Landscape of Machine Learning Evolution: Privacy-Preserving Federated Learning Frameworks and Tools

Giang Nguyen[1,5*†], Judith Sáinz-Pardo Díaz[2*†], Amanda Calatrava[3], Lisana Berberi[4], Oleksandr Lytvyn[5], Valentin Kozlov[4], Viet Tran[1], Germán Moltó[3] and Álvaro López García[2]

[1]Institute of Informatics, Slovak Academy of Sciences, Dúbravská cesta 9, Bratislava, 84507, Slovakia.
[2]Institute of Physics of Cantabria, IFCA, CSIC-UC, Avda. los Castros s/n, Santander, 39005, Cantabria, Spain.
[3]Instituto de Instrumentación para Imagen Molecular (I3M), Universitat Politècnica de València (UPV), Camino de Vera S/N, Valencia, 46022, Spain.
[4]Scientific Computing Center (SCC), Karlsruhe Institute of Technology (KIT), Karlsruhe, Germany.
[5]Faculty of Informatics and Information Technologies, Slovak University of Technology in Bratislava, Ilkovičova 2, Bratislava, 84216, Slovakia.

*Corresponding author(s). E-mail(s): giang.nguyen@stuba.sk; sainzpardo@ifca.unican.es;
Contributing authors: amcaar@i3m.upv.es; lisana.berberi@kit.edu; oleksandr.lytvyn@stuba.sk; valentin.kozlov@kit.edu; viet.tran@savba.sk; gmolto@dsic.upv.es; aloga@ifca.unican.es;
†These authors contributed equally to this work.

Abstract

Machine learning is one of the most widely used technologies in the field of Artificial Intelligence. As machine learning applications

become increasingly ubiquitous, concerns about data privacy and security have also grown. The work in this paper presents a broad theoretical landscape concerning the evolution of machine learning and deep learning from centralized to distributed learning, first in relation to privacy-preserving machine learning and secondly in the area of privacy-enhancing technologies. It provides a comprehensive landscape of the synergy between distributed machine learning and privacy-enhancing technologies, with federated learning being one of the most prominent architectures. Various distributed learning approaches to privacy-aware techniques are structured in a review, followed by an in-depth description of relevant frameworks and libraries, more particularly in the context of federated learning. The paper also highlights the need for data protection and privacy addressed from different approaches, key findings in the field concerning AI applications, and advances in the development of related tools and techniques.

Keywords: Machine Learning, Federated Learning, Privacy-Preserving, Privacy Enhancing Technologies, Frameworks and Tools

1 Introduction

The current landscape of Artificial Intelligence (AI) and its subfields Machine Learning (ML) and Deep Learning (DL) development is shifting from a focus on modeling to a focus on the underlying data used to train and evaluate models. The AI community is increasingly recognizing the need for responsible AI development and deployment, especially in the recent context of the European Artificial Intelligence Act (AI Act) entering into force (AI-Act, 2024) to shape Europe's digital future. The advent of the General Data Protection Regulation (GDPR) (GDPR, 2018) as well as the Act on the Protection of Personal Information (APPI) (APPI, 2019), the California Consumer Privacy Act (CCPA) (CCPA, 2020) and many other worldwide regulations have brought about two trends in data analytics:

- The rise of distributed data analytics: The increased adoption of AI, especially ML technologies, has spurred the demand for distributed data analytics that emerges as data sharing between organizations. This has led to the development of novel distributed ML architectures with Federated Learning (FL) (McMahan et al, 2017), (Soykan et al, 2022), emerging as the most prominent approach. However, other architectures, such as Decentralized Learning (DeCL) (Hegedűs et al, 2019) and Split Learning (SL) (MIT, 2023), are also gaining popularity in the field.
- The importance of Privacy-Enhancing Technologies (PETs): In response to growing concerns about data privacy and data security, various privacy-aware techniques are being employed to protect sensitive data. The typical example is in addition to classic anonymization methods, Differential Privacy (DP) has experienced significant growth in recent years and is being incorporated into numerous applications (OpenMinedDP, 2023).

These two complementary trends have become increasingly prominent in the current AI/ML landscape and are poised to remain at the forefront of innovation in the years to come. From automation and beyond, AI and its core ML is already transforming business (Arrieta et al, 2020). However, there are other potential risks associated with it when addressing areas where controls or processes are lacking or inadequate (Clarke, 2019). Risks include not only data disclosure, content control, bias mitigation, interpretability of decisions, and lack of explainability, but also potential drawbacks related to inadequate or absent robust data access protocols. It is crucial to note that traditional assumptions about the form of the data and the origin of the source may not hold in the era of Big Data, emphasizing the need for adaptation. In addition, the effective implementation of responsible AI principles is vital to successfully navigate these challenges. Therefore, organizations that work and seek to

obtain benefits from AI should ensure that their use of AI satisfies a number of rules, criteria, and requirements (Rauniyar et al, 2023) to limit the potential risks mentioned above (Kairouz et al, 2021). It is appropriate to note that advances in AI technology require significant computational power, memory, and data storage, which still limits their accessibility and practicality. The next important remark is that GDPR protects personal identifiable information (PII) from European citizens also in the context of AI, data-driven, and cross-border solutions. Similar silos occur with other types of sensitive data, such as business or security data, which merge ML in a distributed way (Section 2) with data privacy and security protection requirements.

