INSTITUTO
UNIVERSITÁRIO
DE LISBOA

Applying Federated Learning to a Covid-19 Anomaly Detection Pipeline

Susana Isabel de Carvalho Polido

Master's in Computer Engineering,

Advisor:
Ana Maria Carvalho de Almeida, PhD, Associate Professor,
Iscte Instituto Universitário de Lisboa

Co-Advisor:
Maurício Breternitz Jr., PhD, Invited Assistant Professor,
Iscte Instituto Universitário de Lisboa

October, 2023

TECHNOLOGY
AND ARCHITECTURE

Department of Information Science and Technology

Applying Federated Learning to a Covid-19 Anomaly Detection Pipeline

Susana Isabel de Carvalho Polido

Master's in Computer Engineering,

Advisor:
Ana Maria Carvalho de Almeida, PhD, Associate Professor,
Iscte Instituto Universitário de Lisboa

Co-Advisor:
Maurício Breternitz Jr., PhD, Invited Assistant Professor,
Iscte Instituto Universitário de Lisboa

October, 2023

Acknowledgments

This work was partially supported by Fundação para a Ciência e Tecnologia (FCT) under DSAIPA/AI/0122/2020 AIMHealth – AI- based Mobile Applications for Public Health Response, coordinated by the Information Sciences, Technologies and Architecture Research Center (ISTAR-Iscte) and projects UIDB/50008/2020 and UIDB/04466/2020.

Resumo

A pandemia de COVID-19 de 2020 espalhou-se rapidamente, sobrecarregando os sistemas de saúde e causando milhões de mortes em todo o mundo. Investigadores e cientistas concentraram esforços em encontrar soluções para ajudar a detetar e conter a propagação da doença, incluindo ferramentas alimentadas por modelos de aprendizagem automática. Entre os estudos efetuados, vários exploram diferenças entre sinais biométricos de pessoas que contraíram COVID-19, recolhidos antes e depois da infeção, em busca de padrões que possam ajudar a detetar a doença o mais rápido possível. Em particular, em sinais relacionados com a frequência cardíaca recolhidos através de dispositivos como smartwatches. Esses estudos resultaram em algumas ferramentas de deteção, mas precisam que os utilizadores tenham dados anteriores a contraírem a doença para serem usados, contendo elementos de personalização com base nos dados saudáveis para funcionarem. Mas e se um novo utilizador não possuir dados saudáveis? Poderá um modelo treinado com dados de uns indivíduos detetar a doença noutros, com sucesso? Este trabalho explora essa situação transportando uma ferramenta de deteção de anomalia personalizada para um ambiente de Aprendizagem Federada, a fim de ver seu comportamento em dados de indivíduos que participaram no treino do modelo e de novos indivíduos, incluindo dados recolhidos por dispositivos hospitalares de indivíduos internados numa unidade de cuidados intensivos, que dificilmente contém dados saudáveis. Embora os modelos resultantes, em média, não detetem mais anomalias verdadeiras do que os modelos personalizados, seu desempenho é semelhante quando aplicado a dados de indivíduos de treino e novos indivíduos.

Palavras-Chave: Deteção de Anomalias, Aprendizagem Federada, Modelo na Saúde, Aprendizagem automática

Abstract

The COVID-19 pandemic of 2020 spread around the world fast, overwhelming healthcare systems and causing millions of deaths all over the globe. Researchers and scientists rushed to find tools to help detect and contain the spread of the disease, including automatic ones powered by machine learning models. Amongst the efforts, several studies focused on exploring differences between biometric signals in people with the disease collected before and after the infection, in search of patterns that can help detect it as soon as possible. In particular, heart rate related signals collected via devices such as smartwatches. These studies have resulted in some detection tools, but they always require users to have data from before the infection occurred in order to be used and often contain personalization based on this healthy data. But what if a new user has not yet collected healthy data? Can a model trained with the data of other individuals successfully detect the illness on a novel one? This work explores that situation by taking an individual based anomaly detection and transporting into a Federated Learning environment in order to see its behavior on the data from individuals that trained the model and novel individuals, including data collected via hospital devices from individuals admitted to the intensive care unit which is unlikely to contain healthy data. Although the resulting models, on average, did not detect more true anomalies than the personalized ones, their performance is similar when applied to the training individuals and novel ones.

Keywords: Anomaly Detection, Federated Learning, Health Model, Machine Learning

Contents

List of Figures

List of Tables

Acronyms

AI: Artificial Intelligence
AIM-HEALTH: AI-based mobile applications for public health response
AUC: area under curve
BMI: body max index
BNB: Bernoulli naive bayes
bpm: beats per minute
CNN-VAE: Convolutional Neural Network-based Variational Autoencoder
COVID-19: coronavirus disease 2019
csv: comma-separated value
DT: decision tree
FCT: Foundation for Science and Technology
FedAvg: FederatedAveraging
FedAvgM: Federated Averaging with Server Momentum
FedNova: Federated Normalized Averaging
FL: Federated Learning
FLWR: Flower.Dev
GBM: gradient-boosting machines
GMM: gaussian mixture model
HR: heart rate
HROS: heart rate over steps
HROS-AD: heart rate over steps anomaly detection
HRV: heart rate variability
HSM: Hospital de Santa Maria
ICU: intensive care unit
ISTAR: Information Sciences Technologies and Architecture Research Center
K-NN: k-nearest neighbor
LASSO: least absolute shrinkage and selection operator
LDA: linear discriminant analysis
LOF: Local Outlier Factor
LR: Logistic Regression
LSTM: Long Short-Term Memory
MAE: mean absolute error
MD: Mahalanobis Distance-Minimized Covariance Determinant
ML: Machine Learning

MLP: Multilayer Perception
MSE: mean squared error
MTE: MinMax Threshold Estimation
NN: Neural Network
non-IID: non-independent and identically distributed
OC-SVM: One-Class SVM
PRISMA: Preferred Reporting Items for Systematic Reviews and Meta-Analyses
RF: random forest
RHR: resting heart rate
RHR-AD: resting heart rate anomaly detector
RHR-Diff: resting heart rate difference
RR: respiratory rate
RT-PCR: Reverse transcription-polymerase chain reaction
RW: Rolling Window
SCAFFOLD: Stochastic Controlled Averaging
SpO2: oxygen saturation
STE: Statistical Threshold Estimation
SVM: support vector machine
WHO: World Health Organization
XGBoost: extreme gradient boosting

CHAPTER 1

Introduction

First detected in December of 2019, the coronavirus disease 2019 (COVID-19) quickly spread around the globe, gaining the status of a pandemic in 11 March of 2020 from the World Health Organization (WHO). Its vast success is, in part, due to its ability to be infectious before the carriers feel sick, with several never developing symptoms. But studies have found changes in physiological metrics track-able by commercial wearable devices can occur before symptom onsets, which could be used to warn users of a potential infection [**2, 3**].

This chapter is divided into background and motivation, research goals and contributions and the methodology followed during the development of this work.

1.1. Background and Motivation

Reverse transcription-polymerase chain reaction (RT-PCR) tests are used to detect a COVID-19 case but only work 3 days after the infection, require material and a laboratory which entails costs, rarely detect pre-symptomatic carriers and struggle to detect the disease at an early stage [**4**]. Meanwhile, the interest in wearable sensors, such as smartwatches, increased during the pandemic and several are capable of not only collecting data but also analyze it and provide feedback [**5**]. Several authors and studies suggest these devices and the data they collect can be used to identify potentially infected individuals before they develop symptoms, allowing the detection of asymptomatic carriers without testing an entire population [**2, 6**]. Having an Artificial Intelligence (AI) able to analyze an individual's health data captured by these devices and alert them something might be wrong before they feel ill enables the individual to take preventive measures to protect those around them, such as to enter in isolation or to wear a mask in the case of COVID-19 and other illnesses with similar spread, and to seek guidance from an health professional. But training a good Machine Learning (ML) algorithm requires lots of data which can be difficult to gather both due to the recent nature of the virus and disease and data privacy laws. Federated Learning (FL) is an ML approach where a central global model can be trained using data from several models at different locations without needing to transfer the said data outside of its original location [**7**]. As such, it respects privacy issues while building on distributed computation to capitalize on several and varied data.

This work is integrated in the Information Sciences Technologies and Architecture Research Center (ISTAR) project AI-based mobile applications for public health response

(AIM-HEALTH) funded by the Foundation for Science and Technology (FCT) [**8**][1]. AIM-HEALTH aims to create a smartphone app embodying a trustworthy secure AI distributed and service-based platform. This AI system employs a FL architecture, a technique that enables ML algorithms deployed across multiple decentralized edge devices or servers holding local data samples to collaboratively train a global model. The user's data is collected and stored using their mobile devices and locally processed by the global model, then locally used to train the new model iteration. Upon authorization, the new/updated model is downloaded into the edge devices, in an exchange between the decentralized devices and a server. In particular, this work explores the proposals for the detection of anomalies in a time series signal - resting heart rate (RHR) - collected by wearable devices, where differences between data collected during healthy periods and periods of sickness, namely COVID-19 infection, have been found [**6, 9, 10**].

1.2. Research Goals and Contributions

Goals The main goal of this project is to implement a FL framework to create a Neural Network (NN) model able to identify health anomalies using biometric sensing devices (such as smartwatches, smart wristbands, or medical sensing devices), to enable early abnormal symptomatology alert indicative of a possible COVID-19 infection and compare its results to those obtained by locally or pooled dedicated trained models.

On the other hand, and given that the AIM Health project partners with Hospital de Santa Maria and Professor Luís Rosário, a medical doctor from this hospital and Professor in the University of Lisbon, we portrayed the possibility of exploring the development of an automated system to be used for the detection of a possible COVID-19 infection on a patient under hospital care, especially those patients who are in a intensive care unit (ICU) unit, which are generally open spaces and thus more people are exposed to the virus spread. Given that, any patient that is admitted to an hospital needing treatment for an urgent or emergent disease condition (be it multiple fractures, coronary disease, or other) has no baseline whatsoever to train an individual model, one other goal consists on understanding if pre-trained models from different individual sensors data can be applied to novel individual's data, without any fine-tunning (since no data for training is available), in particular in the case of patient's data collected via hospital devices during normal hospital activities in an ICU.

Towards this goals, this study begins with an exploratory effort for the implementation of an anomaly detection pipeline that receives biometric data that can be collected by wearable or mobile devices, and determines if the person the data belongs to might be infected with COVID-19, with a ML model that is trained via FL, that is, without sharing the biometric data with any external devices and aggregating the individual parameters training into a central model. This approach results are compared with the results from the individual models, after which we investigate on the feasibility of applying this central model to hospital patient biometric data acquired by medical sensing devices.

[1]https://istar.iscte-iul.pt/portfolio-posts/ai-based-mobile-applications-for-public-health-response/

Research questions Given the previously stated general goals, this thesis aims to answer the following research questions:

RQ1: Our first question regards the investigation of the related literature and is a compound one: What is the state-of-the art regarding ML models to detect COVID-19 infections using wearable or mobile devices? What FL applications exist, namely within the context for health anomaly detection of COVID-19 infection? If any, do they make use of data acquired by wearable or mobile devices or sensing data collected with medical devices?

RQ2: Is FL a feasible approach for personal health anomalies detection?

RQ3: Are pre-trained models accurate for anomaly detection in patients other than the users they were trained upon?

RQ4: Are FL aggregated models feasible for the detection of anomalies in a new user?

Contributions The main contributions of this work are two-fold: on the one hand, the exploration of shifting an anomaly detection pipeline, originally developed to be trained by a single individual's data for risk alert of that same individual, into the FL paradigm to understand its validity within this setting. On the other hand, we find that detection anomaly models should be further investigated towards their application in new domain data, that is, new data from new users and acquired in a completely different environment. Our experimentation with the data coming from an ICU of a Portuguese hospital, that is, data collected during a hospital's daily activities from its patients, allows to perceive the potential of pre-trained models using individual data have towards the construction of a global model to be used in a hospital to help with automated patient screening and early alert system.

Finally, we note that some of the exploratory results next described have been submitted to a relevant conference in the area with peer-revision scheme.

1.3. Methodology

As previously described, this project has as main goal that of exploring how an anomaly detection pipeline created to be trained with the data of a single person and detect possible COVID-19 infections on its user behaves when transported into a FL approach and, on a second note, see how it performs with data collected in an hospital during normal daily activity.

With this in mind, this work follows the CRISP-DM[2] methodology, which consists of phases of understanding what is wanted (business understanding), what data is necessary or available (data understanding), how the data must be changed for modeling (data preparation), and the training and evaluation of models. The first cycle through these phases is straightforward: this work aims to implement an anomaly detection pipeline to detect possible COVID-19 infections using data that can be collected by wearable devices, such as smartwatches, while resorting to FL. As such, it is necessary to obtain

[2]https://www.datascience-pm.com/crisp-dm-2/

data that can be collected by those devices related which has been collected during periods of COVID-19 illness and health. There are publicly available datasets and models focused on this premise that do not implement FL, so these are used for a first round of modeling with the FL to see its affects. These results are to be complemented by data collected in a hospital, but exploring it proves it lacks characteristics necessary to apply the chosen pipeline. As it is imperative the final pipeline be applicable to the hospital data, several experiments that fall on going back and forth the different phases of the CRISP-DM are conducted.

In general, chapters 1 and 2 refer to the business understanding phase, while chapter 3 focuses on the data understanding and data preparation. The modeling and evaluation phases are present in chapter 4 and chapter 5 contains a brief explanation of the files that make up the developed final prototype.

This work uses only the language Python[3] and Jupyter Notebooks[4]. All data exploration pertaining to the data provided by the Hospital de Santa Maria is done in a server provided by the AIM-Health project with the Pandas[5] and Dask[6] libraries. All other code is run in a personal laptop. To develop the models and FL experiments, the Tensorflow[7] library and Flower.Dev (FLWR)[8] framework are used. The visualization graphics are built with the Matplotlib[9] and Seaborn[10] libraries.

[3]https://www.python.org/
[4]https://jupyter.org/
[5]https://pandas.pydata.org/
[6]https://docs.dask.org/en/stable/dataframe.html
[7]https://www.tensorflow.org/
[8]https://flower.dev/
[9]https://matplotlib.org/
[10]https://seaborn.pydata.org/

CHAPTER 2

Literature Review

The literature review follows a simplified version of the Preferred Reporting Items for Systematic Reviews and Meta-Analyses (PRISMA) [**11**] methodology.

Three different queries were conducted on the Web Of Science repository to find literature to answer the research questions.

The first focuses on how FL has been applied to COVID-19, with the query 2.1, and returned 98 results. Attempts to narrow the search by adding the keywords "wearable device" or "biometric data" returned 2 and 0 results, respectively, so the full 98 were examined. None of the found records are about the detection of COVID-19 using data from wearable devices.

$$\text{(ALL=(covid-19) AND ALL=(federated learning))} \tag{2.1}$$

The second query 2.2 obtained 69 records, several of which are about the use of data from wearable devices to train ML models during the pandemic or wearable devices not available to the public. The third query 2.3 retrieved 16 records.

$$\text{(ALL=(covid-19) AND ALL=(wearable device) AND ALL=(machine learning))} \tag{2.2}$$

$$\text{(ALL=(smartwatch) AND ALL=(covid-19 detection))} \tag{2.3}$$

The inclusion criteria for the records are:

- Focus on data that can be easily collected by wearable devices available to the public
- Compares models trained with FL to models trained locally

Exclusion criteria are:

- Not about COVID-19 detection
- About COVID-19 at the population level instead of the individual level
- Focus on the creation of COVID-19 detection ML models without data from wearable devices available to the public

Known biases of this literature research are:

- Using only the Web of Knowledge repository
- Only including records in English
- Only including records with free full text available

No data filtering is applied because the inclusion of the keyword "covid-19" in all queries guaranteed all records are from 2019 and afterwards. Figure 1 contains a summary of the selection process of the records selected for this review conducted up to January of 2023.

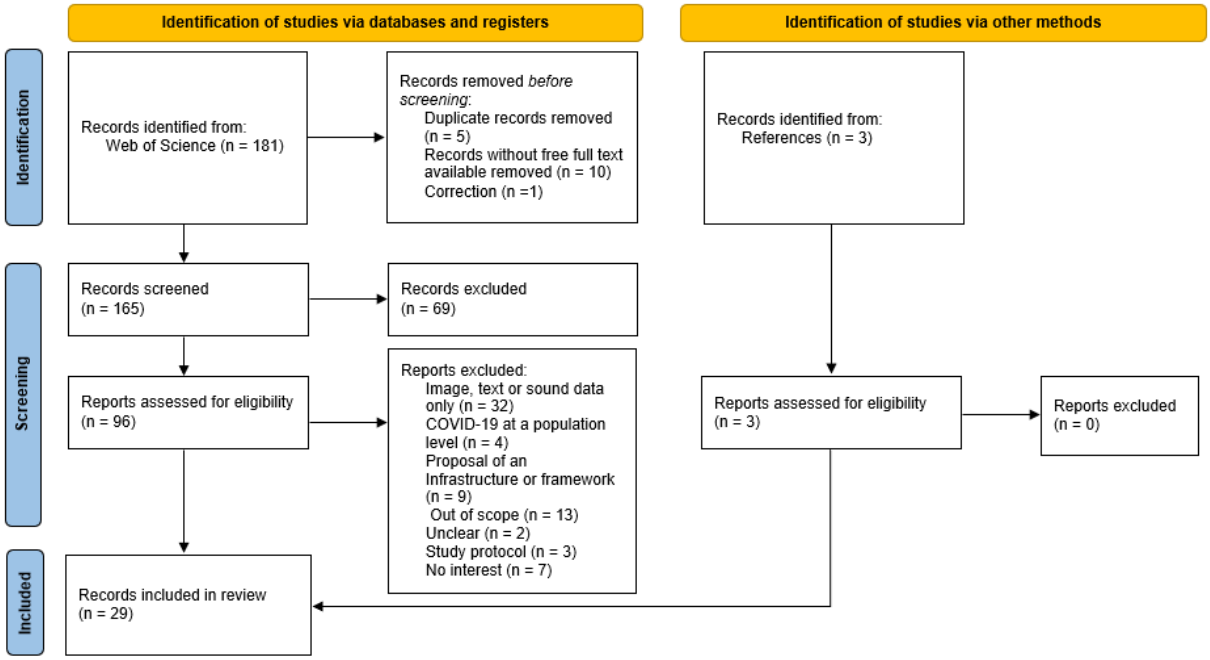


FIGURE 1. Literature review flow diagram, adapted from PRISMA

A follow up search of each query in September of 2023 with index date of "year to date" found 61, 26 and 6 new papers, respectively, of which only 2 from the third query are included.

Articles about the aggregation functions made available by the FL framework used in the course of this work and their respective articles are explored at the end of this chapter.

No works about applying FL to data collected via wearable devices in order to detect if the user has COVID-19 were found during the literature review. Most focus on classifying images as belonging to patients with COVID-19 or not, with the few that utilize data from wearable devices focusing on predicting patients' outcomes and necessities with the help of supplemental data. Outside of the FL scope, most works that utilize data collectable by wearable devices focus on heart rate (HR), activity and sleep data plus symptoms. When several types of ML algorithms are explored, NN do not present the best results. Statistical works focus on finding differences between an individual's healthy and sick data, while automatic anomaly detection works train ML models with an individual's data to detect anomalies in that individual's data.

A short explanation of anomaly detection was added to this work after the final presentation, at the jury's request. This is based on a search containing the keywords "anomaly detection", "time series" and "survey". The search, conducted on November 24 of 2023,

returned 132 results of which 5 are used, and is supplemented by 2 articles provided by the advisors.

2.1. Background Concepts

This section explains what is meant by wearable devices for this project and how they might be used to detect illnesses, then does a short presentation of the concept of FL followed by a subsection about anomaly detection.

2.1.1. Wearable Devices

Wearable devices, as their name implies, are devices people can wear and are able to track a variety of body physiology. This project focuses on wrist and handheld devices such as smart watches, wrist bands and rings which are usually used for fitness tracking and allow for contactless communication and send notifications to the user. Although how and what data a device captures varies between brands and models, they often allow to keep track of workout time, burnt calories, sleep time, step count and heart rate [**12**].

Viral infections can increase a person's heart rate and sleep duration and decrease its quality, and blood oxygen saturation is significantly lower in severe COVID-19 patients, along with an increased respiratory rate (RR), so wearable devices that track these metrics, when coupled with AI techniques, can help detect potential infected users and warn them to take a test and isolate [**3, 13**].

It should be noted the incorrect placement of the device, movement during measurements and even the skin type of the wearer can influence the collected data [**14**] and most wearable devices don't have a temperature sensor and their oxygen saturation (SpO2) sensor, when they exist, tend to have low accuracy [**3, 15**].

2.1.2. Federated Learning

FL is a term coined by Google that represents the decentralized training of a ML model while sharing minimum amounts of information and requires two types of users: Clients and Server.

The clients are the users who contain data that can be used to train the model. Each of them has a copy of the global model and furthers its training with its local data, which is never shared with any of the other users. Once the training of the local model is complete, its parameters are sent over to the Server who aggregates the local model of all the clients into a new global model.

Besides aggregating the models, the server is also responsible for starting a new round of training, deciding how many clients should participate and sharing the global model with the clients. However, due care must be taken since the performance of a FL trained model and the time necessary to train it can be influenced by the aggregation algorithm [**16**].

The main benefit of FL is improved data privacy which opens the possibility to have multiple entities working collaboratively to train a model with a larger amount of data that could not be joined otherwise [**7, 17**]. But it's distributed nature and reliance on

internet connection makes it more vulnerable to security risk such as model inversion attacks (where the model parameters are used to recreate an individual sample), man-in-the-middle attacks (where model updates get intercepted and replaced), and adversarial attacks (where fake data is introduced in the training data). These issues can be circumvented by adding Gaussian noise to the model parameters, applying encryption or block-chain technologies and using an adversarial model [**18**].

FL can be classified into three types depending on the similarity of the data between the different clients: horizontal, vertical and transfer learning. In horizontal FL the data has the same features across all clients, in vertical FL different clients have different information (features) regarding the same samples and, in transfer learning, neither the samples nor features are the same.

Since each client trains a local model based on only its own data, the local models might be trained with non-independent and identically distributed (non-IID) data which can lead to diverging training [**19**]. The studies found suggest FL trained models present better results than models trained with just local data [**17, 20, 21**], but that gathering all the data and training one single model can yield a better performance [**17, 21**].

2.1.3. Anomaly Detection

Anomaly detection is the process of identifying anomalous data, that is data that appears odd in relation to the remaining data [**22, 23**]. This process is used in several domains but the algorithms that work best vary greatly. An anomaly might represent an important event such as heart failure or incidents of financial fraud [**24, 23, 25**]. This work focus on anomaly detection in time series, which are sequences of data ordered by time [**24**]. Time series can be univariate or multivariate depending if they contain only one or more variables ordered by time, respectively [**24, 25**].

In the context of time series, anomalies can be classified as point, subsequence or sequence anomalies depending on their granularity. Point anomalies are data points that deviate from other points in the time series, and can be further divided into global or local anomalies depending on being odd within the entire time series or just for neighbouring points. These are very short in duration, with the data quickly reverting back to normal values [**25**]. Subsequence anomalies are a sequence of data points that are not odd when isolated but together form a strange pattern. Sequence anomalies occur in multivariate time series when one of the variables shows a strange behaviour when considered with the other variables [**26**].

Anomalies can also be classified in terms of their behaviour as point, contextual and collective. In this case, point, or global, anomalies are data points that appear strange when considering the full time series while contextual, or local, anomalies are points that are strange given its neighbouring values. Collective anomalies are when a sequence of points forms an odd pattern [**24, 22, 26, 25**].

Anomaly detection algorithms can be statistical in nature or take advantage of machine learning or data mining techniques [**24, 22**] and can output either an anomaly score, that

indicates how odd the sample is, or a label. How an anomaly score is calculated varies depending on the algorithm. Labels can be binary (normal and anomaly) or multi-class in nature. Often the algorithms calculate a score then label data as normal or anomalous based on a threshold [**25, 27**] Regarding how the learning occurs, anomaly detectors can be classified as supervised, semi-supervised or unsupervised.

Supervised learning is used when the training dataset contains both normal and anomalous labelled data. This method is the least popular due to high costs of obtaining the labelled data and issues with the detection of new types of anomalies. Often, supervised learning tends to struggle with multivariate data [**24, 26, 23**]. NN, support vector machine (SVM) and decision trees are some examples of techniques that use supervised learning [**26**].

Semi-supervised learning requires the availability of normal data to train the anomaly detection mechanism [**24**]. With this type of learning, the algorithms learn from normal data and will behave strangely or poorly when in the presence of anomalous data. Some methods that apply semi-supervised learning are autoencoders, gaussian models and GANs [**22, 28**].

Unsupervised learning assumes anomalous data is less frequent than normal data and presents different behaviours and patterns. Methods that apply unsupervised learning do not require the labelling of any data and often focus on splitting the data [**24, 23**]. This is the most common type of learning and common techniques employed are K-nearest neighbours, clustering algorithms, and statistical methods [**22, 26, 23**].

Anomaly detection algorithms can also be divided by families of methods such as forecasting, reconstruction, encoding, distance, distribution and isolation tree methods [**24**]. Forecasting methods use a model that is continually updated to predict, or forecast, a number of time steps based on the current context window. The predicted values are then compared to the real values to see if they are anomalous. The methods in this family can differ greatly on the model, type of learning and anomaly score calculation employed, but they tend to use semi-learning. These can also be called predictive methods.

Reconstruction methods are methods that build a model from normal data that encodes subsequences of the data in a low dimensional latent space. To detect anomalous data, the latent space is reconstructed and the resulting subsequence is compared to the original values with the assumption the reconstruction of anomalous data will be worse than the reconstruction of normal data. These tend to be semi-supervised methods and autoencoders are part of this family. GANs can also be used in this context [**28**].

Encoding methods also encode time series subsequences into a low latent space but the detection is obtained from this representation and not from a reconstruction of the signal.

Distance methods tend to be unsupervised and use distance metrics to compare points or subsequences of time series with each other, with the assumption that anomalous data will show a larger distance than normal data. Clustering and nearest neighbour methods

are examples of methods in this family. With clustering algorithms, data that is far from dense clusters is considered anomalous.

Distribution methods fit a distribution model to the data or estimate its distribution and anomalies are detected based on probabilities as they are often found in the extremes of the distributions. These methods are usually unsupervised but some semi-supervised approaches exist by fitting the distribution to known normal data and then seeing how other data compares to it. Several of these algorithms use gaussian distributions and histograms.

Isolation tree methods build random trees that isolate the data, be it points or subsequences, with the assumption anomalous data is easier to separate then normal and will be closer to the root of the tree.

No family of algorithms is superior to the other, with each displaying methods that perform better in some cases than in others [**24, 25, 27**]. Using more than one approach can help to achieve better results [**26, 25**].

This work will focus on univariate time series, with a reconstruction based anomaly detection that returns a label (normal or anomaly).

2.2. Related Works

This section is divided into works that use data gathered from wearable devices to detect COVID-19 infection and works that apply FL to COVID-19 related data.

2.2.1. Identifying COVID-19 infection with wearable device data

Fifteen different studies that mention detecting a COVID-19 infection through wearable device data were identified.

The authors in [**29**] propose a crowd-sensing framework where people infected with COVID-19 are detected and tracked in real-time using wearable devices data and security cameras. They train four ML algorithms (Logistic Regression (LR), SVM, decision tree (DT) and Bernoulli naive bayes (BNB)) on a dataset with an even number of COVID-19 positive and negative records that consists on SpO2, pulse rate in beats per minute (bpm), temperature and the COVID-19 status obtaining the lowest false negative rate with the SVM. D. Barbhuiya et al. [**30**] propose a remote health monitoring system where patient's data is collected and analyzed on the edge device to detect anomalies which might indicate a healthcare emergency, suggesting Robust Covariance, One-Class SVM (OC-SVM), Isolation Forest and Local Outlier Factor (LOF) models to find anomalies/outliers in collected data. They also test SVM, linear discriminant analysis (LDA), gaussian mixture model (GMM) and NN to detect COVID-19 on data consisting of diastolic blood pressure, galvanic skin response, pulse, SpO2 and body temperature, some of which can be monitored by some commercially available wearable devices, but achieve poor results with SVM and LDA failing to detect any COVID-19 positive cases and GMM and NN achieving a precision score of 4,33%. They consider the LOF model's ability to detect anomalies acceptable.

G. Quer et al. did a research study where they collect demographic data, symptoms and data from smartwatches, specifically daily RHR (calculated by the device), daily sleep duration in minutes (from noon to noon of the next day) and daily steps taken [**31**]. The baseline data is extracted from 21 days to 7 days before reported symptom onset and the test data from the day of symptom onset to 7 days afterwards. For each datatype, a metric based on removing the median of the baseline data from either the max value of the test data RHR or the mean (sleep and steps) is calculated. Another metric is calculated based on demographic and symptom data, and the 3 sensor-based metrics are merged into a single one which is further added to the symptom metric to achieve an overall metric. They conclude the RHR metric doesn't differentiate significantly between COVID-19 positive and COVID-19 negative patients, with it being better to use either the Sleep metric or the Activity metric. Using all 3 sensor metrics is preferable to using only one or the symptom metric. The best results are achieved using the overall metric.

T. Mishra et al. [**9**] recruit for a study people with confirmed or suspected COVID-19 infection or with a high risk of exposure to the disease and ask them to wear a fitness tracker as much as possible and fill a daily survey to track symptoms and their severity, tests and diagnoses of both COVID-19 and other illnesses and recovery dates. Participants are also asked to provide demographic information, medical history, and current COVID-19 status during enrollment. Of the participants, they select the data collected by Fitbit from 32 COVID-19 positive individuals with either a positive test or symptom onset data, 15 patients with another illness and 73 healthy individuals, plus 7 other individuals from another study, to focus on. They develop three methods to detect anomalies in physiological data: resting heart rate difference (RHR-Diff), heart rate over steps anomaly detection (HROS-AD) and resting heart rate anomaly detector (RHR-AD). The first focuses on detecting time intervals of elevated RHR when compared with the average daily curve using a 28 days sliding window, considering an anomaly is occurring whenever the elevated intervals last for over 24h. The second is an unsupervised method where a gaussian density estimation classifies the heart rate over steps (HROS) data as normal or an anomaly. The third is the same as the second but uses the RHR instead. They applied these methods to the 32 COVID-19 positive individuals, detecting 26 anomalies within a window of 14 days before and 7 days after COVID-19 symptom onset or diagnoses, with a median of 4 days before and 7 days before respectively. It should be noted that the 15 individuals with other illnesses also showed increases of RHR near symptom onset, meaning this is a general signal of respiratory illness. They also create a real-time online detection of early COVID-19 method named CuSum that accumulates deviations of elevated RHR using the data from the previous 28 days as a baseline. To reduce the number of alarms, the RHR must stay elevated for over 24h. When applying this method to the selected 120 individuals, 63% of the COVID-19 positive cases received alarms, as did 9 out of the 15 individuals with other illnesses. Some alarms of shorter duration are also triggered for the 73 healthy individuals, likely resulting of other events associated

with higher heart rate such as holidays. They notice a reduction on the number of steps taken and increase of sleep duration around the time detected by RHR-Diff.

