FeLebrities: a user-centric assessment of Federated Learning frameworks

Walter Riviera, *Member, IEEE,* Ilaria Boscolo Galazzo, *Member, IEEE,* Gloria Menegaz, *Senior Member, IEEE*

Abstract—Federated Learning (FL) is a new paradigm that aims at solving the data access problem. It is gaining an increasing interest in a variety of research fields, including the Biomedical and Financial environments, where lots of valuable data sources are available but not often directly accessible due to the regulations that protect sensitive information. FL provides a wayout enabling the processing and sharing of data modeling solutions moving the focus from data to models. The FL paradigm involves different entities (institutions) holding proprietary datasets, contributing with each other to locally train a copy of a shared Artificial Intelligence (AI) model. Although there are different studies in the literature that suggest how to conceptually implement and orchestrate a federation, fewer efforts have been made on practical implications. With the ambition of helping accelerating the exploitation of FL frameworks, this paper proposes a survey of public tools that are currently available, an objective ranking based on current state of user preferences and the assessment of the growth trend of the tool popularity over a six months time window. Finally, a ranking of the tools maturity is derived based on key aspects to consider when building a FL pipeline.

Index Terms—Federated Learning tools, Distributed systems, AI at scale.

1 Introduction

Federated learning (FL) is the paradigm that aims at solving the data access problem. In the Artificial Intelligence (AI) domain, data represent the starting point for many research and development activities. With a rising attention given to the field, data have also grown in demand and appreciation, redefining the list of priorities in designing and building solutions for real world applications. A clear demonstration of this growing importance is represented by the creation of dedicated laws - such as General Data Privacy Regulations (GDPR) [1] in place in the European Union or the Health Insurance Portability and Accountability Act (HIPAA) [1] in USA, which provide guidelines and constraints to be respected in order to access data, and the Protection of Personal Information Act (POPIA) [2]. From the AI perspective, this reflects into the need of finding ways to access data for advancing the State of the Art (SOA) in a given environment, while being fully compliant with the regulations in place. FL is an effective way to satisfy all those requirements as in a federation of collaborating institutions what is shared is a common model, trained on each single collaborator using local data. Historically, the approach of training AI models would assume 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 [3]) centers are a great examples, as illustrated in Figure 1. Another consequence of the rising importance of data, certified also by the dedicated regulations being created around the globe (GDPR [1], HIPAA [4] and POPIA [5]) priorities have been redefined up to the point where they are impacting the way infrastructures are built. Indeed, in a FL setting, data are expected to stay in the same location where they were collected, while the AI model is shared across all the institutions taking part to a federation. An generic example is provided in Figure 2.

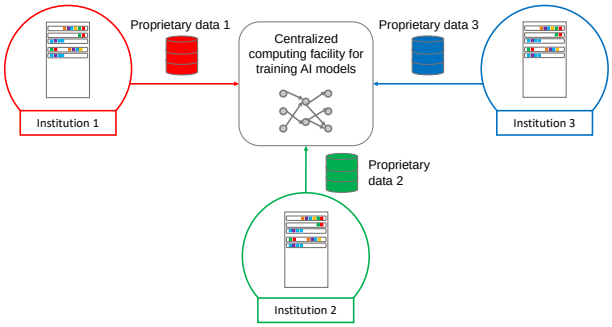

Fig. 1. Data-to-model: example of legacy approach where data would move to a centralized training facility. Here the AI model is represented as a graph or neural network.

Research community has already started investigating on this emerging topic either for its privacy compliant aspects [2] [3] as well as a viable tool for addressing AI challenges also in critical domains such as the biomedical context [4] [5] [6]. Despite the domain being still relatively

- *W. Riviera, I. Boscolo Galazzo and G. Menegaz are with the Department of Computer Science, University of Verona, Verona, Italy. E-mail: walter.riviera, gloria.menegaz, ilaria.boscologalazzo@univr.it*
- *W. Riviera is with Intel Corporation. E-mail: walter.riviera@intel.com*

1. https://www.cdc.gov/phlp/publications/topic/hipaa.html
2. https://popia.co.za/
3. https://www.top500.org/
4. https://www.cdc.gov/phlp/publications/topic/hipaa.html
5. https://popia.co.za/

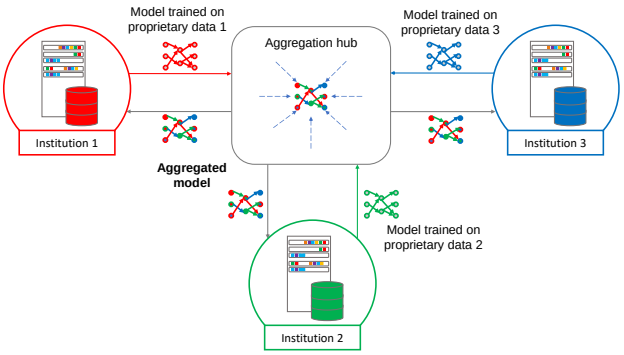

Fig. 2. Model-to-data: example of a federated approach. A central unit called aggregator would clone and distribute copies of the same model to each of the collaborating institutions. Each one of them would then train its own copy of the model using local datasets before sharing it back to the aggregator.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

new, the literature can already provide helpful surveys on how the concept works and complies with privacy aspects [2], how it can be transferred in the *Internet-of-things (IOT)* world [7] and what are the steps to implement it from a protocol, software and hardware standpoint [7]. The rising interest in the research community and industries *R&D* departments have enriched the literature which has in-turn influenced the development and evolution of a multitude of new tools for implementing FL pipelines. If from one perspective this aspect is encouraging, on the other is reflecting the need to get clear indications about what tools are currently available, which one 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 tools, including growth trends;
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 ability to match needs highlighted in the previous point.

We truly 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 the tool providers identifying ways to improve their products. The availability of a ranking of the FL tools will also boost their exploitation for production environment, where such tools are still largely unexplored.

1.1 Paper organization

This paper is composed of six Sections. In section II, we discuss the SOA for FL implementations. Section III focuses on the list of tools currently available to the community, sharing a high-level overview of their popularity and level of adoption. Section IV will augment the retrieved list of tools with the current state of adoption, including the growth trend observed over a six months period, and Section V discusses the key aspects that would need to be considered when implementing federated environments for research purposes. These factors are then translated into requirements that FL tools would need to satisfy for successful exploitation and lead to a ranking presented in the form of a table. Ultimately, we draw some conclusions and share future directions in Section VI.

2 State of the art

FL is a distributed machine learning (ML) approach that enables organizations to collaborate on projects without sharing sensitive data [8], such as patient records [9], [10] or financial data [11], or not easily accessible data, like the ones stored in remote locations as satellites or space stations from high resolution sensors [12]. The basic premise behind FL [13] [14] is that the model `moves` to meet the data rather than the data `moving` to meet the model. Therefore, the minimum data movement needed across the federation is solely the model parameters and their updates.

