INSTITUTO SUPERIOR DE ENGENHARIA DO PORTO
MESTRADO EM ENGENHARIA DE INTELIGÊNCIA ARTIFICIAL




A Federated Learning approach for data privacy in healthcare applications

PEDRO MANUEL RIBEIRO VIEIRA
Junho de 2024

P.PORTO

A Federated Learning approach for data privacy in healthcare applications

Pedro Manuel Ribeiro Vieira

Aluno nº: 1181079

Dissertação para obtenção do Grau de

Mestre em Engenharia de Inteligência Artificial

Orientadora: Doutora Eva Catarina Gomes Maia, Investigadora Auxiliar do Instituto Superior de Engenharia do Instituto Politécnico do Porto

Co-Orientadora: Doutora Isabel Cecília Correia da Silva Praça Gomes Pereira, Professora Coordenadora do Instituto Superior de Engenharia do Instituto Politécnico do Porto

Júri:

Presidente:

Doutor António Constantino Lopes Martins, Professor Adjunto do Instituto Superior de Engenharia do Instituto Politécnico do Porto

Vogais:

Doutora Rita Paula Almeida Ribeiro, Professora Auxiliar da Faculdade de Ciências da Universidade do Porto
Doutora Eva Catarina Gomes Maia, Investigadora Auxiliar do Instituto Superior de Engenharia do Instituto Politécnico do Porto

Porto, junho 2024

Dedication

This thesis is dedicated to my cousin, Hugo Vieira, who passed away last year. His memory lives on in the hearts of those who cherish him.

I also dedicate it to the memory of Keaton Pierce, the singer from *Too Close To Touch*, whose battle with Acute Pancreatitis inspired part of this work. His legacy and music continue to inspire many.

Abstract

In healthcare, actions tend to generate a vast amount of sensitive patient data, which is useful for scientific advancements and new applications, but also presents privacy and security challenges. Artificial intelligence can significantly benefit from this data, but traditional Machine Learning (ML) techniques in collaborative environments expose it excessively. Federated Learning (FL) emerges as a solution, enabling model training without directly sharing patient information, thus reducing the risk of data exposure.

This thesis has three main goals. It aims to understand the most common FL tools in the state of the art, analyzing their advantages and disadvantages to select the most appropriate one. This is due to the need to identify tools that can be effectively applied to ensure both learning efficiency and data security, as well as applicability to the theme at hand. It also addresses the need to understand the most common FL scenarios in the healthcare domain presented in the literature, as it helps to identify best practices and specific challenges in this sector. The last goal is to suggest an effective FL approach that ensures data privacy. This goal is driven by the growing need for solutions that can ensure compliance with privacy regulations while enabling model training in a collaborative environment. Regarding the first objective, it was concluded that Flower is the most suitable tool for the purpose of this thesis. Although other tools, such as PySyft, stood out, Flower was the one that best met the needs of the work. Next, four major technical problems commonly encountered when working with FL were identified: scalability, security, the particularities of each type of FL partition, and data distribution. To deal with some of these technical challenges, techniques such as undersampling were employed. Furthermore, through this investigation, it became clear that a network of hospitals is one of the most common scenarios when it comes to FL in healthcare. A solution was finally proposed, and an FL scenario was designed with three hospitals collaborating to train a global model. First, the robustness and effectiveness of FL compared to traditional ML were analyzed, noting no significant loss in most models. Next, the performance of aggregation algorithms (FedAvg, FedAdam, FedAdagrad) was compared, with FedAvg standing out. Finally, the training time between the various models was compared. This performance analysis derived from two case studies: predicting mortality in patients with Acute Pancreatitis and predicting mortality in patients in Intensive Care Units (ICU) with various diseases. Thus, all the three proposed objectives were completely fulfilled.

Keywords: Machine Learning, Federated Learning, Flower, Acute Pancreatitis, Artificial Intelligence

Resumo

Na área dos cuidados de saúde, as ações tendem a gerar muitos dados sensíveis sobre os pacientes, úteis para avanços científicos e novas aplicações, mas que apresentam desafios de privacidade e segurança. A inteligência artificial pode beneficiar significativamente desses dados, mas técnicas tradicionais de aprendizagem automática (ML) em ambiente colaborativo expõem-nos em demasia. Desta forma, a aprendizagem federada (FL) surge como uma solução, permitindo o treino de modelos sem partilhar diretamente a informação dos pacientes, diminuindo o risco de exposição.

Esta tese tem três objetivos principais. Pretende entender quais são as ferramentas de FL mais comuns no estado da arte, analisando as suas vantagens e desvantagens de forma a selecionar a mais adequada. Isto deve-se à necessidade de identificar ferramentas que possam ser eficazmente aplicadas, de forma a garantir tanto a eficiência na aprendizagem quanto a segurança dos dados e a aplicabilidade no tema em questão. Também trata a necessidade de entender quais cenários de FL são mais comuns no domínio da saúde na literatura, visto que ajuda a identificar boas práticas e desafios específicos desse setor. Por fim, há o objetivo de sugerir uma abordagem eficaz de FL que permita proteger a privacidade dos dados. Este objetivo é motivado pela crescente necessidade de soluções que possam garantir a conformidade com regulamentações de privacidade e, ao mesmo tempo, permitir o treino de modelos em ambiente colaborativo. Relativamente ao primeiro objetivo, concluiu-se que o Flower é a ferramenta mais indicada para o propósito da tese. Embora outras ferramentas se tenham destacado, como é o caso do PySyft, foi o Flower que mais se adequou às necessidades do trabalho. Seguidamente, foi possível identificar quatro grandes problemas técnicos comumente encontrados ao trabalhar com FL: a escalabilidade, a segurança, as particularidades de cada tipo de partição de FL e a distribuição de dados. De forma a lidar com alguns destes problemas, técnicas como *undersampling* foram utilizadas. Além disso, através dessa investigação tornou-se possível perceber que uma rede de hospitais é um dos cenários mais comuns quando se trata de FL na área da saúde. Também foi proposta uma solução e desenhado um cenário de FL com três hospitais que colaboram para treinar um modelo global. Primeiramente, analisou-se a robustez e eficácia do FL em comparação ao ML tradicional, observando que não houve perda significativa na maioria dos modelos. Seguidamente, comparou-se a performance de algoritmos de agregação (FedAvg, FedAdam, FedAdagrad), com o FedAvg destacando-se. Por fim, comparou-se o tempo de treino entre os vários modelos. Esta análise de performance derivou de dois casos de estudo: previsão de mortalidade em doentes com Pancreatite Aguda e previsão de mortalidade em pacientes de Unidades de Cuidado Intensivo (ICU) com diversas doenças. Assim sendo, todos os objetivos propostos foram cumpridos.

Palavras-chave: Aprendizagem Automática, Aprendizagem Federada, Flower, Pancreatite Aguda, Inteligência Artificial

Acknowledgments

I would like to thank the people who supported me during this whole journey, especially my family. Particularly, I would like to thank my parents, Mário and Conceição, and my brother João.

I would also like to thank my supervisors, Eva and Isabel, for their crucial feedback, advice and suggestions.

Finally, I would like to thank GECAD for the opportunity to delve into this challenge.

Index

Lista of Figures

List of Tables

Acronyms and Symbols

List of Acronyms

ABIDE Autism Brain Imaging Data Exchange

AI Artificial Intelligence

ANN Artificial Neural Network

AP Acute Pancreatitis

DL Deep Learning

eICU Electronic Intensive Care Unit

FAQ Frequently Asked Questions

fMRI Functional Magnetic Resonance Imaging

FL Federated Learning

FTL Federated Transfer Learning

GAN Generative Adversarial Network

GDPR General Data Protection Regulation

GECAD Research Group on Intelligent Engineering and Computing for Advanced Innovation and Development

HCC Hepatocellular Carcinoma

HFL Horizontal Federated Learning

HIPAA Health Insurance Portability and Accountability Act

ICD International Classification of Diseases

ICU Intensive Care Unit

IID Independent and identically distributed

ISEP Instituto Superior de Engenharia do Porto

KNN K-Nearest Neighbors

LASI Laboratory of Intelligent Systems

MIMIC Medical Information Mart for Intensive Care

ML	Machine Learning
MLP	Multi-Layer Perceptron
PRISMA	Preferred Reporting Items for Systematic Reviews and Meta-Analyses
SMC	Secure Multi-Party Computation
SMOTE	Synthetic Minority Oversampling Techique
SVC	Support Vector Classifier
UN	United Nations
VFL	Vertical Federated Learning
XGB	Extreme Gradient Boosting

1 Introduction

Before delving into the work developed in this thesis, it is necessary to understand the motivations that lie beneath it. Hence, this chapter is meant to provide context and elucidate the content within this thesis. It aims to present the problem statement, define goals, and articulate the research questions that will be explored.

1.1 Context and Motivation

Healthcare applications are responsible for generating vast amounts of sensitive patient data, diagnostic images, physical and psychological information, and more [16]. The utilization of this data for research, analysis, and improvement of healthcare outcomes holds immense potential [16]. However, it also poses significant challenges, particularly concerning data privacy and security [16].

The proliferation of healthcare data presents a paradox: while it offers extremely positive insights into medical advancements [17], it also asks for strict safeguards to protect the security and privacy of individuals [18]. Patient data confidentiality, integrity, and secure handling are vital, especially if the sensitivity and personal nature of health-related information is considered [19].

Moreover, the Health Insurance Portability and Accountability Act (HIPAA) [20] and the General Data Protection Regulation (GDPR) [21] in Europe, among other regulations [36], [94], decree rigorous guidelines for healthcare data protection, requiring approaches to balance effectiveness and privacy.

The technological utilities of this high quantity of healthcare data available is also making the industry of healthcare even more aware of potential benefits, improvements and medical advances that were not so clear in the past. In other words, the industry is gradually recognizing the potential and utility of Artificial Intelligence (AI) to transform patient care, diagnostics, and operational efficiency. AI has been offering significant benefits, including fast diagnostic

accuracy [95], personalized treatment plans [96], and even predict patient outcomes [97]. However, AI in healthcare presents one major obstacle associated with privacy: the centralized storage and processing of data [98].

More recently, federated learning (FL) has arisen as a promising paradigm in this matter. It is an emerging stochastic machine learning (ML) approach designed to address the issue of data storage while maintaining rigorous data privacy measures [22], [28]. It pertains to a scenario where multiple clients, such as mobile devices, institutions, organizations, and more, collaborate with one or more central servers in decentralized ML configurations [22]. By leveraging this approach, models can learn from distributed data sources without sharing sensitive information, ensuring data privacy while enabling collective insights.

The motivation behind this thesis lies in addressing the critical need for preserving data privacy in healthcare while utilizing the collective potential of healthcare data. Specifically, this thesis aims to explore and demonstrate how FL can be effectively applied in the healthcare sector without compromising patient privacy. It also details the research and development work performed to study and apply FL in the said context. This work was conducted within the ongoing activities of the Intelligent Systems Laboratory belonging to the Research Group on Intelligent Engineering and Computing for Advanced Innovation and Development (GECAD) [12]. GECAD integrates the Associated Laboratory of Intelligent Systems (LASI), which is the first Portuguese Associated Laboratory focused on AI. LASI integrates thirteen Research Units geographically distributed throughout the country, namely: GECAD, 2Ai, ALGORITMI, CIBIT, CISTER, CISUC, CMUP, CTS, IEETA, IPC, LIACC, TEMA, and UNIDEMI [23]. GECAD operates as a research unit situated within Instituto Superior de Engenharia do Porto (ISEP), focusing on the advancement of scientific research and innovation. Its overarching mission is centered on the integration of intelligence in engineering and decision sciences through the development of innovative scientific research and pioneering practices [23].

1.2 Problem Statement

The merging of healthcare applications with the growing volume of sensitive patient data leads into a critical challenge: balancing the necessity of a comprehensive utilization of healthcare data with the vital inevitability of safeguarding data privacy and security. Despite the immense potential for insights and advancements derived from this, ensuring strict data privacy measures while effectively utilizing its collective potential remains an arduous challenge.

The sophisticated nature of healthcare data and the scope of healthcare technologies plead for different methodologies that merge both objectives of using patient's data while preserving individual privacy. AI has become increasingly essential in the healthcare sector due to its ability to analyze large datasets rapidly and accurately. AI can understand patterns and conclusions that are beyond human capability, easing early diagnosis, personalized treatment plans, and efficient management of healthcare resources. This is particularly important in handling complex and vast amounts of healthcare data, where traditional methods tend to fail. However,

the use of AI necessitates access to extensive and diverse datasets, which raises significant privacy concerns. Therefore, it is important to develop methodologies that allow the utilization of AI's full potential while ensuring the privacy and security of sensitive patient information.

Even if FL, as previously indicated, is a great solution, it presents some challenges. Integrating FL seamlessly into the healthcare domain while navigating the details of varied data sources, guaranteeing model accuracy, and addressing regulatory conformity requires innovative strategies and robust frameworks.

The primary focus of this thesis lies in exploring and evaluating frameworks that effectively exploit the potential of FL while preserving the inviolability of patient data privacy in the healthcare context. By addressing these challenges, this research aspires to demonstrate how FL can be utilized to create collaborative, robust, secure and private models. This approach seeks to advance medical research and improve healthcare outcomes without compromising individual privacy rights.

The totality of this context and necessities ends up converging into a very solid problem that will be addressed in this thesis: **How can Federated Learning be applied to data privacy in healthcare?**

1.3 Goals and Research Questions

Considering the problem mentioned in the previous section and with an addressing solution in mind, three goals were defined:

- **G1** – Investigate the state-of-the-art of federated learning tools and frameworks.
- **G2** – Investigate the state-of-the-art of federated learning applications for healthcare.
- **G3** – Propose an effective federated learning approach that ensure data privacy in healthcare.

The first goal is meant to establish a comprehensive understanding of available tools and frameworks in FL context. It intends to analyze and compare various existing FL tools and frameworks to discern their strengths, characteristics, and applicability within the context. This exploration will lay the groundwork for selecting the most suitable tools to support the development and implementation of a robust FL system for healthcare, ensuring efficiency, scalability, and compatibility with the distributed nature of healthcare data.

The second one aims to dig into the scenery of FL applications specifically within healthcare contexts. It involves examining technical challenges, successful implementations, case studies, and research papers to understand how FL techniques have been utilized in healthcare scenarios. This intends to provide understandings about the main challenges and benefits associated with applying FL in healthcare.

The third goal is to create a safe method for using healthcare data in FL. It will ensure that patient data remains private and meets legal requirements. The goal is to find a balance where healthcare data can be shared for improving AI without compromising individual privacy.

To guide the research conducted within this thesis' scope and effectively achieve the established goals, a main research question was formulated as follows: "How can Federated Learning be applied to data privacy in healthcare?", which matches the problem statement previously mentioned. In order to address this question, it was broken into two additional questions:

- **RQ1**: What are the most robust and privacy-preserving tools and frameworks for implementing federated learning?
- **RQ2**: What technical challenges exist in implementing Federated Learning for diverse healthcare datasets, and how can they be overcome?

1.4 Document Structure

To facilitate the Reading of this document, it is divided into multiple chapters, which leads into a better organization of the content present in the thesis.

Chapter 1, the current chapter, presented the goals of this thesis, along with the research questions, the problem statement and the context and motivation behind the work.

Chapter 2 is meant to provide background context about key topics that will be explored in further sections to make it easier for the reader to comprehend whenever they appear.

Chapter 3 presents the state-of-the-art, delving into the two research questions previously presented. It is divided into two main topics, one for each RQ: Tools and Frameworks, and Technical Challenges.

Chapter 4 considers the ethical considerations that should be taken into account when dealing with healthcare in AI.

Chapter 5 presents several datasets divided into different topics and ends up with the dataset chosen for the practical chapters of the thesis.

Chapter 6 is focused on proposing and presenting a solution, especially considering the third goal of this thesis.

Chapters 7 and 8 delve into the practical side of the work. The first one deals with the first case study, which is the mortality prediction of patients with Acute Pancreatitis (AP). The second one is focused on the second case study, which is the mortality prediction of patients in a UTI.

Lastly, Chapter 9 presents the conclusions of the work, highlighting the accomplished goals, limitations and future work, and final considerations.

2 Background

This chapter is meant to clarify and explain crucial terms of this thesis along with explaining what FL is. Hence, its purpose is to facilitate the understanding of the thesis' posterior content, making it more accessible.

As stated in the Introduction section, AI has been turning into an even more strong force in the healthcare domain. AI involves the simulation of human intelligence in machines, enabling them to perform tasks that typically require human cognition. Within AI, ML is a critical subset that focuses on developing algorithms that allow computers to learn from and make decisions based on data. However, traditional ML approaches often necessitate the centralization of large datasets, which poses substantial privacy and security risks, especially concerning sensitive healthcare information. To address these challenges, FL has emerged as a promising solution. FL is a decentralized form of ML that allows models to be trained across multiple devices or institutions without the need to share raw data.

2.1 Machine Learning

ML is one of the ramifications of AI. Traditionally, building an ML model involves a three-step process: training, validating, and testing. The first step aims to find patterns between the input data and the target variable. During training, the model learns from a dataset by adjusting its parameters to minimize the error in its predictions. The second step, validation, involves fine-tuning the model. Finally, the testing phase evaluates the model's performance using a subset of data to assess its accuracy and robustness.

Before these steps, pre-processing of data is commonly needed to grant the quality of the models. Pre-processing can include cleaning the data to remove noise and inconsistencies, normalizing the data to standardize the range of features, imputing new synthetic data, and

transforming the data to enhance the learning process. This step is important since the quality of the input data naturally impacts the model's performance.

ML can be divided into supervised, unsupervised, and semi-supervised. Supervised ML is characterized by the model being trained with a dataset that involves labeled data. The process of model learning is based on calculating an output over an input vector, which ends up enhancing itself after having access to the true target variable. On top of that, supervised ML tasks typically fall into two main categories: classification and regression. Classification tasks involve predicting categorical outcomes, such as determining whether a patient has a certain disease based on their medical records. Regression tasks, on the other hand, involve predicting continuous outcomes, such as estimating the progression of a disease over time. Several methods are commonly used in ML for both classification and regression tasks. In this thesis, the following classifiers are utilized: logistic regression, decision tree, random forest, support vector classifier (SVC) and multi-layer perceptron (MLP). In unsupervised learning, the input data is processed aiming to output few information about patterns in the input data. Cluster analysis and density estimation are two popular applications of unsupervised methods. Finally, semi-supervised learning is basically applying unsupervised techniques to improve supervised ML models.

2.1.1 Classifiers

In this thesis, several classifiers were utilized. In order to facilitate their comprehension in subsequent sections, this subsection is meant to present and provide a brief explanation about those classifiers.

2.1.1.1 Logistic Regression

Logistic Regression is a statistical method known for being utilized for modeling the relationship between one or more independent variables and a binary dependent variable [13]. It is commonly used for problems of binary classification, where the outcome is typically one of two possible classes [113]. It is based on the logistic function, which is responsible to attribute a probability value that can then be mapped to two possible classes. The model estimates the probability that a given input belongs to a particular category by fitting data to a logistic curve. It uses maximum likelihood estimation to find the best-fitting model, which is robust to the assumptions of the input data's distribution. The Trauma and Injury Severity Score (TRISS), which is vastly used to predict mortality in patients with injuries, was developed by Boyd et al. [114] with Logistic Regression.

2.1.1.2 Decision Tree

A Decision Tree is a non-parametric model utilized for prediction that makes use of a tree-shaped graph of decisions and their eventual consequences. It is used for both classification and

regression problems [115]. The tree format consists of nodes, which represent the features of the dataset, branches, which represent the decision rules, and leaves, representing the outcomes. The model works by splitting repeatedly the dataset based on specific criteria, such as the Gini impurity, which measures how often a randomly selected element of a set is mislabeled, or information gain (for classification), or variance reduction (for regression). This recursive partitioning continues until the algorithm determines that further splits would not improve the model [115]. Decision Trees are intuitive and easy to interpret, making them popular for applications where model transparency is crucial. As an example, Sivasree, et al. [116] utilized the Decision Tree to do a loan credibility prediction system.

2.1.1.3 Random Forest

Random Forest is a classifier that consists in a set of independent decision trees, hence the name Random Forest, trained on the same data set [117]. Each tree in the forest is trained on a different subset of the data with a random subset of features. The final prediction is obtained by the average of the results (in the case of regression) or by the majority voting (in the case of classification) of the individual trees' outputs [117]. This approach reduces the risk of overfitting, which is a common issue with single decision trees, and improves the model's overall performances and robustness. It is also known for their high performance in various applications, including fraud detection and medical diagnostics. For example, Kumar, et al. [118] utilized Random Forest with the goal to detect credit card frauds.

2.1.1.4 SVC

SVC is one of the methods derived from Support Vector Machines, the other being Support Vector Regression. SVC is a classification method that aims to find a hyperplane in a high-dimensional space that separates the data into different classes with the largest possible margin [119]. This hyperplane maximizes the distance between the nearest point of each class. SVC is particularly effective in high-dimensional spaces and is versatile due to its use of kernel functions, which allows it to solve non-linear classification problems by transforming the input space into higher dimensions. It is commonly applied in fields such as image recognition and bioinformatics. For example, Kurtulmuş et al. [120] utilized SVC in conjunction with computer vision techniques to detect corn tassels, showcasing its application in agricultural automation.

2.1.1.5 MLP

The MLP is a type of artificial neural network composed of an input layer, one or more hidden layers, and an output layer [121]. Each layer contains nodes, also known as neurons, that are completely connected to the ones in the adjacent layers. These nodes transform the input data through a series of weighted connections and activation functions, modeling complex relationships between inputs and outputs. MLPs are capable of learning non-linear mappings

and are used for both classification and regression tasks. It involves optimizing the weights of via backpropagation, which leads into a minimization of the error between the predicted and actual outputs using gradient descent. The MLP is commonly utilized in speech recognition systems, such as Zhu, et al. did [122].

2.2 Federated Learning

As mentioned, FL is an emerging and decentralized form of ML. In FL, it is aimed to train models across diverse clients, each responsible for local training, which, as the name suggests, is the training of models exclusively in each one of the clients [112]. These locally trained models are then aggregated on a central server. FL is based on an iterative process, where, after each aggregation, the updated global model is redistributed to the clients. The clients train the model locally again and send the updates back to the central server for further aggregation. This cycle can be repeated as many times as necessary [112]. Due to the local training and decentralized characteristics, it is considered a privacy-friendly approach. In addition, the client's raw data is never transmitted, which reduces the risks of data breaches and unauthorized access [22].

Moreover, there are three types of Data Partitioning in FL: Horizontal Federated Learning (HFL), Vertical Federated Learning (VFL) and Federated Transfer Learning (FTL). HFL can be defined as a FL methodology where datasets residing on various devices possess identical attributes but differ in instances [24]. Within this domain of FL, users exhibit similar attributes concerning their domain usage patterns and derived statistical information [24]. Furthermore, it is also important to state that HFL is the most common approach in the scope of healthcare in the literature [39]. VFL is pertinent in scenarios where different domains collaborate to train a global model using shared data that are not linked [25]. This methodology allows the collaboration and utilization of data across unrelated domains while preserving the confidentiality of sensitive information unique to each domain [25]. In other words, in HFL, the features are the same in every client, while, in VFL, the clients have different features. FTL employs the conventional ML-based transfer learning technique to train a new requirement on a pre-trained framework that has already undergone training on a similar dataset [26]. This way it is possible to address an entirely distinct problem [26]. The fundamental concept behind FTL revolves around the diversity in characteristics among different participants [27]. It addresses issues related to limited or inadequate data by effectively leveraging knowledge transfer while simultaneously ensuring the security [27].