A. Alavi et al. [**6**] follow up the work of T. Mishra et al. [**9**], by developing an online real-time alert system with 3 different detection algorithms: the proviously presented RHR-AD and CuSum, and NightSignal. Only the latter's results were made available to the participants of the study, via daily red and green alerts, and participants were expected to complete survey per alarm about diagnosis, symptoms, and activity. This study considered data from both Fitbit and Apple Watch devices, although only the former's data can be used with the RHR-AD and CuSum methods. NightSignal employs a state machine and only takes into consideration overnight RHR (midnight to 7 in the morning) to avoid events that increase the RHR such as stress or exercise. It sends a red alert whenever the overnight RHR is at least 4 bpm above the baseline, calculated by finding the media of all overnight RHR up to the current night. They consider true positives the number of cases who received a red alert between 21 days before and the symptom onset or diagnoses date, false negatives those who didn't receive red alerts during the same period, true negatives as the number of green alerts received between 21 days before and up to a negative COVID-19 test or the entire study period in the case of untested participants, and false positives the number of red alerts sent in these same conditions. NightSignal detected 80% of the Covid-19 infections, CuSum 72% and RHR-AD 69%, all with similar rates of false positives. The red alerts received by healthy individuals tended to last longer than the true positive ones.

H. Cho et al. [**32**] use part of the data made available by T. Mishra et al. (29 out of the 32 COVID-19 infected individuals) to train OC-SVM models, focusing on the early detection of COVID-19, decreasing the minimum period sampling to be able to detect an anomaly and the suppression of false positives. All experiments are done with OC-SVM and Mahalanobis Distance-Minimized Covariance Determinant (MD), which is what the authors in [**9**] used according to H. Cho et al., with both RHR and HROS. In terms of detecting COVID-19, the OC-SVM approaches detected more and earlier positive cases than their MD counterparts while detecting fewer outliers. To decrease the period necessary to collect data before the model can begin to detect anomalies, H. Cho et al decrease the moving average window size from the original 400 hours, used by T. Mishra et al., to 350 and 300 for both methods and both sets of data for a total of 12 experiments. Decreasing the window from 400 to 350 improved the time between the anomaly detection and the symptom onset in 4 of the 3 models (HROS-OC-SVM only saw an increase with 300 hours, which didn't improve the other models). Decreasing the window also improved accuracy and decreased the minimum training period in days (the HROS always requires less time because it has more data per day). To test which approaches detect fewer false positives, they applied the 12 different models to the data of healthy individuals collected by T. Mishra et al.. The model who detected the fewer number of outliers was RHR-OC-SVM with a window size of 350.

Both G. Bogu and M. Snyder [**1**] and F. Abir et al, 2022 [**33**] create Long Short-Term Memory (LSTM) based autoencoders with part of the individuals' data from the study done by T. Mishra et al. (individuals who had less than 20 days of information before symptom onset were removed). Because the publicly available dataset contains the raw heart rate and steps data and not the RHR, they calculate it by averaging the HR in 1 minute, merging it with the steps data and then removing all samples where the number of steps is not 0 for 12 consecutive minutes. They apply a moving average of 400 minutes and aggregate the data into 1 hour samples with the mean. They select the data up to 20 days before the symptom onset of each individual, use data augmentation techniques, and split these data points 95%/5% to obtain a train and a validation sets. All the remaining data is used for test sets. Each individual's data is used to train a different model. Both use the loss of the reconstructed train and validation (baseline) samples to define thresholds to detect if a test sample's loss indicates an anomaly. G. Bogu and M. Snyder use the max mean squared error (MSE) found in the baseline, while F. Abir et al, 2022 add Kullback-Leibler Divergence to the MSE and define two different thresholds: Statistical Threshold Estimation (STE) where the threshold is three standards deviations above the mean of the baseline loss, and MinMax Threshold Estimation (MTE), which is the approach used in the other study. The model in the first of these 2 studies detected 14 cases before symptom onset, 9 after and failed to detect 2 cases amongst the COVID-19 positive cases. Within the other illness cases, it detected anomalies in 7 individuals before symptom onset, 2 after and didn't detect any for the remaining 2. Anomalies were detected in 44 healthy individuals pre onset and in 15 post onset. The second study's MTE detected 44% of the cases before symptom onset, 44% after and failed to detect 12% while the STE detected 80% before and 20% after, with both approaches achieving a better performance than the other model.

F. Abir et al., 2023 [**34**] follow up their work by replacing the model of their anomaly detection framework with a Convolutional Neural Network-based Variational Autoencoder (CNN-VAE) model with LSTM embeddings and pre-training both the CNN-VAE and LSTM components with data from 67 healthy individuals found in the datasets made available by T. Mishra et al. and A. Alavi et al., before finetuning it to each of the COVID-19 infected individual's baseline data. They also increase the rolling window to smooth the data from 400 to 1600 samples and apply linear interpolation after the hour resampling, followed by another rolling average, of 10 samples, to remove spikes caused by interpolation. Applying this to the data of 68 individuals with COVID-19 found in the public datasets achieved a precision of 0.70 and recall of 0.53 when testing on samples from 20 to 10 days prior to symptom onset (considered healthy) and from symptom onset to 14 days after (anomalous samples). They experiment with other test ranges and conclude the current division of samples based on the symptom onset day found in the literature might not be ideal.

The authors in [**35**] also focus on the dataset from [**9**], selecting data from two windows of time, before and after illness (or, in the case of healthy individuals, from an earlier and later time), from which they extract 50 features used to train SVM, LR, DT, random forest (RF) and extreme gradient boosting (XGBoost) models to classify individuals as COVID-19 positive or negative or as COVID-19 positive, infected with another illness or healthy. Their most accurate model is the k-nearest neighbor (K-NN), independently of the window size and type of classification (binary or multi-class) but the best accuracy is achieved by binary classification with a window size of 5 days.

M. M. H. Shandhi et al. [**10**] augment the data they collect with part of the dataset from [**9**], analyzing the data of people with a COVID-19 test, assuming the data from 60 to 22 days before as a baseline period and from 21 days before to the diagnostic date as the detection period. Like other studies, they notice an increase of the daily RHR during the detection period and a decrease of the steps taken. They train LR, K-NN, SVM, RF and XGBoost with features extracted from both RHR and steps taken data, choosing the recall metric as the preferable to measure the models' performances as their consider an unidentified COVID-19 case (false negative) to be more serious since it can spread the disease. The LR model is the best performing model and training with either just the RHR or the steps data decreases its performance.

M. Gadaleta et al. [**36**] collect self-reported COVID-19 symptoms, COVID-19 test results and data from any wearable device that can be connected to either Google Fit or Apple Health kit platform and analyze the accuracy of gradient boosting DT models. Only data from participants with a COVID-19 test is included, with symptom reports being considered when occurring between 15 days before and the day of the test. To counter hardware and software differences, the data collected from the devices is treated to create dynamic daily baseline and a baseline variability. They analyze the data considering a window of 5 days before and after the test and only 5 days before the test, with either samples with symptoms or without. The models trained with symptomatic information achieve a better area under curve (AUC), as did the models that considered data after the test. They note the symptoms are the most important feature when present, followed by activity, sleep and HR.

R. P. Hirten et al. [**37**] collect biometric data using Apple smartwatches, demographic data, medical history, occupation, prior to study COVID-19 diagnosis and daily surveys about COVID-19 related symptoms, symptom severity and tests results from 409 health-care workers from 7 different hospitals and considers samples as positive when they occur 7 days before or after a positive test result. This data is used to train gradient-boosting machines (GBM), elastic-net, partial least squares, SVM and RF models, with GBM being the best one. They notice the most important predictors are related to heart rate variability (HRV), age and body max index (BMI) with RHR having median importance. They found the importance of the individual's sex to be 0 in most models.

B. Conroy et al. [**38**] detect COVID-19 with data collect from military personal using Garmin watch and Oura ring devices, along with demographic data and daily surveys about self-reported symptoms, taken medication and illness diagnosis. From the gathered data, they select the subjects who used both devices, had sleep data from more than 10 nights in the 21 days before a COVID-19 test and whether they had symptoms during the 14 days before the test (requiring symptoms if COVID-19 positive and the absence of if negative). Their algorithm achieved an AUC of 0.82.

A. Aguado-Garcia et al. [**39**] create a statistical biomarker based on Shannon Entropy and Beta Distribution to detect possibly COVID-19 infection. The daily heart rate, activity in steps and duration of sleep stages data of an individual is analyzed separately with this biomarker and values higher than the individual's threshold are kept, with the other being equaled to 0, to calculate the mean of the three values. Consecutive days where this mean is higher than the threshold are indicative of a possible infection. Over 6 months, they collected of 115 health care workers in charge of areas related to COVID-19 patients at the General Hospital of Mexico Dr. Eduardo Liceaga, in Mexico City, gathering data of 18 individuals that tested positive for COVID-19 with at least 2 months of data, the minimum considered for their study. Based on their observations, they suggest the best threshold to be 0.75 standard deviations during 7 consecutive days, having obtained a match of 94.4% between days marked by the algorithm and dates where the individuals had positive RT-PCR tests.

2.2.2. Federated Learning in the context of COVID-19

The literature review process suggests most efforts at applying federated learning to detect COVID-19 have focused on computer vision since 32 of the found records focused on images and only 9 on other types of data. The records found containing some data that might be collectable from wearable devices aimed at predicting the mortality risk of COVID-19 patients [**20**], the likelihood of an infected patient developing acute kidney injuries within 3 and 7 days of admission to the hospital [**18**] and predict a patient oxygen needs within 24 and 72 hours [**21**].

A. Vaid [**20**] and F. F. Gulami et al. [**18**] use electronic health records from five Mount Sinai Health System hospitals in New York, consisting of demographic information, past medical history, vitals, and lab tests to train models in three different ways: via FL, with all data pooled into a single location and with each hospital training a model with their local data. The authors in [**20**] use it to train Multilayer Perception (MLP) and LR with L1-regularization or least absolute shrinkage and selection operator (LASSO) with the federated models outperforming all but one local model from a hospital with a higher mortality rate. When compared to the pooled models, the federated LASSO are outperformed but two of the federated MLP are better than pooled one. In the case of the other work [**18**], the federated models have a similar performance to the pooled model and are better than the locals.

I. Dayan et al. train a model with 20 features consisting of chest x-ray (image) and electronic medical records, including SpO2 which can be collected by some smart wearable devices [**40**]. The data wasn't harmonized between the different participating locations (twenty spread over four continents) and the FL model outperforms all local models.

In terms of the aggregation algorithms, K. M. Elshabrawy et al. [**16**] use FL to train a model using chest x-rays from four different hospitals to see the effect of using different aggregation algorithms when the data is non-IID. Of the five algorithms (FederatedAveraging (FedAvg), FedProx, Federated Normalized Averaging (FedNova), Stochastic Controlled Averaging (SCAFFOLD) and FedBN), FedBN performed the best.

This literature review failed to find any FL applications that use wearable device data to detect COVID-19. Some studies show this type of data can be used for this end, but they gather the data from all participants and examine it together to create either pure statistical analyses or ML models, without the use of an aggregating algorithm.

2.3. Federated Learning Aggregation Functions

The works found during the literature review process apply five different aggregation functions, that is, functions that merge the weights of the models trained by each client into a single global model: FedAvg, FedProx, FedNova, SCAFFOLD and FedBN.

Let w be the parameters of the models, S the set of clients participating in the training round t, n the number of data samples present in a client's set and m the total number of samples across all participating clients. Equation 2.4 represents the FedAvg aggregation function, which consists on calculating the average of the weights of the models trained by the different clients during each round [**41**].

$$w_{t+1} = \sum_{k=S_t} \frac{n_k}{m_t} w_{t+1}^k \tag{2.4}$$

FedProx differs from FedAvg by adding a proximal term to the clients' algorithm to allow different amount of work per client and round due to the heterogeneity of devices and networks that can be found in a FL situation [**42**], while FedNova takes into consideration the number of locals updates done by each client during the averaging process instead of just their number of samples to help prevent convergence towards a subpar solution [**43**]. To increase the speed of convergence, SCAFFOLD adds a control variate based on the difference between the direction of the updated server model and of each client model to the FL process. This variate is added to the client's model after it finishes its local update but before it gets averaged into the global model [**44**]. FedBN focuses on feature shift, that is, clients having different feature distribution but not necessarily different label distribution, adding batch normalization to each layer of each local model before the aggregation phase [**45**].

FLWR [**46**], the FL framework this work applies, comes with implementations of the aggregation functions FedAvg and FedProx, plus the functions Federated Averaging with

Server Momentum (FedAvgM), FedAdagrad, FedAdam and FedYogi, and the ability to implement others through their abstract class Strategy [**47**].

How FedAvg updates the weights of the global model can be rewritten as seen in Equation 2.5, with Δw_{t+1}^k being the update of the k client's parameters during the tth round. FedAvgM replaces Δw with the variable v, which is updated as shown in Equation 2.6 [**48**].

$$\begin{aligned} w_{t+1} &= w_t - \Delta w_{t+1} \\ \Delta w_{t+1} &= \sum_{k=S_t} \frac{n_k}{m_t} \Delta w_{t+1}^k \end{aligned} \tag{2.5}$$

$$v = \beta v + \Delta w \tag{2.6}$$

S. J. Reddi et al. [**49**] add adaptive optimizers by changing the aggregation equation to Equation 2.7 where η is the client learning rate, τ is the degree of adaptability and β_1 is a decay parameter that varies between 0 and 1.

$$\begin{aligned} w_{t+1} &= w_t + \eta \frac{m_t}{\sqrt{v_t}\tau} \\ m_t &= \beta_1 m_{t-1} + (1-\beta_1)\Delta_t \\ \Delta_t &= \frac{1}{|S|} \sum_{k=S_t} w_{t+1}^k - w_t \end{aligned} \tag{2.7}$$

How v_t is calculated depends on the adaptive optimizer algorithm that is being applied. The authors present three, with β_2 being a second decay parameter:

- FedAdagrad

$$v_t = v_{t-1} + \Delta_t^2 \tag{2.8}$$

- FedAdam

$$v_t = \beta_2 v_{t-1} + (1-\beta_2)\Delta_t^2 \tag{2.9}$$

- FedYogi

$$v_t = v_{t-1} - (1-\beta_2)\Delta_t^2 sign(v_{t-1} - \Delta_t^2) \tag{2.10}$$

This work focuses on the aggregation functions FedAvg, FedAvgM, FedAdam, FedAdagrad and FedYogi.

CHAPTER 3

Data Understanding and Preparation

This chapter presents the data explored during the course of this work, its treatment and assumptions.

In order to achieve its goals, this work has access to raw data collected by the Hospital de Santa Maria (HSM) in Lisbon, Portugal, during normal care procedures and ranging between the years of 2019 and 2021. Unfortunately, since this data is collected from patients admitted to the hospital's ICU, it is found to be unsuitable to train an anomaly detection model because it lacks samples that one can confidently consider healthy, or normal, to use as a baseline to train the model. To counter this, data from two publicly available datasets containing time-series of biometric data are used to train the ML model explored. Meaning, in total, data from 3 different sources is explored.

The chapter is divided into two sections according to the origin of the data: the publicly available data collected by the authors of two scientific studies mentioned in the literature review [**9, 6**] and raw data from patient admissions of HSM made available to the AIM-HEALTH project of which this work is part of.

3.1. Publicly Available Data

T. Mishra et al. [**9**] collected heart rate measurements and activity from several users from February to June of 2020. The measurements were obtained via personal wearable devices of volunteers, in particular FitBit[1] devices. The data was made publicly available after anonymization. This public dataset contains data pertaining to 32 individuals that have been infected with COVID-19, 15 that have been infected with other illness and 73 with no reported symptom or illness [**9**]. The individual data covers both infected and uninfected periods.

A. Alavi et al. [**6**] published a dataset with HR and activity data of 2123 volunteers, 84 of which have been infected COVID-19. The data was collected between November of 2020 and July of 2021 by the individuals' personal wearable devices (Fitbit, Apple Watch[2] or Garmin watches[3]) to be used in a real time COVID-19 alert system. [**6**].

Data from the T. Mishra et al. dataset pertaining to individuals with known COVID-19 infection is referred to as "P1" from hereafter, and data from individuals with no known illness or other illness as "P1 Healthy" and "P1 Other Illness". Data from individuals with known COVID-19 infection and from the A. Alavi et al. dataset shall be referred to

[1]Brand of smartwatches focused on activity trackers.
[2]Smartwatches developed by the American technology company Apple.
[3]Smartwatches developed by the American Swiss-domiciled technology company Garmin.

as "P2". The data from individuals with no known illness from the latter dataset are not explored.

In both of the previous cases, the data of each participant is organized into two files - one with HR data and another with activity data. The HR files contain measurements in bpm with the timestamp of the recording, measured multiple times per minute, while the activity data differs depending on the capture device: FitBit records contain the number of steps taken during a minute (represented by a timestamp), while other devices contain a start and an end timestamp of the activity and the number of steps taken during the interval. Both datasets contain information about when individuals first felt symptoms of illness in different time periods (symptom onset) and dates of positive COVID-19 tests.

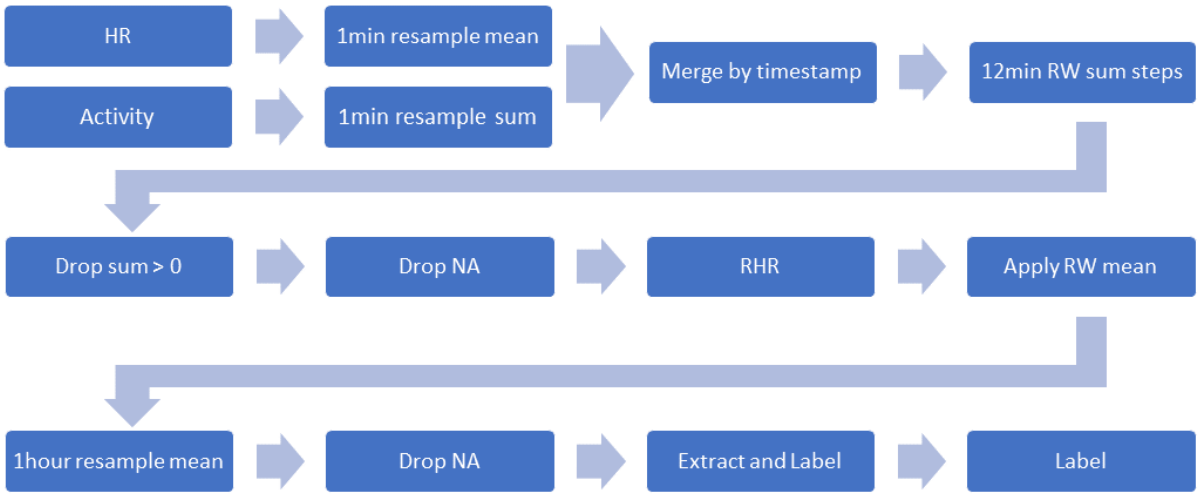


FIGURE 2. Data pre-processing scheme for public datasets.

For the analysis presented in this dissertation, all the publicly available data from both datasets has been treated following G. Bogu and M. Snyder publicly available code [**1**]. To simplify, this work ignores the end timestamps that some devices create when collecting the activity data and uses the start timestamp as if it is the moment where the steps are taken. For each individual:

(1) the HR, measured multiple times per minute by the devices, is averaged at the 1 minute interval the mean bpm of that minute
(2) the activity data is summed at the 1 minute interval to introduce missing minutes with zero steps
(3) the two time series are used to extract the HR measured when there hasn't been any activity, that is steps taken, during the twelve previous minutes. These measurements are considered the individual's RHR.
(4) the RHR is smoothed using a Rolling Window (RW) and the mean RHR of each hour is extracted. How the data is smoothed is explored in section 4.3

This process is schematized in Figure 2, including when the data is cleaned of missing values.

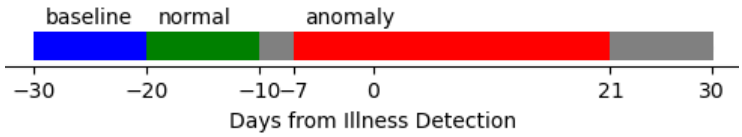


FIGURE 3. Division of data per label according to the distance to day Zero of known illness. Baseline samples are used to train models, normal and anomaly samples to test.

For individuals with a known illness, whether COVID-19 or other, the earliest positive test day is used as day 0 and data from 30 days before until 30 days after is extracted and labeled using appropriate labels as shown in Figure 3. Thus, we considered as baseline data the samples from 30 days before until 20 days before the first positive COVID-19 test, which means that these signals report what we consider to be a period completely free of COVID-19, that is, these are healthy signals. The remaining samples are labeled normal if occurring between 20 to 10 days before the positive test, while the infection period is taken to occur from 7 days before until 21 days after the positive test (marked as day zero). Notice that days colored in grey in Figure 3 represent days that were not explored in this work: the first section refers to days where it is difficult to determine if an individual is already infected with the virus due to the incubation period, while the second block refers to when the individual is likely to be recovering from the illness and still experiencing some symptoms, even if mild.

TABLE 3.1. Number of individuals in each sub-dataset after applying the data-preprocessing (RW=200) and discarding those without enough data for training.

Subset	P1	P2
COVID-19	29	77
Healthy	73	–
Other Illness	9	–

Table 3.1 presents the number of individuals in each dataset after applying data preprocessing and selecting the individuals with enough data to train and test the AI model explored in this work.

Figure 4 shows the mean RHR of each subset per label. This mean is the mean HR in bpm measured when the individuals haven't been moving for at least twelve minutes, which can mean, for example, that they are sleeping, sitting down working at a desk or watching a movie. Notice that, during the day, the mean RHR changes similarly across the different subsets and types of samples. But, more importantly, we can observe that the anomalous samples in both P1 and P1 Other Illness show higher RHR than the corresponding baseline samples. Although there is a short three-hour period where this difference is not observable in P2, for the remaining hours, the pattern is maintained, although at a lower relative difference. Therefore, this study uses the baseline samples to

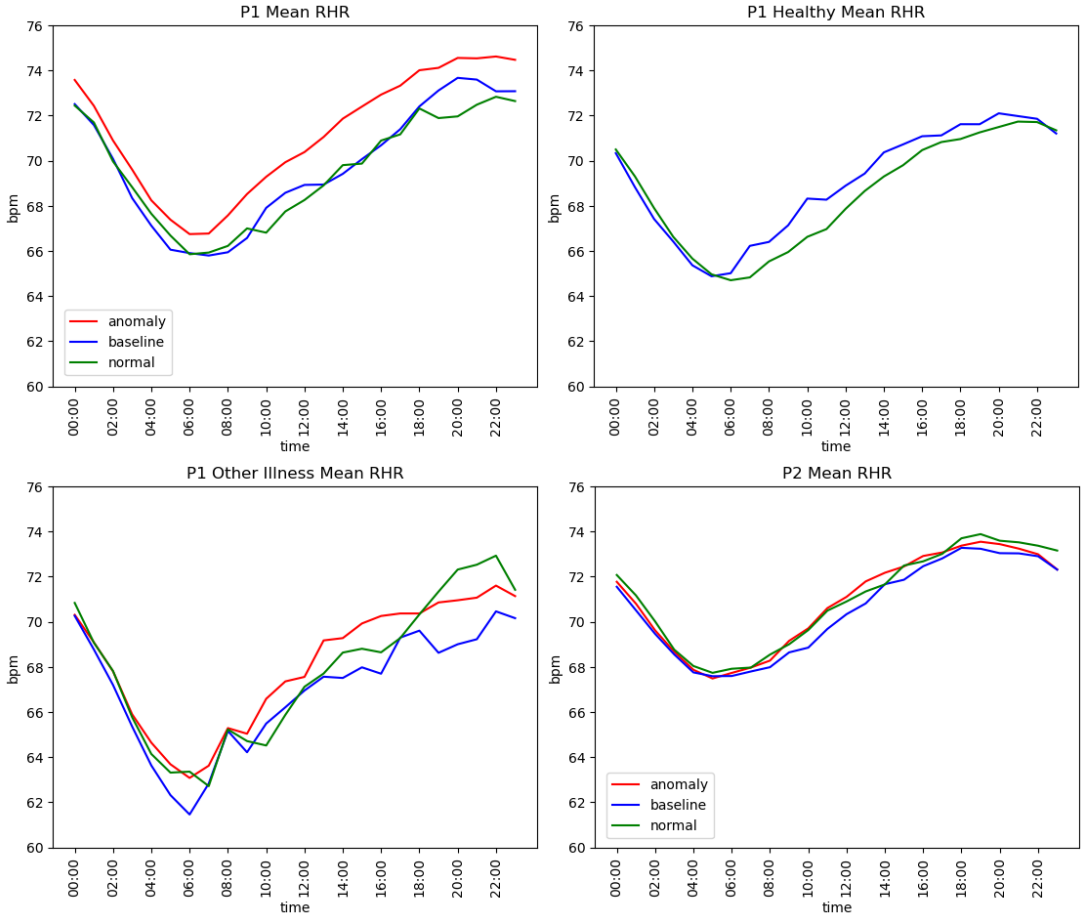


FIGURE 4. Mean hourly RHR after applying 200RW

train and validate the models used and the normal and anomalous samples to test and evaluate the models. The distribution of training or baseline samples and normal and anomaly samples for the test of these sets can be observed in Figure 5. It is noticeable that the anomaly observations show a significant difference in behavior from the remaining observations.

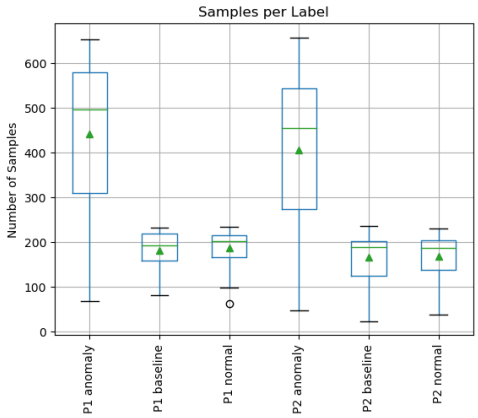


FIGURE 5. Distribution of the type of samples present in P1 and P2.

3.2. Raw Hospital Data

HSM provided data from their databases. These data are generated by their sensing devices but also manually inserted by staff related to patients admitted into the hospital's intensive care unit between the years of 2019 and 2021. This work has used both structured data - in the form of comma-separated value (csv) files - and text reports. Each csv file pertains to a different database table, with the type of data varying greatly from one file to another. The text reports are notes taken by medical staff thus their content, structure and style of writing also vary greatly between different reports. These reports are identified by patient id.

The different csv files were examined to find how they connect and how to identify what information relates to patients infected with COVID-19. As it could be expected, a sample of raw real-data raises several issues for data extraction and treatment.

The data provided is scattered amongst several files without any indication of how they complement each other. As such, it is necessary to examine the features found on each file to discover any connections. Furthermore, some of the features are identified with codes whose location within the many files is not easily discerned and some information is simply not possible to infer from the codes or names on its own, requiring communicating with an expert advisor involved in the involving project.

In terms of the quality of the data, several data is missing and a great number of cells are filled incorrectly with types of data that do not fit with the values for that feature, such as text or numbers in columns that are clearly meant to be marked as either true or false. There are also issues with multiple symbols or words being used with the same meaning, which makes it more difficult to extract data when searching for them, such as multiple ways to write "true". Due to these issues, the high quantity of raw data and the necessity of keeping it stored in a safe server, the data extraction and clean up process took around three to four months, despite the help and information provided by the experts and other researchers involved in the AIM-HEALTH project.

In total, just by exploring the files, six different ways to discover such data have been employed: by looking for the terms "sars" and "covid" in a table listing analyses the hospital provides and linking it to lab analyses results and there on to the patients they are assigned to, filtering for positive results (total crossing of 5 files); looking for the same terms in a table listing the several diagnosis the hospital contains in its information system and linking it to the given patients (3 files crossing); repeating the search in four different columns reserved for free text found in three of the tables explored in the previous methods (only one of these actually returned any results but, given the structure of the raw data, it was feasible to find results in all of them). A seventh method was provided by other researchers involved with the AIM-HEALTH project, starting from a list of equipment reserved for COVID-19 patients (5 tables crossing). Finally, some patients with the illness have been found by exploring, with the help of a text mining expert, the textual reports regarding medical exams the patients went through.

After discovering the patients, it was necessary to find data that can be fed to the ML model by linking three different tables, for a total of 12 files. In the end, 935 patients with mentions of COVID-19 infection have been discovered in the data provided by the hospital, of which 310 contain biometric data records pertaining to the patient's HR.

Each identified patient with a known COVID-19 infection can be associated with multiple candidate dates for the earliest confirmation of the illness because some are admitted with an already known COVID-19 positive status and because one patient can take multiple tests throughout their admission. It is also possible for data connected to the same patient's admission to belong to different times, for example, a patient admitted on the 11th of October only contains data collected after January of the following year. The candidate dates were automatically extracted from the available medical reports and observations, via manual reading of the texts, the lab data tests, and the start of admission dates. The earliest COVID-19 infection date is selected as the first day of known infection (day zero). HR data ranging from 30 days before until 30 days after day zero of patients with known COVID-19 infection is extracted to use in this work. This extraction reduces the number of patients to 286. These patients do not contain data spanning the entire 60 days: only 34 patients contain data up to seven days before their day zero, 118 data collected 21 days after the day zero and 234 data pertaining to the time in between.

After the identification of the COVID-19 positive patients, HR data of the remaining patients found in the provided data, with a known type of diagnosis not related to COVID-19 or pediatric care, are extracted to see if the trained models identify anomalies within it. Diagnosis with less than nine patients, the number of individuals found in the P1 Other Illness dataset, are also discarded as such a small number of patients is likely not statistically relevant. From these patients with other diagnoses, the first 60 days of the data are extracted. In fact, the date of admission is not always accurate and searching the database for a date of analyses or report that confirms the diagnosis would be time consuming task requiring external help and that falls outside of the scope of this work. Overall, we extracted data for 8467 patients. However, the extracted data does not always span 60 days: for 5644 of the patients the data were collected during a single day, for 2216 the data was collected during 10 days, 483 patients contribute with data collected during 11 to 59 days, and 124 patients with data spanning 60 days.

TABLE 3.2. Comparison of the publicly available datasets and the data pertaining to COVID-19 infected patients found in the HSM dataset

Dataset	P1			P2	HSM
	Covid	Other illness	Healthy		Covid
Device	Smartwatch				Hospital
Data	HR + Activity				RHR
#Individuals	29	9	73	77	212
Healthy Data	Yes				No

Table 3.2 highlights the differences between the datasets presented in the last section and the HSM data. Notably, because the available data belongs to patients admitted into the hospital's intensive care unit, none of the samples can be considered a "healthy" baseline. Thus, the data from this set cannot be used to (re)train models and can only be used for testing, serving as anomaly samples. Remaining types of diagnoses are used to explore if the fraction of samples detected as anomalous is similar between data from patients with COVID-19 and other health issues.

