IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received 30 June 2023, accepted 28 August 2023, date of publication 6 September 2023, date of current version 12 September 2023.

Digital Object Identifier 10.1109/ACCESS.2023.3312579

APPLIED RESEARCH

FeLebrities: A User-Centric Assessment of Federated Learning Frameworks

WALTER RIVIERA[1], ILARIA BOSCOLO GALAZZO[2], (Member, IEEE), AND GLORIA MENEGAZ[2], (Senior Member, IEEE)

[1]Department of Computer Science, University of Verona, 37134 Verona, Italy
[2]Department of Engineering for Innovation Medicine, University of Verona, 37134 Verona, Italy

Corresponding author: Walter Riviera (walteriviera@gmail.com)

This work was supported in part by Fondazione CariVerona (Bando Ricerca Scientifica di Eccellenza 2018, EDIPO project) under Grant 2018.0855.2019 and in part by AI4Health: empowering neurosciences with explainable AI methods under Grant MIUR D.M. 737/2021.

ABSTRACT Federated Learning (FL) is a new paradigm aimed at solving data access problems. It provides a solution by moving the focus from sharing data to sharing models. The FL paradigm involves different entities (institutions) holding proprietary datasets that, contributing with each other to train a global Artificial Intelligence (AI) model using their own locally available data. Although several studies have proposed methods to distribute the computation or aggregate results, few efforts have been made to cover on how to implement FL pipelines. With the aim of accelerating the exploitation of FL frameworks, this paper proposes a survey of public tools that are currently available for building FL pipelines, an objective ranking based on the current state of user preferences, and an assessment of the growing trend of the tool's popularity over a one year time window, with measurements performed every six months. These measurements include objective metrics, like the number of "Watch," "Star" and "Follow" available from software repositories as well as thirteen custom metrics grouped into three main categories: Usability, Portability, and Flexibility. Finally, a ranking of the maturity of the tools is derived based on the key aspects to consider when building a FL pipeline.

INDEX TERMS Federated learning tools, distributed systems, AI at scale.

I. INTRODUCTION

Federated learning (FL) is a paradigm that aims to solve the data access problem. In the Artificial Intelligence (AI) domain, data represents the starting point for many research and development activities [1], [2], [3]. With increasing attention given to the field, data have also grown in demand and appreciation, redefining priorities in designing and building solutions for real-world applications. A clear demonstration of this growing importance is the creation of dedicated laws, such as the General Data Privacy Regulations (GDPR) [4] in place in the European Union, the Protection of Personal Information Act (POPIA) [5], and the Health Insurance Portability and Accountability Act (HIPAA) [6] in the USA, which is specific for accessing clinical data and medical records. From the AI perspective, this reflects the need to access data to advance the State of the Art (SOA) in a given environment while fully complying with regulations. FL is an effective way to satisfy all these requirements. In a federation of collaborating institutions, what is shared is a common global model that is partially trained by every collaborator using local data. Historically, the approach of training AI models assumes that data would be collected and centralized in a unique infrastructure appropriately equipped with dedicated hardware and software to sustain the computation. High performance computing (HPC) centers are great examples of this approach, as illustrated in Figure 1. In constrast, in an FL setting, data are expected to stay in the exact location where they were collected, while a copy of the AI global model is shared across all institutions participating in a federation. A generic example is shown in Figure 2.

The research community has already started investigating this emerging topic either for its privacy-compliant aspects [1], [7] or as a viable tool for addressing AI challenges in critical domains such as the biomedical context [8], [9], [10]. Although the domain is still relatively new, the literature

The associate editor coordinating the review of this manuscript and approving it for publication was Fabrizio Marozzo.

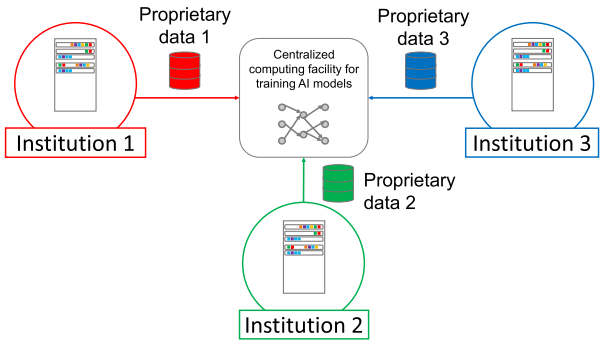

FIGURE 1. Data-to-model: example of legacy approach where data would move to a centralized training facility. Here the AI model is represented as a graph or neural network.

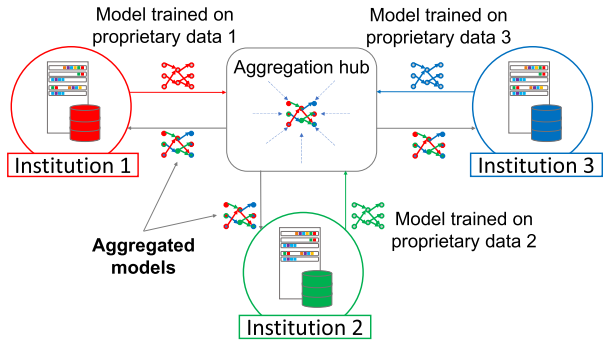

FIGURE 2. Model-to-data: example of a federated approach. A central unit called aggregator would clone and distribute copies of the same model to each collaborating institution. Each of them would then train its copy of the model using local datasets before sharing it back to the aggregator. As the name suggests, the aggregator would ultimately merge the models from different institutions and restart the process by sending out the latest aggregated version.

can already provide helpful surveys on how the concept works and complies with privacy aspects [1], how it can be transferred to the *Internet of Things (IoT)* world [11], and what are the steps to implement it from a protocol, software, and hardware standpoint [11]. The rising interest in the research community and industries *R&D* departments has enriched the literature, which has, in turn, influenced the development and evolution of many new tools for implementing FL pipelines. If, from one perspective, this aspect is encouraging, it reflects the need to obtain clear indications about what tools are currently available, which are the most popular, and what is their level of maturity (in terms of features).

This paper aims at providing four main contributions:

1) Provide an updated list of tools publicly available for implementing FL pipelines.
2) Share the current state of adoption of each tool, including growth trends calculated over a one year time window where measurements from three "harvests" (H_i) were performed.
3) Identify and describe the key aspects required by the research community and map them into a list of features that tools should include.
4) Propose a ranking based on objective metrics, including common indicators and the ability to match the needs highlighted in the previous point.

We genuinely believe that by providing a quantitative and qualitative survey of the FL tools, the research community will be able to: accelerate its activities, promote fairness by proposing an inclusive method to collect comparable studies, and help tool providers identify ways to improve their products. The availability of a ranking of FL tools will also boost their exploitation in the production environment, where such tools still need to be explored.

A. PAPER ORGANIZATION

This paper is composed of six Sections. In section II, we discuss FL implementation related works. Section III focuses on the list of tools currently available to the community, sharing a high-level overview of their popularity and adoption. Section IV augments the retrieved list of tools with the current state of adoption, including the growth trend observed over one year, and Section V discusses the key aspects that should be considered when implementing federated environments for research purposes. These factors are then translated into requirements that FL tools need to satisfy for successful exploitation and consolidated in a raking table. The results are discussed in Section VI and future directions and conclusions are finally addressed in VII.

II. RELATED WORKS

FL is a distributed machine learning (ML) approach that enables organizations to collaborate on projects without sharing sensitive data [12], such as patient records [13], [14] or financial data [15], or data that is not easily accessible, such those stored in remote locations such as satellites or space stations from high-resolution sensors [16]. The basic premise of FL [1], [2] is that the model `moves` to meet the data rather than the data `moving` to meet the model. Therefore, the only minimum data movement required across the federation is the model parameters and their updates.

A. FL SETTINGS

There are two essential components of an FL pipeline: one or multiple institutions owing data and a mechanism to orchestrate the process. Each institution must have local data and be accountable for hosting the training process on proprietary data. The orchestration mechanism may vary, but is mainly of two types: Synchronous or Asynchronous.

In a synchronous scenario, the idea is to have a central unit, often identified as an aggregator [12], [13], acting as a central pivot and determining when to start a new iteration. The aggregator is responsible for cloning the initial model to each collaborating institution, waiting to receive the locally trained copies, and finally merging them, as the name suggests. This type of FL pipeline is usually implemented in big data centers (cross-silo), such as those involved in medical environments [3], [17]. Data centers can store vast amounts of

data and provide the computational power required to process them. In addition, big computing infrastructures, such as HPC centers, can rely on fast and stable connections to the network, simplifying the creation of a more reliable communication channel to interact with a hypothetical aggregator unit.

However, as soon as we move away from data centers towards edge devices, new challenges arise owing to the high variance in products and manufacturers. Devices with different latencies, working frequencies, and hardware features can have different computation times [18], [19]. These are the reasons for the need for an asynchronous FL pipeline. In this scenario, each collaborating institution can share its update at any time, either to a unique aggregator [18], [20], [21] or to other participants in an ''all-to-all'' setup [22], [23].

Another critical point to address is the difference between the horizontal (HFL) and vertical (VFL) federated learning. To understand this difference, we need to consider the space of the features and the model type. In the examples shared thus far, we implicitly refer to the Horizontal FL, where the different collaborators have different data but contribute to the federation by sharing the feature space and training the same model. This is the case for institutions with offices distributed across different locations that would like to train a common model by leveraging the local data stored in each facility in a privacy-compliant manner. In Vertical FL, each collaborator is expected to contribute by providing different bits of information from the same sample. This leads to a scenario in which the feature space accessed by every collaborator may be different from the others. Therefore, each collaborator might train a different model in the vertical configuration. Aggregation, in this case, is represented by the interoperability between collaborators, where to update a model, information coming from the model of another collaborator might be required [24], [25]. For example, in a typical VFL setting, we can see a life insurance agency collaborating with hospitals to build a decision model to obtain more precise estimations of their affiliates. In this case, it is expected that the entities involved in the federation can provide different information about the same user. These two ways of articulating the data for a federation impact the choice of model and how the federation is orchestrated. While in the HFL, there is only one model, and all the collaborators are responsible for ensuring that data are normalized to feed it, the VFL brings some more complexity. In this case, to handle different data types from several institutions, each collaborator should have a local model that can accept the data from that specific institution as input. In addition, there must be a federated model that takes all outputs of the various local models as inputs. As illustrated by Chen et al. [25], the procedure for training Deep Learning (DL) models based on back-propagation [26], [27], needs to deal with the two-level training procedure represented by the different models that need to be managed: one at the collaborator level and the other at the aggregation point. This complexity is also reflected in the challenges that might arise in finding a satisfactory convergence point for the adopted DL model.

