Pooling critical datasets with Federated Learning

Pooling critical datasets with Federated Learning

1st Yasir Arfat
Computer Science Dept.
University of Torino, Italy
yasir.arfat@unito.it

2nd Gianluca Mittone
Computer Science Dept.
University of Torino, Italy
gianluca.mittone@unito.it

3rd Iacopo Colonnelli
Computer Science Dept.
University of Torino, Italy
iacopo.colonnelli@unito.it

4th Fabrizio D'Ascenzo
Dept. of Medical Sciences
University of Torino, Italy
fabrizio.dascenzo@unito.it

5th Roberto Esposito
Computer Science Dept.
University of Torino, Italy
roberto.esposito@unito.it

6th Marco Aldinucci
Computer Science Dept.
University of Torino, Italy
marco.aldinucci@unito.it

***Abstract*—Federated Learning (FL) is becoming popular in different industrial sectors where data access is critical for security, privacy and the economic value of data itself. Unlike traditional machine learning, where all the data must be globally gathered for analysis, FL makes it possible to extract knowledge from data distributed across different organizations that can be coupled with different Machine Learning paradigms. In this work, we replicate, using Federated Learning, the analysis of a pooled dataset (with *AdaBoost*) that has been used to define the PRAISE score, which is today among the most accurate scores to evaluate the risk of a second acute myocardial infarction. We show that thanks to the extended-OpenFL framework, which implements `AdaBoost.F`, we can train a federated PRAISE model that exhibits comparable accuracy and recall as the centralised model. We achieved F1 and F2 scores which are consistently comparable to the PRAISE score study of a 16-parties federation but within an order of magnitude less time.**

***Index Terms*—Federated Learning, Machine Learning, Cardiology, Healthcare, Performance Analysis, Decentralized machine learning, Distributed machine learning, PRAISE score.**

I. Introduction

RECENT years have been characterized by crucial advances in Artificial Intelligence (AI) systems. The ubiquitous availability of data sets and processing elements supported these advances. The consequent deployment of ML methods throughout many industries has been a welcome innovation, albeit one that generated newfound concerns across multiple dimensions, such as performance, energy efficiency, privacy, criticality, and security. Concerns about data access and movement are particularly felt by industrial sectors such as healthcare, defence, finance, et cetera. This work will mainly focus on healthcare and Federated Learning (FL), a learning paradigm where multiple parties *(clients)* collaborate in solving a learning task using their private data. Importantly, each client's local data is not exchanged or transferred to any participant since, in its most common configuration, clients collaborate by exchanging local models instead of moving the data. The *aggregator* collects the local models and aggregates them to produce a global model. The global model is then sent back to the clients, who use it to update their local models. Then, using their private data, they further update the local model. This process is repeated until the global model converges to a satisfactory solution or a maximum number of rounds is reached.

Thanks to its capability to transform inherently distributed and segregated data into shared knowledge, FL is becoming popular in healthcare; we will survey the main related work in Sec. II. Until recently, the usage of FL was restricted to a specific paradigm of ML, namely Artificial Neural Networks (ANN), often in the Deep Neural Network (DNN) variant, in which models can be easily aggregated using simple associative operators, such as the average of the weights of the DNN[1]. Unfortunately, ANN/DNN is not always the best tool to analyze healthcare data, which is most often given in the form of tabular data. A tabular dataset is a type of data structure that organizes information into a table format, with rows and columns. Each row represents a single data point or record, and each column represents a specific attribute or variable. The data in a tabular dataset is usually numerical or categorical in nature and can be easily analyzed and visualized using tools such as spreadsheets, databases, or data visualization software. Examples of tabular datasets include financial data and demographic data. Several factors may undermine the use of ANN/DNN for (some) healthcare tasks:

- ANN/DNN usually require very large data sets, which are often not easy to collect under the strict privacy laws that regulate health institutions;
- ANN/DNN is hardly explainable, and this is often not acceptable in this field;
- while ANN/DNN performance superiority is undisputed for several tasks, such as image classification, voice recognition, and natural language models, they are not particularly suited to tabular data[2];
- representation learning, which is one of the main driving forces underlying the success of DNN, is much less useful for tabular data since, very often the features have been already engineered with great care and carefully tuned;

Recently Polato et al. [3] proposed several novel Federated Learning algorithms not relying on ANN/DNN; they instead extend distributed AdaBoost techniques to the FL case. Among

Authors' copy (preprint) of Y. Arfat, G. Mittone, I. Colonnelli, F. D'Ascenzo, R. Esposito, and M. Aldinucci. Pooling critical datasets with federated learning. In Proc. of 31st Euromicro Intl. Conference on Parallel Distributed and network-based Processing (PDP), Napoli, Italia, 2023. IEEE.

other algorithms, the work introduces `AdaBoost.F`, a federated variant of the Samme algorithm[1] [4], which can be coupled with a *weak learner* to build a global model on the federated dataset. A weak learner is a learning algorithm, such as decision trees or a logistic regression, which is not required to return very good models (instead, they are required to return models that are better than the random guess). It is worth noting that the very weak assumption on the kind of models built by the clients of the federation allows using the ML model that is best suited for the tabular healthcare data at hand. The mentioned work analyzes the performance of `AdaBoost.F` on several standard datasets in a simulated distributed environment. To our knowledge, the proposed methodology has neither been tested on a real-world dataset in the healthcare domain nor in a distributed execution environment.

As a real-world example, we replicated a notable ML-based risk stratification model in cardiology that recently appeared in the Lancet [5]. The study proposed PRAISE[2], an ML-based score to predict all-cause death, recurrent acute myocardial infarction, and major bleeding after Acute Coronary Syndrome (ACS). Several (non-ANN) ML models have been trained in a cohort of 19826 adult patients with ACS, including patients across several hospitals. The cohort has been split into a training cohort (80%) and an internal validation cohort (20%). The PRAISE score is the best-performing model tested in an external validation cohort of 3444 patients with ACS pooled from a randomized controlled trial. The PRAISE score showed a performance across all possible classification thresholds (Area Under the ROC Curve or AUC) far better than the previously known scores for the same classification task: 0.82 in the internal validation cohort and 0.92 in the external validation cohort for 1-year all-cause death; an AUC of 0.74 in the internal validation cohort and 0.81 in the external validation cohort for 1-year myocardial infarction; and an AUC of 0.70 in the internal validation cohort and 0.86 in the external validation cohort for 1-year major bleeding.