2.1 FL settings

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

In the synchronous scenario the idea is to have a central unit, often identified as aggregator [8] [15] [9], acting as central pivot and determining when to start a new iteration. The aggregator would in-fact be responsible of cloning the initial model over to each collaborating institution, waiting to receive the locally trained copies coming back, and finally merging them together as the name suggests. This type of FL pipeline is usually implemented among big data centers, like the ones involved in medical environments [16] [17]. Data centers are capable of storing vast amounts of data as well as of providing the required computational power to process them. On top of this, big computing infrastructures like 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 the data centers towards the edge devices, new challenges arise due to the high variance in products and manufacturers. Devices with different latency, working frequency and hardware features can indeed lead to different computation times [] [18]. These are some of the reasons motivating the need to have an asynchronous FL pipeline. In this scenario, each collaborating institution can share its own update at any time either to a unique aggregator [19], [20], [21] or to the other participants in a "all-to-all" setup [22], [23].

An additional critical point to address is the difference between Horizontal and Vertical FL. To understand this difference we need to consider the space of the features and

the type of model. In the examples shared so far, we were implicitly referring to the Horizontal FL where the different collaborators have different data but contribute to the federation by sharing the feature space and by training the same model. In the Vertical FL, each collaborator is expected to contribute by providing different bits of information of the same samples, leading to a scenario where the feature space accessed by each single collaborator might be different from the others. Because of this, in the Vertical configuration, chances are that each collaborator is training a different model. The aggregation in this case is represented by the interoperability between collaborators, where in order to update a model, information coming from the model of another collaborator might be required [24], [25].

2.2 FL challenges

Regardless to what FL setting (Synchronous or Asynchronous) or configuration (Horizontal or Vertical) is adopted by a given federation, there are three main areas being currently addressed by the research community:

1) Aggregation functions and model convergence given different data distributions;
2) Privacy aspects and ways to build a secure FL pipeline for protecting IP during the 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 [26] [27]. Data distribution can severely impact the training performance by affecting the total accuracy [28], the convergence capability, the authentication processes (especially in case of different devices) and the speed of the process intended as total time-to-train [29]. In a nutshell, under this setting, the performance of the training process may vary significantly according to the unbalance of local data samples as well as the particular statistical distribution of the training examples (i.e., features and labels) stored at the local nodes [14].

In the past years institutions have introduced FL deployments to answer the need of training AI models for the healthcare and financial sectors in a setting that would allow them greater access to larger and more diverse datasets without violating privacy laws [30] [31], such as HIPAA and GDPR [1]. While on one side FL has been designed with security in mind [28], the set-up is just the beginning. To ensure a protected execution there are also other aspects that would need to be considered, some of which are still representing open questions in the research field [32]. Among these are finding a consolidated method to guarantee secure execution (encryption, key exchange, 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 machine/institutions are taking part in a federation, the more important is the ability to scale. As mentioned in the previous Section, to the best of our knowledge, a consolidated way for detecting "poor" training contributions (coming from institutions with corrupted or redundant data) is still missing, and aggregation functions are currently being evaluated by the research community [26] [31] [33]. Another implication when talking about big scales is represented by the infrastructure and the connectivity chosen by the institutions for communication [14].

Several works have proposed surveys to illustrate the advancement in the field [2] [7], however, to the best of our knowledge, no one is providing a ranked list based on ad-hoc quality assessment criteria of all the (possible) tools available to the community to actually implement FL experiments. In [34] a comparison of five tools is provided, some of which accessible through a licensed service, without providing clarifications on why or how those tools were precisely selected. Another work [35] provides an interesting comparison table, but the main focus of that 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 was not clear why and how the discussed tools have been selected. On the same line [36] proposes a complete benchmarking suite with a helpful decision tree to help users choosing a tool based on their needs. Their recommended ranking is also including some of the evaluation metrics proposed in this work, with even a deeper level of details. However, while we believe in the value of such approach, the breadth of the offer in terms of tools that can be chosen might represent a constraint for the end users. In-fact, [36] centers its evaluation around nine tools, but the criteria for which those tools were identified and selected are not clear. As we discovered in this work, the list of open-source FL tools can go beyond 30 and is interesting to note how the most popular tool to-date was not considered in their decision tree.

3 Federated Learning tools

3.1 Methods and premises

The ambition of this article is to provide an inclusive and informative list of the current FL tools available to the community for implementing research pipelines in any environment where accessing distributed data is a challenge. To have better understanding of the current scenario, we performed two literature searches: one carried out on March 28th the other on September 28th. In both cases, the activity was inspired by the Preferred Reporting Items for Systematic Reviews and Meta-Analyses (PRISMA) guidelines. More specifically, we decided to follow the *Preferred reporting items for systematic review and meta-analysis of diagnostic test accuracy studies (PRISMA-DTA): explanation, elaboration, and checklist* [37]. In particular, the guidelines we followed are a selection of the ones described in the *PRISMA 2020 checklist*, accessible on the official PRISMA website: http://www.prisma-statement.org/. Below, a detailed list of the items outlined in the guidelines that we have identified as applicable to this collection:

- Items 5, 6, 7 and 8 have been considered for building this Section;
- Items 13a and 13d, have been followed to build the comparison table in Section IV;

- Items 16a, 16b and 23c, have been finally used to structure the results discussion provided in Section V.

3.2 Exploring tools

To build the list of tools in a objective way, we performed two different reviews (harvests), with roughly 6-months (184 days) as time gap. Capturing the tool lists in one survey would have been enough to draw a general overview about the current environment at that time, but having two observational points for each of the tools was useful to better understand the level of commitment from the engineering team, and the maturity and popularity growth rate beyond each tool. The collection methods were exactly the same and are described below. We relied on three different search engines: Google Scholar[6], Semantic Scholar[7] and standard Google website[8].

For the first two, we developed a script to automatically query the search engines with a collection of keywords on the topic. We built such 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 concept: 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 rich and inclusive search of all the relevant articles and works in the domain.

Google Scholar was helpful to capture all the related works where a given keyword (or part of it) was mentioned in the paper and not only in the title. We could identify a cumulative list of 420 related articles, of which 217 were unique. Despite the encouraging numbers, we soon encountered a bottleneck represented by the API service. For each keyword the system would only return maximum 20 results. Furthermore, after an undefined but limited amount of searches, a CAPTCHA request would rise, blocking any automatic API interrogation. This is done by the service provider to avoid free unlimited search performed by automated systems and to ensure enough resources for manual searches performed by real users. This is why those numbers refers to the article harvest performed in March only. During the September harvest we could not run any new queries using this engine.