The ubiquitous applications of data mining using ML algorithms also raise concerns about whether data mining compromises privacy. The legitimate use of private data would benefit data mining users and private data owners. It should be noted that the field of privacy preservation techniques is a booming sector, with numerous tools that are continually being developed and updated. This sector is manifested through the utilization of a wide range of technologies, including software and hardware solutions. The distributed learning principles work together with the privacy-preserving principles. However, this combination is rapidly evolving with high research and development dynamics. It comes with a number of advances, as well as obstacles and challenges at various levels of implementation.

In this context, the motivation of our work is to provide pioneering navigation for AI practitioners and data mining projects with respect to privacy preservation. It is difficult for the general professional public to deal with or orient themselves towards two broad complementary trends in data analytics, both from a theoretical point of view and practical approaches. It is even difficult for them to select the right tools from the multitude of frameworks, libraries, and approaches available from all the ML/DL user communities in different applicable areas. Our goal is to help them overcome such difficulties comfortably in understanding AI data privacy to progress faster in the implementation of their projects or in the development of their research.

The scope of this document emphasizes the importance of creating AI systems that are reliable and human-centered. It aims to highlight the need for data protection and privacy, as well as the importance of auditing AI systems to ensure that they are operating as intended. The evolution of AI is dynamic and will continue to be shaped by the needs of society and technological advances. The current focus on distributed ML, data privacy, as well as the responsible AI development is a positive step towards creating AI systems that benefit all of humanity.

The work presented in this paper makes key contributions to the landscape of privacy-aware ML with the focus on the evolution of ML frameworks in general and privacy-preserving FL frameworks in particular. These contributions can be summarized as follows.

- Systematic presentation of the synergy between distributed ML and PETs as the emerging trend. The landscape goes from centralized to distributed

learning architectures through data privacy and data security to PETs presentation, together with their practical implementation.
- Detailed and comprehensive analysis, which allows users and researchers to have a clear reference and novel insights based on PETs towards FL frameworks and libraries to make decisions needed to build their ML applications with special attention to data privacy.
- In-depth review of ML landscape with a special focus on distributed learning and data privacy, analyzing different distributed architectures in addition to FL, such as split learning, ensemble learning among others. Moving on with a systematic review of different PETs, from data masking, data anonymization, through DP to Homomorphic Encryption (HE) and Trusted and Secure Execution Environments (TEEs/SEEs).
- Practical information is concisely detailed on the use, availability, and functionality of different frameworks that make up the state-of-the-art in the research field, from the classic ML/DL frameworks to FL and PETs Python libraries, including differential privacy and cryptography libraries.

In the context of motivation and contributions, the structure of the rest of this work is as follows.

- Section 2 (Fig. 1) presents a concise privacy-aware ML landscape starting from the centralized learning paradigms for machines (Section 2.1) to distributed learning (Section 2.2) and FL (Section 2.2.1).
  - Section 2.4 briefly presents the learning approaches with PETs such as data masking, DP, HE, and TEEs/SEEs with various anonymized, secured and encrypted realizations.
  - Section 2.5 goes in depth with a concise overview of FL and its principles, highlighting how it addresses the challenges of training models on distributed data without centralization.
- Section 3 is oriented to the ML frameworks and tools that compose the current state-of-the-art (Section 3.1). Section 3.2 presents privacy-preserving FL frameworks with supporting cryptography libraries (Section 3.3) and differential privacy ones (Section 3.4). Technologies used in frameworks and libraries in Section 3 are presented in Section 2 in detail.
- Section 4 concludes the landscape of privacy-aware ML, the evolution of privacy-preserving FL frameworks with a short summary of the trend and the future research directions.

2 Landscape of Privacy-Aware Machine Learning

The growing digitalization is coupled with the generation of large volumes of data in several privacy-sensitive sectors (Gadekallu et al, 2021), from industrial and banking data to medical data, among others (Lakhan et al, 2024), (Mohammed et al, 2023). Such data are often analyzed and used as input to

AI/ML/DL models. Therefore, it is essential to focus on their security and privacy, in addition to the inherent security of the analysis and inference techniques used, in order to create trustworthy and reliable intelligent applications. The landscape of the privacy-aware ML presented in this Section follows the diagram in Fig. 1.

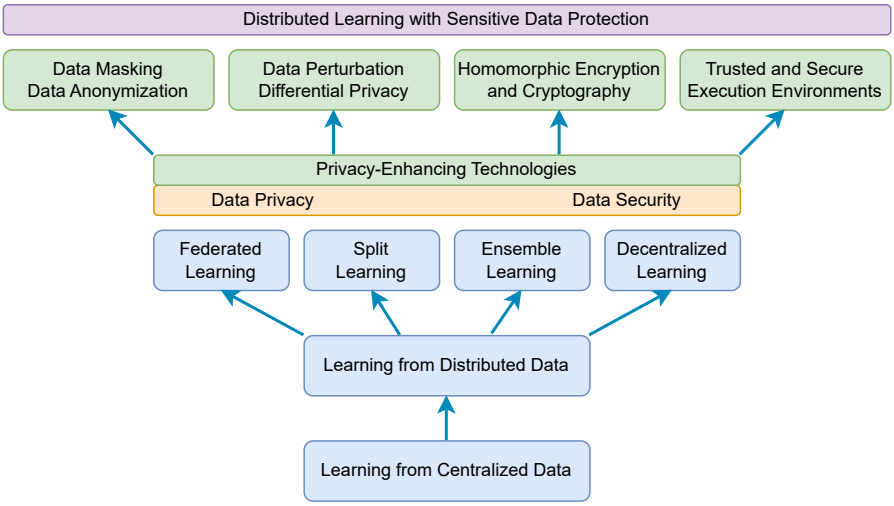

Fig. 1 Landscape of Privacy-Aware Machine Learning.