The PRAISE score authors concur in claiming that one of the ingredients making it possible to define a high-quality ML-based score has been gathering one of the most extensive data sets on ACS ever built. The cohort of 19826 adult patients has been manually gathered from different hospitals in different countries (see [5]). Despite being made on anonymized patients, the gathering itself is reported as a very complex process for the privacy and secrecy concerns related to managing critical data from different hospitals in different countries. The current paper attempts to use the dataset of the PRAISE score to be analyzed through federated learning techniques expecting it will provide better results, evidenced by the discussion in Section IV. Following are the contributions made by the authors:

- This work aims to simplify future studies, making it possible to run ML processes on a virtually pooled dataset using a privacy-preserving FL approach.
- Specifically, we aim to demonstrate that FL `AdaBoost.F` can build an almost equally good ML-based model for the PRAISE score while maintaining the data from different sites distributed and mutually secret to the parties running the FL process.
- We also show that the non-ANN models still have a performance edge over more popular ANN/DNN federated models on this specific data set (and, we argue, on many tabular data sets).
- We study the performance of test accuracy of `AdaBoost.F` (accuracy, F1, F2, precision, recall) and scalability in two different execution environments: a cluster of Virtual Machines on an OpenStack cloud and an HPC cluster.

[1] A multi-class variant of AdaBoost.

[2] PRAISE: PRedicting with Artificial Intelligence riSk aftEr acute coronary syndrome. Available as Software-as-a-Service at https://praise.hpc4ai.it

II. Related Works

Federated Learning (FL) has been proposed by McMahan et al. [1] as a way to develop better AI systems without compromising the privacy of final users and the legitimate interests of private companies. Initially deployed by Google for predicting text input on mobile devices, FL has been adopted by many other industries, such as mechanical engineering and health care [6]. Since then, FL has seen a growing interest from the research community, which has identified a few different and interesting settings.

In cross-device FL, the parties can be edge devices (e.g., smart devices and laptops); they can be numerous (order of thousands or even millions). Parties are considered not reliable and with limited computational power. To name a few examples, cross-device FL setting has been adopted in [7] for training a language model for next-word prediction in a virtual keyboard for smartphones; in [8] it has been used to predict emojis (again on a mobile keyboard), or combined with [9] for learning models to be used with IoT devices.

In the cross-silo FL setting, the involved parties are instead organizations; the number of parties is limited, usually in the range [2, 100]. Given the nature of the parties, it can also be assumed that communication and computation are no real bottlenecks. Cross-silo FL settings have been adopted for [10] investigating brain structural relationships across diseases and clinical cohorts; it has also been used for optimizing production through soybean yield prediction [11] or, combined with differential privacy and secure multi-party computation, for attacking important financial tasks such as optimal trade execution, credit origination, or fraud detection [12].

Another important distinction in FL is the way data are distributed between clients. Based on this assumption, it is customary to distinguish between *horizontal* and *vertical* FL. The most used assumption is the *horizontal* data distribution. In this setting, the data is partitioned horizontally, i.e., each client owns a subset of the rows of the total dataset. In contrast, in vertical FL [13], one assumes that the rows are shared between the parties, but each client has a different view of the data (i.e., a different set of features for describing the rows).

Vertical FL is very appealing in cases where the objects in the data sets overlap a lot, but their descriptions have little overlap. It has been applied to several interesting tasks ranging from 5G communications [14], and proposed as a way to improve small and medium enterprises' credit ratings [13].

As mentioned, ANNs and DNNs are rarely used for prediction tasks involving health care represented in tabular form and, in the few cases where they are applied, they do not show performances that are better than traditional ML approaches [15], [16]. Even though they have the potential to perform as well if not better than other approaches, they are hard to tune, and this makes it difficult for their usage by healthcare institutions.

An alternative to ANN/DNN-based FL has been recently proposed in [17]. This alternative, based on the Extreme Gradient Boosting (XBG) ensemble algorithm, is still based on gradient descent, but it allows one to train decision tree models locally. Interestingly, XBG has been used several times in the medical literature on tabular health care datasets [18], [19], [20], [21] showing promising results. While XBG-based techniques address some of the problems outlined above, they require the clients to adopt specific learning algorithms (usually decision trees) and are thus inflexible.

The work presented in [3] introduces three FL adaptations of the AdaBoost ensemble algorithm that work without exchanging gradients between the aggregator and the clients. The approach allows parties to train any kind of model locally (in principle, even different models on each site), thus overcoming the main inflexibility of the XBG approach.

In this work, we will compare `AdaBoost.F` (one of the FL variants introduced in [22]) and FedAvg, comparing their performances in terms of prediction quality and communication and computation times.

In this paper, we will leverage OpenFL [23] as the base framework for FL, but over the past few years, there have been numerous attempts to apply federated learning (FL) technology to healthcare and other industries. These efforts, which range from open-source projects like TensorFlow Federated [24] and PySyft [25] to commercial offerings like IBM® Federated Learning [26] and HP Swarm Learning [27], have aimed to address the needs of researchers and practitioners in a variety of settings. Some of these efforts have focused on simulated environments for research purposes, while others have been designed specifically for production use cases. Other notable FL projects in the healthcare and other industries include FedML [28], FATE [29], Flower [30], Fed-BioMed [31], FederatedScope [32], FLUTE [33], and FLARE [34].

III. Methods

This section describes the methods and the tools used for the experiments we report in Section IV. As mentioned, FL has been traditionally based on some variation of the gradient descent algorithm, whereas the PRAISE score model has been built on AdaBoost derived models, i.e., a non-gradient descent algorithm [3]. We frame both, gradient descent and other approaches in the FL paradigm; then, we describe a recent extension of the OpenFL framework supporting both approaches. Eventually, we argue on the potentiality of the FL as a general privacy-preserving paradigm to extract shared knowledge from datasets from different organizations, i.e. to define a novel methodology to manage data distributed at the edge.

A. FL with gradient descent

FedAvg is an iterative algorithm where a central node (the *aggregator*) collaborates with the other parties (which are termed *collaborators* or *clients*) to develop a shared model. The aggregator starts the process by sharing a randomly initialized neural network with the collaborators. At each round, all collaborators perform one or more training epochs on the given network using their local datasets. The updated model is then shared with the aggregator. The aggregator averages then the contributions using the weighted average:

$$\mathbf{w}^{t+1} = \sum_{c=1}^{C} \frac{n_c}{n} \mathbf{w}_c^{t+1}$$

where $\mathbf{w}$ is a vector containing the weights of the neural networks, t denotes the current round, n_c is the size of the local dataset of client c and $n = \sum_c n_c$. The $\mathbf{w}$ vector is then redistributed to all clients and a new round can start.