In order to build a more robust and consistent set of related works, we then leveraged the Semantic Scholar service [38]. As defined on their website, *"Semantic Scholar is a free, AI-powered research tool for scientific literature, based at the Allen Institute for AI."* By using this tool we were able to increase the numbers of results obtained for each keyword

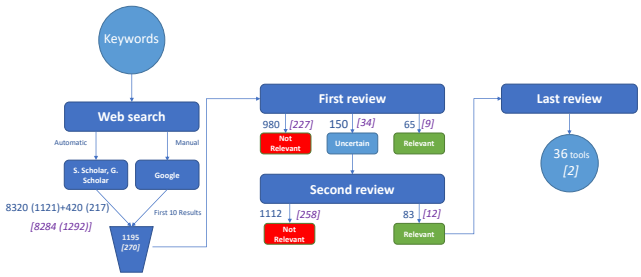

Fig. 3. Collection pipeline implemented for the harvests. On each arrow are outlined the number of articles retrieved and filtered. Numbers between squared brackets refer to the outcome of the September harvest. The output numbers of the second review are obtained by summing the numbers of a given category coming from the first review with the respective category of the second review. Please note that despite the September harvest returned 1292 articles, only 270 were actual new findings.

search up to 100 and access more accurate content given the semantic nature of the search engine. Indeed, thanks to this tool we could add a layer of depth in our searches, by looking at the different ways of sorting the results. The website (and its API) allows users to perform queries and sort the outcome according to four different metrics: by "Relevance", "Citations-count", "Most Influential Paper" and by "Recency". Among these options, a user could also select articles based on where they were published (e.g, conferences, journals, books) and the field of studies and applications (e.g., Medicine, Geology, Economics). For our collection, we did not leverage any of these options as we wanted to be as much inclusive as we could possibly be.

By repeating the research of all the keywords for all the four sorting types mentioned above, we obtained a cumulative list of 8320 articles, of which 1121 were unique during the March harvest and a cumulative list of 8284 articles of which 1292 were unique, during the September harvest. The amount of new articles retrieved in September which were not available yet in March is 270.

The fact that among the Google Scholar search results we obtained 51% of unique contributions versus the 14% of the ones identified through Semantic Scholar is actually an indicator of the goodness of the research. Through Semantic Scholar we were not only able to find more related papers but also consistent articles, that we interpreted as a reliable capability of the tool to capture the semantic aspects of the research.

To make sure we would capture all the relevant FL tools not yet described in a published paper, we also decided to perform a manual search on the standard Google search engine. To do so, we evaluated the first 10 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 (Clara) [9],

6. https://scholar.google.com/
7. https://www.semanticscholar.org/
8. https://www.google.com
9. https://blogs.nvidia.com/blog/2021/11/29/federated-learning-ai-nvidia-flare/

Tensorflow Federated [10] and IBM Federated [11].

Once obtained the three lists of unique titles described above, we finally merged them together, resulting in a total of 1195 unique articles discovered in March and a list of 1292 retrieved in September. We then started pruning results by manually reviewing and labelling the list in three different buckets: "relevant"(R), "non-relevant" (NR) and "uncertain" (TBD).

Articles completely unrelated to the topic (e.g., work mentioning ML methods or collaborative learning platforms for schools) where discarded from the whole collection. At the end of the first labelling cycle we ended up with 65 R, 980 NR and 150 TBD for the March harvest and 9 R, 227 NR and 34 TBD for the September harvest.

An example of articles being captured by the three categories is available in figure 4.

The "uncertain" category, required us a deeper review of the work. All the articles in this list went through a second round of labelling. This time, the objective was to review the 150 papers belonging to the TBD list, with the aim of allocating them either to the R or NR. At the end of this labelling round, the cardinality of the classes was: 83 R and 1112 NR for the March harvest and 12 R and 258 NR for the September harvest.

A summary of the research pipeline adopted 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. In general, for both the harvests, lots of articles were using the world "framework" to suggest methods and approaches to address specific FL tasks, but were not proposing toolkits or open-source products that could be leveraged by the community to implement FL pipelines. This final review allowed us to identify 36 suitable tools during the March harvest, and additional 2 tools during September harvest. The full list of tools retrieved in March with the indicators from the Github and Gitlab repositories is reported in Table 1.

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

4 Tools Popularity and Level of Adoption

After retrieving the list of tools, our goal was to understand what was the level of popularity ad adoption of each tools from the community perspective. Each Git repository has public indicators like the number of Watch (W), Fork (F) and Stars(S). The Watch indicator can capture the number of users actively watching the repository. These users will receive updates when new actions are taken on the repository. The number of Fork indicates the amount of times a repository has been forked. Such number can be a good indicator of how many interested users might be developing code to extend the tool. Finally, the number of Stars indicates the amount of likes that the repository has received. This final indicator might be less accurate in capturing actual users, but can give a good estimation of the reach, in terms of how many people have seen the tool at least once. For practicality we wanted to combine these tree aspects into one consolidated score that we could use for providing a *popularity* driven ranking of the tools. Since the popularity would also depend on the time that a given repository was made available to the community, we wanted to normalize all the values with a timing factor. This was done to make sure that newer repository with less time for exposure to the community would not be affected by getting a low score. To achieve this goal, we combined these tree factors into a consolidated Score calculated as follow:

$$Score_{\text{t}} = \frac{\frac{1}{3}W_t + F_t + S_t}{ETA_{\text{fr}}}$$

where ETA_{fr} is the time elapsed between the day of when the first release of the tool was published and the harvest date. Where the date of first release was not available, the date of the first commit was used for the computation. The Scores associated to the tools retrieved in March harvest, can be found in Table 1.

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

We realized that in order to access this information we should have observed the list of tools over a time window, to check which tools were being considered by the community at higher pace. Thanks to the second harvest (September

10. https://www.tensorflow.org/federated
11. https://ibmfl.mybluemix.net/

	TOOL	Watch	Fork	Star	ETA (days)	SCORES
1	PySyft [39]	211	1800	8000	1714	1.95
2	FATE https://fate.fedai.org/	134	1200	4100	956	1.89
3	FedML [40]	37	331	1100	614	0.80
4	Tensorflow Federated https://www.tensorflow.org/federated	66	447	1800	1301	0.59
5	Flower [41]	20	15	825	770	0.37
6	OpenFL [42]	10	76	286	437	0.28
7	IBM-Federated [43]	20	85	292	647	0.20
8	FedLab [44]	6	30	180	383	0.19
9	LEAF [45]	18	187	470	1249	0.18
10	FedGraphNN [46]	9	33	147	383	0.16
11	Fedlearn-algo [47]	8	34	80	255	0.16
12	FedJAX [48]	11	28	172	461	0.15
13	PyVertical [49] [50]	12	35	101	661	0.07
14	PriMIA [51]	8	18	102	707	0.06
15	Substra [52]	8	24	143	1252	0.05
16	Fedn [53]	7	20	59	622	0.05
17	FedBioMed (GitLab) [54]	NA	23	3	332	0.04
18	OpenFED [35]	2	1	17	300	0.02
19	APPFL [55]	2	1	7	172	0.02
20	HyFed [56]	4	3	9	361	0.01
21	PyFed [57]	3	2	5	544	0.01
22	dsMTL [58]	2	2	0	266	0.01
23	Sunday FL [59]	1	4	2	524	0.00
24	DecFL [60]	2	2	3	717	0.00
25	MTC-ETH [61]	1	1	2	881	0.00
26	Vantage6 [62]	5	2	0	1835	0.00
27	Sherpa ai [63]	3	0	0	0	-
28	FL-Pytorch (Pytorch Federated) [64]	NA	NA	NA	NA	NA
29	Chiron [65]	NA	NA	NA	NA	NA
30	FedHealth [66]	NA	NA	NA	NA	NA
31	FAE [67]	NA	NA	NA	NA	NA
32	GENO [68]	NA	NA	NA	NA	NA
33	FedTGan [69]	NA	NA	NA	NA	NA
34	FL-Bench [70]	NA	NA	NA	NA	NA
35	IPLS [71]	NA	NA	NA	NA	NA
36	Nvidia-Clara https://docs.nvidia.com/clara/	NA	NA	NA	NA	NA

TABLE 1
Tool popularity table, March harvest. 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 not available or not publicly accessible. If not specified otherwise, all the repositories are Github projects.