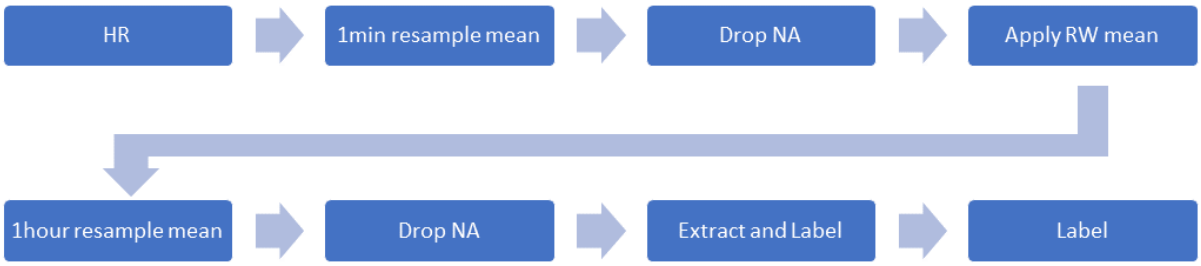


FIGURE 6. Hospital dataset data pre-processing

This process can be observed in Figure 6, which includes the steps where the data is cleaned of missing values.

This work assumes the HR of the hospital dataset is comparable to the RHR of the publicly available dataset, as the provided data contains no information regarding activity and it is collected from people admitted to the hospital's intensive care unit, meaning it is reasonable to assume they are resting in beds, chairs or stretchers and not walking (which is the criteria required to consider the HR from the public datasets as RHR).This is followed by a drop of null values that might have been created by the resampling, which does so whenever it fails to find data to create the mean for a given minute between minutes with known values, before a rolling window with the mean function is applied for smoothing the data. The size in minutes of this rolling window is one of the parameters explored in this work, which reaches the conclusion that 200 is a good compromise between performance and loss of number of patients that do not have that number of minute samples.

After pre-processing, the mean RHR of individuals with known COVID-19 infection from HSM shows higher values than what is found in P1 and P2. Moreover, as it can be observed in Figure 7, it lacks the daily pattern observed in the previous section. Nevertheless, it should be noted that HSM patients are constantly abed, while the same cannot be inferred from P1 and P2 subjects. The fact that these are patients in an ICU, thus in a serious anomalous state that requires constant observation, can also help to explain the higher RHR. Finally, this values are monitorized using specific cardiac clinical devices, which are rather more accurate in the heart measurements than a wearable.

In terms of HSM data, several patients have been identified with illness other than COVID-19. Figure 8a shows the number of patients per diagnosis type with enough data to create the input of the chosen model (eight hour samples) after applying the

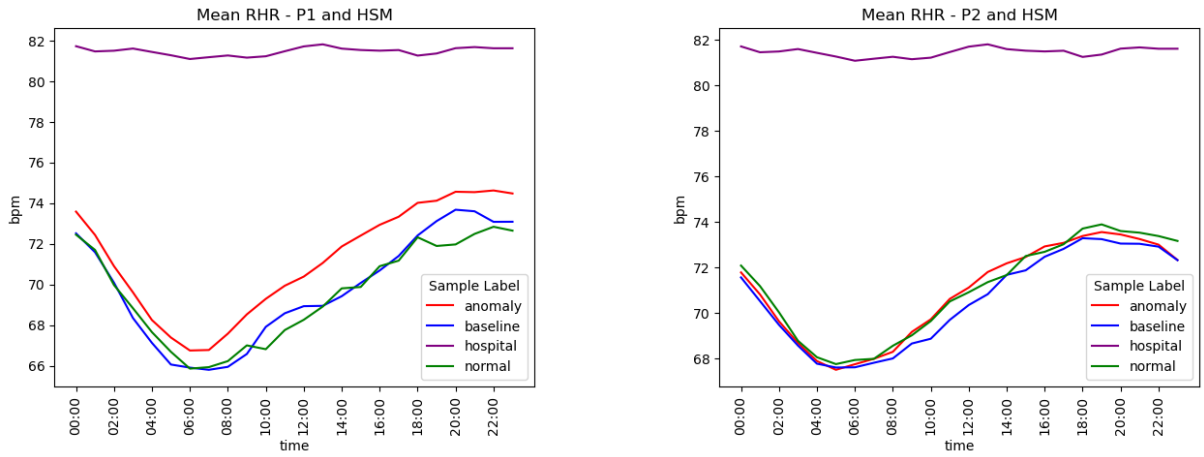


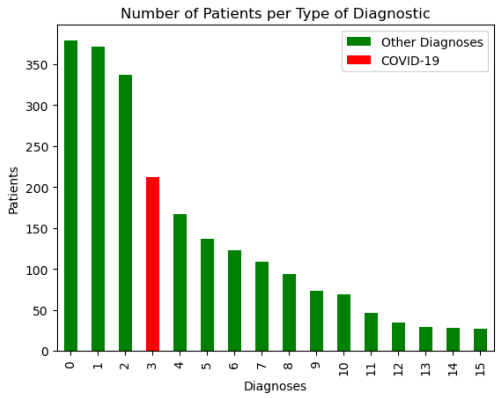
FIGURE 7. Daily mean RHR of individuals with COVID-19 found in P1, P2 and HSM datasets

pre-processing with a RW of 200 minute samples. The correspondence between the label number and type of diagnoses can be found in table 3.3.

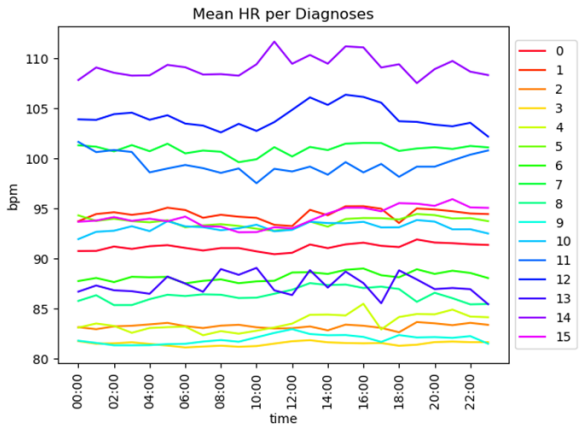
TABLE 3.3. Number assigned to the types of diagnoses present in the Hospital dataset

Number	Type of Diagnoses
0	Intensive Care Medicine Services Main Diagnosis
1	International Classification of Diseases 10th revision
2	Diseases of the Circulatory System
3	COVID-19
4	Neoplasm
5	Diseases of the Genitourinary System
6	Common Diagnoses
7	Diseases of the Respiratory System
8	Diseases of the Nervous System and Sense Organs
9	Trauma and Poisoning
10	Supplementary Classification of Factors that Influence the Health
11	Diseases of the Digestive System
12	Hematological and Hematopoietic Organic Diseases
13	Diseases of the Musculoskeletal system and connective tissue
14	Infectious and Parasitic Diseases
15	Psychiatry

Figure 8b shows the mean HR for patients with the previously identified diagnosis. Same as with COVID-19, all other types of diagnoses appear to show higher HR values than those present in the public datasets and lack the daily pattern previously seen. This visualisation of the HR divided by diagnosis enables the understanding of possible pattern differences between patients with COVID-19 and with other illness based on the mean signal. Patients with COVID-19 are amongst the diagnosis types showing the lowest mean of HR and, although not explored in this work, it's worth noticing the stratification that can be observed might be interesting to explore with a classification model.



(A) Number of patients per diagnosis type



(B) Mean HR per type of diagnosis

FIGURE 8. Statistics and HR signal relating to the hospital data.

CHAPTER 4

Experiments and Results

This chapter focus are the different experiments made to obtain an anomaly detection pipeline trained via FL, that can be applied to both data collected via commercial wearable devices and data collected using hospital devices and, if successful, be used by new users without the need to fine-tune the pipeline with personal healthy data. This anomaly detection pipeline contains a ML model and the generation of a threshold based on the training data.

The chapter starts with a summary of all the experiments conducted during this work, followed with the application of the code made available by G. Bogu and M. Snyder [**1**], which served as the base for the investigation with the FL approach, using both P1 and P2 datasets described in the previous chapter. Next, it proceeds with the changes to better suit the data from HSM and experiment with other methods to calculate the anomaly detection threshold, before comparing different methods of training and some of the FL aggregation functions that are offered by the FLWR framework. An extra step is introduced to the pipeline to explore how refining the number of anomalies needed to consider a detection a true anomaly affects the performance, then the models with the highest mean recall are examined at the individual level and, finally, some of the models are applied to the hospital data and the other subdatasets of P1.

Due to the variety of experiments conducted, this chapter ends with a small recap of the most important results.

4.1. Experiments

The literature review reveals a work that uses a public dataset to train a personalized anomaly detection model for the data of each individual present in the set with a LSTM autoencoder and personalized threshold at its core, to detect anomalies in the individual's RHR that might be indicative of an health issue, in particular of early COVID-19 infection. Since the authors made their code publicly available, it was decided to use this work as the start of this project and see how it performs when moved from the model being individually trained to trained by the data of several individuals via FL. This anomaly detection pipeline, including the NN explored, is detailed in Section 4.2.

Several experiments are conducted to alter the pipeline so it can be applied to the hospital data and to decide how to transfer the focus on the individual into the FL paradigm. Each experiment explores specific changes to the original pipeline and these are incremental. To select which change is brought to following experiments, higher mean

recall scores are favored because this work deems it more critical to minimize false negatives, that is, to decrease the number of anomalous samples that are classified as normal by the anomaly detection pipeline. Changes to the mean precision are also observed, and to the mean specificity because the number of supposed healthy samples, that is that are expected to not be marked as anomalies, used to test the models refer to a period of ten days while the anomaly samples refer to twenty eight days. Please note all models used during in this work have the same structure, differing only in their weights due to changes on how they are trained.

The first experiment consists on recreating the original work, that is, training a model for each of the different individuals present in the public datasets, and then comparing the mean performance of the resulting models to the mean performance of models trained via FL with the same data. The original work classifies whether each hour sample is an anomaly, which might be overwhelming or confusing when transported to the end-user of the pipeline, while the work the other public dataset is part of had interaction with the user and alerted them after two anomalous days. Because of this, detection of anomalous days is also explored during this first experiment, although the following ones do not apply it to have a better sense of the anomaly detection pipeline raw results.

The original work's pre-processing data pipeline contains a standardization process based on part of an individual's healthy samples and applies a rolling window to smooth the time series. The former cannot be applied to the hospital data as it contains no samples than can confidently be considered healthy and applying the latter shows that several of the hospital's individuals do not contain enough data for it. The next set of experiments consist on changing the pre-processing data pipeline to deal with these constraints. This allows not only to use the data made available by the hospital, but also permits users of the possible future end product with less data to use the detection pipeline without needing known healthy data.

In terms of the threshold that determines whether a sample is anomalous or not based, which is calculated based on the differences between the sequence of measurements that is fed into the autoencoder model and the reconstruction sequence it returns (differences between the input and output of the model), two things are explored: how the individual's threshold is calculated, and each individual using their own threshold versus the creation of an averaged threshold. The first exploration is done to try to increase the mean recall while the second is, once again, due to the lack healthy data in the hospital dataset and contains the same benefit mentioned above

Once the changes to the pre-processing pipeline and the method to calculate the threshold are established by training individual models, models are trained by treating the data of all the individuals as a single set of data and via FL.

FL introduces three new variables to the training: the number of participants, the number of training rounds and the aggregation function. The number of participants is how many different users (or clients), each with their own data, participate in the

training of the global model by training a local model and sending its weights to another user (the server) to merge them and create the global model, while the number of rounds refers to the number of times this process occurs in a row. The aggregation function is how the server merges the weights of the different local models. Five of the aggregation functions offered by the FL framework used during this work are explored: FedAvg, FedAvgM, FedAdagrad, FedAdam and FedYogi. The FL is run for thirty rounds with each aggregation function, always starting with a new untrained model and all available individuals participate with their data. Due to memory issues, the number of epochs each individual model is trained for is changed from up to 1200 to 120. The model created during each training round is saved for later testing to create evolution lines of the mean metrics achieved by each aggregation function.

After all the models are trained and tested with data pertaining to the individuals who trained them, how to classify periods of time longer than one hour is explored by, once again, seeing the affect on the metrics score of considering an entire day as anomalous if it contains at least one anomalous hour sample and requiring the previous day to also verify that, and by changing the percentage of anomalous hour samples found in a day required.

Once all these experiments are conducted, the models with the highest mean recall scores are selected to see the results at the individual level in order to see if any patterns emerge such as individuals that achieve similar results with all the best models or models that behave much better on some individuals but worse with others, for example.

Finally, the models are tested with data from the hospital and other data from the public datasets that is not used in the training to see if similar mean scores are achieved. This data includes individuals with known COVID-19 infection, other illnesses and no known illness, that is, supposed healthy individuals

Due to difficulties in running the simulation feature offered by FLWR that simulates the FL with a server and several clients with the data augmentation featured in the chosen work, no data augmentation is applied during the experiments run.

In all experiments that involve training models via FL, the process is run for 30 training rounds. This number was chosen arbitrarily during the first attempts at using the FLWR framework.

4.2. Applying the work of G. Bogu and M. Snyder [1] to Federated Learning

The first experiment of this work consists of adapting the code from G. Bogu and M. Snyder [**1**] to train a personalized model for each individual with known COVID-19 infection found in P1 and P2 and a model trained via FL with the aggregation function FedAvg by the individuals of each dataset.

The authors worked with the data collected by the authors of [**9**] to create personalized anomaly detection pipelines for each individual of the public dataset (P1). This pipeline contains a LSTM-based autoencoder at its core consisting of six layers: two consecutive

LSTM layers with input and output shapes of (8x1)→(8x128) and (8x128)→(1x64), respectively, to extract 64 features from the original input which are replicated by a Repeat Vector layer before going through mirror versions of the first two layers with the sixth, and final layer, being a TimeDistributed (Dense(1)) layer that reconstructs the shape of the input. This autoencoder receives eight measurements of RHR pertaining to eight hours and returns a reconstruction of them.

A single value, referred to as loss, is calculated by comparing the input and output of the autoencoder, which is used to verify if the eighth, or final, hour is anomalous by comparing it to a threshold. In the pipeline developed by G. Bogu and M. Snyder [**1**], the loss corresponds to the MSE of the input and outup sequences and the threshold there proposed is the maximum MSE obtained with the baseline samples, when they are sent through the trained autoencoder model. If the current value is greater than the threshold then the eighth hour is considered anomalous. The main stages of the anomaly detection pipeline can be seen in Figure 9.

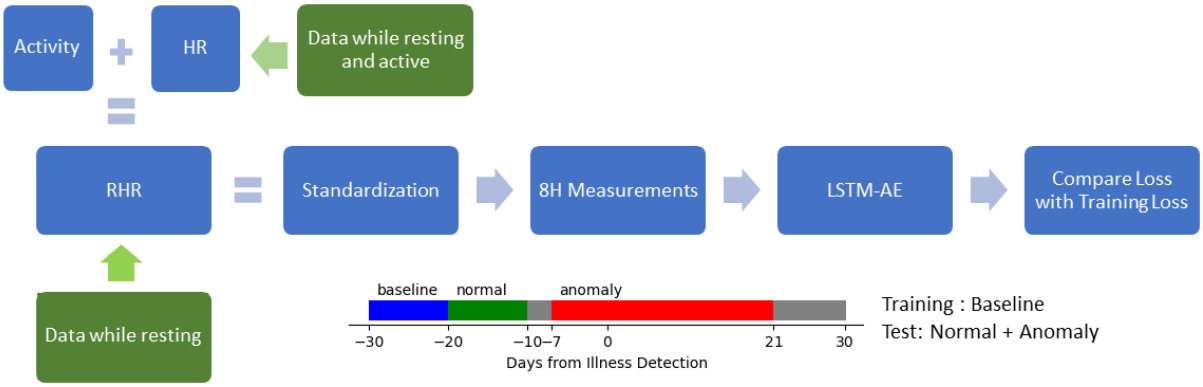


FIGURE 9. Main steps of the anomaly detection pipeline.

The eight hours that make the input do not correspond to consecutive hours, but rather consecutive hours with RHR, that is, if the individual's RHR skips one or more hours these are not filled with a default value, instead the next measurements are used until the input is complete. This experiment applies a rolling window of 400 minute samples, standardization based on baseline samples of each individual and max MAE threshold calculated at the individual level.

A. Alavi et al. [**6**] alert the users of their online anomaly detection when there are two consecutive days of elevated RHR. Figure 10 shows the evolution of the mean precision, recall and specificity scores obtained by the different models over FL rounds (each round consists on the training of individual models and their aggregation into a single global model). The detection is made at an hour sample (Sample) and at the day level, requiring only one anomalous hour sample for the day to be considered an anomaly (1Day) or using also the previous day to issue an anomaly alert (2Days). Note that, over rounds, precision appears to evolve more erratically than recall. The word "solo" refers to when FL is not applied and a personalized model is trained by and for each individual.

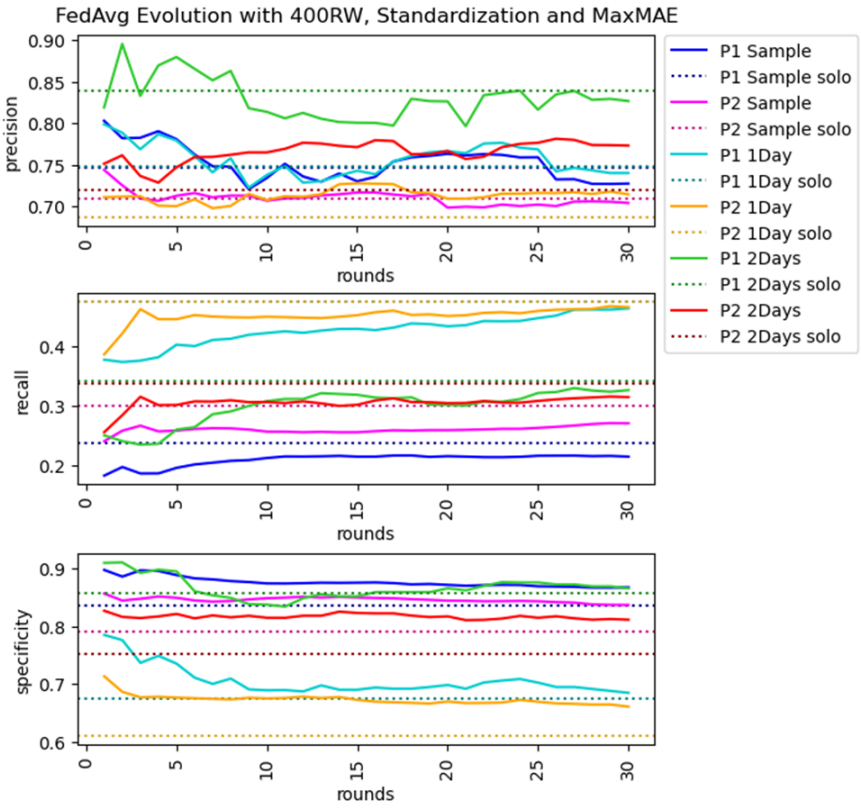


FIGURE 10. Evolution of the mean precision and recall scores (over P1 and P2 datasets) with hourly, daily, or two days in a row detection.

In general, detecting anomalies at the day level (1Day and 2Days) instead of the hour level (Sample) increases the mean recall, with 1Day showing a greater improvement than 2Days. But in none of the three situations the models trained via FL produce a better score than the solo models. In terms of the specificity score, the opposite behavior is observed - the FL models surpass the solo ones and 1Day shows the lowest scores while Sample achieves the highest, although the federated P1 Sample and P1 2Days lines cross several times during the 30 rounds of training. This results in the highest mean precision being achieved by the models trained with the P1 dataset and the 2Days configuration, independently of the training method. Regarding Samples and 1Day, neither P1 nor P2 show pronounced differences.

From now on, the results are presented considering only the hourly samples unless stated otherwise, to better see the effect of the experiments on the results of the original anomaly detection pipeline.

4.3. Changes to the Data Pre-Processing

The data offered by the HSM is collected via clinical monitoring systems, i.e., data from patients admitted into the hospital's intensive care unit. Thus, it contains no samples that can confidently be considered healthy and used as a baseline for standardization. Moreover, a more in-dept exploration of the data shows there are less samples per individual than in the publicly available datasets. This means that applying the 400 RW

leads to losing 20% of the individual observations. Thus, this section focus on the affects of removing the standardization and also decreasing the RW previously described for the anomaly detection process in order to achieve a pipeline that can be applied to the hospital data without compromising too much the performance of the system.

Table 4.1 contains a summary of the mean precision, recall and specificity scores obtained by each experiment with each dataset. The first two lines refer to models trained with data that went through standardization and augmentation processes, the third does not apply augmentation and other do not apply either technique. The loss and threshold are calculated using mean absolute error (MAE), unless stated otherwise.

TABLE 4.1. Mean precision, specificity and recall scores obtained when training a model per participant (solo training) with different data preprocessing configurations and changing the LR parameters (the changes are cumulative). The higher and lower scores obtained in each dataset per metric are always marked using (+) and (-), respectively.

Experiment	P1			P2		
	Prec	Recall	Spec	Prec	Recall	Spec
Public Model (MSE)	0.838	0.120	0.929			
Public Model (MSA)	0.842	0.118(-)	0.931			
Remove Augmentation step	0.797	0.222	0.867	0.730	0.296(+)	0.782(-)
Remove Standardization step	0.741(-)	0.200	0.858(-)	0.778(+)	0.204	0.854
Increase LR	0.846	0.224(+)	0.893	0.771	0.282	0.826
Decrease RW	0.789	0.180	0.904	0.734	0.216	0.882
Remove RW step	0.888(+)	0.123	0.950(+)	0.711(-)	0.066(-)	0.953(+)

In what concerns the use of P1, removing both the data augmentation and standardisation steps decreases the mean precision scores. This effect can be minimized by increasing the LR, achieving the second-best precision value and the best recall with this dataset. Decreasing the RW to 200 samples decreases both scores while removing this step completely increases the precision but lowers the recall. The variation of the RW is cumulative with the removal of the augmentation and standardization steps. The public models, built by G. Bogu and M. Snyder, achieved lower scores than those reported by the authors [**1**], possibly because they split the individuals' data based on the date of symptom onset and this work uses the date of the positive COVID-19 test.

The same experiments using the P2 dataset achieved generally lower scores when compared with P1, with the last experiment performing the worst of all.

Hereinafter, further experiments are performed choosing a RW of 200, no standardization and a LR of 0.001. The reason for these choices lies in the fact that these are the settings that applied to the hospital dataset balance the loss in performance with the loss of observations that can be used.

4.4. Threshold Variation

The next experiment focuses on varying the threshold that determines if the loss obtained by a sample is indicative of an anomaly or not, with each individual still creating their own model and threshold. This work explores four different methods to calculate the threshold: the maximum MAE (Max) used in the previous section, and three other thresholds, where MAE is represented by $\overline{X}$, defined by

$$\overline{X} + k\sigma, k = 1, 2, 3. \tag{4.1}$$

In general, the effect on the three metrics values is now inverted, with higher thresholds increasing the precision and specificity but lowering the recall. The changes have a higher impact on recall, with a difference of 0.240 between the highest and the lowest values, while the specificity varies by 0.200 and precision only varies 0.075, at most. The mean scores can be seen in Table 4.2 and the distribution of the threshold value per dataset can be found in Figure 11, showing the obtained loss does not follow a normal distribution since $\overline{X} + 3\sigma$ can be higher than the maximum loss.

TABLE 4.2. Mean scores obtained with solo training models varying how the threshold is calculated. The highest value per metric is in bold.

Threshold	P1			P2		
	Prec	Rec	Spec	Prec	Rec	Spec
$\overline{X} + \sigma$	0.748(-)	**0.396(+)**	0.713(-)	0.744	0.390(+)	0.729(-)
$\overline{X} + 2\sigma$	0.807	0.249	0.856	0.757	0.250	0.858
$\overline{X} + 3\sigma$	**0.823(+)**	0.156(-)	0.913(+)	0.760(+)	0.165(-)	**0.918(+)**
Max	0.789	0.180	0.904	0.734(-)	0.216	0.882

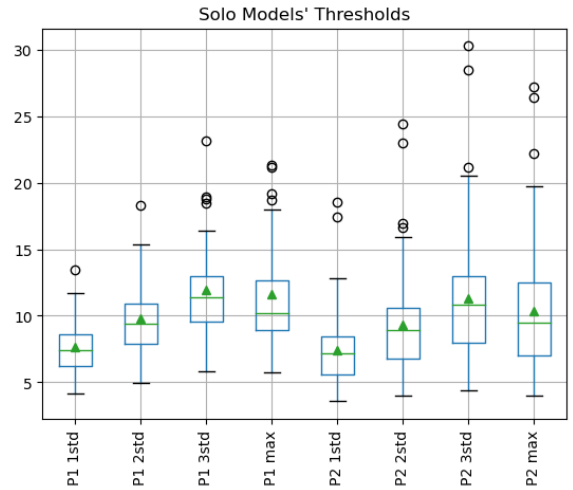


FIGURE 11. Value distribution of the different thresholds per dataset.

For the aim of this work (abnormal symptomatology alert for early disease detection), preference is given to decrease the number of anomalous samples that are not detected, that is, to increase the recall.

As the results present in Table 4.2 show that, despite decreasing the precision to 0.748 with P1 and 0.744 with P2 and the specificity to 0.713 and 0.729, using $\overline{X} + \sigma$ as threshold improves the recall the most, the remaining experiments are conducted setting the anomaly detecting threshold by this formula.

4.5. Training Methodology

The works found during the literature review about FL implementations showcase results pertaining to three different training methods: each client trains its own model with its own data (solo training), gathering the data of all clients into a single dataset used to train a general model (local training), and the FL approach: train locally and aggregate model parameters for a central global model.

TABLE 4.3. Mean scores of the solo, local and local averaged models.

Model	P1			P2		
	Prec	Rec	Spec	Prec	Rec	Spec
Solo	0.748	0.396	0.713	0.744	0.390	0.729
P1 Local	0.717	0.274	0.775	0.735(+)	0.321(+)	0.750
P1 Local Averaged	0.711(-)	0.258(-)	0.770	0.722	0.243(-)	0.777(+)
P2 Local	0.721(+)	0.269	0.782(+)	0.715	0.311	0.738(-)
P2 Local Averaged	0.720	0.308(+)	0.762(-)	0.699(-)	0.279	0.741

Due to the nature of the chosen anomaly detection pipeline, containing an autoencoder model and using a threshold to determine if an anomaly has occurred, and the constraints of the hospital dataset (no baseline data to enable the creation of a threshold), this section explores the scenario of having each individual creating its own threshold (local models) and using all the individuals to create a general/global threshold. In the latter case, the threshold consists of the mean of the individual thresholds (the averaged approach).

Table 4.3 contains a summary of the mean scores obtained by the solo, local and local averaged approaches, marking the highest and lowest scores obtained by the non-solo models. Figures 12 and 13 show the evolution of the scores of the models trained by the two datasets when tested with P1 and P2, respectively.

In general, the solo models show higher precision and recall but lower specificity, while the local models achieve higher precision. When an averaged threshold is used, the models show the lowest precision. Examining the mean recall and specificity of these approaches, the local models obtain higher precision with P1 due to classifying less normal samples as anomalies, and with P2 by detecting more true anomalies.

In terms of the FL approach, the precision of the models evolves in a more erratic way than the other two metrics, but within a smaller range. The mean specificity of this training method is higher than what is obtained with the other methods with both types of thresholds (individual and averaged) managing to obtain the maximum precision in a few rounds, although at the expanse of the recall. The one exception is the model trained with P2 with an averaged threshold that never surpass the specificity of all non FL

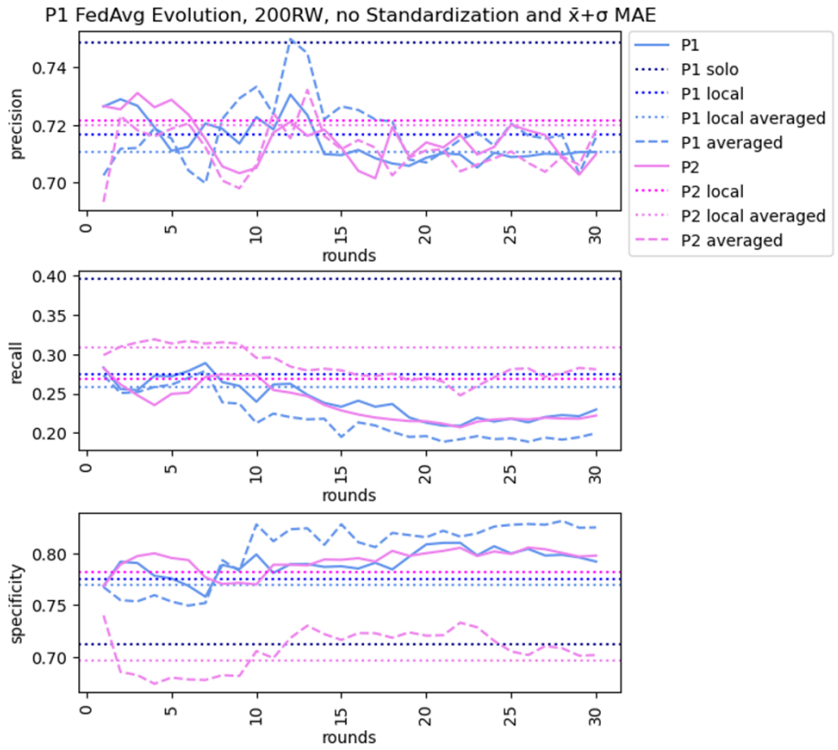


FIGURE 12. Mean scores of solo, local and FL with FedAvg models tested with P1

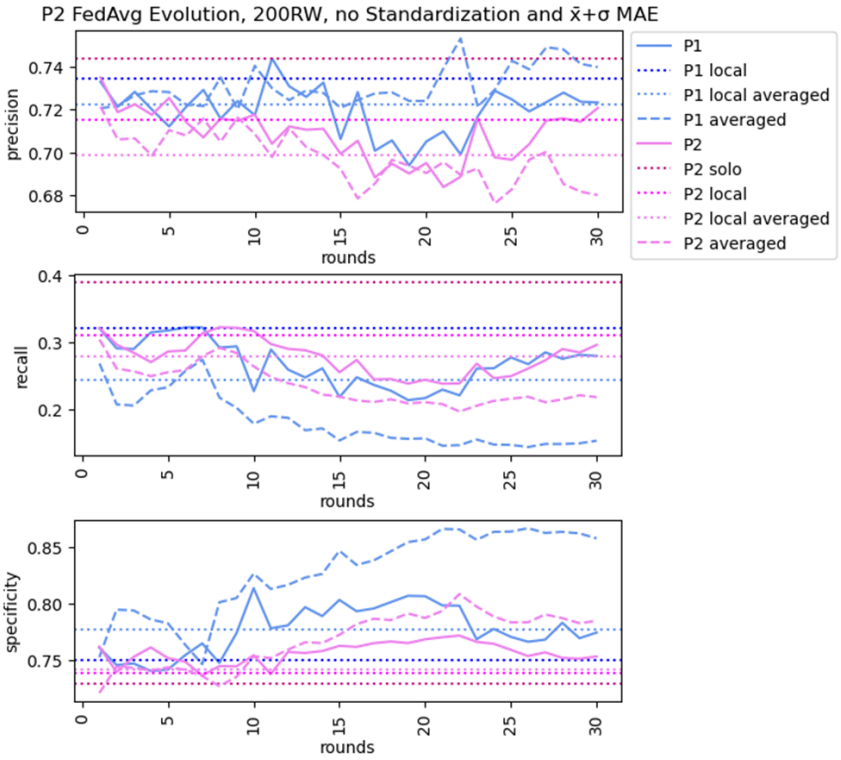


FIGURE 13. Mean scores of solo, local and FL with FedAvg models tested with P2

models when applied to P1. This seems to indicate that applying FL using this particular aggregation function (FedAvg) is good for decreasing the number of false positives.