One of the difficulties in FL is dealing with clients having examples from different distributions (i.e., they cannot be assumed to be independent and identically distributed (IID)). While we are not addressing this issue in this paper, it is worth mentioning that a few algorithms have been proposed to better cope with these cases. Among them, we can cite FedCurv [35], FedProx‘[36], FedNova [37], and SCAFFOLD [38], which substantially increase the complexity of the federation protocol, but fall short in equally increasing the prediction performance of the resulting models. We refer to the literature for further details and comparative studies [39].

Another difficulty in FL is handling a large number of parties in the federation, which can be in the hundreds or even more. These cases can be handled by including in a round a random selection of clients.

B. FL without gradient descent

In this paper, we experiment with the `AdaBoost.F` algorithm, first introduced in [22]. We report the pseudo-code of the algorithms using the same notation as in the original paper in Algorithms 1 and 2. These notations are: to identify a message that carries x from a client to the aggregator, we will use the function **send**$_x$(aggregator, x) (client-side). The function **broadcast**$_x(x)$, which transmits x to all clients, is used on the aggregator side. There is a **receive**$_x(s)$ in the receiver for every **send**$_x$(aggregator, x) or **broadcast**$_x(x)$

[3]Technically AdaBoost can be explained as an additive algorithm performing a coordinate-wise gradient descent of an exponential loss. The gradient descent is however implicit and does not require gradients to be exchanged nor calculated.

Algorithm 1: `AdaBoost.F` (aggregator)

Input: C: number of clients
T: dimension of the ensamble
K: number of classes
Output: $\text{ens}(\mathbf{x}) \triangleq \text{vote}([h^{t\star}]_{t=1}^{T}, [\alpha^t]_{t=1}^{T}, \mathbf{x})$

for $t \in \{1 \dots T\}$ **do**
  $Z \leftarrow \| [\textbf{receive}_Z(c)]_{c=1}^{C} \|_1$
  $\mathbf{h}^t \leftarrow [\textbf{receive}_h(c)]_{c=1}^{C}$
  $\textbf{broadcast}_{\mathbf{h}}(\mathbf{h}^t)$
  $\mathbf{E}^t \leftarrow \frac{1}{Z} [\textbf{receive}_{\boldsymbol{\epsilon}}(c)]_{c=1}^{C}$ ▷ $C \times C$ errors matrix
  $c^{t\star} \leftarrow \arg\min_c \sum_{c'=1}^{C} \mathbf{E}^t_{cc'}$
  $\epsilon^{t\star} \leftarrow \sum_{c=1}^{C} \mathbf{E}^t_{cc^{t\star}}$
  $\alpha^t \leftarrow \log\left(\frac{1-\epsilon^{t\star}}{\epsilon^{t\star}}\right) + \log(K-1)$
  $\textbf{broadcast}_{\alpha}(\alpha^t)$
  $\textbf{broadcast}_{c}(c^{t\star})$
$\textbf{broadcast}_{\text{stop}}(\text{stop})$

Algorithm 2: `AdaBoost.F` (client)

Input: $\mathcal{A}$: weak learner
$\mathbf{X} \in \mathbb{R}^{n \times m}$: training data
$\mathbf{y} \in \{1, \dots, K\}^n$: training labels

$\mathbf{d} \leftarrow \mathbf{1}$
while *not stop* **do**
  $\textbf{send}_Z(\text{aggregator}, \|\mathbf{d}\|_1)$
  $h \leftarrow \mathcal{A}\left(\mathbf{X}, \mathbf{y}, \frac{\mathbf{d}}{\|\mathbf{d}\|_1}\right)$
  $\textbf{send}_h(\text{aggregator}, h)$
  $\mathbf{h} \leftarrow \textbf{receive}_{\mathbf{h}}(\text{aggregator})$
  $\boldsymbol{\epsilon} \leftarrow \left[\mathbf{d}^\top [\![\mathbf{y} \neq h_c(\mathbf{X})]\!]\right]_{c=1}^{|\mathbf{h}|}$
  $\textbf{send}_{\boldsymbol{\epsilon}}(\text{aggregator}, \boldsymbol{\epsilon})$
  $\alpha^\star \leftarrow \textbf{receive}_{\alpha}(\text{aggregator})$
  $c^\star \leftarrow \textbf{receive}_{c}(\text{aggregator})$
  $\mathbf{d} \leftarrow [d_i \exp(-\alpha^\star [\![h_{c^\star}(x_i) \neq y_i]\!])]_{i=1}^{n}$

from a sender, where s denotes the sender. Other notations represent weighted error (ϵ^t), weight of the weak hypothesis (α^t), distribution $\mathbf{d}$ and $\mathbf{h}^{t\star}$ represents "global" weak classifier.

The training phase of `AdaBoost.F` is similar to the one of AdaBoost, but it happens in a distributed manner. At each iteration, a new weak hypothesis is learned from each client and sent to the aggregator. The aggregator collects the weak hypotheses and broadcasts them all to all clients. The clients evaluate the received hypotheses on the local dataset and send the weighted errors ϵ to the aggregator, which is then able to aggregate these values into a matrix $\mathbf{E}^t$. Values in $\mathbf{E}^t$ are then used to find the best hypothesis for the current round c^{t*} and to compute the current α^t term. By propagating these pieces of information to the clients, they are then able to update their local copy of the ensemble and to update the local examples' weights $\mathbf{d}$. Overall, the algorithm has strong resemblances with the original AdaBoost algorithm; one interesting difference is

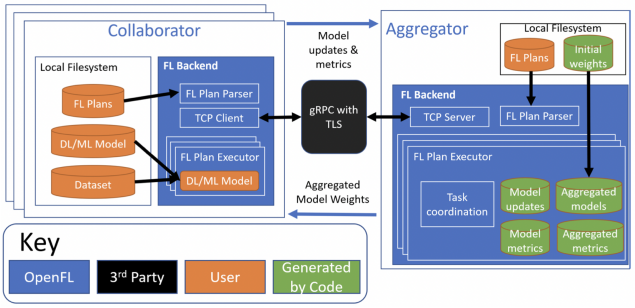


Fig. 1: The Intel® OpenFL software stack.

the fact that $\mathbf{d}$ is kept un-normalized in the client. This is important to make it possible to compute a global normalization factor in the aggregator.

C. The Intel® OpenFL framework

OpenFL is an open-source tool for cross-silo FL based on Python 3, designed to be flexible, extensible, community-driven, and easy to learn for data scientists. The potential of OpenFL has already been showcased in [40], where the tool was used to build the world's largest federation to date.