FL also contemplates two types of federation: cross-device and cross-silo. The first one involves small, dispersed entities such as smartphones, wearables, and edge devices. Each of these entities typically possesses a relatively limited amount of local data [31]. Successful implementation of cross-device FL often needs the involvement of a substantial number, potentially reaching millions, of these edge devices to actively partake in the training process [31]. On the other hand, cross-silo FL involves clients that typically represent larger entities, such as companies or organizations like hospitals and banks. The number of participating entities in cross-silo FL is comparably small, ranging from two to around a hundred [31]. Notably,

each of these clients is expected to join in and contribute to the entire training process, potentially resulting in more broad and consolidated learning outcomes [31].

2.2.1 Aggregation Algorithms

As it was already mentioned, in FL the server is responsible for aggregating the models trained locally by the clients. To achieve this, aggregation algorithms, as they are typically called, are utilized [133]. In this work, three of those algorithms were applied due to their popularity and prominence in the literature. These algorithms are utilized during the model aggregation phase of the FL process, where the server combines the locally trained models from each client into a single global model. The specific roles and mechanisms of these algorithms are detailed below:

- **FedAVG**: It is widely known for being the most popular aggregation algorithm. It involves training a global model by averaging the client-trained model parameters. Each client trains its local model on its private data and sends the model parameters to the server. The server then computes the average of these parameters to update the global model. FedAVG is particularly utilized and suitable for a wide range of FL applications, especially when the data distribution is non-Identically Independently Distributed (IID) across clients [134]. Its simplicity, popularity, and effectiveness make it a benchmark algorithm in FL research and practice.

- **FedAdam**: This algorithm adapts the Adam optimizer, which is commonly used in traditional ML, for federated settings. FedAdam utilizes adaptive learning rates for quicker convergence and better handling of sparse gradients. In this approach, the server not only aggregates the client model parameters but also adjusts the learning rates for each parameter based on past gradients. This helps in speeding up the convergence of the global model, particularly in scenarios with heterogeneous datasets where the data distribution among clients varies significantly [88]. On top of that, FedAdam is used to enhance performance and efficiency in training models where standard methods like FedAvg might converge slowly.

- **FedAdagrad:** It is an adaptation of the Adagrad optimizer for FL. Similar to FedAdam, it assigns adaptive learning rates to each one of the parameters. This means that parameters with infrequent updates receive higher learning rates, benefitting from varying learning speeds among different parameters. This algorithm is particularly useful in scenarios where some parameters need more frequent updates than others, which is often the case in large-scale and complex models [93]. FedAdagrad is used to maintain robust training performance over iterations, especially in non-stationary environments where the data characteristics can change over time.

2.3 Metrics

A total of five metric values were analyzed in the practical experiments of this thesis. Therefore, to make it easier to understand what those metric values represent, this subsection is meant to provide context about all five of them. These are the employed metrics [123]:

- **Accuracy:** It is known to be the proportion of correct predictions - True Positives (TP) and True Negatives (TN) - in relation to the number of total predictions – TP, TN, False Positives (FP) and False Negatives (FN). It can be represented by the following equation: $\frac{TP+TN}{TP+TN+FP+FN}$.

- **Precision:** It is the prediction of TP in relation with the number of positive predictions – TP and FP. It shows how many positive predictions are actually true and can be represented by the following equation: $\frac{TP}{TP+FP}$.

- **Recall (or Sensitivity):** It represents the proportion between TP and the actual total positive values – TP and FN. It can be represented by this equation: $\frac{TP}{TP+FN}$.

- **F1-Score**: It is the representation of the relation between both Precision and Recall. Therefore, F1-Score makes it possible to combine both metrics in a single one and can be represented this way: $2 \times \frac{Precision \times Recall}{Precision+Recall}$.

- **Specificity:** It is the relation between the number of TN and the total number of negatives – TN and FP. It can be represented by the following equation: $\frac{TN}{TN+FP}$.

3 State-of-the-art

This chapter encompasses the comprehensive literature review conducted to thoroughly explore the formulated research questions. Subsequent sections delineate the research methodology adopted for each question, followed by the presentation and discussion of the findings.

3.1 Tools and Frameworks

3.1.1 Research Methodology

The exploration of RQ1 - What are the most robust and privacy-preserving tools and frameworks for implementing federated learning? - followed the Preferred Reporting Items for Systematic Reviews and Meta-Analyses (PRISMA) [1], a standardized guideline designed to enhance the transparency of literature reviews. Search terms were employed within reputable bibliographic databases, and specific inclusion and exclusion criteria were established to sift through located publications. However, as evaluating the titles and abstracts of these publications proved adequate to determine their eligibility, their full texts were directly assessed, rendering systematic exclusion rounds unnecessary.

After an overall analysis of the literature, a few terms related to question scope were decided, based on their prominence in this research question. In other words, it was necessary to find works that focus on FL and ML, while mentioning the utilization of tools, frameworks or platforms in the scope of healthcare. Table 1 provides an overview of the terms that were used for each scope.

Table 1. Scopes and Terms for RQ1.

Scope	Terms
Federated Learning and Machine Learning	(federated learning AND (machine learning OR deep learning))
Tools	(tool or framework or platform)
Healthcare	healthcare

It's noteworthy that the various scopes were amalgamated in a search query using AND operators. The primary search databases utilized were Science Direct [2], a comprehensive bibliographic repository encompassing scientific journals and conference proceedings provided by the internationally publisher Elsevier, Web of Science [11], a repository that summarizes scientific journals and conference papers, along with other sources, provided by Clarivate, and b-on [12], which also provides multiple scientific sources, by FCCN.

The search for terms was conducted specifically within the abstract or the keywords of the documents. As a requirement, these terms were expected to be present in at least one of these sections to qualify for inclusion in the search results. This criterion ensured a focused retrieval of documents where the specified terms were directly associated with the core context, or key descriptors outlined in the abstract or keywords. This resulted in the queries shown in **Erro! A origem da referência não foi encontrada.**, that only differ from each other because each database utilizes different syntaxes and ScienceDirect database limits the query to eight Boolean connectors.

Table 2. Queries utilized in each database for RQ1

Database	Query
Web Of Science	(AK=("federated learning" AND ("machine learning" OR "deep learning") AND ("tool" OR "framework" OR "platform") AND "healthcare")) OR (AB=("federated learning" AND ("machine learning" OR "deep learning") AND ("tool" OR "framework" OR "platform") AND "healthcare"))
ScienceDirect	("federated learning" AND ("machine learning" OR "deep learning") AND ("tool" OR "framework" OR "Platform") AND "healthcare")) – only in the “title, abstract, keywords” field
b-on	(SU("federated learning" AND ("machine learning" OR "deep learning") AND ("tool" OR "framework" OR "platform") AND "healthcare")) OR (AB("federated learning" AND ("machine learning" OR "deep learning") AND ("tool" OR "framework" OR "platform") AND "healthcare"))

During the process of selecting the best sources, some inclusion and exclusion criteria were defined. This aims to identify and select the most relevant sources for the domain in question. Since FL is an on-going area of research, the search was limited to peer-reviewed publications from 2019 onwards, in the English language. Sources were also excluded if they did not belong to Computer Science or Engineering subject area, its full text was not available, it was unable to access or to read or it was duplicated, as Table 3 shows.

Table 3. Inclusion and exclusion criteria for RQ1.

Inclusion Criteria	**Exclusion Criteria**
IC1: Peer-reviewed journal article or conference paper	EC1: The source does not belong to (Computer) Science or Engineering subject areas
IC2: Published from 2019 onwards	EC2: Full text not available
IC3: Available in the English language	EC3: Duplicated publication
	EC4: Unable to access or read the source

Initially, 288 results were found with the query referenced in **Erro! A origem da referência não foi encontrada.**, which were narrowed down to 86 after applying inclusion and exclusion criteria. Of these 86 sources, 17 were duplicated, which made 69 the number of viable results. However, after carefully reviewing the titles and abstracts, only 11 sources were deemed useful. All the other sources were not focused on the frameworks or tools, even though they mention those specific words, in the context of FL and Machine or Deep Learning. Additionally, five new papers were discovered by analyzing previous findings and examining their citations, an approach known as “Snowballing” [32], which resulted in a total of 16 approved sources, as it is shown by Figure 1.

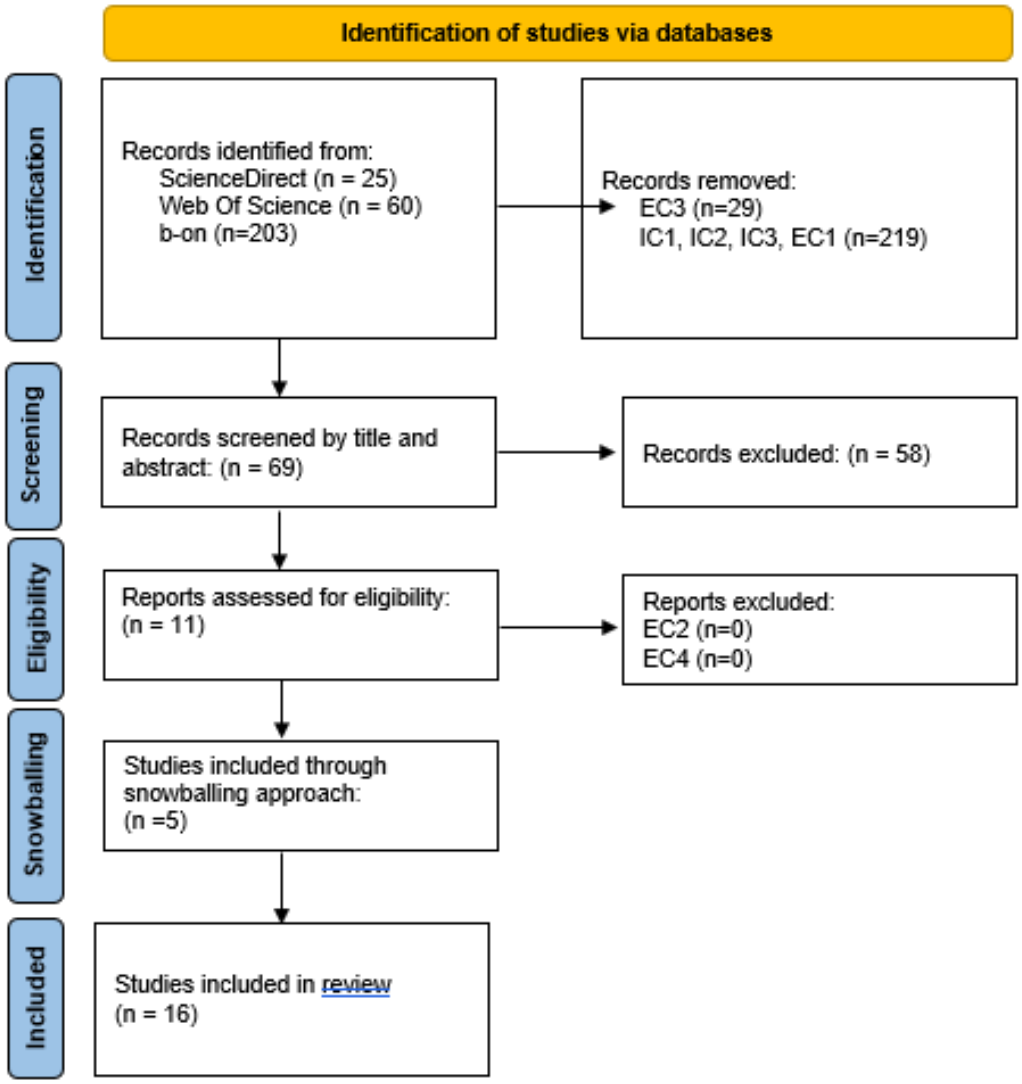


Figure 1. Sources found for RQ1

3.1.2 Findings and Discussion

Upon extensive review and analysis of multiple sources, a comprehensive identification process revealed nine prominently cited and widely utilized tools within the domain:

- TensorFlow Federated [99];
- PySyft [100];
- Paddle FL [101];
- FATE [102];
- Substra [103];

- FedML [104];
- FederatedScope [105];
- Flower [106];
- IBM Federated Learning [107].

This discovery prompted the necessity to delineate their primary disparities to facilitate a comprehensive comparison. As a result, distinct analytical categories were delineated for evaluation:

- **Open-Source – Paid**: it analyzes whether the tool is open-source or it is paid.
- **Library – Framework – App – Service**: it is meant to declare if the tool is a library, a framework, an app or a service;
- **Documentation**: it informs if it has available documentation;
- **Forum/Frequently Asked Questions (FAQ)**: it informs if the tool has an available forum or FAQ section;
- **Operating System (Linux, Windows, MacOS and Mobile)**: it signalizes which tools are compatible with each one of the most well-known Operating Systems;
- **Data Partitioning (Horizontal, Vertical and Transfer Learning)**: it gives information about the availability of HFL, VFL and TFL in the tool**;**
- **Communication Protocol (gRCP, Websockets, MPI, MQTT and ZeroMQ):** it informs about the communication protocols utilized by the tool;
- **Type Of Federation (cross-silo and cross-device)**: it is meant to put in the picture what type of federations are accepted by the tool;
- **Aggregator Node (centralized and decentralized)**: it tells which aggregator nodes are available in the tool**;**
- **Scalability**: it signalizes whether the tool is considered to have scalable proprieties;
- **Allowed Data Formats**: it informs about the allowed input data formats;
- **Privacy and Security**: it tells whether the tool is considered secure and privacy-friendly or not, based not only on the tools' documentation but also on the literature;
- **Another platform needed**: it represents the need of another platform for the tool to work.

These outlined categories aim to provide a structured outline for a meticulous comparative analysis, supporting a comprehensive understanding of the distinguishing features among these tools in the realm of FL.

To ensure a rigorous comparative analysis of the assorted tools under scrutiny, an additional strategic layer was incorporated into the research methodology. This entailed a deliberate inclusion of thorough examination encompassing the comprehensive documentation and official websites associated with each respective tool. Also, papers specifically reviewing the selected frameworks were analyzed. This proactive measure was deemed crucial, recognizing that a robust evaluation demands an expansive exploration beyond the confines of the initially procured sources.

In the initial assessment, it became evident that among the range of tools evaluated, IBM Federated Learning stood out as the sole tool operating on a paid model, while the remaining tools were characterized as open-source [29], [30]. Notably, an open-source variant of IBM Federated Learning does exist; nevertheless, it offers a reduced set of functionalities and comes with restrictions against commercial use within its licensing terms. It is also notable that IBM Federated Learning diverges from the other tools by being categorized as a service (full-version) and as a framework (open-source) [3]. Conversely, the rest of the tools fall under the classification of frameworks, encapsulating libraries within their architecture, or libraries. An exception arises with Substra, which occupies a hybrid classification, encompassing attributes of both app and library functionalities [4]. All of this is systematized in Table 4.

Table 4. Tool, Developer, Open-source – Paid and Library – Framework – App – Service

Tool	Developer	Open Source	Paid	Library	Framework	App	Service
TensorFlow Federated	Google	✓	X	✓	✓	X	X
PySyft	OpenMined	✓	X	✓	X	X	X
Paddle FI	PaddlePaddle	✓	X	✓	✓	X	X
FATE	Webank & Linux Foundation	✓	X	✓	✓	X	X
Substra	OWKIN	✓	X	✓	X	✓	X
FedML	FedML Community	✓	X	✓	X	X	X
FederatedScope	Alibaba Damo Academy	✓	X	✓	✓	X	X
Flower	Oxford, UCL and Cambridge	✓	X	✓	✓	X	X
IBM Federated Learning	IBM Watson	Limited	✓	✓	✓	X	✓

Furthermore, it is noteworthy that each of the evaluated tools offers extensive documentation along with a dedicated forum or FAQ section, as Table 5 exhibits. However, it is crucial to mention that PySyft documentation - and tutorials - is completely outdated, which can cause problems when trying to learn how to utilize the most recent versions of this tool. This provision of comprehensive documentation coupled with accessible forums or FAQs serves as a facilitative resource, significantly streamlining the process for users to comprehend and effectively utilize the functionalities and features inherent in these tools. Such support infrastructure contributes significantly to enhancing the user experience and fostering a smoother learning curve for employing these tools.

Across the examined tools, compatibility with Linux and MacOS is universal. However, PySyft uniquely stands out as the sole tool available across all platforms analyzed [5], [9]. In contrast, TensorFlow Federated, FATE, and FedML exhibit limitations, lacking compatibility with the Windows operating system [5], [9].

Table 5. Tool, Documentation, Forum/FAQ and Operating System

Tool	Documentation	Forum / FAQ	OS - Linux	OS - Windows	OS - MacOS	OS – Mobile (Android / iOs)
TensorFlow Federated	✓	✓	✓	X	✓	X
PySyft	✓ (outdated)	✓	✓	✓	✓	✓
Paddle Fl	✓	✓	✓	✓	✓	X
FATE	✓	✓	✓	X	✓	X
Substra	✓	✓	✓	✓	✓	X
FedML	✓	✓	✓	X	✓	X
FederatedScope	✓	✓	✓	✓	✓	X
Flower	✓	✓	✓	✓	✓	X
IBM Federated Learning	✓	✓	✓	✓	✓	X

When examining data partitioning capabilities among the tools, all demonstrate support for HFL. However, PySyft, PaddleFL, FATE, FedML, FederatedScope, and, more recently, Flower extend their functionalities to encompass VFL [5], [6], [75]. Remarkably, Paddle FL and FATE set themselves apart by not only enabling HFL and VFL but also facilitating FTL. This unique feature distinguishes Paddle FL and FATE as the only tools encompassing all three types of data partitioning methodologies among those evaluated [5], [6], [33]. This information can be seen in Table 6.

Table 6. Data Partitioning

Tool	Horizontal	Vertical	Transfer
TensorFlow Federated	✓	X	X
PySyft	✓	✓	X

Tool	Horizontal	Vertical	Transfer
Paddle Fl	✓	✓	✓
FATE	✓	✓	✓
Substra	✓	X	X
FedML	✓	✓	X
FederatedScope	✓	✓	X
Flower	✓	✓	X
IBM Federated Learning	✓	X	X

In terms of Communication Protocol, the predominant choice among the tools is gRPC, with most tools adopting this protocol. Notably, PySyft and Paddle FL deviate from this trend by employing alternative protocols: PySyft employs Websockets, while Paddle FL utilizes ZeroMQ. Regarding IBM Federated Learning, while their documentation specifies the utilization of gRPC and Websockets [4], it suggests support for multiple communication protocols, albeit without explicit details. Conversely, FedML demonstrates a broader spectrum by incorporating three distinct communication protocols: gRPC, MPI, and ZeroMQ [5], [6].

Table 7. Communication Protocols

Tool	gRPC	Websockets	MPI	MQTT	ZeroMQ
TensorFlow Federated	✓	X	X	X	X
PySyft	X	✓	X	X	X
Paddle Fl	X	X	X	X	✓
FATE	✓	X	X	X	X
Substra	✓	X	X	X	X
FedML	✓	X	✓	✓	X
FederatedScope	✓	X	X	X	X
Flower	✓	X	X	X	X
IBM Federated Learning	✓	✓	No Info	No Info	No Info

All evaluated tools demonstrate support for cross-silo federation [9]. However, a select few extend their capabilities to include cross-device federation [9]. Notably, PySyft, FedML, FederatedScope, Flower, and IBM Federated Learning offer functionalities for this type of federation [9], [15]. It's noteworthy that Paddle FL currently lacks this specific type of federation but plans to incorporate it in future [5], as can be seen in Table 8.

In the context of aggregator nodes among the evaluated tools, FATE stands out as the singular tool not offering a decentralized aggregator node, offering only a centralized one [5]. Conversely, TensorFlow Federated, PySyft and FedML demonstrate the provision of both centralized and decentralized aggregator nodes within their frameworks or architectures [9]. This dual offering highlights their flexibility, allowing users to choose between centralized and decentralized aggregator nodes based on their specific requirements or preferences. A summary of this content is also presented in Table 8.

In the realm of scalability, the literature raises concerns regarding PySyft's scalability, suggesting limitations in this aspect [7], as Table 8 shows. Unfortunately, specific information regarding the scalability of PaddleFL and FederatedScope is notably absent within the available sources. This absence of information signifies a lack of explicit details or assessments pertaining to the scalability attributes of PaddleFL and FederatedScope within the current literature or documentation.

Table 8. Type Of Federation, Aggregator Node and Scalability

Tool	Type Of Federation – Cross-silo	Type Of Federation – Cross-device	Aggregator Node - Centralized	Aggregator Node - Decentralized	Scalability
TensorFlow Federated	✓	X	✓	✓	✓
PySyft	✓	✓	✓	✓	X
Paddle Fl	✓	In The Future	X	✓	No Info
FATE	✓	X	✓	X	✓
Substra	✓	X	X	✓	✓
FedML	✓	✓	✓	✓	✓
FederatedScope	✓	✓	X	✓	No Info
Flower	✓	✓	X	✓	✓
IBM Federated Learning	✓	✓	X	✓	✓

Regarding the acceptance of various data formats, IBM Federated Learning distinguishes itself by asserting the capability to directly accept all types of data without the need for pre-processing [10]. In contrast, the remaining tools possess the ability to accept diverse data formats but typically require pre-processing facilitated by libraries like Pandas [108] or numPy [109] to ensure compatibility and seamless integration of different data types within their respective frameworks, as it is showed in Table 9. In addition, as it is also illustrated in Table 9, all the tools adhere to stringent standards concerning data privacy and security, ensuring the safeguarding of sensitive information throughout their utilization [8], [9], [10], [14], [15].

Ultimately, among the evaluated tools, the paid version of IBM Federated Learning stands alone in its requirement for an additional platform [10], as it can be observed in Table 9. Specifically, IBM Cloud Pak for Data is a prerequisite for leveraging this particular tool's functionalities [10].

Table 9. Data Format Acceptance, Security and Privacy Compliance and Platform needing

Tool	Any Data Format Acceptane	Security and Privacy Compliance	Does not need another platform (such as IBM Cloud Pak for Data)
TensorFlow Federated	Indirectly	✓	✓
PySyft	Indirectly	✓	✓
Paddle Fl	Indirectly	✓	✓
FATE	Indirectly	✓	✓

Tool	Any Data Format Acceptane	Security and Privacy Compliance	Does not need another platform (such as IBM Cloud Pak for Data)
Substra	Indirectly	✓	✓
FedML	Indirectly	✓	✓
FederatedScope	Indirectly	✓	✓
Flower	Indirectly	✓	✓
IBM Federated Learning	✓	✓	X

Moreover, it is important to notice that Substra notably distinguishes itself as a framework initially designed primarily for healthcare applications [8]. However, its current adaptability extends beyond its original purpose, allowing its utilization across various other domains and scopes beyond healthcare [8].