The *evolution* comes from classical ML towards distributed ML that is accompanied by data privacy and data security concerns and requirements. PETs appear as a response to these emerging needs, leading to distributed learning with sensitive data protection.

2.1 Learning from Centralized Data

In data mining using ML, different learning approaches can be considered, starting from the most classical one: learning from centralized data or centralized learning. In this case, the ML process in many areas of life follows the CRoss Industry Standard Process for Data Mining (CRISP-DM, 1996) (Shearer, 2000), which consists of six steps: (1) business understanding, (2) data understanding, (3) data preparation, (4) modeling, (5) evaluation, and (6) deployment. The whole CRISP-DM cycle is repetitive. It can be divided into two groups: (1) the development phase is the group of the first five steps; and (2) the deployment phase, which is critical for real production.

In 2015, the IBM Analytics Solutions Unified Method for Data Mining/Predictive Analytics (ASUM-DM) (IBM, 2015) has been considered an extended and refined CRISP-DM. It has five phases: (1) Analyze, (2) Design, (3) Configure and Build, (4) Deploy, and (5) Operate and Optimize. Similarly, in 2016,

the Microsoft Team Data Science Process (TDSP) (Microsoft, 2016), considers only five phases that are executed iteratively: (1) Business Understanding, (2) Data Acquisition and Understanding, (3) Modeling, (4) Deployment, and (5) Customer Acceptance.

All of these methodologies have similar phases with minor differences in the decomposition of particular phases. The whole ML lifecycle (Nguyen, 2022) could be generalized and divided into several stages:

1. Analysis and requirements stage,
2. Data-oriented stage,
3. Model-oriented stage,
4. Operation stage.

In scenarios in which we want to train ML/DL models with data from different clients or data owners, the most intuitive and widely applied approach is the centralized one. The idea is to centralize all available data in a single location to train the model using all the data stored. Thus, the trained model will have better generalizability, as it has been trained with more data from different data owners (AbdulRahman et al, 2020).

The main problem with this approach can be related to the integrity, privacy, and security of the data since they must be centralized for training purposes, so they will leave the device or institution that manages them. However, this is not the only issue, as centralizing data can sometimes be unfeasible from a technical or connectivity point of view or even due to legal restrictions.

In this sense, the need arises to develop a learning architecture to deal with all these issues. On the one hand, we can solve problems related to computational issues using distributed learning. Regarding data security and privacy, in recent years there has been a boom in privacy-preserving ML architectures, which will be discussed in detail in the following, among which FL is particularly prominent (Asad et al, 2023).

2.2 Learning from Distributed Data

The major breakthroughs in the field of AI with ML at the core are derivatives of the collection of a large amount of data in one place. For example, ImageNet (StanfordVisionLab, 2020), a dataset with 14 million annotated images, encourages the development of state-of-the-art image classification neural networks. Other examples are the GPT (Brown et al, 2020) and Galactica (Taylor et al, 2022) models. In addition, they are robust Large Language Models (LLMs) trained on 45 TB of textual data crawled from all over the Internet. To train this type of model, it is essential to have large volumes of data, some of which are openly available for free use, while others may contain sensitive information.

Meanwhile, gathering data in one place becomes a more challenging task due to the increasing size, together with stricter privacy and security regulations. The use of distributed data maintains its demand, encouraging the development of collaborative learning approaches from distributed data

sources. Collaborative learning is a situation in which two or more parties learn or attempt to learn together (Yang, 2023).

The idea of learning from distributed data presents two new challenges (Verbraeken et al, 2020), (Su et al, 2022):

- the efficient parallelization of the training process across private networks;
- the creation of robust, coherent, and privacy-aware models.

In this work, the term *distributed learning* is used with the same meaning as learning from distributed data. The following subsections explore different distributed learning architectures such as Federated Learning (FL), Split Learning (SL), Ensemble Learning (EL), Decentralized Learning (DecL) in more detail.

2.2.1 Federated Learning

Federated Learning (FL) is a collaborative and decentralized approach to ML, based on the idea of data decentralization. The main idea in FL in the DL context (Shokri and Shmatikov, 2015) is closely related to Privacy-Preserving ML (PPML) (Dua and Du, 2016). FL was first introduced by McMahan (McMahan et al, 2017) and refers to a technique that allows the building of data-driven models using distributed data without the need to store them centrally.

There are scenarios in which it is beneficial or even mandatory to isolate different subsets of the training data from each other. The farthest extent of this is when a model needs to be trained on datasets that each live on different machines or clusters and may under no circumstance be co-located or even moved (Wen et al, 2023).

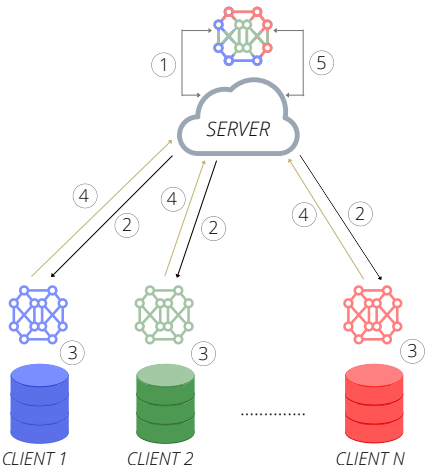

Fig. 2 Federated Learning schema.

