International Journal of Machine Learning and Cybernetics (2024) 15:5257–5278
https://doi.org/10.1007/s13042-024-02234-z

ORIGINAL ARTICLE

Comparative analysis of open-source federated learning frameworks - a literature-based survey and review

Pascal Riedel[1,3] · Lukas Schick[2] · Reinhold von Schwerin[1] · Manfred Reichert[3] · Daniel Schaudt[1] · Alexander Hafner[1]

Received: 13 August 2023 / Accepted: 28 May 2024 / Published online: 28 June 2024

Abstract
While Federated Learning (FL) provides a privacy-preserving approach to analyze sensitive data without centralizing training data, the field lacks an detailed comparison of emerging open-source FL frameworks. Furthermore, there is currently no standardized, weighted evaluation scheme for a fair comparison of FL frameworks that would support the selection of a suitable FL framework. This study addresses these research gaps by conducting a comparative analysis of 15 individual open-source FL frameworks filtered by two selection criteria, using the literature review methodology proposed by Webster and Watson. These framework candidates are compared using a novel scoring schema with 15 qualitative and quantitative evaluation criteria, focusing on features, interoperability, and user friendliness. The evaluation results show that the FL framework Flower outperforms its peers with an overall score of 84.75%, while Fedlearner lags behind with a total score of 24.75%. The proposed comparison suite offers valuable initial guidance for practitioners and researchers in selecting an FL framework for the design and development of FL-driven systems. In addition, the FL framework comparison suite is designed to be adaptable and extendable accommodating the inclusion of new FL frameworks and evolving requirements.

Keywords Federated learning · Machine learning · Privacy · Open source · Framework comparison

1 Introduction

Federated Learning (FL) is a semi-distributed Machine Learning (ML) concept that has gained popularity in recent years, addressing data privacy concerns associated with centralized ML [1–7]. For example, data-driven applications with sensitive data such as in healthcare [8–12], finance [13, 14], personalized IoT devices [15, 16] or public service [17, 18] require a technical guarantee of data privacy, which can be achieved by the use of FL.

In FL, a predefined number of clients with sensitive training data and a coordinator server jointly train a global model, while the local training data remains on the original client and is isolated from other clients [1, 19]. In the FL training process, the global model is created by the server with randomly initialized weights and distributed to the clients of the FL system [20, 21]. The goal of a federated training process is the minimization of the following objective function:

✉ Pascal Riedel
pascal.riedel@uni-ulm.de

Lukas Schick
lukas.schick@student.uni-tuebingen.de

Reinhold von Schwerin
reinhold.vonschwerin@thu.de

Manfred Reichert
manfred.reichert@uni-ulm.de

Daniel Schaudt
daniel.schaudt@thu.de

Alexander Hafner
alexander.hafner@thu.de

[1] University of Applied Sciences Ulm, Prittwitzstraße 10, 89075 Ulm, Baden-Württemberg, Germany

[2] University of Tübingen, Geschwister-Scholl-Platz, 72074 Tübingen, Baden-Württemberg, Germany

[3] University of Ulm, Helmholzstraße 16, 89081 Ulm, Baden-Württemberg, Germany

$$\min f(w) = \sum_{k=1}^{N} \frac{n_k}{n} F_k(w),$$

where N is the number of clients, n_k the amount of sensitive training data on client k, n the total amount of training data on all clients and $F_k(w)$ is the local loss function [1, 22, 23]. Each client trains an initial model obtained by the coordinator server with the client's local training data [24]). The locally updated model weights are asynchronously sent back to the coordinator server, where an updated global model is computed using an aggregation strategy such as Federated Averaging (FedAvg) [1, 7, 20, 25–27]. The new global model is distributed back to the clients for a new federated training round. The number of federated training rounds is set in advance on the server side and is a hyperparameter that can be tuned [1, 5, 28, 29]. An overview of the FL architecture is introduced in Fig. 1. Also, FL can reduce the complexity and cost of model training by allowing a model to be trained on multiple smaller datasets on different clients, rather than on a single large, centralized dataset that requires an exhaustive data collection process beforehand [30–32]. Although there are a several key challenges to solve in the FL domain, security features such as homomorphic encryption [33, 34] and differential privacy [6, 35, 36] are already used to guarantee and improve data privacy and security in FL systems [37–44].

The advent of FL has spurred the development of various FL frameworks aimed at facilitating the deployment of FL applications, offering standardized functionalities and enhanced usability. Despite the proliferation of these frameworks, the selection of an optimal FL framework for specific project requirements remains a non-trivial challenge for practitioners due to the diversity and complexity of the choices available. This situation is exacerbated by two notable deficiencies in the FL research literature: first, the absence of a methodologically rigorous, in-depth comparative analysis of the most relevant open-source FL frameworks; and second, the lack of a standardized, weighted scoring scheme for a systematic and objective evaluation of these frameworks.

To the best of our knowledge, this comparative study is the most thorough to date, assessing the widest array of open-source FL frameworks against the broadest spectrum of criteria. Consequently, this study endeavors to fill the aforementioned research gaps by providing a robust FL framework comparison suite that could serve as a research-based guide for practitioners navigating the selection of suitable FL frameworks for their projects.

This study provides a comprehensive and user targeted comparison of 15 open-source FL frameworks by performing a systematic literature review according to Webster and Watson [45]. In this way, relevant FL frameworks and comparison criteria are identified, which are the basis for the comperative analysis. A novel weighted scoring system is proposed for the evaluation of FL frameworks. The proposed comparison criteria and the scoring system in this study can

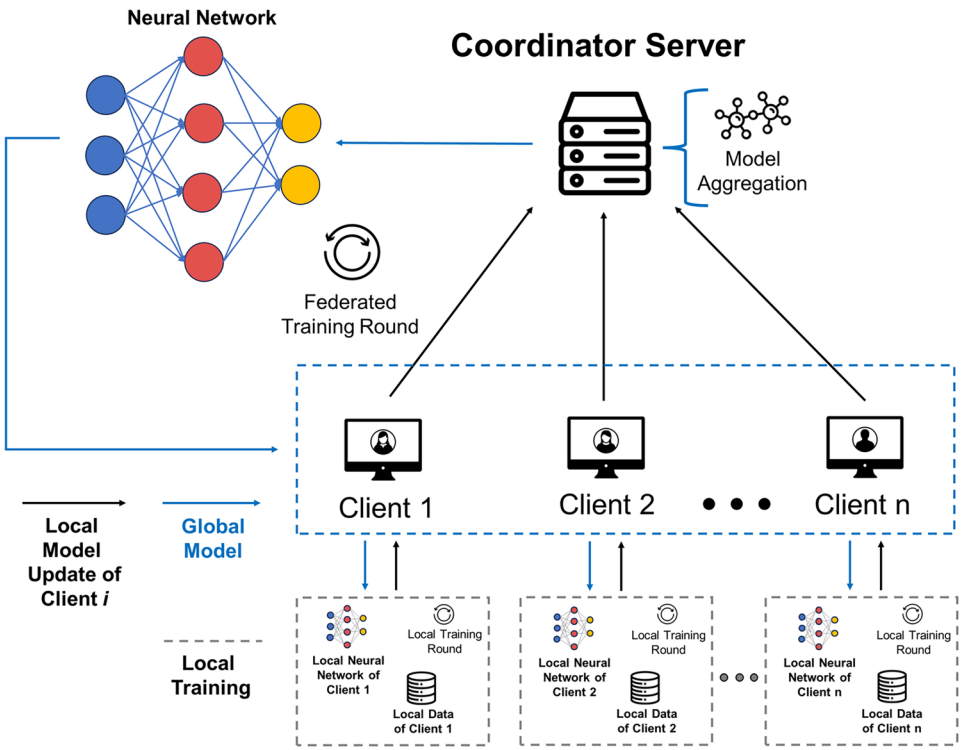

Fig. 1 Basic FL architecture overview

be utilized by practitioners and researchers to determine whether a particular FL framework fulfills their needs. Thus, the major contributions of this study can be summarized as follows:

- Proposing 15 comparison criteria for the evaluation of FL frameworks based on a methodological literature review.
- Introducing a novel weighted scoring matrix for these comparison criteria.
- Conducting an in-depth comparison of 15 relevant open-source FL Frameworks.

In addition, a **Research Question (RQ)** oriented approach is used in this study with the aim to answer the following three RQs:

- **RQ 1: Which relevant frameworks for FL exist and are open-source?**
- **RQ 2: Which criteria enable a qualitative and quantitative comparison of FL frameworks?**
- **RQ 3: Which FL framework offers the most added value to practitioners and researchers?**

The RQs are addressed and answered in ascending order in Sect. 5.4 on page 16.

The remainder of this paper is organized as follows. Section 2 discusses relevant related work and shows how the main contribution of this paper differs from the others. Section 3 details the literature review methodology applied in this work. Section 4 briefly introduces inclusion criteria and the FL framework candidates. Section 5 presents and discusses the comparison criteria, the weighting schema and the scoring results from the conducted FL framework comparison analysis. Section 6 describes the limitations of this study and suggests future work. Finally, Sect. 7 draws the conclusions of this survey.

2 Related work

In recent years, several research papers have been published dealing with individual FL frameworks. Some developers published works detailing and highlighting their own FL frameworks. For instance, the developers of FedML [46], Sherpa.ai FL [47], IBM FL [48], OpenFL [49], FATE [50], Flower [51], FLUTE [52], FederatedScope [53], FedLab [54] and EasyFL [55] have all published white papers introducing the features of their released frameworks. These papers include a general introduction to FL, open FL challenges, and how their FL framework can address them, while [29, 46, 47, 51, 52, 55] also provide small comparisons of a few existing FL frameworks. These comparisons were chosen subjectively and are biased, usually in favor of the FL framework developed by the author making the comparison, meaning a neutral, independent and holistic comparison is missing so far. In addition, there are research papers that address the current state of FL research, some of them using specific FL frameworks for technical implementation or evaluation purposes. For example, [5] showed a general and comprehensive overview of FL. They examined possible future research directions and challenges of FL, such as protection strategies against federated security attacks, and mentioned sources of federated bias. Moreover, they briefly introduced and described some popular FL frameworks, including FATE [56], PaddleFL [57], NVIDIA Clara (now a platform offering AI models for healthcare applications) [58], IBM FL [59], Flower [51] and FedLearner [60]. Another work [61] followed a similar approach as [5] and described central FL concepts such as the training process and FL algorithms in more detail before including a brief comparison overview of several FL frameworks. The authors of both works ([5] and [61]) refrain from evaluating FL frameworks and drawing conclusions from their conducted comparison analyses. In contrast to the aforementioned works, [62] described an in-depth comparison of multiple FL frameworks (TFF [63], FATE [56], PySyft [64] PaddleFL [57], FL &DP [65]). Both qualitative (in the form of a table comparing features of the frameworks) and quantitative comparisons (in the form of experiments, measuring training time and accuracy for three classification problems) are performed. Based on their evaluations, [62] recommended PaddleFL for the industrial usage, citing its high test accuracy for model inference tasks and range of features ready for practical use. A similar qualitative and quantitative FL framework comparison is provided by [66]. Their comparison contained more FL framework candidates than in the comparison conducted by [62] (9 vs 5). Furthermore, [66] performed a larger set of benchmark experiments, in which different FL paradigms were considered. The qualitative comparison was of a similar scope as in [62], although some criteria were left out (e.g., supported data types and protocols) and others have been added (e.g., documentation availability and GPU Support). Although the authors did not make a recommendation for a particular FL framework, they described a general decision-making process that can be used to determine the most appropriate FL framework.

In contrast to previous works, where the selection of comparison criteria for FL frameworks was often arbitrary, our study introduces a methodologically rigorous approach for a comparative analysis of FL frameworks. Prior works did not incorporate weighted importance of criteria nor did they employ a scoring mechanism for a systematic evaluation of FL frameworks. In addition, there was a lack of comprehensiveness in the inclusion of available and pertinent open-source FL frameworks. Our work advances the field by encompassing a broader spectrum of framework candidates

and employing a more integrative methodology for evaluating FL frameworks with a novel weighted scoring approach. Leveraging the structured literature review methodology by Webster and Watson, this comparative study identifies the most pertinent quantitative and qualitative criteria for FL framework users, ensuring a selection of comparison criteria that is both comprehensive and methodically sound, surpassing the scope of similar studies.