28th) we were able to discover new tools to add to the list, and update the values of W, F and S for each of the tools discovered in March. Knowing the differences between the indicators value in March and in September, we computed the growth rate for each of the tools as follow:

$$gr_{\mathrm{t}} = \frac{\frac{1}{3}((W_t + F_t + S_t)^{March} - (W_t + F_t + S_t)^{September})}{ETA}$$

where ETA in this case corresponds to 184 days. The outcome of this computation can be appreciated in Figure 5.

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

5 Priorities in FL tools for Research

Previous Sections illustrated how we could retrieve a list of relevant tools for building FL pipelines and which ones seem to be preferred by the community. In this Section, our goal is to provide a new ranking of the tools in order to

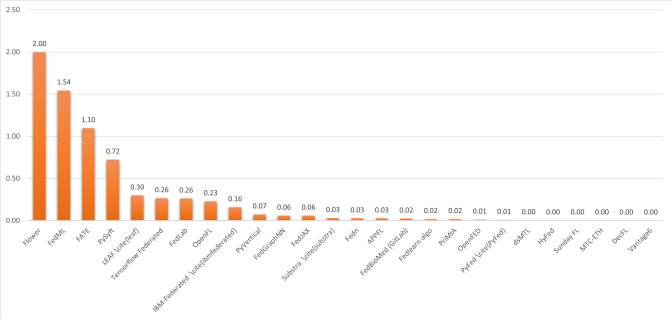

Fig. 5. Popularity growth rate: this graph illustrates which tools have been gaining more popularity by the community, over an observation window of 184 days (roughly 6 months). Tools that did not have a repository available in March were excluded from this chart.

identify the most mature ones. For this part of the study, we will focus on the specific features of each tool, regardless of any popularity aspects as outlined in the previous Sections.

The task consists in retrieving and evaluating the FL tools that could be adopted to boost 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.

As described in the SOA Section, there are several ways of implementing FL. From bridging different data-center institutions together at production level to leveraging the agile nature of IoT devices, FL pipelines have to be shaped according to the needs, goal and constraints to consider. However, as previously outlined in the FL settings section, since our main ambition here is to identify the most mature FL tools suitable for supporting 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.

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

Indeed, those aspects are key because they can directly reflect into requirements that FL tools would need to fulfil in order to be considered. For example, all of them are highlighting the need of having a flexible and modular architecture to allow maximum customization for research, either for aggregation functions, communication protocols or privacy preserving and security features. Another practical insight that can be derived by items 2 and 3 concerns the ability of a tool to scale-out on multiple computing machines. As demonstrated in [12], the applicability of FL is not only related to the need of accessing data being respectful of current regulations, it can also offer access to data which are not easily retrievable, like the ones on a satellite or space stations. Furthermore, the Medical [16] or Geo-spatial [12] environments are usually sources of high resolution data acquired by machines manufactured by different companies and this could map in the need of having dedicated pre-processing routines in order to be used to feed an AI model. Being able to test an FL approach on multiple machines hosting different datasets (generated by different equipments) instead of relying on simulations of multiple instances hosted on the same computing node, can surely enhance the reliability of the conclusions driven when investigating privacy and communication efficiency aspects.

In addition to what outlined so far, other practical considerations related to the research environment might apply. The easier the path to results in research environment is, the fastest could then be the path to 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 (like Python) and AI platforms (like PyTorch or Tensorflow to mention two), or having access to direct support channels or useful documentation, can represent critical aspects for simplifying research activities in different domains.

5.1 Evaluation metrics

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

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
   - 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 has been populated with information retrieved from the resources publicly available for each tool. As one can easily see, there is a mismatch between the tools listed in Table 1 and the ones reported in Table 2. This is mainly due to the following four reasons:

1) Tools not open-source, like Sherpa-ai [63]
2) Missing repositories: is the case of tools that have not yet released their codes after the paper publication: Chiron [65], FedHealth [66], FAE [67], GENO [68], FedTGan [69] and IPLS [71].
3) Coherent but not suitable: is the case of LEAF [45], FL-Bench [70], and PyFed [57] which are positioned for benchmarking purposes and therefore, might lack of essential features for conducting more extensive research activities. FedGraphNN [46] is a

sub-project of the bigger initiative called FedML [40] already included in this survey.

4) New tools or new openings: is the case of Nvidia-Flare (a sub-project of Nvidia-Clara https://docs.nvidia.com/clara/) and FL-Pytorch [64] which opened their repositories at some point after March harvest, as well as FLUTE [72] and PLATO [73] which have been retrieved (and added) during the September harvest.

After pruning the 12 tools that did not qualify, adding the 2 (despite the substitution with Nvidia-Flare, Nvidia-Clara was already captured by Table 1) from the September harvest, we ended up with a list of 27 total tools.

In a second instance, a score would be associated to each cell based on a quantitative assessment. Since the driving ambition was to propose an objective way for classifying the tools, we tried to capture qualitative aspects in a very inclusive way, rewarding tools that demonstrated additional development effort for the community through the available material, without penalizing newer promising tools that might still be under development. More in details, we adopted a simple approach to assign a score to each cell and designed the "Score Table" 3:

- Documentation: we considered having a Paper P and/or a public repository of the tool Gh (Github) or Gl (Gitlab) to be a minimum requirement. Therefore we scored zero all the tools not matching this expectation, rewarding with 1 point all those tools that had at list one additional source of information (like a dedicated web-page or a richer documentation that would go beyond Readme files on repositories or slack support), and 1.5 all those that provided two or more sources.
- User Interface (UI): 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 [12] or google colab [13] to mention two). We gave 1 point to all the tools that had at least 1 form of user interface abstracting from programming on command line. Finally, 1.5 points were given to all the tools that had 2 or more user interface.
- Language: We assigned zero points where the information about the supported version did not seem to be clearly outlined in the documentation, while 1 point was given to the tools supporting at least 1 language (or 1 version of a language), and 1.5 points were given to all the tools supporting 2 languages (or 2 versions of a language). Finally, 2 points were given to all the tools where the engineering team made the extra effort to support more than 2 languages (or more than 2 versions of the same language).
- Supported AI frameworks: We assigned zero point where the information about the supported AI frameworks did not seem to be clearly outlined in the documentation; 1 point was given to each framework supported. When the number of different frameworks supported would go beyond 2, we would just assign the maximum score of 2.5 points.
- Type of AI: zero if not mentioned in the documentation, 1 for each type of AI supported (ML or DL).
- Templates/Examples availability
- Distribution channels: we set as minimum requirement the ability to download a repository and install the tool from there. Therefore we assigned zero points to all the tools respecting this minumum requirement, 1 point to all the tools that had at least 1 additional way to access the software package (e.g., Pipy or Anaconda), finally, 1.5 points were given to all the tools that could be installed in two or more ways.
- Multi-node mode: zero when only simulated environment on a single computing machine were mentioned. 1 point to all the tools that allow to implement real federation on multiple nodes and 0.5 if this capability comes with 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 seem to provide any of the two options. 1 point was given where either a containerized or virtualized environment were provided. 1.5 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.
- Modular architecture: based on the analysis of the repositories of all the tools, we trust that all of them respect this parameter.
- Horizontal or Vertical: a tool must implement at least one of the two, therefore 1 point was given only to the tools that would allow executions in both settings.
- 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 to tweak or inject any privacy or security module (i.e., Homomorphic encryption, secured communication protocols, blockchain, etc.). 1 point to all the tools that included at least one of the two.
- Easy integration with other tools: same process applied for evaluating the containers and virtualization mechanism.

6 Results discussion and future directions

6.1 Discussion

We firstly performed an extensive and inclusive semi-automated research to identify all possible suitable tools for implementing FL pipelines. Then we built a popularity ranking based on a Score that we calculated by combining the Stars, Forks and Watch indicators coming from each Git repositories. Following the same Scoring function, we were also able to draw the growth rate for each of the tools over a six months time window. We ultimately defined a score matrix and evaluated each tool to calculate a "maturity

12. https://jupyter.org/
13. https://colab.research.google.com/

TOOL	Docs.	DX	Lang.	AI frameworks	AI type	Examples	Dist. channels	Multinode?	C/V	H/V	PVC/SEC	Tools integration
APPFL	P, Gh, Pipy, W	N.M.	Python = 3.6	Pythorc	DL	Yes	Gh, Pipy	Yes	Docker	H	N.M.	MPI orchestration
DecFL	P, Gh	N.M.	N.M.	TF	DL	Yes	Gh	Yes	Required	H	Yes	N.M.
dsMTL	P, Gh,	N.M.	R only	R-based	ML	Yes	Gh	Yes	N.M.	H	Yes	N.M.
FATE	P, Gh, W	N.M.	3.8, 3.9	N.M.	ML, DL	Yes	Gh	Yes	Docker	H,V	Yes	KubeFATE, flake, Spark
FedBioMed (Gl)	P, Gh	Jupyter	python	Pythorc, TF	DL	Yes	Gh	Yes	Required	H	Yes	Flake, Sklearn
FedJAX	P, Gh, W	G. Colab, Jupyter	Python	Pythorc, TF	DL	Yes	Gh, Pipy	Simulated	N.M.	H	N.M.	N.M.
FedLab	P, Gh, W	N.M.	python = 3.6	Pytorch	DL	Yes	Gh, Pipy	Yes	Docker	H	Yes	N.M.
Fedlearn-algo	P, Gh	N.M.	Python 3.6, 3.7	Pythorc, TF	ML, DL	Yes	Gh	Yes	Docker	H,V	Yes	Hugging Face, Sklearn
FedML	P, Gh, W, dedicated links, Slack	N.M.	Python	Pytorch, TF, JAX, mxnet	ML, DL	Yes	Gh, Pipy	Yes	Docker	H	N.M.	MPI, NCCL, MQTT, gRPC, Pytorch RPC
Fedn	P, Gh, dedicated docs	Yes	Python 3.8	Pytorch, TF	DL	Yes	Gh	Yes	Required	H,V	Yes	C++, scikit-learn
FLOM	Missing repo											
Flower	P,	Jupyter, colab	Python = 3.6	Pytorch, TF, MxNet, Jax	ML, DL	Yes	Gh, Pipy	Yes	Yes	H	Yes	Android, flake, hugging Face
FLUTE	P, Gh	N.M.	3.8	Pytorch	DL	Yes	Gh	Azure Cloud	N.M.	H	N.M.	HuggingFace
HyFed	P, Gh	WebAPP	N.M.	N.M.	N.M.	Yes	Gh	Yes	N.M.	H	Yes	N.M.
IBM-Federated	P, Gh, W, video tutorials	Jupyter	Python 3.6	Pytorch, TF	ML, DL	Yes	Gh, wheel	Yes	Docker	H	Yes	Ray, Openshift
IPLS												
MTC-ETH	P, Gh	N.M.	N.M.	N.M.	N.M.	N.M.	Gh	Yes, but low number	Required	H	Yes	N.M.
Nvidia Flare	Gh, Website	N.M.	3.8	Pytorch, TF	DL, ML	Yes	Gh, pip	Yes	Docker	H	Yes	MONAI
OpenFED	P, Gh, W	Jupyter	Python 3.6	Pytorch	DL	Yes	Gh, Pipy	Simulated	N.M.	H	Yes	N.M.
OpenFL	P, Gh, slack, W, videos	Jupyter, Colab	Python 3.6, 3.7, 3.8	Pytorch, TF	DL	Yes	Gh, Pipy	Yes	Docker	H	Yes	MonAI, Hugging Face
PLATO	P, Gh, Website	N.M.	3.9	Pytorch, TF, Mindspore	DL	Yes	Gh, pip	Yes	Docker	H	N.M.	Catalyst, Mindspore
PriMIA	P, Gh, W	N.M.	N.M.	Pytorch	DL	Yes	Gh	Simulated	N.M.	H	Yes	PySyft, K8s
PySyft	P, Gh, Slack,	Jupyter	3.8	Pythorc, TF	DL	Yes	Gh, Pipy	Yes	Docker, VMs. +	H	Yes	PySyft, k8s
FL-Pytorch	P, Gh, W, slack, videos	Custom GUI	3.9	Pytorch	DL	Yes	Gh, pipy	Simulated	N.M.	H	N.M.	N.M.
PyVertical	P, Gh	Jupyter	3.6,3.7,3.8	N.M.	DL	Yes	Gh	Yes	Docker	V	Yes	Syft
Substra	P, Gh, W, Slack	N.M.	N.M.	N.M.	N.M.	Yes	Gh, Pipy	yes	docker	H	N.M.	N.M.
Sunday FL	P, Gh, youtube	N.M.	Python, java	N.M.	N.M.	Yes	Gh	Yes	Docker	H	N.M.	Azure
Tensorflow Federated	P, Gh, W	Colab	Python 3	TF	DL	Yes	Gh, Pipy	Simulated	N.M.	H	N.M.	N.M.
Vantage6	P, Gh, W, youtube	N.M.	Python 3.7	N.M.	N.M.	N.M.	Gh, Pipy	Yes	Docker	H	N.M.	N.M.