Fig. 1 shows an overview of the OpenFL architecture. Note that the vanilla OpenFL only supports neural networks, as most of the FL frameworks in the market. Our re-engineering effort to support `AdaBoost.F` targeted specifically the software component coloured in blue and marginally the orange ones. This way, the user interaction with the framework is only marginally affected. A detailed description of the re-engineering effort, including design choices and implementation details, will be published elsewhere.

There are two key actors in a federation. Each *collaborator* accesses its own data to train a replica of the ML model. A central *aggregator* collects and merges the updates produced by each collaborator. While collaborators live at the edges of the federation, the aggregator usually runs on a data centre or in the cloud. This is necessary due to its high network usage and its reachability and reliability requirements.

D. Federated pooling and FL-as-a-Service

Edge computing is a distributed computing paradigm that brings computation and data storage closer to the data sources [41]. Edge computing developed with the expectation of keeping part of the computation close to the data sources reducing latency and bandwidth requirements in processing inherently distributed data streams (as opposite to BigData batch processing typical of cloud computing). FL addresses the same conceptual architecture, where data is mainly processed near the data sources, where the two presented FL settings (cross-device and cross-site) define their features in terms of scalability, computing power, energy efficiency, security and reliability. In this work, we mainly focus on cross-site scenarios, which fit a consortium of trusted data owners (e.g., hospitals) connected with a secure and reliable network that do not wish to share their private data. In this respect, FL might be

considered an example of edge computing that extends original motivations beyond execution performance.

The PRAISE score offers a prediction performance far better than similar scores [5]. Notice that the pooled dataset used to train the PRAISE score model is orders of magnitude larger than those driving similar studies, typically gathered within a single organization. A basic assumption underneath all FL algorithms is that they can *approximate* a traditional model trained on a pooled dataset, making it possible to train models on an increasingly larger dataset from different organizations.

There are two assumptions subject to empirical verification. First, a consortium of organizations, each of them owning private data, can train a model virtually pooling all datasets via FL and this model *exhibits a comparable prediction performance* of the best-known model that can be built from the union of all datasets stored in a single data lake. Second, the kind of models that can be trained is not limited to ANN but encompasses explainable ML models, such as decision trees. Formally proving these assumptions goes far beyond the scope of the present work. Nevertheless, the empirical validation makes it possible to envision entirely novel data operation for edge systems enabling data analysis: *federated pooling*.

Generally speaking, the two basic operations on data are *read* and *write*. Concurrent systems (e.g., parallel, distributed) require an additional *atomic-read-write* primary operation to enforce data integrity on shared data, which is needed to support data sharing primitives among concurrent activities (e.g., transactions). A further step in distributed systems is reaching a *consensus* among parties, i.e., agreeing on some data value needed during computation. A distributed consensus protocol makes it possible to implement a distributed ledger to distinguish a digital object from its copy.

Federated pooling can make a further step in distributed data management since it can virtually pool many datasets for a specific data analytic task. Federated pooling does not subsume data operations (read, write, compare-and-swap); it is stateless because it does not permanently affect data. For this, it can be re-executed many times and with different organizations, thus finely controlled and billed. We envision federated pooling as the basic API of a new kind of service for edge computing: FL-as-a-Service (FLaaS).

IV. Experiments

This section compares boosting algorithms and neural network models on a real-world binary classification problem, assessing their prediction accuracy and training time performance. We also compare federated algorithms against their non-federated counterpart, which serves as a baseline for the prediction performance.

In particular, we trained a simple Feed-forward Neural Network (FNN) model and an AdaBoost ensemble on the PRAISE dataset [5], containing 19826 adult patients suffering from Acute Coronary Syndrome (ACS) with one year of follow-up. This dataset is the union of two previously existing registries, BleeMACS (NCT02466854) and RENAMI [42] and contains 25 features categorical and ordinal, and three categorical outcomes: all-cause death, recurrent acute myocardial infarction, and major bleeding one year after discharge. We only trained the models for this study to predict the all-cause death outcome.

As a first step, we pre-processed the dataset to mitigate the high imbalance in the outcomes (using SMOTE [43]) and to handle missing data in the features (using the median value along each column). Then we split the dataset to use 80% of the rows for training, leaving the remaining ones for validation. The FNN model is a two-layer perceptron with 35 inputs, 35 hidden units, a single output and a binary cross-entropy loss. We trained it for 100 rounds of one epoch each, using Adam [44] ($lr = 10^{-3}$, $\beta_1 = 0.9$, $\beta_2 = 0.999$, $\epsilon = 10^{-7}$) as the optimizer and FedAvg [1] as the aggregation strategy. The federated AdaBoost ensemble is built by running 100 rounds of the `AdaBoost.F` algorithm [3], using a decision tree with at most 10 leaves as the weak learner.

All FL runs have been orchestrated using the Intel® OpenFL framework [23] with a single aggregator and up to 16 collaborators. We tested two different configurations to asses both the strong and weak scaling of the training processes. To test the strong scaling, we divided the entire dataset into n i.i.d. subsets without replacement and assigned a subset to each of the n collaborators involved in the federation. This configuration keeps the same amount of total rows for each experimental configuration. Conversely, to test the weak scaling, we sampled 16 subsets of the complete datasets and assigned one of them to each collaborator. In this setting, the size of the problem increases linearly with the number of collaborators involved in the federation.

We took both learning performances and training times for each setting. Learning performances have been measured on a virtualized environment on top of the OpenStack-based HPC4AI cloud infrastructure [45], with the aggregator running into a VM with 4 cores and 8GB RAM and up to 16 collaborators hosted in VMs with 8 cores and 8GB RAM each. Conversely, times have been measured on the C3S HPC facility [46], allocating an entire bare metal node with 2 Intel® Xeon E5-2697 sockets (18 cores, 2.30GHz) and 128GB RAM to each component of the OpenFL deployment.

A. Results

We analyze the performance of the algorithms in terms of prediction quality and computational/communication times.

To assess the prediction performances, we used the five metrics reported in Table III: accuracy, F1 score, F2 score, precision, and recall. Despite being the most known and used metric, accuracy is a very bad indicator of a model's performance when the dataset is not balanced, as in this case. The accuracy of a constant model always predicting *false* would score 97% on the PRAISE dataset. F_1 and F_2 scores are better candidates in these cases since, by averaging precision and recall, they require the classifier to recover most of the positive cases (to score a high recall) and to be correct on them (to score a high precision). The main difference between

TABLE I: Prediction performance of the FNN model. Values reported are the average ± stdev of 5 runs. The first run in the strong scaling setting is equivalent to the non-federated case.

