Baili, and J.-H. Cha, "Wrist-based electrodermal activity monitoring for stress detection using federated learning," *Sensors*, vol. 23, no. 8, 2023. [Online]. Available: https://www.mdpi.com/1424-8220/23/8/3984

[22] A. Nandi and F. Xhafa, "A federated learning method for real-time emotion state classification from multi-modal streaming," *Methods*, vol. 204, pp. 340–347, 2022. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S104620232200072X

[23] M. M. Qirtas, D. Pesch, E. Zafeiridi, and E. B. White, "Privacy preserving loneliness detection: A federated learning approach," in *2022 IEEE International Conference on Digital Health (ICDH)*, July 2022, pp. 157–162.

[24] S. Sakib, M. M. Fouda, Z. Md Fadlullah, K. Abualsaud, E. Yaacoub, and M. Guizani, "Asynchronous federated learning-based ecg analysis for arrhythmia detection," in *2021 IEEE International Mediterranean Conference on Communications and Networking (MeditCom)*, Sep. 2021, pp. 277–282.

[25] A. Raza, K. P. Tran, L. Koehl, and S. Li, "Anofed: Adaptive anomaly detection for digital health using transformer-based federated learning and support vector data description," *Engineering Applications of Artificial Intelligence*, vol. 121, p. 106051, 2023. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S095219762300235X

[26] ——, "Designing ecg monitoring healthcare system with federated transfer learning and explainable ai," *Knowledge-Based Systems*, vol. 236, p. 107763, 2022. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0950705121009862

[27] Y. M and K. S. Srinivas, "Using federated learning in anomaly detection and analytics on real-time streaming data of healthcare," in *Proceedings of the 2023 7th International Conference on Graphics and Signal Processing*, ser. ICGSP '23. New York, NY, USA: Association for Computing Machinery, 2023, p. 29–34. [Online]. Available: https://doi.org/10.1145/3606283.3606288

[28] S. Baghersalimi, T. Teijeiro, D. Atienza, and A. Aminifar, "Personalized real-time federated learning for epileptic seizure detection," *IEEE Journal of Biomedical and Health Informatics*, vol. 26, no. 2, pp. 898–909, 2022.

[29] J. Jorge, P. H. Barros, R. Yokoyama, D. Guidoni, H. S. Ramos, N. Fonseca, and L. Villas, "Applying federated learning in the detection of freezing of gait in parkinson's disease," in *2022 IEEE/ACM 15th International Conference on Utility and Cloud Computing (UCC)*, Dec 2022, pp. 195–200.

[30] S. Naz, K. Phan, and Y. Chen, "A comprehensive review of federated learning for covid-19 detection," *International Journal of Intelligent Systems*, vol. 37, no. 3, pp. 2371–2392, Mar 2022. [Online]. Available: https://onlinelibrary.wiley.com/doi/10.1002/int.22777

[31] T. Xia, J. Han, A. Ghosh, and C. Mascolo, "Cross-device federated learning for mobile health diagnostics: A first study on covid-19 detection," in *ICASSP 2023 - 2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, June 2023, pp. 1–5.

[32] D. R. Kandati and T. R. Gadekallu, "Genetic clustered federated learning for covid-19 detection," *Electronics*, vol. 11, no. 17, 2022. [Online]. Available: https://www.mdpi.com/2079-9292/11/17/2714

[33] Y. Gupta, Z. M. Fadlullah, and M. M. Fouda, "Toward asynchronously weight updating federated learning for ai-on-edge iot systems," in *2022 IEEE International Conference on Internet of Things and Intelligence Systems (IoTaIS)*, Nov 2022, pp. 358–364.

[34] M. Ali, F. Naeem, M. Tariq, and G. Kaddoum, "Federated learning for privacy preservation in smart healthcare systems: A comprehensive survey," *IEEE Journal of Biomedical and Health Informatics*, vol. 27, no. 2, pp. 778–789, 2023.