B. FL CHALLENGES

Regardless of which FL setting (Synchronous or Asynchronous) or configuration (Horizontal or Vertical) is adopted by a given federation, the research community currently adresses three main areas:

1) Aggregation functions and model convergence starting from different data distributions.
2) Privacy aspects and ways to build a secure FL pipeline for protecting the IP during experiments.
3) Communication efficiency and protocols to improve the FL base infrastructure.

Protecting dataset ownership implies that in most cases, the assumption of dealing with independent and identically distributed (i.i.d.) samples across local nodes does not hold for FL setups [28], [29]. Data distribution can severely impact the training performance by affecting the total accuracy [30], convergence capability, authentication processes (especially in the case of different devices), and speed of the process intended as total time-to-train [31]. In summary, in this setting, the performance of the training process may vary significantly according to the imbalance of the local data samples and the particular statistical distribution of the training examples (i.e., features and labels) stored at the local nodes [2].

In the past few years, institutions have introduced FL deployments to address the need to train AI models. Sectors such as healthcare and finance would benefit from having a setting with greater access to more extensive and diverse datasets without violating privacy laws [32], [33], such as HIPAA, GDPR [4], and POPIA [5]. On the one hand, FL has been designed with security in mind [30], and the set up is just the beginning. Securing execution environments introduces many open challenges to the research field [34]. Key questions include finding a consolidated method to guarantee secure execution (encryption, key exchange, and hardware features) and validating the reliability of intermediate results and collaborators within the federation.

Massive amounts of data are usually stored in ''Data-Lake'' infrastructures. The more machines/institutions that participate in a federation, the more critical is the ability to scale. As mentioned in the previous section, a consolidated method for detecting scarse training contributions (coming from institutions with corrupted or redundant data) is still lacking, to the best of our knowledge. Aggregation functions are currently being evaluated by the research community [28], [33], [35]. Another implication when discussing big scales is the infrastructure and the connectivity chosen by the institutions for communication [2].

C. STUDY RELEVANCE

Several studies have proposed surveys to illustrate the advancement in the field [1], [11]; however, to the best of our knowledge, no one has provided a ranked list based on the ad hoc quality assessment criteria of all the (possible) tools available to the community to implement FL

experiments. A comparison of five tools is provided in [36], which are accessible through a licensed service, without clarifying why or how these tools were precisely selected. Another study [37] provides an attractive comparison table. However, the main focus of this work is to promote an alternative tool specifically for FL benchmarks instead of providing a complete list of the available options to boost the exploitation of FL across the community. Even in this related work, it is unclear why and how the tools discussed were selected. Similarly, [38] proposed a complete benchmarking suite with a helpful decision tree to help users choose a tool based on their requirements. Their recommended ranking also includes some of the evaluation metrics proposed in this study with an even deeper level of detail. However, while we believe in the value of such an approach, the breadth of the offer in terms of tools that can be chosen might represent a constraint for end users. In fact, [38] centers its evaluation on nine tools, but the criteria for which those tools were identified and selected need to be clarified. As we discovered in this work, the list of open-source FL tools can exceed 30, and it is interesting to note how the most popular tool to date was not considered in their decision tree.

III. FEDERATED LEARNING TOOLS

A. METHODS AND PREMISES

This article aims to provide an inclusive and informative list of the current FL tools available to the community for implementing research pipelines in any environment in which accessing distributed data is challenging. To better understand the present scenario, we performed three literature searches, H_i, where $i \in 1, 2, 3$. H_1 was conducted on March 28, 2022, H_2 on September 28, 2022, and H_3 on April 10, 2023.

This activity was inspired by the Preferred Reporting Items for Systematic Reviews and Meta-Analyses (PRISMA) guidelines. More specifically, we followed the *Preferred Reporting Items for Systematic Review and Meta-Analysis of Diagnostic Test Accuracy Studies (PRISMA-DTA): explanation, elaboration, and checklist* [39]. In particular, the guidelines we followed were a selection of the guidelines described in the *PRISMA 2020 checklist*, accessible on the official PRISMA website: http://www.prisma-statement.org/. Below is a detailed description of the items extracted from the PRISMA guidelines that were identified as applicable to this collection. Items not included in the list below were either not sharing best practices on the "Method" (i.e., best practices for "Title," "Introduction," and "Abstract" for systematic reviews) or not directly relatable to this contribution as it does not fully match a "Systematic Review." Some examples of discarded items are as follows:

- 12) *Specify for each outcome the effect measure(s) (e.g., risk ratio, mean difference) used in the synthesis or presentation of results.*
- 14) *Describe any methods used to assess risk of bias due to missing results in a synthesis (arising from reporting biases).*
- 19) *For all outcomes, present, for each study: (a) summary statistics for each group (where appropriate) and (b) an effect estimate and its precision (e.g., confidence/credible interval), ideally using structured tables or plots.*

On the contrary, here is the list of items considered for this work. The numbers reported below are a direct references to the PRISMA document.

- 5) *Specify the inclusion and exclusion criteria for the review and how studies were grouped for the syntheses.*
- 6) *Specify all databases, registers, websites, organisations, reference lists and other sources searched or consulted to identify studies. Specify the date when each source was last searched or consulted.*
- 7) *Present the full search strategies for all databases, registers and websites, including any filters and limits used.*
- 8) *Specify the methods used to decide whether a study met the inclusion criteria of the review, including how many reviewers screened each record and each report retrieved, whether they worked independently, and if applicable, details of automation tools used in the process.*
- 13.b) *Describe any methods required to prepare the data for presentation or synthesis, such as handling of missing summary statistics, or data conversions.*
- 13.d) *Describe any methods used to synthesize results and provide a rationale for the choice(s). If meta-analysis was performed, describe the model(s), method(s) to identify the presence and extent of statistical heterogeneity, and software package(s) used.*
- 16.a) *Describe the results of the search and selection process, from the number of records identified in the search to the number of studies included in the review, ideally using a flow diagram.*
- 16.b) *Cite studies that might appear to meet the inclusion criteria, but which were excluded, and explain why they were excluded.*
- 23c) *Discuss any limitations of the review processes used.*

Each of these items was used to frame the study. The following map illustrates how the single guidelines contributed to shaping the sections:

- Items 5, 6, 7 and 8 were considered for building this Section;
- Items 13a and 13d, were used to build the comparison table in Section IV;
- Items 16a, 16b and 23c, were used to structure the discussion of the results provided in Section V.

B. EXPLORING TOOLS

To objectively build the list of tools, we performed three harvests, H_1, H_2, and H_3, with roughly six months of cadence (184 and 194 days respectively). The collection method was the same as that described below. We used three different

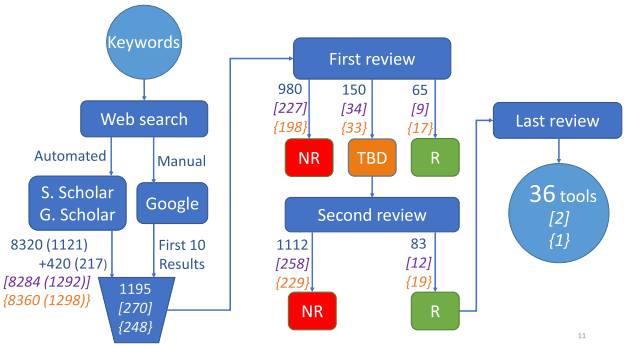

FIGURE 3. Collection pipeline implemented for the harvests. On each arrow is outlined the number of articles retrieved and filtered. Numbers between squared brackets refer to the outcome of H_2, while the curly brackets summarize the numbers from H_3. The output numbers of the second review are obtained by summing the numbers of a given category from the first review with the respective class of the second review. Please note that despite the September harvest returning 1292 articles, only 270 were new findings. The same consideration applies for H_3, with 1298 to 248 articles.

fl_pytorch: optimization research simulator for federated learning
fedlab: a flexible federated learning framework
sunday-fl – developing open source platform for federated learning
gfl: a decentralized federated learning framework based on blockchain
hyfed: a hybrid federated framework for privacy-preserving machine learning
graphfl: a federated learning framework for semi-supervised node classification
openfl: an open-source framework for federated learning

federated learning for vehicular networks
securefl: privacy preserving federated learning with sgx and trustzone
secureml: a system for scalable privacy-preserving machine learning
fedgraphnn: a federated learning system
flaas: federated learning as a service
a crowdsourcing framework for on-device federated learning
fl-ntk: a neural tangent kernel-based framework for federated learning analysis

effects of mobile gaming patterns on learning outcomes: a literature review
collaborative logical framework: an e-learning assesment tool in .lrn platform
sagittarius: a tool to enhance the collaboratibe work in virtual learning environments
predicting machine learning pipeline runtimes in the context of automated ML
model-sharing games: analyzing federated learning under voluntary participation
tensorflow lite micro: embedded machine learning on tinyml systems
collaborative creation and training of social bots in learning communities

FIGURE 4. A subset of examples of selected articles for each category: "relevant" green, "uncertain" blue and "non-relevant" red.

search engines: Google Scholar [40], Semantic Scholar [41], and the standard Google website.

For the first two, we developed a script to automatically query search engines using a collection of keywords on the topic. We built such a collection by combining each item p of a list of prefixes P with each element s in a list of suffixes S. The set of prefixes was populated with the ''federated learning'' keyword, and other synonyms or more related terms used in the literature to express similar concepts: P='federated learning', 'privacy-preserving machine learning', 'collaborative learning', 'collaborative machine learning'.