Clients	Accuracy	F1 Score	F2 Score	Precision	Recall
Strong scaling setting					
1	.39 ± .47	.14 ± .08	.22 ± .04	.17 ± .09	.72 ± .39
2	.56 ± .47	.19 ± .09	.26 ± .06	.15 ± .09	.61 ± .36
4	.88 ± .01	.23 ± .01	.30 ± .01	.17 ± .01	.39 ± .02
8	.72 ± .38	.20 ± .06	.27 ± .04	.16 ± .06	.48 ± .29
16	.90 ± .01	.24 ± .01	.29 ± .01	.12 ± .01	.35 ± .02
Weak scaling setting					
1	.56 ± .47	.16 ± .07	.22 ± .03	.12 ± .07	.56 ± .40
2	.69 ± .37	.17 ± .05	.25 ± .04	.12 ± .06	.49 ± .30
4	.72 ± .38	.20 ± .07	.27 ± .05	.15 ± .06	.49 ± .29
8	.90 ± .04	.18 ± .10	.24 ± .13	.13 ± .08	.30 ± .17
16	.55 ± .46	.17 ± .08	.26 ± .06	.11 ± .06	.63 ± .34

TABLE II: Prediction performance of `AdaBoost.F`. Values reported are the average ± stdev of 5 runs. The first run in the strong scaling setting is equivalent to the non-federated case.

Clients	Accuracy	F1 Score	F2 Score	Precision	Recall
Strong scaling setting					
1	.95 ± .00	.19 ± .07	.15 ± .06	.35 ± .10	.13 ± .05
2	.95 ± .00	.23 ± .03	.19 ± .03	.36 ± .04	.17 ± .03
4	.94 ± .00	.19 ± .02	.16 ± .02	.26 ± .04	.15 ± .02
8	.94 ± .00	.20 ± .04	.17 ± .03	.28 ± .06	.16 ± .03
16	.94 ± .00	.19 ± .03	.17 ± .03	.25 ± .04	.16 ± .03
Weak scaling setting					
1	.95 ± .00	.09 ± .02	.06 ± .01	.33 ± .05	.05 ± .01
2	.95 ± .00	.10 ± .02	.07 ± .01	.45 ± .05	.05 ± .01
4	.95 ± .00	.15 ± .04	.12 ± .04	.32 ± .06	.10 ± .10
8	.95 ± .00	.17 ± .02	.14 ± .01	.28 ± .04	.13 ± .01
16	.94 ± .00	.20 ± .03	.18 ± .02	.27 ± .04	.16 ± .02

these two metrics is about the relative importance of precision and recall: F_1 gives them the same importance, while F_2 is better for cases where precision is to be considered twice as important as recall.

Table I reports the results obtained with the FNN model, while Table II refers to the `AdaBoost.F` ensemble.

We start by noticing that, from the point of view of accuracy, `AdaBoost.F` dominates by reaching 95% accuracy in most experiments. As mentioned, however, accuracy is a bad metric in this particular case, and the high accuracy values suggest that the ensemble model probably categorises most of the examples as negatives. The rest of the metrics confirm this intuition: recall values are much lower in the case of `AdaBoost.F` than in the case of the FNN model. Accuracies, in the case of the FNN model, are much more erratic, showing both a higher variance as the number of collaborators grows and a higher variance in the 5 experiment repetitions. However, the important metrics (F_1 and F_2) show much better performances of the FNN models w.r.t. the ensemble model. Indeed, as far as we can tell, results reported in table I are even better than those shown in the state-of-the-art technique [5]

TABLE III: Summary of statistics used to evaluate prediction performances.

Measure	Definition	Description
Accuracy	$\frac{\text{TP+TN}}{\text{TOT}}$	Fraction of the examples correctly classified.
Precision	$\frac{\text{TP}}{\text{TP+FP}}$	Fraction of examples predicted as positive that are actually positive.
Recall	$\frac{\text{TP}}{\text{POS}}$	Fraction of positive examples that are correctly predicted as positive.
F_1 score	$2\frac{\text{precision}\cdot\text{recall}}{\text{precision+recall}}$	Harmonic mean of precision and recall.
F_2 score	$5\frac{\text{precision}\cdot\text{recall}}{4\text{precision+recall}}$	Weighted harmonic mean of precision and recall (giving precision twice the importance of recall).

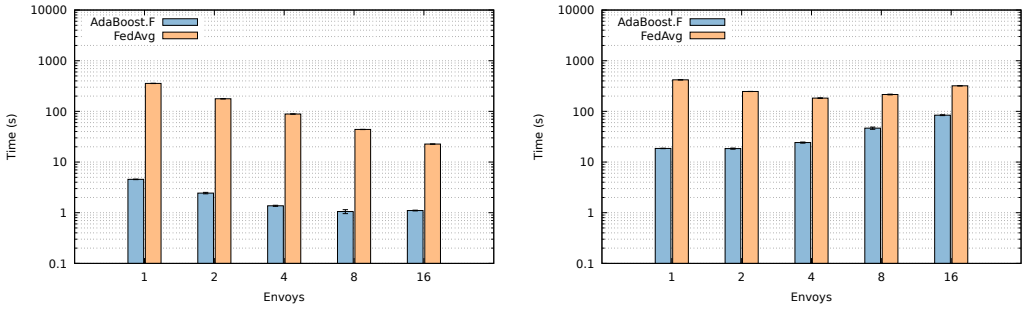
(the paper that introduced the dataset and the PRAISE score).

While these are indeed good news, we refrain from calling these models better because the models in [5] have been thoroughly evaluated under a multitude of aspects, not just F_1 and F_2. Nonetheless, we plan to investigate these models in the future further. Another interesting facet of the reported FNN results is that the F metrics do not follow a growing pattern as the number of collaborators grows. They show an inverted v shape, which is difficult to explain. In fact, at least in the weak scaling setting, we would have expected the performances to continue to grow since adding more collaborators amount, in this latter case, to increasing the dataset size. A possible explanation might be that the FL procedure finds it difficult to leverage all the available data when the number of involved parties grows (to the point of making adding new parties to the federation no longer worthwhile).

In summary, from the point of view of prediction quality, quite unexpectedly, the FNN model appears to be better than the ensemble of trees acquired by `AdaBoost.F`. Whether they are overall better models is, however, to be decided since, for the specific case of health institutions, other factors (above all, the interpretability of the results) might prevail in evaluating the possible solutions.