3.1.3 Conclusions

The analysis of FL tools has provided valuable perceptions into their diverse characteristics and functionalities. These conclusions aim to summarize key finding and identify the chosen tool for the proposed work.

Open-Source:

- IBM Federated Learning (full version) is the only paid tool; others are open-source.

Documentation and Operating System Support:

- All tools have extensive documentation and forums/FAQs.
- PySyft's documentation is outdated.
- Compatibility with Linux and MacOS is universal.
- PySyft is available across all platforms.
- Compatibility issues with Windows for TensorFlow Federated, FATE, and FedML.

Data Partitioning:

- All tools support HFL.
- PySyft, Paddle FL, FATE, FedML, FederatedScope, and Flower support VFL.
- Paddle FL and FATE also support FTL.

Communication Protocols:

- gRPC is predominant, with exceptions like PySyft (Websockets) and Paddle FL (ZeroMQ).
- IBM Federated Learning supports gRPC and Websockets, with unspecified additional protocols.
- FedML supports gRPC, MPI, and ZeroMQ.

Type of Federation and Aggregator Nodes:

- All tools support cross-silo federation.
- PySyft, FedML, FederatedScope, Flower, and IBM Federated Learning support cross-device federation.
- FATE lacks decentralized aggregator nodes; others offer both centralized and decentralized options.

Scalability, Privacy and Security, and Data Formats:

- Concerns about PySyft's scalability; limited information on Paddle FL and FederatedScope.
- IBM Federated Learning accepts all data formats imput.
- All tools prioritize data privacy and security.

Platform Dependency:

- IBM Federated Learning's paid version requires IBM Cloud Pak for Data.
- Other tools do not have additional platform dependencies.

After careful consideration, **PySyft, TensorFlow Federated and Flower have emerged as the best choices for the intended work,** but only one of them is truly suitable, as the following explanation indicates. The outdated documentation of **PySyft** makes it extremely difficult for new users to understand how to make proper use of this framework. The fact that the very first line of its tutorial is now unfunctional really demonstrates how difficult it is to understand how to utilize it for those who are new to it. **TensorFlow Federated** would be a great option, but the fact it is not available for Windows, as some of its dependencies are unavailable for this particular Operating System, makes it more difficult to utilize by a Windows user. Therefore, **Flower** ends up being the ideal choice. Its selection was based not only on its comprehensive open-source nature but also on its compatibility across all operating systems, except the mobile ones. Additionally, Flower's support for both cross-silo and cross-device federation types is an undeniable positive point. Furthermore, its versatility in accommodating both HFL and VFL adds to its suitability for meeting the potential needs of this project. While compliance with privacy and security standards is a feature shared by all tools, it remains noteworthy to mention that this standard played an important role in the selection process. The guarantee of compliance with privacy and security measures was a non-negotiable characteristic, as any tool lacking in this aspect would not have been considered for adoption. This emphasizes the vital importance placed on safeguarding data privacy and security throughout the tool selection process.

3.2 Technical Challenges

3.2.1 Research Methodology

The investigation into RQ2 - What technical challenges exist in implementing Federated Learning for diverse healthcare datasets, and how can they be overcome? – also adhered to the

guidelines set by the Preferred Reporting Items for Systematic Reviews and Meta-Analyses (PRISMA) [1]. As previously mentioned, this standardized framework aims to improve the transparency of literature reviews. To conduct this exploration, relevant search terms were utilized in well-regarded bibliographic databases. Specific criteria for inclusion and exclusion were established to sort through the gathered publications. However, since assessing the titles and abstracts of these publications was sufficient to determine their suitability, the documents were directly accessed, eliminating the need for systematic exclusion rounds, the same it was done for RQ1. In case the titles and abstracts were not sufficient to determine whether the works are useful or not, there would be a need to do more exclusion rounds, with more tight criteria.

Following a comprehensive analysis of the literature, specific terms related to the question scope were selected. There was a need to find works that deal with challenges in the domain of FL and ML, while focusing on the healthcare scope and mentioning the utilization of a dataset. Table 10 presents an outline of the terms employed for each scope.

Table 10. Scopes and Terms for RQ2.

Scope	**Terms**
Federated Learning and Machine Learning	(federated learning AND (machine learning OR deep learning))
Challenges	(challenges OR difficulties))
Healthcare	healthcare
Dataset	dataset

It's noteworthy that the various scopes were merged in a search query using AND operators, as in RQ1 case. Again, Web of Science, Science Direct and b-on were the databases utilized for this matter.

Once again, the terms were sought specifically within the abstract or keywords of the documents. To be considered for inclusion in the search results, it was required that these terms appear in at least one of these sections. This criterion aimed to secure a targeted retrieval of documents where the specified terms were directly linked to the primary context, or crucial descriptors highlighted in the abstract or keywords. Consequently, the queries outlined in Table 11 vary only due to the distinct syntaxes used by each database, with ScienceDirect imposing a limit of eight Boolean connectors.

Table 11. Queries utilized in each database for RQ2

Database	**Query**
Web Of Science	(AK=((federated learning AND (machine learning OR deep learning)) AND (challenges OR difficulties) AND healthcare AND dataset)) OR (AB=((federated learning AND (machine learning OR deep learning)) AND (challenges OR difficulties) AND healthcare AND dataset))
ScienceDirect	(federated learning AND (machine learning OR deep learning)) AND (challenges OR difficulties) AND healthcare AND dataset – only in the “title, abstract, keywords” field

Database	Query
b-on	(SU((federated learning AND (machine learning OR deep learning)) AND (technical challenges OR difficulties) AND healthcare AND dataset)) OR (AB((federated learning AND (machine learning OR deep learning)) AND (challenges OR difficulties) AND healthcare AND dataset))

In the endeavour to select the most suitable sources, specific inclusion and exclusion criteria were established. This was intended to pinpoint and select the most pertinent sources within the domain under scrutiny. As already mentioned, FL remains an evolving research area, thus the search was constrained to peer-reviewed publications from 2019 onwards, specifically in the English language. Sources outside the subject areas of Computer Science or Engineering were excluded, along with those lacking full-text availability, inaccessible for reading or duplicated entries as detailed in Table 12.

Table 12. Inclusion and exclusion criteria for RQ2

Inclusion Criteria	Exclusion Criteria
IC1: Peer-reviewed journal article or conference paper	EC1: The source does not belong to (Computer) Science or Engineering subject areas
IC2: Published from 2019 onwards	EC2: Full text unavailable
IC3: Available in the English language	EC3: Duplicated publication
	EC4: Unable to access or read the source

Initially, there were a total of 107 results found. However, 16 of those results were excluded due to being duplicated. After applying the inclusion and exclusion criteria, 11 more sources were removed, which means that 80 sources were left remaining. After carefully reading the titles and abstracts of these 80 sources, 14 of them were deemed useful, with one of those 14 being excluded due to being unable to access or read the source. There was a total of four sources added though a snowballing process. This means that the final number of sources accessed for the review is 17, as Figure 2 shows.

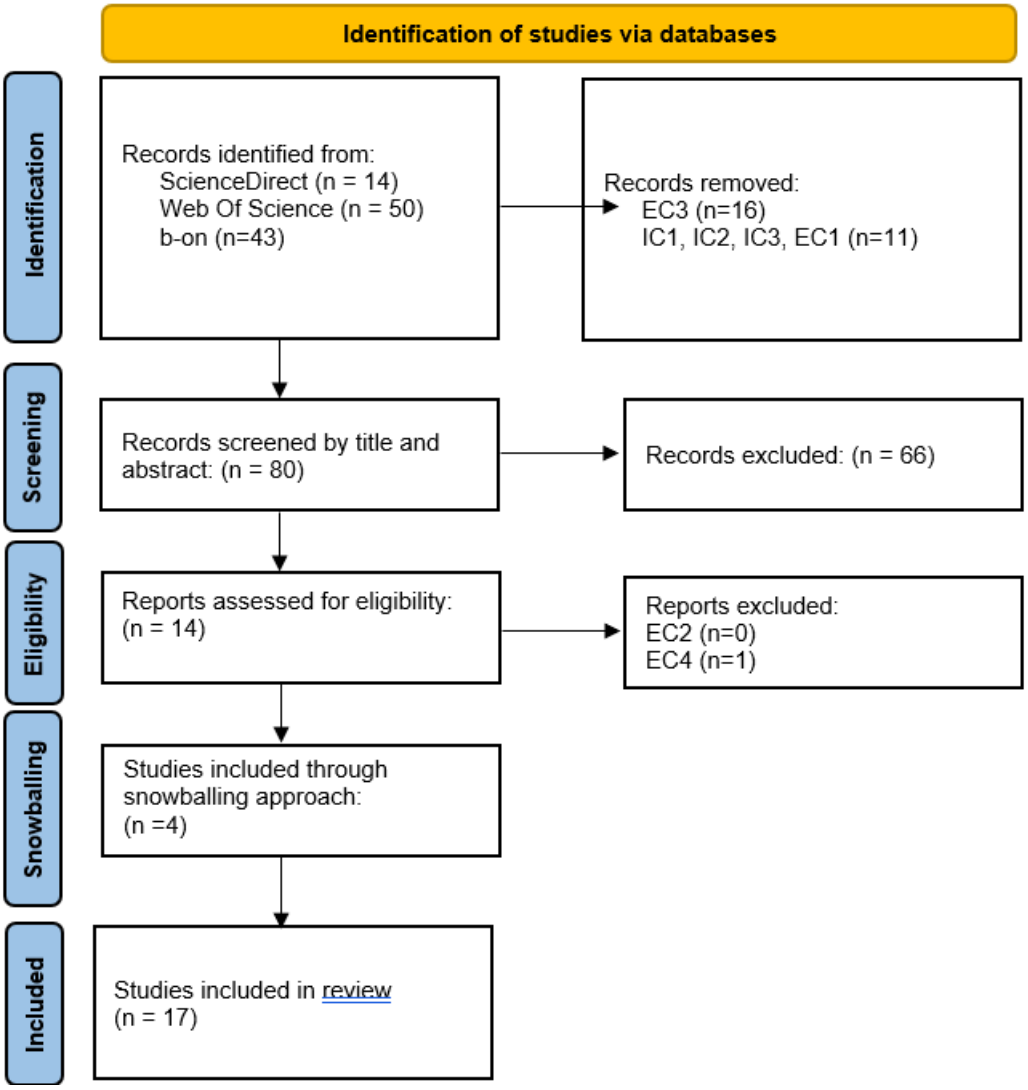


Figure 2. Sources found for RQ2

3.2.2 Findings and Discussion

After carefully reading the selected sources, some prominent challenges related to FL were identified and analyzed to produce a meaningful discussion about this matter in the context of healthcare.

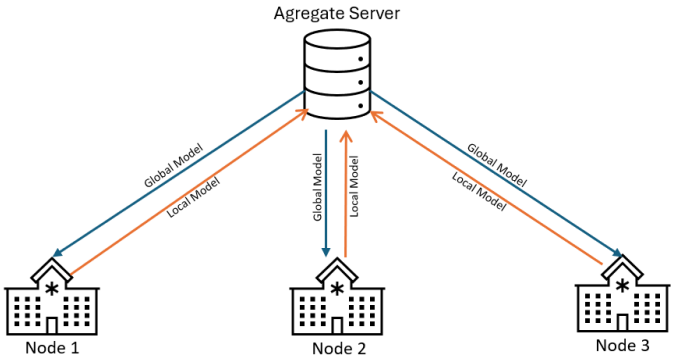
3.2.2.1 Data Distribution

The primary challenge identified in this context pertains to data distribution, particularly the statistical heterogeneity of data due to distribution variations across different clients [33], [40], [42]. This presents a challenge because the diverse data distributions among clients can adversely impact the overall performance of the global model [33]. Zhao et al. [34]

demonstrated that these divergent data distributions might significantly impact FL models due to the divergence of weights resulting from differing population distributions. In the context of FL environments, data distribution is commonly classified into two categories: IID and non-IID. Non-IID scenarios can arise due to imbalances in the quantity of data, features, or labels, which is frequently observed in healthcare applications [33]. The presence of non-identical data distributions is attributed to various factors such as different medical tool manufacturers, diverse calibration techniques, and variations in medical data acquisition methods across different healthcare institutions [33]. The non-IID characteristics among healthcare nodes in the FL environment can take on four different forms such as **quantity distribution skew, label distribution skew, feature distribution skew, and concept shift skew** [33].

Quantity distribution skew is characteristic in non-IID arises when the distribution of data instances among different nodes in the FL framework is unequal or significantly imbalanced [33], as Figure 3 demonstrates. Huang et al. [35] attempted to address this challenge by utilizing Electronic Intensive Care Unit (eICU) dataset [110], which contains data associated with over 200,000 patient stays in Intensive Care Units (ICU). Their aim was to predict patient mortality, constructing a dataset with a ratio of 5% for instances classified as death and 95% for those classified as alive. This imbalance in class distribution within the dataset simulated a real-world scenario often encountered in medical settings, enabling a focused exploration of prediction models in such imbalanced conditions. To solve this, they opted to utilize PR AUC metric, which measures the success of a model in datasets with imbalanced data, which presented good results.

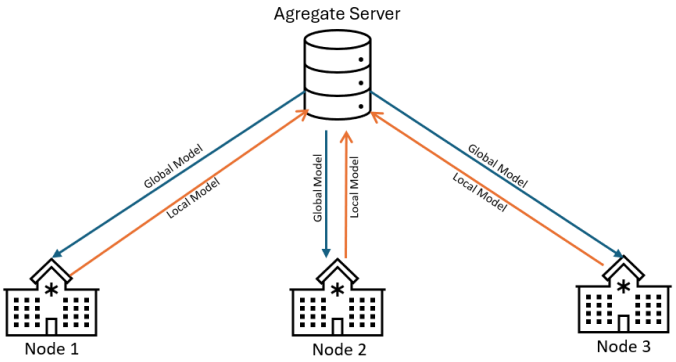
However, the most common solutions for this issue are balancing the dataset with local data augmentation, when each node generates a synthetic sample to reach the necessary balance – with techniques such as synthetic minority oversampling technique (SMOTE) and generative adversarial method (GAN) -, and server data sharing, when the aggregate server shares a portion of data to the node [33]. Nevertheless, it is crucial to refer that the second option raises model communication costs while being prone to attacks against the data privacy, even though sharing just 5% of data being able to lead to a 30% boost in accuracy score, according to Shao, et al [38]. Besides data augmentation, undersampling, which involves reducing the size of the majority class in imbalanced datasets, is also a commonly utilized approach to deal with the quantity distribution skew [111].



Node	Fever	No Fever
Node 1 - Imbalanced	3%	97%
Node 2 - Imbalanced	1%	99%
Node 3 - Balanced	45%	55%

Figure 3. Quantity distribution skew

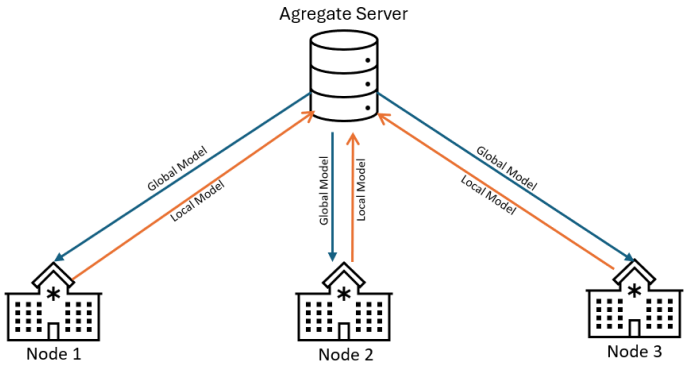
In the case of **label distribution skew** (Figure 4), it is common that the distribution of labels varies between nodes [33]. A practical example is the comparison between larger and smaller hospitals. The larger ones will most definitely have more records disease-related than the smaller ones. This can lead to a deficit on labels in one or more of the nodes related to the smaller hospitals [33]. This issue was first described in a FedAvg's experiment [33, 36]. The authors considered five different model architectures and four datasets to demonstrate the robustness of their method in a label distribution skew scenario. The found working solution in this experiment lays in dividing the samples with the same label into subsets, with each FL client being assigned to no more than two subsets with different labels [33, 36].



Node	Label A	Label B	Label C
Node 1	✓	X	✓
Node 2	X	✓	✓
Node 3	✓	X	✓

Figure 4. Label distribution skew

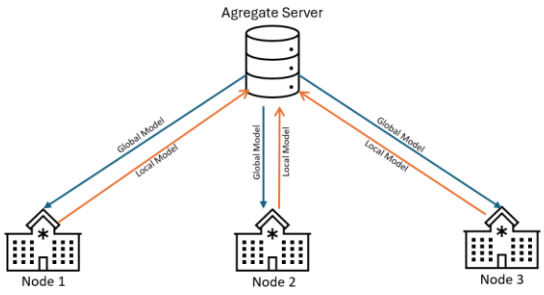
Feature distribution skew (Figure 5) is about the absence of certain features in one of the nodes. This problem, which is present in non-IID cases, results in a variation of features between each one of the nodes [33]. For example, it is possible to have a node with five specific features, while three of them are not present in another node, which can be a problem for the creation of the final model. It is also important to note that this is a common problem in the healthcare domain [33]. For instance, in a healthcare scenario, different hospitals have different types of recorded data. So, one hospital's system may include detailed patient history, while another may lack this information. In order to solve this problem, data imputation techniques can be utilized. Probability principal component analysis and multiple imputations using chained equations are two of those techniques [37].



Node	Feature 1	Feature 2	Feature 3
Node 1	✓	X	✓
Node 2	✓	✓	✓
Node 3	✓	X	X

Figure 5. Feature distribution skew

Concept shift skew (Figure 6) has a pair of different concepts: the same label with different features and the same features with different label [33]. The first one generally is related to VFL, when nodes have the same sample indexes, but have a difference in features. For example, in a scenario where all the nodes have the same features, but two of them have the label “Fever” and the other one has the label “No Fever”. The second one usually is not possible to applicate in the majority of studies concerning FL, because it needs an extremely highly controlled environment, where the labels are consistent across nodes while allowing variance in features, which is rare in a real scenario, where data heterogeneity is prominent. However, an example would be a situation where all the labels are “Fever”, but the features present in each node are different.



Node	Feature 1	Feature 2	Label
Node 1	✓	✓	Fever
Node 2	✓	✓	No Fever
Node 3	✓	✓	Fever

Same features, but different labels

Node	Feature 1	Feature 2	Label
Node 1	X	✓	Fever
Node 2	✓	X	Fever
Node 3	✓	✓	Fever

Same labels, but different features

Figure 6. Concept shift skew

3.2.2.2 Scalability

The scalability of models in FL also presents a considerable challenge [39], [42]. As the number of clients increases, so does the complexity and computational time, posing considerable challenges to any FL framework. Addressing these scalability issues requires adapting various parameters during the training process to hold varying numbers of clients [39]. This adjustment becomes necessary to manage the increased computational demand and complexities that arise with the expansion of the client base in FL setups [39]. The work done by Paul et al. [43] offers insights into addressing these challenges through their utilization of the FLAPS (Federated Learning and Privately Scaling), a clustering approach. It involves a dynamic adjustment mechanism where the computational load is efficiently distributed across various clients by clustering them based on their computational capacities and data distributions. With this strategy, Paul et al. aimed not only to enhance scalability but also to improve robustness within the FL paradigm.

3.2.2.3 Different types of Partition in Federated Learning

As mentioned earlier, FL encompasses three distinct types: HFL, VFL, and FTL. The variations in data characteristics can sometimes pose limitations to the work due to their high complexities [39].

In HFL, datasets may share the same attributes related to different patients, creating a limitation that makes it primarily compatible with IID data [39]. HFL is normally utilized in scenarios where clients have data with the same features. This uniformity in features across datasets makes it easier to aggregate and train models. The primary challenge in HFL lies in ensuring data privacy and efficient communication among clients. Furthermore, HFL is the most common approach in healthcare applications because it is well-adapted to the standardization of medical records across different institutions.
In VFL, data originates from different hospitals and institutes but pertains to the same patients, leading to differences in features and dataset sizes [39]. For example, one hospital might have lab results for a patient, while another might have x-ray files for the same patient. VFL facilitates these institutions to collaborate without needing to share raw data, thereby preserving patient privacy. The challenge here is to align the features from different sources and manage the varying dataset sized. In addition, VFL is rarely used in healthcare compared to HFL due to its need for more complex data management and integration processes, which can be technically challenging and may demand a huge amount of resources [39].

In FTL, there are disparities in data sizes, distinct features, and variations in patients [39]. FTL is designed to handle scenarios where there is little overlap in features or samples among datasets. For instance, a new hospital may have a unique set of medical tests that other hospitals do not have. FTL leverages transfer learning techniques to transfer knowledge from one domain to another, facilitating collaboration among institutions with heterogeneous data. However, exploring the entire spectrum of complexities within FTL is challenging, and only a few

researchers have delved into its intricate characteristics. The intrinsic nature of FTL, which includes significant variations in data types and sizes, makes it hard to understand and implement effectively, thus limiting its widespread adoption in the healthcare domain [39].

In FTL, there are disparities in data sizes, distinct features, and variations in patients [39]. It leverages transfer learning techniques to transfer knowledge from one domain to another, facilitating collaboration among institutions with heterogeneous data. However, exploring the entire spectrum of complexities within FTL is challenging, and only a few researchers have delved into its intricate characteristics, making it hard to understand the challenges that its intrinsic nature may offer [39].

3.2.2.4 Privacy and security

While FL enables the training of shared models by sharing data in a collaborative learning environment, also aiming to be more secure, it still introduces privacy and security challenges, particularly in the presence of malicious devices [40]. For instance, malicious actors could misuse model parameters and the shared model to learn sensitive information [40].
Privacy-related information can also be inferred from shared weights without direct access to the underlying data [40]. To mitigate the risk of privacy leakage from the shared model, various privacy-preserving techniques can be employed. These include cryptographic approaches and differential privacy [41]. So, leveraging these techniques helps safeguard sensitive information and ensures a more robust and secure FL framework in the face of potential adversarial threats.

The fundamental principle of privacy control in FL dictates that data should never leave the local environment. Instead, the global model server receives updates from local models. However, these local updates are susceptible to privacy attacks if not adequately protected. To address this vulnerability, key methods of safeguarding information in the FL process include [44]:

- **Global Differential Privacy**: This approach introduces noise or perturbation to the aggregated model updates, ensuring that individual influences from each client are not detectible [45].

- **Model Encryption**: Encrypting the model parameters or updates before transmission ensures that the information remains confidential during the communication process [46].

- **Secure Multi-Party Computation (SMC)**: SMC involves distributed computation, which means that multiple parties collaboratively compute over their inputs while keeping the privacy of those inputs [47]. In the context of FL, it permits the global model to be updated without revealing individual details of local modes.