TABLE 2
Tool evaluation table. Legend: 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. The NM label refers to "Not Mentioned", meaning that the information did not appeared as mentioned in the available documentation.

table" as in Table 3. Based on the resulting ranking, the most mature tools are Flower [41], OpenFL [42] and IBM-Federated [43]. While the last two seems to be fairly closed to each other, the table leader does not appear to have a solid distance. PySyft [39] is following alone despite FedML [40] and Fedn [53] being very close behind.

This is just an initial observation, but things changes when we integrate into the equation the popularity results highlighted in Table 1 and the growth rate outlined in Figure 5. In Table 1, PySyft [39] and FATE https://fate.fedai.org/ are the two most popular tools according to the developers community, while Flower [41], OpenFL [42] and IBM-federated [43], cover the 5th, 6th and 7th placement, respectively, with a huge distance to the first two. An interesting aspect here is that there is a clear gap between what the community awarded as the most popular tools and what this work outlined as the most mature ones. On the same line, other important element can be deduced when we

#	TOOL	Docs.	DX	Lang.	AI frameworks	AI type	Examples	Dist. channels	Multinode?	C/V	H/V	PVC/SEC	Tools integration	TOTAL
1	Flower	1	1.5	2	2.5	2	1	1	1	1	0	1	1.5	15.5
2	OpenFL	1.5	1.5	2	2	1	1	1	1	1	0	1	1.5	14.5
3	IBM-Federated	1.5	1	1	2	2	1	1	1	1	0	1	1.5	14
4	PySyft	1	1	1	2	1	1	1	1	1.5	0	1	1.5	13
5	FedML	1.5	0	1	2.5	2	1	1	1	1	0	0	1.5	12.5
6	Fedn	1	1	1	2	1	1	0	1	0.5	1	1	1.5	12
7	Fedlearn-algo	0	0	1.5	2	2	1	0	1	1	1	1	1.5	12
8	PLATO	1	0	1	2.5	1	1	1	1	1	0	0	1.5	11
9	Nvidia Flare	0	0	1	2	2	1	1	1	1	0	1	1	11
10	FATE	1	0	1.5	0	2	1	0	1	1	1	1	1.5	11
11	APPFL	1	0	2	1	1	1	1	1	1	0	0	1	10
12	FedLab	1	0	2	1	1	1	1	1	1	0	1	0	10
13	FedBioMed (GitLab)	0	1	1	2	1	1	0	1	0.5	0	1	1.5	10
14	OpenFED	1	1	2	1	1	1	1	0.5	0	0	1	0	9.5
15	FedJAX	1	1.5	1	2	1	1	1	0.5	0	0	0	0	9
16	PyVertical	0	1	2	0	1	1	0	1	1	0	1	1	9
17	Tensorflow Federated	1	1	2	1	1	1	1	0.5	0	0	0	0	8.5
18	FL-Pytorch	1.5	1	1	1	1	1	1	0.5	0	0	0	0	8
19	PriMIA	1	0	0	1	1	1	0	0.5	0	0	1	1.5	7
20	Sunday FL	1	0	1.5	0	0	1	0	1	1	0	0	1	6.5
21	dsMTL	0.0	0.0	1.0	1.0	1.0	1.0	0.0	1.0	0.0	0.0	1.0	0.0	6
22	FLUTE	0	0	1	1	1	1	0	0.5	0	0	0	1	5.5
23	DecFL	0	0	0	1	1	1	0	1	0.5	0	1	0	5.5
24	Vantage6	1.5	0	1	0	0	0	1	1	1	0	0	0	5.5
25	Substra	1	0	0	0	0	1	1	1	1	0	0	0	5
26	HyFed	0	1	0	0	0	1	0	1	0	0	1	0	4
27	MTC-ETH	0.0	0.0	0.0	0.0	0.0	0.0	0.0	1.0	0.5	0	1.0	0.0	2.5

TABLE 3
Tool scoring table

also consider the results highlighted by the growth-rates reported in Figure 5. Leading that ranking is Flower [41], followed by FedML [40], FATE https://fate.fedai.org/ and PySyft [39]. OpenFL [42] is in the 8th placement, with a growth-rate of 0.26 which is approximately 10 times smaller compared to the Table leader which has value of 2.

One interesting aspect to notice is that regardless to which scoring we decide to consider, the top 5 placements seems to be occupied by the same names, just re-shuffled a bit. In fact, out of 15 possible different names, we can only count up to 8 different tools. Out of those 8 different names, 4 are the more dominants as they appear in at least 2 rankings: Flower [41], FedML [40], FATE https://fate.fedai.org/ and PySyft [39]. The remaining 4 are LEAF [45], Tensorflow-Federated (tff) https://www.tensorflow.org/federated, OpenFL [42] and IBM-federated [43].

Despite our best efforts to adopt objective scoring when building Table 3, we are aware that other valid alternatives might exist. For example, a more in depth analysis of all the functional features provided by each tools (such as communication protocols), as well as a more granular differentiation of external tools that can be integrated in the FL pipeline, could lead to different results highlighted. However, although we appreciate that such finer approach might eventually change the current points distances between elements in the lists, we would not expect major shifts in the main order. This consideration arises when looking at what is defining the current ranking. The success of the top two tools is mainly justified by the high score obtained in the "Usability" and "Portability" factors outlined in Section V. This might suggest that when tools have similar features with similar 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 could be key for the community. This consideration would also be confirmed when looking at the lower part of the Table 3: low scoring for these apparently worse ranked tools, might not necessarily be related to a lack of critical features but more to an insufficient documentation that might have compromised the exploitation. Oh the other side, the fact that the community have preferred other tools, as outlined in Table 1, led us to an open question mark about this discrepancy: why the tools with better documentation and easier entry points but similar features to the most popular tools are not currently being considered at the same (or higher) level by the community?

Among the possible causes, we have identified three main ones: 1) participation to bigger international projects involving multiple institutions; 2) tool adoption in various application fields; 3) more dissemination and marketing activities by the respective engineering teams. This suggests the revision of the proposed criteria in order to account for all these and potentially other factors to get closer to a comprehensive measure harmonizing the overall results.

6.2 Conclusions

There are several tools for implementing FL pipelines that could help accelerating the research activities of the community in this field. In this paper we provided a survey of all open source solutions and a ranking based on tool popularity and tool readiness with the ambition to guide also non-expert users in the adoption of FL solutions, boost their exploitation and possibly accelerate their research and development. What we have learnt through this work is that what can be mostly adopted by the community might not necessarily be the most mature tool. Although summarizing the results of the three tables might be difficult, we can say that if somebody does not know where to start with FL, tools like Flower [41], or PySyft [39] can represent a good compromise between maturity and popularity. At the same time, we recognize that a more in depth benchmarking with dedicated tools like [36], LEAF [45] or FL-bench [70] might be needed to properly asses the different peculiarities of each tool.

ACKNOWLEDGMENTS

The authors would like to thank...