The FL architecture includes different steps in which communication between a central server and different clients is established. In this sense, this communication should be encrypted. The steps to perform FL training can be summarized as follows (Sáinz-Pardo Díaz and López García, 2023b): (1) the clients agree on the model to be trained. In some cases, this model can be provided by the server; (2) all the clients receive a copy of the model to be trained (or the weights that define it); (3) the clients train the model locally; (4) the clients send to the server the model (or the update regarding the initial one) obtained after training; (5) the server aggregates all the models (or weights) according to the aggregation strategy chosen; after that, the server sends the updated model to all the clients, and the process is repeated from step (2) as many rounds as established to achieve model convergence. This schema is summarized in Fig. 2.

2.2.2 Split Learning

Another emerging decentralized learning architecture is Split Learning (SL) (Gupta and Raskar, 2018), (Khalifa et al, 2019), which is distributed DL and inference without sharing raw data, as proposed by MIT Media Lab (MIT, 2023). The intuitive idea of the simplest SL configuration is the following: (1) Each client trains a neural network up to a specific layer (cut layer). (2) The output at that layer is sent to the server (or another client, depending on the configuration), which completes the training without seeing the raw data. This completes a round of forward propagation. (3) Gradients are backpropagated from the last layer to the cut layer. (4) Gradients in the cut layer are sent back to the initial client. (5) The rest of the backpropagation is completed by the initial client (Vepakomma et al, 2018). The schema for this architecture is presented in Fig. 3.

In this area, the recent U-Shaped SL is an approach without label sharing to the central server with the highest computing power. The input and output layers are located on the client side, and the inside layers are located on the server side. However, this comes with an increase in communication cost, as gradients need to be transmitted over the network twice: first, the gradients of the input layers are sent from the client to the server, and second, the gradients of the intermediary layers are sent from the server to the output layers of the client (Thapa et al, 2022).

Despite the numerous approaches proposed, SL is still an active research area, and ongoing work aims to further improve the accuracy and efficiency of SL models and explore its potential application in other domains.

2.2.3 Ensemble Learning

With the success of knowledge transfer, recent advancements in ensemble learning (EL) are dominated by the student-teacher learning paradigm. Ensemble learning aggregates the knowledge of multiple teacher models (base estimators) before distilling it into the student model. In the FL context, training models

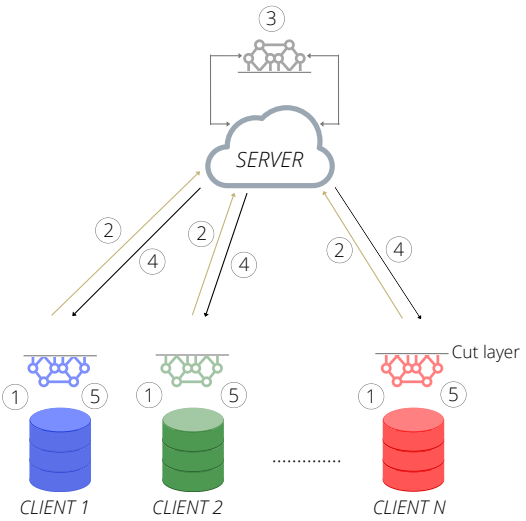

Fig. 3 Split Learning schema.

in a privacy-sensitive context could lead to the use of a Distributed Ensemble Learning (DEL) (Chatterjee and Hanawal, 2022) and federated distillation methods that exchange the output of the model rather than the model parameters (Gong et al, 2022). This allows for perfect separation of the training data subsets, with the drawback that a method must be found that properly balances the output of each trained model for an unbiased result. Server-based parameter systems can be useful in the context of privacy, as the training of a model can be separated from the training result. FL systems can be deployed where multiple parties jointly learn an accurate Deep Neural Network (DNN) while keeping the data themselves local and confidential.

2.2.4 Decentralized Learning

In general, two main distributed ML approaches can be seen as responses to the situation where collecting such data at a central location has become more and more problematic due to novel data protection rules and the increasing public awareness of issues related to data handling.

- The first approach is centralized FL (Section 2.2.1) including SL (Section 2.2.2) and EL (Section 2.2.3) where the parameter server maintains the current model and regularly distributes it to clients who calculate a gradient update and send it back to the server (Yuan et al, 2024). The server then aggregates all updates and redistributes them to all clients. This is repeated until the model converges.
- The second approach is fully decentralized without a parameter server, called Decentralized Learning (DecL) architectures (Hegedűs et al, 2021). In the

FL context, it is called a decentralized FL (DFL) (Beltrán et al, 2023). A prominent architecture is Gossip Learning (GL) (Table 1).

Table 1 Decentralized Learning Architectures

Type	Works	Description
Gossip Learning (GL) architecture	(Blot et al, 2016) (Hegedűs et al, 2019) (Giaretta and Girdzijauskas, 2019) (Hegedűs et al, 2021) (Su et al, 2022)	FL variation in which no central server is required. Instead, clients directly share their updates among them and can choose conditionally or randomly with whom they will communicate. In this approach, the aggregation takes place in a distributed manner. Note that this approach is fully decentralized.
Neighbour Learning architecture	(Hegedűs et al, 2019) (Su et al, 2022)	The architecture is similar to the GL one; each client only communicates with its neighbors. Note that the FL neighbour architectures can also be presented (see Table 3 from (Su et al, 2022)).
All-Reduce architecture	(Patarasuk and Yuan, 2009) (Su et al, 2022)	This architecture simply consists of removing the dependency on the central server of the FL schema.
Ring-All-Reduce architecture	(Jiang et al, 2020) (Yu et al, 2022) (Su et al, 2022)	Alternative to the All-Reduce architecture in order to reduce bandwidth consumption, it consists of setting up the clients on a ring structure.