- **Blockchain**: A proposed solution for storing and sharing the reputation of each FL edge node during the FL training process involves the utilization of blockchain technology, as discussed by Kang et al [56]. In this approach, the decentralized and transparent nature

of blockchain is leveraged to maintain a secure and immutable record of the reputation of individual FL edge nodes [56], [57].

3.2.3 Conclusions

In conclusion, the challenges within FL are diverse, encompassing issues related to data distribution, scalability, different types of FL, privacy and security, and ethical considerations in healthcare.

Data Distribution Challenges:

- The primary challenge lies in the statistical heterogeneity of data due to distribution variations among different clients, impacting the overall performance of global models.
- Non-Identically Distributed (non-IID) scenarios, including quantity distribution skew, label distribution skew, feature distribution skew, and concept shift skew, can significantly affect FL outcomes.

Scalability Challenges: As the number of clients increases, the complexity and computational time escalate, posing significant challenges to FL frameworks. The adjustment of parameters during the training process becomes crucial to manage these challenges.

Different Types of Partition in FL: HFL, VFL, and FTL present complexities, with HFL being the most common approach in healthcare literature. Exploring the intricate characteristics of FTL remains challenging, with limited research in this area.

Privacy and Security Challenges: FL, while aiming for secure collaborative learning, introduces privacy concerns, especially in the presence of malicious devices. Privacy-preserving techniques like global differential privacy, model encryption, and secure multi-party computation are crucial for safeguarding sensitive information.

Therefore, to develop the work present within this thesis, it was decided to adopt some strategies to mitigate these challenges, such as proceeding to balance the dataset, in order to avoid the quantity distribution skew. Furthermore, utilizing data from a single dataset will also prevent label, feature, and concept shift skews. Moreover, this extensive analysis also made it clear that the vast majority of FL sets in the healthcare domain are based on healthcare networks, with different institutions, such as hospitals, collaborating between each other to create a robust model.

4 Ethical Considerations

When talking about ethical considerations in healthcare, they have been around for a long time, with Hippocrates protecting the notion of medical ethics [48]. However, when talking about healthcare associated with FL and ML, there's a need to go further. There is a consensus about the importance of developing standards for ML made by professional societies such as the American College of Radiology [49].

The World Health Organization has recently issued guidelines, developed collaboratively by industry experts, academics, and public sector officials. These guidelines place a strong emphasis on safeguarding human autonomy, promoting equity, ensuring transparency, and fostering sustainability. This initiative reflects a broader trend within the United Nations (UN) to encourage mindfulness and ethical considerations in the deployment of ML [50]. The UN also delved into the tie between AI, healthcare, and ethics, with the goal of establishing a dialogue on a global level concerning the challenges that are getting more prominent due to the development of AI [51].

The emergence of potential biases, particularly stemming from the disproportionate representation of minority groups, remains a significant ethical challenge in current ML systems, commonly labeled as "Algorithmic Discrimination" [52]. An illustrative example is the criticism faced by Google's facial recognition algorithm in 2015 for inaccurately classifying black individuals as apes. The response to this issue involved a rather crude resolution: preventing the algorithm from classifying gorillas altogether [53].

More recently, the data dynamics that emerged from the COVID-19 outbreak introduced a series of obstacles that presented a challenge to the efforts to establish balanced and representative datasets [55]. The tendency to generate health data silos creates a funneling effect, where electronic health records from patients who have contracted COVID-19 may disproportionately represent subpopulations with non-random access to specific hospitals in affluent neighborhoods, consisting in an ethical issue that can generate some kind of

discrimination against poorer people [55]. This problem arises because the resources required to ensure satisfactory dataset quality and integrity may be limited to digitally mature hospitals, which disproportionately serve a privileged segment of the population while excluding others [55]. When electronic health records from these contexts contribute to the composition of training data, concerns about discriminatory effects emerge [55].

Nowadays, ML continues to present ethical challenges in healthcare, such as melanoma detection algorithms predominantly trained on light-skinned individuals. Even though they are less likely to develop melanoma, they are likely to die from it [54]. These instances emphasize the constant problem of not taking in account the diversity of people that inhabit the whole world when training algorithms and underscore the importance of not ignoring these ethical issues and addressing such biases for equitable and effective ML applications, especially in critical domains like healthcare.

FL represents an excelling path to address several ethical considerations, especially in the healthcare domain. Firstly, it facilitates the development of ethically compliant models by decentralizing the training process. Unlike traditional centralized approaches, FL allows training to happen locally on individual clients, preserving data privacy and confidentiality. FL intrinsically reduces some risks of bias by not aggregating raw data in a central location, which can help mitigate the potential for biased data handling. Despite that, the concern with racial bias is still very present in the core of this work's development. Therefore, apart from the nature of FL acting as a solution, it was also decided to not utilize any racial-related data with the intention to completely prevent this bias to happen.

5 Datasets

During the elaboration of the state-of-the-art chapter, it became evident that the selection of appropriate datasets is crucial in the context of AI and FL. Datasets play a pivot role in training and testing models, directly influencing their accuracy and robustness. The quality and characteristics of a dataset can impact the reliability of the model's predictions. Recognizing the critical importance of datasets, it was decided to analyze some of the prominent datasets identified during the state-of-the-art writing process. The ones present in this section stood out due to their relevance, frequency of usage in the read documents, and their alignment with the goals of this study. To facilitate the understanding and comparison between datasets, this section was divided into thematic subsections.

5.1 Covid-19 Datasets

Two major datasets in the scope of Covid-19 were found during the State-of-the-art section elaboration**: Covid-19 Image Data Collection** [59] and **COVIDx** [66]. The first one contains hundreds of frontal view X-rays, claiming to be the largest public resource for COVID-19 image and prognostic data. It also contains various respiratory diseases, with COVID-19 standing out as the most common one. The second one consists of 13,975 chest X-ray images distributed across 13,870 patient cases. As far as the authors are aware, it includes the largest number of publicly available positive cases for COVID-19. The authors merged several COVID-19 datasets, including the previously mentioned **COVID-19 Image Data Collection.** Even though **COVIDx** also utilizes **COVID-19 Image Data Collection** pictures, their labels are different, with *Covid-19 (SARs-CoV-2)* being the only one in common. To illustrate the comparison between the labels of both datasets, Table 13 shows all the labels present in each one of the datasets.

Table 13. Covid-19 Image Data Collection and COVIDx labels

Labels	Covid-19 Image Data Collection	COVIDx
Covid-19 (SARs-CoV-2)	✓	✓
SARS (SARSr-CoV-1)	✓	X
MERS-CoV	✓	X
Varicella	✓	X
Influenza	✓	X
Herpes	✓	X
***Streptococcus* spp.**	✓	X
***Klebsiella* spp.**	✓	X
Escherichia coli	✓	X
***Nocardia* spp.**	✓	X
***Mycoplasma* spp.**	✓	X
***Legionella* spp.**	✓	X
Unknown Bacteria	✓	X
Chlamydophila spp.	✓	X
Staphylococcus spp.	✓	X
Pneumocystis spp.	✓	X
Asperggillosis spp.	✓	X
Lipoid	✓	X
Aspiration	✓	X
Unknown	✓	X
Pneumonia	X	✓
Normal Images	X	✓

In terms of FL, these datasets can be applied to Covid-19 diagnosis and detection, such as the one exemplified by Liu, et al. [40]. They conducted experiments with FL on Covid-19 chest X-ray images. The authors utilized four models: MobileNet, ResNet18, MoblieNet, and COVID-Net, concluding that ResNeXt and ResNet18 are the best ones for Covid-19 identification. ResNet18 presented the best performance both in training with FL and without FL, while ResNeXt had the best performance in images with COVID-19 labels only. Moreover, the results with FL were always better than the ones that didn't utilized FL.

5.2 Autism

Autism Brain Imaging Data Exchange (ABIDE) I [58] is a collaborative initiative that engaged 17 international sites in sharing resting state functional magnetic resonance imaging, anatomical, and phenotypic datasets for the advantage of the scientific community. This collaborative effort led to the compilation of 1112 datasets, containing 539 individuals with autism spectrum disorder and 573 typical controls, covering ages from 7 to 64 years with an average age of 14.7 years across groups. It was made publicly available in August 2012 and lastly updated in 2016. Following to HIPAA guidelines and the 1000 Functional Connectomes Project / INDI protocols, all datasets within ABIDE I have undergone anonymization, ensuring the exclusion of protected

health information. It contains 73 different features, that go from simple variables, such as "Body Mass Index", to more complex ones, like "Social Responsiveness Scale Social Communication Subscore Raw Total". Li et al. [125] utilized FL to do a multi-site Functional Magnetic Resonance Imaging (fMRI) analysis with the information contained in the dataset, for example. They suggested a FL approach, where they considered the systemic differences of fMRI distributions from different sites. The, they proposed two domain adaptation methods in their FL formulation: Mixture of Experts and Adversarial Domain Alignment. They concluded that theses domain adaptions positively contribute to the performance and that FL can be advantageous when utilizing multi-site data without sharing it.

5.3 Emotion Recognition

Facial Emotion Recognition (FER) 2013 [60] is a dataset that encompasses grayscale images of faces, each standardized to 48x48 pixels, that were automated registered in order to have consistent centering and framing, preserving a uniform composition across all of them. The dataset labelling allowed to categorize each face based on the expressed emotion, with seven emotions (angry, disgust, fear, happy, sad, surprise and neutral). The training set has 28,709 examples and the public test set consists of 3,589 examples. The work published by Shome et al. [126] addresses the concept of facial expression recognition using a FL approach with few examples, known as few-shot learning. They proposed a novel framework called FedAffect, which combines the principles of FL and few-shot learning to effectively recognize facial expressions across different devices while preserving user privacy. The results showed that FedAffect was able to achieve competitive performance in facial expression recognition tasks compared to traditional ML, even surpassing them. In addition, their framework showed resistance to data distribution variance across different clients.

5.4 Tumors

Brain Tumor Image Segmentation Benchmark (BraTS) 2017 and 2018 [61] is a dataset which includes 285 MRI scans focusing on brain tumors, offering four distinct Magnetic Resonance Imaging modalities for each scan—T1, T1ce, T2, and Flair. It also incorporates masks outlining brain tumors, containing labels for ED (Edema), ET (Enhancing Tumor), and NET/NCR (Non-Enhancing Tumor/Non-Contrast-Enhancing Tumor). The tumor's evaluation is made around three key tasks: WT (Whole Tumor), TC (Tumor Core), and ET (Enhancing Tumor) segmentation. As an example of its utilization, one of the tasks present in Nalawade et al.'s [127] work involved utilizing FL to train a robust network with BraTS dataset. In their study, the goal was utilizing FL to develop a model for brain tumor segmentation from MRI scans. They proposed a framework that combines FL with transformers, which are widely known for being able to deal with tasks that involve complex medical imaging. They concluded that the FL approach with transformers was able to achieve comparable performance with traditional ML.

Breast Cancer Diagnostic Dataset (BCD) [70] is another tumor related dataset. It is focused on Breast Cancer Diagnostic, having 201 instances belonging to one class (no-recurrence-events) and 85 instances going to another class (recurrence-events). Each instance is characterized by nine attributes, which include a combination of linear and nominal features. These are the features that this dataset contains: class, age, menopause, tumor-size, inv-nodes, node-caps, deg-malig, breast, breast-quad, irradiate. Therefore, it does not include image features. presents an approach to building ML models using clinical data that partially overlapping between different institutions, while preserving data privacy. Park et al.'s [128] work, which utilized BCD, presented an approach to build models using clinical data that is partially overlapping between different institutions, with the help of FL to ensure data privacy. In the end, the obtained results showed that the global model generally outperformed the respective local model, in situations of clinical data overlapping. They also concluded that with less overlap, there is not necessarily less improvement in model performance between local and global models.

5.5 Human Activity Recognition

MobiAct [62] is a dataset that includes data retrieved from smartphones when people were performing different tasks combined with a range of fall. The dataset includes four specific types of falls to provide coverage of scenarios that might be encountered in real life. These falls are:

- Forward-Trip: The subject falls forward, simulating a trip over an obstacle.
- Forward-Lunge: The subject lunges forward and falls, mimicking scenarios where balance is lost during a forward movement.
- Backward-Sit: The subject falls backward, similar to missing a seat while attempting to sit down.
- Sideward-Step: The subject falls sideways, representing a loss of balance during a lateral movement.

The dataset also includes 12 different Activities of Daily Living, and scenarios mimicking daily living. It is made of 66 subjects, having over 3200 trials. The activities were selected based on three criteria: Activities where fall-like behavior is common, activities that are rapid and sudden that can be similar to falls, and activities that are common in the everyday life, such as walking, standing, ascending, and descending stairs. However, this particular dataset needs to be requested from the creators for access and utilization. It also presents an innovative approach to human activity recognition using semi-supervised and personalized FL. Bettini et al.'s work [129] presented an approach to human activity recognition using semi-supervised and personalized FL with the help of MobiAct dataset. They proposed a method named FedHAR for Human Activity Recognition that combined semi-supervised learning with FL. The results showed that the purposed framework showed comparable performance to fully supervised training.

The **Human Activity Recognition Using Smartphones (HAR)** [63] dataset is another example of a human activity related dataset. It is derived from data collected from 30 subjects doing six distinctive activities: walking, walking upstairs, walking downstairs, sitting, standing, and laying. Furthermore, it is noteworthy that it involves inertial sensor data, collected through a smartphone carried by the subjects. Using the mentioned dataset, Ouyang et al.'s [130] presented ClusterFL, a similarity-aware FL system that aims to provide high model accuracy and low communication overhead for Human Activity Recognition applications. They concluded that their work surpassed some learning paradigms, such as FedAvg and FTL, and being comparable to traditional ML at times. In addition, they were able to reduce more than 50% communication latency, even though it led to a minor loss of accuracy.

5.6 Other Medical Information

Medical Information Mart for Intensive Care (MIMIC) IV [65], [71], [81] is a dataset that contains information about patients staying in critical care units at a tertiary care hospital. The dataset has an extensive collection of medical data, including vital signs, medications, laboratory measurements, observations, notes charted by care providers, fluid balance, procedure codes, diagnostic codes, imaging reports, hospital length of stay, survival data, and more. However, it is worth noting that this dataset requires permission to be utilized. The access to this dataset is restrict and needs previous acceptance by the authors, which demands being a credentialed user, completing the required training and signing the data use agreement. Several works utilized this dataset, including the one by Wang, et al [140]. They utilized this dataset to analyze how personalization affects model fairness across different patient demographics. Their framework aimed to ensure that models trained on federated data maintained high accuracy while being fair and unbiased across various subgroups. They used methods such as FedAvg to address this challenge. Their findings demonstrated that personalized FL is able to achieve fairer results in comparison with standalone FL. Despite of personalization enhancing fairness for more biased hospitals, it also worsens the fairness of less biased hospitals.

Kvasir [67] dataset is a collection of medical images annotated and verified by experienced endoscopists, providing valuable data for various applications in the gastrointestinal tract domain. It contains images ranging from 720x576 to 1920x1072 pixels. The work presented by Yang et al. [131] focused on creating an algorithm named "Federated Learning on Medical Datasets using Partial Networks (FLOP)" in which only a partial model is shared between clients and server. This work ended up applying the developed algorithm to the Kvasir dataset, among others, including the previously mentioned **COVIDx**, to obtain experimental results. The experimental results enlightened that the FLOP algorithm was able to achieve similar performance to more common FL approaches. Additionally, this approach led to a reduction in the communication costs, enhancing data privacy.

Indian Liver Patient Dataset (ILPD) [68] is another dataset that contains medical information. The instances in this dataset characterize Indian patients, specifically 416 diagnosed with liver disease and 167 without liver disease. Besides an indication about having or not having a liver disease, called “selector”, the dataset contains 10 more features per patient: age, gender, total bilirubin, direct bilirubin, total proteins, albumin, A/G ratio, SGPT, SGOT and alkphos. It is worth mentioning that the imbalance in ILPD is not only present in the “selector” feature, but also in the gender feature, because of the 583 patient records, 441 are male, and 142 are female. Matschinske et al.’s [132] presented a platform to apply FL in the biomedicine field, named FeatureCloud. To test the developed platform, the authors utilized the ILPD to do “classification analyses”. The results showed that the FL approach implemented on the FeatureCloud platform achieved performance comparable to traditional ML methods. The study also enlightened the huge potential to applicate FL in biomedicine.

Finally, **WESAD (Wearable Stress and Affect Detection)** [64] is a dataset for wearable stress and affect detection research. It encompasses data from 15 subjects participating in a stress-affect lab study while wearing physiological and motion sensors, having 63.000.000 instances. It included the following measuring modalities: blood volume pulse, electrocardiogram, electrodermal activity, electromyogram, respiration, body temperature, and three-axis acceleration. Alhmador et al.’s [69] work aimed to classify wearable-based electrodermal activities while preserving privacy. To do that, the authors opted for FL, due to its intrinsic privacy-preserving nature, utilizing **WESAD** for experimentation. The study concluded that the technology makes it possible to analyse data accurately and utilize this data to measure various aspects in human lives, such as stress levels. In addition, their global model achieved an accuracy of 0,868. Moreover, it was concluded that by monitoring data such as sleep patterns and physical activity, it is possible to help people to manage stress efficiently and detect diseases early.

5.7 Data Partition for the Datasets

As it was previously mentioned, HFL happens when “datasets residing on various devices possess identical attributes but differ in instances” and VFL “is pertinent in scenarios where different domains collaborate to train a global model using shared data that are not linked”. Considering that, it is natural that the literature has more cases of HFL when dealing with each dataset alone [74]. Most of them split the datasets in a way that all the clients have the same features, while having different instances [74]. The author of this thesis’s strongly agrees that it is the most natural approach for all the datasets. This opinion is also supported by the fact that, in HFL, the global model update at a server involves an aggregation of the local models, with each client updating its own model using its individual data [74]. In opposition, in VFL, the global model is formed by concatenating local models, that are coupled by the loss function. Updating a client's local model in VFL demands information about the models of other clients [74]. Naturally, this dependence between models is a major issue in terms of privacy protection and communication efficiency [74]. However, it is the most usual approach for a multi-dataset

approach, as the dataset most likely will not have the same features. It is still worth noting that this can be done in a single dataset too, which would imply dividing the dataset between clients, giving different features to all of them. Overall, VFL is less common in literature, but it is still possible to find it in works such as [72], [73] and [74].

5.8 Chapter Remarks

There are several datasets available across several scopes in the healthcare domain. Each one of the datasets has its own characteristics and can be useful for a certain purpose. However, after careful consideration, it was opted to utilize the **Medical Information Mart for Intensive Care (MIMIC) IV** dataset in the practical work of this thesis. This dataset is notable for its extensive features and collection of medical data, which are highly remarkable advantages for the work in question. In addition, its high quantity and multitude of data available makes the dataset more versatile.

6 Proposed Solution

This chapter describes the proposed solution, defining the FL scenario utilized, along with detailing how the third goal is meant to be achieved. It also introduces the two case studies defined for this purpose.

6.1 Implementation

As stated in the Tools and Frameworks subsection, Flower was the chosen framework for the practical work of this thesis. With Flowers' help, it was possible to build a solution that contemplates two main actors: server and clients. As it is possible to verify in Figure 7, the server starts by initializing the global model.

Once the global model is initialized, it is evenly distributed to all the clients. This distribution ensures that each client starts with the same model parameters, providing a uniform starting point for local training. Each client then proceeds to train the model locally on its own dataset. This local training allows the model to learn from diverse data sources, leveraging the unique data available to each client.

After completing the local training, the clients send their updated model parameters back to the server. The server then aggregates the received models, combining the individual updates into a single, unified global model. This aggregation process is crucial as it integrates the knowledge gained from all clients, leading to a more robust and generalized model.

This entire process occurs iteratively over a number of rounds that is predefined. In each round, the global model is redistributed to the clients, trained locally, and the results are aggregated by the server. This iterative loop continues until a final global model is obtained.

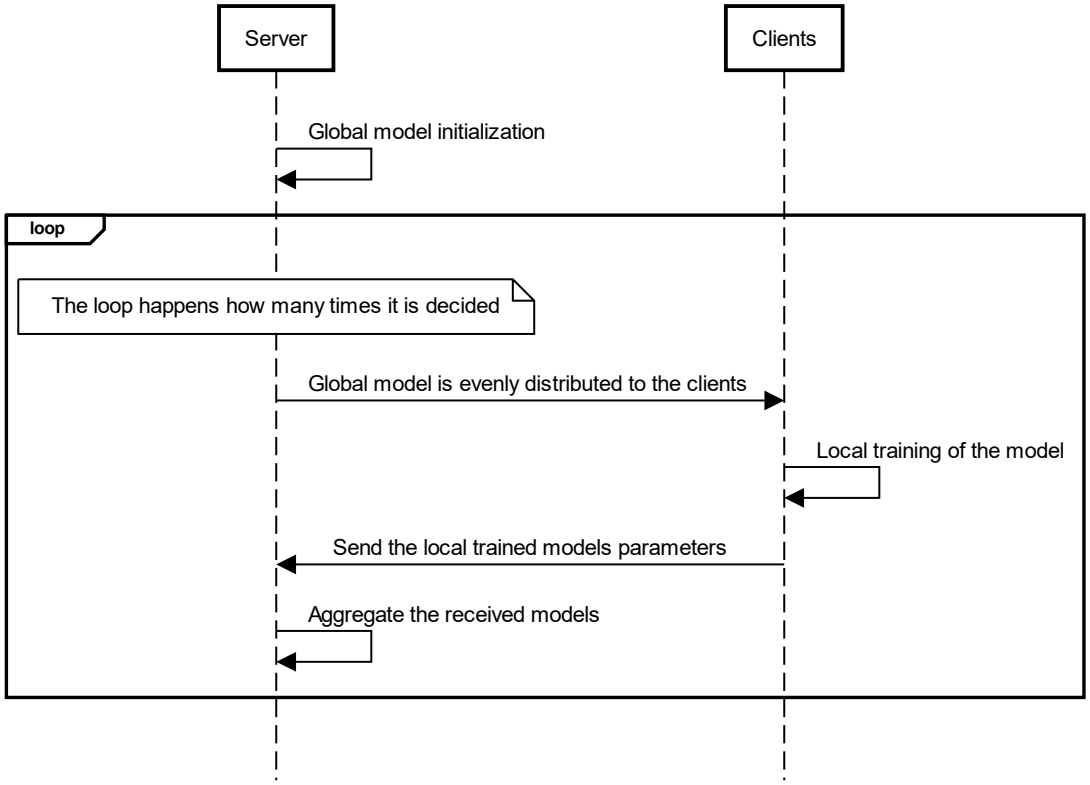


Figure 7. High-level Sequence Diagram

In a low-level view, illustrated in Figure 8, the execution of the FL process begins with the *set_initial_params()* function, initiated by the Server and directed to the Strategy component. This step initializes the model parameters, preparing them for the subsequent training rounds. Following this, the Server invokes the *fit_round()* method on the Strategy, signaling the start of a new training round. The Strategy then generates the training instructions (*fit_instructions*), which are dispatched to all the available clients through the *fit()* method. Upon receiving the instructions, each client sets the model parameters using the *set_model_params(parameters)* method. This ensures that the local models on each client are synchronized with the initialized or updated parameters provided by the Server. The clients then proceed to train their models locally. Once the local training is completed, the clients retrieve the updated mode. The trained model parameters from all clients are then sent back to the Server. The Server aggregates these parameters, combining the individual contributions from each client to form a new global model. This aggregation step also includes the evaluation of training metrics, providing insights into the performance of the aggregated model. Subsequently, the Server configures the evaluation process on the Strategy, which prepares the evaluation instructions. These instructions are then sent to the clients. Each client sets the evaluation parameters using the *set_model_parameters(parameters)* method and performs local testing of the model by invoking the test() method. The clients then return the evaluation metrics back to the Server. The Server aggregates these evaluation results, compiling the performance metrics from all clients. Finally, it worth mentioning this process is iteratively repeated for a predefined number of rounds (num_rounds).