The set of suffixes was built around adjectives and secondary aspects, like 'tools, library' and 'open-source': S='framework', 'tool and framework open source', 'tool and framework open-source', 'open source framework', 'open source tool', 'open source library'.

This led to a prosperous and inclusive search of all the relevant articles and works in the domain.

Google Scholar helped capture all related works where a given keyword (or part of it) was mentioned in the paper and not only in the title. In H_1, we identified a cumulative list of 420 related articles, of which 217 were unique. Despite the contribution, due to a service limitation, the reported numbers refer to H_1: the article harvest completed in March '22 only.

To build a more robust and consistent set of related works, we leveraged the Semantic Scholar service [42]. The website allows users to perform queries and sort the outcome according to four metrics: ''Relevance'', ''Citations-count'', ''Most Influential Paper'', and ''Recency''.

We repeated searches using all the keywords for all the four sorting types mentioned above, obtaining 1121 (out of 8320) unique articles during H_1, 1291/8284 unique articles with H_2 and 1298/8360 unique articles in H_3. The number of new articles retrieved in H_2 that were unavailable in H_1 was 270, while H_3 enriched the search with 248 new findings.

Finally, we used the standard Google search engine to ensure we could capture all the relevant FL tools yet to be described in a published paper. To do so, we evaluated the first ten results obtained by querying the search engine with the same list of keywords used previously. This step allowed us to enrich the list with additional FL frameworks, such as Nvidia Flare [43], Tensorflow Federated [44], and IBM Federated [45].

Once we obtained the three lists of unique titles described above, we finally merged them, resulting in 1195 unique articles discovered in H_1, 1292 retrieved in H_2, and 1298 in H_3. We then started pruning results by manually reviewing and labeling the list in three different buckets: ''relevant''(R), ''non-relevant'' (NR), and ''uncertain'' (TBD).

Articles considerably unrelated to the topic (e.g., work mentioning ML methods or collaborative learning platforms for schools) were discarded from the collection. After the first labeling cycle, we had 65 R, 980 NR, and 150 TBD for H_1, 9 R, 227 NR, and 34 TBD for H_2 and 17 R, 198 NR, and 33 TBD for H_3

Figure 4 shows an example of articles captured by the three categories.

The ''uncertain'' category required us to conduct a deeper review of the work. All articles in this list underwent a second round of labeling. The objective was to review 150 papers on the TBD list and to allocate them to the R or NR. As a result, we obtained 83 R and 1112 NR for H_1, 12 R and 258 NR for H_2 and 19 R with 229 NR for H_3.

A summary of the adopted research pipeline and the results collected during each harvest is shown in Figure 3.

Ultimately, a deeper understanding of the relevant papers was performed to draw the final list of FL tools. This final review allowed us to identify 36 suitable tools during H_1, two additional tools during H_2, and one last tool added to H_3. The complete list of tools retrieved in H_1 with the indicators from the Github and Gitlab repositories is presented in Table 1.

TABLE 1. Tool popularity table, March harvest H_1. Legend: This table shows the list of 36 tools retrieved in March with their respective Git indicators and the cumulative scores. The results are sorted from the most popular tools on the top to the less popular tools on the bottom. Indicated with *NA* are the tools for which the Git repository was unavailable or not publicly accessible. If not specified otherwise, all the repositories are Github projects.

	TOOL	Watch	Fork	Star	ETA (days)	SCORES
1	PySyft [46]	211	1800	8000	1714	1.95
2	FATE [47]	134	1200	4100	1134	1.60
3	FedML [48]	37	331	1100	614	0.80
4	Tensorflow Federated [44]	66	447	1800	1301	0.59
5	Flower [49]	20	15	825	770	0.37
6	OpenFL [50]	10	76	286	437	0.28
7	IBM-Federated [45]	20	85	292	647	0.20
8	FedLab [51]	6	30	180	383	0.19
9	LEAF [52]	18	187	470	1249	0.18
10	FedGraphNN [53]	9	33	147	383	0.16
11	Fedlearn-algo [54]	8	34	80	255	0.16
12	FedJAX [55]	11	28	172	461	0.15
13	PyVertical [56] [57]	12	35	101	661	0.07
14	PriMIA [58]	8	18	102	707	0.06
15	Substra [59]	8	24	143	1252	0.05
16	Fedn [60]	7	20	59	622	0.05
17	FedBioMed (GitLab) [61]	NA	23	3	332	0.04
18	OpenFED [37]	2	1	17	300	0.02
19	APPFL [62]	2	1	7	172	0.02
20	HyFed [63]	4	3	9	361	0.01
21	PyFed [64]	3	2	5	544	0.01
22	dsMTL [65]	2	2	0	266	0.01
23	Sunday FL [66]	1	4	2	524	0.00
24	DecFL [67]	2	2	3	717	0.00
25	MTC-ETH [68]	1	1	2	881	0.00
26	Vantage6 [69]	5	2	0	1835	0.00
27	Sherpa ai [70]	3	0	0	0	-
28	FL-Pytorch (Pytorch Federated) [71]	NA	NA	NA	NA	NA
29	Chiron [72]	NA	NA	NA	NA	NA
30	FedHealth [73]	NA	NA	NA	NA	NA
31	FAE [74]	NA	NA	NA	NA	NA
32	GENO [75]	NA	NA	NA	NA	NA
33	FedTGan [76]	NA	NA	NA	NA	NA
34	FL-Bench [77]	NA	NA	NA	NA	NA
35	IPLS [78]	NA	NA	NA	NA	NA
36	Nvidia-Flare [43]	NA	NA	NA	NA	NA

IV. TOOLS POPULARITY AND LEVEL OF ADOPTION

After retrieving the list of tools, our goal was to understand each item's popularity and adoption level from a community perspective. Each Git repository has public indicators, such as the number of watches (W), forks (F), and stars(S). The Watch indicator captures the number of users who actively watch the repository. These users receive updates when new actions are taken from a repository. The number of forks indicates the number of times a repository has been forked. It is a good indicator of how many interested users might develop code to extend the tool. Finally, the number of stars indicates the number of likes that the repository has received. This final indicator might need to be more accurate in capturing actual users, but it can provide a reasonable estimation of reach in terms of how many people have seen the tool at least once. For practicality, we aimed to combine these three aspects into

one consolidated score to provide a *popularity* driven ranking of the tools. Since popularity would also depend on the time a given repository was made available to the community, we wanted to normalize all the values with a timing factor. This step ensured that newer repositories with less exposure to the community would not be affected by a low score. The scores are calculated as follows:

$$\text{Score}_{\text{H}_\text{i}} = \frac{\frac{1}{3}(W + F + S)^{H_i}}{\text{ET}_{\text{H}_\text{i}}} \tag{1}$$

where $\text{ET}_{\text{H}_\text{i}}$ is the time elapsed between the day of the tool's first commit and the harvest date H_i. Table 1 shows the scores associated with the tools retrieved in H_1, the March harvest.

An initial understanding of which tools were accessible to the community was helpful but could provide a limited view of the bigger picture. Indeed, while Git indicators can share important insights about user preferences in a given time frame, they do not necessarily capture community trends from a popularity growth rate perspective.

To access this information, we observed the list of tools over a time window to determine which tools were being considered by the community at a higher pace. Thanks to the second harvest H_2 held on September 22) and the third H_3 done on April 23, we discovered new tools to add to the list and updated the values of W, F, and S for each of the tools found in March. Knowing the differences between the indicator's value recorded in the various harvests, given two harvests, H_i and H_j where $j > i$, we computed the growth rate for each tool as follows:

$$\text{GR}_{\text{ij}} = \frac{\frac{1}{3}((W + F + S)^{H_j} - (W + F + S)^{H_i})}{\Delta_{ji}} \tag{2}$$

where Δ_{ji}, is the difference in days between Date(H_j) and Date(H_i). More precisely, $\Delta_{21} = 184$ days, $\Delta_{32} = 194$ days and $\Delta_{31} = 378$ days. We calculated the growth rate GR for all possible combinations: GR_{12} captures the growth of the repositories between H_1 and H_2, GR_{23} for the growth between H_2 and H_3 and GR_{13} captures the yearly evolution between H_1 and H_3. The results of these computations are shown in Figure 5.

Interestingly, the order of the tools based on the popularity level observed in H_1 and captured in Table 1 does not match the growth rate highlighted in Figure 5.

V. PRIORITIES IN FL TOOLS FOR RESEARCH

The previous Sections illustrated how we could retrieve a list of relevant tools for building FL pipelines and which are preferred by the community. In this Section, our goal is to provide a new ranking of tools to identify the most mature. We focused on the specific features of each tool, regardless of the popularity aspects, as outlined in the previous Sections.

The task consists of retrieving and evaluating FL tools that can be adopted to boost the exploitability. To perform this classification, we defined a set of measures based on the different needs and expectations that a tool should satisfy according to the application field and final objectives.

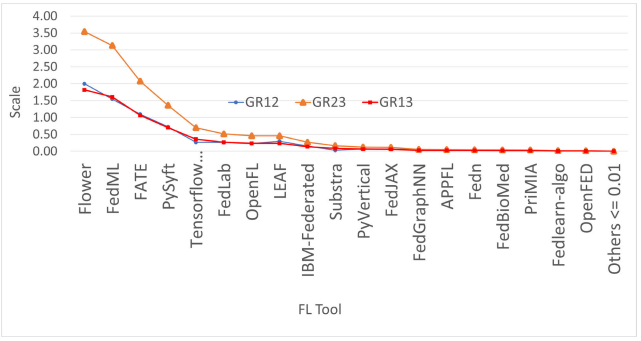

FIGURE 5. Popularity growth rate: this graph illustrates which tools have been gaining more popularity in the community between March and September '22 GR_{12}, between September '22 and April 23 GR_{23}, and over an observation window of 374 days (roughly one year), GR_{13}. Tools that did not have a repository available in H_1 (March '22) were excluded from this chart as it would not be possible to track any evolution.