These architectures are fully decentralized. Nodes exchange and aggregate models directly without being dependent on a central server. Since there is no infrastructure required and there is no single failure point, GL can have significantly lower scalability and better robustness (Hegedűs et al, 2019), (Su et al, 2022). Not being dependent on a central server can avoid privacy issues that arise in cases where the server is not trusted. In this context, different decentralized learning architectures are proposed in addition to the FL one as an alternative to distributed ML performed on different clients, without data communication between them.

However, the verification of learning architectures and their applicability for life problem solving depends on the framework and tools available and their learning and adoption step curves (more details in Section 3.2).

In this Section, a brief presentation of different distributed learning architectures is given. In the next Section 2.3, data privacy and data security are presented as the basis to create coherent and privacy-aware ML models.

2.3 Data Privacy and Data Security

Data became a crucial prerequisite for any advancement in numerous domains. When processed and interpreted accurately, data provide valuable information and insights to drive decision-making, innovation, and continuous progress (Nguyen et al, 2024). In general, a significant portion of the data is already publicly available from many distributed sources, allowing us to have a solid foundation for many intelligent solutions. However, an enormous amount of stored data is distributed and locked in company policies or under data protection and regulations. It is important to note that sensitive information is often contained in data, and data leaks and data failure can cause enormous problems in finance and security (Wen et al, 2023).

Sensitive data are not only personal data. Security information and business data are also sensitive data with protection requirements very similar to those of personal data. Privacy and security are important concerns for handling and protecting data. The definitions of these concepts often overlap; however, they represent two different concepts (May and Denecke, 2022).

- Privacy refers to individuals' ability to control their personal information and determine how it is collected, used, and shared. Privacy also incorporates the right to keep certain information private, maintaining confidentiality and anonymity when desired. These principles also apply in the digital space, where control over sensitive information must be even more precise to prevent potential disclosures of sensitive information.
- Security, on the other hand, focuses on protecting information during all its states from unauthorized access, disruptions, uncontrolled modifications, inspection, copying, and destruction. Security involves implementing technical, organizational, and procedural measures to safeguard data against hacking, malware, human error, and other possible threats. Different technologies can be utilized for each security aspect. For example, access controls and authorization mechanisms prevent unauthorized access, while encryption protects data during storage.

During the data life cycle, data go through three stages: data at rest, data in transit, and data in use (Faridoon and Kechadi, 2020).

- Data at rest are data that do not move actively from device to device or network to network. Examples are data stored on a hard drive, flash drive, or in a database. Data protection at rest aims to secure data stored on any device or network. Data at rest are secured with common symmetric encryption protocols, such as the Advanced Encryption Standard (AES), which have faster encryption and decryption speeds and are suitable for large amounts of data.
- Data in transit or data in motion are data that actively move from one location to another, e.g., over the Internet. Data protection in transit aims to secure these data while traveling or being transferred, e.g., from a local storage device to a cloud storage. Data in transit are mainly secured with

asymmetric encryption protocols, such as Rivest-Shamir-Adleman (RSA), which are used for the implementation of Secure Sockets Layer (SSL) and other secure protocols at the application layer.
- Data in use do not have a widely adopted and standardized security approach (more details are given in Section 2.4.3 and Section 2.4.4). At the same time, in the ML lifecycle, data are used for a considerable period of time, which is enough for sophisticated adversaries to carry out different types of attacks (Kairouz et al, 2021).

Generally, data privacy defines the appropriate handling and use of data, while data security prevents and mitigates data risks from various threats (Bertino, 2016). Decentralized ML techniques, such as distributed learning, FL, SL or GL, among other architectures, aim to address siloed and unstructured data including privacy and regulation of data sharing and incentive models for data transparent ecosystems. In Section 2.4, we will discuss different technologies that can be used to enhance data privacy in various settings.

2.4 Privacy-Enhancing Technologies

The preservation of privacy is almost ubiquitous in various disciplines of information technology, including, but not limited to, financial analysis (Cheng et al, 2020), cybersecurity (Nguyen et al, 2020), and bioinformatics (Xu et al, 2021). It significantly influences cybersecurity with the recent development of information collection and dissemination technologies. The unlimited explosion of new information through the Internet and other media has inaugurated a new era of research in which data mining algorithms should be considered from the point of view of privacy preservation, called privacy-preserving data mining, which is with ML at its core (Privacy-Preserving Machine Learning - PPML). The ubiquitous applications of data mining and ML algorithms allow malicious users to employ data mining to obtain private information, and hence raise the following question: will data mining compromise privacy? This concern can be addressed from two points of view: ethical (in compliance with ethics and regulations in all respects) and technological (as a robust foundation to address bias and to ensure privacy, security, fairness, accountability, transparency, and explainability). The legitimate use of private data would benefit data mining users and private owners. In this regard, it is also important to focus on other techniques that can be applied in combination with the above to add an additional layer of privacy in both data processing and analysis.