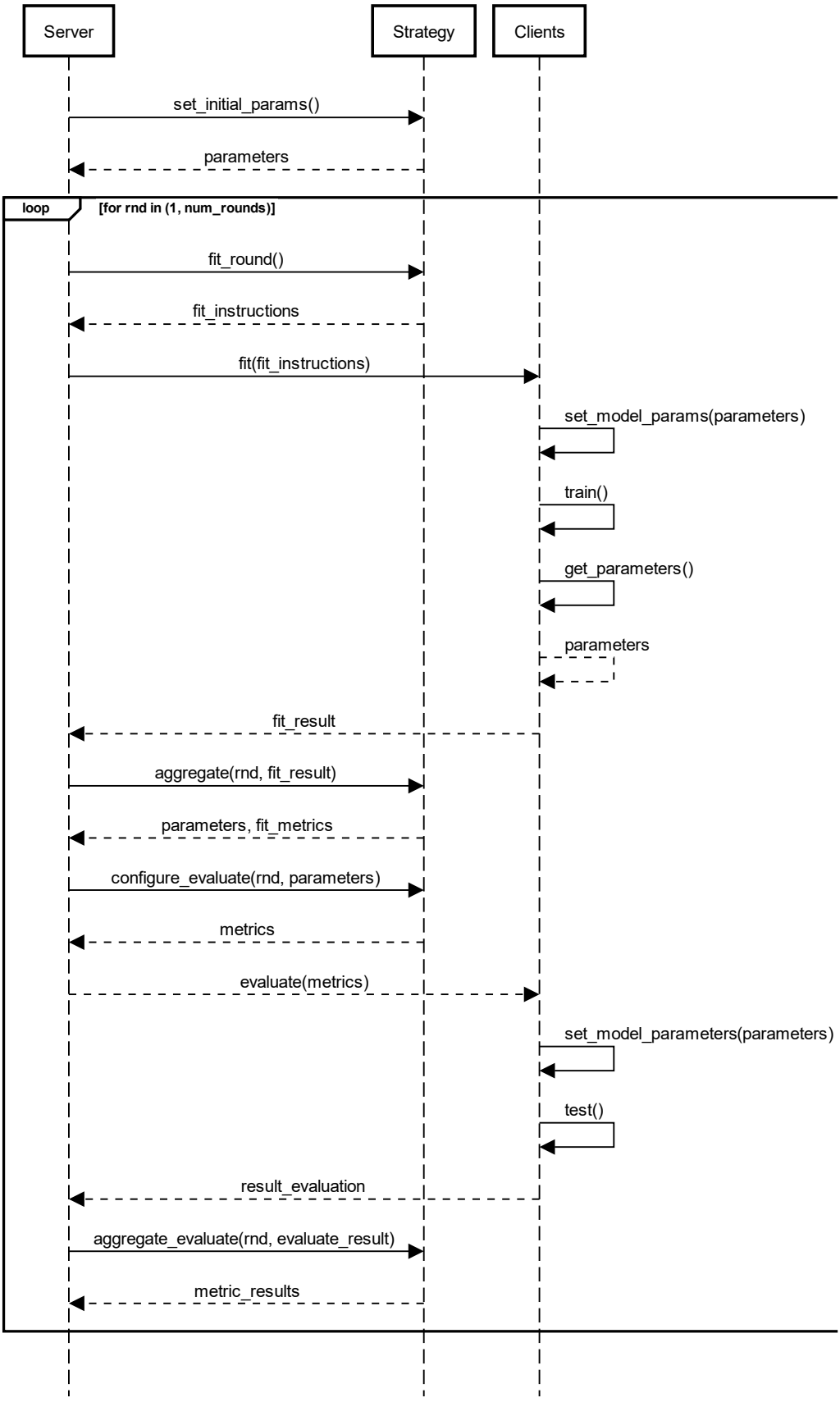


Figure 8. Low-level Sequence Diagram

As illustrated in Figure 9, the FL architecture constructed using the Flower framework is made of different components for the server and clients. On the server side, three aggregation algorithms are represented: FedAvg, FedAdam, and FedAdagrad. These algorithms, which are utilized one at a time, provide different methods for aggregating the model updates from the clients. Pivotal to the server's operations is the FL Loop, which coordinates the iterative process of distributing the global model to the clients and aggregating their locally trained models, according to the selected aggregation algorithm. Communication between the server and clients is facilitated by the gRPC server, which manages the transmission of global model parameters to the clients and the reception of updated parameters from them. On the client side, each identical client comprises three main components: the gRPC client, the Flower client, and the local data. The gRPC client handles the communication with the server's gRPC server, ensuring the smooth exchange of model parameters. The Flower client, implemented in Python, is responsible for integrating the local ML environment with the FL process. It receives the global model parameters from the server, applies them to the local model, trains the model using the local data, and sends the updated model parameters back to the server.

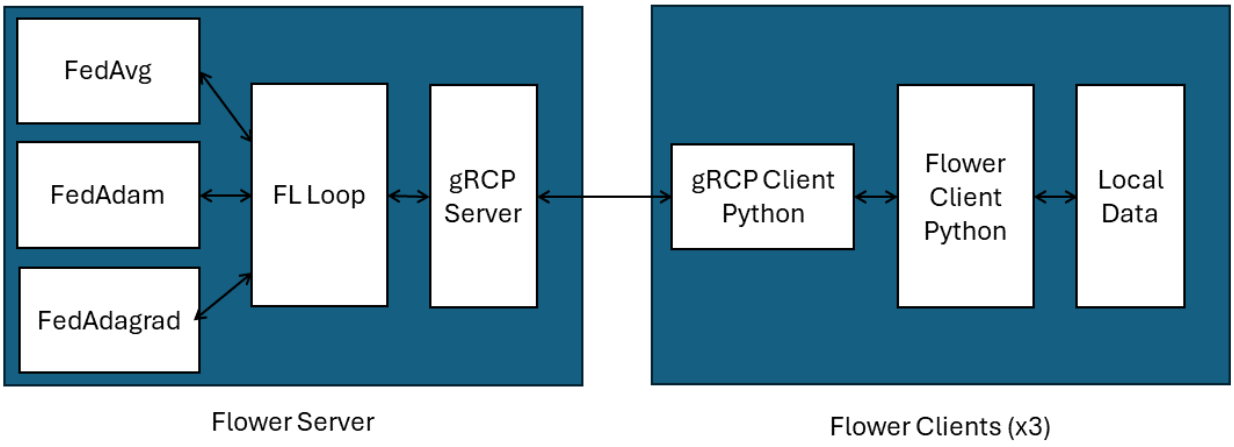


Figure 9. FL Architecture

6.2 Federated Learning Scenario

To accomplish the third goal – "propose an effective federated learning approach that ensure data privacy in healthcare" – it was also necessary to decide how to evaluate the different alternatives offered by FL. Hence, the next step was defining the exact scenario that should be addressed. According to what was concluded for the second objective, a healthcare network set is one of the most popular approaches when dealing with FL in the healthcare domain. Therefore, it was opted to trail this path, having a network of hospitals collaborating between each other.

The envisioned scenario reflects a network of three hospitals that collaborate to develop a predictive model for patient mortality. This set leverages FL to ensure data privacy while keeping the predictive accuracy of the model. Each hospital acts as an independent client,

locally training the model on its own patient data. The locally trained models are then sent to a central server, where they are aggregated to form a single, unified model. This scenario is illustrated in Figure 10 and serves as the framework for both case studies.

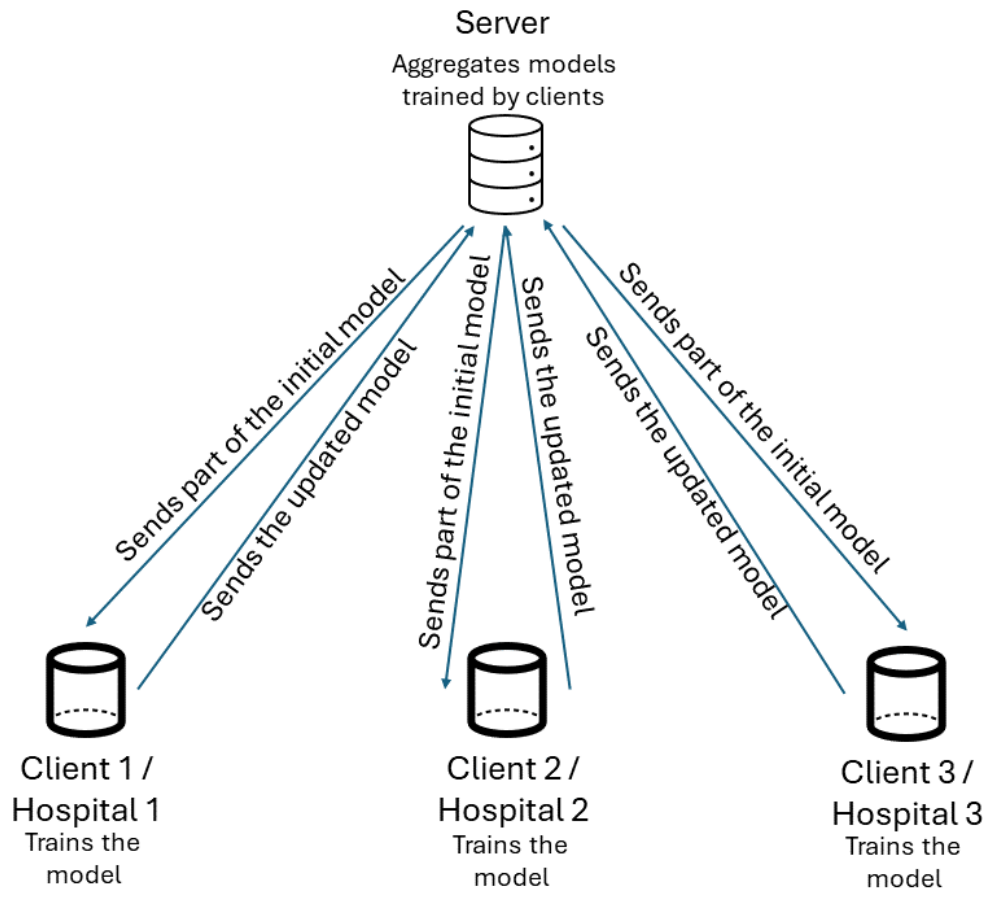


Figure 10. Mortality Prediction - Hospital Network Scenario

To provide a comprehensive understanding of how this FL setup operates, the following detailed explanation breaks down each component and their actions and interactions within the system:

- **Hospitals as Independent Clients**: Each of the three hospitals in the network acts as a client that behaves independently in the setup. These clients are responsible for the local training of the model using their respective datasets, which normally show variations.

- **Local Model Training**: Each client trains the model on its own data. Hence, the local training ensures that sensitive patient data never leaves the hospital premises, significantly reducing the risk of data breaches.

- **Communication between Server and Client**: The communication between the clients and the central server is essential in FL. In this setup, the communication is typically conducted over secure channels to ensure data integrity and confidentiality. The clients send their

locally trained model parameters to the central server. This is achieved through secure gRPC protocols, which Flower supports and facilitates.

- **Central Aggregation Server**: The central server plays a crucial role in the FL set. Its function involves aggregating the locally trained models received from all the clients to create a global model. It is here that the selected aggregation algorithm will be utilized. Here, model parameters are averaged to produce a new global model. Moreover, this aggregated model is sent back to the clients for further training, iterating through multiple rounds.

To achieve this set, Flower Framework was utilized to create both server and clients, in a simulated scenario. A previously pre-processed dataset was randomly distributed between the three clients. The clients trained the model locally and then sent into to the server, which aggregated those models.

The code that represents this set is present in Code Snippet 1 and Code Snippet 2, which are the server and the client, respectively. Each client deals with a section of the dataset, ensuring that sensitive patient information is kept within the confines of each institution. To ensure fairness and consistency in comparison, the Flower framework was used to generate and deploy the same type of models across all clients, ensuring the use of the same classifier. This methodology assures that the evaluation is purely based on the effectiveness of the models developed using both approaches. One characteristic worth stating is the fact that the "tau" value for FedAdam and FedAdagrad was not the default one. According to Reddi, et. al [86], its optimal value is 10^{-3} and not the 10^{-9} originally present in the Flower Framework. So, this value was changed accordingly. Moreover, the data was distributed evenly among the three clients, with Flower acting as a coordinator. Six federated rounds, representing iterations of the training process, were complete. The number of rounds was determined because no significant improvements were observed after the sixth. Furthermore, it is worth noting that the data distribution in each client was split 70/30 for training and testing. Furthermore, the efficacy of the model in predicting outcomes for patients must be evaluated using a variety of indicators to ensure a thorough grasp of the model's capabilities [80]. As a result, both ML and FL models were evaluated using the performance metrics mentioned in Background: accuracy, precision, F1-Score, specificity, and recall (sensitivity).

```
MIN_CLIENTS = 3

def fit_round(server_round: int) -> Dict:
    """Send round number to client."""
    return {"server_round": server_round}

def get_evaluate_fn(model: MLPClassifier, X_test, y_test):
    def evaluate(server_round, parameters: fl.common.NDArrays, config):
        utils.set_model_params(model, parameters)
        y_pred = model.predict(X_test)
        loss = log_loss(y_test, model.predict_proba(X_test))
        accuracy = model.score(X_test, y_test)
        precision = precision_score(y_test, y_pred)
        recall = recall_score(y_test, y_pred)
        f1 = f1_score(y_test, y_pred)
        #y_pred_proba = model.predict_proba(X_test)[:, 1]
        #auc = roc_auc_score(y_test, y_pred_proba)
        tn, fp, fn, tp = confusion_matrix(y_test, y_pred).ravel()
```

```
        specificity = tn / (tn + fp)

        return loss, {"accuracy": accuracy, "precision": precision, "recall":
recall, "f1": f1, "specificity": specificity}
    return evaluate

# Load data
…

# Split data into train and test sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3,
stratify=y,                                                    random_state=42)

model = MLPClassifier()
model.fit(X_train, y_train)

evaluate_fn = get_evaluate_fn(model, X_test, y_test)

if __name__ == "__main__":
    utils.set_initial_params(model)
    strategy = fl.server.strategy.FedAvg(
        min_available_clients=MIN_CLIENTS,
        evaluate_fn=evaluate_fn,
        on_fit_config_fn=fit_round,
        initial_parameters=
ndarrays_to_parameters(utils.get_model_parameters(model))
    )
    fl.server.start_server(
        strategy=strategy,
        config=fl.server.ServerConfig(num_rounds=6)
    )
```

Code Snippet 1. Flower FL server code

```
if __name__ == "__main__":
    parser = argparse.ArgumentParser(description="Flower")
    parser.add_argument("--partition-id", type=int, required=True,
                        help="Specifies the artificial data partition")
    args = parser.parse_args()
    partition_id = args.partition_id

# Load the partition data
…
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3,
stratify=y, random_state=42)

model = MLPClassifier()
model.fit(X_train, y_train)

utils.set_initial_params(model)

class CustomClient(fl.client.NumPyClient):
    def get_parameters(self, config):
```

```
            return utils.get_model_parameters(model)

        def fit(self, parameters, config):
            utils.set_model_params(model, parameters)
            with warnings.catch_warnings():
                warnings.simplefilter("ignore")
                model.fit(X_train, y_train)
            print(f"Training finished for round {config['server_round']}")
            return utils.get_model_parameters(model), len(X_train), {}

        def evaluate(self, parameters, config):
            utils.set_model_params(model, parameters)
            y_pred = model.predict(X_test)
            loss = log_loss(y_test, model.predict_proba(X_test))
            accuracy = model.score(X_test, y_test)
            precision = precision_score(y_test, y_pred)
            recall = recall_score(y_test, y_pred)
            f1 = f1_score(y_test, y_pred)

            tn, fp, fn, tp = confusion_matrix(y_test, y_pred).ravel()
            specificity = tn / (tn + fp)

            return loss, len(X_test), {"accuracy": accuracy, "precision": precision,
            "recall": recall, "f1": f1, "specificity": specificity}

    fl.client.start_client(server_address="localhost:8080",
    client=CustomClient().to_client())
```

Code Snippet 2- Flower FL client code

In addition, as described in Implementation, the FL algorithm followed a systematic process, which is described below:

- **Random Initialization**: the global model was initialized by the server;
- **Local Training**: each client trains a subset of the dataset locally;
- **Model Upload**: the clients upload the local models to the server
- **Model Aggregation**: the server aggregates the received local models to create a new global model. This step involved combining the parameters of local models using one of the aggregation algorithms previously mentioned: FedAVG, FedAdam, or FedAdagrad;
- **Iteration**: This process was repeated for a total of six rounds.

On top of that, there was a special attention to the privacy and ethical aspects of the work, as the goal is to propose privacy-friendly FL approach. Healthcare networks normally consist of multiple hospitals that can benefit from sharing knowledge and resources. However, patient data is highly sensitive and subject to strict privacy regulations, such as the GDPR in Europe, as it was mentioned in Ethical Considerations. Therefore, traditional approaches can lead into

significant risks and are often impractical. On the other hand, this approach aligns with the ethical and legal conditions of keeping patient confidentiality and privacy while being able to share the models between the different clients. This happens because it is not the data that is sent between the clients and the server. Therefore, it is not possible for the collaborative set to leak patient data. Moreover, the MIMIC-IV dataset pays special attention to the clients' privacy, making it impossible to identify the patients present in the study.

Finally, there was an attention to the findings related to RQ2, which highlighted the technical challenges that can be found in FL. Specifically, the data distribution challenges were highly regarded. The chosen dataset presented a quantity distribution skew, which is common in healthcare datasets. To deal with this, undersampling and oversampling techniques were tested. After carefully analysis, undersampling ended up being the most successful approach, being selected for the whole practical work.

6.3 Models analysis and evaluation

Training the models with FL was only one of the steps. It was also needed to train them with a traditional ML approach for comparative purposes, along with analyzing and evaluating the obtained results. These tasks are directly aligned with the practical objectives of this thesis. The aim was to evaluate the performance of FL in healthcare settings from multiple dimensions, which could only could only be done with a fair comparison.

- **Impact on Metric Values**: Aimed to investigate whether the adoption of a FL approach results in any degradation of performance metrics compared to traditional ML. This analysis is crucial to ensure that data privacy enhancements do not come at the cost of model accuracy and reliability.

- **Comparison of FL Aggregation Algorithms**: Englobes the evaluation of three of the most prominent FL aggregation algorithms present in Flower—FedAvg, FedAdam, and FedAdagrad. By comparing these algorithms, it would be possible to identify which one delivers the greatest results in terms of model performance, robustness, and speed. On the other hand, it would also make it possible to understand if there are some classifiers that show clear underperformances with certain algorithms.

- **Time and Resource Efficiency**: Aims to understand the computational time requirements of different FL approaches which is essential for practical implementation. Therefore, an analysis of the time consumption of each algorithm was crucial to provide insights into their feasibility for real-world healthcare applications, where time-sensitive decision-making is often critical. Moreover, saving time also means saving energy, which leads not only into a reduction of costs but also into the safeguarding of the environment, which are two important topics in the modern world.

- **Dataset Size and Model Efficiency**: The size of the dataset plays a significant role in the efficiency of FL models. Therefore, it was a goal to slightly understand how the size of the dataset affects performance and scalability of FL approaches.

The ML models implemented were trained according to what is shown in Code Snippet 3. The same classifiers were utilized for both FL and traditional ML, with the same data split percentages, as previously mentioned.

```
# Divide between training and test
def ds_split(dataset, dependent_var, split):
    x = dataset.drop(dependent_var, axis=1)
    y = dataset[dependent_var]
    x_train, x_test, y_train, y_test = train_test_split(x, y, train_size=split,
random_state=1)

    select = SelectKBest(k="all")
    selected_features = select.fit(x_train, y_train)
    indices_selected = selected_features.get_support(indices=True)
    colnames_selected = [x.columns[i] for i in indices_selected]

    x_train_selected = x_train[colnames_selected]
    x_test_selected = x_test[colnames_selected]

    return x_train_selected, x_test_selected, y_train, y_test

# Carregamento do conjunto de dados
datasets = {
    'processed': ds_split(undersampled_df, 'hospital_expire_flag', 0.7)
}

# Definition of the classifiers
algorithms = {
    'LogisticRegression': {
        'constructor': LogisticRegression(solver='newton-cg'),
        'predict': lambda m, x_test: m.predict(x_test),
        'predict_prob': lambda m, x_test: m.predict_proba(x_test)[:, 1],
    },
    'LinearRegression': {
        'constructor': LinearRegression(),
        'predict': lambda m, x_test: m.predict(x_test),
        'predict_prob': None,
    },
    'DecisionTreeClassifier': {
        'constructor': DecisionTreeClassifier(),
        'predict': lambda m, x_test: m.predict(x_test),
        'predict_prob': None,
    },
    'RandomForestClassifier': {
        'constructor': RandomForestClassifier(n_estimators=100),
        'predict': lambda m, x_test: m.predict(x_test),
        'predict_prob': None,
    },
```

```
    'SVC': {
        'constructor': SVC(),
        'predict': lambda m, x_test: m.predict(x_test),
        'predict_prob': None,
    },
    'MLPClassifier': {
        'constructor': MLPClassifier(),
        'predict': lambda m, x_test: m.predict(x_test),
        'predict_prob': None,
    },
}
# Function to evaluate the model
def evaluate_model(algo, x_train, y_train, x_test, y_test, model_filename):
    start_time = time.time()  # Registra o tempo de início do treinamento
    model = algo['constructor']
    model.fit(x_train, y_train)
    training_time = time.time() - start_time

    with open(model_filename, 'wb') as file:
        pickle.dump(model, file)

    y_pred = algo['predict'](model, x_test)
    acc = accuracy_score(y_test, y_pred)
    prec = precision_score(y_test, y_pred)
    rec = recall_score(y_test, y_pred)
    f1 = f1_score(y_test, y_pred)
    cm = confusion_matrix(y_test, y_pred)
    tn, fp, fn, tp = cm.ravel()
    specificity = tn / (tn + fp)

    return {'acc': acc, 'prec': prec, 'rec': rec, 'f1': f1, 'kappa': kappa,
            'specificity': specificity, 'training_time': training_time}

# List of models
models = [
    {'algo': 'LogisticRegression', 'ds': 'processed'},
    {'algo': 'DecisionTreeClassifier', 'ds': 'processed'},
    {'algo': 'RandomForestClassifier', 'ds': 'processed'},
    {'algo': 'MLPClassifier', 'ds': 'processed'},
    {'algo': 'SVC', 'ds': 'processed'},
]

# Model evaluation
for model in models:
    x_train, x_test, y_train, y_test = datasets[model['ds']]
    model_name = model['algo']
    model_filename = f"{model_name}_model.pickle"
    model['metrics']   =   evaluate_model(algorithms[model['algo']],   x_train,
y_train, x_test, y_test, model_filename)
    model_acc = model['metrics']['acc']
    model_prec = model['metrics']['prec']
    model_rec = model['metrics']['rec']
    model_f1 = model['metrics']['f1']
    model_kappa = model['metrics']['kappa']
    model_auc = model['metrics']['auc']
```

```
        model_cm = model['metrics']['cm']
        model_specificity = model['metrics']['specificity']
        training_time = model['metrics']['training_time'] / 60

        print(f'Algorithm: {model_name} | accuracy: {model_acc} | precision:
    {model_prec} | rec: {model_rec} | f1: {model_f1} | specificity:
    {model_specificity} | training time: {training_time:.2f} minutes')
```

Code Snippet 3. ML model training code

To facilitate the comparisons between aggregation algorithms and between FL and traditional ML approaches, the metrics mentioned in Section 2.3 were utilized: Accuracy, Precision, Recall, F1-Score and Specificity.