REFERENCES

[1] K. Hjerppe, J. Ruohonen, and V. Leppänen, "The general data protection regulation: Requirements, architectures, and constraints," *CoRR*, vol. abs/1907.07498, 2019. [Online]. Available: http://arxiv.org/abs/1907.07498

[2] 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.

[3] 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 Transactions on Parallel and Distributed Systems*, vol. 31, no. 11, pp. 2524–2541, 2020.

[4] M. Y. Lu, R. J. Chen, D. Kong, J. Lipkova, R. Singh, D. F. Williamson, T. Y. Chen, and F. Mahmood, "Federated learning for computational pathology on gigapixel whole slide images," *Medical image analysis*, vol. 76, p. 102298, 2022.

[5] H. R. Roth, K. Chang, P. Singh, N. Neumark, W. Li, V. Gupta, S. Gupta, L. Qu, A. Ihsani, B. C. Bizzo *et al.*, "Federated learning for breast density classification: A real-world implementation," in *Domain Adaptation and Representation Transfer, and Distributed and Collaborative Learning*. Springer, 2020, pp. 181–191.

[6] 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)*. IEEE, 2019, pp. 270–274.

[7] 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 of Things Journal*, vol. 8, no. 7, pp. 5476–5497, 2020.

[8] 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," *Brainlesion*, vol. 11383, pp. 92–104, 2019. [Online]. Available: https://pubmed.ncbi.nlm.nih.gov/31231720

[9] N. Rieke, J. Hancox, W. Li, F. Milletari, H. R. Roth, S. Albarqouni, S. Bakas, M. N. Galtier, B. A. Landman, K. Maier-Hein *et al.*, "The future of digital health with federated learning," *NPJ digital medicine*, vol. 3, no. 1, pp. 1–7, 2020.

[10] D. Stripelis, J. L. Ambite, P. Lam, and P. Thompson, "Scaling neuroscience research using federated learning," in *2021 IEEE 18th International Symposium on Biomedical Imaging (ISBI)*. IEEE, 2021, pp. 1191–1195.

[11] G. Long, Y. Tan, J. Jiang, and C. Zhang, "Federated learning for open banking," in *Federated learning*. Springer, 2020, pp. 240–254.

[12] J. So, K. Hsieh, B. Arzani, S. Noghabi, S. Avestimehr, and R. Chandra, "Fedspace: An efficient federated learning framework at satellites and ground stations," 2022. [Online]. Available: https://arxiv.org/abs/2202.01267

[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] T. Li, A. K. Sahu, A. Talwalkar, and V. Smith, "Federated learning: Challenges, methods, and future directions," *IEEE Signal Processing Magazine*, vol. 37, no. 3, pp. 50–60, 2020.

[15] C. Xu, Y. Qu, Y. Xiang, and L. Gao, "Asynchronous federated learning on heterogeneous devices: A survey," *CoRR*, vol. abs/2109.04269, 2021. [Online]. Available: https://arxiv.org/abs/2109.04269

[16] M. J. Sheller, B. Edwards, G. A. Reina, J. Martin, S. Pati, A. Kotrotsou, M. Milchenko, W. Xu, D. Marcus, R. R. Colen *et al.*, "Federated learning in medicine: facilitating multi-institutional collaborations without sharing patient data," *Scientific reports*, vol. 10, no. 1, pp. 1–12, 2020.

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

[18] A. Imteaj, U. Thakker, S. Wang, J. Li, and M. H. Amini, "A survey on federated learning for resource-constrained iot devices," *IEEE Internet of Things Journal*, vol. 9, no. 1, pp. 1–24, 2021.

[19] Q. Wu, K. He, and X. Chen, "Personalized federated learning for intelligent iot applications: A cloud-edge based framework," *IEEE Open Journal of the Computer Society*, vol. 1, pp. 35–44, 2020.

[20] Y. Chen, Y. Ning, M. Slawski, and H. Rangwala, "Asynchronous online federated learning for edge devices with non-iid data," in *2020 IEEE International Conference on Big Data (Big Data)*, 2020, pp. 15–24.

[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. [Online]. Available: https://arxiv.org/abs/1812.02903

[22] S. Savazzi, M. Nicoli, and V. Rampa, "Federated learning with cooperating devices: A consensus approach for massive iot networks," *IEEE Internet of Things Journal*, vol. 7, no. 5, pp. 4641–4654, 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 Communications Surveys & Tutorials*, vol. 23, no. 3, pp. 1622–1658, 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," *arXiv preprint arXiv:2202.04309*, 2022.

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

[26] X. Li, K. Huang, W. Yang, S. Wang, and Z. Zhang, "On the convergence of fedavg on non-iid data," *arXiv preprint arXiv:1907.02189*, 2019.

[27] F. Sattler, S. Wiedemann, K.-R. Müller, and W. Samek, "Robust and communication-efficient federated learning from non-iid data," *IEEE transactions on neural networks and learning systems*, vol. 31, no. 9, pp. 3400–3413, 2019.

[28] C. Briggs, Z. Fan, and P. Andras, "Federated learning with hierarchical clustering of local updates to improve training on non-iid data," in *2020 International Joint Conference on Neural Networks (IJCNN)*. IEEE, 2020, pp. 1–9.

[29] K. Bonawitz, H. Eichner, W. Grieskamp, D. Huba, A. Ingerman, V. Ivanov, C. Kiddon, J. Konečný, S. Mazzocchi, B. McMahan *et al.*, "Towards federated learning at scale: System design," *Proceedings of Machine Learning and Systems*, vol. 1, pp. 374–388, 2019.

[30] R. Shokri and V. Shmatikov, "Privacy-preserving deep learning," in *Proceedings of the 22nd ACM SIGSAC conference on computer and communications security*, 2015, pp. 1310–1321.

[31] 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 *proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, 2017, pp. 1175–1191.

[32] E. Bagdasaryan, A. Veit, Y. Hua, D. Estrin, and V. Shmatikov, "How to backdoor federated learning," in *International Conference on Artificial Intelligence and Statistics*. PMLR, 2020, pp. 2938–2948.

[33] Y. Qin, H. Matsutani, and M. Kondo, "A selective model aggregation approach in federated learning for online anomaly detection," in *2020 International Conferences on Internet of Things (iThings) and IEEE Green Computing and Communications (GreenCom) and IEEE Cyber, Physical and Social Computing (CPSCom) and IEEE Smart Data (SmartData) and IEEE Congress on Cybermatics (Cybermatics)*. IEEE, 2020, pp. 684–691.

[34] 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, 2021. [Online]. Available: https://www.mdpi.com/1424-8220/21/1/167

[35] D. Chen, V. Tan, Z. Lu, and J. Hu, "Openfed: A comprehensive and versatile open-source federated learning framework," 2021.

[36] 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," *arXiv preprint arXiv:2207.10308*, 2022.

[37] J.-P. Salameh, P. M. Bossuyt, T. A. McGrath, B. D. Thombs, C. J. Hyde, P. Macaskill, J. J. Deeks, M. Leeflang, D. A. Korevaar, P. Whiting *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, 2020.