Privacy-Enhancing Technologies (PETs) are a wide range of technologies (hardware or software solutions) designed to extract data value to unleash its full commercial, scientific, and social potential, without risking the privacy and security of this information (Heurix et al, 2015). PETs can be divided into the main groups based on their computational functionality as follows:

- Data Masking (DM) and Anonymization: computation on the data to prevent a person from being identified in the database or to avoid the extraction of unauthorized information (Section 2.4.1);

- Data Perturbation and Differential Privacy (DP): computation on noised data or adding noise to the data, the queries or the learning process (Section 2.4.2);
- Homomorphic Encryption (HE) and Cryptography: secure paradigm that enables computation on encrypted data (Section 2.4.3);
- Trusted Execution Environments (TEEs) and Secure Execution Environments (SEEs): computation in secure environments (Section 2.4.4).

The comparison of PETs in the FL context is presented in Table 2 in terms of functionality, strength, and potential weakness. Here is also no one-size-fits-all solution. The complementary approach is needed to ensure the best protection based on the individual data protection requirements of each application or case study.

Table 2 Privacy Enhancing Technologies (PETs) functionality, strength and weakness

	Functionality	**Strength**	**Potential Weakness**
DM	Computation on original data or a generalization of the database	Easy to realize Different pseudonymization and anonymization methods can be applied Anonymity level is under control	Require direct access to original data Information lost
DP	Computation on noised data	Information hiding and output data protection by algorithms Offers quantifiable privacy guarantee Definitions for local and global DP	Trade-off in data accuracy Protection limited by privacy budget Information lost
HE	Computation on encrypted data	Strong input data protection by cryptography algorithms Provides secure data processing	High computational cost for encryption and processing Increased memory requirements Limited functionality for simple functions
TEEs SEEs	Computation in secure environments	Data protection in isolated and secure environments Keep data and code secure during use	Require specialized hardware or support from the e-infrastructure Lack of standards High cost

The comparison of PETs in the FL context is presented in Table 3 in terms of computational complexity, communication cost, accuracy loss, and special hardware requirements. The aim is to understand various strengths and potential weaknesses of PETs on the basis of various characteristics of the technological realization. PETs are rarely used in isolation. They are often combined to create privacy-preserving solutions.

Table 3 Comparison of Privacy Enhancing Technologies (PETs)

	DM	**DP**	**HE**	**TEEs**	**SEEs**
Computational complexity	Low	Low	High	Low	Low
Communication cost	Low	Low	High	Low	Low
Accuracy loss	Depend	High	Low	Low	Low
Information loss	Depend	Depend	Usually no Depend	No	No
Special hardware requirement	No	No	No	Yes	No

The following subsections will go into the details of each PET presented in Table 2 and Table 3.

2.4.1 Data Masking and Anonymization

Data Masking, also called data transformation, considers various techniques to replace sensitive information by adding distracting or misleading data. Examples are anonymization techniques such as *k-anonymity* (Sweeney, 2002), *t-closeness* (Li et al, 2006), *ℓ-diversity* (Machanavajjhala et al, 2007) or among many others (Majeed and Lee, 2020) available in software packages as tools (Sáinz-Pardo Díaz and López García, 2022). It is appropriate to mention that one of the most popular open source packages for data anonymization is a Java-based desktop application called ARX (ARX, 2024). However, the Python toolkit is narrower, currently only with some basic implementations available on GitHub for the most commonly used functions.

The impact of the application of such techniques can be directly reflected in the performance of ML and DL models, so it is important to strike a balance between the level of data privacy and the amount of data to be used in inference processes. This issue has been investigated in different works, for example, with respect to the application of *k-anonymity* (Slijepčević et al, 2021), but also with respect to other methods (Sáinz-Pardo Díaz and López García, 2023a).

This group also contains pseudonymization tools, which replace identifier fields that contain information specific to an individual with fictitious or artificial data using different tools, such as random numbers or hash functions. According to GDPR (GDPR, 2018), anonymized data are not personal data. However, several attacks can be carried out on them to extract information (Bauer and Bindschaedler, 2020).

It is appropriate to note that the boundary between data masking techniques and data perturbation, concretely differential privacy (Section 2.4.2) is not very strict. We separate differential privacy from data masking to highlight its population.

2.4.2 Data Perturbation and Differential Privacy

Data perturbation is a data security technique that adds *noise* to data such as records in databases, allowing the confidentiality of individual records. This technique allows users to obtain summary key information about the data that is not distorted and does not lead to a security breach (Lambert et al, 2023).

Differential Privacy (DP) belongs to the latest methods for preserving privacy, which utilizes data perturbation techniques. The main idea behind DP is that the absence of a single record does not affect the overall characteristics of the dataset. The implementation of DP was proposed in (Dwork et al, 2014) with the additional parameter ε that defines the level of privacy. In particular, this technique is of great interest when combined with DL modeling.

The most well-known formal definition of (ε, δ)-*DP* is the Cynthia Dwork's formula (Dwork et al, 2014) which is summarized below.

Definition 1 A randomized algorithm $\mathcal{M}$ with domain $\mathcal{D}$ is (ε, δ)-differentially private if $\forall S \subseteq \text{Range}(\mathcal{M})$ and $\forall x, y \in \mathcal{D}$ such that $\|x - y\|_1 \leq 1$ (x and y are adjacent inputs), the following equation is verified:

$$\Pr[\mathcal{M}(x) \in S] \leq \exp(\varepsilon) \Pr[\mathcal{M}(y) \in S] + \delta, \tag{1}$$

where:

$\mathcal{M}$ is a random algorithm (or also called the query mechanism);
S is the set of possible outcomes of $\mathcal{M}$;
epsilon (ε) is the privacy budget and presents the maximum distance between $\mathcal{M}(x)$ and $\mathcal{M}(y)$, then $\varepsilon \geq 0$;
delta (δ) is the probability that information is accidentally leaked, then $\delta \in [0, 1]$.