6.4 Case studies

Considering the previously detailed topics, there was a need to develop more than one case study. By doing so, it was possible to address different challenges presented by diverse medical scenarios, thus providing a more thorough analysis. Therefore, to apply the previously mentioned subjects of analysis and evaluation, a total of two case studies were employed. By focusing on both, their differences allow for different aspects to be analyzed:

- **AP Mortality Prediction**: This case study focuses on predicting the mortality of patients with AP in an ICU setting. Given that AP is a specific condition within the larger dataset, the available data for this case study is relatively less. In this case study, the relatively small dataset size presents unique challenges.

- **General Diseases Mortality Prediction**: This broader case study aims to predict the mortality of ICU patients suffering from the whole spectrum of diseases present in the MIMIC-IV dataset. Naturally, the dataset for this scenario is substantially larger, encompassing the diverse conditions documented in database.

The option for these two case studies makes it possible to investigate some key topics. Firstly, it allows us to understand the influence of dataset size on FL outcomes. Logically, the data for general disease mortality prediction is more extensive than the data for AP mortality prediction, as AP represents just one of many conditions in the dataset. This comparison is instrumental in highlighting how dataset size can affect model performance and resource requirements. It also allows a comparative analysis between a niche topic and a broader one.

Secondly, while the AP mortality prediction case study does not provide extensive insights into time consumption due to its smaller dataset, the general diseases mortality prediction case study is more suitable for this analysis. The larger amount of data in the second case allows for a more efficient contemplation of the time of different approaches. Therefore, it allows to support more informed decision-making regarding their practical deployment in healthcare environments.

In conclusion, the investigation of these two case studies underscores the importance of dataset size, time efficiency, and comparative analysis in the development and evaluation of FL models for healthcare applications. By exploring both niche and broad scenarios, this study provides a holistic view of the potential and challenges of FL in improving patient outcomes in ICU settings. The findings contribute to the growing body of knowledge in the field of FL, offering valuable insights in this scope. Finally, both case studies are fully detailed in Sections 7 and 8, respectively. These sections provide an in-depth analysis of the methodologies, pre-processing, results, and implications of the predictive models developed for each case study. The detailed analysis also includes the evaluation and discussion of model performance metrics.

6.5 Chapter Remarks

Overall, the proposed solution contemplates a scenario where a network of hospitals, represented by clients, collaborate with a central server to train a global model capable of predicting mortality. This mortality prediction is twofold: in the first study case, the prediction is solely based on AP patients, while in the second one it reflects all the patients in an ICU. To achieve this set, Flower Framework acts as coordinator between server and clients, also allowing their implementation. Finally, traditional ML models were also trained for comparative purposes. This comparison naturally involved analyzing the metric values obtained in all the models (both in traditional ML and in all three aggregation algorithms' FL approaches).

7 Acute Pancreatitis Mortality Prediction

This chapter presents the first study case, which involves the mortality prediction of AP patients that are staying in an ICU. Therefore, it represents the first task developed with both Flower and the MIMIC-IV dataset.

AP is a common medical disorder that affects many people [76], being the most common type of pancreatitis [77]. It manifests as a recurrent, acute, or chronic pancreatic inflammation and can lead to death [78]. Forecasting mortality caused by AP helps to allocate health resources better and provide timely and effective treatment to stop people from perishing. Aside from developing a predictive model, it was also intended to investigate if FL results in a significant decrease in metric assessments when compared to traditional ML methods. Moreover, as explained in Proposed , this use case was also meant to analyze which aggregation algorithm performed better between FedAvg, FedAdam and FedAdagrad, which are three of the most popular aggregation algorithms present in Flower.

Furthermore, as it was previously mentioned, the chosen dataset was MIMIC-IV. It is broken up into multiple tables, each with an own set of contents. So, only some of the tables were necessary to perform the AP mortality prediction. The following tables are the ones that were needed:

- admissions - Data regarding hospital patient admissions is shown in this table. Although a patient may be admitted to the hospital more than once, each row represents a single admission incident.

- diagnoses_icd - This table includes International Classification of Diseases (ICD) codes, which represent diagnoses associated with all the patient admissions. Each row corresponds to a distinct code issued to a patient throughout their hospitalization.

- d_icd_diagnoses - This tables does the mapping between each code previously mentioned and the disease or condition it represents. Therefore, the table provides the mapping between diagnosis codes and their corresponding meanings.

- patients - The patients table has demographic information about each admitted patient. Age and gender are two examples of what can be found in there.

- chartevents - This one has recorded measurements and observations about patients during their time at the hospital. It is one of the most important as it includes a diverse range of medical data, mainly vital signs, but also laboratory results, intake/output measurements, and other clinical observations.

- d_items - It serves as a "code-name translator" for items recorded in the chartevents. Here, every ID present in the previous table is associated with its meaning.

7.1 Pre-processing And Model Creation

The pre-processing began by limiting the dataset to only include individuals admitted to the hospital with AP who were above the age of 18. As previously indicated, each disease is identified by a code. There are various options available for individuals with AP, as Table 14 shows. It's worth noting that all the AP-related ICD codes were considered. It was also decided to remove the data related to patients with less than 18 years old, as minores can introduce variables that can impact the models differently [135].

Table 14. ICD Codes for AP clinical cases

ICD Version	ICD Code	Description
9	7550	Acute Pancreatitis
10	K85	Acute Pancreatitis
10	K850	Idiopathic acute pancreatitis
10	K8500	Idiopathic acute pancreatitis without necrosis or infection
10	K8501	Idiopathic acute pancreatitis with uninfected necrosis
10	K8502	Idiopathic acute pancreatitis with infected necrosis
10	K851	Biliary acute pancreatitis
10	K8510	Biliary acute pancreatitis without necrosis or infection
10	K8511	Biliary acute pancreatitis with uninfected necrosis
10	K8512	Biliary acute pancreatitis with infected necrosis
10	K852	Alcohol induced acute pancreatitis
10	K8520	Alcohol induced acute pancreatitis without necrosis or infection
10	K8521	Alcohol induced acute pancreatitis with uninfected necrosis
10	K8522	Alcohol induced acute pancreatitis with infected necrosis
10	K853	Drug induced acute pancreatitis
10	K8530	Drug induced acute pancreatitis without necrosis or infection
10	K8531	Drug induced acute pancreatitis with uninfected necrosis
10	K8532	Drug induced acute pancreatitis with infected necrosis

ICD Version	ICD Code	Description
10	K858	Other acute pancreatitis
10	K8580	Other acute pancreatitis without necrosis or infection
10	K8581	Other acute pancreatitis with uninfected necrosis
10	K8582	Other acute pancreatitis with infected necrosis
10	K859	"Acute pancreatitis, unspecified
10	K8590	"Acute pancreatitis without necrosis or infection, unspecified"
10	K8591	"Acute pancreatitis with uninfected necrosis, unspecified"
10	K8592	"Acute pancreatitis with infected necrosis, unspecified"

Following that, the one-hot encoding approach was used to convert all category labels to binary format. To replace these missing values, a K-Nearest Neighbors (KNN) imputer was used to address the NaN values, as it is shown in Code Snippet 4. Removing the lines with NaN was also considered, but the fact that this study case has less data available led to the adoption of the KNN approach.

```
def custom_impute(df):
    imputer = KNNImputer()
    imputed_df = pd.DataFrame(imputer.fit_transform(df), columns=df.columns)
return imputed_df
```

Code Snippet 4. KNN Imputation in AP Mortality Prediction

In order to proceed with the binarization process, some categories were further split into subcategories. For example, the age category was divided into two groups: those under 65 and those with 65 and older, as Code Snippet 5 depicts. This division recognized the heightened vulnerability of the elderly, as evidenced by [79].

```
categories = ['-65', '+65']
age_limits = [0, 65, float('inf')]
merged_df['age_category'] = pd.cut(merged_df['anchor_age'], bins= age_limits,
labels=categorias, right=False)
dummies = pd.get_dummies(merged_df['age_category'], prefix='age')

merged_df = pd.concat([merged_df, dummies], axis=1)
merged_df = merged_df.drop(columns=['anchor_age', 'age_category'])
```

Code Snippet 5. Binarization of the Age Category

An imbalance in the "hospital_expire_flag" category was discovered during dataset analysis. This category, which is considered the dependent variable, indicates whether the patient passed away or survived. So, this imbalance may cause the model to become biased. Due to their popularity in the literature, two methods — SMOTE and undersampling, with a 60:40 ratio — were tested to remedy this issue. Both approaches produced satisfactory results, but undersampling performed slightly better. Therefore, this method, present in Code Snippet 6, was chosen for its effectiveness and because it exclusively utilizes real data, avoiding the need for synthetic data generation.

```
# Count the number of examples in each class
class_counts = merged_df['hospital_expire_flag'].value_counts()

# Determine the majority and minority class
majority_class = class_counts.idxmax()
minority_class = class_counts.idxmin()

# Calculate the number of examples to keep from the majority class
minority_count = class_counts[minority_class]
majority_count = class_counts[majority_class]
desired_majority_count = int(minority_count / 0.4 * 0.6)

# Randomly select examples from the majority class to match the desired ratio
undersampled_majority = merged_df[merged_df['hospital_expire_flag'] ==
majority_class].sample(n=desired_majority_count)

# Select all examples from the minority class
undersampled_minority = merged_df[merged_df['hospital_expire_flag'] ==
minority_class]

# Concatenate the undersampled majority and all examples from the minority
class
undersampled_df = pd.concat([undersampled_majority, undersampled_minority])

# Shuffle the data
undersampled_df = undersampled_df.sample(frac=1, random_state=42)
```

Code Snippet 6. Undersampling in AP Mortality Prediction

Furthermore, the FL models were created according to what was stated in Section 6, with a server and three clients collaborating to create all the models. Moreover, the ML were also created accordingly to what it was previously described.

7.2 Results

As previously mentioned, it was decided to train different ML models, without considering the FL approach, as a first step. Therefore, the results achieved are present in Table 15. The first conclusion that can be made through the obtained results is the fact that the MLP was the classifier that obtained most success, as its metrics were the highest. It had the greatest values for every single one of them except the Recall metric, which was higher in the SVC. After comparing the performance of the models with the state-of-the-art works, also presented in Table 15, it becomes clear that both MLP and SVC outperformed the others across various evaluation metrics. Specifically, they exhibited the highest Precision, Recall, and F1-score values among all the works. Still, it is noteworthy that Mofidi et al's [83] ANN presented the greatest Accuracy and Specificity values. Even though the ideal scenario would be comparing the achieved FL models with other FL models, this was not possible as this study is pioneering in the context of utilizing FL to predict mortality by AP.

Table 15. AP Mortality Prediction - Machine Learning results compared to state-of-the-art works

Classifier	Accuracy	Precision	Recall	F1-Score	Specificity
Current Work					
Logistic Regression	0.910	0.865	0.900	0.882	0.917
DecisionTreeClassifier	0.836	0.780	0.780	0.780	0.869
RandomForestClassifier	0.896	0.846	0.880	0.863	0.905
SVC	0.918	0.855	**0.940**	0.895	0.905
MLPClassifier	0.925	**0.885**	0.920	**0.902**	0.929
State-of-the-art works					
Ding et al's ANN [82]	0.662	0.563	0.666	0.610	0.661
Mofidi et al's ANN [83]	**0.975**	0.750	0.880	0.809	**0.980**
Hameed et al.'s Random Forest [84]	Unknown	0.255	0.833	0.658	0.870
Ren et al.'s GNB With MIMIC-IV [85]	0.787	0.683	0.839	0.774	0.792

After this approach, different FL models were trained in the previously described scenario. In exploring FL aggregation algorithms, the evaluation of various techniques makes it possible to obtain valuable insights into their comparative efficacy. The findings presented in Table 16 contribute to explaining the comparative efficacy of various previously mentioned FL aggregation methods, namely FedAvg, FedAdam, and FedAdagrad, implemented with the Flower framework. FedAvg emerged as the best performer, showcasing superior metric values across all parameters, except the Recall metric for the MLP model. Additionally, it is worth noticing that FedAdagrad showed a notable underperformance for the SVC. The superior performance of FedAvg and underperformance of FedAdagrad was previously stated in other works when dealing with independent and identically distributed data [87, 88]. Notably, the SVC trained via FedAvg aggregation emerged as the best performer, which will be further discussed in subsequent elaboration.

Table 16. Comparison between aggregation algorithms for AP mortality prediction

Classifier	Accuracy	Precision	Recall	F1-Score	Specificity
FedAvg					
Logistic Regression	0.843	0.829	0.708	0.764	0.919
DecisionTreeClassifier	0.858	0.796	0.813	0.804	0.884
RandomForestClassifier	**0.903**	0.857	0.875	0.866	0.919
SVC	**0.903**	0.818	0.938	**0.874**	0.884
MLPClassifier	0.881	**0.881**	0.771	0.822	**0.942**
FedAdam					
Logistic Regression	0.813	0.820	0.714	0.763	0.821
DecisionTreeClassifier	0.855	0.777	0.840	0.808	0.864
RandomForestClassifier	0.877	0.800	0.880	0.838	0.875
SVC	0.884	0.783	0.940	0.855	0.852
MLPClassifier	0.783	0.778	0.560	0.651	0.909
FedAdagrad					
Logistic Regression	0.814	0.819	0.712	0.762	0.803

Classifier	Accuracy	Precision	Recall	F1-Score	Specificity
DecisionTreeClassifier	0.848	0.774	0.820	0.796	0.863
RandomForestClassifier	0.857	0.810	0.840	0.825	0.875
SVC	0.384	0.370	0.840	0.514	0.341
MLPClassifier	0.811	0.667	**0.960**	0.787	0.727

Next it was important to analyze if there was a loss of metrics when adopting the FL approach. As a result of FedAvg having the best results between the three aggregation algorithms, it was chosen for this comparison. In the first place, it was possible to say that both ML and FL approaches had a great performance across various evaluation metrics in most models. In the second place, there was generally no significant degradation in performance metrics when transitioning from ML to FL. These results are expected, since FedAvg is known to have comparable results to ML, as previously mentioned. However, Logistic Regression showed itself as an exception as its values generally decreased (excluding the specificity value). Even though FedAVG tends to keep the metric values similar, it is not impossible to find this type of situation in Logistic Regression in FL [89], mainly in the healthcare domain, as this also happened in the literature. MLP also experienced some loss in the recall value and, consequently, in the F1-Score value too. Conversely, Decision Tree experienced a small improvement in almost all evaluation metrics. Random Forest also presented a very small increase in the accuracy metric, when the FL approach was used. Once again, these smalls improvements are understandable, as FedAVG presents comparable performance to ML. Therefore, the observed results show that the performance of FL models is generally comparable to ML ones, with MLP being the highlight in ML and SVC standing out in FL.

7.3 Chapter Remarks

In this case study, the results showed FedAvg having an overall better performance than the other aggregation algorithms. In this realm, FedAdagrad also presented the worst metrics when compared to the other two FL aggregation algorithms. Furthermore, the chapter has also shown that, even though there can be a small loss of metrics when opting for the FL approach, it is not significant. In fact, the FedAvg results are very similar to the ones found in traditional ML, which makes it viable to opt for a more privacy-friendly and collaborative environment.

8 Diseases Mortality Prediction

This chapter represents the second study case. Conversely to what happened in the previous chapter, this is not meant to deal with a single disease. Instead, it is meant to predict the mortality for general diseases, considering patients staying in an ICU. Consequently, it is the second task developed with MIMIC-IV dataset and Flower framework.

Just as predicting mortality caused by AP helps to organize health resources in a more efficient way and provide timely and effective treatment to prevent fatalities, the same principle can be applied to this wider case. In this case study, the ultimate goal is threefold: first, to develop a predictive model capable of predicting mortality without limiting the scope to a single disease. This is important because it allows for a broader application of the model in various healthcare settings, providing a more versatile and comprehensive tool for predicting patient outcomes across different conditions. By not restricting the model to a single disease, we introduce a novel use-case that can handle different medical scenarios, thereby enhancing the generalizability of the experiment. Second, just as in Section 7, it is aimed to evaluate whether FL techniques result in significant decreases in key performance metrics compared to conventional ML methods. Finally, it is also a goal to compare the performance of three distinct FL aggregation algorithms – FedAVG, FedAdam, and FedAdagrad – to understand which one performs better. The expanded scope to include a wide range of diseases increases the size of the data. As a result, it also becomes possible to analyze how FL models deal with larger datasets and whether they can maintain the patterns seen in the previous case. Additionally, the scenario of a larger dataset also makes it possible to analyze the time needed to train each one of the models. Minimizing training time can be an important factor due to both economic and environmental reasons. Models that require less time to be trained are usually preferable as they lead into an operational costs and carbon footprint reduction.

8.1 Pre-processing And Model Creation

In this section, the pre-processing started with the removal of the underage patients. The reason for this, as previously stated, is the fact that minors may introduce variables that could impact the prediction differently [135]. Underage individuals have clinical particularities that can impact the outcome of the model. As a consequence of this being a bigger use case, that utilizes more data, it was decided to remove all the lines that included null values, as it is shown in Code Snippet 7. This data removal process was divided into batches to prevent an "out of memory" error. This allows the computer to deal with only a portion of the data at each time. After all the batches were finished, they were concatenated in a single dataset without the NaN values. Consequently, unlike the previous case, no imputation was done on the dataset. This approach has the advantage of assuring the model is only trained on real data, thereby avoiding the inclusion of synthetic values. Furthermore, it also allows faster model training as it utilizes less data, while keeping the characteristics of dealing with a much bigger dataset than the one utilized in the previous case study. The combination of these factors led to this decision, even though data imputing was a considered option.

```
chunksize = 100000
processed_batches = []
for i, chunk in enumerate(pd.read_csv("CSVs\\final_merged_dataset.csv",
chunksize=chunksize)):
    chunk = merged_df.dropna()
    processed_batches.append(chunk)
    print(f"Processed batch {i+1}")
    print(chunk.head())
df_no_nulls = pd.concat(processed_batches)
```

Code Snippet 7. Removing lines with NaN values

After the removal, the variables were converted into binary format, as previously illustrated in Code Snippet 8. Most models benefit from binary data, allowing them to have better results. Furthermore, some models, such as the logistic regression, cannot deal with categorical variables. Therefore, when binarizing categorical variables, this issue was also being addressed. Post-binarization, attention shifted towards addressing the issue of class imbalance in the “hospital_expire_flag”. To mitigate this, both SMOTE and undersampling, with different distributions, methods were evaluated. However, the 50/50 undersampling technique, demonstrated in Code Snippet 8, presented superior results in terms of model performance, when analyzing the traditional ML model results.

```
# Count the number of examples in each class
class_counts = merged_df['hospital_expire_flag'].value_counts()
# Determine the majority and minority class
majority_class = class_counts.idxmax()
minority_class = class_counts.idxmin()
# Calculate the number of examples to keep from the majority class
minority_count = class_counts[minority_class]
majority_count = class_counts[majority_class]
desired_majority_count = int(minority_count / 0.5 * 0.5)
```

```
# Randomly select examples from the majority class to match the desired ratio
undersampled_majority = merged_df[merged_df['hospital_expire_flag'] ==
majority_class].sample(n=desired_majority_count)
# Select all examples from the minority class
undersampled_minority = merged_df[merged_df['hospital_expire_flag'] ==
minority_class]
# Concatenate the undersampled majority and all examples from the minority class
undersampled_df = pd.concat([undersampled_majority, undersampled_minority])
```

Code Snippet 8. Undersampling in general Diseases Mortality Prediction

Following, the FL and traditional ML models were created accordingly to the description provided in Proposed Solution, which means that Flower coordinated a set of three clients and a server collaborating to create FL models.

Finally, this case study presented a particularity related to time constraints. After delving into the training of the models, it became evident that it was not feasible to train the SVC through FL with the whole dataset. Due to the limited resources available, it was taking more than one day just to finish the first round of FL. Therefore, two different approaches were chosen. The first one utilized a portion of 10% of the dataset for training and testing, which is still a large part of it (englobing more than 35 000 lines). This approach allowed a fair comparison between all the models, including the SVC. The second one utilized the complete dataset. However, it excluded the SVC from the comparison. This twofold approach granted the SVC could be analysed without suffering from an unfair disadvantage: being trained and still compared with less data than the others, which could lead into misleading results. In the author's opinion, this is the most suitable solution for the scientific purpose of comparing several models.

8.2 Results

After completing the training of the ML models and the FL models, a comprehensive comparison was done to explore different aspects. This analysis aimed to reflect the points mentioned in Chapter 6. Therefore, it was meant to determine whether the adoption of a FL approach leads into a loss of metrics or not, compare the performance of different aggregation algorithms (FedAvg, FedAdam, and FedAdagrad), and assess the time efficiency of each algorithm and model. The results obtained for ML and FL, with only 10% of the dataset being utilized, are reflected in Table 17.