3 Research method

We applied the literature review methodology proposed by Webster and Watson [45] to address the RQs (see Sect. 1 on page 1). They introduced a systematic in-depth review schema for the identification and evaluation of relevant research literature. Webster and Watson's literature review method was published in response to the lack of reviewing articles in the information systems field, which the authors believe has slowed the progress in the field [45]. Their methodology has gained popularity since publication, with over 10 000 citations (based on Google Scholar citation count). According to [45], the collection process of relevant research literature should be concept-oriented or author-centric and is not limited to individual journals or geographical regions. They recommend to identify appropriate journal articles and conference proceedings by conducting a keyword-based search in different literature databases. Additional relevant sources should be identified by searching the references of the literature collected in this manner. This technique is called *backward search* and can be combined with *forward search*, which locates literature that cites one of the originally identified documents as a literature source. An overview of the searching methodology applied in this paper is shown in Fig. 2. We used the research literature review of Webster and Watson [45] to build the knowledge base for a literature-driven comparison analysis of open-source FL frameworks.

3.1 Literature databases and search string

For the literature search, the publication databases ACM Digital Library, EBSCOhost and IEEE Xplore were used to identify relevant publications and literature sources (see Fig. 2). As recommended by [45] we mainly searched for peer-reviewed journal articles and conference proceedings, so that a reliable research is feasible. A logical combination of the following terms served as the search string:

'federated learning' AND 'framework' AND 'open-source' OR 'federated framework' AND 'privacy-preserving machine learning' AND 'open-source'.

In some cases, additional search keywords were used, determined by reviewing the tables of contents of the retrieved literature based on the search string [45]. In addition, the research literature was filtered by publication date from 2016 to 2024 to obtain more recent sources. 2016 was chosen as date filter because that was the first year the term *federated learning* was officially used in a publication [1]. The forward and backward searches, as described by Webster and Watson [45], were used to identify additional relevant sources. This made it possible to identify publications that referenced other relevant publications, most of which were not much older than the origin publications. One reason for this could be that the term *federated learning* did not exist before 2016, so the range of publication dates is quite narrow. For the forward search, Google Scholar, Science

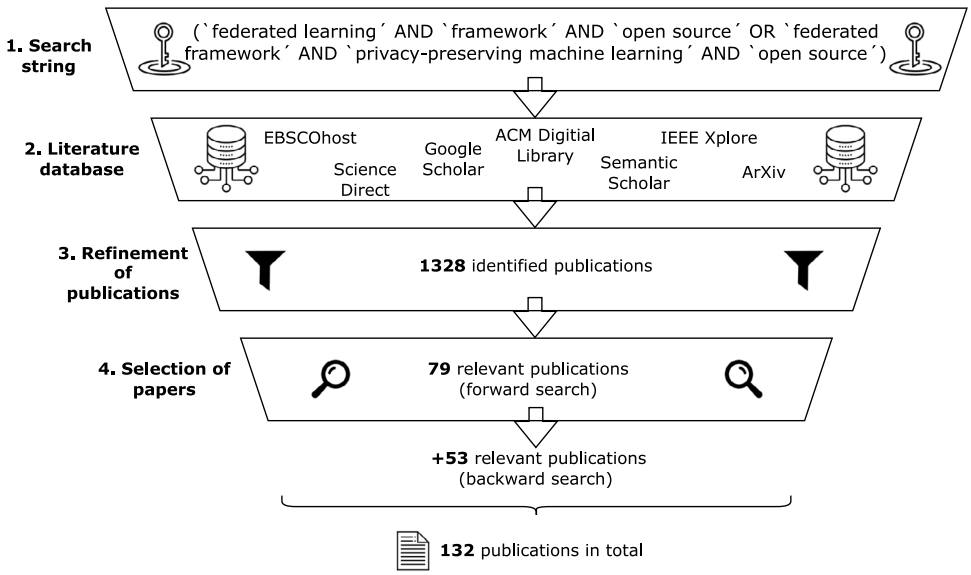

Fig. 2 Process flow used in this study to identify and filter relevant publications for the literature review

Direct, Semantic Scholar, and ArXiv were used in addition to the literature databases mentioned above.

3.2 Inclusion and exclusion criteria

To further filter the identified publications, the following certain inclusion and exclusion criteria were used, defined as follows:

Inclusion Criteria:

- The identified publication deals with the topic of federated learning and contributes answers to at least one of the RQs (see Sect. 1 on page 1).
- The title and the abstract seem to contribute to the RQs and contain at least one of the following terms: framework, federated learning, machine learning, evaluation or open-source.

Exclusion Criteria:

- The publication is not written in English.
- The title and abstract do not appear to contribute to the RQs and do not contain a term from the search string (see Subsect. 3.1) or inclusion criteria.
- The publication is a patent, master thesis, or a non-relevant web page.
- The publication is not electronically accessible without payment (i.e. only print issue).
- All relevant aspects of the publication are already included in another publication.
- The publication only compares existing research and has no new input.

A publication is included in the pool of relevant literature for reviewing if both inclusion criteria are met, and it is excluded if any of the exclusion criteria is fulfilled. Exceptions that are not subject to these criteria are sources that additionally serve to quantitatively or qualitatively support the comparison, such as GitHub repositories or the websites from the FL frameworks. Such sources are also included in our literature database, having a low relevance score.

3.3 Pool of publications

We initially checked the titles and abstracts of the publications for the individual key words of the search term (see Subsect. 3.1 on page 4) and added the publications to the literature pool if there were any matches based on the defined inclusion and exclusion criteria (see Subsect. 3.2 on page 5). Thus, 1328 individual publications from the literature databases were obtained. With the introduction and conclusion, 1196 publications have been eliminated due to lack of relevance. As a result, 132 publications, including 60 peer-reviewed journal articles, 27 conference proceedings, 10 white papers and 35 online sources form the basis for the literature-driven comparative analysis. In the refinement process (see step 3 on Fig. 2 on page 4), duplicated sources were removed, since in some cases the same publication was listed in at least or more than two literature databases.

3.4 Literature review

For the literature review a concept-oriented matrix according to Webster and Watson was used, which enables a systematic relevance assessment of the identified literature [45]. A publication is rated according to the number of concepts covered. Based on the RQs (see Sect. 1 on page 1), the individual concepts or topics for the literature review in this study are defined as follows:

- FL General Information (GI)
- FL Security Mechanisms (SM)
- FL Algorithms (AL)
- FL Frameworks (FW)

For each identified source, the title, the type of publication, the name of the publishing journal or conference if applicable, the number of citations, and a brief summary of the relevant content were noted. Afterwards, the literature was scored based on a scale of 1 to 4, with a publication scored 4 representing high relevance and a publication scored 1 representing low relevance. The rating schema is based on the concepts described above and defined as follows:

- 1 Point: Relevant to one specific concept except for FW.
- 2 Points: Relevant to at least two concepts or FW.
- 3 Points: Relevant to at least three concepts or FW and one or two other concepts.
- 4 Points: Relevant to all four concepts (GI, SM, AL and FW).

Additional sources not directly related to the concepts defined above were included in the concept *Misc.* and have been automatically assigned a relevance score of 1. An excerpt of the applied concept-oriented tabular literature review according to Webster and Watson [45] can be found in Table 1 on page 7. In this study, the knowledge base obtained from the literature review forms the basis for the weighted comparison and evaluation of different open-source FL frameworks (see Sect. 5 on page 8).

Table 1 Excerpt of the applied tabular research literature review according to Webster and Watson [45]

Author(s)	Title	Type, Publisher & Year	GI	SM	AL	FW	Misc.	Key Statements	Rel.
Kairouz et al	Advances and Open Problems in FederatedLearning	Article published in Founddations and Trends in Machine Learning, 2019	X	X	X	X		Presentation of FL and its open problems, as well as the possible possible solutions to these problems. Different security mechanisms are presented, as well as attack scenarios. It is noted that the areas of communication cost- differential privacy, server attacks, cryptography-based aggregation, collusion aggregation, the interplay between robustness and fairness and the and fairness, and the system characteristics of clients for FLpose challenges	4
Lo et al	A Systematic Literature Review on Federated Machine Learning: From A Software Engineering Perspective	Article published in ACM Computing Surveys, 2021	X	X	X			Presentation of the characteristics, applications, challenges and evaluation methods of FL by answering four key RQs ('What is FL?', 'What challenges of FL are being addressed?', 'How are FL challenges being addressed?', 'How are the approaches evaluated?'). Five action areas for FL (Model Aggregation, Training Management, Incentive Mechanism, Privacy Preservation, Resource Management) are identified	3
Liu et al	FATE: An Industrial Grade Platform for Collaborative Learning With Data Protection	Article published in Journal of Machine Learning Research, 2021				X		Article presenting FATE, an FL framework developed by WeBank. Components and features are discussed and performance benchmarks are presented	2
Bagdasaryan et al	How To Backdoor Federated Learning	Article published in Proceedings of Machine Learning Research, 2020		X				Presentation of a backdoor attack against a Federated Learning application, where attackers modify the model. It is shown how attackers can manipulate the FL training process and adapt the model at will	1

Table 2 Allocation of LDA topics to tabular literature review concepts

LDA Topics	Literature Review Concept
Topic 0, Topic 1, Topic 2	FL General Information
Topic 4, Topic 6, Topic 7	FL Security Mechanisms
Topic 3, Topic 5, Topic 8	FL Algorithms
Topic 9	FL Frameworks

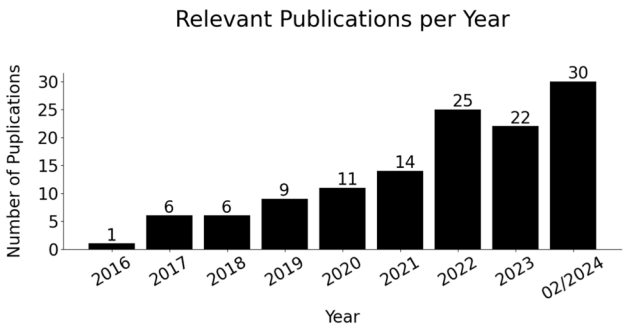

Fig. 3 Histogram of reviewed literature by year of publication from 2016 (first FL publication) to February 2024 (current research)

3.5 Literature analysis

To analyze the research literature, a Latent Dirichlet Allocation (LDA) was applied on the identified publications to discover common overlapping topics [67]. This could be used to verify the relevance of our chosen Literature Review Concepts. Stop words, numerical characters and conjunctions have been filtered out in advance. The number of components of the LDA was set to 10. This number was chosen after conducting a grid search and analyzing the generated topics. With the number of components set to 10, a topic that could be assigned to the Literature Review Concept 'FL Frameworks' was included for the first time. Thus, this was the lowest number of topics with which all four of the identified Literature Review Concepts were captured by the LDA. In each topic, the LDA determined the 20 most relevant words from the provided literature. Relevance represents the amount of times a word was assigned to a given topic [67]. Figure 5 (see Appendix, on page 18) displays these identified topics and their most relevant words. The topics were further condensed into the previously defined four concepts in Table 2. A word cloud consisting of the most common words in the identified literature can be seen in Fig. 6 (see Appendix, on page 19).

The literature-driven analysis reveals that FL frameworks have not often been part of research works on FL (see Table 2). This work aims to close this research gap. Figure 3 on page 6 shows the distribution of reviewed FL sources by the publication year. Noticeable is that FL received an overall boost in research interest in 2022 compared to 2021 (25 vs 14 publications). We expect the number of research publications on the four FL concepts described (see Subsect. 3.4 on page 5) to increase in the future as more user-friendly FL frameworks facilitate accessibility to FL to a wider range of users. It is worth to mention that some sources dealing with FL frameworks are GitHub repositories and white papers of the framework developers. In conducting the literature review (see Table 1 on page 7), a total of **18** FL frameworks were identified for the comparison and evaluation. To filter the number of FL frameworks, inclusion criteria are defined and used in this study. These filter criteria and the selected FL frameworks are described in the next section.

4 Federated learning frameworks

Although the term FL was coined as early as in 2016 [1], it is only in recent years that more Python-based frameworks have emerged that attempt to provide FL in a more user-friendly and application-oriented manner (see Fig. 3 on page 6). Some of the identified FL frameworks are hidden behind paywalls or are completely outdated and no longer actively developed and supported, making it impractical to include them for a fair comparison. Therefore, the following two inclusion criteria must be fulfilled by the FL frameworks in order to be considered as comparison candidates.