If $\delta = 0$, we say that $\mathcal{M}$ is ε-differentially private or, in short, $(\varepsilon, 0)$-*DP* or ε-*DP*. Otherwise, we use the term (ε, δ)-differential privacy or (ε, δ)-*DP*. $(\varepsilon, 0)$-*DP* controls the amount of privacy protection provided, while, (ε, δ)-*DP* is adding the second layer of privacy protection that represents the maximum probability of privacy violation. The idea is that including or excluding an individual's data should not have a major impact on the outcome of a query or expose specific information about that individual. (ε, δ)-*DP* offers a quantitative measure of privacy guarantees, which denotes the level of privacy protection. (ε, δ)-*DP* guarantees that the information disclosed about individuals remains within acceptable limits by imposing specific constraints on data release techniques, such as adding properly adjusted noise to the query results.

Different methods can be applied to achieve differential privacy, such as the Laplace, Exponential, or Gaussian mechanisms. As an example, let us introduce the classic Laplace mechanism for ε-*DP*.

Definition 2 Be $f : \mathcal{A} \longrightarrow \mathbb{R}^k$, we define the l_1-*sensitivity* as follows:

$$\Delta_1(f) := \max_{\|x-y\|_1} f(x) - f(y)_1 \tag{2}$$

Definition 3 Given any function $f : \mathcal{A} \longrightarrow \mathbb{R}^k$, we define the *Laplace mechanism* ($\mathcal{M}_L$) for achieving ε-*DP* (as given in Rodríguez et al (2020)) as follows:

$$\mathcal{M}_L(x, f(\cdot), \varepsilon) := f(x) + (Y_1, \ldots, Y_k), \tag{3}$$

where $Y_i, \forall i \in \{1, \ldots, k\}$ are independent and identically distributed (i.i.d.) variables from the distribution $Laplace(0, \Delta_1(f)/\varepsilon)$.

DP can be applied to a DL model during training to avoid extracting information from the weights or parameters (Ponomareva et al, 2023). Examples are the application of noise using a Gaussian mechanism during gradient descent (El Ouadrhiri and Abdelhadi, 2022) (Ahmadzai and Nguyen, 2024). All *noisy* model updates are then transmitted to a centralized server. These updates are combined to produce a global model. This model is ultimately improved by repeated iterations of local training and global aggregation while preserving data privacy.

$$f(w) = \frac{1}{K} \sum_{k=1}^{K} f_k(w_k) \tag{4}$$

where:

$f(w)$ is the objective function of FL, also known as the loss function;
K is the number of total clients in the FL system;
w_k indicates the weight of the model in each client;
f_k is the local objective function of the client.

The most challenging problem of the DP mechanism is that the privacy leakage increases due to composition. Determining a tighter bound of the privacy leakage due to composition allows one to learn more features from a data set while protecting individual sensitive information. This leads to the definition of the Rényi divergence (Mironov et al, 2019) (El Ouadrhiri and Abdelhadi, 2022) and guarantees the composition of many steps of a private process.

In FL, it can significantly protect clients' private data from being exposed to adversaries. However, private information can still be divulged by analyzing uploaded parameters from clients, e.g., weights trained in deep neural networks. Therefore, DP is used in FL to effectively prevent information leakage. For example, if it is applied to the weights or gradients of a model prior to sending it to the server in an FL scheme, an additional layer of privacy is added, which can be key, since information from the original data can be extracted from the parameters that define a model (Wei et al, 2020).

In addition, when there are few clients participating in the FL training, another way to add global DP in the FL setting is to do it from the server side when aggregating the weights or the models with the selected aggregation strategy. Thus, if one of the clients acted as an attacker, knowing his/her model or weights update and the aggregated ones, he/she would not be able to infer accurate information from the remaining participants.

DP can be divided into global differential privacy (GDP) and local differential privacy (LDP). GDP adds noise to the output of queries over the entire dataset, whereas LDP adds noise to each individual's data prior to any analysis.

It is appropriate to note that the boundary between differential privacy and cryptographic techniques (Section 2.4.3) is also not strict. Cryptographic algorithms are often applied to implement differential privacy.

2.4.3 Homomorphic Encryption and Cryptography

Homomorphic Encryption (HE) is a cryptography paradigm that allows computation to be performed on encrypted data. HE ensures that performing an operation on encrypted data and decrypting the result is equivalent to performing analogous operations without encryption (Acar et al, 2018), (Lytvyn and Nguyen, 2023a). HE is based on the following two theoretical principles:

- Encryption is a process of transforming the information in a way to hide the information's initial properties.
- Homomorphism, intuitively, is a mathematical function between two objects with the same algebraic structure that preserves the operations on them (structure and basic properties).

In short, HE is a form of encryption that allows computation on encoded data, preserving the initial properties and structure. Theoretically, this technique ensures that operating on the encrypted data and decryption the result is equivalent to performing such operations without enciphering.

Be $(A, \star)$ and $(B, \diamond)$ two groups with $\star$ and $\diamond$ representing the specific operation for each group. Note that a group $(G, +)$ is abelian if and only if it verifies the group axioms and $\forall a, b \in G \; a + b = b + a$.

Definition 4 Given two groups $(A, \star)$ and $(B, \diamond)$, the function $\psi : A \to B$ is a group homomorphism if $\forall a, a' \in A$ is verified:

$$\psi(a \star a') = \psi(a) \diamond \psi(a') \tag{5}$$

In addition, in HE both groups or rings are usually used as algebraic structure. An example is the BGV scheme that uses polynomial rings.