4.6. FL Aggregation Functions

A model trained via FL with the FedAvg does not reach the mean precision and recall of personalized models, but can surpass models trained with data from multiple individuals and decrease the number of false anomalies detected, as seen in the previous section. This section focus on trying the different aggregation functions offered by FLWR, in particular FedAvgM, FedAdagrad, FedAdam and FedYogi with two settings: using an individual threshold or an averaged one.

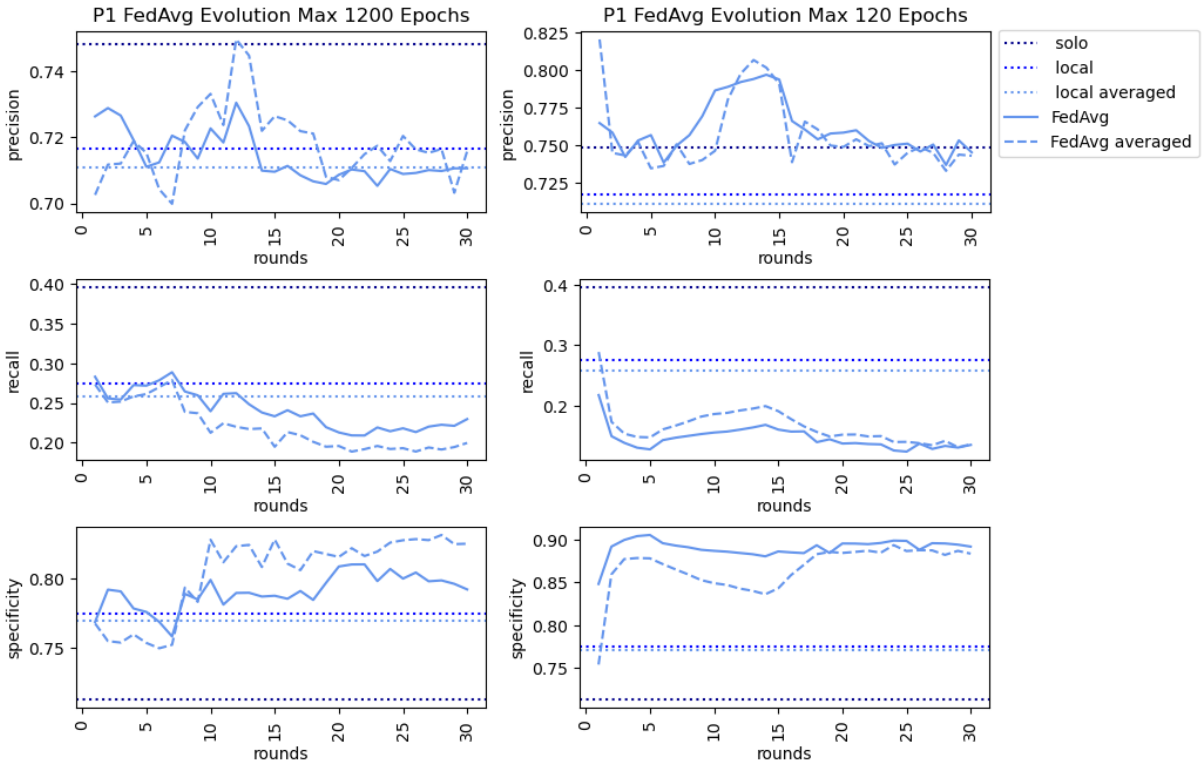


FIGURE 14. Metric scores evolution of the models trained with P1 with the FedAvg aggregation function for up to 1200 and 120 epochs per client per round.

We start allowing each client to train its local model for up to 1200 epochs on each round, as in the previous experiment, after which we decrease the number of epochs to 120. Only the second setting is applied to P2 due to difficulties in running the simulation with P2 (out of memory crash on the Jupyter notebook during the training process) and the remaining aggregation functions. The FedAvgM aggregation function is used setting the momentum to 0.5, while the remaining functions are used with the default values of FLWR. All graphics are shown with the solo, local, and local averaged scores previously presented.

The forced decrease in the number of epochs each client trains the local model for in each training round results in a lower mean recall score and higher mean precision and

specificity scores across all five different aggregation functions explored. These changes can be seen in Figure 14 with FedAvg, graphics of the remaining four aggregation functions can be found in Annex A (1200 epochs) and B (120).

Examining the behavior of the different aggregation functions when local training runs for up to 120 epochs, FedAvgM behaves similarly to FedAvg for both types of threshold (individual and averaged) except the scores are more erratic during the first rounds. The mean scores of both FedYogi and FedAdagrad remain stable during earlier rounds, with FedYogi doing so for longer. The averaged threshold results in the mean recall score starting high and remaining unchanging until it begins a drop that matches with the increase of the specificity (after round 20 for FedYogi, before 10 for FedAdagrad). The averaged precision also starts high and ends lowering. Utilizing individual thresholds results in curves with similar evolution but lower recall and precision and higher specificity than the using the averaged threshold. Finally, FedAdam presents a cyclical pattern that differs between the averaged and the individual, or personalized, threshold. As with the other functions, the averaged threshold results in higher recall but lower specificity. The evolution of the models aggregated with FedYogi and FedAdam can be seen in Figure 15, the affect of the remaining functions can be found in Annex B.

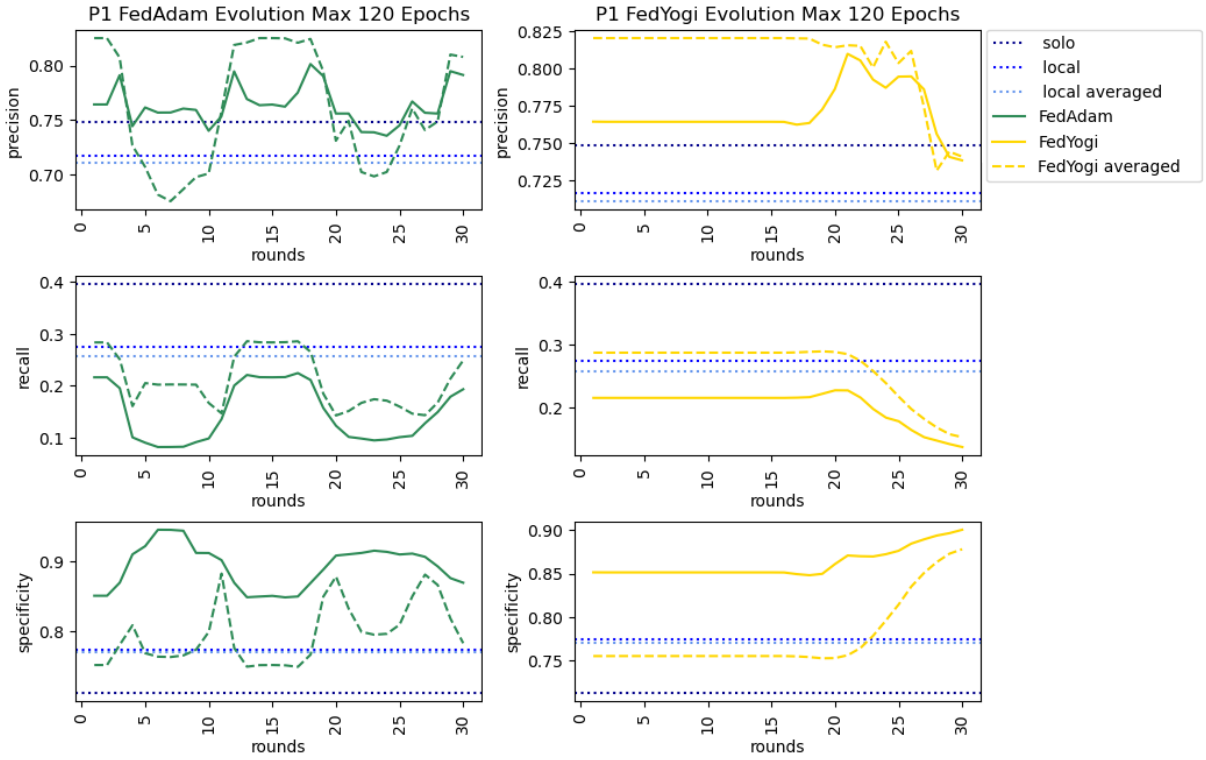


FIGURE 15. Metric scores evolution of the models trained with P1 with the FedAdam and FedYogi aggregation functions.

The differences in behavior found when training the models with P2 are as follow: FedAvg with averaged threshold achieved better mean recall but worse mean precision; the recall with individual, or personalized, threshold is higher than the averaged one; and the recall and specificity obtained with both FedAdagrad and FedYogi remain stable

during the entire process, when applying the personalized threshold. Figure 16 shows the evolution of FedAvgM and FedAdagrad, graphics with the other functions can be found in Annex B.

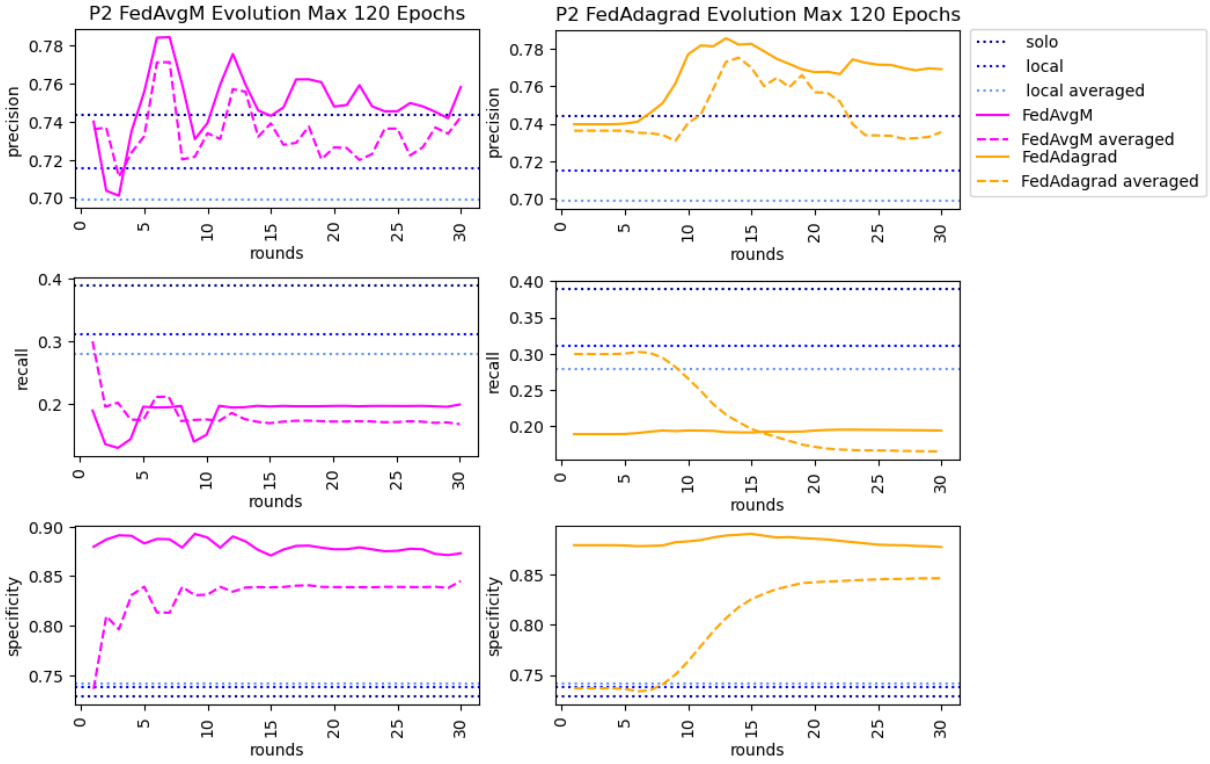


FIGURE 16. Metric scores evolution of the models trained with P2 with the FedAAvgM and FedAdaagrad aggregation functions.

With both datasets, FedYogi with the averaged threshold achieves the highest mean recall scores, reaching its maximum value within the first rounds and keeping it the longest.

Four of the five aggregation functions obtain models whose average loss decreases with the number of rounds, with FedAdam showing more of a cyclical evolution. The loss of the baseline samples is almost always less than that of the anomaly samples, but the difference is very small and all are below the average threshold. The average loss of the normal samples is also very similar to that of the baseline and anomaly samples. The average threshold approaches that of the solo models at later rounds, even becoming smaller with FedAvg and FedAvgM, suggesting the models are becoming better at reproducing the input but without that leading to being able to better distinguish between the baseline and the anomalies (Figure 17).

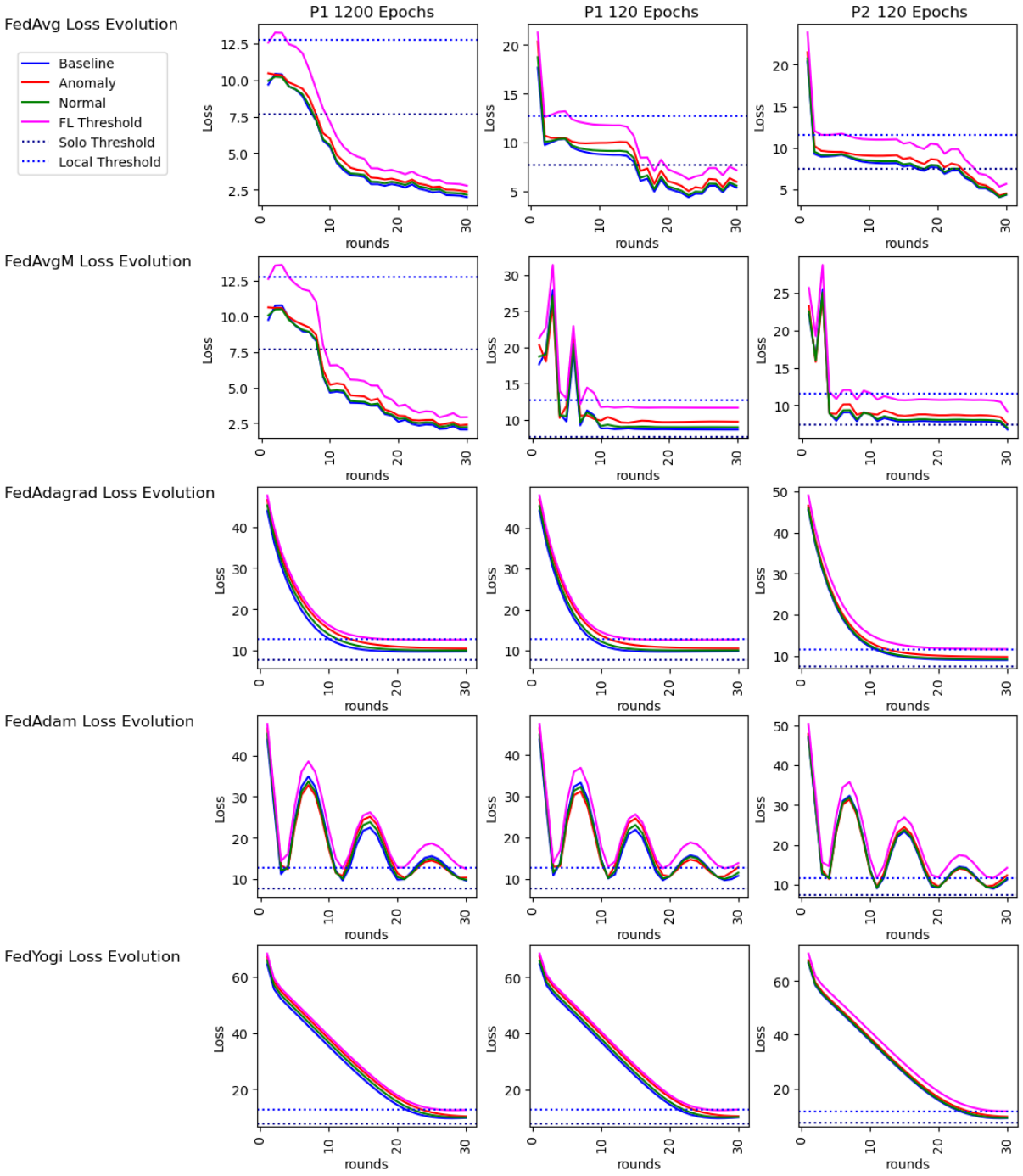


FIGURE 17. Evolution of the average loss obtained when training with each aggregation function.

4.7. Anomaly Detection Fine-Tuning

This work is exploring how an anomaly detection pipeline behaves when trained with FL to see how feasible it would be to implement such a system in a mobile app to alert the users of a possible COVID-19 infection. Given this context, detecting anomalies every hour might be bothersome for the user, or even confusing if they are told contradicting results in consecutive or close by hours. To address this issue, this section explores the impact of changing the anomaly detection sampling from hours to days, by varying

how many hours in one day need to be classified as anomalous so that the entire day is considered anomalous and also how many days should be considered anomalous for the alert to be issued, similarly to what has been done in Section 4.1. This is done using the results obtained in the previous section where the clients trained their models for 120 epochs between rounds.

By considering each hour sample, the mean recall scores obtained by the models trained via FL vary between 0.081 and 0.302, independent of the dataset, aggregation function and if the threshold in personalized or averaged, while the precision ranges from 0.691 and 0.825 and the specificity between 0.724 and 0.946. When considering the detection at the day level (1Day), the mean recall scores increase to a minimum of 0.371 and maximum of 0.594, with a penalty to the precision and specificity scores whose values variation shifts to $[0.647, 0.792]$ and $[0.473, 0.638]$, respectively. Requiring the previous day to also contain at least one identified anomalous hour (2Days) increases the range of values obtained by each metrics (precision: $[0.632, 0.806]$, recall: $[0.289, 0.610]$, specificity $[0.462, 0.703]$). Although the maximum value reached by the recall with 2Days is higher than with 1Day, examining the evolution of the curve shows it tends to be lower (Figure 18 and Annex C). Otherwise, the metrics of both 1Day and 2Days behave similarly.

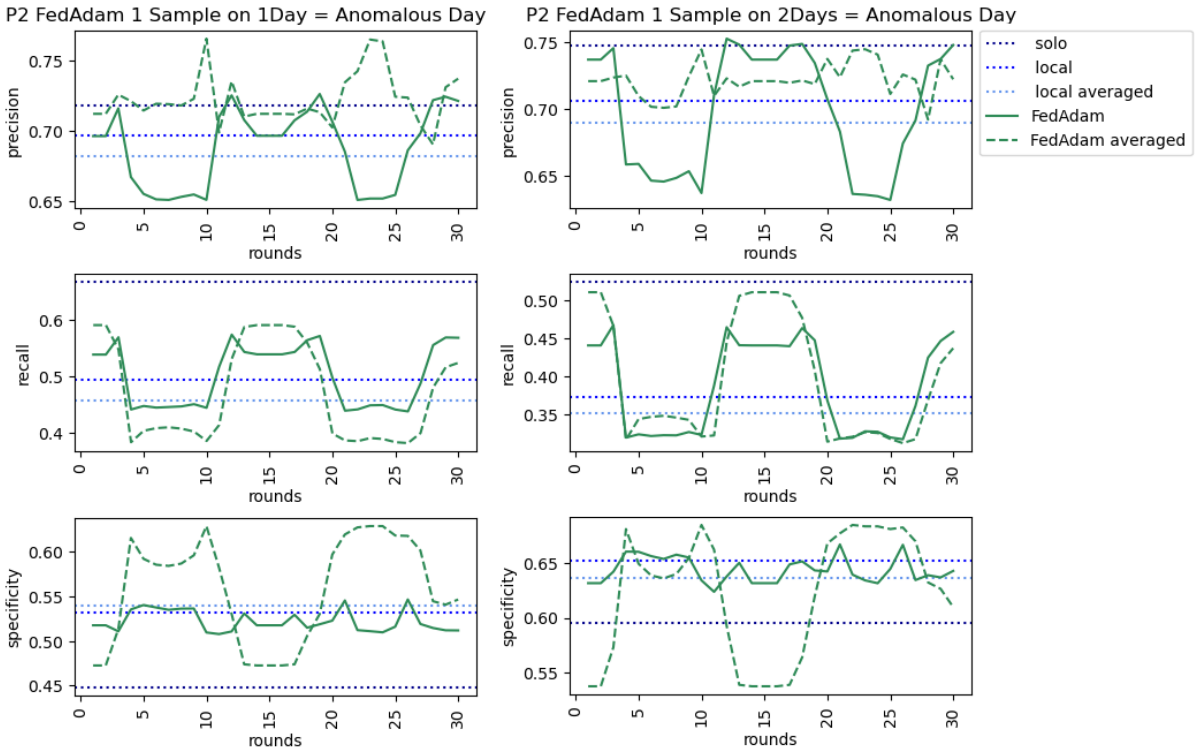


FIGURE 18. Metric scores evolution of models aggregated with FedAdam and trained with P2 when detecting anomalous days.

Varying the percentage of hour samples that are considered to classify one day as anomalous fails to improve the specificity and precision scores without lowering the recall. The only improvement is by setting the hours' percentage to 10%, which gives an increase

of 0.1 on the lowest mean specificity score without penalising the other two mean scores (Table 4.4).

TABLE 4.4. Approximate score ranges of varying the percentage hour anomalies required to label a day as anomalous.

%	Precision		Recall		Specificity	
	Min	Max	Min	Max	Min	Max
0	0.7	0.8	0.4	0.7	0.3	0.6
10	0.7	0.8	0.4	0.7	0.4	0.6
20	0.7	0.8	0.3	0.6	0.5	0.7
40	0.6	0.8	0.2	0.5	0.6	0.8
60	0.5	0.8	0.1	0.4	0.7	0.9

4.8. Individual Threshold Results

All results presented in this work, so far, have been the averages of the metrics scores obtained by the models when applied to each individuals' data. This section explores how the models perform at the individual level to see if any patterns arise, such as if there are individuals where all models perform equally well, or poorly.

To do so, the best models, that is the models with the highest mean recall score when detecting anomalous days with more than 10% of anomalous hour samples are selected. This selection is done by dataset (P1 and P2), by aggregation function (FedAvg, FedAvgM, FedAdagrad, FedAdam and FedYogi) and how the threshold is calculated (each individual creates their own threshold and calculating a shared threshold, averaged). Note that the highest recall is achieved during the first round of training in several occasions but, as these models are the worst at recreating the signals (Figure 17) and this value is often reached again in later rounds, the results from the first round of training are ignored during the selection process. The scores obtained by the solo, local, and local averaged models are also shown for comparison.

The precision appears with the most consistent behaviour across all models. Albeit, the models built using the averaged threshold consistently fail to detect some anomalies (in five of the individuals from P1 and eleven from P2), most other individual and model pairings achieve a score over 60% (Figure 19). In terms of recall, the averaged thresholds achieve worse scores in more individuals than the non-averaged, or personalized, anomaly detection (Figure 20). The models obtain lower specificity scores with the P1 dataset when compared to the recall of the same data, a difference that is not noticeable with the P2 dataset (Figure 21).

Overall, the solo models achieve higher precision with more of the individuals in the test than the FL approaches but the latter present higher recall, in particular with personalised thresholds and the P1 dataset. The local averaged approach presents worse scores than the FL models in terms of recall.

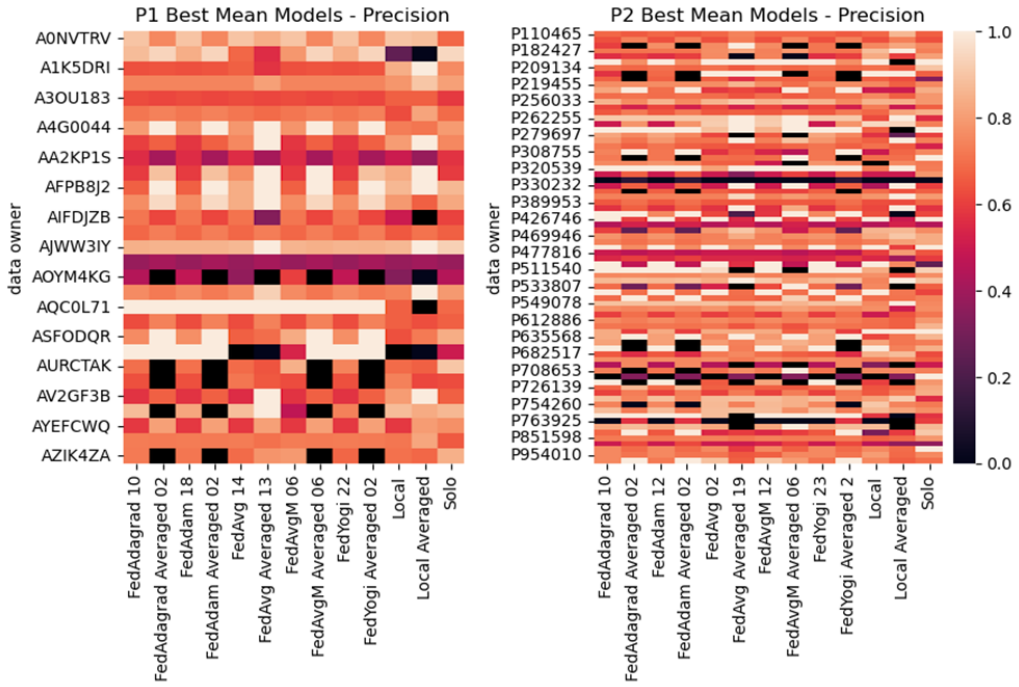


FIGURE 19. Heatmaps of the precision of the best models.

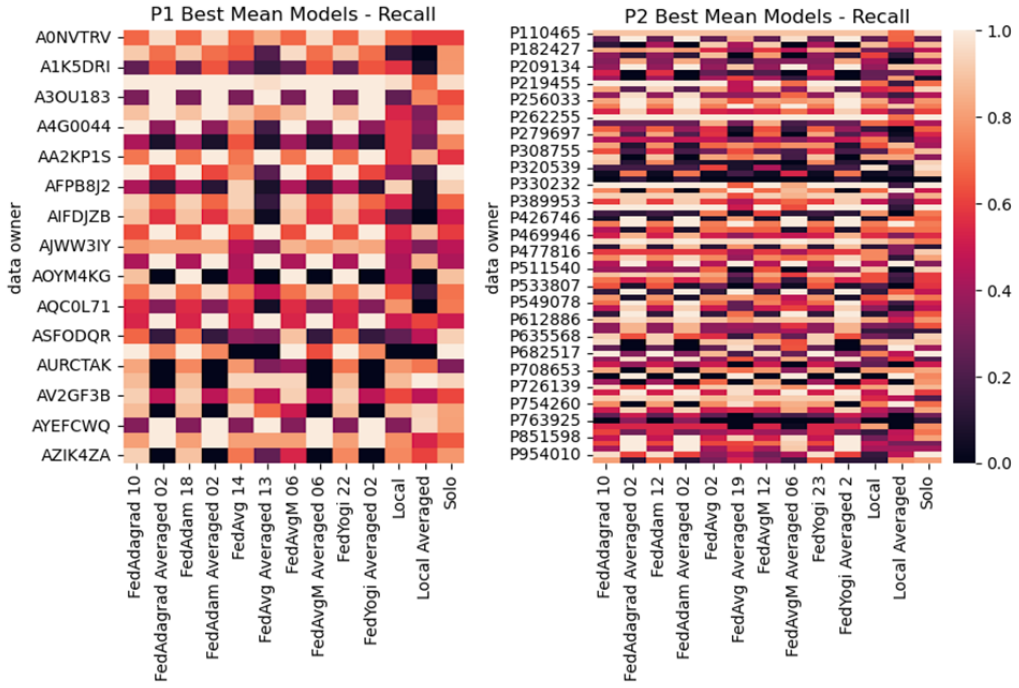


FIGURE 20. Heatmaps of the recall of the best models

4.9. Testing the Models with New Data

In this section, models trained with multiple individuals (Local and FL models) are applied to data from new individual, that is, from individuals that do not train the models. All results presented in this section are obtained by using averaged threshold - the mean threshold obtained by the individuals that participated in the training of the models -

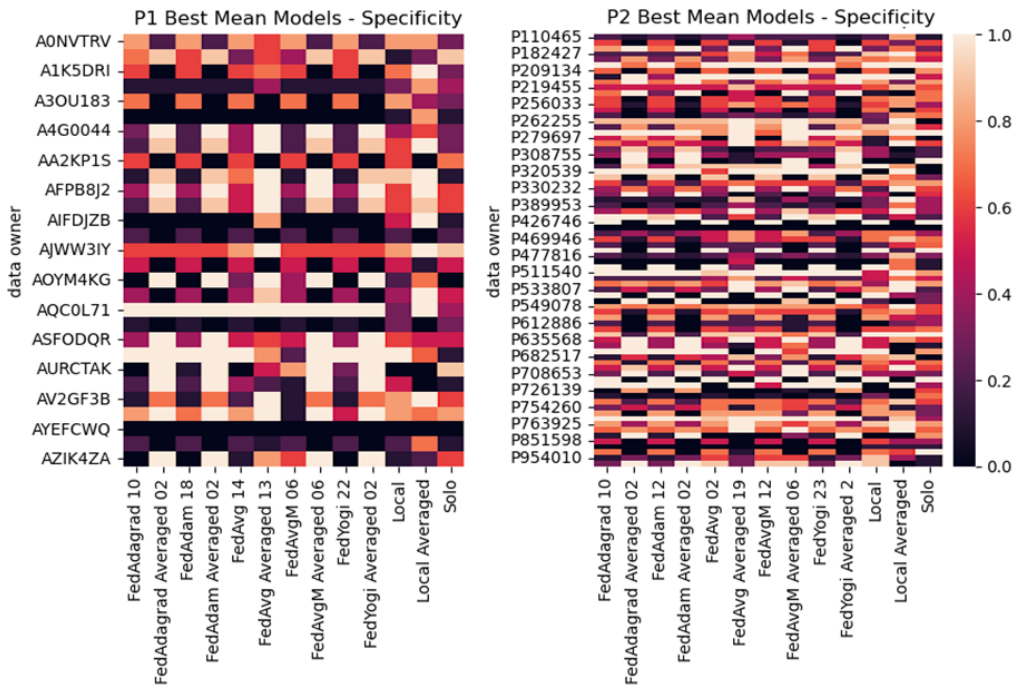


FIGURE 21. Heatmaps of the specificity of the best models

and entire days are considered anomalous when 10% of its hour measurements are detected as anomalies by the pipeline.