4.1 Inclusion criteria

Open-Source Availability In this paper, we also want to contribute to the topic of open-source in AI solutions and affirm its importance in the research community. In times when more and more AI applications are offered behind obfuscated paywalls (e.g., OpenAI [68]), researchers and developers should also consider the numerous advantages when developing innovative AI solutions as open-source products. After all, the rapid development of AI has only been possible due to numerous previous relevant open-source works. Thus, for the comparison study only open-source FL frameworks are chosen.

A few enterprises, such as IBM [59] or Microsoft [69], offer both a commercial integration and a open-source version of their FL frameworks for research purposes. For such FL frameworks only the free versions are considered in our comparison analysis.

Commercial FL frameworks such as Sherpa.ai FL [47, 65] are not considered in this work as they do not follow

the spirit of open-source. Benchmarking frameworks such as LEAF [70] or FedScale [71] were also excluded.

Community Popularity Another inclusion criterion used for filtering FL frameworks is the popularity in the community. It can be assumed that FL frameworks with an active and large GitHub community are more actively developed, more likely to be supported in the long term and thus more beneficial for practitioners. Therefore, this criterion excludes smaller or experimental FL frameworks, such as OpenFed [72].

As a metric for community activity the number of GitHub Stars are used. FL frameworks that have received at least 200 GitHub Stars for their code repositories are considered. The GitHub Stars indicate how many GitHub users bookmarked the repository, which can be interpreted as a reflection of the popularity of a GitHub repository. In fact, only FL frameworks provided by a company or an academic institution are considered in this study.

4.2 Considered frameworks

To provide a first initial overview of the 15 filtered FL frameworks, a comparison of them is shown in Table 3 on page 9 based on the following metrics: the developer country of origin, GitHub stars, the number of Git releases, dates of the initial and lates releases. Notably, PySyft is the most popular FL framework with over 9000 GitHub stars, followed by FATE AI and FedML. In general, FL frameworks which were released earlier have a higher numbers of GitHub stars. PySyft and TFF have been updated the most, while FLUTE has not yet had an official release on GitHub. Apart from Flower, all other FL frameworks were developed either in China or in the USA. 200 was chosen as the critical value, as this produces a manageable number of FL frameworks with the greatest popularity. In addition, a clear break between the much and little observed frameworks can be seen in this value range, as only a few frameworks can be found between 500 and 200, before the number of repositories increases drastically below 200 stars.

5 Framework comparison and evaluation

This section starts with the introduction of the comparison criteria and the weighted scoring system in Subsec. 5.1 on page 8. Then, the comparison and evaluation of the 15 FL frameworks is performed and the results are presented in 5.2 on page 11. This section closes with a discussion and analysis of our findings in 5.3 on page 14.

Table 3 An overview of general information about the FL frameworks selected for the comparative analysis, sorted by stars on their respective GitHub repositories (retrieved in February 2024, rounded to the nearest 100)

Framework / Criterion	PySyft [64, 73]	FATE AI [56, 74]	FedML [46, 75, 76]	Flower [51, 77, 78]	TFF [63, 79]	Fed-Learner [60]	PaddleFL [57, 80]	OpenFL [49, 81, 82]	IBM FL [48, 59, 83]	FLARE [58, 84–86]	FLSim [87]	FLUTE [52, 69, 88]	FedScope [53, 89, 90]	FedLab [54, 91, 92]	EasyFL [55, 93, 94]
Date of initial release	Jan 19, 2020	Feb 18, 2019	Jul 27, 2020	Nov 11, 2020	Feb 20, 2019	Oct 26, 2020	Apr 6, 2020	Feb 1, 2021	Aug 28, 2020	Nov 23, 2021	Dec 9, 2021	Jan 25, 2022	May 06, 2022	Aug 21, 2021	Apr 10, 2022
Date of latest release	Feb 20, 2024	Dec 31, 2023	Oct 28, 2023	Feb 06, 2024	Feb 13, 2024	Apr 21, 2023	Dec 6, 2021	Oct 24, 2023	Aug 01, 2023	Jul 28, 2023	Jul 27, 2022	Jan 25, 2022	Apr 03, 2023	Oct 26, 2022	Apr 10, 2022
Number of Git releases	108	48	7	18	75	4	5	9	10	24	3	0	3	9	1
Number of Git stars for repository	9100	5400	4000	3700	2200	900	500	600	500	500	200	200	1200	600	300
Country	USA	China	China	Germany	China	China	China	USA	USA	USA	USA	USA	China	China	China

5.1 Criteria and weighting definition

To ensure a detailed comparison, the FL frameworks are examined from three different perspectives, namely **Features**, **Interoperability** and **User Friendliness** using a weighted scoring system. All three main comparison categories each make up 100%. For each comparison category, this subsection describes individual comparison criteria and their weighting in descending order of relevance. The comparison criteria in each perspective category were selected based on the systematic literature review described in 3.4 on page 5.

Features This comparison category aims to examine and compare the inherent features of each FL framework. From the user's point of view, it is mandatory to know the relevant features of an FL framework in order to select a suitable framework for an FL project. Typical FL framework features include the support of different *FL Paradigms* (horizontal, vertical, and federated transfer learning), *Security Mechanisms* (cryptographic and algorithm-based methods), different *FL Algorithms* and specific federated *ML Models* [33, 34, 95–101].

In terms of weighting, *Security Mechanisms* is weighted most heavily at 35%, because increased data privacy and security is the main motivation for using FL in most applications [102] and the inherent properties of FL do not guarantee complete security [34, 103–106].

FL Algorithms and *ML Models* are given equal weighting at 25%, as both a wide range of algorithms and models are important to make an FL framework adaptable to different data-driven use cases [62, 66, 102].

The criterion *FL Paradigms* is weighted at 15%, because horizontal FL is still the most common FL paradigm [102], making the inclusion of other FL paradigms (i.e. vertical FL [107], and federated transfer learning [108]) less pertinent.

Interoperability

Interoperability is a mandatory factor in the evaluation of FL frameworks, particularly in terms of their compatibility with various software and hardware environments. This category includes support for multiple operating systems beyond the universally supported Linux containerization via Docker, CUDA support for leveraging GPUs, and the feasibility of deploying federated applications to physical edge devices [66].

The criterion *Rollout To Edge Devices* is weighted at 50%. This comparison criterion is crucial for the practical deployment of FL applications, enabling real-world applications rather than mere simulations confined to a single device [62, 66]. Without this, the scope of FL frameworks would be significantly limited to theoretical or constrained environments.

Support for different *Operating Systems* is assigned a weight of 25%. This inclusivity ensures that a broader range of practitioners can engage with the FL framework, thereby expanding its potential user base and facilitating wider adoption across various platforms [62].

GPU Support is considered important due to the acceleration it can provide to model training processes, and is weighted at 15%. Although beneficial for computational efficiency, GPU support is not as critical as the other criteria for the core functionality of an FL framework [66].

Lastly, *Docker Installation* is recognized as a criterion with a 10% weight. Docker's containerization technology offers a uniform and isolated environment for FL applications, mitigating setup complexities and compatibility issues across diverse computing infrastructures [109]. While Docker support enhances versatility and accessibility, it is deemed optional since there are FL frameworks available that may not necessitate containerization for running on other OSes. Although Docker's containerization is a beneficial attribute for FL frameworks, it is not as heavily weighted as the capacity for edge device deployment or OS support, which are more essential for the practical implementation and broad usability of FL applications.

User Friendliness The aim of this comparison category is to examine and compare the simplicity and user-friendliness of the individual FL frameworks when creating FL applications. The simple use of an FL framework can shorten the development times in an FL project and thus save costs. Therefore, the following comparison criteria should be considered in this criteria group: *Development Effort* needed to create and run an FL session, federated *Model Accuracy* on unseen data, available online *Documentation*, FL *Training Speed*, *Data Preparation Effort*, *Model Evaluation* techniques and, if existing, the *Pricing Systems* for additional functionalities (e.g., online dashboards and model pipelines) [62, 66].

The criteria *Development Effort* and *Model Accuracy* are deemed most critical, each carrying a 25% weight, due to their direct impact on the usability of FL frameworks and the effectiveness of the resultant FL applications [110]. The focus is on quantifying the ease with which developers can leverage the framework to create and deploy FL applications. This facet is critical as it directly influences the time-to-market and development costs of FL projects. Also for the FL application's success it is important how well a federated model can perform on unseen new data [62, 66].

The *Documentation* aspect is weighted with 20%. Given the novelty of many FL frameworks and the potential scarcity of coding examples, the availability and quality of documentation are evaluated [66]. This criterion underscores the importance of well-structured and informative

Table 4 Parameters and functions used for practical testing of User Friendliness

Parameters	Values
Number of clients	5
Federated rounds	3
Local epochs per round	1
Learning rate	5e-2
Model architecture	2 layers (784×10 parameters)
Hidden activation	ReLU
Output function	softmax
Loss function	categorical cross-entropy
Batch size	64
Train-test split (shuffled)	90:10

documentation that can aid developers in effectively utilizing the FL framework, encompassing tutorials, API documentation, and example projects.

The *Training Speed* criteria is weighted lower with 10%, since a faster training time is advantageous for any FL framework, but is less relevant compared to a high model accuracy [62, 66]. It reflects on the optimization and computational efficiency of the framework in processing FL tasks.

The *Data Preparation Effort* is assigned a weight of 10%. It evaluates the degree to which an FL framework supports data preprocessing and readiness, considering the ease with which data can be formatted, augmented, and made suitable for federated training. Although not critical for the operational use of an FL framework, streamlined data preparation processes can enhance developer productivity.

Model Evaluation receives the lowest weighting of 5%. It scrutinizes the methodologies and tools available within the FL framework for assessing global model performance and robustness, including validation techniques and metrics. Different model evaluation methods are helpful for practitioners, but not necessary for the effective use of an FL framework [66]. Thus, this criterion has more a supportive role in the broader context of FL application development.

Since the focus of this work is on open-source FL frameworks, the *Pricing Systems* is also only weighted at 5%. For FL frameworks that offer additional functionalities through paid versions, this evaluates the cost-benefit ratio of such features. While the core focus is on open-source frameworks, the assessment of pricing systems is still relevant for understanding the scalability and industrial applicability of the framework's extended features.

To assess the scores for the *Development Effort*, *Model Accuracy*, *Training Speed*, *Data Preparation Effort* and *Model Evaluation* criteria, a federated test application has been created, simulating an FL setting while running on a single device. This application used the MNIST dataset [111, 112] and performed an image multi-class classification task with a multi-layer perceptron neural network model. A grid search approach was used to identify an optimal hyperparameter configuration. The selected hyperparameters for the model trainings were used identically for testing each FL framework (see Table 4 on page 11).

Weighted Scoring In each of the three comparison categories mentioned above, the criteria are assigned weights that sum up to 100%. Consequently, the total score for all comparison criteria within a category represents the percentage score obtained by an evaluated FL framework in that particular category. These percentage scores for each category are then combined using a weighted sum to derive an overall total score. This serves as a final metric for selecting the best FL framework across all categories. All criterion weights are also listed in Table 7 on page 20 in the Appendix.

The distribution of the weighting of the three top level categories is as follows:

- **User Friendliness** has the highest weighting (**50%**), as the criteria in this category have the greatest impact for practitioners working with FL frameworks.
- **Features** has the second highest weighting (**30%**), as this category indicates which functionalities such as Security Mechanisms or FL Paradigms are supported in an FL framework.
- **Interoperability** is weighted as the lowest (**20%**), as it primarily indicates the installation possibilities of an FL framework, but does not represent core functionalities or the framework's usability.

The FL frameworks can achieve one of three possible scores in each criterion: a score of zero is awarded if the FL framework does not fulfill the requirements of the criterion at all. A half score is awarded if the FL framework partially meets the requirements. A score of one is awarded if the FL framework fully meets the requirements. If a criterion cannot be verified or tested at all, then it is marked with N.A. (Not Available). This is treated as a score of zero in this criterion when calculating the total score. The detailed scoring schemes for each criterion are given in Table 7 on page 20 in the Appendix.