As described in the SOA Section, there are several ways of implementing FL. From bridging different data-center institutions together at the production level to leverage the agile nature of IoT devices, FL pipelines must be shaped according to the needs, goals, and constraints to consider. However, in line with our purpose of identifying the most mature FL tools for research activities, there is no need to filter results based on data centers or edge devices as long as the tools will provide the possibility to simulate multiple decentralized abstracted computing hubs.

However, other considerations should be drawn around what has been highlighted in the FL challenges regarding data distribution, confidential computing, and communication efficiency.

Indeed, those aspects are essential because they can directly reflect requirements that FL tools need to fulfill to be considered. For example, they all highlight the need for a flexible and modular architecture to allow maximum research customization for aggregation functions, communication protocols, or privacy-preserving and security features. Another practical insight derived from Items 2 and 3 concerns the ability of a tool to scale out on multiple computing machines. As demonstrated in [16], the applicability of FL is not only related to the need to access data complying with regulations. It can also refer to data that are not readily retrievable, such as those from a satellite or space station. Furthermore, the Medical [3], or Geo-spatial [16] environments are usually sources of high-resolution data acquired by machines manufactured by different companies, which could map the need for dedicated pre-processing routines to be used to feed an AI model. FL approaches can be tested on multiple machines hosting different datasets (generated by different equipment) or by simulating multiple parallel instances running on the same computing node. The first setting is preferred because it enhances the reliability of the conclusions when investigating the privacy and communication efficiency aspects.

In addition to what has been outlined, other practical considerations related to the research environment may also

apply. The easier the path to results in a research environment, the faster the deployment of this technology in real institutions. For example, being able to set up a federated environment quickly by leveraging friendly API, re-using common and well-established language (such as Python), and AI platforms (such as PyTorch or Tensorflow to mention two) having access to direct support channels or useful documentation can represent critical aspects for simplifying research activities in different domains.

A. EVALUATION METRICS

Based on these observations, we consolidated a list of evaluation parameters and guidelines organized as follows:

1) Usability
   - Documentation (Docs.)
   - Developer Experience (DX)
   - Language (Lang.): Python, R, C++, etc.
   - Supported AI frameworks: PyTorch, Tensorflow (TF), etc.
   - Type of AI: Machine Learning (ML), Deep Learning (DL)
2) Portability
   - Templates/Examples availability (T/E)
   - Distribution channels: Anaconda, Pipy, etc.
   - Multi-node mode
   - Open-source
3) Flexibility
   - Containers/Virtualization (C/V)
   - Modular architecture
   - Horizontal or Vertical (H/V) FL
   - Synchronous or Asynchronous (S/A) FL
   - Privacy and Security independent module? (PVC/SEC)
   - Easy integration with other tools

We used these parameters to build an "Evaluation Table" 2 for the tools identified in the previous Section. The table was populated with information retrieved from publicly available resources for each tool (see "Docs." column in Table 2). As can be seen, there is a mismatch between the tools listed in Table 1 and those listed in Table 2. This is mainly due to the following four reasons:

1) Tools not open-source, like Sherpa-ai [70].
2) Missing repositories: is the case of tools that have not yet released their codes after the paper publication: Chiron [72], FedHealth [73], FAE [74], GENO [75], FedTGan [76] and IPLS [78].
3) Coherent but not suitable: is the case for LEAF [52], FL-Bench [77], and PyFed [64], which are positioned for benchmarking purposes and, therefore, might lack essential features for conducting more extensive research activities. FedGraphNN [53] is a sub-project of the more significant initiative known as FedML [48] already included in this survey.
4) New tools or new openings: is the case for Nvidia-Flare [43] (a sub-project of Nvidia-Clara) and FL-Pytorch [71] which opened their repositories at some point after H_1 harvest, as well as FLUTE [79], and PLATO [80], which were retrieved (and added) during H_2 and XFL [81] retrieved in H_3.

After pruning the 12 tools that did not qualify, adding 2 (despite the substitution with Nvidia-Flare [43], Nvidia-Clara was already captured by Table 1) at the end of April '23 harvest H_3, we ended up with a list of 28 total tools.

In a second instance, a score was associated with each cell based on a quantitative assessment. Aiming at objective classification of the tools, we captured qualitative aspects in a very inclusive manner. These rewarding tools demonstrated additional development efforts for the community through the available material without penalizing new promising tools that might still be under development.

More precisely, the proposed scoring method rewards completeness rather than excellence. This means that a tool that supports 20 programming languages but needs other relevant features would not outperform a tool that supports only one programming language but has more features that simplify the user experience. This is achieved using scores with a reduced range of values to allow newer but promising tools to compete with more mature tools. More in detail, we adopted a simple approach to assign a score to each cell and designed the "Score Table" 3:

- Documentation: We considered having Paper *P* and/or a public repository for the tool *Gh* (Github) or *Gl* (Gitlab) as a minimum requirement. Therefore, we assigned zero to all the tools that did not match this expectation; rewarded with 1 point the tools with at least one additional source of information (such as a dedicated web page or richer documentation that would go beyond Readme files on repositories or Slack support). Finally, 1.5 points were given to all those that provided two or more sources.
- Developer Experience (DX): we assigned 0 points to all the tools that did not seem to mention nor provide a user interface of some sort (e.g., Jupyter notebook [82] or Google Colab [83] to mention two). We assigned 1 point to all the tools with at least 1 form of user interface abstracting from programming on the command line. Finally, 1.5 points were given to all the tools with two or more user interfaces.
- Language: We assigned 0 points where information about the supported version was not clearly outlined in the documentation. One point was given to the tools supporting at least one language (or one version), and 1.5 points were given to all tools supporting two languages (or two versions of a language). Finally, 2 points were given to all the tools where the engineering team made an the extra effort to support more than two languages (or more than two versions of the same language).
- Supported AI frameworks: We assigned 0 points where the information about the supported AI frameworks was not clearly outlined in the documentation; 1 point was

TABLE 2. Tool evaluation table: in documentation ("Docs.") and distribution channels ("Dist. channel") columns, *P* = Paper, *Gh* = Github, *Gl* = GitLab, *W* = Website. In the "Supported AI Type" column, DL = Deep Learning and ML = Machine Learning. Column "H/V" differentiates between "Horizontal" and "Vertical" FL, while the "Sync/Async" column indicates whether the tool supports synchronous *S* or asynchronous (*A*) workflows. The *NM* label refers to "Not Mentioned", meaning that the information did not appear as mentioned in the available documentation.

TOOL	Docs.	DX	Lang.	AI frameworks	AI type	Examples	Dist. channels	Multinode?	C/V	H/V	Sync/Async	PVC/SEC	Tools integration
APPFL [62]	P, Gh, Pipy, W	N.M.	Python 3.6+	Pythorc	DL	Yes	Gh, Pipy	Yes	Docker	H	S,A	N.M.	MPI orchestration
DecFL [67]	P, Gh	N.M.	N.M.	TF	DL	Yes	Gh	Yes	Required	H	S	Yes	N.M.
dsMTL [65]	P, Gh,	N.M.	R only	R-based	ML	Yes	Gh	Yes	N.M.	H	N.M.	Yes	N.M.
FATE [47]	P, Gh, W	N.M.	3.8, 3.9	N.M.	ML, DL	Yes	Gh	Yes	Docker	H,V	S,A	Yes	KubeFATE, flake, Spark
FedBioMed [61] (Gl)	P, Gh	Jupyter	python	Pythorc, TF	DL	Yes	Gh	Yes	Required	H	N.M.	Yes	Flake, Sklearn
FedJAX [55]	P, Gh, W	G. Colab, Jupyter	Python	Pythorc, TF	DL	Yes	Gh, Pipy	Simulated	N.M.	H	S,A	N.M.	N.M.
FedLab [51]	P, Gh, W	N.M.	python 3.6+	Pytorch	DL	Yes	Gh, Pipy	Yes	Docker	H	S,A	Yes	N.M.
Fedlearn-algo [54]	P, Gh	N.M.	Python 3.6, 3.7	Pythorc, TF	ML, DL	Yes	Gh	Yes	Docker	H,V	N.M	Yes	Hugging Face, Sklearn
FedML [48]	P, Gh, W, dedicated links, Slack	N.M.	Python	Pytorch, TF, JAX, mxnet	ML, DL	Yes	Gh, Pipy	Yes	Docker	H	S,A	N.M.	MPI, NCCL, MQTT, gRPC, Pytorch RPC
Fedn [60]	P, Gh, dedicated docs	Yes	Python 3.8, 3.9, 3.10	Pytorch, TF	DL	Yes	Gh	Yes	Required	H,V	N.M.	Yes	C++, scikit-learn
Flower [49]	P, Gh, slack, W, +	Jupyter, colab	Python 3.6+	Pytorch, TF, MxNet, Jax	ML, DL	Yes	Gh, Pipy	Yes	Yes	H	S,A	Yes	Android, flake, hugging Face
FLUTE [79]	P, Gh, W	N.M.	Python 3.8	Pytorch	DL	Yes	Gh	Azure Cloud	N.M.	H	S,A	N.M.	HuggingFace
HyFed [63]	P, Gh	WebAPP	N.M.	N.M.	N.M.	Yes	Gh	Yes	N.M.	H	N.M.	Yes	N.M.
IBM-Federated [45]	P, Gh, W, video tutorials	Jupyter	Python 3.6	Pytorch, TF	ML, DL	Yes	Gh, wheel	Yes	Docker	H	N.M.	Yes	Ray, Openshift
IPLS [78]													
MTC-ETH [68]	P, Gh	N.M.	N.M.	N.M.	N.M.	N.M.	Gh	Yes, but low number	Required	H	N.M.	Yes	N.M.
Nvidia Flare [43]	P, Gh, W	Jupyter	Python 3.8, 3.10	Pytorch, TF	DL, ML	Yes	Gh, pip	Yes	Docker	H	S	Yes	MONAI
OpenFED [37]	P, Gh, W	Jupyter	Python 3.6	Pytorch	DL	Yes	Gh, Pipy	Simulated	N.M.	H	N.M.	Yes	N.M.
OpenFL [50]	P, Gh, slack, W, videos	Jupyter, Colab	Python 3.6, 3.7, 3.8	Pytorch, TF	DL	Yes	Gh, Pipy	Yes	Docker	H,V	S,A	Yes	MonAI, Hugging Face
PLATO [80]	P, Gh, Website	N.M.	Python 3.9	Pytorch, TF, Mindspore	DL	Yes	Gh, pip	Yes	Docker	H	S,A	N.M.	Catalyst, Mindspore
PriMIA [58]	P, Gh, W	N.M.	N.M.	Pytorch	DL	Yes	Gh	Simulated	N.M.	H	N.M.	Yes	PySyft, K8s
PySyft [46]	P, Gh, Slack, Video, Courses	Jupyter	Python 3.8	Pythorc, TF	DL	Yes	Gh, Pipy	Yes	Docker, VMs. +	H	N.M.	Yes	PySyft, k8s
FL-Pytorch [71]	P, Gh, W, slack, videos	Custom GUI	Python 3.9	Pytorch	DL	Yes	Gh, pipy	Simulated	N.M.	H	N.M.	N.M.	N.M.
PyVertical [56] [57]	P, Gh	Jupyter	Python 3.6,3.7,3.8	N.M.	DL	Yes	Gh	Yes	Docker	V	N.M.	Yes	Syft
Substra [59]	P, Gh, W, Slack	N.M.	N.M.	N.M.	N.M.	Yes	Gh, Pipy	Yes	docker	H	S,A	N.M.	N.M.
Sunday FL [66]	P, Gh, youtube	N.M.	Python, java	N.M.	N.M.	Yes	Gh	Yes	Docker	H	N.M.	N.M.	Azure
Tensorflow Federated [44]	P, Gh, W	Colab	Python 3	TF	DL	Yes	Gh, Pipy	Simulated	N.M.	H	S,A	N.M.	N.M.
Vantage6 [69]	P, Gh, W, youtube	N.M.	Python 3.7	N.M.	N.M.	N.M.	Gh, Pipy	Yes	Docker	H	N.M.	N.M.	N.M.
XFL [81]	P, Gh, W	N.M.	Python 3.9	Pytorch, TF	DL, ML	Yes	Gh	Yes	Docker	H,V	N.M.	Yes	Crypto Algorithms