Table 17. ML and FL results to General Diseases Mortality Prediction – 10% of the dataset

Classifier	**Accuracy**	**Precision**	**Recall**	**F1-Score**	**Specificity**	**Training Time - minutes**
Machine Learning						
Logistic Regression	0.839	0.832	0.790	0.810	0.863	**0.00**
DecisionTreeClassifier	0.856	**0.892**	0.794	**0.841**	0.918	**0.00**
RandomForestClassifier	**0.857**	0.883	0.802	**0.841**	0.909	0.03

Classifier	Accuracy	Precision	Recall	F1-Score	Specificity	Training Time - minutes
SVC	0.843	0.852	0.798	0.824	0.881	0.89
MLPClassifier	0.843	0.865	0.784	0.822	0.895	0.55
FedAvg						
Logistic Regression	0.833	0.822	0.806	0.814	0.855	0.02
DecisionTreeClassifier	0.860	0.883	0.796	0.838	0.913	0,02
RandomForestClassifier	0.859	0.869	0.812	0.840	0.898	0,30
SVC	0.841	0.836	0.811	0.823	0.868	66,89
MLPClassifier	0.747	0.844	0.544	0.662	0.917	3,32
FedAdam						
Logistic Regression	0.827	0.820	0.798	0.809	0.805	0.02
DecisionTreeClassifier	0.860	0.884	0.797	0.839	0.911	0,02
RandomForestClassifier	0.858	0.868	0.812	0.839	0.897	0,29
SVC	0.841	0.832	0.814	0.823	0.863	65,10
MLPClassifier	0.660	0.816	0.324	0.468	**0.939**	3,30
FedAdagrad						
Logistic Regression	0.818	0.819	0.795	0.807	0.845	0.02
DecisionTreeClassifier	0.860	0.883	0.798	0.838	0.912	0.02
RandomForestClassifier	0.857	0.866	0.811	0.838	0.896	0,30
SVC	0.457	0.455	**0.996**	0.625	0.451	68,83
MLPClassifier	0.552	0.890	0.014	0.028	0.936	3,25

When comparing all the trained models, it becomes clear that the Random Forest and Decision Tree trained by traditional ML are the ones with the greatest metric values. Random Forest showed the greatest Accuracy and F1-Score, while the Decision Tree had the highest precision and F1-Score. Moreover, the FL MLP associated with FedAdam showed the highest specificity and the FL SVC aggregated with FedAdagrad presented the highest recall. However, other conclusions can be made. Once again, it is possible to conclude that, generally, all the FedAvg models have a comparable performance with the traditional ML models, as it is stated in the literature [87]. Still, the MLP is associated with a higher decrease in metric values when compared with the others. This decrease was already present in some metrics in the previous section results, which leads the author to infer that this is a typical behavior in this kind of data. Moreover, the MLP also shows an even worse performance for the other aggregation algorithms. It is also able to observe that the three aggregation algorithms have similar performances in three out of the five classifiers. Despite this, there's two exceptions: FedAdagrad showed terrible metrics when training the SVC and the MLP in comparison with the other options. This tendency was already observed in the previous section and in other author's works, which strongly suggests that this is a common behavior with this algorithm. Furthermore, as it was possible to understand from previous conclusions, the MLP also underperformed with FedAdam in comparison to FedAvg.

Moreover, analyzing the time necessary to train each one of the algorithms is also one of the goals of this experiment. Therefore, it was possible to analyze that ML algorithms are more quickly trained than the FL ones. This is an expected behavior, as ML does not need to aggregate

other models, contrary to what happens in FL, and does not involve multiple rounds. However, it is worth noticing the needed time was not extremely high for the models with the best performance. Table 17 also makes it clear that the SVC is the slowest classifier. Therefore, if the intention is to save energy, time and resources this is the least suitable option. On the other hand, Logistic Regression and Decision Tree were the fastest classifiers, never needing more than 0.02 minutes to finish training. Random Forest is also quick when analyzing the ML results, although it is a bit slower in the FL setting. Still, it only took about 0,30 minutes to finish training, so it presents itself as an option to consider, especially when looking into the performance metric values. Note that only 10% of the dataset was utilized in this case, so all the models would need more time to train the complete pre-processed dataset, as it will be shown next. The last conclusion illustrated by the results is the fact that all the three algorithms need approximate time to finish training. So, the time issue becomes practically indifferent when choosing between them.

In addition, the results obtained for ML and FL, with the complete dataset being utilized, are reflected in Table 18.

Table 18. ML and FL results to General Diseases Mortality Prediction – complete dataset.

Classifier	**Accuracy**	**Precision**	**Recall**	**F1-Score**	**Specificity**	**Training Time - minutes**
Machine Learning						
Logistic Regression	0.843	0.823	0.794	0.808	0.875	0.03
DecisionTreeClassifier	0.892	0.933	0.820	0.873	0.951	**0.01**
RandomForestClassifier	**0.893**	**0.936**	0.821	**0.875**	0.953	0.34
MLPClassifier	0.843	0.865	0.784	0.822	0.895	4,57
FedAvg						
Logistic Regression	0.840	0.824	0.796	0.810	0.868	0.08
DecisionTreeClassifier	0.892	0.933	**0.822**	0.874	0.948	0,17
RandomForestClassifier	0.886	0.935	0.817	0.870	**0.965**	3,04
MLPClassifier	0.756	0.720	0.749	0.734	0.818	36,78
FedAdam						
Logistic Regression	0.830	0.821	0.799	0.810	0.845	0.08
DecisionTreeClassifier	0.890	0.930	0.821	0.871	0.951	0,02
RandomForestClassifier	0.884	0.928	0.808	0.864	0.938	3,02
MLPClassifier	0.546	0.900	0.454	0.603	0.832	35,88
FedAdagrad						
Logistic Regression	0.829	0.815	0.795	0.808	0.834	0.08
DecisionTreeClassifier	0.889	0.928	**0.822**	0.856	0.950	0.19
RandomForestClassifier	0.885	0.927	0.820	**0.875**	0.942	3,12
MLPClassifier	0.546	0.645	0.102	0.176	0.435	36,56

This time, when comparing all the trained models, it becomes clear that the Random Forest trained by traditional ML is the one with the best metric values overall. It presented the best Accuracy, Precision and F1-Score. The best Recall metric was present in both FedAvg and

FedAdagrad Decision Trees, and the best Specificity value was shown by the FedAvg's Random Forest. Once again, the MLP underperformed in the FL setting, especially in FedAdam and FedAdagrad, when compared to ML. However, all the other classifiers showed comparable performance to the traditional ML results and between each other. Nevertheless, FedAvg performed slightly better, even though the difference is almost indifferent in three out of the four models. Finally, it is also possible to conclude that FL takes much more time than traditional ML, due to its iterative and aggregative nature. On top of that, when analyzing the time needed to train each one of the models, Logistic Regression and Decision Rree are the fastest, as expected. On the other hand, the MLP is the slowest, a tendency already shown in the 10% of the dataset case. Finally, the Random Forest shows a great balance between performance and needed time.

As this topic has already been studied in other ML works, it was possible to make a comparison between the best ML model trained in this work and traditional ML models from other works. So, Table 19 illustrates this comparison, contemplating some of the most prominent works that have focused on training ML models for ICU mortality prediction.

Table 19. General Diseases Mortality Prediction – State-of-the-art ML results comparison

Classifier	**Accuracy**	**Precision**	**Recall**	**F1-Score**	**Specificity**	**Training Time - minutes**
This Work's ML Random Forest	**0.893**	**0.936**	0.821	**0.875**	0.953	**0.34**
Iwase, et al.'s RandomForest [124]	Unknown	Unknown	**0.865**	Unknown	0.875	Unknown
Nistal-Nuño's Extreme Gradient Boosting (XGB) [136]	0.855	0.528	0.831	0.645	0.860	Unknown
Pang et al.'s XGBoost [137]	0.834	0.842	0.822	0.831	0.846	Unknown
Chia, et al.'s best XGB [138]	0.819	0.420	0.615	0.499	0.689	Unknown
Alghatani et al.'s Random Forest [139]	0.885	0.840	0.095	0.171	**0.997**	Unknown

The ML Random Forest from this work provided the best performance metrics in three out of the five analyzed metrics. In other words, it showed the best Accuracy, Precision and F1-Score, despite Iwase, et al.'s [124] Random Forest presenting the best Recall value. It also presented the second-best Specificity, trailing behind Alghatani et al.'s [139] Random Forest.

Contrary to what happened in Acute Pancreatitis Mortality Prediction, this time it was possible to find other works that utilized FL to predict mortality for general diseases. Therefore, it was

feasible to establish a comparison with the results presented in Table 20, which also includes the previously presented results for the Random Forest, with FedAvg, for the complete dataset, from this work. This table includes all the found works that utilized at least one of the metrics used in this thesis. However, the fact that none of them utilized all the metrics becomes an obstacle in the task of making a perfect comparison. Other works utilized different metrics to evaluate their models, but it was not possible to establish a comparison with their results, as those metrics were not utilized in this thesis.

Table 20. General Diseases Mortality Prediction – State-of-the-art FL results comparison

Classifier	**Accuracy**	**Precision**	**Recall**	**F1-Score**	**Specificity**	**Training Time - minutes**
This Work's RandomForest - FedAvg	**0.886**	**0.935**	**0.817**	**0.870**	**0.965**	**3,04**
Randl, et al. best FL model [92]	Unknown	0.520	0.460	0.480	Unknown	Unknown
Georgoutsos best FL model [91]	Unknown	Unknown	Unknown	0.512	Unknown	Unknown
Mondrejevski et al. best FL model [90]	Unknown	Unknown	Unknown	0.830	Unknown	Unknown

Firstly, Randl,et al. [92] were the authors that presented most metrics, which makes it possible to do a fairer comparison with their work than with the other authors' work. Even though their research has several results, as a consequence of having various models, their best FL model was selected for this comparison. Hence, it is possible to understand that their results are inferior to the majority of the models trained in this work, both in ML and FL. Unfortunately, Georgoutsos [91] and Mondrejevski et al. [90], from all the metrics utilized in this work, only presented the results for the F1-Score. The first one showed a low F1-Score, which is also inferior to most of the models presented above (the exceptions is the MLP model with the FedAdagrad algorithm). Finally, Mondrejevski et al. [90] showed the best F1-Score between the three state-of-the-art works. Even though it is a good result, and surpasses some of the models in this chapter, it does not show a better performance than the best models from Table 18, which are the Random Forest and Decision Tree trained with any of the aggregation algorithms.

8.3 Chapter Remarks

After carefully analysing the results for both approaches (10% of the dataset, and the total dataset), it was possible to confirm some results already shown in the first case study. The MLP tends to be less accurate in a FL set and the SVC performance with FedAdagrad was once more much worse than in any other approach (both in FL and traditional ML). Moreover, as expected,

the SVC is the slowest classifier, while Logistic Regression is the fastest. No significant difference between the time needed. In terms of performance, most of the aggregation algorithms' models showed comparable performances between each other and with the ML models. However, traditional ML models still had the best metrics, even though the difference was not significant. In terms of FL only, FedAvg was slightly better than the other two aggregation algorithms, even though the performances were practically equivalent.

9 Conclusions

This chapter presents the essential conclusions of this thesis, emphasizing the objectives that have been achieved. It is also meant to outline the limitations of the work, pointing out what can be improved in the future

9.1 Accomplished goals

This thesis was delineated to address a privacy-friendly FL approach in the healthcare context. Therefore, it was meant to provide insights related to the topic. This end was only possible by establishing three different goals, which were all achieved. Two case studies were developed to better understand the particularities of FL, revealing useful conclusions. The main results for each objective were:

- G1 – A literature review provided revealing insights about different FL tools and frameworks. Tools like PySyft and Flower stood out, with the second being the chosen one for the practical part of this thesis.

- G2 – It was possible to understand how FL is being applied to healthcare by analyzing the challenges faced by its users. Problems like data distribution challenges, scalability challenges, specificities of each type of FL partition, and privacy challenges were identified. Moreover, by analyzing that it was possible to understand that most of the FL settings contemplate hospital networks, which ended being a pivotal foundation of the practical experiments.

- G3 – With the goal of proposing “an effective FL approach that ensure data privacy in healthcare”, diverse models were created to understand a few things: which one of them shows the best performance, which aggregation algorithm has the best performance, the existence or not of a loss of metric values when opting for FL and the

time needed for each model to be trained. Hence, it was possible to conclude that even though there might be a loss of metrics when adopting a FL approach, it is not significant. Furthermore, between FedAvg, FedAdam, and FedAdagrad, the first one was the aggregation algorithm that showed the least metric loss. Consequently, for the vast majority of the models, FedAvg was the greatest aggregation algorithm. Finally, the SVC was the slower model, which makes it the worst option in terms of time and energy saving. On the other hand, the Decision Tree and Logistical Regression were the quickest models, while Random Forest showed a great balance between speed and accuracy. In addition, when training these models, it was taken into consideration the technical challenges found regarding G2. For example, to avoid the Data Quantity Skew, undersampling techniques were employed. Taking into consideration the results obtained in both Sections 7 and 8, it was possible to conclude that the solution proposed in Section 6 meets what was delineated.

Moreover, it is important to note that a paper focused on exploring the first case study – Acute Pancreatitis Mortality Prediction – was also accepted at the EPIA conference [141].

The obtained results strongly support that prioritizing privacy and security with a FL approach does not necessarily mean that the robustness of the models must be sacrificed. Contrarily, it rarely shows significant losses in metric values. Therefore, if privacy and security are a priority, FL becomes an important weapon.

9.2 Limitations and Future Work

Despite the promising results obtained in this thesis, some aspects can be enhanced to strengthen the findings and extend the applicability of the research.

Firstly, the study did not incorporate cross-fold validation, a vastly recognized technique for evaluating the robustness of models. Cross-fold validation helps to mitigate the risk of overfitting by ensuring that the model is tested on various subsets of the data. Its absence means that the performance metrics reported may be higher than in reality, because they are based on a single train-test split. Therefore, incorporating cross-fold validation in future work would provide a more rigorous evaluation of the models' performance.

Secondly, while the study evaluated three of the most popular FL aggregation algorithms—FedAvg, FedAdam, and FedAdagrad—it did not explore other potentially effective algorithms, such as FedYogi. As these algorithms may outperform the ones tested in specific scenarios, they might be considered in future research.

Additionally, the study was conducted considering a healthcare network with three hospitals. Although this setting provided valuable insights, it certainly does not fully capture the particularities of larger hospital networks. The findings may differ in more extensive networks

where variations in data distributions and clinical practices are more pronounced. Consequently, future work should extend the analysis to also include bigger networks of hospitals.

Furthermore, the research focused on two specific case studies: AP and general diseases mortality prediction. While these case studies are relevant and are enough for the purpose of this work, they are the only a fraction of the potential applications of FL in healthcare. The healthcare universe is obviously composed of an almost infinite range of possibilities. So, future work can explore more medical conditions to validate the findings across different contexts and demonstrate the versatility of the FL approach in healthcare.

Moreover, the practical side of this thesis exclusively focused on HFL, as it was the partition type the most suited the dataset. Therefore, in the future, there's room for exploring both VFL and FTL, which can bring new insights related to both traditional ML/FL and aggregation algorithms comparisons. They allow to explore scenarios where the clients don't have the same features or scenarios where a pre-trained model is utilized.

There is also room to implement more security and privacy techniques, like cryptography. As previously stated, even though FL is a privacy-friendly approach, the models' weights can still be intercepted and sometimes reveal private information. With cryptography, the data is even more secure.

Lastly, as this is a simulated environment, practical challenges associated with deploying FL in real-world, such as latency were not addressed in this study. Future work should investigate these challenges through pilot studies and real-world setting to obtain valuable conclusion on the applicability of FL.

9.3 Final considerations

Overall, this thesis is a representation of all the work that led into the results presented within it, from the initial investigations about FL to the last words written in this document. Personally, it was an extremely interesting topic, as I started with any solid notions about it and finished with a great knowledge about it. Therefore, it allowed me to develop not only my comprehension about FL but also my technical skills along with the application of previously known information. I believe that developing this work was immensely beneficial in personal, academical, and professional aspects.

References

[1] D. Moher et al., “Preferred reporting items for systematic review and meta-analysis protocols (PRISMA-P) 2015 statement,” Syst. Rev., vol. 4, no. 1, p. 1, 2015, doi: 10.1186/2046-4053-4-1.

[2] “Elsevier ScienceDirect Search Source.” https://www.sciencedirect.com/search (accessed Dec. 07, 2023).

[3] Ludwig, H., Baracaldo, N., Thomas, G., Zhou, Y., Anwar, A., Rajamoni, S., Ong, Y., Radhakrishnan, J., Verma, A., Sinn, M., & outros. (2020). IBM Federated Learning: an Enterprise Framework White Paper V0.1. arXiv preprint arXiv:2007.10987.

[4] Substra documentation. Substra documentation - Substra 0.34.0 documentation. (n.d.). https://docs.substra.org/en/stable/ (accessed Dec. 11, 2023)

[5] Kholod, I., Yanaki, E., Fomichev, D., Shalugin, E., Novikova, E., Filippov, E., & Nordlund, M. (2020). Open-source federated learning frameworks for IoT: A comparative review and analysis. Sensors, 21(1), 167.

[6] Liu, X., Shi, T., Xie, C., Li, Q., Hu, K., Kim, H., ... & Song, D. (2022). Unifed: A benchmark for federated learning frameworks. arXiv preprint arXiv:2207.10308.

[7] Guendouzi, B. S., Ouchani, S., Assaad, H. E., & Zaher, M. E. (2023). A systematic review of federated learning: Challenges, aggregation methods, and development tools. Journal of Network and Computer Applications, 103714.

[8] Galtier, M. N., & Marini, C. (2019). Substra: a framework for privacy-preserving, traceable and collaborative machine learning. *arXiv preprint arXiv:1910.11567*.

[9] Beltrán, E. T. M., Pérez, M. Q., Sánchez, P. M. S., Bernal, S. L., Bovet, G., Pérez, M. G., ... & Celdrán, A. H. (2023). Decentralized federated learning: Fundamentals, state of the art, frameworks, trends, and challenges. IEEE Communications Surveys & Tutorials.

[10] Ibm (no date) IBM/Federated-Learning-Lib: A library for federated learning (a distributed machine learning process) in an enterprise environment., IBM Federated Learning - GitHub. Available at: https://github.com/IBM/federated-learning-lib (Accessed: 11 December 2023).

[11] “Web of Science Search Source” https://www.webofscience.com/wos/woscc/basic-search (accessed Dec. 07, 2023).

[12] “b-on Search Source” https://www.b-on.pt/ (accessed Dec. 11, 2023).

[13] “PaddleFL” https://paddlefl.readthedocs.io/en/latest/introduction.html (acessed Dec. 12, 2023)

[14] “FATE” https://github.com/FederatedAI/FATE (acessed Dec. 13, 2023)

[15] Beutel, D. J., Topal, T., Mathur, A., Qiu, X., Fernandez-Marques, J., Gao, Y., ... & Lane, N. D. (2022). Flower: A friendly federated learning framework.

[16] Munos, B., Baker, P. C., Bot, B. M., Crouthamel, M., de Vries, G., Ferguson, I., ... & Wang, P. (2016). Mobile health: the power of wearables, sensors, and apps to transform clinical trials. *Annals of the New York Academy of Sciences*, *1375*(1), 3-18.

[17] Deshmukh, S. A., Kasar, S. L., & Chichani, Y. (2023, July). Analysis of Challenges in Decentralized Storage Framework for Sharing Medical Data. In *2023 14th International Conference on Computing Communication and Networking Technologies (ICCCNT)* (pp. 1-10). IEEE.

[18] Margam, R. (2023). Ethics And Data Privacy: The Backbone of Trustworthy Healthcare Practices. *Socio-Economic and Humanistic Aspects for Township and Industry*, *1*(2), 232-236.

[19] Choudhary, P. (2022). Digital Information Security and Privacy Protection in Healthcare Sector in India. *Issue 2 Indian JL & Legal Rsch.*, *4*, 1.

[20] "Health Insurance Portability and Accountability Act of 1996 (HIPAA)" https://www.cdc.gov/phlp/publications/topic/hipaa.html (accessed Dec. 16, 2023)

[21] "European Data Protection Supervisor" https://edps.europa.eu/data-protection/our-work/subjects/health_en

[22] Li, L., Fan, Y., Tse, M., & Lin, K. Y. (2020). A review of applications in federated learning. *Computers & Industrial Engineering*, *149*, 106854.

[23] "GECAD Research Group." https://www.gecad.isep.ipp.pt/ (accessed Dec. 18, 2023)

[24] Yang, Q.; Liu, Y.; Chen, T.; and Tong, Y. 2019. Federated machine learning: Concept and applications. ACM Transactions on Intelligent Systems and Technology (TIST), 10(2): 1–19.

[25] Yan, Z.; Guoliang, L.; and Jianhua, F. 2016. A survey on entity alignment of knowledge base. Journal of Computer Research and Development, 53(1): 165.

[26] Liu, Y.; Kang, Y.; Xing, C.; Chen, T.; and Yang, Q. 2018. A secure federated transfer learning framework. IEEE Intelligent Systems, 35(4): 70–82.

[27] Yang, A., Ma, Z., Zhang, C., Han, Y., Hu, Z., Zhang, W., ... & Wu, Y. (2023). Review on application progress of federated learning model and security hazard protection. Digital Communications and Networks, 9(1), 146-158.

[28] Siniosoglou, I., Argyriou, V., Sarigiannidis, P., Lagkas, T., Sarigiannidis, A., Goudos, S. K., & Wan, S. (2023). Post-processing fairness evaluation of federated models: an unsupervised approach in healthcare. IEEE/ACM Transactions on Computational Biology and Bioinformatics.

[29] Rani, S., Kataria, A., Kumar, S., & Tiwari, P. (2023). Federated learning for secure IoMT-applications in smart healthcare systems: A comprehensive review. Knowledge-Based Systems, 110658.

[30] Quan, P. K., Kundroo, M., & Kim, T. (2023). Experimental Evaluation and Analysis of Federated Learning in Edge Computing Environments. IEEE Access, 11, 33628-33639.

[31] Huang, C., Huang, J., & Liu, X. (2022). Cross-silo federated learning: Challenges and opportunities. arXiv preprint arXiv:2206.12949.

[32] Wohlin, C. (2014, May). Guidelines for snowballing in systematic literature studies and a replication in software engineering. In Proceedings of the 18th international conference on evaluation and assessment in software engineering (pp. 1-10).

[33] Prayitno, Shyu, C. R., Putra, K. T., Chen, H. C., Tsai, Y. Y., Hossain, K. T., ... & Shae, Z. Y. (2021). A systematic review of federated learning in the healthcare area: From the perspective of data properties and applications. *Applied Sciences*, *11*(23), 11191.

[34] Zhao, Y., Li, M., Lai, L., Suda, N., Civin, D., & Chandra, V. (2018). Federated learning with non-iid data. *arXiv preprint arXiv:1806.00582*.

[35] Huang, L., Shea, A. L., Qian, H., Masurkar, A., Deng, H., & Liu, D. (2019). Patient clustering improves efficiency of federated machine learning to predict mortality and hospital stay time using distributed electronic medical records. *Journal of biomedical informatics*, *99*, 103291.

[36] Li, J., Cai, X., & Cheng, L. (2023). Legal regulation of generative AI: a multidimensional construction. International Journal of Legal Discourse, 8(2), 365-388.

[37] Hegde, H.; Shimpi, N.; Panny, A.; Glurich, I.; Christie, P.; Acharya, A. MICE vs. PPCA: Missing data imputation in healthcare. Inform. Med. Unlocked 2019, 17, 100275.

[38] Zhao, Y.; Li, M.; Lai, L.; Suda, N.; Civin, D.; Chandra, V. Federated learning with non-IID data.

[39] Sharma, S., & Guleria, K. (2023). A comprehensive review on federated learning based models for healthcare applications. Artificial Intelligence in Medicine, 146, 102691.