Three datasets with distinct characteristics are explored:

- **P1-Other**, consisting of data from nine individuals with a known infection other than COVID-19, collected at the same time as P1 and containing both "healthy" or "normal" and "sick" or "anomalous" samples;
- **P1-Healthy**, which contains only "healthy" data of seventy-three individuals, also collected at the same time as P1;
- **HSM**, Data from patients admitted to the intensive care unit of the HSM with known infection where, naturally, all samples are assumed to be anomalous.

The hospital data is further divided into patients with known COVID-19 infection and patients with other types of diagnosis.

Figure 22 shows the mean scores obtained when applying the previously trained models to P1-Other with the averaged threshold. The specificity reaches higher values than those seen in the previous experiments. However, both the precision and recall show lower values than before. This could be because some of the illnesses contained in this dataset don't affect the RHR as much as COVID-19.

P1-Healthy only contains healthy samples, meaning that only the specificity scores can be examined. The range of mean scores obtained with this dataset is within the values already seen using P1 and P2 (Figure 23).

Likewise, only the values obtained for recall are explored when testing on SMH since all samples are assumed to be anomalies. Figure 24 shows the mean evolution of the score obtained by the models when applied to the data of patients with known COVID-19

infection. The results for these models in the previous sections achieve similar performances and notably the maximum mean recall is equal to what was obtained with the original datasets. nevertheless, the worst scores now obtained are lower than the minimum that was observed when testing P1 and P2 in previous sections, occurring during the first rounds aggregated with FedAvgM or with the cyclical FedAdam. FedYogi and FedAdagrad sustain the highest score for several consecutive rounds.

Data from individuals found in the hospital data marked with other types of diagnosis achieves similar mean curves and ranges of recall scores, as it can be seen in Annex D.

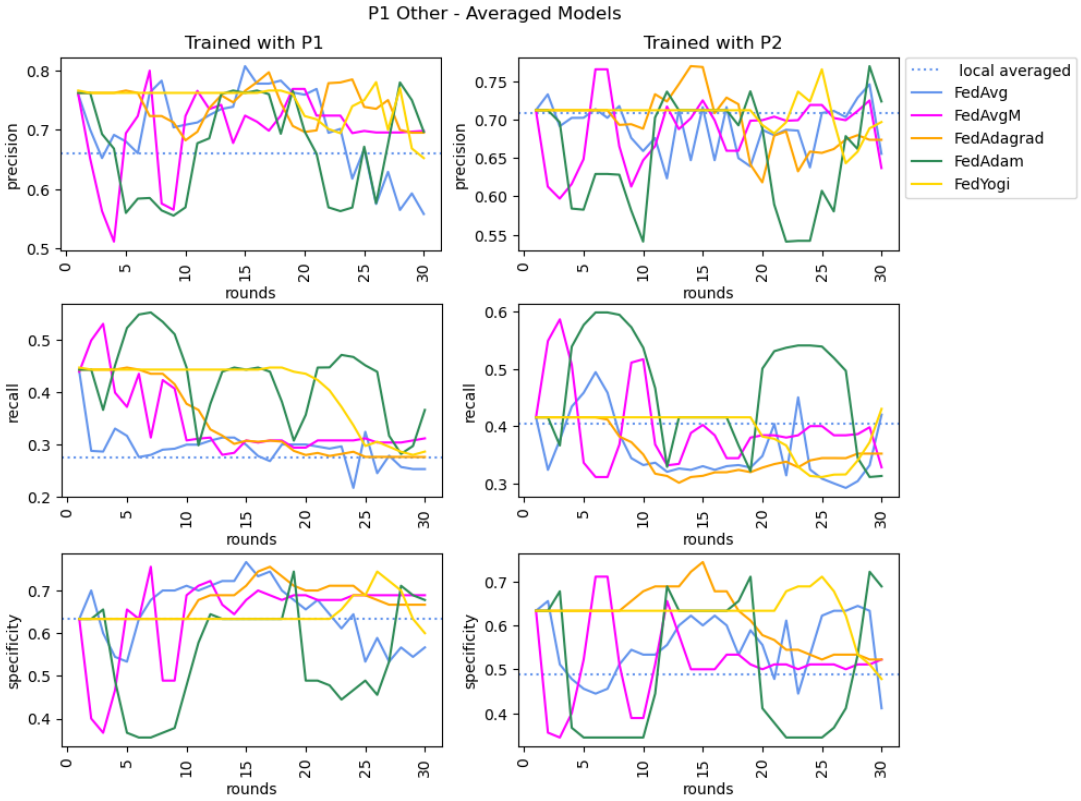


FIGURE 22. Metric evolution of the models trained with P1 and P2 applied to P1 Other with an averaged threshold.

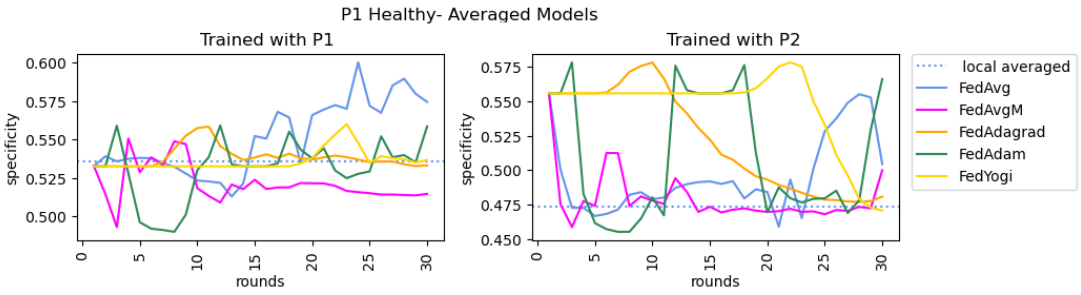


FIGURE 23. Metric evolution of the models trained with P1 and P2 applied to P1 Healthy with an averaged threshold.

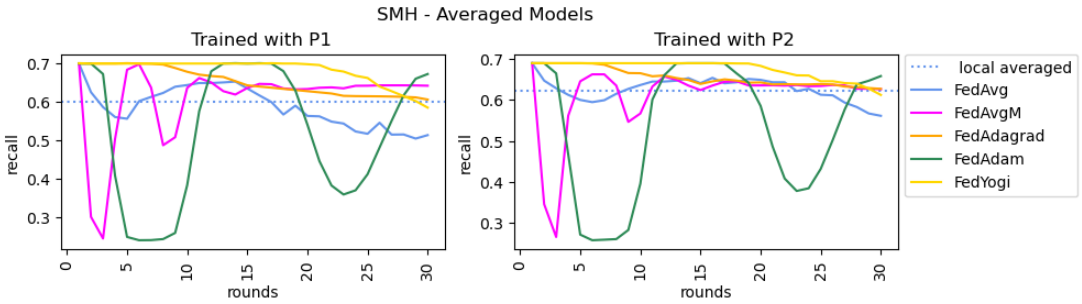


FIGURE 24. Metric evolution of the models trained with P1 and P2 with an averaged threshold applied to hospital patients with known COVID-19 infection.

4.10. Summary

This section provides a small recap of the most important results presented throughout this chapter, which is expansive and contains a high variety of experiments. A graphical recap of the main experiments conducted during this work is shown in Figure 25.

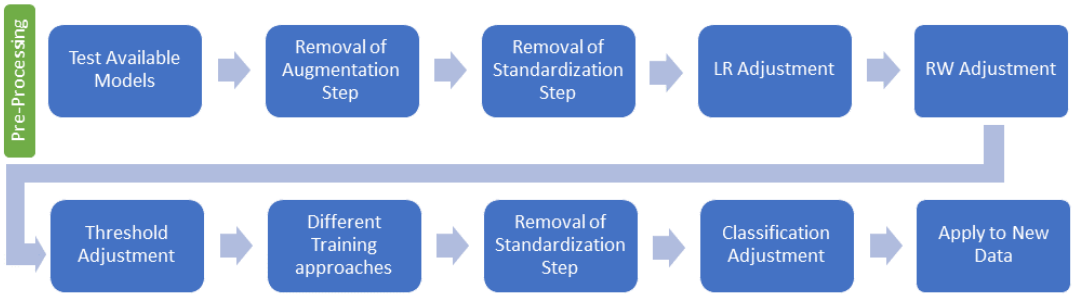


FIGURE 25. Flow of the main experiences conducted during this work.

The anomaly detection pipeline provided by G. Bogu and M. Snyder [**1**] where each individual trains their own pipeline to detect anomalies on their own data is used as the starting point to obtain a pipeline trained via FL that might be applicable to new users that might not contain healthy data to personalise the pipeline. To do this, the autoencoder model contained in the pipeline is trained via FL, a global threshold is created by averaging the threshold calculated by each individual that participates in the training process, and the standardization step is removed from the pipeline. These changes are highlighted in Figure 26.

Between having each individual train their own pipeline (Solo) and several create a single pipeline by treating multiple individuals as a single one (Local) or by using FL, the Solo approach obtains the highest recall. But the other two approaches might be applicable to users with no healthy data, since models trained and tested with data from P1 behaves similarly when tested with P2 and vice-versa, and the models trained with FL are able to surpass the performance achieved by the Local approach.

Regarding the aggregation functions explored in this work, FedYogi is deemed the most appropriate for achieving high and stable recall values for several training rounds,

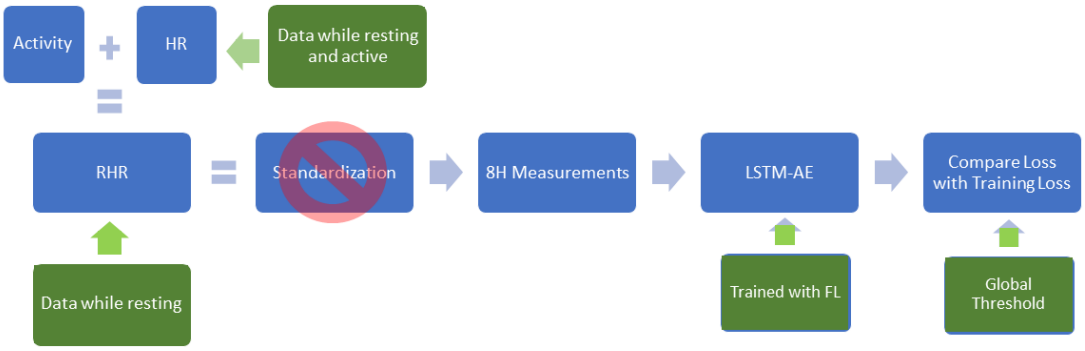


FIGURE 26. Main steps of the anomaly detection pipeline with the changes that allow to bring it into the paradigm of FL.

which might make deciding for how many rounds the training process should last less critical.

In terms of the threshold, lower values increase the number of detected anomalies but also increase the number of false detections. The same occurs if the detection of anomalies is done daily instead of every hour (to avoid overflowing users with alerts). This can be improved by altering the number of anomalous hours found within a day to consider that day anomalous.

Finally, when the pipelines obtained via the Local and the FL approaches are applied to new individuals there are less anomalies detected in individuals with illnesses other than COVID-19 (dataset P1-Other), and the amount of non anomalous samples in healthy individuals (P1-Healthy) correctly identified as such is similar. In terms of the data collected by the HSM, the performance is similar to what is obtained by the original datasets, with some models achieving higher recall but a few also performing worse.

CHAPTER 5

Alert System Prototype

This chapter proposes an implementation for an FL approach to detect anomalies in biometric data, such as RHR, that might be indicative of a COVID-19 infection. In fact, this architecture may be used for other arising health issues that may be infered from biometric signals alike RHR. The alert system is divided into two main components: the server and the client. Each one of these components will be described by a separate section.

It is important to note this prototype focuses only on the training of the model and calculation of the threshold. This is not yet an end-to-end application for the clients to receive either. This occurs both due to constraints of the FL framework used and by design, as it is unlikely all clients will participate in all training processes leading to different clients using different models and thresholds to detect anomalies in their data. Instead, the aggregated model and threshold resulting from the training processes should be sent to all clients via an update of the app that is still under development within the AIM Health project, which this work is a part of.

How the training process is started is also not considered. The scripts created in this work should be modified or added to enveloping code so the server starts a new training process regularly (this work assumes the training will not happen more than once a day) and the clients are warned of it so they can join the process, if they wish to. Due to the behavior of the framework, the server should initiate the process because it is capable of waiting for clients to connect to, while clients will terminate if they attempt to reach the server and it is not available.

5.1. Server

The server consists of one Python script, one configuration JSON file, one folder with the trained models and one csv file to save the thresholds of the anomaly detection and the timestamp of when they were calculated.

The inclusion of a JSON file serves to decrease the need to change the code on the script. Its edition allows for the change of the path to the folder where the models are stored and the path to the threshold folder. It also allows to change configuration parameters like how many clients must be available to train the model and which fraction of those should participate in each training round, the maximum time a round should take and what strategy, or aggregation function, should be used during the aggregation process, including setting some of the strategy's parameters. If the parameters of the strategy are set to null, the pre-existing configurations of the framework are used.

The models' folder should contain files in Hierarchical Data 5 (h5) Format format, with the identifier being composed of the day and training round in which the model is created. The models are kept in h5 format instead of just saving the weights to avoid hard coding the structure of the model. This way, if it becomes necessary to change the full model, one simply needs to add the new one to the folder. Moreover, while not including it in this proposal right now since its adequacy has yet been investigated and the project is not at a producing stage at the moment, this process allows for later inclusion of a champion/challenger process. Since the model the clients have at the time of the end of a training process might not be the newest model, the server always loads its newest model and sends the weights to all participating clients that are licensed to receive automatic updates.

The FLWR framework does not allow for the trained models to be saved natively. Instead, the aggregation strategy must implement the saving process. The Python script includes extended versions of all five strategies used in this work: FedAvg, FedAvgM, FedAdagrad, FedAdam and FedYogi. For each of them, the function '`aggregate_fit`', which aggregates the weights of the models trained by the clients on each round into a single global model, is overrun to call the parent function and save the model with the new aggregated weights. To calculate the averaged threshold, the function '`aggregate_evaluate`', which serves for the server to receive the results of the clients testing a newly aggregated model, is taken advantage of: all clients add their own threshold and number of samples used in the training and the server, using that information, calculates the mean and saves it in the thresholds file. If it becomes necessary to add a new strategy, it must be implemented with the code that allows to save the trained models and to calculate and save the corresponding thresholds, plus be added to the '`get_strategy`' function of the Python script, which allows the server to receive the strategy described in the configuration file, in order to function properly (any parameters of the new strategy that can be edited, should be added to the file).

This architecture allows to keep the model completely separated from the server script, as well as minimise the changes to the code when the training process needs to the altered. As mentioned previously, the model and threshold that result from the training process should be sent over to all existing users of the app via a separate process to avoid users who don't participate in the training having outdated information.

The Server communicates with each Client to know who is available to participate in the training process, to send the weights of the global models, and to receive the weights of the Clients' models and thresholds, for a total of five main interactions per training round. A scheme of these moments can be seen in Figure 27.

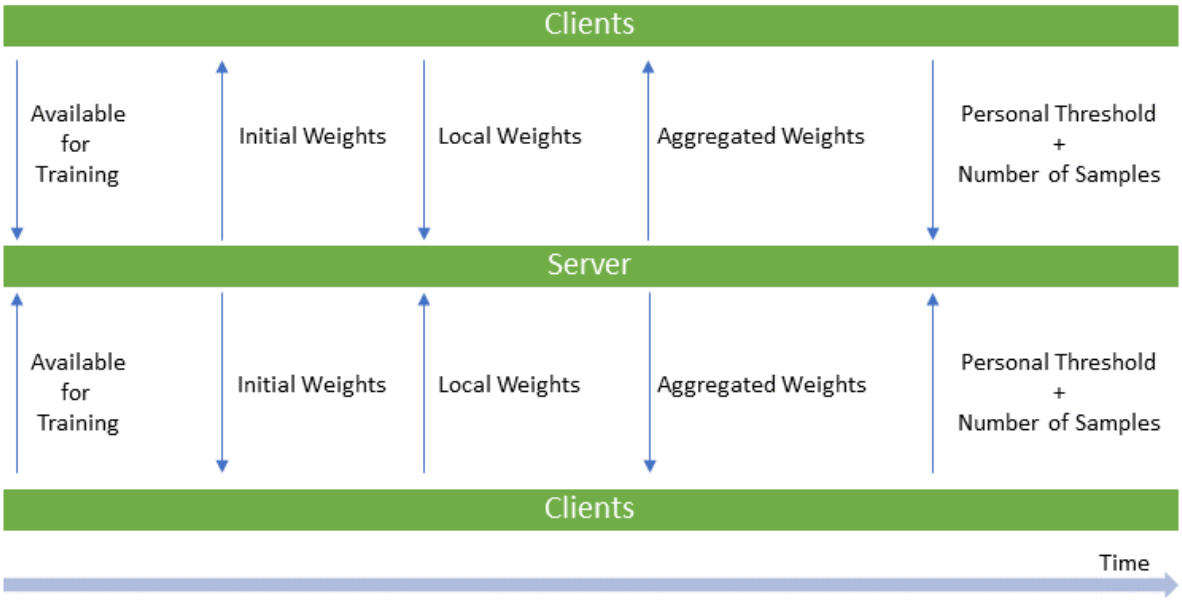


FIGURE 27. Information exchanged between the Server and the Clients.

5.2. Client

The client consists of a Python script, a configuration JSON file, a csv file with their data, another with the averaged thresholds and a h5 file with the newest available model.

Because the goal of the AIM Health project is to create a mobile app and the data used in this anomaly detection process will be collected via a variety of wearable devices, the gathering and transformation of the data are not considered in the client script. Instead, it is presumed the client starts with the data file depicting:

- DateTime: timestamp of when the data was measured;
- Data: the measured data (which in this work is the RHR);
- Loss: the loss obtained by the sample from going through the model
- Threshold: the threshold used to label the sample as anomalous or normal
- AnomalousHour: if the measurement is considered an anomaly by the model and threshold;
- PercentageAnomalousDay: if the day the sample was collected is considered an anomaly based on the percentage of anomalous samples found during it;
- AnomalousDay: if the day is considered anomalous based on the classification of previous days;
- ConfirmedAnomaly: this column serves to correct mistakes made by the anomaly detection process[1];
- DateTimeOfDetection: when the samples were evaluated.

This file should be updated regularly by enveloping code that was not considered during the course of this work.

[1]If a day is marked as True then it is anomalous, regardless of the result of the other columns, and is normal if marked as False. If a row is empty, the previous columns must be considered.

The client contains two distinct processes: participating in the training process and detecting anomalies in their data. The latter should be set to occur regularly and consists in:

(1) Loading the model and the latest threshold value;
(2) Loading the data and discovering which samples must yet be analysed by the anomaly detection process based on when they were collected and the last time this process happened;
(3) Classification of each sample based on the loss generated by the model and the threshold
(4) Classification of the day based on the percentage of anomalous samples present and the minimum percentage needed for the day to be considered anomalous;
(5) Classification of the day based on the number of previous days that were considered anomalous, according to the stipulated number of days necessary.

Once the process is over, the data file is updated.

For the training process, the client must first ensure it has enough data to participate. This means that data pertaining to thirty to twenty days before the current date, or day chosen to act as day zero (the functions that require this date have it set up as a parameter with the current date as the default value), for training and twenty to ten days before for testing must be retrieved and both sets of data contain enough samples to input into the model. Because the experiments where the anomaly detection is observed at the day level resulted in a high amount of false positives, it is possible to set up a maximum number of anomalous days the training and testing data can contain and still be used in the FL process, for example, it can be decided that one anomalous day in the extracted data does not disqualify the user from participating. If the data is acceptable, it is transformed to be inputted into the model and the client connects to the server.

The actual client class that connects to the server is an extension of `NumPyClient` class offered by FLWR. The constructor receives the training and test input data, the current model (used to retrieve the structure of the model because the FL process does not transfer it) and configurations for the local training. The previously mentioned class contains two functions necessary for the FL process: '`fit`' and '`evaluate`. The former receives the global model's parameters, or weights, from the server and retrains it with local data, sending the new weights back to the server plus the number of samples used for the training. The latter is meant to receive the new global model, test its performance on the client's test data and send the results to the server. Since the server expects to receive the loss obtained by the model evaluation, the length of the test dataset and a dictionary with a free structure, the ´`evaluate`' function uses the new model and the training data to create the local threshold, and sends it to the server inside the dictionary, along with the number of training samples so the server can create the averaged threshold.

All constants, parameters, paths and the names of the columns present in the data file can be edited via the configuration file. Enveloping code to make the two processes start

automatically must ensure the training process only begins after the server is expecting clients and the detection process should run at least once before the client participates in the training process for the first time.

Conclusions

This work set out to explore how an anomaly detection model to detect a possible COVID-19 infection behaves when trained via FL. Moreover, this model was later applied to data from novel individuals, in particular data collected during the normal daily activities of a hospital's ICU, to understand its generalization performance when applied to new anomalous data.

The first research question, RQ1, sets a focus on investigating the state-of-the-art regarding ML models to detect COVID-19 infections using wearable or mobile devices and discovering what FL applications exist, namely within the context of health anomaly detection of COVID-19 infection and if they make use of data acquired by wearable or mobile devices or sensing data collected with medical devices. The literature review revealed several works about detecting COVID-19 infections by examining data collectable via wearable devices, some purely statistical and some that implement ML. All the discovered works explore signals related to the heart, such as the heart rate, the resting heart rate (RHR) and blood pressure, with a few also examining the oxygen saturation (SpO2) and a person's temperature. Unfortunately, the SpO2 sensors found in wearable devices tend to not be very accurate and these often do not possess temperature sensors. The majority of the works focused on RHR signals and on the same publicly available dataset, that of Mishra et al. [**9**], which this work also explores. Three of the studies focus on anomaly detection on data collected via wearable devices, each being based on the previous one. No work focused on detecting COVID-19 via data from wearable devices that applied FL was found.

To answer RQ2, that is, if FL is a feasible approach for personal health anomalies detection, this work focuses on recreating a FL context of one of the three works that explore anomaly detection with publicly available data and code. The chosen work was that of G. Bogu and and M. Snyder [**1**], using public data, where the data of each individual is used to create a personalised LSTM autoencoder model and threshold to detect anomalies in the individual's data. Simply applying FL with the default aggregation function FedAvg to their training process achieves a worse mean recall when compared to the mean recall of each individual training their own personal model, but higher specificity which leads to a higher precision score in some training rounds. In other words, the model detects fewer anomalies but, when it does, it is more certain they truly are anomalies.

Besides the publicly available data gathered by T. Mishra et al. [**9**] and A. Alavi et al. [**6**], this work also explores data from HSM for the development of the enveloping

project AIM-HEALTH. This data pertained to individuals with known COVID-19 infection, with the exploration revealing no samples could be considered healthy and used for training the selected anomaly detection process. This led to RQ3 and RQ4. In order to answer them changes to the pre-processing of the data and how to calculate the threshold for anomaly detection had to be explored. Changes included removing the standardization of the data, as there is no known healthy data to serve as a baseline. The removal of the standardization results in a decrease of the mean precision and recall, which could be reverted by increasing the LR. Halving the RW results in a compromise deemed acceptable between the loss of patients and the loss of performance of the model.

Still focusing on non-FL training, different ways to calculate the threshold are explored, utilizing the new pre-processing and LR configuration. In general, methods that lead to a higher threshold achieve higher mean precision and specificity but lower recall. Assuming it is preferable to decrease the number of false negatives, that is, the number of anomalous samples labelled as normal, or increase the recall, the threshold set by $\overline{X} + \sigma$ is deemed the best, as it obtains the higher mean recall.

For this work, the FL training used each source of data working as a different client and also explored the possibility of having each individual create its own threshold and the creation of an average threshold both due to the FL nature of this work and the lack of data in the hospital dataset that allows it to create a threshold. Between the individual and the local approaches, the individual achieves better performance independently of how the threshold is calculated (individual or averaged). But the FL models achieve higher precision than the individual models in several rounds due to a decrease of the number of false anomalies detected and surpass or come close to the mean recall of the local models. The aggregation functions FedYogi, FedAdagrad and FedAdam, in particular, when coupled with an averaged threshold achieve the highest mean recall of the FL approaches in several rounds, as well as the highest precision when applied to the data collected by Mishra et al. Of these three aggregation functions, FedYogi is deemed the best as it sustains its mean scores stable for longer, possibly making deciding the number of rounds the FL process should run for less critical. Interestingly, all aggregation functions present an averaged threshold that is higher than the mean loss of both the baseline and anomaly samples and FedYogi presents the closest lines, with the decline of its recall score matching its mean loss becoming less than what is achieved by the individual models.

The model detects anomalies in hourly samples, but a user likely will not want to be disturbed every hour and be told an anomaly was detected in one hour, only for the next to be normal, which can be both confusing and unnecessary. This work explores labelling a day as anomalous based on the percentage of anomalous hour samples it contains and if previous days are considered anomalous. The change from hours to days increases the recall with a slight penalty to the precision which is reverted when it is required that the previous day also have anomalous samples in order for it to be anomalous. To balance the rate of false negatives and false positives, this work suggests that anomaly detection

be made at the day level, requiring that at least 10% of the day's samples be anomalous, regardless of the classification of previous days but ensures this is easily modified, due to the low specificity obtained when tested with data of individuals with no known illness - between 0.53 and 0.55 with the FedYogi aggregation function with averaged threshold, which is considered the best of the approaches to increase the recall.

Applying the trained models and calculated averaged thresholds to the data collected by T. Mishra et al. pertaining to individuals with other illnesses shows similar mean precision and recall scores, reinforcing what was discovered during the literature review process - that the changes caused by COVID-19 are not restricted to this illness and this approach can detect a possible illness but it might not be COVID-19. This is seen once more when the models and thresholds are applied to the hospital data as the obtained mean recall scores are similar across all types of diagnoses.

With these results, the answer to RQ3 is that (non FL) pre-trained models trained with the data of multiple individuals can be just as accurate when detecting anomalies in users other than the ones they were trained upon, but having each user train its own model is better. And, although still not better than having a personalized model, it is preferable to use a model trained by other people's data via FL than by pooling the data into a single location.

In conclusion, the main contributions of this work are that of exploring the transportation of an anomaly detection pipeline, originally developed to be trained by the data of one individual to issue an alert in case of anomalous situations, into the FL paradigm and exploring how anomaly detection models behave when applied to data from novel users acquired in different environments. For the anomaly detection pipeline explored, if an individual has enough healthy data to train its own personalised model and a device, a model trained via FL might not offer advantages when compared to a personalised model. But the FL approach may allow for the detection of anomalies in RHR, which might be indicative of a health issue, in individuals who cannot train their personalised pipeline due to lack of training data or that are unwilling to allocate resources to do so.

It is important to note this work assumes the HR collected by hospital grade devices is equivalent to the RHR calculated from HR and activity data collected via wearable devices sold to the general public. This is not necessarily true so a study that compares the two types of data needs to be conducted.

In terms of the built prototype, two distinct components are created - the server and the client. The server is in charge of the FL training process and saving the resulting models and averaged thresholds. The client can detect anomalies in its data with the global model and threshold and participate in the training process when its data is deemed normal enough. Both components have several parameters that can be edited via a JSON file and the model can be changed to a different type without needing to alter any code. How to activate the training process automatically has not been explored, nor how to gather the data used for the training and anomaly detection process due to the variety

of available devices. This work also does not explore retraining the model with new data after a first FL training process.

The work of A. Alavi et al. [**6**] looks for deviations in the RHR only between midnight and seven in the morning to avoid changes to the HR due to events, such as watching a horror movie, that are not detected and can be excluded through the activity data. This work does not apply that data selection because not everyone has that sleeping schedule or wears their wearable device during the same hours. It would be interesting to see how the anomaly detection pipeline acts when trained with data from people with similar habits or demographic characteristics but specialising the model in such ways might decrease the privacy offered by FL. Another unexplored suggestion is to break the day into periods longer than an hour and do the final detection based on those rather than the daily percentage of anomalous hour samples. It should be noted the available data was labeled as anomalous and normal, or healthy, according to what is found in the literature review, but Abir et al., 2023 [**34**], who explore a similar anomaly detection pipeline without the use of FL, question if the current division is accurate as they achieve higher recall when discarding data from 14 days after the symptom onset.

References

[1] G. K. Bogu and M. P. Snyder, "Deep learning-based detection of covid-19 using wearables data," *medRxiv*, 2021. [Online]. Available: https://www.medrxiv.org/content/early/2021/01/09/2021.01.08.21249474

[2] F. Firouzi, B. Farahani, M. Daneshmand, K. Grise, J. Song, R. Saracco, L. L. Wang, K. Lo, P. Angelov, E. Soares, P.-S. Loh, Z. Talebpour, R. Moradi, M. Goodarzi, H. Ashraf, M. Talebpour, A. Talebpour, L. Romeo, R. Das, H. Heidari, D. Pasquale, J. Moody, C. Woods, E. S. Huang, P. Barnaghi, M. Sarrafzadeh, R. Li, K. L. Beck, O. Isayev, N. Sung, and A. Luo, "Harnessing the power of smart and connected health to tackle covid-19: Iot, ai, robotics, and blockchain for a better world," *IEEE Internet of Things Journal*, vol. 8, no. 16, pp. 12 826–12 846, 2021.

[3] D. R. Seshadri, E. V. Davies, E. R. Harlow, J. J. Hsu, S. C. Knighton, T. A. Walker, J. E. Voos, and C. K. Drummond, "Wearable sensors for covid-19: A call to action to harness our digital infrastructure for remote patient monitoring and virtual assessments," *Frontiers in Digital Health*, vol. 2, 2020. [Online]. Available: https://www.frontiersin.org/articles/10.3389/fdgth.2020.00008

[4] K. Alyafei, R. Ahmed, F. F. Abir, M. E. Chowdhury, and K. K. Naji, "A comprehensive review of covid-19 detection techniques: From laboratory systems to wearable devices," *Computers in Biology and Medicine*, vol. 149, p. 106070, 2022. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0010482522007806

[5] M. Altini, L. Dunne, and L. Dunne, "What's next for wearable sensing?" *IEEE Pervasive Computing*, vol. 20, pp. 87–92, 10 2021.

[6] A. Alavi, G. K. Bogu, M. Wang, E. S. Rangan, A. W. Brooks, Q. Wang, E. Higgs, A. Celli, T. Mishra, A. A. Metwally, K. Cha, P. Knowles, A. A. Alavi, R. Bhasin, S. Panchamukhi, D. Celis, T. Aditya, A. Honkala, B. Rolnik, E. Hunting, O. Dagan-Rosenfeld, A. Chauhanl, J. W. Li, C. Bejikian, V. Krishnan, L. McGuire, X. Li, A. Bahmani, and M. P. Snyder, "Real-time alerting system for covid-19 and other stress events using wearable data," *NATURE MEDICINE*, vol. 28, no. 1, pp. 175+, JAN 2022.