5.2 Comparison results

The scoring Table 5 on page 12 shows the comparison matrix of the 15 FL framework candidates on the basis of the defined categories and criteria from Subsect. 5.1 on page 8. In the following, we explain our assessment of the

Table 5 Weighted scoring of the frameworks in the individual criteria. After evaluating the criteria in a category (based on the criteria defined in Table 7 on page 20), the category score is calculated by multiplying the criteria scores by the corresponding weights. In the bottom row, the total scores are calculated by multiplying the intermediate category scores by the corresponding total weights of each category

FL Framework / Criterion	PySyft	FATE AI	FedML	Flower	TFF	FedLearner	PaddleFL	OpenFL	IBM FL	FLARE	FLSim	FLUTE	FedScope	FedLab	EasyFL	Weight
Category: **Features**																
Security Mech	1	0.5	0.5	0.5	0.5	0.5	1	0.5	0.5	1	1	0.5	1	0	0	35%
FL Algorithms	0	0.5	1	1	1	0.5	0.5	1	1	1	0.5	1	1	1	0.5	25%
ML Models	1	1	1	1	0.5	0.5	0.5	1	1	1	0.5	0.5	1	0.5	0.5	25%
FL Paradigms	1	1	1	1	0	1	1	0	0	0	0	0	1	0	0	15%
Score	75%	70%	82.5%	82.5%	55%	57.5%	75%	67.5%	67.5%	85%	60%	55%	100%	37.5%	25%	**30%**
Category: **Interoperability**																
Rollout	1	0.5	1	1	0	0	0.5	0	1	1	0	0	0.5	0	0.5	50%
OS Support	1	0	1	1	0.5	0	0	0.5	1	1	1	0.5	1	1	0.5	25%
GPU Support	1	1	1	1	1	1	1	1	1	1	0	1	1	1	1	15%
Docker Install	1	1	1	1	1	1	1	1	1	1	0	0	1	1	1	10%
Score	100%	50%	100%	100%	37.5%	25%	50%	37.5%	100%	100%	25%	27.5%	75%	50%	62.5%	**20%**
Category: **User Friendliness**																
Dev. Effort	0	N.A	0.5	0.5	1	N.A	N.A	1	0.5	0	0.5	0.5	0.5	1	1	25%
Model Accuracy	1	N.A	0.5	1	0.5	N.A	N.A	1	1	1	1	0.5	1	0.5	1	25%
Documentation	1	0.5	0.5	1	1	0	0	0.5	0.5	1	0	0	0.5	0.5	1	20%
Training Speed	0	N.A	0.5	0.5	1	N.A	N.A	1	0	0.5	1	0.5	0.5	1	1	10%
Data Prep	1	N.A	0.5	1	0	N.A	N.A	0.5	0	1	0.5	1	0.5	1	0.5	10%
Evaluation	0	N.A	0.5	0.5	1	N.A	N.A	0.5	1	1	1	0.5	1	1	1	5%
Pricing Sys	1	1	1	1	1	1	1	1	0	1	1	0	1	1	1	5%
Score	60%	15%	52.5%	80%	77.5%	5%	5%	82.5%	52.5%	70%	62.5%	42.5%	67.5%	77.5%	95%	**50%**
Total Score	**72.5%**	**38.5%**	**71%**	**84.75%**	**62.75%**	**24.75%**	**35%**	**69%**	**66.5%**	**80.5%**	**54.25%**	**43.25%**	**78.75%**	**60%**	**67.5%**	**100%**

For better highlighting, the main categories, the category weighting and the total score for each framework are in bold

individual comparison criteria for the FL frameworks. Note: we write the individual comparison criteria in capital letters to highlight them.

Evaluation of Features It can be noted that for the first criterion, *Security Mechanisms*, five FL frameworks (PySyft, PaddleFL, FLARE, FLSim and FederatedScope) provide both cryptographic and algorithmic security features such as differential privacy, secure aggregation strategies, secure multiparty computation, trusted execution environments and homomorphic encryption [6, 34, 35, 53, 108, 113–122]. Therefore, these FL frameworks receive the full score for this criterion. On the other hand, FATE AI, FedML, TFF, Flower, FedLearner, OpenFL, IBM FL and FLUTE all provide only one type of security mechanism. Thus, these FL frameworks receive half the score [48–50, 52, 57, 63, 64, 66, 78, 86, 87, 123]. FedLab and EasyFL provide no security mechanisms and receive a score of zero in this criterion [54, 55, 92].

For the next criterion, *FL Algorithms*, the FL frameworks: FedML, TFF, Flower, OpenFL, IBM FL, FLARE, FLUTE, FederatedScope and FedLab receive full scores, because they provide out-of-the-box implementations of the FedAvg [1] algorithm as well as several different adaptive FL algorithms such as FedProx, FedOpt and FedAdam [124, 125]. On the other hand, FATE AI, FedLearner, PaddleFL, FLSim and EasyFL only provide FedAvg as an aggregation strategy; other algorithms are not available in these FL frameworks by default, resulting in a halving of the score on this criterion. PySyft is the only FL framework candidate that requires manual implementation of an FL strategy (even for FedAvg). Therefore, PySyft receives a zero score on this criterion as it requires more effort to set up a training process [46, 51, 52, 57, 60, 62, 63, 73, 81, 83, 86, 87, 89, 92, 93].

For building *ML Models*, PySyft, FATE AI, FedML, Flower, OpenFL, IBM FL, FLARE and FederatedScope support the deep learning libraries Tensorflow and PyTorch. They provide users with a wide range of federatable ML models. Therefore, these FL frameworks are awarded the full marks on this criterion. However, TFF (Tensorflow), FedLearner (Tensorflow), PaddleFL (PaddlePaddle), FLSim (PyTorch), FLUTE (PyTorch), FedLab (PyTorch) and EasyFL (PyTorch) receive half the score because users are limited to only one supported ML library [52, 57, 60, 62–64, 76, 78, 81, 83, 86, 87, 89, 91, 93].

In terms of *FL Paradigms*, there are seven FL frameworks that support both horizontal and vertical FL and therefore receive full marks: PySyft, FATE AI, FedML, Flower, FedLearner, PaddleFL and FederatedScope. TFF, OpenFL, IBM FL, FLARE, FLSim, FLUTE, FedLab and EasyFL receive a zero score because they only support the standard horizontal FL paradigm [55, 57, 66, 74, 78, 81, 83, 86–88, 90, 92, 126].

Evaluation of Interoperability The *Rollout To Edge Devices* that allows FL applications to be implemented in real-world environments (e.g., on thin-clients or IoT devices) is possible with PySyft, FedML, Flower, IBM FL and FLARE. Therefore, they receive full marks on this criterion. However, PySyft only supports Raspberry Pi, while the other four FL frameworks also support the Nvidia Jetson Development Kits [86]. FATE AI, PaddleFL, FederatedScope and EasyFL each receive half the possible score because the rollout process on edge devices is more cumbersome compared to the other FL frameworks. For example, FATE AI and PaddleFL require edge devices with at least 100 GB of storage and 6 GB of RAM, which excludes most single-board computers. The FL frameworks TFF, FedLearner, OpenFL, FLUTE, FLSim and FedLab do not score on this criterion because they only support FL in simulation mode on a single device [46, 52, 60, 62–64, 77, 83, 87, 89, 91, 93].

For the *Operating System* support, PySyft, FedML, Flower, IBM FL, FLARE, FLSim, FederatedScope and FedLab receive full marks, as Windows and MacOS are natively supported. On the other hand, the following FL framework candidates support only one of each: TFF (MacOS), OpenFL (MacOS), FLUTE (Windows) and EasyFL (MacOS) receive half the score. FATE AI, FedLearner and PaddleFL run only on Linux and require Docker containers when used on Windows or MacOS. Therefore, these three FL frameworks do not receive any points for this criterion [50, 57, 60, 73, 76, 78, 79, 81, 83, 84, 87, 88, 90, 91, 93].

All compared FL frameworks offer *GPU Support* and receive full scores on this criterion, except for FLSim. The documentation of FLSim makes no reference to a CUDA acceleration mode during FL training and CUDA could not be enabled during the conducted experiments. Therefore, this FL framework receives a score of zero in this criterion [51, 52, 63, 66, 73, 74, 76, 81, 83, 86, 87, 90, 91, 93].

13 of the 15 FL framework candidates have a *Docker containerization* option and therefore receive full marks. These frameworks provide Docker images, which can be installed using the Docker Engine. By setting up a Docker container, it is possible to create an isolated environment which makes it possible to install software even though its requirements are not supported by the system specifications [109]. Some frameworks like FLARE and OpenFL provide a Dockerfile which builds the image automatically, while other frameworks like PaddleFL provide a documentation on how to install the Docker image manually. Surprisingly, FLSim and Microsoft's FLUTE do not seem to support Docker containers. The use of Docker containers was not mentioned in the documentations and was not possible during the experiments conducted. Therefore, these two FL frameworks receive zero

Table 6 Model performances of the tested FL framework candidates in horizontal FL

Framework / Performance	PySyft	FedML	Flower	TFF	OpenFL	IBM FL	FLARE	FLSim	FLUTE	FedScope	FedLab	EasyFL
Accuracy in %	93.9	83.2	91.1	81.5	90.6	95.8	94.5	92.8	79.2	90.1	87.2	90.9
Training speed in minutes and seconds	03:02	01:32	01:07	00:43	00:35	03:40	02:41	00:42	02:18	01:02	00:55	00:33

Accuracy is computed as the share of correctly predicted classes (holdout)

points for this criterion [57, 60, 73, 74, 76, 78, 79, 81, 83, 84, 87, 88, 90, 91, 93].

Evaluation of User Friendliness For the FATE AI, PaddleFL, and FedLearner FL frameworks, it is not possible to evaluate the criteria *Development Effort*, *Model Accuracy*, *Training Speed*, *Data Preparation* effort, and model *Evaluation* because of a number of issues with these FL frameworks, such as failed installations on Windows, Linux or MacOS. Thus, these FL frameworks are marked as N.A. in the mentioned criteria, because test experiments could not be performed with them.

For *Development Effort*, TFF, OpenFL, FedLab and EasyFL receive a score of one as the setup of applications with these frameworks was intuitive, fast and required few lines of code. FedML, Flower, IBM FL, FLSim, FLUTE and FederatedScope receive a half score, since development requires more lines of code than with the four frameworks mentioned previously, but aspects of the training process like the federated aggregation step or the local loss step are implemented. PySyft and FLARE require the most development effort because parts of the training process, such as gradient descent, must be implemented and set by the user, which is not the case for the other FL framework candidates. Thus, PySyft and FLARE are rewarded with zero points on Development Effort.

As for the global *Model Accuracy*, PySyft, Flower, OpenFL, IBM FL, FLARE, FLSim, FedLab and EasyFL achieved a test accuracy of over 90% in the performed MNIST classification simulation. On the other hand, FedML, TFF, FLUTE and FederatedScope performed worse, achieving an accuracy below the 90% threshold, thus receiving only half the score, even though the same model architecture, configuration and parameters have been used (see Table 4 on page 11). The test accuracies for the tested frameworks can be found in Table 6 on page 13.

Surprisingly, the amount and quality of *Documentation* available for the FL frameworks varies widely. PySyft [64, 73], TFF [63, 79], Flower [51, 77, 78] FLARE [84–86] and EasyFL [55, 93, 94] provide extensive API documentation, several sample applications and video tutorials to learn how to use these frameworks. These FL frameworks receive the full score on the criterion Documentation. However, FedLearner [60], PaddleFL [57], FLSim [87], and FLUTE [69, 88] provide only little and mostly outdated documentation. Therefore, this group of FL frameworks receive zero points here. For FATE AI [56, 74], FedML [46, 75, 76], OpenFL [49, 81] IBM FL [48, 59, 83], FederatedScope [53, 89, 90] and FedLab [54, 91, 92], the available documentation is less extensive and at times outdated. These FL frameworks receive a score of 0.5 for this criterion.

When performing the test experiments with the FL framework candidates, there were also differences in the model *Training Speed*. With TFF, OpenFL, FLSim, FedLab and EasyFL, the federated training was completed in less than a minute, giving these frameworks a full score. FL Frameworks with a training speed between one and three minutes (FedML, Flower, FLARE, FLUTE, FederatedScope) received half of the score, while training on PySyft and IBM FL took longer than three minutes, resulting in a score of zero for these two frameworks. Since FLUTE can only be used on Windows [88], the training speed measurement may not be directly comparable to the measurements of the other FL frameworks which were computed on another computer running MacOS with a different hardware specification. The exact training speeds for the tested frameworks can be found in Table 6 on page 13.