[38] S. Fricke, "Semantic scholar," *Journal of the Medical Library Association: JMLA*, vol. 106, no. 1, p. 145, 2018.

[39] 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," 2021.

[40] 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," *arXiv preprint arXiv:2007.13518*, 2020.

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

[42] G. A. Reina, A. Gruzdev, P. Foley, O. Perepelkina, M. Sharma, I. Davidyuk, I. Trushkin, M. Radionov, A. Mokrov, D. Agapov *et al.*, "Openfl: An open-source framework for federated learning," *arXiv preprint arXiv:2105.06413*, 2021.

[43] H. Ludwig, N. Baracaldo, G. Thomas, Y. Zhou, A. Anwar, S. Rajamoni, Y. Ong, J. Radhakrishnan, A. Verma, M. Sinn *et al.*, "Ibm federated learning: an enterprise framework white paper v0. 1," *arXiv preprint arXiv:2007.10987*, 2020.

[44] X. H. Dun Zeng, Siqi Liang and Z. Xu, "Fedlab: A flexible federated learning framework," *arXiv preprint arXiv:2107.11621*, 2021.

[45] S. Caldas, S. M. K. Duddu, P. Wu, T. Li, J. Konečný, H. B. McMahan, V. Smith, and A. Talwalkar, "Leaf: A benchmark for federated settings," 2018. [Online]. Available: https://arxiv.org/abs/1812.01097

[46] C. He, K. Balasubramanian, E. Ceyani, C. Yang, H. Xie, L. Sun, L. He, L. Yang, P. S. Yu, Y. Rong, P. Zhao, J. Huang, M. Annavaram, and S. Avestimehr, "Fedgraphnn: A federated learning system and benchmark for graph neural networks," 2021.

[47] B. Liu, C. Tan, J. Wang, T. Zeng, H. Shan, H. Yao, H. Heng, P. Dai, L. Bo, and Y. Chen, "Fedlearn-algo: A flexible open-source privacy-preserving machine learning platform," *arXiv preprint arXiv:2107.04129*, 2021. [Online]. Available: https://arxiv.org/abs/2107.04129

[48] J. H. Ro, A. T. Suresh, and K. Wu, "Fedjax: Federated learning simulation with jax," *arXiv preprint arXiv:2108.02117*, 2021.

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

[50] 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," *arXiv preprint arXiv:2104.00489*, 2021.

[51] G. Kaissis, A. Ziller, J. Passerat-Palmbach, T. Ryffel, D. Usynin, A. Trask, I. Lima, J. Mancuso, F. Jungmann, M.-M. Steinborn *et al.*, "End-to-end privacy preserving deep learning on multi-institutional medical imaging," *Nature Machine Intelligence*, vol. 3, no. 6, pp. 473–484, 2021.

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

[53] M. Ekmefjord, A. Ait-Mlouk, S. Alawadi, M. Åkesson, D. Stoyanova, O. Spjuth, S. Toor, and A. Hellander, "Scalable federated machine learning with fedn," *arXiv preprint arXiv:2103.00148*, 2021.

[54] 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*, S. Albarqouni, S. Bakas, K. Kamnitsas, M. J. Cardoso, B. Landman, W. Li, F. Milletari, N. Rieke, H. Roth, D. Xu, and Z. Xu, Eds. Cham: Springer International Publishing, 2020, pp. 201–210.

[55] M. Ryu, Y. Kim, K. Kim, and R. K. Madduri, "APPFL: open-source software framework for privacy-preserving federated learning," *CoRR*, vol. abs/2202.03672, 2022. [Online]. Available: https://arxiv.org/abs/2202.03672

[56] R. Nasirigerdeh, R. Torkzadehmahani, J. Matschinske, J. Baumbach, D. Rueckert, and G. Kaissis, "Hyfed: A hybrid federated framework for privacy-preserving machine learning," 2021.

[57] H. Bouraqqadi, A. Berrag, M. Mhaouach, A. Bouhoute, K. Fardousse, and I. Berrada, "Pyfed: extending pysyft with n-iid federated learning benchmark," *Proceedings of the Canadian Conference on Artificial Intelligence*, 6 2021, https://caiac.pubpub.org/pub/7yr5bkck. [Online]. Available: https://caiac.pubpub.org/pub/7yr5bkck

[58] H. Cao, Y. Zhang, J. Baumbach, P. R. Burton, D. Dwyer, N. Koutsouleris, J. Matschinske, Y. Marcon, S. Rajan, T. Rieg, P. Ryser-Welch, J. Späth, T. C. consortium, C. Herrmann, and E. Schwarz, "dsmtl - a computational framework for privacy-preserving, distributed multi-task machine learning," *bioRxiv*, 2021. [Online]. Available: https://www.biorxiv.org/content/early/2021/08/28/2021.08.26.457778

[59] P. Niedziela, A. Danilenka, D. Kolasa, M. Ganzha, M. Paprzycki, and K. Nalinaksh, "Sunday-fl -developing open source platform for federated learning," 05 2021.

[60] F. Morsbach and S. Z. Toor, "Decfl: An ubiquitous decentralized model training protocol and framework empowered by blockchain," *Proceedings of the 3rd ACM International Symposium on Blockchain and Secure Critical Infrastructure*, 2021.

[61] C. Schneebeli, S. Kalloori, and S. Klingler, "A practical federated learning framework for small number of stakeholders," in *Proceedings of the 14th ACM International Conference on Web Search and Data Mining*, 2021, pp. 910–913.

[62] 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 *AMIA Annual Symposium Proceedings*, 2020, pp. 870–877.

[63] N. R. Barroso, G. Stipcich, D. Jim'enez-L'opez, 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, 2020.

[64] K. Burlachenko, S. Horváth, and P. Richtárik, "Fl_pytorch: optimization research simulator for federated learning," in *Proceedings of the 2nd ACM International Workshop on Distributed Machine Learning*, 2021, pp. 1–7.

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

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

[67] Y. Yue, Z. Ming, Q. Zhijie, L. Lei, and C. Hong, "A data protection-oriented design procedure for a federated learning framework," in *2020 International Conference on Wireless Communications and Signal Processing (WCSP)*, 2020, pp. 968–974.

[68] S. Carpov, N. Gama, M. Georgieva, and D. Jetchev, "Genoppml–a framework for genomic privacy-preserving machine learning," *Cryptology ePrint Archive*, 2021.

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

[70] Y. Liang, Y. Guo, Y. Gong, C. Luo, J. Zhan, and Y. Huang, "Flbench: A benchmark suite for federated learning," in *BenchCouncil International Federated Intelligent Computing and Block Chain Conferences*. Springer, 2020, pp. 166–176.

[71] C. Pappas, D. Chatzopoulos, S. Lalis, and M. Vavalis, "Ipls: A framework for decentralized federated learning," in *2021 IFIP Networking Conference (IFIP Networking)*. IEEE, 2021, pp. 1–6.

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

[73] Z. Jiang, W. Wang, B. Li, and B. Li, "Pisces: Efficient federated learning via guided asynchronous training," *arXiv preprint arXiv:2206.09264*, 2022.