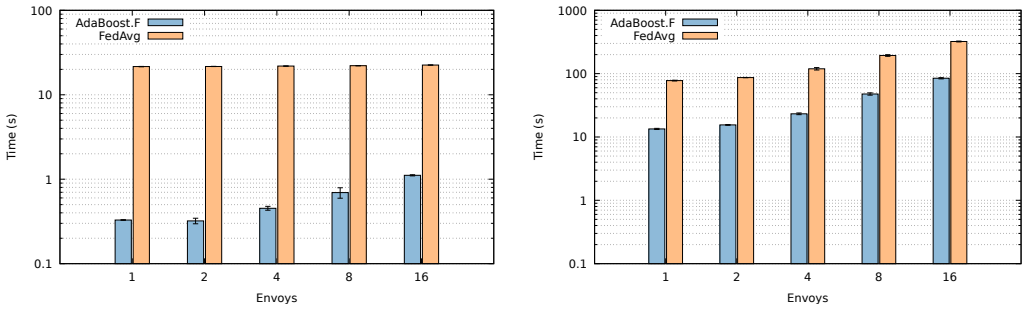
As stated earlier, this dataset is highly imbalanced, so it is tough to get good F_1 and F_2. However, it was an unexpected observation that the best F_1 scores are for models acquired in the federated case, even in the strong scaling setting. Contrary to our expectations, the classical case (corresponding to the first line of the strong scaling experiments) was not the best. From the point of view of training the model, this is the easiest configuration, where all data are located in a single point, and there are no privacy issues. This is a further point to be further investigated, but a possible explanation could be that the FL process acts as a regularization factor preventing the model from overfitting the training set.

Fig. 2 and 3 report the execution times of 100 training rounds for the FNN model and the `AdaBoost.F` ensemble in the strong and weak scaling settings, respectively. In the strong scaling setting, the size of the pooled dataset is constant; data is equally partitioned among envoys, whereas in the weak scaling setting, the size of data associated with each envoy is constant; the more envoy, the more data. The baseline is the



(a) Computation time (for a single envoy) (b) Total time

Fig. 2: `AdaBoost.F` and FedAvg training time for 100 rounds executed on the C3S machines in the strong scaling setting.



(a) Computation time (for a single envoy) (b) Total time

Fig. 3: `AdaBoost.F` and FedAvg training time for 100 rounds executed on the C3S machines in the weak scaling setting.

strong scaling setting with a single envoy, which is identical to a non-federated process. The most relevant thing to notice is that training an FNN with FedAvg is between 5 and 30 times longer than training an ensemble of decision trees with `AdaBoost.F` in both settings. However, this gap is much more evident when considering only the actual computation time, as the overhead introduced by serialization and communication is much more evident with `AdaBoost.F`. Plus, the communication time appears to be the actual bottleneck in the overall execution, as the total training time increases with the number of federation members. This finding suggests that the benefit of using `AdaBoost.F` will be much more evident with a more efficient FL framework, whose development is already in the research plan.

Analyzing the two algorithms' strong and weak scaling behaviour, it is worth noting that the FNN model follows a common trend in both settings. Indeed, the total time to solution decreases with more collaborators in the strong scaling setting, while it remains almost constant in the weak scaling one. Conversely, with `AdaBoost.F` the time to solution decreases only up to 8 collaborators in the strong scaling setting, and it linearly grows up in the weak scaling setting. This behaviour is justified by the fact that the second phase of the algorithm requires each collaborator to evaluate n decision trees on the local data to determine the best one, where n is the number of collaborators in the federation. A more efficient version of the algorithm, aiming to reduce the computation overhead to select the best model at each round, is currently in plans.

V. Conclusion

Federated learning is an essential tool to allow building machine learning models across parties that need to maintain the privacy of their local datasets. The most used FL algorithms are based on some variation of the gradient descent algorithm and are used to train neural networks. However, neural networks are not always desirable models and are often harder to train when tabular data is involved. In this work, we have investigated how the FedAvg algorithm and `AdaBoost.F` compare for the task of predicting all-cause mortality. This exploratory study constitutes a proof-of-concept of a more general paradigm, the *federated pooling*, which allows to compose complex "virtual" pooled datasets while leaving the actual data undisclosed at the edge.

We compared the performance of these two methods from the point of view of prediction performances as well as from the point of view of computation and communication time. While we expected `AdaBoost.F` to be a better solution for this specific use-case, we found mixed results: the decision trees trained by `AdaBoost.F` are faster to train, but the resulting ensemble does not perform as well as the FedAvg

model prediction-wise. Computationally `AdaBoost.F` seems to require less resources, but it does not scale well as the number of involved parties grow (a less demanding algorithm is being developed by the original authors, but we couldn't test it as it has not been released yet).

For future work in addition to further investigate the unexpected good performances of the FNN model, we would also like to investigate this dataset's two other target variables, such as RENAMI and BARCMB. We would also like to better understand the power consumption of these two techniques. Finally, we will work on developing a FLaaS infrastructure for federated pooling in the continuum, which will serve as a framework to experiment with novel ML/DNN models and datasets coming from a diverse set of use-cases.

ACKNOWLEDGMENT

This article describes work undertaken in the context of the ADMIRE project[4], *"Adaptive multi-tier intelligent data manager for Exascale"* which has received funding from the European High-Performance Computing Joint Undertaking (JU) under grant agreement No. 956748.

REFERENCES

[1] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas, "Communication-efficient learning of deep networks from decentralized data," in *Proc. of the 20th Intl. Conference on Artificial Intelligence and Statistics, AISTATS 2017, 20-22 April 2017, Fort Lauderdale, FL, USA*, ser. Proc. of Machine Learning Research, A. Singh and X. J. Zhu, Eds., vol. 54. PMLR, 2017, pp. 1273–1282.

[2] R. Shwartz-Ziv and A. Armon, "Tabular data: Deep learning is not all you need," *Inf. Fusion*, vol. 81, pp. 84–90, 2022.

[3] M. Polato, R. Esposito, and M. Aldinucci, "Boosting the federation: Cross-silo federated learning without gradient descent," in *International Joint Conference on Neural Networks, IJCNN 2022, Padua, Italy, July 18-23, 2022*. IEEE, 2022, pp. 1–10.

[4] T. Hastie, S. Rosset, J. Zhu, and H. Zou, "Multi-class adaboost," *Statistics and its Interface*, vol. 2, no. 3, pp. 349–360, 2009.