For the assessment of the *Data Preparation* effort, we considered the effort required to transform proxy training datasets such as MNIST [112] into the required data format of the FL frameworks. Here, PySyft, Flower, FLARE, FLUTE and FedLab required only minor adjustments (e.g., reshaping the input data) and therefore received full scores, while TFF and IBM FL required more preparation, so both FL frameworks received no scores. FedML, OpenFL, FLSim, FederatedScope and EasyFL received a score of 0.5.

For the *Evaluation* criterion, TFF, OpenFL, IBM FL, FLSim, FederatedScope, FedLab and EasyFL provide built-in evaluation methods that display test set loss and accuracy metrics for the federated training of a global model, resulting in a full score for these FL frameworks in the Model Evaluation criterion. Since the main category is User Friendliness, PySyft receives a score of zero here because in PySyft all evaluation metrics must be implemented manually, which may include the requirements of additional libraries (e.g., TensorBoard). FedML, Flower, OpenFL and FLUTE

provided evaluation methods with incomplete or convoluted output and thus received a score of 0.5.

For the *Pricing System* criterion, all FL framework candidates except FLUTE and IBM FL receive full marks because their features are freely accessible. FLUTE is integrated with Azure ML Studio [69]. Microsoft touts a faster and easier federated development process by leveraging its cloud service and proclaiming FLUTE's integration with Azure ML as one of its key benefits, as the federated application can be used directly in the Azure ecosystem [69]. On the other hand, IBM FL is part of IBM Watson Studio cloud service, where additional features such as a UI-based monitoring and configuration are available that cannot be used in the open-source community edtion [59]. Therefore, FLUTE and IBM FL do not score on this criterion.

5.3 Discussion

Considering the scores at the category level, there are some FL frameworks that received notable scores in certain categories. FederatedScope received the highest score in the Features category with 100%, offering differential privacy and homomorphic encryption as security mechanisms, support for different ML libraries and many FL algorithms like FedAvg, FedOpt and FedProx. Meanwhile, EasyFL received only 25% of the score, offering no security mechanisms, FedAvg as the only implemented FL algorithm and one ML library, while only horizontal FL is available as a paradigm.

The FL frameworks PySyft, FedML, Flower, IBM FL and FLARE earned a perfect score of 100% in the Interoperability category, while FedLearner and FLSim performed joint-worst, receiving 25% of the category score (see Table 5 on page 12). FedLearner does not offer a rollout on edge devices and is not available for installation on either Windows or MacOS, limiting its potential user base. FLSim is available for both Windows and MacOS, but does not support a rollout on edge devices, GPU-based computation, or a Docker containerization.

Remarkably, EasyFL received the highest score of 95% in the User Friendliness category, fullfilling the most important criteria: Development Effort, Model Accuracy Documentation and Training Speed. The FL frameworks for which no test application could be created received the lowest scores, with FedLearner and PaddleFL receiving the lowest score in this category with 5%, and FATE AI receiving 15%. These low scores are noteworthy, since these three FL frameworks all have a long development history and are popular within the community (see Table 3 on page 9).

Based on the conducted comparison and evaluation, a ranking of FL frameworks can be constructed, which is visualized in Fig. 4 on page 15. It can be concluded that in terms of the overall score, Flower performed best with 84.75%, followed by FLARE with 80.5% and FederatedScope with 78.75% (see Table 5 on page 12). PySyft, FedML, OpenFL, EasyFL, IBM FL, TFF and FedLab all received scores at or above 60% overall. FLSim received a score of 54.25%

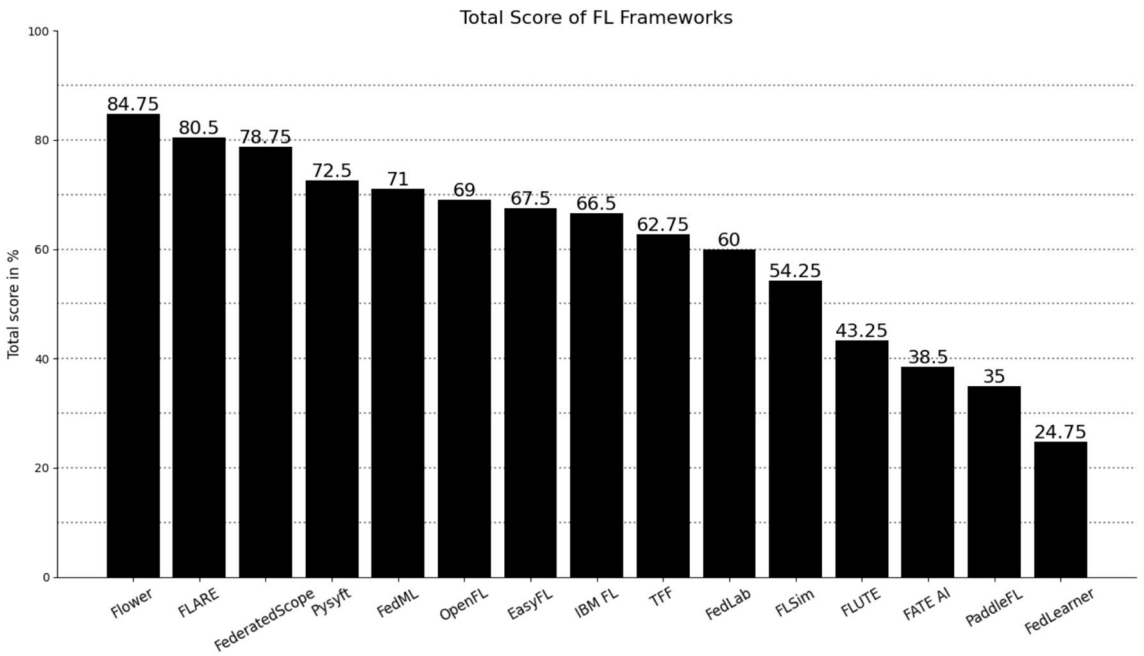

Fig. 4 Total scores (in percentage) of the compared frameworks

and FLUTE scored 43.25%, while FATE AI, PaddleFL and FedLearner all scored below 40% in total, with FedLearner's 24.75% marking the lowest score of the frameworks in this comparison.

The graphical representation of the scores on the bar plot further shows that the top ten FL frameworks, although with big differences in the category scores, all achieved relatively high total scores (at or above 60%). This suggests that a number of FL frameworks could already offer a satisfying solution for practitioners. The total score for the final five FL frameworks on the bar plot decreases sharply, indicating significant shortcomings in categories or specific criteria. FLSim and FLUTE scored low in the Interoperability category at 25% and 27.5% respectively, while FATE AI, PaddleFL and FedLearner received low User Friendliness scores (15%, 5%, and 5%).

Generally, the difference in score between the FL frameworks in the Features category is small compared to the other categories. Only two frameworks score below 50%. Most variance in this category is introduced by the security and paradigm criteria. Should secure computation and communication be the focal point of development, then PySyft, PaddleFL, FLARE, FLSim and FederatedScope would provide the most extensive features for this use case.

In the Interoperability category, it is observable that only five of the FL frameworks (PySyft, FedML, Flower, IBM FL, FLARE) support a rollout on edge devices without strong limitations. This explains the high fluctuation of scores for this category, as the Rollout criterion was weighted heavily. Should the development of a fully realized, distributed FL application be central to a project, these five FL frameworks offer the best conditions and are most suitable for communication and real-time computing with IoT edge devices.

Examining the User Friendliness category, the Development Effort and Documentation criteria explain a lot of variability, while most FL frameworks generally perform well when tested for model test accuracy and federated training speed. An unexpectedly large variance was observed in the Training Speed criterion, with times ranging from under one minute to over three minutes. This may be explained by the different architecture of the FL frameworks and sequential and parallel computing approaches in simulation mode. Overall, the three FL frameworks (FATE AI, FedLearner, PaddleFL) for which no test application could be created are big outliers in this category. These three frameworks consequently also received the lowest total score, as displayed in Fig. 4 on page 15.

Furthermore, there are specific use cases for which some frameworks may be particularly suitable. FLARE is being developed by the same company (NVIDIA) which released Clara, which is an artificial intelligence suite focused on medical use cases. It may therefore be argued that FLARE profits from experiences made during the development of Clara. Meanwhile, FedML provides a website with an FL dashboard, where projects can be tracked and shared with collaborators, allowing for easy deployment, and sharing of applications. This may be advantageous when developing an FL applications across organizations. Furthermore, an extension for FATE called FATE-LLM has been released, targeting development of large language models in a federated setting, giving FATE a strong foundation in this area [127].

It can be concluded that the evaluated FL frameworks are relatively homogeneous regarding the criteria in the Features category. Support for a rollout on edge devices in the Interoperability category and differences in the availability and quality of documentation in the User Friendliness category are the major reasons for the variance in total score between the FL frameworks. To attract practitioners to their FL frameworks, these two aspects need to be most urgently improved by the underperforming FL frameworks.

5.4 Result summary

Based on the literature-driven comparison and analysis results, the RQs posed at the beginning of this paper (see Subection 1 on page 1) can be answered as follows:

RQ 1: Which relevant frameworks for FL exist and are open-source? **15** relevant FL frameworks were selected, reduced from a total of **18** identified FL frameworks after applying the inclusion criteria defined in SubSect. 4.1 on page 7. Table 3 on page 9 gives an overview of the selected FL frameworks. These filtered frameworks are all available as open-source software and have community and industry support. The FL frameworks are used as objects of study in the FL framework comparative analysis (see Sect. 5 on page 8).

RQ 2: Which criteria enable a qualitative and quantitative comparison of FL frameworks? The criteria, weights and evaluation schema introduced in Sect. 5.1, summarized in Table 7 on page 20, are used in the comparison in SubSect. 5.2. The criteria include quantitative measures such as Model Accuracy and Training Speed as well as qualitative measures such as the included Security Mechanisms and the quality and scope of the available Documentation. The evaluation schema based on these criteria creates a versatile and comprehensive comparison of FL frameworks.

RQ 3: Which FL framework offers the most added value to practitioners and researchers? Different FL frameworks received the highest scores in each of the three formulated categories (FederatedScope in Features, PySyft,

FedML, Flower, IBM FL and FLARE in Interoperability and EasyFL in User Friendliness). This indicates that one of several FL Frameworks might provide the most added value depending on one's preferences and needs regarding a particular project. The criteria, their weights and the presented result can in this case act as guidelines for FL framework selection. However, based on the comparative results (see SubSect. 5.2 on page 11), the FL framework Flower currently offers the most overall added value to practitioners and researchers.

6 Limitations and outlook

In this study, not all currently available FL frameworks are represented, since we formulated inclusion criteria to limit the number of FL framework candidates (see SubSect. 4.1 on page 7). The field of FL frameworks for the proposed comparison suite can be extended to include, for example, proprietary framework candidates that have not been considered in this study. A comparison of these with open-source FL frameworks could provide further interesting insights into the alignment and target audience of each FL framework. Additional experiments with FL frameworks in different FL settings could lead to more comprehensive benchmarking results. The vertical FL and federated transfer learning settings would be possible additions, should more frameworks support these paradigms in the future. Depending on the use case, an adjustment of the criteria weighting might also be required. Therefore, the comparison evaluation schema proposed in this paper can be adapted as desired to reflect the priorities of practitioners and researchers for particular FL projects.

FL is still a niche research field, but the number of scientific papers published each year is steadily increasing (see Fig. 3 on page 6) [128–132]. Based on this trend, we also expect a large number of new FL frameworks to be released in the near future. These emerging FL frameworks can be evaluated and compared to other FL frameworks upon release using the comparison methodology proposed in this paper.