[35] A. Shen, L. Francisco, S. Sen, and A. Tewari, "Exploring the relationship between privacy and utility in mobile health: Algorithm development and validation via simulations of federated learning, differential privacy, and external attacks," *Journal of Medical Internet Research*, vol. 25, 2023. [Online]. Available: https://www.jmir.org/2023/1/e43664

[36] A. Rehman, S. Abbas, M. Khan, T. M. Ghazal, K. M. Adnan, and A. Mosavi, "A secure healthcare 5.0 system based on blockchain technology entangled with federated learning technique," *Computers in Biology and Medicine*, vol. 150, p. 106019, 2022. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0010482522007417

[37] Z. Lian, W. Wang, Z. Han, and C. Su, "Blockchain-based personalized federated learning for internet of medical things," *IEEE Transactions on Sustainable Computing*, vol. 8, no. 4, pp. 694–702, 2023.

[38] D. Gupta, O. Kayode, S. Bhatt, M. Gupta, and A. S. Tosun, "Hierarchical federated learning based anomaly detection using digital twins for smart healthcare," in *2021 IEEE 7th International Conference on Collaboration and Internet Computing (CIC)*, Dec 2021, pp. 16–25.

[39] P. V. Astillo, D. G. Duguma, H. Park, J. Kim, B. Kim, and I. You, "Federated intelligence of anomaly detection agent in iotmd-enabled diabetes management control system," *Future Generation Computer Systems*, vol. 128, pp. 395–405, 2022. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0167739X21004192

[40] S. Kalabakov, B. Jovanovski, D. Denkovski, V. Rakovic, B. Pfitzner, O. Konak, B. Arnrich, and H. Gjoreski, "Federated learning for activity recognition: A system level perspective," *IEEE Access*, vol. 11, pp. 64 442–64 457, 2023.

[41] A. N. Navaz, M. A. Serhani, and H. T. El Kassabi, "Federated quality profiling: A quality evaluation of patient monitoring at the edge," in *2022 International Wireless Communications and Mobile Computing (IWCMC)*, May 2022, pp. 1015–1021.

[42] D. Y. Zhang, Z. Kou, and D. Wang, "Fedsens: A federated learning approach for smart health sensing with class imbalance in resource constrained edge computing," in *IEEE INFOCOM 2021 - IEEE Conference on Computer Communications*, May 2021, pp. 1–10.

[43] M. N. Mohammed, X. Zhang, M. Valero, and Y. Xie, "Poster: Asyncfedkd: Asynchronous federated learning with knowledge distillation," in *2023 IEEE/ACM Conference on Connected Health: Applications, Systems and Engineering Technologies (CHASE)*, June 2023, pp. 207–208.

[44] A. A. Abdellatif, N. Mhaisen, A. Mohamed, A. Erbad, M. Guizani, Z. Dawy, and W. Nasreddine, "Communication-efficient hierarchical federated learning for iot heterogeneous systems with imbalanced data," *Future Generation Computer Systems*, vol. 128, pp. 406–419, 2022. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0167739X2100412X

[45] D. Moher, A. Liberati, J. Tetzlaff, and D. G. Altman, "Pre-symptomatic detection of covid-19 from smartwatch data," *Nat Biomed Eng 4*, 2020.

[46] Flutter. [Online]. Available: https://flutter.dev

[47] A. B. d. C. B. G. Vieira, "mhealth application based on artificial intelligence for decision support system," Master's thesis, ISCTE – Instituto Universitário de Lisboa, Lisbon, Nov. 2023.

[48] A. Alavi, G. Bogu, M. Wang, and et al., "Real-time alerting system for covid-19 and other stress events using wearable data," *Nature Medicine*, 2022, published: 29 November 2021, Issue Date: January 2022.