[5] F. D'Ascenzo, O. De Filippo, G. Gallone, G. Mittone, M. A. Deriu, M. Iannaccone, A. Ariza-Solé, C. Liebetrau, S. Manzano-Fernández, G. Quadri *et al.*, "Machine learning-based prediction of adverse events following an acute coronary syndrome (PRAISE): a modelling study of pooled datasets," *The Lancet*, vol. 397, no. 10270, pp. 199–207, 2021.

[6] L. Li, Y. Fan, M. Tse, and K.-Y. Lin, "A review of applications in federated learning," *Computers & Industrial Engineering*, vol. 149, p. 106854, 2020.

[7] A. Hard, K. Rao, R. Mathews, S. Ramaswamy, F. Beaufays, S. Augenstein, H. Eichner, C. Kiddon, and D. Ramage, "Federated learning for mobile keyboard prediction," *arXiv preprint arXiv:1811.03604*, 2018.

[8] S. Ramaswamy, R. Mathews, K. Rao, and F. Beaufays, "Federated learning for emoji prediction in a mobile keyboard," *arXiv preprint arXiv:1906.04329*, 2019.

[9] Y. Zhao, J. Zhao, L. Jiang, R. Tan, D. Niyato, Z. Li, L. Lyu, and Y. Liu, "Privacy-preserving blockchain-based federated learning for iot devices," *IEEE Internet of Things Journal*, vol. 8, no. 3, pp. 1817–1829, 2020.

[10] S. Silva, B. A. Gutman, E. Romero, P. M. Thompson, A. Altmann, and M. Lorenzi, "Federated learning in distributed medical databases: Meta-analysis of large-scale subcortical brain data," in *2019 IEEE 16th International Symposium on Biomedical Imaging (ISBI 2019)*, 2019, pp. 270–274.

[11] A. Durrant, M. Markovic, D. Matthews, D. May, J. Enright, and G. Leontidis, "The role of cross-silo federated learning in facilitating data sharing in the agri-food sector," *Computers and Electronics in Agriculture*, vol. 193, p. 106648, 2022.

[12] D. Byrd and A. Polychroniadou, "Differentially private secure multi-party computation for federated learning in financial applications," in *Proceedings of the First ACM International Conference on AI in Finance*, 2020, pp. 1–9.

[13] Q. Yang, Y. Liu, T. Chen, and Y. Tong, "Federated machine learning: Concept and applications," *ACM Transactions on Intelligent Systems and Technology (TIST)*, vol. 10, no. 2, pp. 1–19, 2019.

[14] S. Niknam, H. S. Dhillon, and J. H. Reed, "Federated learning for wireless communications: Motivation, opportunities, and challenges," *IEEE Communications Magazine*, vol. 58, no. 6, pp. 46–51, 2020.

[15] D. M. J. Gutierrez, H. M. Hassan, L. Landi, A. Vitaletti, and I. Chatzigiannakis, "Application of federated learning techniques for arrhythmia classification using 12-lead ecg signals," *arXiv preprint arXiv:2208.10993*, 2022.

[16] Y. Arfat, G. Mittone, R. Esposito, B. Cantalupo, G. M. de Ferrari, and M. Aldinucci, "Machine learning for cardiology," *Minerva Cardiology and Angiology 2022 February; 70 (1): 75-91*.

[17] Y. Liu, Z. Ma, X. Liu, S. Ma, S. Nepal, R. H. Deng, and K. Ren, "Boosting privately: Federated extreme gradient boosting for mobile crowdsensing," in *2020 IEEE 40th International Conference on Distributed Computing Systems (ICDCS)*, 2020, pp. 1–11.

[18] Y. Chen and B. Qi, "Representation learning in intraoperative vital signs for heart failure risk prediction," *BMC Medical Informatics and Decision Making*, vol. 19, no. 1, p. 260, Dec. 2019.

[19] E. D. Adler, A. A. Voors, L. Klein, F. Macheret, O. O. Braun, M. A. Urey, W. Zhu, I. Sama, M. Tadel, C. Campagnari *et al.*, "Improving risk prediction in heart failure using machine learning," *European journal of heart failure*, vol. 22, no. 1, pp. 139–147, 2020.

[20] C. Ye, J. Li, S. Hao, M. Liu, H. Jin, L. Zheng, M. Xia, B. Jin, C. Zhu, S. T. Alfreds *et al.*, "Identification of elders at higher risk for fall with statewide electronic health records and a machine learning algorithm," *International journal of medical informatics*, vol. 137, p. 104105, 2020.

[21] A. Kilic, A. Goyal, J. K. Miller, E. Gjekmarkaj, W. L. Tam, T. G. Gleason, I. Sultan, and A. Dubrawksi, "Predictive utility of a machine learning algorithm in estimating mortality risk in cardiac surgery," *The Annals of Thoracic Surgery*, vol. 109, no. 6, pp. 1811–1819, 2020.

[22] M. Polato, R. Esposito, and M. Aldinucci, "Boosting the federation: Cross-silo federated learning without gradient descent," in *International Joint Conference on Neural Networks, IJCNN 2022, Padua, Italy, July 18-23, 2022*. IEEE, July 2022.

[23] P. Foley, M. J. Sheller, B. Edwards, S. Pati, W. Riviera, M. Sharma, P. N. Moorthy, S.-h. Wang, J. Martin, P. Mirhaji, P. Shah, and S. Bakas, "OpenFL: the open federated learning library," *Physics in Medicine & Biology*, 2022.

[24] K. Bonawitz, E. Hubert, W. Grieskamp, H. Dzmitry, I. Alex, I. Vladimir, K. Chloe, K. Jakub, M. Stefano, M. Brendan *et al.*, "Towards federated learning at scale: System design," *Proceedings of Machine Learning and Systems*, vol. 1, pp. 374–388, 2019.

[25] Z. Alexander, T. Andrew, L. Antonio, S. Benjamin, W. Bobby, B. Emma, N. Jean-Mickael, P.-P. Jonathan, P. Kritika, R. Nick *et al.*, "Pysyft: A library for easy federated learning," in *Federated Learning Systems*. Springer, 2021, pp. 111–139.

[26] L. Heiko, B. Nathalie, T. Gegi, Z. Yi, A. Ali, R. Shashank, O. Yuya, J. Radhakrishnan, V. Ashish, S. Mathieu *et al.*, "Ibm federated learning: an enterprise framework white paper v0. 1," *arXiv preprint arXiv:2007.10987*, 2020.