7 Conclusion

In this study, a comparison suite to evaluate open-source Federated Learning (FL) frameworks was introduced. For this, a literature review was conducted following the guidelines set by Webster and Watson. The review method involved identifying relevant literature and organizing it based on the most significant concepts discovered through the use of a Latent Dirichlet Allocation (LDA) applied on identified publications relevant to FL. Based on filtered relevant literature, comparison criteria were formulated, and a weighted scoring system has been proposed. The criteria were categorized into the overarching categories of Features, Interoperability, and User Friendliness. Additionally, two inclusion criteria, namely the open-source availability and community popularity were established to narrow down the number of FL frameworks under consideration. This enabled us to conduct a more detailed comparison and evaluation of 15 relevant open-source FL frameworks as the study subjects. Both qualitative and quantitative aspects of the FL frameworks were compared, and a detailed score was calculated for each FL framework as a percentage. The conducted comparison analysis demonstrated that among the investigated FL frameworks, Flower performed the best, achieving a total score of 84.75%. Other FL framework candidates such as FLARE, FederatedScope, PySyft, FedML, OpenFL, EasyFL, IBM FL, TFF and FedLab also achieved a high total score (at or above 60%) but could not beat Flower in all aspects. Additionally, we observed that FederatedScope performed best in the Features category. PySyft, FedML, Flower, IBM FL and FLARE all received highest scores in the Interoperability category, while EasyFL performed best in the User Friendliness category. The worst performing FL frameworks were FATE AI, PaddleFL and FedLearner with a total score of 38.5%, 35% and 24.75% respectively, because they lacked in the Interoperability and particularily in the User Friendliness category. Due to their limitations, test experiments could not be conducted to accurately measure criteria such as Model Accuracy or Training Speed. While this study demonstrated the superior performance of FL frameworks such as Flower, FLARE or FederatedScope in most baseline scenarios, it is important to note that the priorities and requirements of practitioners and researchers may vary. Therefore, the results of this study can be used primarily as a guiding tool in the FL framework selection process for federated-driven analyses.

Appendix A Supplemental Material

See Figs. 5, 6 and Table 7.

```
Topic Nr.1:
federated 25 | domain 14 | privacy 14 | fl 10 | communication 10 | private 10 | models 9 | distributed 8 | methods
7 | local 7 | information 7 | devices 7 | performance 7 | different 7 | server 6 | public 6 | cross 6 | output 6 |
accuracy 5 | security 5 |

Topic Nr.2:
federated 16 | fl 11 | privacy 9 | local 7 | clients 7 | distributed 6 | devices 6 | client 6 | communication 5 |
server 5 | models 5 | algorithms 5 | security 4 | performance 4 | networks 4 | big 4 | systems 4 | dataset 4 | fra
mework 4 | neural 4 |

Topic Nr.3:
fl 850 | federated 809 | client 422 | privacy 405 | clients 386 | server 361 | local 341 | models 277 | communicat
ion 260 | devices 250 | distributed 210 | global 203 | time 193 | attacks 192 | performance 189 | accuracy 189 | a
ggregation 179 | iid 172 | system 171 | device 170 |

Topic Nr.4:
big 56 | algorithms 52 | local 50 | different 45 | processing 36 | deep 33 | distributed 32 | size 30 | algorithm
30 | privacy 30 | federated 29 | neural 28 | distribution 28 | batch 27 | fl 27 | information 27 | challenges 27 |
datasets 27 | fig 27 | accuracy 26 |

Topic Nr.5:
attack 41 | poisoning 41 | attacks 35 | classi 27 | error 23 | attacker 21 | backdoor 20 | section 20 | compromise
d 19 | fl 18 | gender 18 | accuracy 17 | cation 17 | privacy 17 | average 16 | word 15 | benign 15 | adversary 15
| models 14 | dataset 14 |

Topic Nr.6:
federated 272 | privacy 158 | fl 136 | attack 93 | research 74 | inference 74 | attacks 72 | performance 71 | loca
l 68 | security 67 | distributed 65 | label 65 | systems 63 | models 63 | system 62 | information 61 | communicati
on 56 | server 52 | adversary 51 | datasets 51 |

Topic Nr.7:
federated 23 | privacy 22 | protocol 17 | private 16 | entity 15 | system 15 | datasets 14 | encryption 12 | resol
ution 12 | regression 11 | owners 11 | scheme 10 | dataset 10 | distributed 10 | security 10 | section 9 | algorit
hm 9 | step 9 | preserving 9 | linear 9 |

Topic Nr.8:
enclave 8 | fl 6 | federated 5 | tap 5 | frameworks 4 | models 4 | software 4 | security 4 | framework 4 | server
4 | time 4 | privacy 4 | secure 4 | platform 3 | communication 3 | state 3 | application 3 | attacks 3 | clients 3
| adversary 3 |

Topic Nr.9:
privacy 358 | algorithm 197 | private 194 | database 145 | mechanism 143 | queries 134 | query 103 | probability 9
5 | theorem 88 | differential 83 | distribution 81 | log 76 | function 71 | section 70 | given 67 | local 63 | nois
e 61 | case 60 | bound 57 | output 56 |

Topic Nr.10:
fl 273 | federated 156 | edge 92 | privacy 92 | models 84 | research 83 | devices 79 | network 79 | enclave 79 | s
ecurity 70 | iot 65 | framework 64 | frameworks 59 | server 53 | computing 52 | local 52 | aggregator 48 | applica
tions 46 | ml 45 | global 44 |
```

Fig. 5 List of topics, words and frequencies using LDA

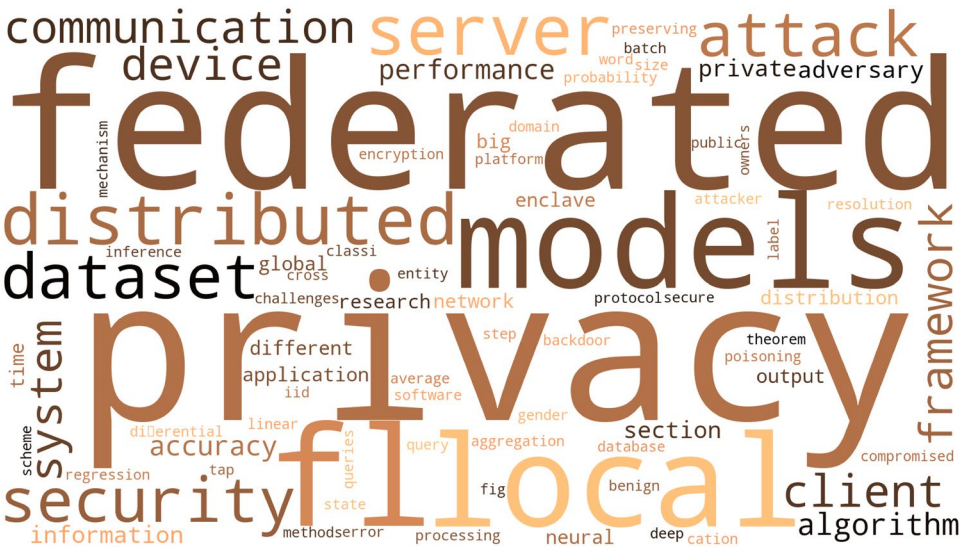

Fig. 6 Graphical representation of the most common words used in the identified literature

Table 7 FL Framework comparison criteria, sorted by category, with associated weighting and evaluation schema

Comparison Criterion	Weighting	Scoring Schema
Category: Features		
Security Mechanisms	35%	0: Neither cryptographic nor algorithmic methods available. 0.5: Either cryptographic or algorithmic methods available. 1: Both cryptographic and algorithmic methods available.
FL Algorithms	25%	0: No out-of-the-box algorithms available. 0.5: Only FedAvg available. 1: Additionally to FedAvg, adaptive optimization algorithms or asynchronous algorithms available.
ML Models	25%	0: No ML library supported. 0.5: One ML library supported. 1: Multiple ML libraries supported.
FL Paradigms	15%	0: Only horizontal FL available. 0.5: N.A. 1: Horizontal FL and vertical FL
Category: Interoperability		
Rollout To Edge Devices	50%	0: No rollout to edge devices possible. 0.5: Rollout with restrictions possible. 1: Full rollout possible.
OS support	25%	0: Neither Windows or MacOS supported. 0.5: Either Windows or MacOS supported. 1: Windows and MacOS supported.
GPU Support	15%	0: No GPU support. 0.5: N.A. 1: GPU support.
Docker Installation	10%	0: No Docker installation possible. 0.5: N.A. 1: Docker installation possible.
Category: User Friendliness		
Development Effort	25%	0: Group of frameworks with most development effort needed. 0.5: Group of frameworks with moderate development effort needed. 1: Group of frameworks with least development effort needed.
Model Accuracy	25%	0: Model accuracy below 50%. 0.5: Model accuracy between 50 and 90%. 1: Model accuracy above 90%.
Documentation	20%	0: Group of frameworks with little available documentation. 0.5: Group of frameworks with moderate amount of available documentation. 1: Group of frameworks with high amount of available documentation.
Training Speed	10%	0: Training time of more than three minutes. 0.5: Training time between one and three minutes. 1: Training time of under one minute.
Data Preparation Effort	5%	0: Group of frameworks with relatively high data preparation requirement. 0.5: Group of frameworks with moderate data preparation requirement. 1: Group of frameworks with low data preparation requirement.
Model Evaluation	5%	0: No built-in model evaluation. 0.5: Built-in model evaluation with difficulties in implementation. 1: Built-in model evaluation with no difficulties.
Pricing Systems	5%	0: Additional features hidden behind paywall. 0.5: N.A. 1: All features freely available.

Funding Open Access funding enabled and organized by Projekt DEAL.

Data availability The MNIST [111, 112] proxy dataset that supports the findings of this study is openly available in http://yann.lecun.com/exdb/mnist/

Declarations

Conflict of interest The authors have no Conflict of interest to declare that are relevant to the content of this article and there are no financial interests

Ethical approval The data and models used are purely for scientific purposes and do not replace a clinical COVID-19 diagnosis by medical specialists