given to each supported framework. When the number of different supported frameworks exceeded 2, we assigned a maximum score of 2.5 points.

- Type of AI: 0 points if not mentioned in the documentation, and 1 for each type of AI supported (ML or DL).
- Templates/Examples availability.
- Distribution channels: we set the minimum requirement for the ability to download a repository and install the tool from there. Therefore we assigned zero points to all the tools respecting this minimum requirement, 1 point to all the tools that had at least one additional way to access the software package (e.g., Pipy or Anaconda); finally, 1.5 points for all the tools that could be installed in two or more ways.
- Multi-node mode: zero when only the simulated environment on a single computing machine was mentioned. One point to all the tools that allow implementing real federation on multiple nodes and 0.5 if this capability has limitations or constraints.
- Open-source: all the tools presented in the table are open-source. This column was not included in the table.
- Containers/Virtualization: zero was given where the documentation did not provide any of the two options. One point was given where either a containerized or virtualized environment was supplied. 1.5 points when two or more options were listed and 0.5 when containers were available but also presented as the only way to access the tool.

TABLE 3. Tool scoring table.

#	TOOL	Docs.	DX	Lang.	AI frameworks	AI type	Examples	Dist. channels	Multinode?	C/V	H/V	Sync/Async	PVC/SEC	Tools integration	TOTAL
1	Flower [49]	1.5	1.5	2	2.5	2	1	1	1	1	0	1	1	1.5	17
2	OpenFL [50]	1.5	1.5	2	2	1	1	1	1	1	1	1	1	1.5	16.5
3	IBM-Federated [45]	1.5	1	1	2	2	1	1	1	1	0	0	1	1.5	14
4	PySyft [46]	1.5	1	1	2	1	1	1	1	1.5	0	0	1	1.5	13.5
5	Nvidia Flare [43]	1	1	1.5	2	2	1	1	1	1	0	0	1	1	13.5
6	FedML [48]	1.5	0	1	2.5	2	1	1	1	1	0	1	0	1.5	13.5
7	Fedn [60]	1	1	2	2	1	1	0	1	0.5	1	0	1	1.5	13
8	Fedlearn-algo [54]	0	0	1.5	2	2	1	0	1	1	1	0	1	1.5	12
9	XFL [81]	1	0	1	2	2	1	0	1	1	1	0	1	1	12
10	PLATO [80]	1	0	1	2.5	1	1	1	1	1	0	1	0	1.5	12
11	FATE [47]	1	0	1.5	0	2	1	0	1	1	1	1	1	1.5	12
12	APPFL [62]	1	0	2	1	1	1	1	1	1	0	1	0	1	11
13	FedLab [51]	1	0	2	1	1	1	1	1	1	0	1	1	0	11
14	FedBioMed [61] (GitLab)	0	1	1	2	1	1	0	1	0.5	0	0	1	1.5	10
15	FedJAX [55]	1	1.5	1	2	1	1	1	0.5	0	0	1	0	0	10
16	OpenFED [37]	1	1	2	1	1	1	1	0.5	0	0	0	1	0	9.5
17	Tensorflow Federated [44]	1	1	2	1	1	1	1	0.5	0	0	1	0	0	9.5
18	PyVertical [56] [57]	0	1	2	0	1	1	0	1	1	0	0	1	1	9
19	FL-Pytorch [71]	1.5	1	1	1	1	1	1	0.5	0	0	0	0	0	8
20	FLUTE [79]	0	0	1	1	1	1	0	0.5	0	0	1	0	1	7.5
21	PriMIA [58]	1	0	0	1	1	1	0	0.5	0	0	0	1	1.5	7
22	Sunday FL [66]	1	0	1.5	0	0	1	0	1	1	0	0	0	1	6.5
23	dsMTL [65]	0.0	0.0	1.0	1.0	1.0	1.0	0.0	1.0	0.0	0.0	0	1.0	0.0	6
24	Substra [59]	1	0	0	0	0	1	1	1	1	0	1	0	0	6
25	DecFL [67]	0	0	0	1	1	1	0	1	0.5	0	0	1	0	5.5
26	Vantage6 [69]	1.5	0	1	0	0	0	1	1	1	0	0	0	0	5.5
27	HyFed [63]	0	1	0	0	0	1	0	1	0	0	0	1	0	4
28	MTC-ETH [68]	0.0	0.0	0.0	0.0	0.0	0.0	0.0	1.0	0.5	0	0	1.0	0.0	2.5

- Modular architecture: based on the analysis of the repositories of the tools, we trust that all of them respect this parameter. More specifically, they all have proven to have separate entities (such as client-server and orchestration processes) that can be launched independently.
- Horizontal or Vertical: a tool must implement at least one. Therefore we rewarded 1 point only to the tools that would allow executions in both settings.
- Synchronous or Asynchronous: Asynchronous tools can simulate synchronous orchestration with fewer efforts (i.e., active waiting from the aggregation point) than the one required in the opposite scenario (i.e., different communication protocol orchestration). Therefore, we rewarded 1 point only to the tools that would allow execution in Asynchronous. Where ''Not mentioned'', given that a tool must implement at least one, we assumed the default to be synchronous and assigned 0 points.
- Privacy and Security independent module: zero point to the tools that focused on the ability to implement an FL pipeline but did not seem to mention nor highlight the possibility of tweaking or injecting any privacy or security module (i.e., Homomorphic encryption, secured communication protocols, blockchain). One point to all the tools that included at least one.
- Easy integration with other tools: same process applied for evaluating the containers and virtualization mechanism.

VI. DISCUSSION

This paper proposes a survey of the public tools currently available for building FL pipelines. After retrieving the list of tools, we evaluated them using three different metrics: the tool's popularity (based on community adoption), growth rate, and maturity (based on our proposed review). These evaluations led to various rankings. An in-depth discussion of the results is provided below.

Based on the resulting ranking in Table 3, the most mature tools are Flower [49], OpenFL [50], and IBM-Federated [45]. Although the firsts two are fairly close to each other, the third appears to have a solid distance. PySyft [46], Nvidia-flare [43] and FedML [48] followed the same rate. Fedn [60] comes 7th.

This is just an initial observation, but things change when we integrate the popularity results highlighted in Table 1 and the growth rate outlined in Figure 5 into the equation. In Table 1, PySyft [46] and FATE [47] are the two most popular tools according to the developer's community, while Flower [49], OpenFL [50], and IBM-federated [45], cover the 5th, 6th, and 7th placement, respectively, with a considerable distance from the first two. An interesting aspect is the clear gap between what the community awarded as the most popular tools and what this work outlined as the most matures. Similarly, another essential element is the result highlighted by the growth rates reported in Figure 5. Leading that ranking is Flower [49], followed by FedML [48], FATE [47] and PySyft [46]. OpenFL [50] is in the 7th placement, with a growth rate of 0.23, approximately nine times smaller than the Table leader, which has a value of 1.82. With a deeper look at the various growth rates, we can see how generally $GR_{23} > GR_{12}$. This means that the tools grew more between H_2 and H_3, than between H_1 and H_2 despite having a time gap of approximately six months in each window. While this is true for almost all tools, the first four seem to have taken a much more significant leap. This could confirm the effectiveness of the work done by the development team and community contributions. However, it is crucial to note that these leaps did not have a significant impact over an observation window of 374 days (nearly one year). With some exceptions, growth rate GR_{13} seems to confirm the growth rates highlighted by GR_{12}. By comparing these two curves, we can see how Flower [49] had a stronger start in the first six months but then decreased over one year (i.e., $GR_{12} > GR_{13}$) compared to FedML [48], which gained more points in GR_{23} than it

had in GR_{12}. The same considerations apply to Tensorflow-Federated (tff) [44], and Substra [59].