[27] S. Warnat-Herresthal, H. Schultze, K. L. Shastry, S. Manamohan, S. Mukherjee, V. Garg, R. Sarveswara, K. Händler, P. Pickkers, N. A. Aziz *et al.*, "Swarm Learning for decentralized and confidential clinical machine learning," *Nature*, vol. 594, no. 7862, pp. 265–270, Jun. 2021.

[28] C. He, S. Li, J. So, M. Zhang, H. Wang, X. Wang, P. Vepakomma, A. Singh, H. Qiu, L. Shen, P. Zhao, Y. Kang, Y. Liu, R. Raskar, Q. Yang, M. Annavaram, and S. Avestimehr, "Fedml: A research library and benchmark for federated machine learning," *CoRR*, vol. abs/2007.13518, 2020.

[29] Y. Liu, T. Fan, T. Chen, Q. Xu, and Q. Yang, "FATE: an industrial grade platform for collaborative learning with data protection," *J. Mach. Learn. Res.*, vol. 22, pp. 226:1–226:6, 2021.

[30] D. J. Beutel, T. Topal, A. Mathur, X. Qiu, T. Parcollet, and N. D. Lane, "Flower: A friendly federated learning research framework," *CoRR*, vol. abs/2007.14390, 2020.

[31] S. Silva, A. Altmann, B. Gutman, and M. Lorenzi, "Fed-biomed: A general open-source frontend framework for federated learning in healthcare," in *Domain Adaptation and Representation Transfer, and*

[4] https://www.admire-eurohpc.eu

Distributed and Collaborative Learning - Second MICCAI Workshop, DART 2020, and First MICCAI Workshop, DCL 2020, Held in Conjunction with MICCAI 2020, Lima, Peru, October 4-8, 2020, Proceedings, ser. Lecture Notes in Computer Science, vol. 12444. Springer, 2020, pp. 201–210.

[32] Y. Xie, Z. Wang, D. C. an Dawei Gao, L. Yao, W. Kuang, Y. Li, B. Ding, and J. Zhou, "Federatedscope: A comprehensive and flexible federated learning platform via message passing," *CoRR*, vol. abs/2204.05011, 2022.

[33] D. Dimitriadis, M. H. Garcia, D. M. Diaz, A. Manoel, and R. Sim, "FLUTE: A scalable, extensible framework for high-performance federated learning simulations," *CoRR*, vol. abs/2203.13789, 2022.

[34] N. Wang, Y. Xiao, Y. Chen, Y. Hu, W. Lou, and Y. T. Hou, "FLARE: defending federated learning against model poisoning attacks via latent space representations," in *ASIA CCS '22: ACM Asia Conference on Computer and Communications Security, Nagasaki, Japan, 30 May 2022 - 3 June 2022*, Y. Suga, K. Sakurai, X. Ding, and K. Sako, Eds. ACM, 2022, pp. 946–958.

[35] I. J. Goodfellow, M. Mirza, X. Da, A. C. Courville, and Y. Bengio, "An empirical investigation of catastrophic forgeting in gradient-based neural networks," in *2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Conference Track Proceedings*, Y. Bengio and Y. LeCun, Eds., 2014.

[36] T. Li, A. K. Sahu, M. Zaheer, M. Sanjabi, A. Talwalkar, and V. Smith, "Federated optimization in heterogeneous networks," in *Proc. of Machine Learning and Systems 2020, MLSys 2020, Austin, TX, USA, March 2-4, 2020*, I. S. Dhillon, D. S. Papailiopoulos, and V. Sze, Eds. mlsys.org, 2020.

[37] J. Wang, Q. Liu, H. Liang, G. Joshi, and H. V. Poor, "Tackling the objective inconsistency problem in heterogeneous federated optimization," in *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, and H. Lin, Eds., 2020.

[38] S. P. Karimireddy, S. Kale, M. Mohri, S. J. Reddi, S. U. Stich, and A. T. Suresh, "SCAFFOLD: stochastic controlled averaging for federated learning," in *Proc. of the 37th Intl. Conference on Machine Learning, ICML 2020, 13-18 July 2020, Virtual Event*, ser. Proc. of Machine Learning Research, vol. 119. PMLR, 2020, pp. 5132–5143.

[39] B. Casella, R. Esposito, C. Cavazzoni, and M. Aldinucci, "Benchmarking fedavg and fedcurv for image classification tasks," in *Proceedings of the 1st Italian Conference on Big Data and Data Science, ITADATA 2022, September 20-21, 2022*, ser. CEUR Workshop Proceedings, M. Anisetti, A. Bonifati, N. Bena, C. Ardagna, and D. Malerba, Eds. CEUR-WS.org, 2022.

[40] S. Pati, U. Baid, B. Edwards, M. Sheller, S.-H. Wang, G. A. Reina, P. Foley, A. Gruzdev, D. Karkada, C. Davatzikos *et al.*, "Federated learning enables big data for rare cancer boundary detection," *arXiv preprint arXiv:2204.10836*, 2022.

[41] K. Arabi, "Mobile computing opportunities, challenges and technology drivers," The 51st Annual Design Automation Conference 2014, DAC '14, San Francisco, CA, USA, June 1-5, 2014, Jun. 2014, keynote Talk.

[42] O. De Filippo, F. D'Ascenzo, S. Raposeiras-Roubin, E. Abu-Assi, M. Peyracchia, P. P. Bocchino, T. Kinnaird, A. Ariza-Solé, C. Liebetrau, S. Manzano-Fernández *et al.*, "P2y12 inhibitors in acute coronary syndrome patients with renal dysfunction: an analysis from the renami and bleemacs projects," *European Heart Journal-Cardiovascular Pharmacotherapy*, vol. 6, no. 1, pp. 31–42, 2020.

[43] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, "SMOTE: synthetic minority over-sampling technique," *Journal of Artificial Intelligence Research*, vol. 16, pp. 321–357, 2002.

[44] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, Y. Bengio and Y. LeCun, Eds., 2015.

[45] M. Aldinucci, S. Rabellino, M. Pironti, F. Spiga, P. Viviani, M. Drocco, M. Guerzoni, G. Boella, M. Mellia, P. Margara *et al.*, "HPC4AI: an AI-on-demand federated platform endeavour," in *Proceedings of the 15th ACM International Conference on Computing Frontiers*, 2018, pp. 279–286.

[46] M. Aldinucci, S. Bagnasco, S. Lusso, P. Pasteris, S. Rabellino, and S. Vallero, "OCCAM: a flexible, multi-purpose and extendable HPC cluster," in *Journal of Physics: Conference Series*, vol. 898, no. 8. IOP Publishing, 2017, p. 082039.