References

1. McMahan HB, Moore E, Ramage D, Hampson S, Arcas BA (2017) Communication-efficient learning of deep networks from decentralized data. J Mach Learn Res 54:1273–1282
2. Abadi M, Chu A, Goodfellow I, McMahan HB, Mironov I, Talwar K, Zhang L (2016) Deep learning with differential privacy. 23rd ACM conference on computer and communications security (CCS 2016), 308–318. https://doi.org/10.1145/2976749.2978318
3. Hard A, Rao K, Mathews R, Beaufays F, Augenstein S, Eichner H, Kiddon C, Ramage D (2018) Federated learning for mobile keyboard prediction arXiv:1811.03604
4. Li T, Sahu AK, Talwalkar A, Smith V (2020) Federated learning: challenges, methods, and future directions. IEEE Signal Process Mag 37:50–60. https://doi.org/10.1109/MSP.2020.2975749
5. Kairouz P, McMahan HB, Avent B, Bellet A, Bennis M, Bhagoji AN, Bonawitz KA, Charles Z, Cormode G, Cummings R, D'Oliveira RGL, Rouayheb SE, Evans D, Gardner J, Garrett Z, Gascon A, Ghazi B, Gibbons PB, Gruteser M, Harchaoui Z, He C, He L, Huo Z, Hutchinson B, Hsu J, Jaggi M, Javidi T, Joshi G, Khodak M, Konecny J, Korolova A, Koushanfar F, Koyejo S, Lepoint T, Liu Y, Mittal P, Mohri M, Nock R, Ozgur A, Pagh R, Raykova M, Qi H, Ramage D, Raskar R, Song D, Song W, Stich SU, Sun Z, Suresh AT, Tramer F, Vepakomma P, Wang J, Xiong L, Xu Z, Yang Q, Yu FX, Yu H, Zhao S (2021) Advances and open problems in federated learning. Found Trends Mac Learn 14:1–121. https://doi.org/10.1561/2200000083
6. Zhang L, Zhu T, Xiong P, Zhou W, Yu P (2023) A robust game-theoretical federated learning framework with joint differential privacy. IEEE Trans Knowl Data Eng 35:3333–3346. https://doi.org/10.1109/TKDE.2021.3140131
7. Jin H, Bai D, Yao D, Dai Y, Gu L, Yu C, Sun L (2023) Personalized edge intelligence via federated self-knowledge distillation. IEEE Trans Parallel Distrib Syst 34:567–580. https://doi.org/10.1109/TPDS.2022.3225185
8. Nguyen DC, Pham Q-V, Pathirana PN, Ding M, Seneviratne A, Lin Z, Dobre O, Hwang W-J (2022) Federated learning for smart healthcare: a survey. ACM Comput Surv 55:1–37
9. Antunes RS, da Costa CA, Küderle A, Yari IA, Eskofier B (2022) Federated learning for healthcare: systematic review and architecture proposal. ACM Trans Intell Syst Technol 13:1–23
10. Xing H, Xiao Z, Qu R, Zhu Z, Zhao B (2022) An efficient federated distillation learning system for multi-task time series classification. IEEE Trans Instrum Meas 71:1–12. https://doi.org/10.1109/TIM.2022.3201203
11. Riedel P, von Schwerin R, Schaudt D, Hafner A, Resnetfed S (2023) Federated deep learning architecture for privacy-preserving pneumonia detection from covid-19 chest radiographs. J Healthcare Inf Res 7:203–224
12. Rahman A, Hossain MS, Muhammad G, Kundu D, Debnath T, Rahman M, Khan MSI, Tiwari P, Band SS (2023) Federated learning-based ai approaches in smart healthcare: concepts, taxonomies, challenges and open issues. Clust Comput 26:2271–2311. https://doi.org/10.1007/s10586-022-03658-4
13. Bharati S, Mondal MRH, Podder P, Prasath VBS (2022) Federated learning: applications, challenges and future directions. Int J Hybrid Intell Syst 18:19–35
14. Witt L, Heyer M, Toyoda K, Samek W, Li D (2023) Decentral and incentivized federated learning frameworks: a systematic literature review. IEEE Internet Things J 10:3642–3663
15. Xiao Z, Xu X, Xing H, Song F, Wang X, Zhao B (2021) A federated learning system with enhanced feature extraction for human activity recognition. Knowl-Based Syst. https://doi.org/10.1016/j.knosys.2021.107338
16. Boobalan P, Ramu SP, Pham QV, Dev K, Pandya S, Maddikunta PKR, Gadekallu TR, Huynh-The T (2022) Fusion of federated learning and industrial internet of things: a survey. Comput Netw **212**
17. Pandya S, Srivastava G, Jhaveri R, Babu MR, Bhattacharya S, Maddikunta PKR, Mastorakis S, Thippa MJP, Gadekallu R (2023) Federated learning for smart cities: a comprehensive survey. Sustain Energy Technol Assess 55:2–13
18. Zhang T, Gao L, He C, Zhang M, Krishnamachari B, Avestimehr AS (2022) Federated learning for the internet of things: applications, challenges, and opportunities. IEEE Internet Things Mag 5:24–29
19. Zhang K, Song X, Zhang C, Yu S (2021) Challenges and future directions of secure federated learning: a survey. Front Comput Sci 16:1–8
20. Li C, Zeng X, Zhang M, Cao Z (2022) Pyramidfl: a fine-grained client selection framework for efficient federated learning. Proceedings of the 28th annual international conference on mobile computing and networking 28, 158–171
21. Huang W, Ye M, Du B (2022) Learn from others and be yourself in heterogeneous federated learning. 2022 IEEE/CVF conference on computer vision and pattern recognition (CVPR)
22. Wen J, Zhang Z, Lan Y, Cui Z, Cai J, Zhang W (2023) A survey on federated learning: challenges and applications. Int J Mach Learn Cybern 14:513–535. https://doi.org/10.1007/s13042-022-01647-y
23. Guendouzi BS, Ouchani S, Assaad HE, Zaher ME (2023) A systematic review of federated learning: challenges, aggregation methods, and development tools. J Netw Comput Appl. https://doi.org/10.1016/j.jnca.2023.103714
24. Zhao Y, Li M, Lai L, Suda N, Civin D, Chandra V (2018) Federated learning with non-iid data arXiv:1806.00582
25. Almanifi ORA, Chow C-O, Tham M-L, Chuah JH, Kanesan J (2023) Communication and computation efficiency in federated learning: a survey. Internet Things 22:100742

26. Xu C, Qu Y, Xiang Y, Gao L (2023) Asynchronous federated learning on heterogeneous devices: a survey. Comput Sci Rev. https://doi.org/10.1016/j.cosrev.2023.100595
27. Qi P, Chiaro D, Guzzo A, Ianni M, Fortino G, Piccialli F (2023) Model aggregation techniques in federated learning: a comprehensive survey. Futur Gener Comput Syst 150:272–293. https://doi.org/10.1016/j.future.2023.09.008
28. Li Q, Diao Y, Chen Q, He B (2022) Federated learning on non-iid data silos: an experimental study. 2022 IEEE 38th iInternational conference on data engineering (ICDE)
29. Wang Z, Xu H-Z, Xu Y, Jiang Z, Liu J, Chen S (2024) Fast: enhancing federated learning through adaptive data sampling and local training. IEEE Trans Parallel Distrib Syst 35:221–236. https://doi.org/10.1109/TPDS.2023.3334398
30. Abreha HG, Hayajneh M, Serhani MA (2022) Federated learning in edge computing: a systematic survey. Sensors 22:450
31. Ticao Zhang SM (2021) An introduction to the federated learning standard. GetMobile Mobile Comput Commun 25:18–22
32. Beltrán ETM, Pérez MQ, Sánchez PMS, Bernal SL, Bovet G, Pérez MG, Pérez GM, Celdrán AH (2023) Decentralized federated learning: fundamentals, state of the art, frameworks, trends, and challenges. IEEE Commun Surv Tutorials 25:2983–3013. https://doi.org/10.1109/COMST.2023.3315746
33. Yang Q, Liu Y, Chen T, Tong Y (2019) Federated machine learning: concept and applications. ACM Trans Intell Syst Technol 10:1–19. https://doi.org/10.1145/3298981
34. Gong X, Chen Y, Wang Q, Kong W (2023) Backdoor attacks and defenses in federated learning: state-of-the-art, taxonomy, and future directions. IEEE Wirel Commun 30:114–121. https://doi.org/10.1109/MWC.017.2100714
35. Dwork C, Roth A (2014) The algorithmic foundations of differential privacy. Found Trends Theor Comput Sci 9:211–407. https://doi.org/10.1561/0400000042
36. McMahan HB, Ramage D, Talwar K, Zhang L (2018) Learning differencially private recurrent language models. International Conference on Learning Representations
37. Shaheen M, Farooq MS, Umer T, Kim B-S (2022) Applications of federated learning; taxonomy, challenges, and research trends. Electronics 11:670
38. Rodríguez-Barroso N, Jiménez-López D, Luzón MV, Herrera F, Martínez-Cámara E (2023) Survey on federated learning threats: concepts, taxonomy on attacks and defences, experimental study and challenges. Inf Fusion 90:148–173
39. Cummings R, Gupta V, Kimpara D, Morgenstern JH (2019) On the compatibility of privacy and fairness. Adjunct publication of the 27th conference on user modeling, adaptation and personalization, 309–315 https://doi.org/10.1145/3314183.3323847
40. Kusner MJ, Loftus JR, Russell C, Silva R (2017) Counterfactual fairness. 31st conference on neural iInformation processing systems 30, 4069–4079
41. Ding J, Tramel E, Sahu AK, Wu S, Avestimehr S, Zhang T (2022) Federated learning challenges and opportunities: an outlook. ICASSP 2022 - 2022 IEEE iInternational conference on acoustics, speech and signal processing (ICASSP)
42. Buolamwini J, Gebru T (2018) Gender shades: intersectional accuracy disparities in commercial gender classification. Proc Mach Learn Res 81:1–15
43. Zhang X, Kang Y, Chen K, Fan L, Yang Q (2023) Trading off privacy, utility, and efficiency in federated learning. ACM Trans Intell Syst Technol 14:98–18931. https://doi.org/10.1145/3595185
44. Khan M, Glavin FG, Nickles M (2023) Federated learning as a privacy solution - an overview. Procedia Comput Sci 217:316–325. https://doi.org/10.1016/j.procs.2022.12.227
45. Webster J, Watson RT (2002) Analyzing the past to prepare for the future: writing a literature review. MIS Q 26(2),
46. He C, Li S, So J, Zhang M, Wang H, Wang X, Vepakomma P, Singh A, Qiu H, Shen L, Zhao P, Kang Y, Liu Y, Raskar R, Yang Q, Annavaram M, Avestimehr S (2020) Fedml: a research library and benchmark for federated machine learning arXiv:2007.13518
47. Barroso NR, Stipcich G, Jimenez-Lopez D, Ruiz-Millan JA, Martinez-Camara E, Gonzalez-Seco G, Luzon MV, Veganzones MA, Herrera F (2020) Federated learning and differential privacy: software tools analysis, the sherpa.ai fl framework and methodological guidelines for preserving data privacy. Inf Fusion 64:270–292
48. Ludwig H, Baracaldo N, Thomas G, Zhou Y, Anwar A, Rajamoni S, Ong YJ, Radhakrishnan J, Verma A, Sinn M, Purcell M, Rawat A, Minh TN, Holohan N, Chakraborty S, Witherspoon S, Steuer D, Wynter L, Hassan H, Laguna S, Yurochkin M, Agarwal M, Chuba E, Abay A (2020) Ibm federated learning: an enterprise framework white paper v0.1 arXiv:2007.10987
49. Reina GA, Gruzdev A, Foley P, Perepelkina O, Sharma M, Davidyuk I, Trushkin I, Radionov M, Mokrov A, Agapov D, Martin J, Edwards B, Sheller MJ, Pati S, Moorthy PN, Wang HS, Shah P, Bakas S (2021) Openfl: an open-source framework for federated learning arXiv:2105.06413
50. Liu Y, Fan T, Qian Xu TC, Yang Q (2021) Fate: an industrial grade platform for collaborative learning with data protection. J Mach Learn Res 22:1–6
51. Beutel DJ, Topal T, Mathur A, Qiu X, Parcollet T, Lane ND (2020) Flower: a friendly federated learning research framework arXiv:2007.14390
52. Dimitriadis D, Garcia MH, Diaz DM, Manoel A, Sim R (2022) Flute: a scalable, extensible framework for high-performance federated learning simulations arXiv:2203.13789
53. Xie Y, Wang Z, Gao D, Chen D, Yao L, Kuang W, Li Y, Ding B, Zhou J (2023) Federatedscope: a flexible federated learning platform for heterogeneity. Proc VLDB Endowment 16: 1000–1012. https://doi.org/10.14778/3579075.3579076
54. Zeng D, Liang S, Hu X, Wang H, Xu Z (2023) Fedlab: a flexible federated learning framework. J Mach Learn Res 24:1–7
55. Zhuang W, Gan X, Wen Y, Zhang S (2022) Easyfl: a low-code federated learning platform for dummies. IEEE Internet Things J 9:13740–13754. https://doi.org/10.1109/JIOT.2022.3143842
56. FedAI: what is FATE? https://fate.fedai.org/overview/ Accessed 20 Feb 2024
57. PaddlePaddle: GitHub Repository PaddlePaddle/PaddleFL. https://github.com/PaddlePaddle/PaddleFL Accessed 20 Feb 2024
58. NVIDIA: NVIDIA Clara: an application framework optimized for healthcare and life sciences developers. https://developer.nvidia.com/clara Accessed 30 May 2023
59. IBM Research: IBM Federated Learning. https://ibmfl.res.ibm.com Accessed 20 Feb 2024
60. ByteDance: GitHub Repository FedLearner. https://github.com/bytedance/fedlearner Accessed 20 Feb 2024
61. Liu J, Huang J, Zhou Y, Li X, Ji S, Xiong H, Dou D (2022) From distributed machine learning to federated learning: a survey. Knowl Inf Syst 64:885–917
62. Kholod I, Yanaki E, Fomichev D, Shalugin ED, Novikova E, Filippov E, Nordlund M (2021) Open-source federated learning frameworks for iot: a comparative review and analysis. Sensors 21:167–189. https://doi.org/10.3390/s21010167
63. TensorFlow: TensorFlow Federated: Machine Learning on Decentralized Data. https://www.tensorflow.org/federated Accessed 20 Feb 2024
64. OpenMined: OpenMined. https://www.openmined.org Accessed 20 Feb 2024
65. Sherpa.ai: Sherpa.ai: Privacy-Preserving Artificial Intelligence. https://www.sherpa.ai Accessed 20 Feb 2024