One interesting aspect is that regardless of the scoring we decide to consider, the top five placements seem to be occupied by the same names, re-shuffled a bit. Of the 15 possible names, we can only count up to eight different tools. Of these eight different names, four are more dominant as they appear in at least two rankings: Flower [49], FedML [48], FATE [47], and PySyft [46]. The remaining four were Tensorflow-Federated (tff) [44], OpenFL [50], Nvidia-flare [43] and IBM-federated [45]. Among these eight, only OpenFL [50] supports Vertical FL.

Despite our efforts to adopt objective scoring when building Table 3 as described in Section V, we are aware that other valid scoring alternatives might exist. For example, a more in-depth analysis of all functional features provided by each tool (such as communication protocols or level of modularity in the architecture) and a more granular differentiation of external tools that can be integrated into the FL pipeline could lead to different results. However, although we appreciate that such a finer approach might eventually change the distances of the current points between the elements in the lists, we would expect the main order to remain the same. This consideration arises when examining the definition of the current ranking. The success of the top two tools is mainly justified by the high scores obtained in the ''Usability'' and ''Portability'' factors outlined in Section V. This suggests that when tools have similar features with an equivalent level of maturity, the preference goes to the one with a lower entry barrier for users. Providing different documentation sources, tutorials, and access to multiple standard languages and tools may be critical for the community. As confirmed in the lower part of Table 3, low scoring for the worse-ranked tools might not necessarily be related to a lack of critical features, but rather to insufficient documentation that might have compromised exploitation. However, we noticed a discrepancy in Table 1 that led us to the following question: Why are tools with features comparable to the most popular ones, but with better documentation and more accessible entry points, not currently being considered at the same (or higher) level by the community?

Among the possible causes, we identified three main factors: participation in more significant international projects involving multiple institutions, tool adoption in various application fields, and dissemination and marketing activities by the respective engineering teams.

Although summarizing the results of the three tables might be difficult, we can say that if someone does not know where to start with FL, tools such as Flower [49] or PySyft [46] represent a good compromise between maturity and popularity for horizontal pipelines (HFL) either in data-centers or IoT devices. When Vertical FL (VFL) is required, OpenFL [50] can be the tool of choice. These recommendations are valid regardless of the application fields, as all the tools can support different models and data types. At the same time, we recognize that as future directions, more in-depth benchmarking with dedicated tools such as [38], LEAF [52], or FL-bench [77] may be needed to further understand the different peculiarities of each tool.

Another future goal is to revise the proposed criteria to account for these arguments and other factors to get closer to a comprehensive measure harmonizing the overall results.

VII. CONCLUSION

Several tools for implementing FL pipelines can accelerate research activities in this field. In this study, we provided a survey of all open-source solutions and two rankings based on the tools' popularity and readiness with the aim of guiding users (including non-experts) in adopting FL solutions, boosting their exploitation, and accelerating their research and development. One key aspect of this study is that tools primarily adopted by the community are not necessarily the most mature tools available. Owing to the three harvests (searches) performed over nearly one year (374 days), we could understand the growth rate of the majority of the tools. With all the data collected, we were able to provide clear recommendations to end-users on what tool to choose when starting a new journey in FL research.

A. COPYRIGHT FORM

Authors must submit an electronic IEEE Copyright Form (eCF) upon submitting their final manuscript files. You can access the eCF system through your manuscript submission system or through the Author Gateway. You are responsible for obtaining any necessary approvals and/or security clearances. For additional information on intellectual property rights, visit the IEEE Intellectual Property Rights department web page at http://www.ieee.org/publications_standards/publications/rights/index.html.

REFERENCES

[1] Q. Yang, Y. Liu, T. Chen, and Y. Tong, ''Federated machine learning: Concept and applications,'' *ACM Trans. Intell. Syst. Technol.*, vol. 10, no. 2, pp. 1–19, 2019.

[2] T. Li, A. K. Sahu, A. Talwalkar, and V. Smith, ''Federated learning: Challenges, methods, and future directions,'' *IEEE Signal Process. Mag.*, vol. 37, no. 3, pp. 50–60, May 2020.

[3] M. J. Sheller, B. Edwards, G. A. Reina, J. Martin, S. Pati, A. Kotrotsou, M. Milchenko, W. Xu, D. Marcus, R. R. Colen, and S. Bakas, ''Federated learning in medicine: Facilitating multi-institutional collaborations without sharing patient data,'' *Sci. Rep.*, vol. 10, no. 1, pp. 1–12, Jul. 2020.

[4] K. Hjerppe, J. Ruohonen, and V. Leppänen, ''The general data protection regulation: Requirements, architectures, and constraints,'' in *Proc. IEEE 27th Int. Requirements Eng. Conf. (RE)*, Jeju Island, South Korea, D. E. Damian, A. Perini, and S. Lee, Eds. Sep. 2019, pp. 265–275.

[5] (2019). *Website: Protection of Personal Information Act (Popi Act)—Popia*. [Online]. Available: https://popia.co.za

[6] S. T. Rosenbloom, J. R. L. Smith, R. Bowen, J. Burns, L. Riplinger, and T. H. Payne, ''Updating HIPAA for the electronic medical record era,'' *J. Amer. Med. Inform. Assoc.*, vol. 26, no. 10, pp. 1115–1119, Oct. 2019.

[7] L. Lyu, J. Yu, K. Nandakumar, Y. Li, X. Ma, J. Jin, H. Yu, and K. S. Ng, ''Towards fair and privacy-preserving federated deep models,'' *IEEE Trans. Parallel Distrib. Syst.*, vol. 31, no. 11, pp. 2524–2541, Nov. 2020.

[8] M. Y. Lu, R. J. Chen, D. Kong, J. Lipkova, R. Singh, D. F. K. Williamson, T. Y. Chen, and F. Mahmood, ''Federated learning for computational pathology on gigapixel whole slide images,'' *Med. Image Anal.*, vol. 76, Feb. 2022, Art. no. 102298.

[9] H. R. Roth et al., "Federated learning for breast density classification: A real-world implementation," in *Domain Adaptation and Representation Transfer, and Distributed and Collaborative Learning* (Lecture Notes in Computer Science), vol. 12444, S. Albarqouni, S. Bakas, K. Kamnitsas, M. J. Cardoso, B. A. Landman, W. Li, F. Milletari, N. Rieke, H. Roth, D. Xu, and Z. Xu, Eds. Lima, Peru: Springer, Oct. 2020, pp. 181–191.

[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 *Proc. IEEE 16th Int. Symp. Biomed. Imag. (ISBI)*, Venice, Italy, Apr. 2019, pp. 270–274.

[11] S. Abdulrahman, H. Tout, H. Ould-Slimane, A. Mourad, C. Talhi, and M. Guizani, "A survey on federated learning: The journey from centralized to distributed on-site learning and beyond," *IEEE Internet Things J.*, vol. 8, no. 7, pp. 5476–5497, Apr. 2021.

[12] M. J. Sheller, G. A. Reina, B. Edwards, J. Martin, and S. Bakas, "Multi-institutional deep learning modeling without sharing patient data: A feasibility study on brain tumor segmentation," in *Brainlesion: Glioma, Multiple Sclerosis, Stroke and Traumatic Brain Injuries*, (Lecture Notes in Computer Science), vol. 11383, A. Crimi, S. Bakas, H. J. Kuijf, F. Keyvan, M. Reyes, and T. van Walsum, Eds. Granada, Spain: Springer, 2018, pp. 92–104.

[13] N. Rieke, J. Hancox, W. Li, F. Milletarì, H. R. Roth, S. Albarqouni, S. Bakas, M. N. Galtier, B. A. Landman, K. Maier-Hein, S. Ourselin, M. Sheller, R. M. Summers, A. Trask, D. Xu, M. Baust, and M. J. Cardoso, "The future of digital health with federated learning," *NPJ Digit. Med.*, vol. 3, no. 1, pp. 1–7, Sep. 2020.

[14] D. Stripelis, J. L. Ambite, P. Lam, and P. Thompson, "Scaling neuroscience research using federated learning," in *Proc. IEEE 18th Int. Symp. Biomed. Imag. (ISBI)*, Nice, France, Apr. 2021, pp. 1191–1195.

[15] G. Long, Y. Tan, J. Jiang, and C. Zhang, "Federated learning for open banking," 2021, *arXiv:2108.10749*.

[16] J. So, K. Hsieh, B. Arzani, S. A. Noghabi, S. Avestimehr, and R. Chandra, "FedSpace: An efficient federated learning framework at satellites and ground stations," 2022, *arXiv:2202.01267*.

[17] I. Dayan, H. R. Roth, A. Zhong, A. Harouni, A. Gentili, A. Z. Abidin, A. Liu, A. B. Costa, B. J. Wood, and C.-S. Tsai, "Federated learning for predicting clinical outcomes in patients with COVID-19," *Nature Med.*, vol. 27, no. 10, pp. 1735–1743, 2021.

[18] Y. Chen, Y. Ning, M. Slawski, and H. Rangwala, "Asynchronous online federated learning for edge devices with non-IID data," in *Proc. IEEE Int. Conf. Big Data (Big Data)*, Atlanta, GA, USA, X. Wu, C. Jermaine, L. Xiong, X. Hu, O. Kotevska, S. Lu, W. Xu, S. Aluru, C. Zhai, E. Al-Masri, Z. Chen, and J. Saltz, Eds. Dec. 2020, pp. 15–24.

[19] A. Imteaj, U. Thakker, S. Wang, J. Li, and M. H. Amini, "A survey on federated learning for resource-constrained IoT devices," *IEEE Internet Things J.*, vol. 9, no. 1, pp. 1–24, Jan. 2022.

[20] Q. Wu, K. He, and X. Chen, "Personalized federated learning for intelligent IoT applications: A cloud-edge based framework," *IEEE Open J. Comput. Soc.*, vol. 1, pp. 35–44, 2020.

[21] T. Yang, G. Andrew, H. Eichner, H. Sun, W. Li, N. Kong, D. Ramage, and F. Beaufays, "Applied federated learning: Improving Google keyboard query suggestions," 2018, *arXiv:1812.02903*.

[22] S. Savazzi, M. Nicoli, and V. Rampa, "Federated learning with cooperating devices: A consensus approach for massive IoT networks," *IEEE Internet Things J.*, vol. 7, no. 5, pp. 4641–4654, May 2020.