[7] A. Rahman, M. S. Hossain, G. Muhammad, D. Kundu, T. Debnath, M. Rahman, M. S. I. Khan, P. Tiwari, and S. S. Band, "Federated learning-based ai approaches in smart healthcare: concepts, taxonomies, challenges and open issues," *Cluster Computing*, vol. 26, no. 4, pp. 2271–2311, Aug 2022. [Online]. Available: https://doi.org/10.1007/s10586-022-03658-4

[8] "Aimhealth - ai-based mobile applications for public health response," https://istar.iscte-iul.pt/portfolio-posts/ai-based-mobile-applications-for-public-health-response/, 2021 (accessed January 18, 2023).

[9] T. Mishra, M. Wang, A. A. Metwally, G. K. Bogu, A. W. Brooks, A. Bahmani, A. Alavi, A. Celli, E. Higgs, O. Dagan-Rosenfeld, B. Fay, S. Kirkpatrick, R. Kellogg, M. Gibson, T. Wang, E. M. Hunting, P. Mamic, A. B. Ganz, B. Rolnik, X. Li, and M. P. Snyder, "Pre-symptomatic detection of covid-19 from smartwatch data," *Nature Biomedical Engineering*, vol. 4, no. 12, pp. 1208–1220, Dec 2020. [Online]. Available: https://doi.org/10.1038/s41551-020-00640-6

[10] M. M. H. Shandhi, P. J. Cho, A. R. Roghanizad, K. Singh, W. Wang, O. M. Enache, A. Stern, R. Sbahi, B. Tatar, S. Fiscus, Q. X. Khoo, Y. Kuo, X. Lu, J. Hsieh, A. Kalodzitsa, A. Bahmani, A. Alavi, U. Ray, M. P. Snyder, G. S. Ginsburg, D. K. Pasquale, C. W. Woods, R. J. Shaw,

and J. P. Dunn, "A method for intelligent allocation of diagnostic testing by leveraging data from commercial wearable devices: a case study on covid-19," *npj Digital Medicine*, vol. 5, no. 1, p. 130, Sep 2022. [Online]. Available: https://doi.org/10.1038/s41746-022-00672-z

[11] PRISMA, "Preferred reporting items for systematic reviews and meta-analyses," https://www.prisma-statement.org/, 2021 (accessed January 22, 2023).

[12] S. Chaudhary, R. Kakkar, N. Jadav, A. Nair, R. Gupta, S. Tanwar, S. Agrawal, M. Alshehri, R. Sharma, G. Sharma, and I. Davidson, "A taxonomy on smart healthcare technologies: Security framework, case study, and future directions," *Journal of Sensors*, vol. 2022, 07 2022.

[13] N. Anjum, M. Alibakhshikenari, J. Rashid, F. Jabeen, A. Asif, E. M. Mohamed, and F. Falcone, "Iot-based covid-19 diagnosing and monitoring systems: A survey," *IEEE Access*, vol. 10, pp. 87 168–87 181, 2022.

[14] C. Goergen, M. Tweardy, S. Steinhubl, S. Wegerich, K. Singh, R. Mieloszyk, and J. Dunn, "Detection and monitoring of viral infections via wearable devices and biometric data," *Annual Review of Biomedical Engineering*, vol. 24, 06 2022.

[15] S. Mehrdad, Y. Wang, and S. F. Atashzar, "Perspective: Wearable internet of medical things for remote tracking of symptoms, prediction of health anomalies, implementation of preventative measures, and control of virus spread during the era of covid-19," *Frontiers in Robotics and AI*, vol. 8, 2021. [Online]. Available: https://www.frontiersin.org/articles/10.3389/frobt.2021.610653

[16] K. M. Elshabrawy, M. M. Alfares, and M. A.-M. Salem, "Ensemble federated learning for non-ii d covid-19 detection," in *2022 5th International Conference on Computing and Informatics (ICCI)*, 2022, pp. 057–063.

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

[18] F. Gulamali and G. Nadkarni, "Federated learning in risk prediction: A primer and application to covid-19-associated acute kidney injury," *Nephron*, vol. 147, no. 1, pp. 52–56, Feb. 2023, funding Information: Dr. G.N. Nadkarni is supported by R01DK127139 and R01HL155915. Publisher Copyright: © 2022 S. Karger AG, Basel.

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

[20] A. Vaid, S. Jaladanki, J. Xu, S. Teng, A. Kumar, S. Lee, S. Somani, I. Paranjpe, J. Freitas, B. Wanyan, K. Johnson, M. Bicak, E. Klang, J. K. Young, A. Costa, S. Zhao, R. Miotto, A. Charney, and B. Glicksberg, "Federated learning of electronic health records to improve mortality prediction in hospitalized patients with covid-19: Machine learning approach," 02 2021.

[21] I. Dayan, H. R. Roth, A. Zhong, A. Harouni, A. Gentili, A. Z. Abidin, A. Liu, A. B. Costa, B. J. Wood, C.-S. Tsai, C.-H. Wang, C.-N. Hsu, C. K. Lee, P. Ruan, D. Xu, D. Wu, E. Huang, F. C. Kitamura, G. Lacey, G. C. de Antônio Corradi, G. Nino, H.-H. Shin, H. Obinata, H. Ren, J. C. Crane, J. Tetreault, J. Guan, J. W. Garrett, J. D. Kaggie, J. G. Park, K. Dreyer, K. Juluru, K. Kersten, M. A. B. C. Rockenbach, M. G. Linguraru, M. A. Haider, M. AbdelMaseeh, N. Rieke, P. F. Damasceno, P. M. C. e Silva, P. Wang, S. Xu, S. Kawano, S. Sriswasdi, S. Y. Park, T. M. Grist, V. Buch, W. Jantarabenjakul, W. Wang, W. Y. Tak, X. Li, X. Lin, Y. J. Kwon, A. Quraini, A. Feng, A. N. Priest, B. Turkbey, B. Glicksberg, B. Bizzo, B. S. Kim, C. Tor-Díez, C.-C. Lee, C.-J. Hsu, C. Lin, C.-L. Lai, C. P. Hess, C. Compas, D. Bhatia, E. K. Oermann, E. Leibovitz, H. Sasaki, H. Mori, I. Yang, J. H. Sohn, K. N. K. Murthy, L.-C. Fu, M. R. F. de Mendonça, M. Fralick, M. K. Kang, M. Adil, N. Gangai, P. Vateekul, P. Elnajjar, S. Hickman, S. Majumdar, S. L. McLeod, S. Reed, S. Gräf, S. Harmon, T. Kodama, T. Puthanakit, T. Mazzulli, V. L. de Lavor,

Y. Rakvongthai, Y. R. Lee, Y. Wen, F. J. Gilbert, M. G. Flores, and Q. Li, "Federated learning for predicting clinical outcomes in patients with covid-19," *Nature Medicine*, vol. 27, no. 10, pp. 1735–1743, Oct 2021. [Online]. Available: https://doi.org/10.1038/s41591-021-01506-3

[22] J. S. Sunny, C. P. K. Patro, K. Karnani, S. C. Pingle, F. Lin, M. Anekoji, L. D. Jones, S. Kesari, and S. Ashili, "Anomaly detection framework for wearables data: A perspective review on data concepts, data analysis algorithms and prospects," *Sensors*, vol. 22, no. 3, 2022. [Online]. Available: https://www.mdpi.com/1424-8220/22/3/756

[23] I. Aguilera-Martos, M. García-Barzana, D. García-Gil, J. Carrasco, D. López, J. Luengo, and F. Herrera, "Multi-step histogram based outlier scores for unsupervised anomaly detection: Arcelormittal engineering dataset case of study," *Neurocomputing*, vol. 544, p. 126228, 2023. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S092523122300351X

[24] S. Schmidl, P. Wenig, and T. Papenbrock, "Anomaly detection in time series: A comprehensive evaluation," *Proc. VLDB Endow.*, vol. 15, no. 9, p. 1779–1797, may 2022. [Online]. Available: https://doi.org/10.14778/3538598.3538602

[25] A. A. Cook, G. Mısırlı, and Z. Fan, "Anomaly detection for iot time-series data: A survey," *IEEE Internet of Things Journal*, vol. 7, no. 7, pp. 6481–6494, 2020.

[26] Y. Liu, Y. Zhou, K. Yang, and X. Wang, "Unsupervised deep learning for iot time series," *IEEE Internet of Things Journal*, vol. 10, no. 16, pp. 14 285–14 306, 2023.

[27] A. I. Rana, G. Estrada, M. Solé, and V. Muntés, "Anomaly detection guidelines for data streams in big data," in *2016 3rd International Conference on Soft Computing Machine Intelligence (ISCMI)*, 2016, pp. 94–98.

[28] H. Li and Y. Li, "Anomaly detection methods based on gan: a survey," *Applied Intelligence*, vol. 53, no. 7, pp. 8209–8231, Apr 2023. [Online]. Available: https://doi.org/10.1007/s10489-022-03905-6

[29] H. Mankodiya, P. Palkhiwala, R. Gupta, N. K. Jadav, S. Tanwar, B.-C. Neagu, G. Grigoras, F. Alqahtani, and A. M. Shehata, "A real-time crowdsensing framework for potential covid-19 carrier detection using wearable sensors," *Mathematics*, vol. 10, no. 16, 2022. [Online]. Available: https://www.mdpi.com/2227-7390/10/16/2927

[30] D. H. Barbhuiya, A. Dey, R. Ghosh, K. Das, K. Ray, and N. Medhi, "Resource aware fog based remote health monitoring system," in *IEEE INFOCOM 2022 - IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*, 2022, pp. 1–6.

[31] G. Quer, J. M. Radin, M. Gadaleta, K. Baca-Motes, L. Ariniello, E. Ramos, V. Kheterpal, E. J. Topol, and S. R. Steinhubl, "Wearable sensor data and self-reported symptoms for covid-19 detection," *Nature Medicine*, vol. 27, no. 1, pp. 73–77, Jan 2021. [Online]. Available: https://doi.org/10.1038/s41591-020-1123-x

[32] H. R. Cho, J. H. Kim, H. R. Yoon, Y. S. Han, T. S. Kang, H. Choi, and S. Lee, "Machine learning-based optimization of pre-symptomatic covid-19 detection through smartwatch," *Scientific Reports*, vol. 12, no. 1, p. 7886, May 2022. [Online]. Available: https://doi.org/10.1038/s41598-022-11329-y

[33] F. F. Abir, K. Alyafei, M. E. Chowdhury, A. Khandakar, R. Ahmed, M. M. Hossain, S. Mahmud, A. Rahman, T. O. Abbas, S. M. Zughaier, and K. K. Naji, "Pcovnet: A presymptomatic covid-19 detection framework using deep learning model using wearables data," *Computers in Biology and Medicine*, vol. 147, p. 105682, 2022. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S001048252200470X

[34] F. F. Abir, M. E. Chowdhury, M. I. Tapotee, A. Mushtak, A. Khandakar, S. Mahmud, and A. Hasan, "Pcovnet+: A cnn-vae anomaly detection framework with lstm embeddings for smartwatch-based covid-19 detection," *Engineering Applications of Artificial Intelligence*, vol. 122, p. 106130, 2023. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0952197623003147

[35] J. Skibinska, R. Burget, A. Channa, N. Popescu, and Y. Koucheryavy, "Covid-19 diagnosis at early stage based on smartwatches and machine learning techniques," *IEEE Access*, vol. 9, pp. 119 476–119 491, 2021.

[36] M. Gadaleta, J. M. Radin, K. Baca-Motes, E. Ramos, V. Kheterpal, E. J. Topol, S. R. Steinhubl, and G. Quer, "Passive detection of covid-19 with wearable sensors and explainable machine learning algorithms," *npj Digital Medicine*, vol. 4, no. 1, p. 166, Dec 2021. [Online]. Available: https://doi.org/10.1038/s41746-021-00533-1

[37] R. P. Hirten, L. Tomalin, M. Danieletto, E. Golden, M. Zweig, S. Kaur, D. Helmus, A. Biello, R. Pyzik, E. P. Bottinger, L. Keefer, D. Charney, G. N. Nadkarni, M. Suarez-Farinas, and Z. A. Fayad, "Evaluation of a machine learning approach utilizing wearable data for prediction of SARS-CoV-2 infection in healthcare workers," *JAMIA Open*, vol. 5, no. 2, p. ooac041, 05 2022. [Online]. Available: https://doi.org/10.1093/jamiaopen/ooac041

[38] B. Conroy, I. Silva, G. Mehraei, R. Damiano, B. Gross, E. Salvati, T. Feng, J. Schneider, N. Olson, A. G. Rizzo, C. M. Curtin, J. Frassica, and D. C. McFarlane, "Real-time infection prediction with wearable physiological monitoring and ai to aid military workforce readiness during covid-19," *Scientific Reports*, vol. 12, no. 1, p. 3797, Mar 2022. [Online]. Available: https://doi.org/10.1038/s41598-022-07764-6

[39] A. Aguado-García, A. Arroyo-Valerio, G. Escobedo, N. Bueno-Hernández, P. Olguín-Rodríguez, M. F. Müller, J. D. Carrillo-Ruiz, and G. Martínez-Mekler, "Opportune warning of covid-19 in a mexican health care worker cohort: Discrete beta distribution entropy of smartwatch physiological records," *Biomedical Signal Processing and Control*, vol. 84, p. 104975, 2023. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1746809423004081

[40] A. E. Mason, F. M. Hecht, S. K. Davis, J. L. Natale, W. Hartogensis, N. Damaso, K. T. Claypool, S. Dilchert, S. Dasgupta, S. Purawat, V. K. Viswanath, A. Klein, A. Chowdhary, S. M. Fisher, C. Anglo, K. Y. Puldon, D. Veasna, J. G. Prather, L. S. Pandya, L. M. Fox, M. Busch, C. Giordano, B. K. Mercado, J. Song, R. Jaimes, B. S. Baum, B. A. Telfer, C. W. Philipson, P. P. Collins, A. A. Rao, E. J. Wang, R. H. Bandi, B. J. Choe, E. S. Epel, S. K. Epstein, J. B. Krasnoff, M. B. Lee, S.-W. Lee, G. M. Lopez, A. Mehta, L. D. Melville, T. S. Moon, L. R. Mujica-Parodi, K. M. Noel, M. A. Orosco, J. M. Rideout, J. D. Robishaw, R. M. Rodriguez, K. H. Shah, J. H. Siegal, A. Gupta, I. Altintas, and B. L. Smarr, "Detection of covid-19 using multimodal data from a wearable device: results from the first tempredict study," *Scientific Reports*, vol. 12, no. 1, p. 3463, Mar 2022. [Online]. Available: https://doi.org/10.1038/s41598-022-07314-0

[41] H. B. McMahan, E. Moore, D. Ramage, and B. A. y Arcas, "Federated learning of deep networks using model averaging," *CoRR*, vol. abs/1602.05629, 2016. [Online]. Available: http://arxiv.org/abs/1602.05629

[42] A. K. Sahu, T. Li, M. Sanjabi, M. Zaheer, A. Talwalkar, and V. Smith, "On the convergence of federated optimization in heterogeneous networks," *CoRR*, vol. abs/1812.06127, 2018. [Online]. Available: http://arxiv.org/abs/1812.06127

[43] J. Wang, Q. Liu, H. Liang, G. Joshi, and H. V. Poor, "Tackling the objective inconsistency problem in heterogeneous federated optimization," *CoRR*, vol. abs/2007.07481, 2020. [Online]. Available: https://arxiv.org/abs/2007.07481

[44] S. P. Karimireddy, S. Kale, M. Mohri, S. J. Reddi, S. U. Stich, and A. T. Suresh, "SCAFFOLD: stochastic controlled averaging for on-device federated learning," *CoRR*, vol. abs/1910.06378, 2019. [Online]. Available: http://arxiv.org/abs/1910.06378

[45] X. Li, M. Jiang, X. Zhang, M. Kamp, and Q. Dou, "Fedbn: Federated learning on non-iid features via local batch normalization," *CoRR*, vol. abs/2102.07623, 2021. [Online]. Available: https://arxiv.org/abs/2102.07623

[46] T. F. Authors, "Flower: A friendly federated learning framework," https://flower.dev/, (accessed August 22, 2023).

[47] "flwr (python api reference) - flower framework 1.5.0," https://flower.dev/docs/framework/apiref-flwr.html, (accessed August 22, 2023).

[48] T. H. Hsu, H. Qi, and M. Brown, "Measuring the effects of non-identical data distribution for federated visual classification," *CoRR*, vol. abs/1909.06335, 2019. [Online]. Available: http://arxiv.org/abs/1909.06335

[49] S. J. Reddi, Z. Charles, M. Zaheer, Z. Garrett, K. Rush, J. Konečný, S. Kumar, and H. B. McMahan, "Adaptive federated optimization," *CoRR*, vol. abs/2003.00295, 2020. [Online]. Available: https://arxiv.org/abs/2003.00295

APPENDIX A

This annex contains the evolution of the mean metric scores of the models trained via FL with the P1 dataset when each client is allowed to train their local model for up to 1200 epochs during each round, for a total of 30 training rounds. Each graphic contains information related to applying a different algorithm during the aggregation process.

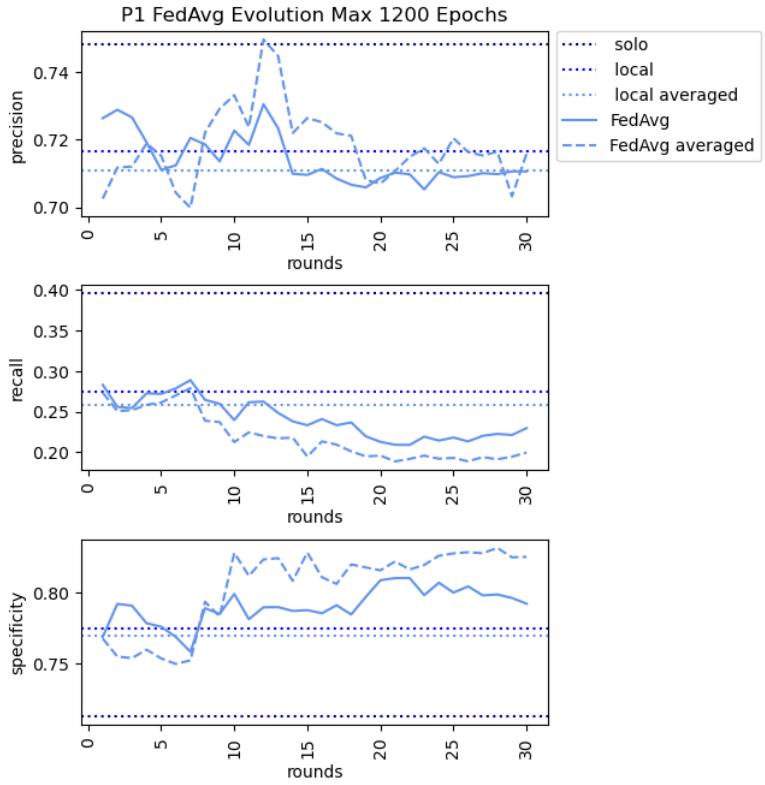


FIGURE 28. Metric evolution of the models trained with P1 with the FedAvg aggregation function for up to 1200 epochs.

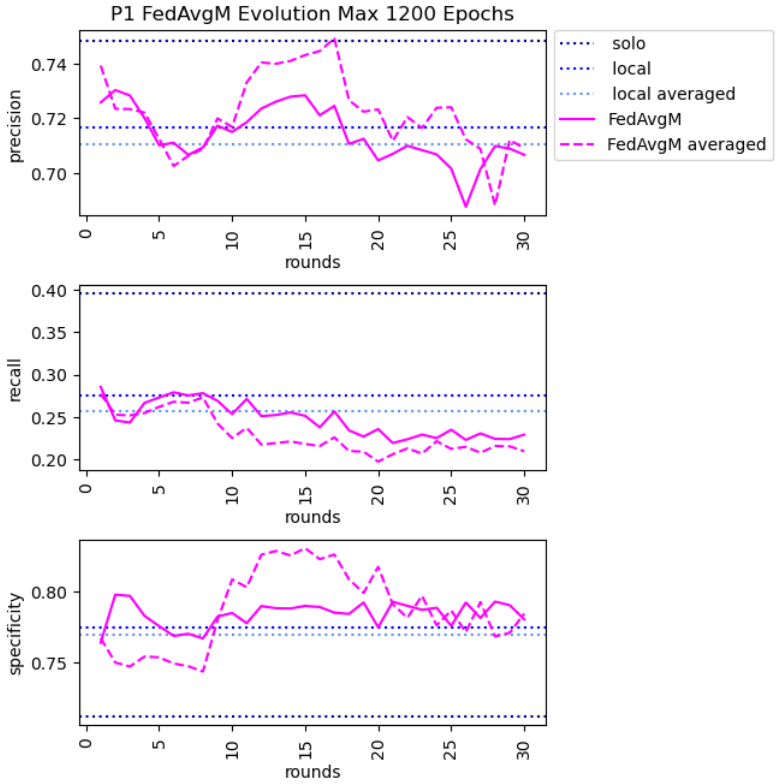


FIGURE 29. Metric evolution of the models trained with P1 with the FedAvgM aggregation function for up to 1200 epochs.

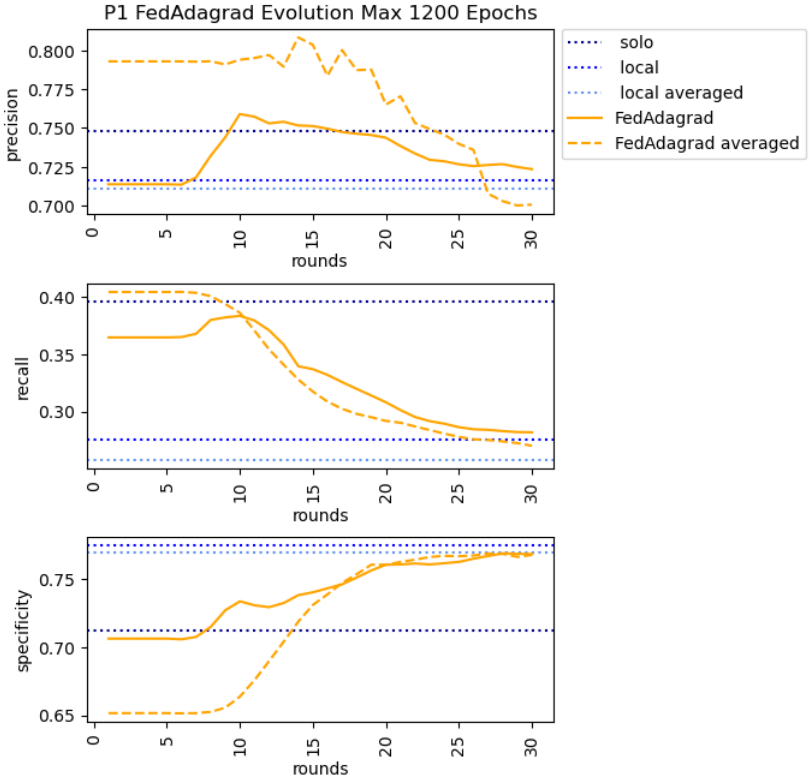


FIGURE 30. Metric evolution of the models trained with P1 with the FedAdagrad aggregation function for up to 1200 epochs.

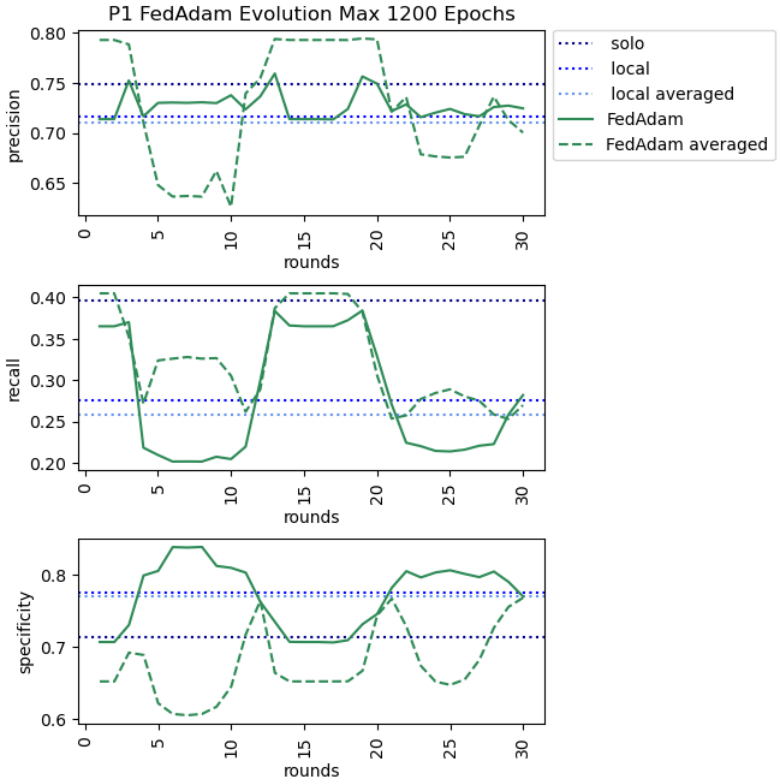


FIGURE 31. Metric evolution of the models trained with P1 with the FedAdam aggregation function for up to 1200 epochs.

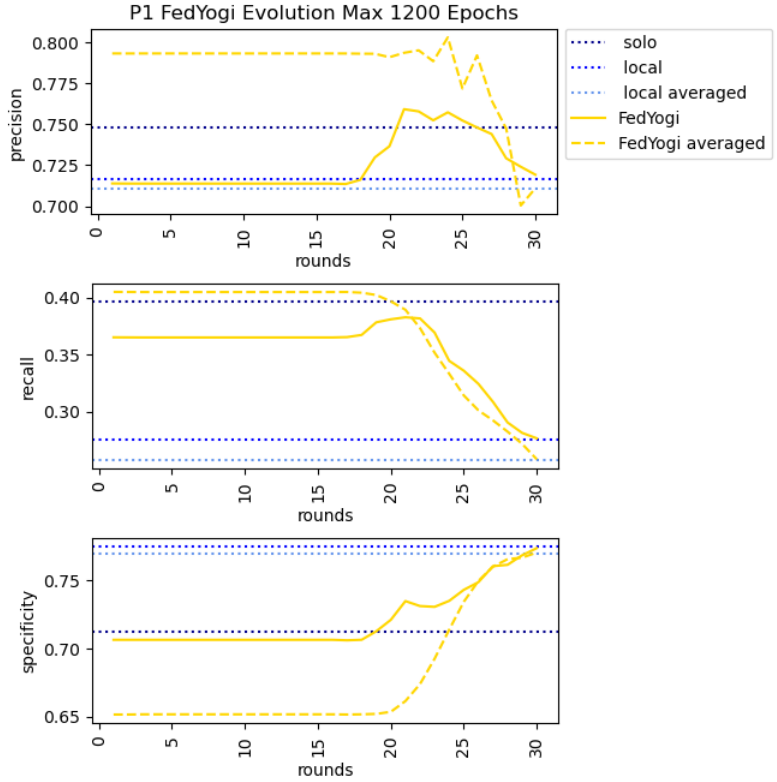


FIGURE 32. Metric evolution of the models trained with P1 with the FedYogi aggregation function for up to 1200 epochs.

APPENDIX B

This annex contains the evolution of the mean metric scores of the models trained via FL with the P1 and P2 datasets when each client is allowed to train their local model for up to 120 epochs during each round, for a total of 30 training rounds. The figures are organized by aggregation function and dataset.

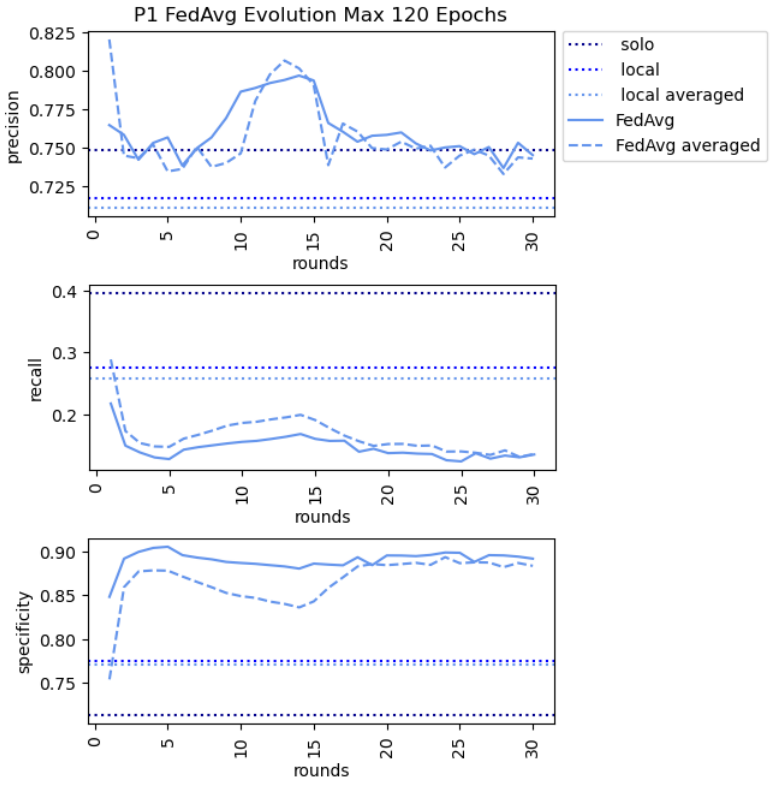


FIGURE 33. Metric evolution of the models trained with P1 with the FedAvg aggregation function for up to 120 epochs.

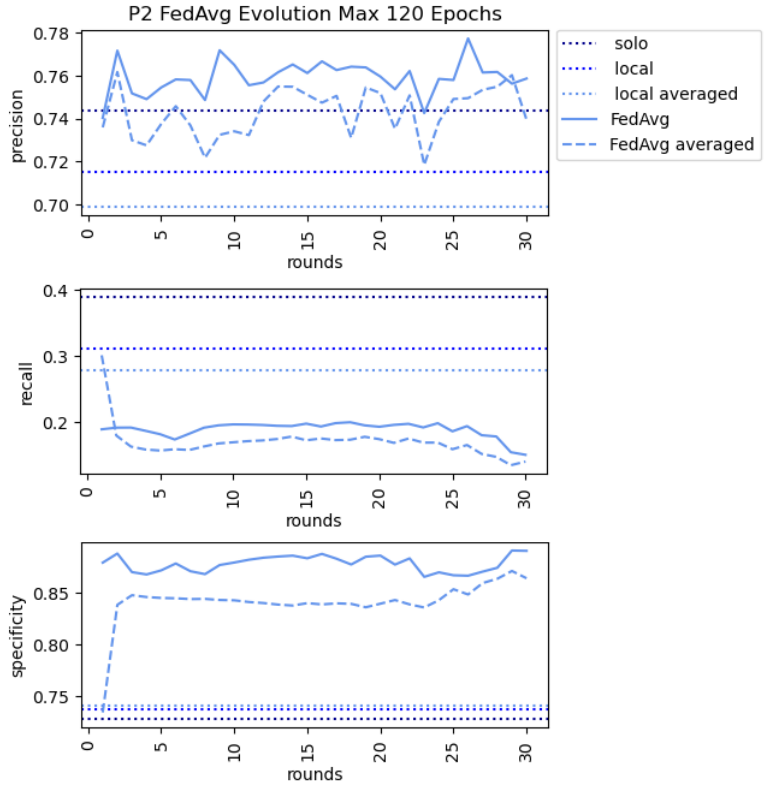


FIGURE 34. Metric evolution of the models trained with P2 with the FedAvg aggregation function for up to 120 epochs.