[40] Liu, B., Yan, B., Zhou, Y., Yang, Y., & Zhang, Y. (2020). Experiments of federated learning for covid-19 chest x-ray images. *arXiv preprint arXiv:2007.05592*.

[41] A. Qayyum, J. Qadir, M. Bilal, and A. Al-Fuqaha, “Secure and robust machine learning for healthcare: A survey,” IEEE Rev. Biomed. Eng., vol. 14, pp. 156–180, 2021

[42] Sirohi, D., Kumar, N., Rana, P. S., Tanwar, S., Iqbal, R., & Hijjii, M. (2023). Federated learning for 6G-enabled secure communication systems: a comprehensive survey. Artificial Intelligence Review, 1-93.

[43] Paul S, Sengupta P, Mishra S (2020) Flaps: federated learning and privately scaling. In: 2020 IEEE 17th international conference on mobile ad hoc and sensor systems (MASS), pp 13–19. IEEE

[44] Topaloglu, M. Y., Morrell, E. M., Rajendran, S., & Topaloglu, U. (2021). In the pursuit of privacy: the promises and predicaments of federated learning in healthcare. Frontiers in Artificial Intelligence, 4, 746497.

[45] Lu, Z., & Shen, H. (2017, December). A new lower bound of privacy budget for distributed differential privacy. In 2017 18th International Conference on Parallel and Distributed Computing, Applications and Technologies (PDCAT) (pp. 25-32). IEEE.

[46] Boneh, D., Sahai, A., & Waters, B. (2011). Functional encryption: Definitions and challenges. In Theory of Cryptography: 8th Theory of Cryptography Conference, TCC 2011, Providence, RI, USA, March 28-30, 2011. Proceedings 8 (pp. 253-273). Springer Berlin Heidelberg.

[47] GOLDREICH, O. (1998). Secure Multi-Party Computation. http://www. wisdom. weizmann. ac. il/~ oded/pp. html.

[48] Bujalkova, M. (2001). International Guidelines on Bioethics. Bratisl Lek Listy 27 (2), 117. doi:10.1136/jme.27.2.117

[49] Geis, J. R., Brady, A. P., Wu, C. C., Spencer, J., Ranschaert, E., Jaremko, J. L., et al. (2019). Ethics of Artificial Intelligence in Radiology: Summary of the Joint European and North American Multisociety Statement. J. Am. Coll. Radiol. 16 (11), 1516–1521. doi:10.1016/j.jacr.2019.07.028

[50] WHO (2021). Ethics and Governance of Artificial Intelligence for Health: WHO Guidance. Geneva, Switzerland: WHO.

[51] Azoulay, A. (2019). Towards an Ethics of Artificial Intelligence. United Nations. Available at: https://www.un.org/en/chronicle/article/towards-ethics-artificial-intelligence

[52] Köchling, A., and Wehner, M. C. (2020). Discriminated by an Algorithm: a Systematic Review of Discrimination and Fairness by Algorithmic Decision-Making in the Context of HR Recruitment and HR Development. Bus Res. 13 (3), 795–848. doi:10.1007/s40685-020-00134-w

[53] Mulshine, M. (2015). A Major Flaw in Google's Algorithm Allegedly Tagged Two Black People's Faces with the Word 'gorillas'. New York City: Business Insider.

[54] Noor, P. (2020). Can We Trust AI Not to Further Embed Racial Bias and Prejudice? BMJ 368, m363. doi:10.1136/bmj.m363

[55] Leslie, D., Mazumder, A., Peppin, A., Wolters, M. K., & Hagerty, A. (2021). Does “AI” stand for augmenting inequality in the era of covid-19 healthcare?. bmj, 372.

[56] Kamal, M., & Tariq, M. (2019). Light-weight security and blockchain based provenance for advanced metering infrastructure. IEEE Access, 7, 87345-87356.

[57] Gu, X., Sabrina, F., Fan, Z., & Sohail, S. (2023). A Review of Privacy Enhancement Methods for Federated Learning in Healthcare Systems. International Journal of Environmental Research and Public Health, 20(15), 6539.

[58] ABIDE. (n.d.). ABIDE I: Autism Brain Imaging Data Exchange. https://fcon_1000.projects.nitrc.org/indi/abide/abide_I.html (accessed Jan. 13, 2024).

[59] Cohen, J., Morrison, P., Dao L., COVID-19 Image Data Collection https://paperswithcode.com/dataset/covid-19-image-data-collection (accessed Jan. 13, 2024).

[60] Sambare, M. (n.d.). FER2013: Facial Expression Recognition 2013. https://www.kaggle.com/datasets/msambare/fer2013 (accessed Jan. 13, 2024).

[61] B. H. Menze et al., "The Multimodal Brain Tumor Image Segmentation Benchmark (BRATS)," in IEEE Transactions on Medical Imaging, vol. 34, no. 10, pp. 1993-2024, Oct. 2015, doi: 10.1109/TMI.2014.2377694.

[62] Human and Machine Vision Laboratory. (n.d.). The Mobifall and MobiAct Datasets. Harokopio University of Athens. https://bmi.hmu.gr/the-mobifall-and-mobiact-datasets-2/ (accessed Jan. 13, 2024).

[63] Reyes-Ortiz,Jorge, Anguita,Davide, Ghio,Alessandro, Oneto,Luca, and Parra,Xavier. (2012). Human Activity Recognition Using Smartphones. UCI Machine Learning Repository. https://doi.org/10.24432/C54S4K.

[64] Philip Schmidt, Attila Reiss, Robert Duerichen, Claus Marberger and Kristof Van Laerhoven, "Introducing WESAD, a multimodal dataset for Wearable Stress and Affect Detection", ICMI 2018, Boulder, USA, 2018

[65] Johnson, A., Bulgarelli, L., Pollard, T., Horng, S., Celi, L. A., & Mark, R. (2023). MIMIC-IV (version 2.2). PhysioNet. https://doi.org/10.13026/6mm1-ek67.

[66] Wang, L., & Wong, A. (2020). COVID-Net: A Tailored Deep Convolutional Neural Network Design for Detection of COVID-19 Cases from Chest X-Ray Images https://paperswithcode.com/dataset/covidx (accessed Jan. 14, 2024).

[67] Pogorelov, K., Randel, K. R., Griwodz, C., Eskeland, S. L., de Lange, T., Johansen, D., Spampinato, C., Dang-Nguyen, D.-T., Lux, M., Schmidt, P. T., Riegler, M., & Halvorsen, P. (2017). KVASIR: A Multi-Class Image Dataset for Computer Aided Gastrointestinal Disease Detection. In Proceedings of the 8th ACM on Multimedia Systems Conference (MMSys'17), 164-169. ACM. doi:10.1145/3083187.3083212

[68] Ramana,Bendi and Venkateswarlu,N.. (2012). ILPD (Indian Liver Patient Dataset). UCI Machine Learning Repository. https://doi.org/10.24432/C5D02C.

[69] Almadhor, A., Sampedro, G. A., Abisado, M., Abbas, S., Kim, Y. J., Khan, M. A., ... & Cha, J. H. (2023). Wrist-based electrodermal activity monitoring for stress detection using federated learning. *Sensors*, *23*(8), 3984.

[70] Raikwar, D (n.d.) Breast Cancer Diagnostic Dataset (BCD) https://www.kaggle.com/datasets/devraikwar/breast-cancer-diagnostic (accessed Jan. 15, 2024).

[71] Johnson, A.E.W., Bulgarelli, L., Shen, L. et al. MIMIC-IV, a freely accessible electronic health record dataset. Sci Data 10, 1 (2023). https://doi.org/10.1038/s41597-022-01899-x

[72] Das, A., Castiglia, T., Wang, S., & Patterson, S. (2022). Cross-silo federated learning for multi-tier networks with vertical and horizontal data partitioning. ACM Transactions on Intelligent Systems and Technology (TIST), 13(6), 1-27.

[73] Liu, C., Yang, Y., Cai, X., Ding, Y., & Lu, H. (2022). Completely Heterogeneous Federated Learning. arXiv preprint arXiv:2210.15865.

[74] Chen, T., Jin, X., Sun, Y., & Yin, W. (2020). Vafl: a method of vertical asynchronous federated learning. *arXiv preprint arXiv:2007.06081*.

[75] Flower. (2023, november 28). Flower Tutorial | Vertical Federated Learning using Flower [Video]. YouTube. https://www.youtube.com/watch?v=56-GvUaXKXo

[76] Bhatia, M., Wong, F.L., Cao, Y., Lau, H.Y., Huang, J., Puneet, P., Chevali, L.: Pathophysiology of acute pancreatitis. Pancreatology 5(2-3), 132–144 (2005)

[77] Xiao, A.Y., Tan, M.L., Wu, L.M., Asrani, V.M., Windsor, J.A., Yadav, D., Petrov, M.S.: Global incidence and mortality of pancreatic diseases: a systematic review, meta-analysis, and meta-regression of population-based cohort studies. The lancet Gastroenterology & hepatology 1(1), 45–55 (2016)

[78] Szatmary, P., Grammatikopoulos, T., Cai, W., Huang, W., Mukherjee, R., Hallo- ran, C., Beyer, G., Sutton, R.: Acute pancreatitis: diagnosis and treatment. Drugs 82(12), 1251–1276 (2022)

[79] Baeza-Zapata, A.A., García-Compeán, D., Jaquez-Quintana, J.O., ScharrerCabello, S.I., Del Cueto-Aguilera, Á.N., Maldonado-Garza, H.J.: Acute pancreatitis in elderly patients. Gastroenterology 161(6), 1736–1740 (2021)

[80] Hossin, M., Sulaiman, M.N.: A review on evaluation metrics for data classification evaluations. International journal of data mining & knowledge management process 5(2), 1 (2015)

[81] Johnson, A.E., Bulgarelli, L., Shen, L., Gayles, A., Shammout, A., Horng, S., Pollard, T.J., Hao, S., Moody, B., Gow, B., et al.: Mimic-iv, a freely accessible electronic health record dataset. Scientific data 10(1), 1 (2023)

[82] Ding, N., Guo, C., Li, C., Zhou, Y., Chai, X., et al.: An artificial neural networks model for early predicting in-hospital mortality in acute pancreatitis in mimic-iii. BioMed research international 2021 (2021)

[83] Mofidi, R., Duff, M.D., Madhavan, K.K., Garden, O.J., Parks, R.W.: Identification of severe acute pancreatitis using an artificial neural network. Surgery 141(1), 59– 66 (2007)

[84] Hameed, M.A.B., Alamgir, Z.: Improving mortality prediction in acute pancreatitis by machine learning and data augmentation. Computers in Biology and Medicine 150, 106077 (2022)

[85] Ren, W., Zou, K., Huang, S., Xu, H., Zhang, W., Shi, X., Shi, L., Zhong, X., Peng, Y., Tang, X., et al.: Prediction of in-hospital mortality of intensive care unit patients with acute pancreatitis based on an explainable machine learning algorithm. Journal of Clinical Gastroenterology pp. 10–1097 (2023)

[86] Reddi, S., Charles, Z., Zaheer, M., Garrett, Z., Rush, K., Konečný, J., ... & McMahan, H. B. (2020). Adaptive federated optimization. *arXiv preprint arXiv:2003.00295*.

[87] Tu, K., Zheng, S., Wang, X., & Hu, X. (2022, August). Adaptive Federated Learning via Mean Field Approach. In *2022 IEEE International Conferences on Internet of Things (iThings) and IEEE Green Computing & Communications (GreenCom) and IEEE Cyber, Physical & Social Computing (CPSCom) and IEEE Smart Data (SmartData) and IEEE Congress on Cybermatics (Cybermatics)* (pp. 168-175). IEEE.

[88] Çelik, E., & Güllü, M. K. (2023, October). Comparison of Federated Learning Strategies on ECG Classification. In 2023 Innovations in Intelligent Systems and Applications Conference (ASYU) (pp. 1-4). IEEE.

[89] Huang, W., Li, T., Wang, D., Du, S., Zhang, J., & Huang, T. (2022). Fairness and accuracy in horizontal federated learning. Information Sciences, 589, 170-185.

[90] Mondrejevski, L., Miliou, I., Montanino, A., Pitts, D., Hollmén, J., & Papapetrou, P. (2022, July). Flicu: A federated learning workflow for intensive care unit mortality prediction. In *2022 IEEE 35th International Symposium on Computer-Based Medical Systems (CBMS)* (pp. 32-37). IEEE.

[91] Georgoutsos, A. (2023). Analysis of Deep Federated Learning on Early Prediction of ICU Mortality Risk.

[92] Randl, K., Armengol, N. L., Mondrejevski, L., & Miliou, I. (2023, June). Early prediction of the risk of ICU mortality with Deep Federated Learning. In *2023 IEEE 36th International Symposium on Computer-Based Medical Systems (CBMS)* (pp. 706-711). IEEE.

[93] Çelik, E., Güllü, M.K.: Comparison of federated learning strategies on ecg classification. In: 2023 Innovations in Intelligent Systems and Applications Conference (ASYU). pp. 1–4. IEEE (2023)

[94] Netshakhuma, N. S. (2020). Assessment of a South Africa national consultative workshop on the Protection of Personal Information Act (POPIA). *Global Knowledge, Memory and Communication*, *69*(1/2), 58-74.

[95] Wani, S. U. D., Khan, N. A., Thakur, G., Gautam, S. P., Ali, M., Alam, P., ... & Shakeel, F. (2022, March). Utilization of artificial intelligence in disease prevention: Diagnosis, treatment, and implications for the healthcare workforce. In *Healthcare* (Vol. 10, No. 4, p. 608). MDPI.

[96] Kasula, B. Y. (2024). Advancements in AI-driven Healthcare: A Comprehensive Review of Diagnostics, Treatment, and Patient Care Integration. International Journal of Machine Learning for Sustainable Development, 1(1), 1-5.

[97] Sharrett, A. R., Ballantyne, C. M., Coady, S. A., Heiss, G., Sorlie, P. D., Catellier, D., & Patsch, W. (2001). Coronary heart disease prediction from lipoprotein cholesterol levels, triglycerides, lipoprotein (a), apolipoproteins AI and B, and HDL density subfractions: The Atherosclerosis Risk in Communities (ARIC) Study. Circulation, 104(10), 1108-1113.

[98] Davenport, T., & Kalakota, R. (2019). The potential for artificial intelligence in healthcare. *Future healthcare journal*, *6*(2), 94.

[99] Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., ... & Zheng, X. (2016). Tensorflow: Large-scale machine learning on heterogeneous distributed systems. *arXiv preprint arXiv:1603.04467*.

[100] OpenMined. (n.d.). PySyft: A library for encrypted, privacy-preserving machine learning. OpenMined. Retrieved from https://github.com/OpenMined/PySyft

[101] PaddlePaddle. (n.d.). Paddle FL: Federated Learning Framework based on PaddlePaddle. PaddlePaddle. Retrieved from https://github.com/PaddlePaddle/PaddleFL

[102] WeBank. (n.d.). Federated AI Technology Enabler (FATE). FATE. Retrieved from https://github.com/FederatedAI/FATE

[103] Owkin, & Linux Foundation for AI and Data. (n.d.). Substra. Substra. Retrieved from https://github.com/SubstraFoundation/substra

[104] He, C., Li, S., So, J., Zeng, X., Zhang, M., Wang, H., ... & Avestimehr, S. (2020). Fedml: A research library and benchmark for federated machine learning. *arXiv preprint arXiv:2007.13518*.

[105] Xie, Y., Wang, Z., Gao, D., Chen, D., Yao, L., Kuang, W., ... & Zhou, J. (2022). Federatedscope: A flexible federated learning platform for heterogeneity. arXiv preprint arXiv:2204.05011.

[106] Beutel, D. J., Topal, T., Mathur, A., Qiu, X., Fernandez-Marques, J., Gao, Y., ... & Lane, N. D. (2020). Flower: A friendly federated learning research framework. *arXiv preprint arXiv:2007.14390*.

[107] Ludwig, H., Baracaldo, N., Thomas, G., Zhou, Y., Anwar, A., Rajamoni, S., ... & Abay, A. (2020). Ibm federated learning: an enterprise framework white paper v0. 1. *arXiv preprint arXiv:2007.10987*.

[108] McKinney, W., & Team, P. D. (2015). Pandas-Powerful python data analysis toolkit. Pandas—Powerful Python Data Analysis Toolkit, 1625.

[109] Oliphant, T. E. (2006). Guide to numpy (Vol. 1, p. 85). USA: Trelgol Publishing.

[110] Pollard, T. J., Johnson, A. E., Raffa, J. D., Celi, L. A., Mark, R. G., & Badawi, O. (2018). The eICU Collaborative Research Database, a freely available multi-center database for critical care research. *Scientific data*, *5*(1), 1-13.

[111] Zhou, Q., & Sun, B. (2023). Adaptive K-means clustering based under-sampling methods to solve the class imbalance problem. Data and Information Management, 100064.

[112] Gafni, T., Shlezinger, N., Cohen, K., Eldar, Y. C., & Poor, H. V. (2022). Federated learning: A signal processing perspective. *IEEE Signal Processing Magazine*, *39*(3), 14-41.

[113] Hosmer Jr, D. W., Lemeshow, S., & Sturdivant, R. X. (2013). *Applied logistic regression*. John Wiley & Sons.

[114] Boyd, C. R., Tolson, M. A., & Copes, W. S. (1987). Evaluating trauma care: the TRISS method. Journal of Trauma and Acute Care Surgery, 27(4), 370-378.

[115] Quinlan, J. R. (1986). Induction of decision trees. Machine learning, 1, 81-106.

[116] Sivasree, M. S., & Sunny, T. R. (2015). Loan credibility prediction system based on decision tree algorithm. *Int. J. Eng. Res. Technol*, *4*(09), 825.

[117] Breiman, L. (2001). Random forests. *Machine learning*, *45*, 5-32.

[118] Kumar, M. S., Soundarya, V., Kavitha, S., Keerthika, E. S., & Aswini, E. (2019, February). Credit card fraud detection using random forest algorithm. In 2019 3rd International Conference on Computing and Communications Technologies (ICCCT) (pp. 149-153). IEEE.

[119] Noble, W. S. (2006). What is a support vector machine?. Nature biotechnology, 24(12), 1565-1567.

[120] Kurtulmuş, F., & Kavdir, I. (2014). Detecting corn tassels using computer vision and support vector machines. *Expert Systems with Applications*, *41*(16), 7390-7397.

[121] Taud, H., & Mas, J. F. (2018). Multilayer perceptron (MLP). *Geomatic approaches for modeling land change scenarios*, 451-455.

[122] Zhu, Q., Stolcke, A., Chen, B. Y., & Morgan, N. (2005, September). Using MLP features in SRI's conversational speech recognition system. In *Interspeech* (Vol. 2005, pp. 2141-2144).

[123] Grandini, M., Bagli, E., & Visani, G. (2020). Metrics for multi-class classification: an overview. *arXiv preprint arXiv:2008.05756*.

[124] Iwase, S., Nakada, T. A., Shimada, T., Oami, T., Shimazui, T., Takahashi, N., ... & Kawakami, E. (2022). Prediction algorithm for ICU mortality and length of stay using machine learning. Scientific reports, 12(1), 12912.

[125] Li, X., Gu, Y., Dvornek, N., Staib, L. H., Ventola, P., & Duncan, J. S. (2020). Multi-site fMRI analysis using privacy-preserving federated learning and domain adaptation: ABIDE results. *Medical Image Analysis*, *65*, 101765.

[126] Shome, D., & Kar, T. (2021). FedAffect: Few-shot federated learning for facial expression recognition. In Proceedings of the IEEE/CVF international conference on computer vision (pp. 4168-4175).

[127] Nalawade, S., Ganesh, C., Wagner, B., Reddy, D., Das, Y., Yu, F. F., ... & Maldjian, J. A. (2021, September). Federated learning for brain tumor segmentation using mri and transformers. In *International MICCAI Brainlesion Workshop* (pp. 444-454). Cham: Springer International Publishing.

[128] Park, Y., Schmidt, C. E., Batton, B. M., & Hauschild, A. C. (2024). Federated Random Forest for Partially Overlapping Clinical Data. *arXiv preprint arXiv:2405.20738*.

[129] Bettini, C., Civitarese, G., & Presotto, R. (2021). Personalized semi-supervised federated learning for human activity recognition. *arXiv preprint arXiv:2104.08094*.

[130] Ouyang, X., Xie, Z., Zhou, J., Huang, J., & Xing, G. (2021, June). Clusterfl: a similarity-aware federated learning system for human activity recognition. In *Proceedings of the 19th Annual International Conference on Mobile Systems, Applications, and Services* (pp. 54-66).

[131] Yang, Q., Zhang, J., Hao, W., Spell, G. P., & Carin, L. (2021, August). Flop: Federated learning on medical datasets using partial networks. In Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining (pp. 3845-3853).

[132] Matschinske, J., Späth, J., Bakhtiari, M., Probul, N., Majdabadi, M. M. K., Nasirigerdeh, R., ... & Baumbach, J. (2023). The FeatureCloud platform for federated learning in biomedicine: unified approach. *Journal of Medical Internet Research*, *25*(1), e42621.

[133] Lazzarini, R., Tianfield, H., & Charissis, V. (2023). Federated learning for IoT intrusion detection. *AI*, *4*(3), 509-530.

[134] Nilsson, A., Smith, S., Ulm, G., Gustavsson, E., & Jirstrand, M. (2018, December). A performance evaluation of federated learning algorithms. In *Proceedings of the second workshop on distributed infrastructures for deep learning* (pp. 1-8).

[135] Bavdekar, S. B. (2013). Pediatric clinical trials. *Perspectives in clinical research*, *4*(1), 89-99.

[136] Nistal-Nuño, B. (2022). Developing machine learning models for prediction of mortality in the medical intensive care unit. *Computer Methods and Programs in Biomedicine*, *216*, 106663.

[137] Pang, K., Li, L., Ouyang, W., Liu, X., & Tang, Y. (2022). Establishment of ICU mortality risk prediction models with machine learning algorithm using MIMIC-IV database. *Diagnostics*, *12*(5), 1068.

[138] Chia, A. H. T., Khoo, M. S., Lim, A. Z., Ong, K. E., Sun, Y., Nguyen, B. P., ... & Pang, J. (2021). Explainable machine learning prediction of ICU mortality. *Informatics in Medicine Unlocked*, *25*, 100674.

[139] Alghatani, K., Ammar, N., Rezgui, A., & Shaban-Nejad, A. (2021). Predicting intensive care unit length of stay and mortality using patient vital signs: machine learning model development and validation. *JMIR medical informatics*, *9*(5), e21347.

[140] Wang, T., Zhang, K., Cai, J., Gong, Y., Choo, K. K. R., & Guo, Y. (2024). Analyzing the Impact of Personalization on Fairness in Federated Learning for Healthcare. *Journal of Healthcare Informatics Research*, *8*(2), 181-205.

[141] EPIA. (2024). EPIA 2024 - 22nd EPIA Conference on Artificial Intelligence. https://epia2024.pt/ (accessed Jun. 28, 2024)