[23] D. C. Nguyen, M. Ding, P. N. Pathirana, A. Seneviratne, J. Li, and H. V. Poor, "Federated learning for Internet of Things: A comprehensive survey," *IEEE Commun. Surveys Tuts.*, vol. 23, no. 3, pp. 1622–1658, 3rd Quart., 2021.

[24] K. Wei, J. Li, C. Ma, M. Ding, S. Wei, F. Wu, G. Chen, and T. Ranbaduge, "Vertical federated learning: Challenges, methodologies and experiments," 2022, *arXiv:2202.04309*.

[25] T. Chen, X. Jin, Y. Sun, and W. Yin, "VAFL: A method of vertical asynchronous federated learning," 2020, *arXiv:2007.06081*.

[26] P. Baldi and P. Sadowski, "A theory of local learning, the learning channel, and the optimality of backpropagation," *Neural Netw.*, vol. 83, pp. 51–74, Nov. 2016.

[27] P. J. Werbos, "Backpropagation through time: What it does and how to do it," *Proc. IEEE*, vol. 78, no. 10, pp. 1550–1560, 1990.

[28] X. Li, K. Huang, W. Yang, S. Wang, and Z. Zhang, "On the convergence of FedAvg on non-IID data," in *Proc. 8th Int. Conf. Learn. Represent. (ICLR)*, Addis Ababa, Ethiopia, Apr. 2020, pp. 1–26. [Online]. Available: https://OpenReview.net

[29] F. Sattler, S. Wiedemann, K. Müller, and W. Samek, "Robust and communication-efficient federated learning from non-IID data," 2019, *arXiv:1903.02891*.

[30] C. Briggs, Z. Fan, and P. Andras, "Federated learning with hierarchical clustering of local updates to improve training on non-IID data," in *Proc. Int. Joint Conf. Neural Netw. (IJCNN)*, Glasgow, U.K., Jul. 2020, pp. 1–9.

[31] K. A. Bonawitz, H. Eichner, W. Grieskamp, D. Huba, A. Ingerman, V. Ivanov, C. Kiddon, J. Konečný, S. Mazzocchi, B. McMahan, T. V. Overveldt, D. Petrou, D. Ramage, and J. Roselander, "Towards federated learning at scale: System design," in *Proc. Mach. Learn. Syst.*, Stanford, CA, USA, Apr. 2019, A. Talwalkar, V. Smith, and M. Zaharia, Eds. 2019, pp. 1–15. [Online]. Available: https://mlsys.org

[32] R. Shokri and V. Shmatikov, "Privacy-preserving deep learning," in *Proc. 53rd Annu. Allerton Conf. Commun., Control, Comput. (Allerton)*, Denver, CO, USA, I. Ray, N. Li, and C. Kruegel, Eds. Sep. 2015, pp. 909–910.

[33] K. Bonawitz, V. Ivanov, B. Kreuter, A. Marcedone, H. B. Mcmahan, S. Patel, D. Ramage, A. Segal, and K. Seth, "Practical secure aggregation for privacy-preserving machine learning," in *Proc. ACM SIGSAC Conf. Comput. Commun. Secur.*, Oct. 2017, pp. 1175–1191.

[34] E. Bagdasaryan, A. Veit, Y. Hua, D. Estrin, and V. Shmatikov, "How to backdoor federated learning," in *Proc. Int. Conf. Artif. Intell. Statist.*, 2020, pp. 2938–2948.

[35] Q. Yang, H. Matsutani, and M. Kondo, "A selective model aggregation approach in federated learning for online anomaly detection," in *Proc. Int. Conf. Internet Things (iThings) IEEE Green Comput. Commun. (GreenCom) IEEE Cyber, Phys. Social Comput. (CPSCom) IEEE Smart Data (SmartData) IEEE Congr. Cybermatics (Cybermatics)*, Nov. 2020, pp. 684–691.

[36] I. Kholod, E. Yanaki, D. Fomichev, E. Shalugin, E. Novikova, E. Filippov, and M. Nordlund, "Open-source federated learning frameworks for IoT: A comparative review and analysis," *Sensors*, vol. 21, no. 1, p. 167, Dec. 2020.

[37] D. Chen, "OpenFed: An open-source security and privacy guaranteed federated learning framework," 2021, *arXiv:2109.07852*.

[38] X. Liu, T. Shi, C. Xie, Q. Li, K. Hu, H. Kim, X. Xu, B. Li, and D. Song, "UniFed: A benchmark for federated learning frameworks," 2022, *arXiv:2207.10308*.

[39] J.-P. Salameh et al., "Preferred reporting items for systematic review and meta-analysis of diagnostic test accuracy studies (PRISMA-DTA): Explanation, elaboration, and checklist," *BMJ*, vol. 370, p. m2632, Aug. 2020.

[40] *Google Scholar*. Accessed: Mar. 28, 2022. [Online]. Available: https://scholar.google.com/

[41] *Semantic Scholar, a Free, AI-Powered Research Tool for Scientific Literature*. Accessed: Mar. 28, 2022. [Online]. Available: https://www.semanticscholar.org/

[42] S. Fricke, "Semantic scholar," *J. Med. Library Assoc.*, vol. 106, no. 1, p. 145, Jan. 2018.

[43] H. R. Roth et al., "NVIDIA FLARE: Federated learning from simulation to real-world," 2022, *arXiv:2210.13291*.

[44] *TensorFlow Federated*. Accessed: Sep. 28, 2022. [Online]. Available: https://www.tensorflow.org/federated

[45] H. Ludwig et al., "IBM federated learning: An enterprise framework white paper V0.1," 2020, *arXiv:2007.10987*.

[46] A. Ziller, A. Trask, A. Lopardo, B. Szymkow, B. Wagner, E. Bluemke, J.-M. Nounahon, J. Passerat-Palmbach, K. Prakash, N. Rose, T. Ryffel, Z. N. Reza, and G. Kaissis, "PySyft: A library for easy federated learning," in *Federated Learning Systems: Towards Next-Generation AI*. Cham, Switzerland: Springer, 2021, pp. 111–139.

[47] *Fate AI*. Accessed: Mar. 28, 2022. [Online]. Available: https://fate.fedai.org/

[48] 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," 2020, *arXiv:2007.13518*.

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

[50] G. A. Reina, A. Gruzdev, P. Foley, O. Perepelkina, M. Sharma, I. Davidyuk, I. Trushkin, M. Radionov, A. Mokrov, D. Agapov, J. Martin, B. Edwards, M. J. Sheller, S. Pati, P. N. Moorthy, H. S. Wang, P. Shah, and S. Bakas, "OpenFL: An open-source framework for federated learning," 2021, *arXiv:2105.06413*.

[51] D. Zeng, S. Liang, X. Hu, and Z. Xu, "FedLab: A flexible federated learning framework," 2021, *arXiv:2107.11621*.

[52] S. Caldas, P. Wu, T. Li, J. Konečný, H. B. McMahan, V. Smith, and A. Talwalkar, "LEAF: A benchmark for federated settings," 2018, *arXiv:1812.01097*.

[53] C. He, K. Balasubramanian, E. Ceyani, Y. Rong, P. Zhao, J. Huang, M. Annavaram, and S. Avestimehr, "Fedgraphnn: A federated learning system and benchmark for graph neural networks," 2021, *arXiv:2104.07145*.

[54] B. Liu, C. Tan, J. Wang, T. Zeng, H. Shan, H. Yao, H. Huang, P. Dai, L. Bo, and Y. Chen, "Fedlearn-Algo: A flexible open-source privacy-preserving machine learning platform," 2021, *arXiv:2107.04129*.

[55] J. H. Ro, A. T. Suresh, and K. Wu, "FedJAX: Federated learning simulation with JAX," 2021, *arXiv:2108.02117*.

[56] N. Angelou, A. Benaissa, B. Cebere, W. Clark, A. J. Hall, M. A. Hoeh, D. Liu, P. Papadopoulos, R. Roehm, R. Sandmann, P. Schoppmann, and T. Titcombe, "Asymmetric private set intersection with applications to contact tracing and private vertical federated machine learning," 2020, *arXiv:2011.09350*.

[57] D. Romanini, A. J. Hall, P. Papadopoulos, T. Titcombe, A. Ismail, T. Cebere, R. Sandmann, R. Roehm, and M. A. Hoeh, "PyVertical: A vertical federated learning framework for multi-headed SplitNN," 2021, *arXiv:2104.00489*.

[58] G. Kaissis, A. Ziller, J. Passerat-Palmbach, T. Ryffel, D. Usynin, A. Trask, I. Lima, J. Mancuso, F. Jungmann, M.-M. Steinborn, A. Saleh, M. Makowski, D. Rueckert, and R. Braren, "End-to-end privacy preserving deep learning on multi-institutional medical imaging," *Nature Mach. Intell.*, vol. 3, no. 6, pp. 473–484, May 2021.

[59] M. N. Galtier and C. Marini, "Substra: A framework for privacy-preserving, traceable and collaborative machine learning," 2019, *arXiv:1910.11567*.

[60] M. Ekmefjord, A. Ait-Mlouk, S. Alawadi, M. Åkesson, P. Singh, O. Spjuth, S. Toor, and A. Hellander, "Scalable federated machine learning with FEDn," in *Proc. 22nd IEEE Int. Symp. Cluster, Cloud Internet Comput. (CCGrid)*, Taormina, Italy, May 2022, pp. 555–564.

[61] 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 Distributed and Collaborative Learning* (Lecture Notes in Computer Science), vol. 12444, S. Albarqouni, S. Bakas, K. Kamnitsas, M. J. Cardoso, B. A. Landman, W. Li, F. Milletari, N. Rieke, H. Roth, D. Xu, and Z. Xu, Eds. Lima, Peru: Springer, 2020, pp. 201–210.

[62] M. Ryu, Y. Kim, K. Kim, and R. K. Madduri, "APPFL: Open-source software framework for privacy-preserving federated learning," in *Proc. IEEE Int. Parallel Distrib. Process. Symp. Workshops (IPDPSW)*, Lyon, France, May 2022, pp. 1074–1083.