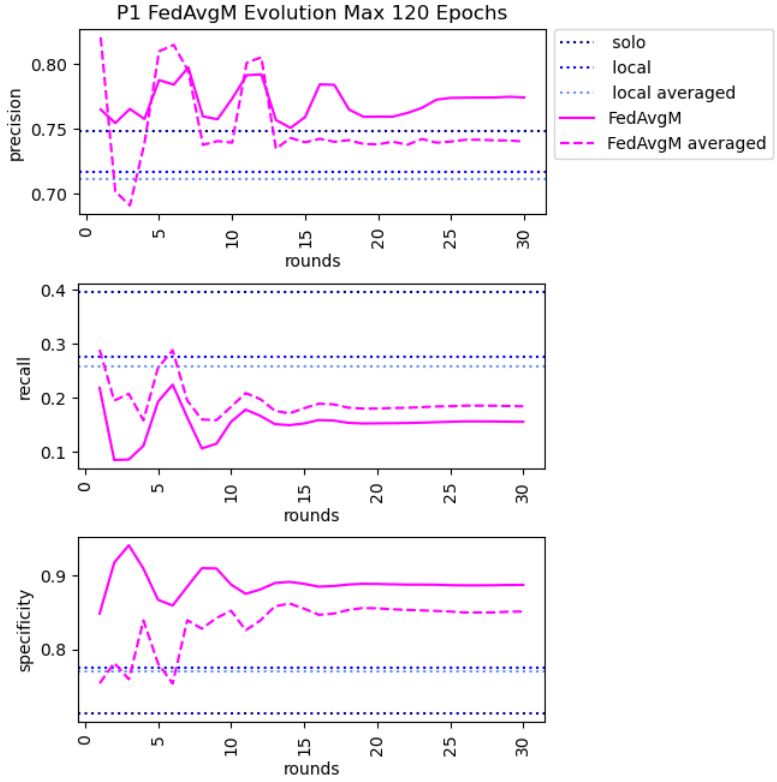


FIGURE 35. Metric evolution of the models trained with P1 with the FedAvgM aggregation function for up to 120 epochs.

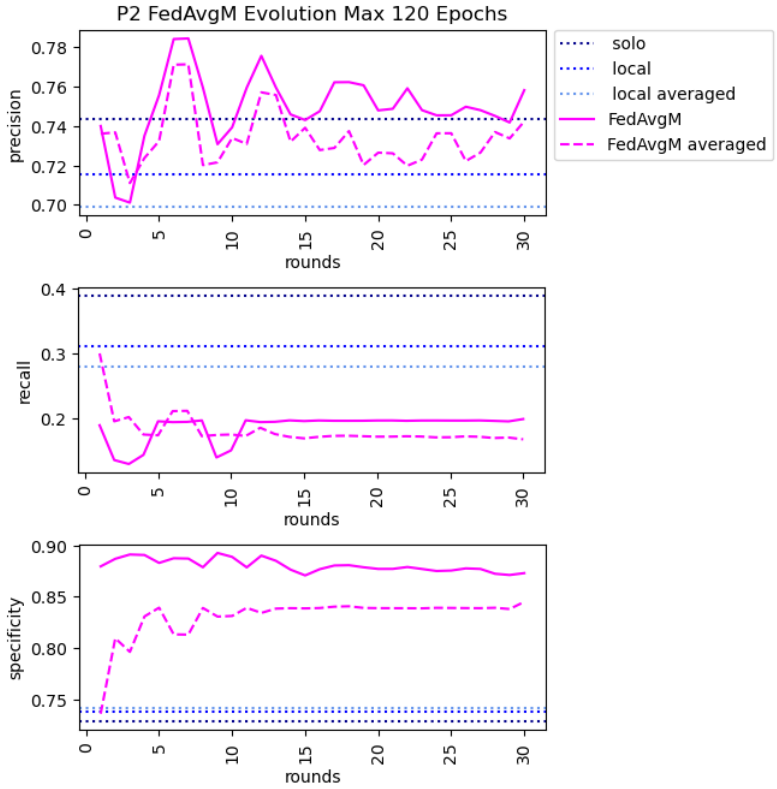


FIGURE 36. Metric evolution of the models trained with P2 with the FedAvgM aggregation function for up to 120 epochs.

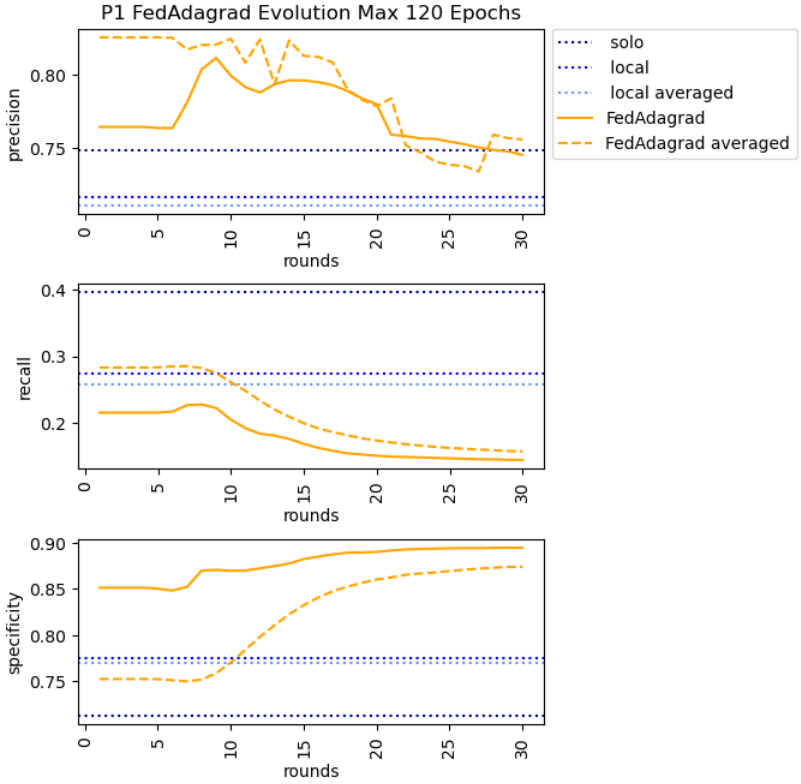


FIGURE 37. Metric evolution of the models trained with P1 with the FedAdagrad aggregation function for up to 120 epochs.

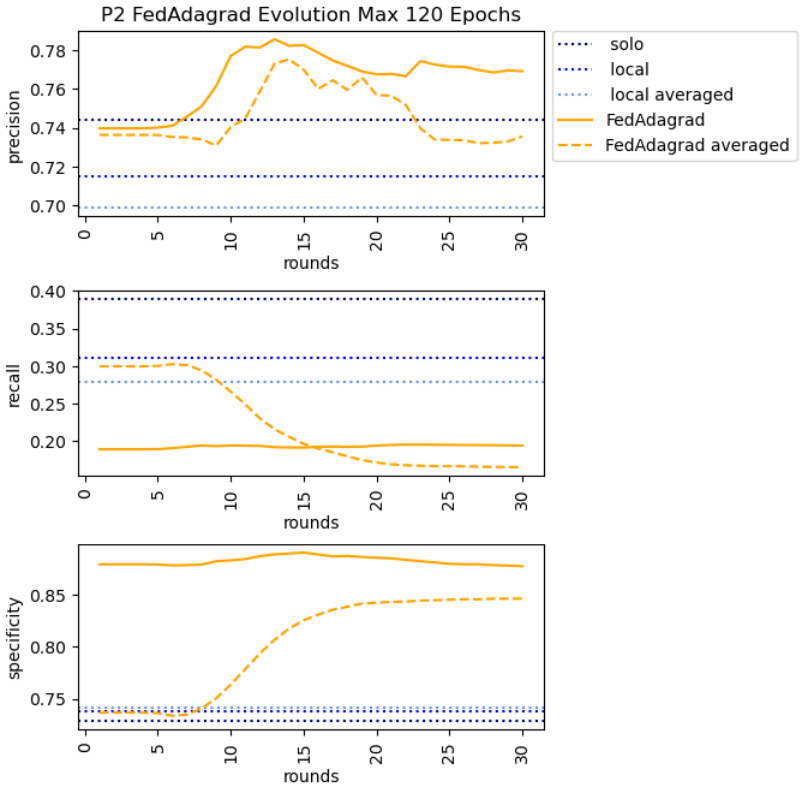


FIGURE 38. Metric evolution of the models trained with P2 with the FedAdagrad aggregation function for up to 120 epochs.

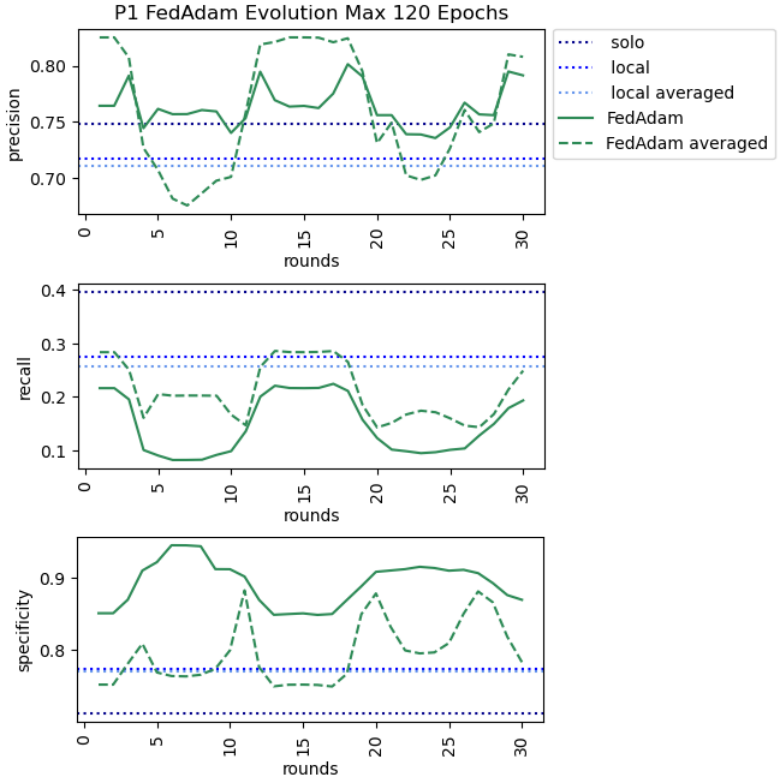


FIGURE 39. Metric evolution of the models trained with P1 with the FedAdam aggregation function for up to 120 epochs.

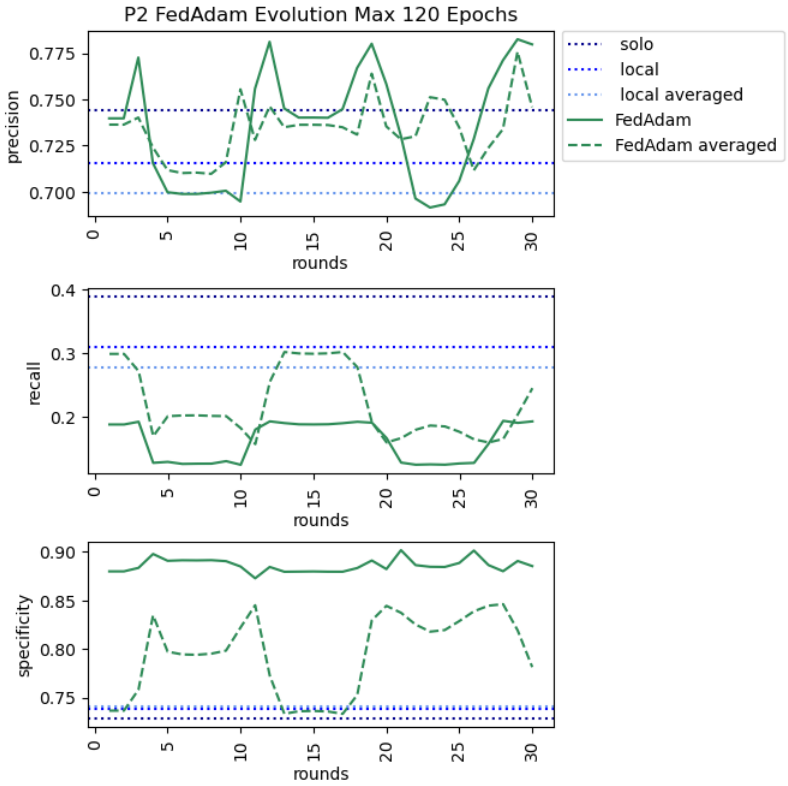


FIGURE 40. Metric evolution of the models trained with P2 with the FedAdam aggregation function for up to 120 epochs.

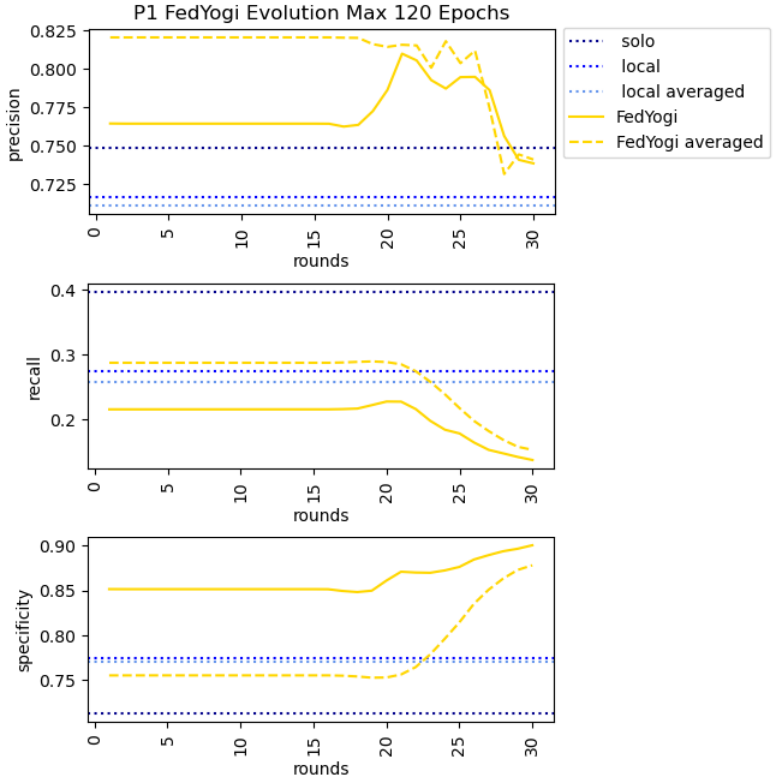


FIGURE 41. Metric evolution of the models trained with P1 with the FedYogi aggregation function for up to 120 epochs.

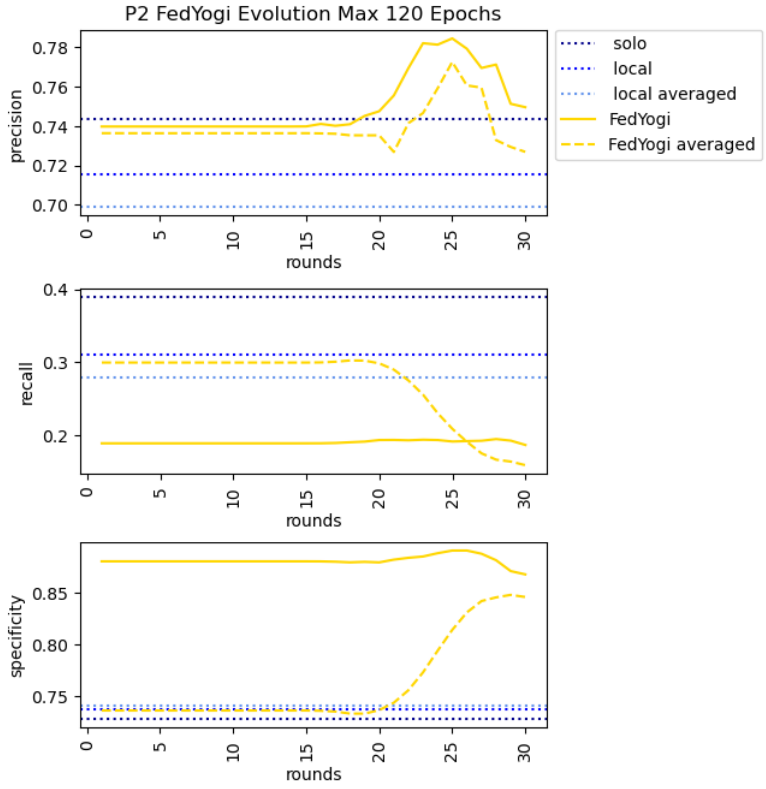


FIGURE 42. Metric evolution of the models trained with P2 with the FedYogi aggregation function for up to 120 epochs.

APPENDIX C

The figures in this annex are obtained by shifting the anomaly detection from 1 hour samples to an entire day (1Day) and 2 consecutive days (2Days). The clients of the FL process train their models for up to 120 epochs per round, for a total of 30 rounds. The figures are organized by dataset and aggregation function.

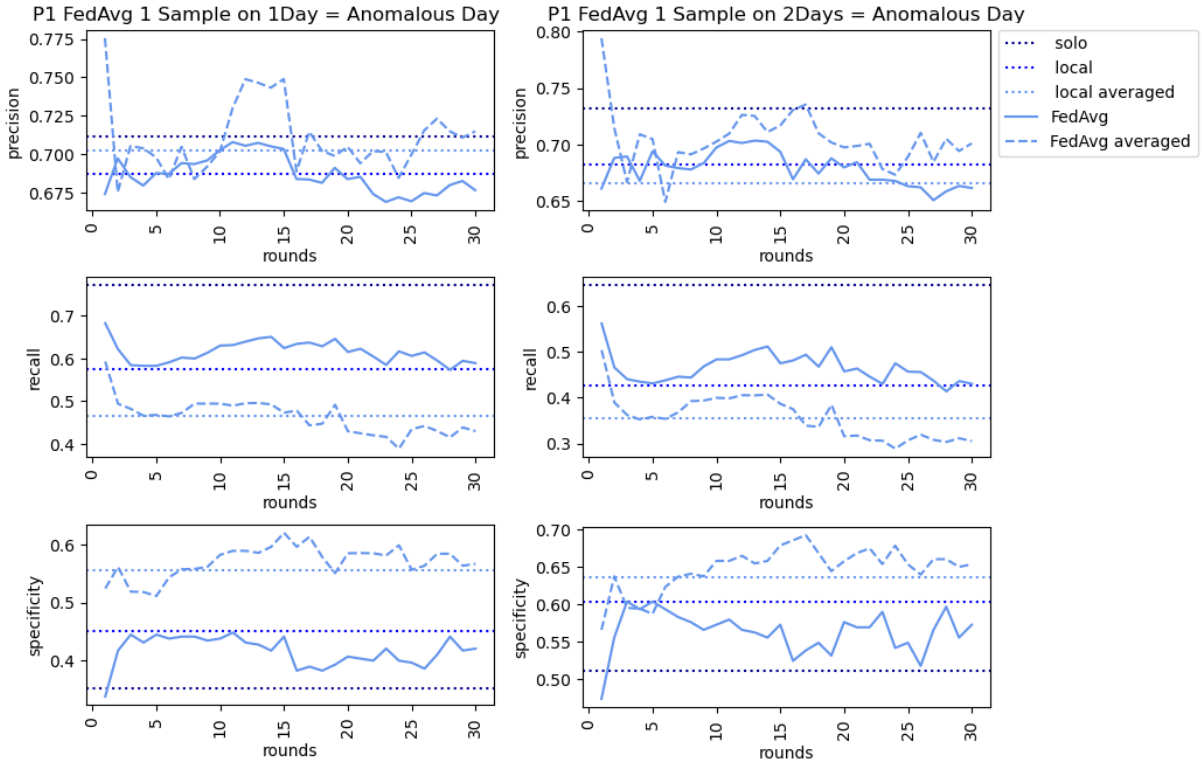


FIGURE 43. Metric scores evolution of models aggregated with FedAvg and trained with P1 when detecting anomalous days.

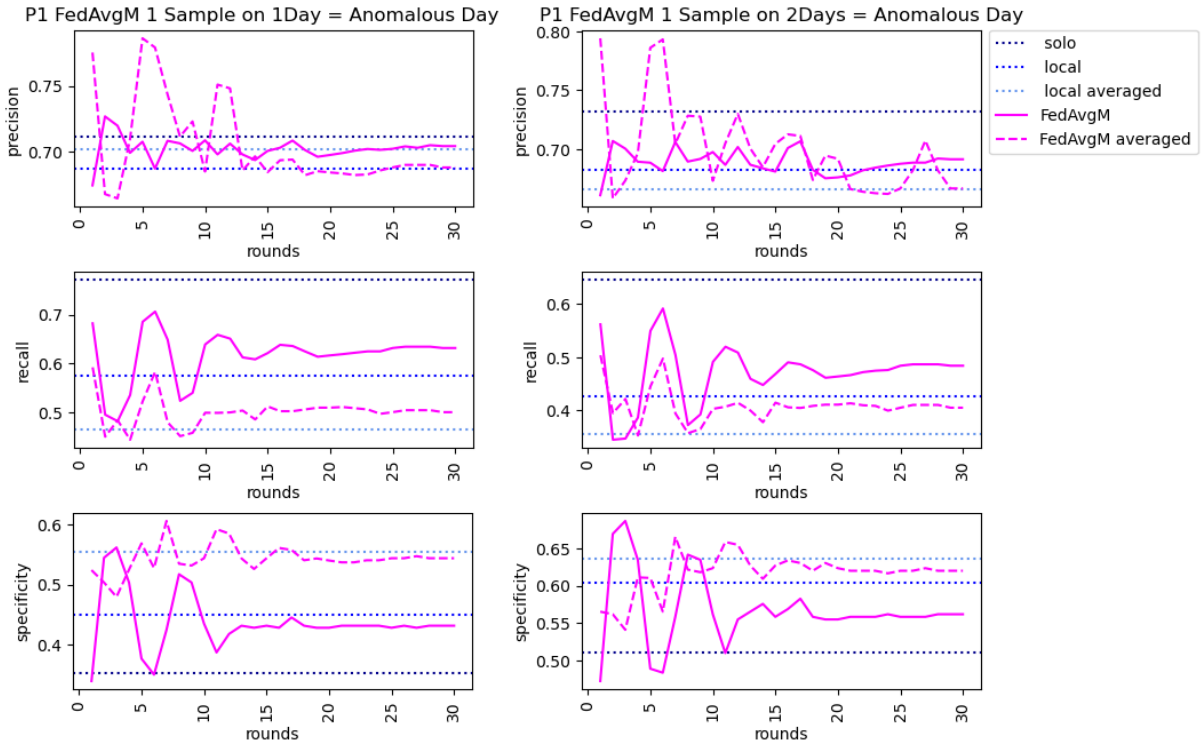


FIGURE 44. Metric scores evolution of models aggregated with FedAvgM and trained with P1 when detecting anomalous days.

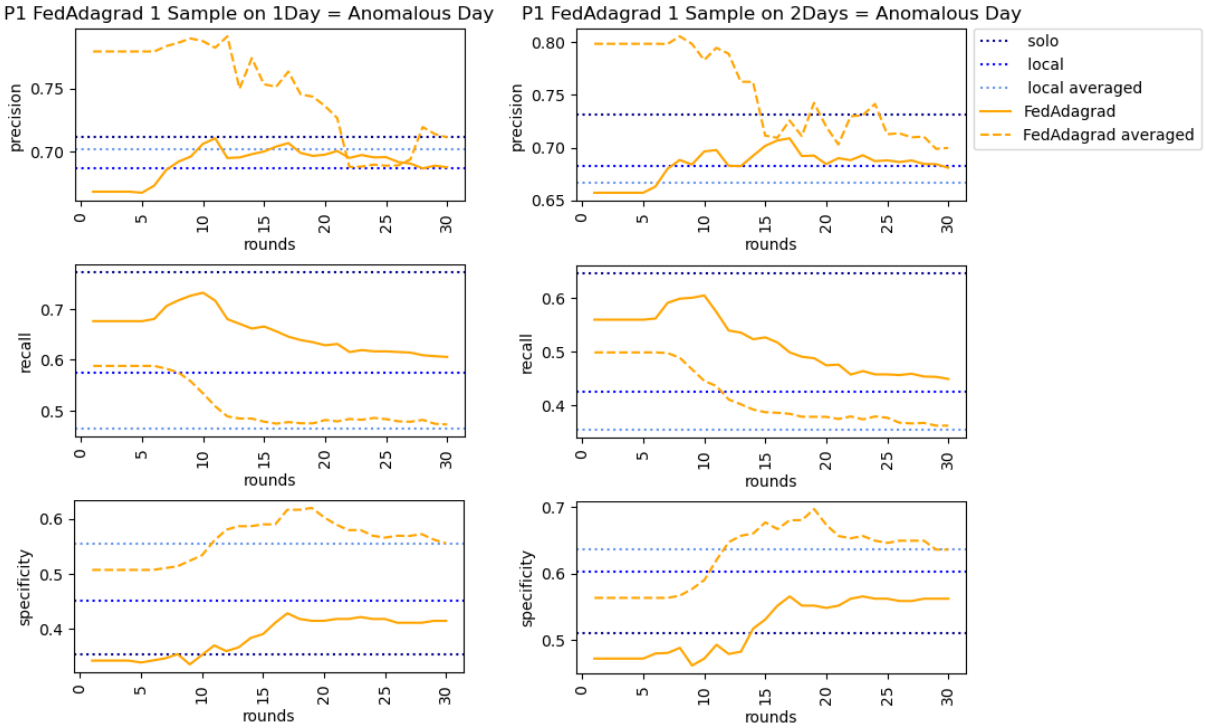


FIGURE 45. Metric scores evolution of models aggregated with FedAdagrad and trained with P1 when detecting anomalous days.

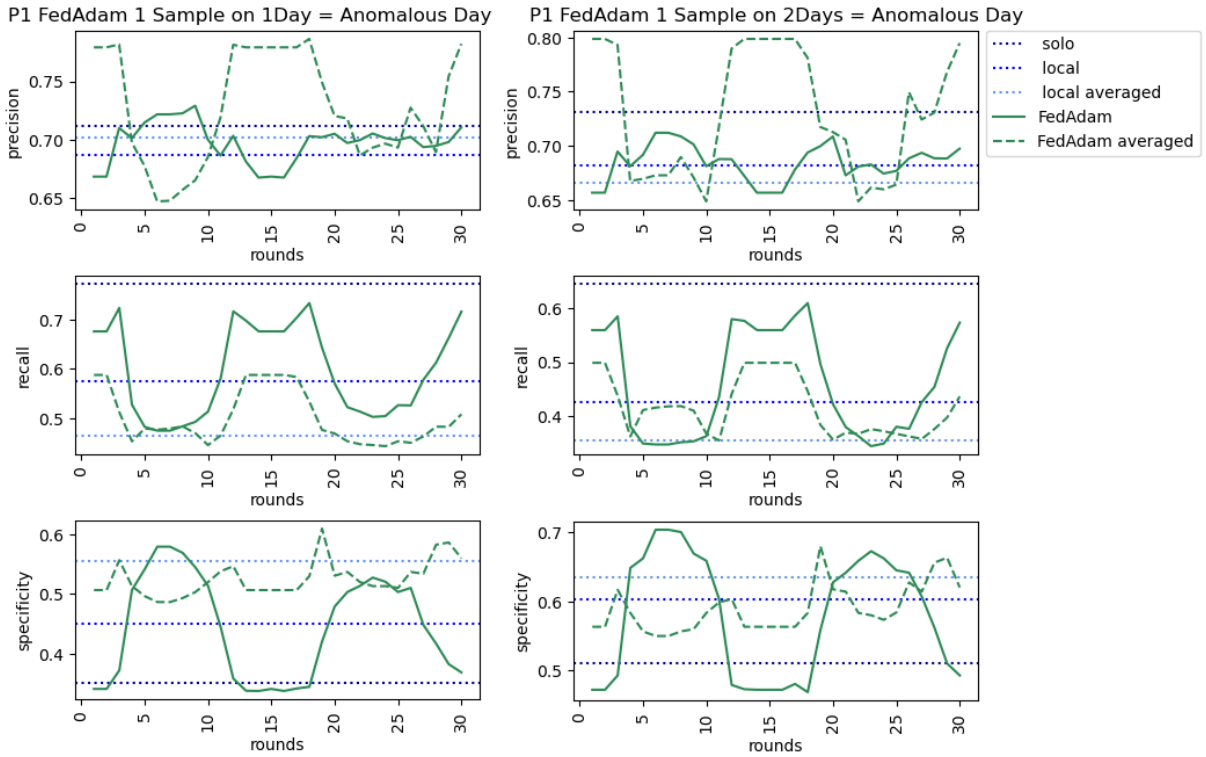


FIGURE 46. Metric scores evolution of models aggregated with FedAdam and trained with P1 when detecting anomalous days.