[49] Normal resting heart rate by age (chart). Forbes. [Online]. Available: https://www.forbes.com/health/wellness/normal-heart-rate-by-age

[50] What is lstm – long short term memory? geeksforgeeks. [Online]. Available: https://www.geeksforgeeks.org/deep-learning-introduction-to-long-short-term-memory/

[51] Implement convolutional autoencoder in pytorch with cuda. geeksforgeeks. [Online]. Available: https://www.geeksforgeeks.org/implement-convolutional-autoencoder-in-pytorch-with-cuda/

[52] E. McClenaghan, "The wilcoxon signed-rank test," *Technology Networks*, 2024, available at: https://www.technologynetworks.com, accessed on 12 September 2024.

Annex A

```
import keras
import numpy as np
import tensorflow as tf

def lstm_autoencoder(input_shape=(16, 1), learning_rate=0.001):
    model = keras.Sequential()
    model.add(keras.layers.RepeatVector(n=input_shape[0]))
    model.add(keras.layers.LSTM(units=64, return_sequences=True))
    model.add(keras.layers.LSTM(units=128, return_sequences=True)
       )
    model.add(keras.layers.TimeDistributed(keras.layers.Dense(
       units=input_shape[1])))
    model.compile(loss=tf.losses.MeanSquaredError(),
    optimizer=tf.optimizers.Adam(learning_rate=learning_rate),
                  metrics=[tf.metrics.MeanSquaredError()])
    model.add(
        keras.layers.TimeDistributed(keras.layers.Dense(units=
           input_shape[1]))
    )
    model.compile(
        loss=tf.losses.MeanSquaredError(),
        optimizer=tf.optimizers.Adam(learning_rate=learning_rate)
           ,
        metrics=[tf.metrics.MeanSquaredError()],
    )
    return model

def convolutional_autoencoder(input_shape=(16, 1), learning_rate
   =0.001):
    model = keras.Sequential()

    # Encoder
    model.add(keras.Input(shape=input_shape))
    model.add(keras.layers.Conv1D(filters=64, kernel_size=3,
       activation="relu", padding="same"))
    model.add(keras.layers.MaxPooling1D(pool_size=2,
```

```
31      padding="same"))
32      model.add(keras.layers.Conv1D(filters=32, kernel_size=3,
            activation="relu", padding="same"))
33      model.add(keras.layers.MaxPooling1D(pool_size=2,
34      padding="same"))
35
36      # Decoder
37      model.add(keras.layers.Conv1D(filters=32, kernel_size=3,
            activation="relu",
38      padding="same"))
39      model.add(keras.layers.UpSampling1D(size=2))
40      model.add(keras.layers.Conv1D(filters=64, kernel_size=3,
            activation="relu",
41      padding="same"))
42      model.add(keras.layers.UpSampling1D(size=2))
43      model.add(keras.layers.Conv1D(filters=1, kernel_size=3,
            activation="sigmoid",
44      padding="same"))
45
46      # Compile model
47      model.compile(
48          loss=tf.losses.MeanSquaredError(),
49          optimizer=tf.optimizers.Adam(learning_rate=learning_rate)
                ,
50          metrics=[tf.metrics.MeanSquaredError()],
51      )
52
53      return model
54
55  class MAE:
56       def generate_loss(self, inputs, outputs):
57          losses = np.mean(np.abs(outputs - inputs), axis=(1, 2))
58          return losses
59
60  class MSE:
61      def generate_loss(self, inputs, outputs):
62          losses = np.mean((outputs-inputs)**2, axis=(1, 2))
63          return losses
64
65  class Threshold1STD:
66      def calculate_threshold(self, losses):
67          value_mean = np.mean(losses)
68          value_std = np.std(losses)
```

```
        value = value_mean + value_std
        return value

class PercentileThreshold:
    def __init__(self, percentile=75):
        self.percentile = percentile

    def calculate_threshold(self, losses):
        return np.percentile(losses, self.percentile)

class VarianceThreshold:
    def __init__(self, factor=1.0):
        self.factor = factor

    def calculate_threshold(self, losses):
        value_mean = np.mean(losses)
        value_var = np.var(losses)
        return value_mean + self.factor * value_var
```