66. Liu X, Shi T, Xie C, Li Q, Hu K, Kim H, Xu X, Li B, Song D (2022) Unifed: a benchmark for federated learning frameworks arXiv:2207.10308
67. SciKitLearn: Latent Dirichlet Allocation. https://scikit-learn.org/stable/modules/generated/sklearn.decomposition.LatentDirichletAllocation.html Accessed 24 April 2023
68. OpenAI: OpenAI: Pricing. https://openai.com/pricing Accessed 20 Feb 2024
69. Microsoft: FLUTE: a scalable federated learning simulation platform. https://bit.ly/3KnvugJ Accessed 20 Feb 2024
70. Caldas S, Duddu SMK, Wu P, Li T, Konečný J, McMahan HB, Smith V, Talwalkar A (2018) Leaf: a benchmark for federated settings
71. Lai F, Dai Y, Singapuram S, Liu J, Zhu X, Madhyastha H, Chowdhury M Fedscale: Benchmarking model and system performance of federated learning at scale. Proceedings of the 39th international conference on machine learning **162** (2022)
72. FederalLab: GitHub Repository OpenFed. https://github.com/FederalLab/OpenFed Accessed 20 Feb 2024
73. OpenMined: GitHub Repository OpenMined/PySyft. https://github.com/OpenMined Accessed 20 Feb 2024
74. FedAI: GitHub Repository FedAI/FATE. https://github.com/FederatedAI/FATE Accessed 20 Feb 2024
75. FedML: FedML: The Federated Learning/Analytics and Edge AI Platform. https://fedml.ai Accessed 20 Feb 2024
76. FedML: GitHub Repository FedML-AI. https://github.com/FedML-AI Accessed 20 Feb 2024
77. Adap: Adap: Fleet AI. https://adap.com/en Accessed 20 Feb 2024
78. Adap: GitHub Repository Adap/Flower. https://github.com/adap/flower Accessed 20 Feb 2024
79. TensorFlow: GitHub Repository TensorFlow/Federated. https://github.com/tensorflow/federated Accessed 20 Feb 2024
80. Baidu research: Baidu PaddlePaddle releases 21 new capabilities to accelerate industry-grade model development. http://research.baidu.com/Blog/index-view?id=126 Accessed 07 Aug 2023
81. Intel: GitHub Repository Intel/OpenFL. https://github.com/intel/openfl Accessed 20 Feb 2024
82. University of Pennsylvania: CBICA: The Federated Tumor Segmentation (FeTS) Initiative. https://www.med.upenn.edu/cbica/fets/ Accessed 24 Aug 2022
83. IBM: GitHub Repository IBM Federated Learning. https://github.com/IBM/federated-learning-lib Accessed 20 Feb 2024
84. NVIDIA: GitHub Repository NVIDIA FLARE. https://github.com/NVIDIA/NVFlare Accessed 20 Feb 2024
85. Dogra, P.: Federated learning with FLARE: NVIDIA brings collaborative AI to healthcare and beyond. https://blogs.nvidia.com/blog/2021/11/29/federated-learning-ai-nvidia-flare/ Accessed 02 Aug 2023
86. NVIDIA: NVIDIA FLARE Documentation. https://nvflare.readthedocs.io/en/2.1.1/index.html Accessed 20 Feb 2024
87. Meta Research: GitHub Repository FLSim. https://github.com/facebookresearch/FLSim Accessed 20 Feb 2024
88. Microsoft: GitHub Repository Microsoft FLUTE. https://github.com/microsoft/msrflute Accessed 20 Feb 2024
89. FederatedScope: FederatedScope. https://federatedscope.io Accessed 20 Feb 2024
90. FederatedScope: GitHub FederatedScope. https://github.com/alibaba/FederatedScope Accessed 20 Feb 2024
91. FedLab: GitHub FedLab. https://github.com/SMILELab-FL/FedLab Accessed 20 Feb 2024
92. FedLab: ReadTheDocs FedLab. https://fedlab.readthedocs.io/en/master/ Accessed 20 Feb 2024
93. EasyFL: GitHub EasyFL. https://github.com/EasyFL-AI/EasyFL/tree/master Accessed 20 Feb 2024
94. EasyFL: ReadTheDocs EasyFL. https://easyfl.readthedocs.io/en/latest/ Accessed 20 Feb 2024
95. Bonawitz K, Eichner H, Grieskamp W, Huba D, Ingerman A, Ivanov V, Kiddon C, Koneny J, Mazzocchi S, McMahan B, Overveldt TV, Petrou D, Ramage D, Roselander J (2019) Towards federated learning at scale: system design. Proc Mach Learn Syst 1:374–388
96. Mansour Y, Mohri M, Ro J, Suresh AT (2020) Three approaches for personalization with applications to federated learning arXiv:2002.10619
97. Silva PR, Vinagre J, Gama J (2023) Towards federated learning: an overview of methods and applications. WIREs Data Min Knowl Discov 13:1–23
98. Zhu H, Xu J, Liu S, Jin Y (2021) Federated learning on non-iid data: a survey. Neurocomputing 465:371–390. https://doi.org/10.1016/j.neucom.2021.07.098
99. Nilsson A, Smith S, Ulm G, Gustavsson E, Jirstrand M (2018) A performance evaluation of federated learning algorithms. DIDL '18: Proceedings of the second workshop on distributed infrastructures for deep learning 2, 1–8 . https://doi.org/10.1145/3286490.3286559
100. Asad M, Moustafa A, Ito T, Aslam M (2020) Evaluating the communication efficiency in federated learning algorithms. Proceedings of the 27th ACM symposium on operating systems principles. https://doi.org/10.1109/CSCWD49262.2021.9437738
101. Smith V, Chiang C-K, Sanjabi M, Talwalkar A (2017) Federated multi-task learning. 31st conference on neural information processing systems (NIPS 2017), 4427–4437
102. Lo SK, Lu Q, Wang C, Paik H, Zhu L (2021) A systematic literature review on federated machine learning: from a software engineering perspective. ACM Comput Surv 54(5):1–39. https://doi.org/10.1145/3450288
103. Lyu L, Yu H, Zhao J, Yang Q (2020) Threats to federated learning. Lecture Notes Artif Intell 12500:3–16. https://doi.org/10.1007/978-3-030-63076-8_1
104. Bagdasaryan E, Veit A, Hua Y, Estrin D, Shmatikov V (2020) How to backdoor federated learning. Proceedings of the 23rd international conference on artificial intelligence and statistics, 2938–2948
105. Shejwalkar V, Houmansadr A, Kairouz P, Ramage D (2022) Back to the drawing board: a critical evaluation of poisoning attacks on production federated learning. 2022 IEEE symposium on security and privacy (SP)
106. Fu J, Zhang X, Ji S, Chen J, Wu J, Guo S, Zhou J, Liu AX, Wang T (2022) Label inference attacks against vertical federated learning. Proceedings of the 31st USENIX security symposium **31**
107. Feng S, Yu H (2020) Multi-participant multi-class vertical federated learning arXiv:2001.11154
108. Liu Y, Kang Y, Xing C, Chen T, Yang Q (2020) A secure federated transfer learning framework. IEEE Intell Syst 35(4):70–82. https://doi.org/10.1109/MIS.2020.2988525
109. Docker Inc.: The industry-leading container runtime. https://www.docker.com/products/container-runtime/ Accessed 07 June 2023
110. Fayad M, Schmidt D (1997) Object-oriented application frameworks. Commun ACM 40(10):32–38. https://doi.org/10.1145/262793.262798
111. Ge D-Y, Yao X-F, Xiang W-J, Wen, X-J, Liu, E-C (2019) Design of high accuracy detector for mnist handwritten digit recognition based on convolutional neural network. 2019 12th international conference on intelligent computation technology and automation (ICICTA), 658–662 . https://doi.org/10.1109/ICICTA49267.2019.00145
112. Deng L (2012) The mnist database of handwritten digit images for machine learning research. IEEE Signals Process Mag 29(6):141–142. https://doi.org/10.1109/MSP.2012.2211477
113. Avent B, Korolova A, Zeber D, Hovden T, Livshits B (2017) Blender enabling local search with a hybrid differential privacy model. J Privacy Confid 9, 747–764. DOIurlhttps://doi.org/10.29012/jpc.680

114. Cheu A, Smith AD, Ullman J, Zeber D, Zhilyaev M (2019) Distributed differential privacy via shuffling. IACR Cryptol. ePrint Arch, 375–403 . https://doi.org/10.1007/978-3-030-17653-2_13
115. Roth E, Noble D, Falk BH, Haeberlen A (2019) Honeycrisp: large-scale differentially private aggregation without a trusted core. Proceedings of the 27th ACM Symposium on Operating Systems Principles, 196–210. https://doi.org/10.1145/3341301.3359660
116. Song S, Chaudhuri K, Sarwate AD (2013) Stochastic gradient descent with differentially private updates. 2013 IEEE global conference on signal and information processing, 245–248. https://doi.org/10.1109/GlobalSIP.2013.6736861
117. Masters O, Hunt H, Steffinlongo E, Crawford J, Bergamaschi F (2019) Towards a homomorphic machine learning big data pipeline for the financial services sector. IACR Cryptol. ePrint Arch, 1–21
118. Yao AC-C (1986) How to generate and exchange secrets. Proceedings of the 27th annual symposium on foundations of computer science, 162–167
119. Kaissis G, Ziller A, Passerat-Palmbach J, Ryffel T, Usynin D, Trask A, Lima I, Mancuso J, Jungmann F, Steinborn M-M, Saleh A, Makowski M, Rueckert D, Braren R (2021) End-to-end privacy preserving deep learning on multi-institutional medical imaging. Nat Mach Intell 3(6):473–484. https://doi.org/10.1038/s42256-021-00337-8
120. Subramanyan P, Sinha R, Lebedev IA, Devadas S, Seshia SA (2017) A formal foundation for secure remote execution of enclaves. Proceedings of the 2017 ACM SIGSAC conference on computer and communications security, 2435–2450. https://doi.org/10.1145/3133956.3134098
121. Hardy S, Henecka W, Ivey-Law H, Nock R, Patrini G, Smith G, Thorne B (2017) Private federated learning on vertically partitioned data via entity resolution and additively homomorphic encryption arXiv:1711.10677
122. Nikolaenko V, Weinsberg U, Ioannidis S, Joye M, Boneh D, Taft N (2013) Privacy-preserving ridge regression on hundreds of millions of records. 2013 IEEE symposium on security and privacy, 334–348. https://doi.org/10.1109/SP.2013.30
123. So J, He C, Yang C-S, Li S, Yu Q, Ali RE, Guler B, Avestimehr S (2022) Lightsecagg: a lightweight and versatile design for secure aggregation in federated learning. Proc Mach Learn Syst 4:694–720
124. Li T, Sahu AK, Zaheer M, Sanjabi M, Talwalkar A, Smith V (2020) Federated optimization in heterogeneous networks. Proc Mach Learn Syst 2:429–450
125. Reddi SJ, Charles Z, Zaheer M, Garrett Z, Rush K, Konečný J, Kumar S, McMahan HB (2021) Adaptive federated optimization. International conference on learning representations ICLR 2021
126. Romanini D, Hall AJ, Papadopoulos P, Titcombe T, Ismail A, Cebere T, Sandmann R, Roehm R, Hoeh MA (2021) Pyvertical: a vertical federated learning framework for multi-headed splitnn. ICLR 2021 Workshop on distributed and private machine learning
127. Fan T, Kang Y, Ma G, Chen W, Wei W, Fan L, Yang Q (2023) Fate-llm: a industrial grade federated learning framework for large language models. Arxiv Preprint
128. Velez-Esteveza A, Ducangeb P, Perezc IJ, Coboc MJ (2022) Conceptual structure of federated learning research field. Procedia Comput Sci 214:1374–1381
129. Farooq A, Feizollah A, Rehman MH (2021) Federated learning research trends and bibliometric analysis. Stud Comput Intell 965:1–19. https://doi.org/10.1007/978-3-030-70604-3_1
130. Gong M, Zhang Y, Gao Y, Qin AK, Wu Y, Wang S, Zhang Y (2024) A multi-modal vertical federated learning framework based on homomorphic encryption. IEEE Trans Inf Forensics Secur 19:1826–1839. https://doi.org/10.1109/TIFS.2023.3340994
131. Caramalau R, Bhattarai B, Stoyanov D (2023) Federated active learning for target domain generalisation. ArXiv **abs/2312.02247**. https://doi.org/10.48550/arXiv.2312.02247
132. Matsuda K, Sasaki Y, Xiao C, Onizuka M (2024) Benchmark for personalized federated learning. IEEE Open J Comput Soc 5:2–13. https://doi.org/10.1109/OJCS.2023.3332351