[63] R. Nasirigerdeh, R. Torkzadehmahani, J. O. Matschinske, J. Baumbach, D. Rueckert, and G. Kaissis, "HyFed: A hybrid federated framework for privacy-preserving machine learning," 2021, *arXiv:2105.10545*.

[64] Y. Feng, M. Yu, W. Xiong, X. Guo, J. Huang, S. Chang, M. Campbell, M. A. Greenspan, and X. Zhu, "PyFed: Extending PySyft with N-IID federated learning benchmark," in *Proc. 34th Can. Conf. Artif. Intell.*, L. Antonie and P. M. Zadeh, Eds. Ottawa, AI, Canadian: Canadian Artificial Intelligence Association, 2021, pp. 1–6.

[65] H. Cao, Y. Zhang, J. Baumbach, P. Burton, D. Dwyer, N. Koutsouleris, J. Matschinske, Y. Marcon, S. Rajan, T. Rieg, P. Ryser-Welch, J. Späth, E. Schwarz, D. Alnæs, O. Andreasssen, J. Chen, F. Degenhardt, D. Doncevic, R. Eils, and A. Meyer-Lindenberg, "dsMTL—A computational framework for privacy-preserving, distributed multi-task machine learning," *Bioinformatics*, vol. 38, pp. 4919–4926, Sep. 2022.

[66] P. Niedziela, A. Danilenka, D. Kolasa, M. Ganzha, M. Paprzycki, and K. Nalinaksh, "Sunday-FL–developing open source platform for federated learning," in *Proc. Emerg. Trends Ind. 4.0 (ETI 4.0)*, May 2021, pp. 1–6.

[67] F. Morsbach and S. Toor, "DecFL: An ubiquitous decentralized model training protocol and framework empowered by blockchain," in *Proc. 3rd ACM Int. Symp. Blockchain Secure Crit. Infrastructure*, Hong Kong, K. Gai and K. R. Choo, Eds. May 2021, pp. 61–70.

[68] C. Schneebeli, S. Kalloori, and S. Klingler, "A practical federated learning framework for small number of stakeholders," in *Proc. 14th ACM Int. Conf. Web Search Data Mining*, Israel, L. Lewin-Eytan, D. Carmel, E. Yom-Tov, E. Agichtein, and E. Gabrilovich, Eds. Mar. 2021, pp. 910–913.

[69] A. Moncada-Torres, F. Martin, M. Sieswerda, J. van Soest, and G. Geleijnse, "VANTAGE6: An open source privacy preserving federated learning infrastructure for secure insight exchange," in *Proc. Amer. Med. Inform. Assoc. Annu. Symp.*, Nov. 2020, pp. 1–9.

[70] N. Rodríguez-Barroso, G. Stipcich, D. Jiménez-López, J. A. Ruiz-Millán, E. Martínez-Cámara, G. González-Seco, M. V. Luzón, M. A. Veganzones, and F. Herrera, "Federated learning and differential privacy: Software tools analysis, the Sherpa.Ai FL framework and methodological guidelines for preserving data privacy," *Inf. Fusion*, vol. 64, pp. 270–292, Dec. 2020.

[71] K. Burlachenko, S. Horváth, and P. Richtárik, "Fl_Pytorch: Optimization research simulator for federated learning," 2022, *arXiv:2202.03099*.

[72] T. Hunt, C. Song, R. Shokri, V. Shmatikov, and E. Witchel, "Chiron: Privacy-preserving machine learning as a service," 2018, *arXiv:1803.05961*.

[73] Y. Chen, X. Qin, J. Wang, C. Yu, and W. Gao, "FedHealth: A federated transfer learning framework for wearable healthcare," *IEEE Intell. Syst.*, vol. 35, no. 4, pp. 83–93, Jul. 2020.

[74] Y. Yue, Z. Ming, Q. Zhijie, L. Lei, and C. Hong, "A data protection-oriented design procedure for a federated learning framework," in *Proc. Int. Conf. Wireless Commun. Signal Process. (WCSP)*, Nanjing, China, Oct. 2020, pp. 968–974.

[75] S. Carpov, N. Gama, M. Georgieva, and D. Jetchev, "GenoPPML—A framework for genomic privacy-preserving machine learning," in *Proc. IEEE 15th Int. Conf. Cloud Comput. (CLOUD)*, Barcelona, Spain, C. N. Chang, E. Damiani, G. B. Dasgupta, F. Gagliardi, C. Hagleitner, D. S. Milojicic, T. M. H. Trong, R. Ward, F. Xhafa, and J. Zhang, Eds. Jul. 2022, pp. 532–542.

[76] Z. Zhao, R. Birke, A. Kunar, and L. Y. Chen, "Fed-TGAN: Federated learning framework for synthesizing tabular data," 2021, *arXiv:2108.07927*.

[77] Y. Liang, Y. Guo, Y. Gong, C. Luo, J. Zhan, and Y. Huang, "FLbench: A benchmark suite for federated learning," in *Proc. BenchCouncil Int. Federated Intell. Comput. Block Chain Conf.* Cham, Switzerland: Springer, 2020, pp. 166–176.

[78] C. Pappas, D. Chatzopoulos, S. Lalis, and M. Vavalis, "IPLS: A framework for decentralized federated learning," in *Proc. IFIP Netw. Conf. (IFIP Networking)*, Helsinki, Finland, Z. Yan, G. Tyson, and D. Koutsonikolas, Eds. Jun. 2021, pp. 1–6.

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

[80] Z. Jiang, W. Wang, B. Li, and B. Li, "Pisces: Efficient federated learning via guided asynchronous training," in *Proc. 13th Symp. Cloud Comput.*, San Francisco, CA, USA, A. Gavrilovska, D. Altinbüken, and C. Binnig, Eds. Nov. 2022, pp. 370–385.

[81] H. Wang, Y. Zhou, C. Zhang, C. Peng, M. Huang, Y. Liu, and L. Zhang, "XFL: A high performace, lightweighted federated learning framework," 2023, *arXiv:2302.05076*.

[82] T. Kluyver, B. Ragan-Kelley, F. Pérez, B. Granger, M. Bussonnier, J. Frederic, K. Kelley, J. Hamrick, J. Grout, S. Corlay, P. Ivanov, D. Avila, S. Abdalla, and C. Willing, "Jupyter notebooks—A publishing format for reproducible computational workflows," in *Positioning and Power in Academic Publishing: Players, Agents and Agendas*, F. Loizides and B. Schmidt, eds. Amsterdam, The Netherlands: IOS Press, 2016, pp. 87–90.

[83] T. Carneiro, R. V. M. Da NóBrega, T. Nepomuceno, G.-B. Bian, V. H. C. De Albuquerque, and P. P. R. Filho, "Performance analysis of Google colaboratory as a tool for accelerating deep learning applications," *IEEE Access*, vol. 6, pp. 61677–61685, 2018.

WALTER RIVIERA received the B.Sc. degree in multimedia information technology and the M.Sc. degree in visual computing and pattern recognition from the University of Verona, Italy, in 2012 and 2014, respectively. He is currently pursuing the Ph.D. degree in federated learning and distributed systems. He is with Technical Europe, Middle-East, and Africa (EMEA) and a lead for AI with Intel Corporation. He then won two research grants focused on information retrieval and data mining techniques. He mainly focuses on applied research at a different level: from testing novel scientific approaches to tackling production and deployment challenges. His research interests include medical environment but eventually includes collaborations on many other segments.

ILARIA BOSCOLO GALAZZO (Member, IEEE) received the degree (cum laude) in biomedical engineering from the University of Padova, in 2010, and the Ph.D. degree in neuroscience from the University of Verona, Italy, in May 2014.

She took a position as a Research Associate with the Institute of Nuclear Medicine, UCL, London, from 2014 to 2016, and then with the Department of Computer Science, University of Verona, from 2016 to July 2020, where she is currently a temporary Assistant Professor of bioengineering with the Department of Engineering for Innovation Medicine, within the BraiNAVLab. Her research interests include imaging genetics, modeling of functional MRI data, brain connectivity, and multimodal neuroimaging data integration relying on classical, and AI-based methods. She is a member of the IEEE Bioimaging and Signal Processing (BISP) Technical Committee. She is an Associate Editor of IEEE Access.

GLORIA MENEGAZ (Senior Member, IEEE) received the M.Sc. degree in electronic engineering and the master's degree in information technology from Politecnico di Milano, Italy, in 1993 and 1995, respectively, and the Ph.D. degree in applied sciences from EPFL, in 2000.

From 2000 to 2002, she was a first Research Assistant with the Telecommunications Laboratory (LCAV), EPFL, then she left to join the Computer Science Department, University of Fribourg, Switzerland, as an Assistant Professor. She joined the Department of Information Engineering, University of Siena, Italy, as an Assistant Professor. In 2007, she joined the Department of Computer Science, University of Verona, Italy, as an Associate Professor and became a Full Professor of bioengineering (ING-INF/06), in 2021. She is currently a Professor of bioengineering with the Department of Engineering for Innovation Medicine, University of Verona, where she leads the BraiNAVLab. Her research interests include neuroimaging and imaging genetics, including, brain connectivity, microstructure modeling and numerical biomarker discovery relying on classical and AI (statistical/machine/deep learning) focusing on explainable methods.

Dr. Menegaz is a member of the Ad Hoc Group (AHG) on Digital Media Storage using DNA (JPEG-DNA) and the MICCAI Special Interest Group on Biomedical Image Analysis Challenges (SIG-BIAC) and a Observer for the DICOM WG-32 (Neurophysiology data). She is also a member of the Signal Processing Society (SPS) and a Liason representative of the IEEE-SPS at the IEEE Transactions on Medical Imaging. In 2004, she was awarded the Rita Levi Montalcini Grant by the Italian Ministry of University and Research (MIUR) (DM 20/03/2003 n. 501). She is the Co-Chair of the TC on Challenges and Data Collections of the IEEE Bioimaging and Signal Processing (BISP).

Open Access funding provided by 'Università degli Studi di Verona' within the CRUI CARE Agreement