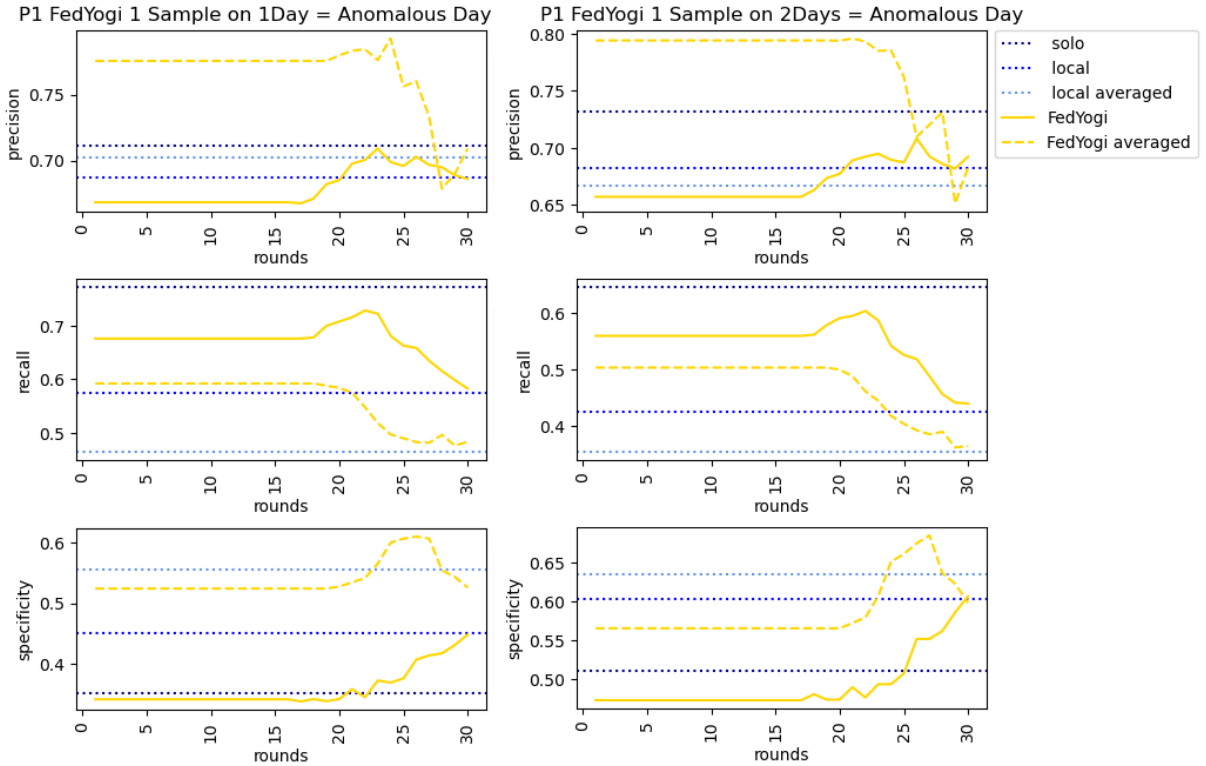


FIGURE 47. Metric scores evolution of models aggregated with FedYogi and trained with P1 when detecting anomalous days.

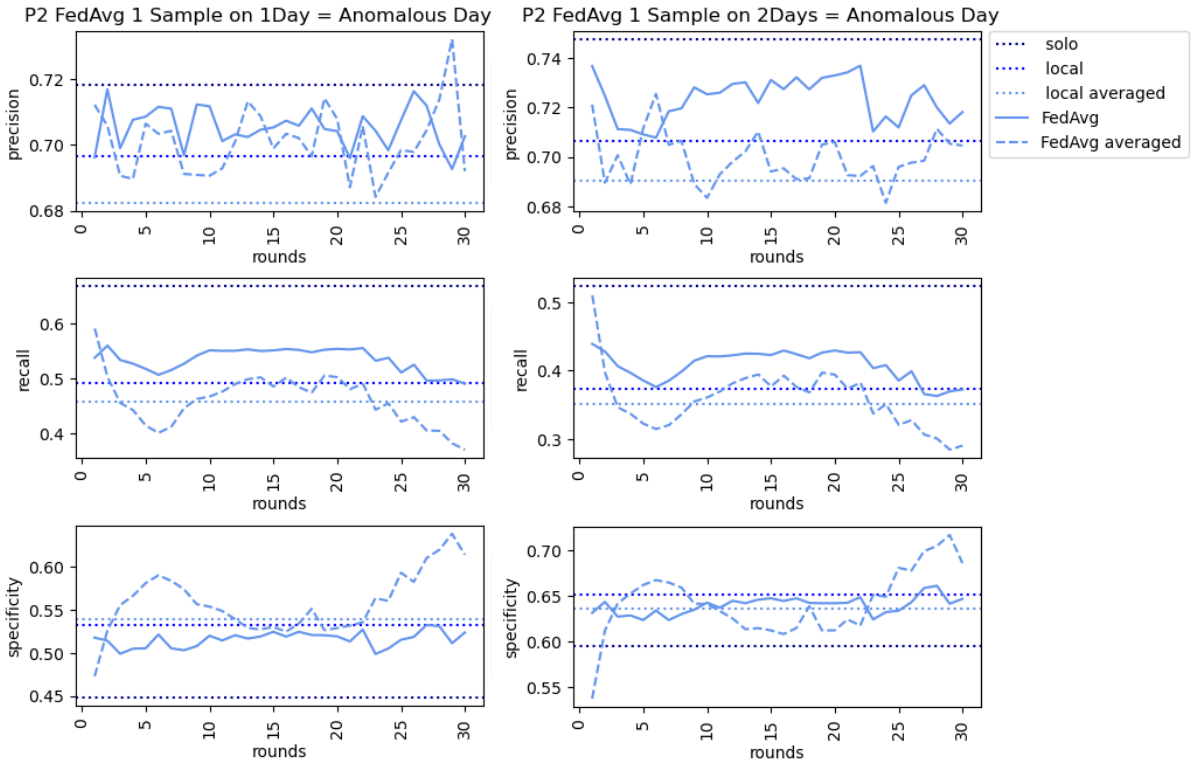


FIGURE 48. Metric scores evolution of models aggregated with FedAvg and trained with P2 when detecting anomalous days.

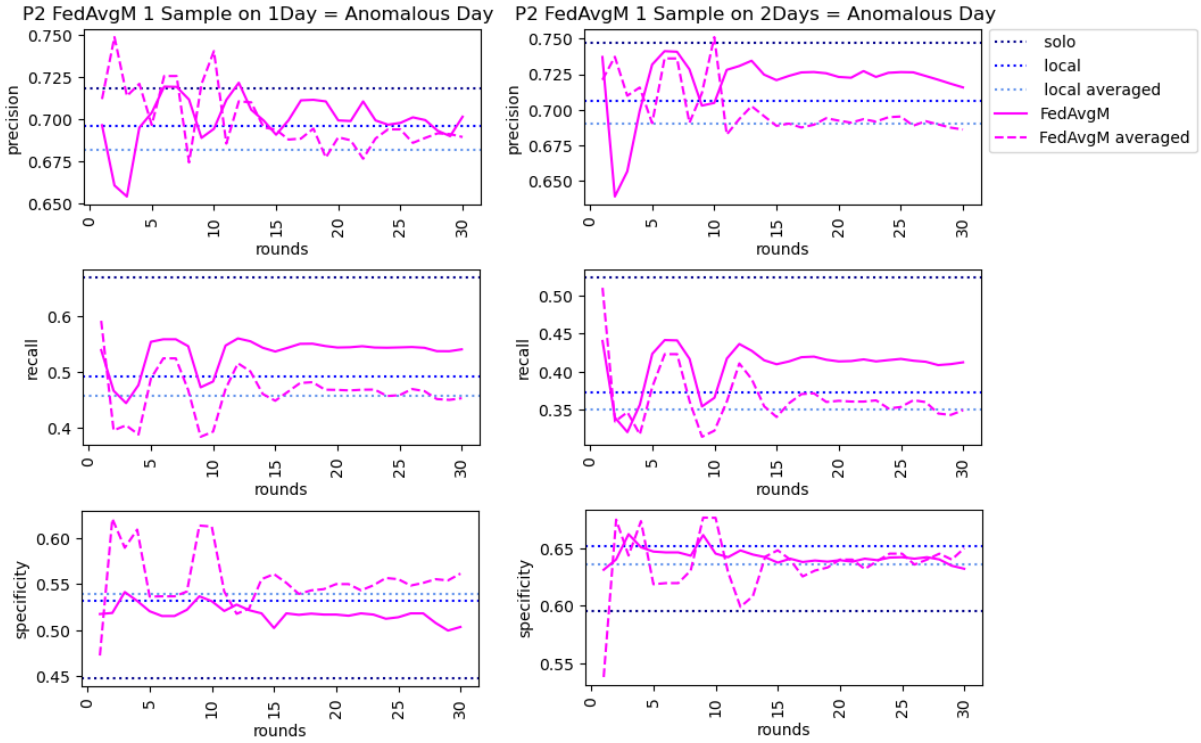


FIGURE 49. Metric scores evolution of models aggregated with FedAvgM and trained with P2 when detecting anomalous days.

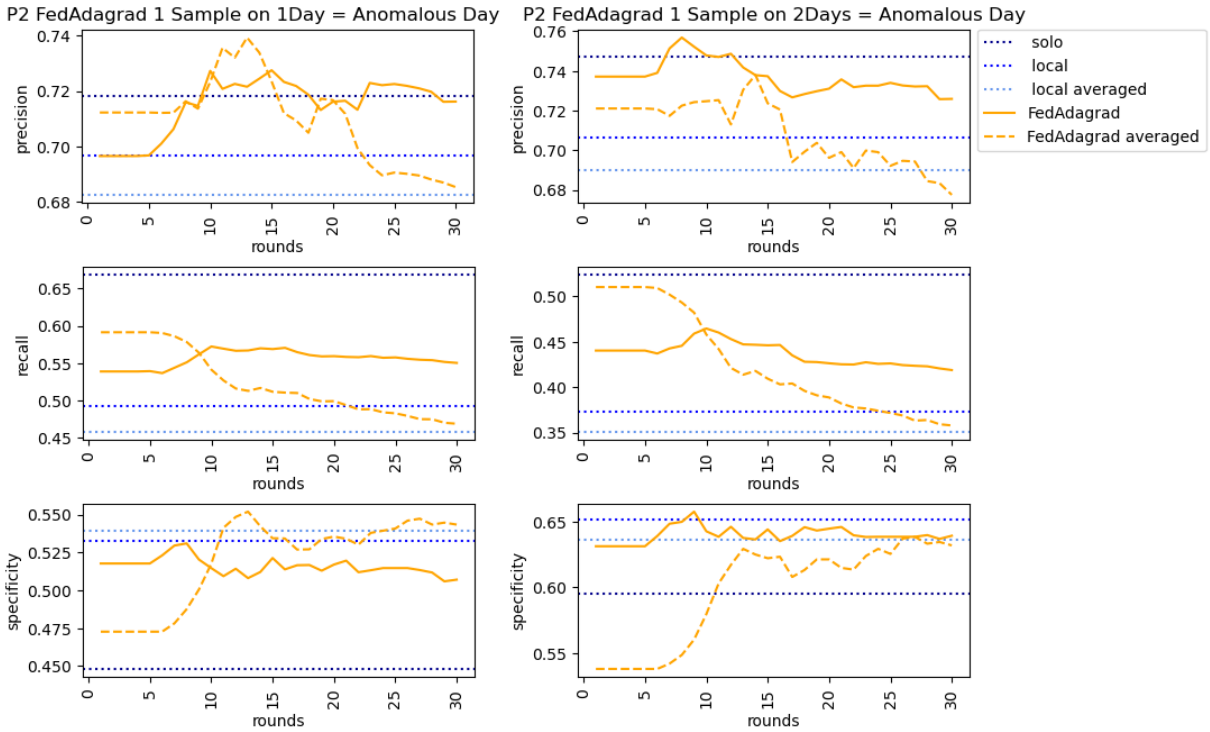


FIGURE 50. Metric scores evolution of models aggregated with FedAdagrad and trained with P2 when detecting anomalous days.

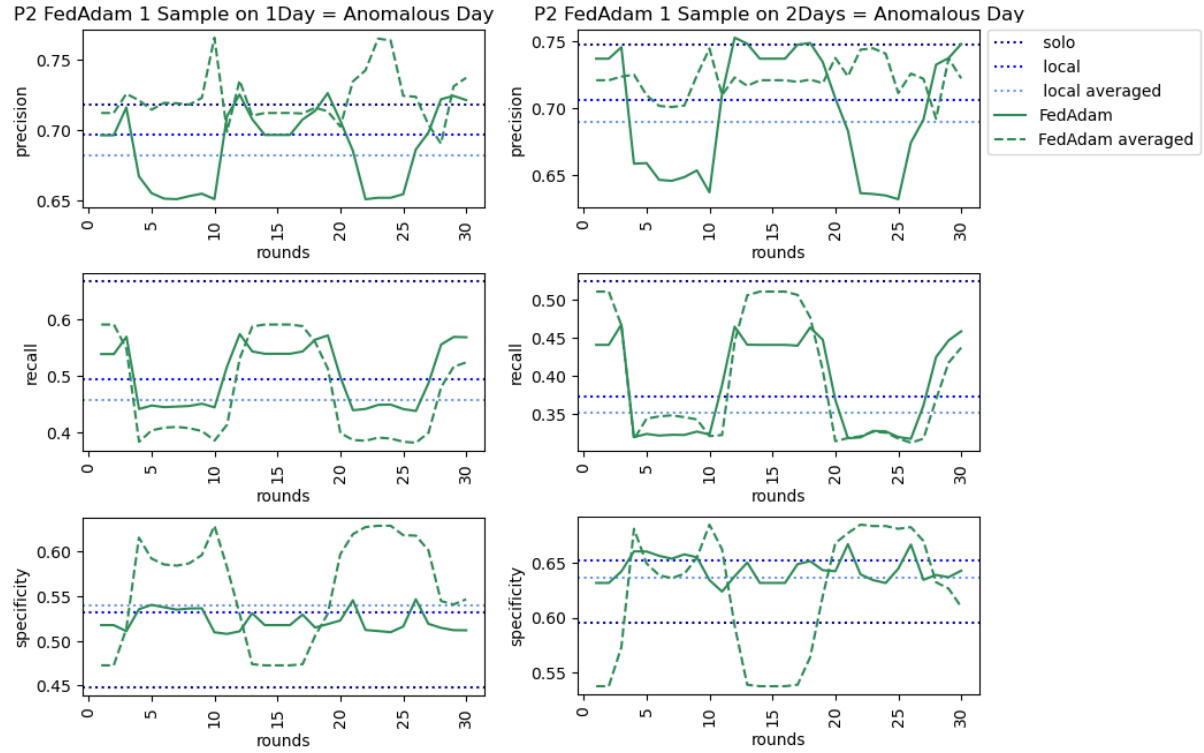


FIGURE 51. Metric scores evolution of models aggregated with FedAdam and trained with P2 when detecting anomalous days.

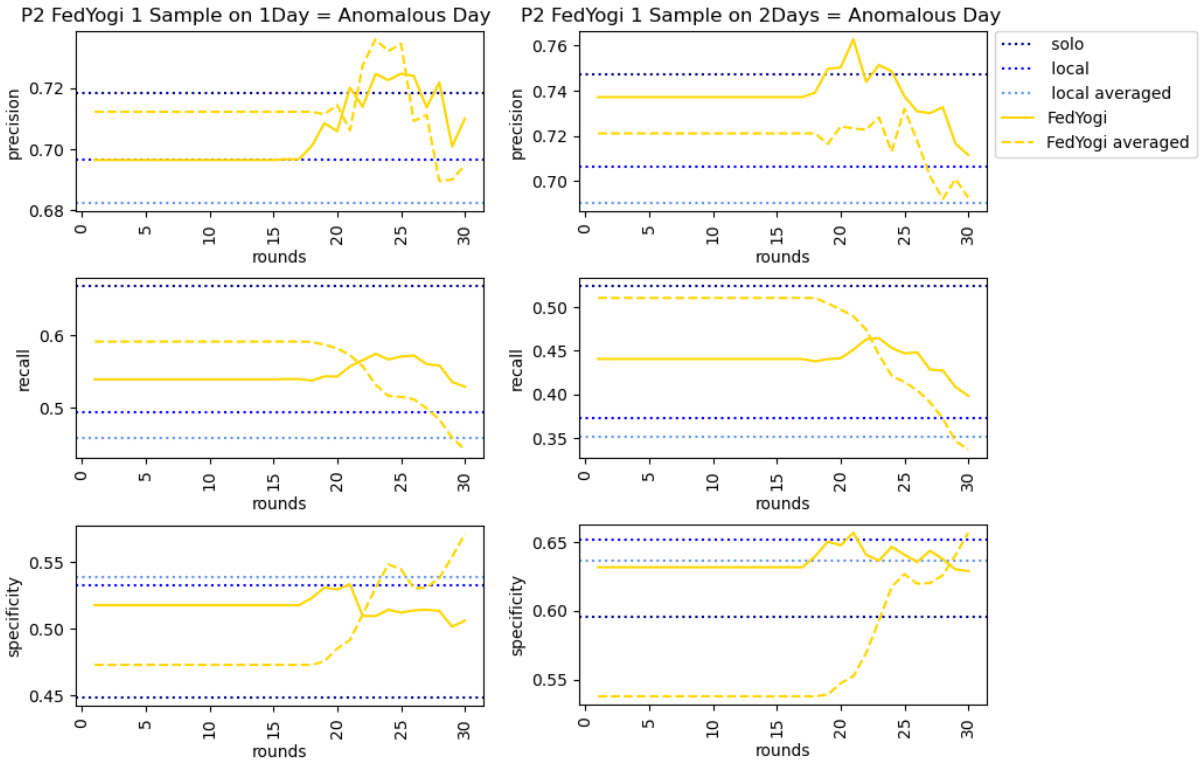


FIGURE 52. Metric scores evolution of models aggregated with FedYogi and trained with P2 when detecting anomalous days.

APPENDIX D

This annex contains result of applying the models trained by multiple individuals of P1 and P2 (local and FL training methods) with averaged thresholds to the data provided by the SMH. The detection is made at 1Day, requiring 10% of the samples to be detected as anomalous for the day to be labeled as such. The results are divided by type of diagnoses.

TABLE D.1. Number assigned to the types of diagnoses present in the Hospital dataset

Number	Type of Diagnoses
0	Intensive Care Medicine Services Main Diagnosis
1	International Classification of Diseases 10th revision
2	Diseases of the Circulatory System
3	COVID-19
4	Neoplasm
5	Diseases of the Genitourinary System
6	Common Diagnoses
7	Diseases of the Respiratory System
8	Diseases of the Nervous System and Sense Organs
9	Trauma and Poisoning
10	Supplementary Classification of Factors that Influence the Health
11	Diseases of the Digestive System
12	Hematological and Hematopoietic Organic Diseases
13	Diseases of the Musculoskeletal system and connective tissue
14	Infectious and Parasitic Diseases
15	Psychiatry

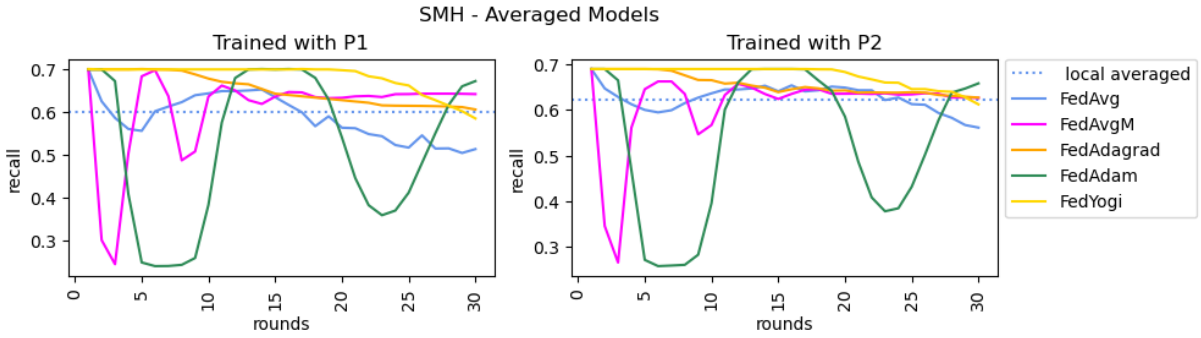


FIGURE 53. Metric evolution of the models trained with P1 and P2 with an averaged threshold applied to hospital patients with known COVID-19 infection.

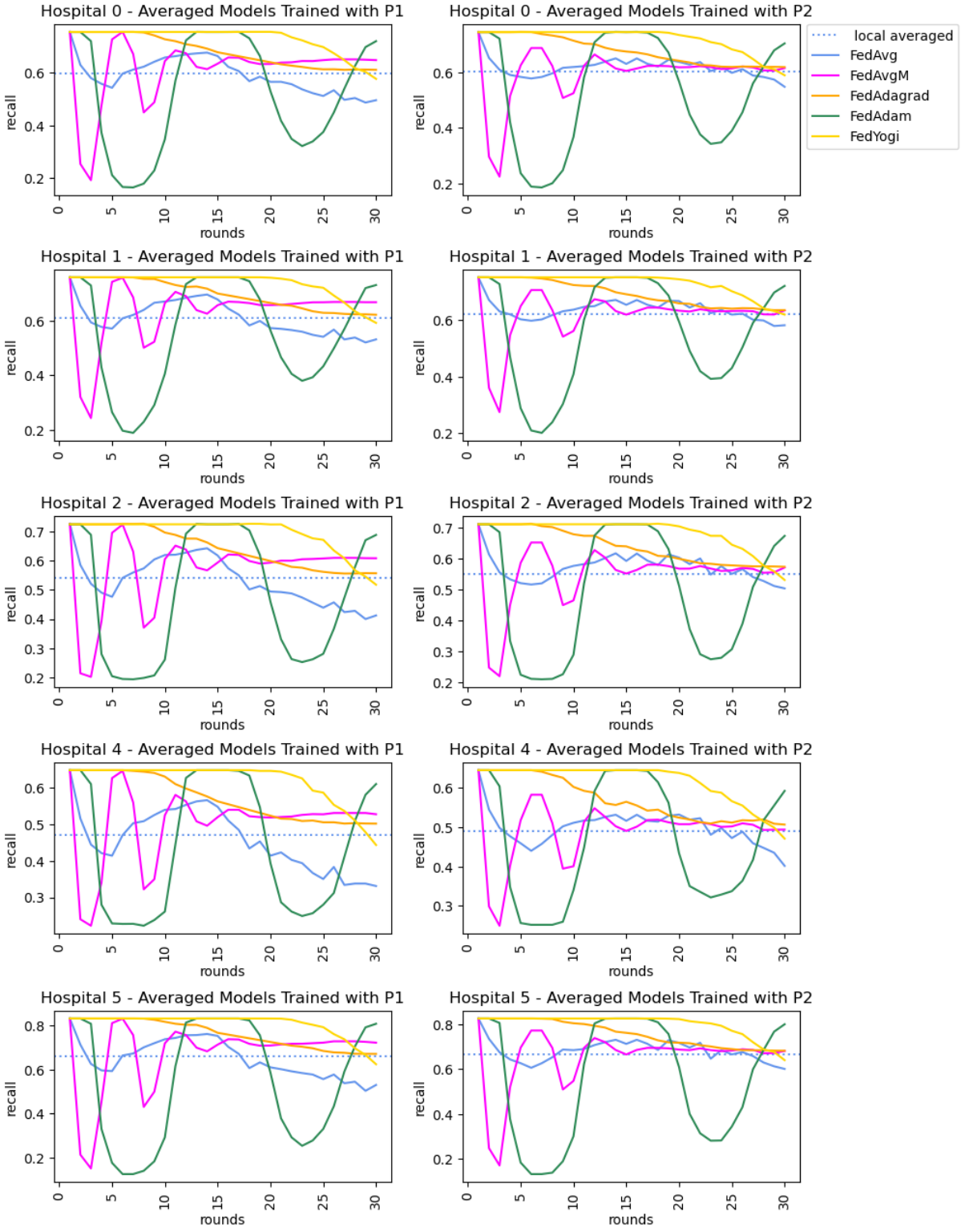


FIGURE 54. Metric evolution of the models trained with P1 and P2 with an averaged threshold applied to hospital patients with diagnostic types 0, 1, 2, 4 and 5.

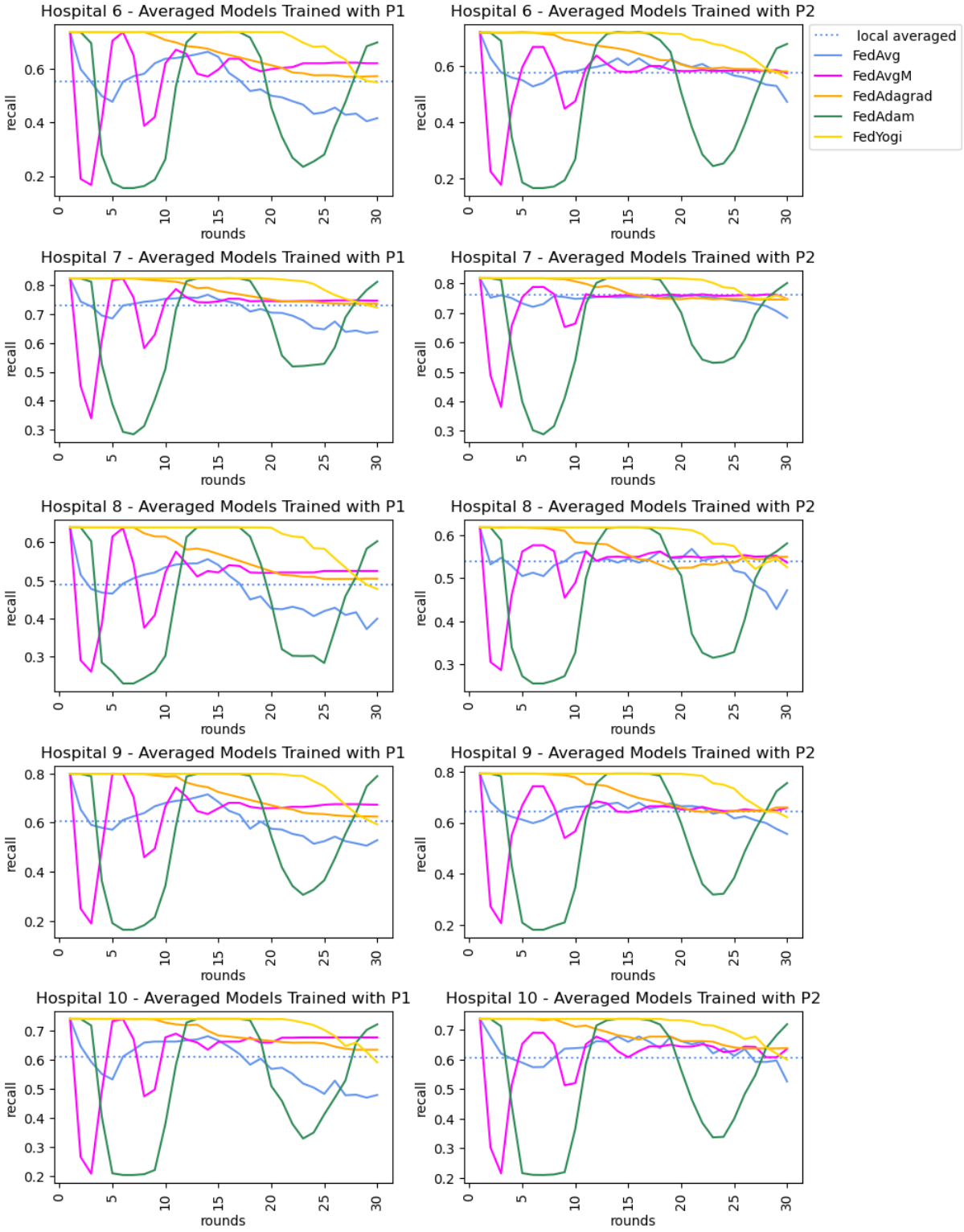


FIGURE 55. Metric evolution of the models trained with P1 and P2 with an averaged threshold applied to hospital patients with diagnostic types 6 through 10.

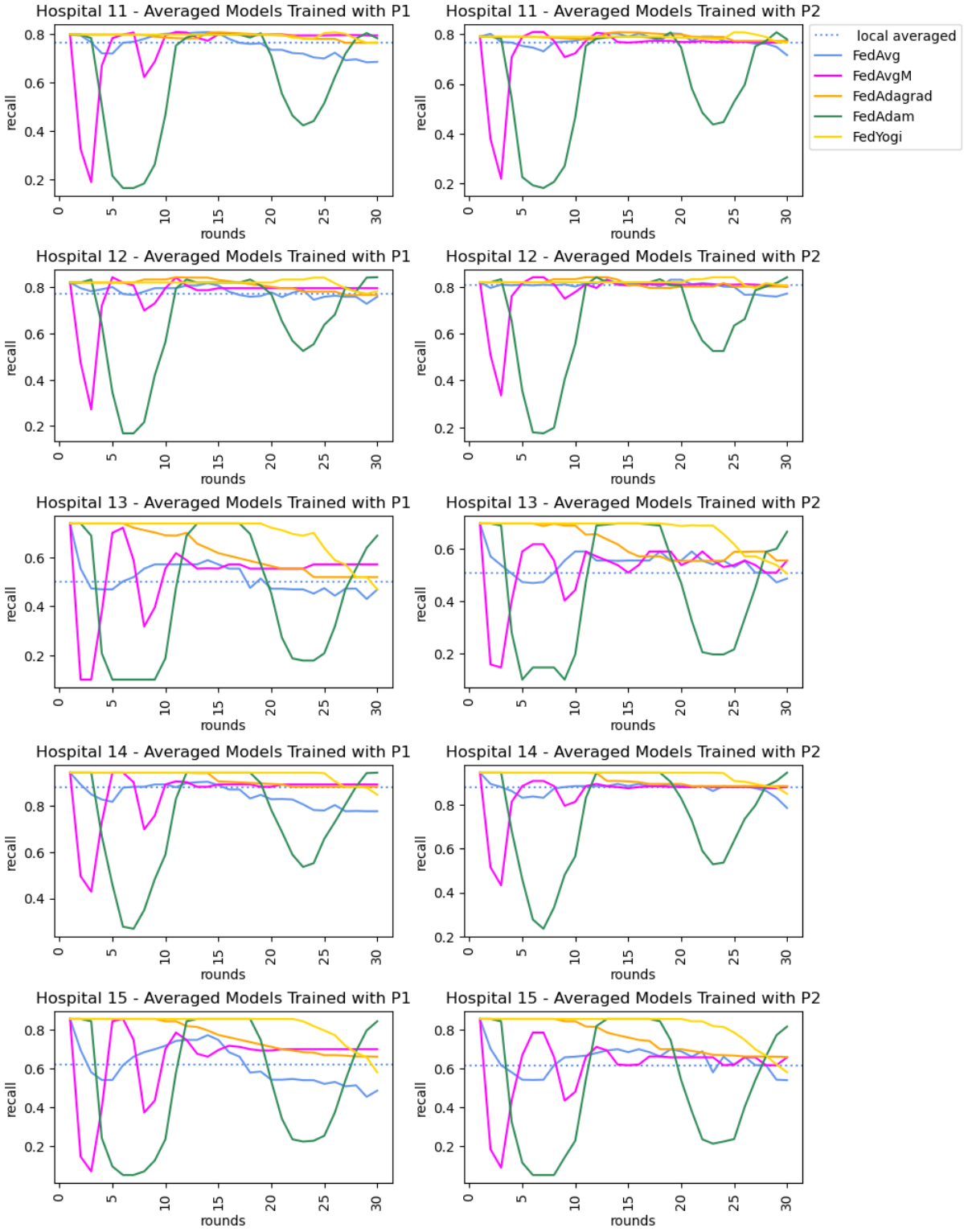


FIGURE 56. Metric evolution of the models trained with P1 and P2 with an averaged threshold applied to hospital patients with diagnostic types 11 through 15.