Definition 5 A **ring** is a tern $(R, \star, \diamond)$ verifying:

$(R, \star)$ is an abelian group.
$(R, \diamond)$ is a commutative monoid.
The distributive property is verified as follows:

$$x \diamond (y \star z) = (x \diamond y) \star (x \diamond z), \quad \forall x, y, z \in R, \tag{6}$$

$$(y \star z) \diamond x = (y \diamond x) \star (z \diamond x), \quad \forall x, y, z \in R, \tag{7}$$

Note that the Ring Learning With Errors (RLWE) (Mouchet et al, 2021) computational problem is the key for HE schemes and approaches. We can give a standard definition for HE according to (Yi et al, 2014) as follows:

Definition 6 Be the encryption scheme $ES = (P, C, K, E, D)$, where P is the plaintext, C is the ciphertext, K is the key space, E is the encryption algorithm, and D is the decryption algorithm. Let's denote the group P with operator $\star$ as $(P, \star)$ and the group C with operator $\diamond$ as $(C, \diamond)$.

The encryption algorithm E is a mapping from P to C with a secret or public key $k \in K$ (according to the selected cryptosystem) as follows:

$$E_k : P \to C \tag{8}$$

Then, the encryption scheme ES is **homomorphic** if the following condition is fulfilled:

$$E_k(x \star y) = E_k(x) \diamond E_k(y) \qquad \forall x, y \in P; \qquad k \in K \tag{9}$$

It is important to note that not every encryption scheme has an homomorphic property, and not all encryption schemes that have homomorphic properties have a reasonable level of security. Depending on the number of operations that can be carried out on the encrypted data, there are different homomorphic encryption schemes, such as the following:

- Fully Homomorphic Encryption (FHE): with FHE, businesses can analyze and process sensitive data while maintaining privacy and compliance controls. With this technology, internal or external parties can perform data analysis and processing without requiring data to be exposed (decrypted) (Gentry, 2009), (Hergenrother and Park, 2021). Theoretically, any function can be computed. FHE allows performing any number of operations, so it is very expensive computationally in practice (Lytvyn and Nguyen, 2023b).
- Somewhat Homomorphic Encryption (SWHE): it is a scheme more feasible in practice, but limits the number of operations that can be performed on encrypted data (Mahato and Chakraborty, 2023).
- Partial Homomorphic Encryption (PHE): allows for only one type of operation to be performed (any number of times). It is a straightforward and efficient scheme despite its limitations (Munjal and Bhatia, 2023).

In addition to HE, which is frequently mentioned in the context of FL, other encryption techniques such as symmetric encryption, asymmetric encryption, and hashing are used to ensure data security in their three states (Section 2.3 and Section 2.4.4). In the FL context with HE, Secure Multi-Party Computation (SMPC) is a cryptographic technique that allows two or more parties to securely compute a function over their private data without being able to access the data of the other parties (Zhao et al, 2019), (Lindell, 2020). In other words, SMPC ensures that the private data of each participant remain secure while still allowing the parties to obtain the result of the computation over all the data without ever having to share the data themselves.

In FL architecture, HE can be applied if the central server is not trusted, so that the models or weights are sent in encrypted form to the server in charge of performing the aggregation. Thus, the server would be able to aggregate the weights without knowing the individual updates of each client.

- One problem that arises in this regard is the handling of HE public and private keys, which cannot be the same for all parties. In order to solve this problem, different multi-key HE (MKHE) scheme are proposed, in which all the n keys that are used to encrypt the model or the weight updates in each client must be used to carry out the decryption process (López-Alt et al, 2012) (Chen et al, 2019).
- Secret Sharing (SS) offers a suitable alternative for securing the aggregation process in FL. SS is a scheme of sharing (t, n), where the secret is divided into n shares, such that any $(t-1)$ does not reveal information about s, while any t shares allow a complete reconstruction of secret s (Evans et al, 2018). Multiple protocols based on SS have been proposed in the context of FL, such as *SecAgg* (Bonawitz et al, 2017) and *SecAgg+* (Bell et al, 2020), *FastSecAgg* (Kadhe et al, 2020) and *LightSecAgg* (So et al, 2022), representing the most recent advances in secure aggregation via SS.

The inclusion of different PETs, such as HE, MKHE, SMPC or SS in a FL setting where sensitive data from different clients or data owners are handled, can be key to prevent a malicious party (clients or server) from extracting knowledge from the trained models.

2.4.4 Trusted and Secure Execution Environments

In addition to data masking, differential privacy, and cryptographic techniques, trusted execution environments (TEEs) also belong to PETs. TEEs are secure isolated environments designed to provide a high level of protection for sensitive communication and the storage of confidential data (Chen et al, 2020). The key characteristics of trusted environments are:

- *Trusted* means that the environment is robust against known malicious software, tampering or unauthorized access, and the output of the computation is genuine;
- *Execution* considers the elimination of limitation of software that could be executed in the TEE;
- *Environment* represents the separation of TEE from the existing system and making it available through the strictly defined set of Application Programming Interface (API).

In the context of ML and FL, models are often deployed in the cloud and are accessible to users to make predictions through pre-defined interfaces. To make a prediction, the user must send their data to the cloud to receive the inference results (Lakhan et al, 2021). Consequently, user data could be compromised, leading to the exposure of private data. In general, TEE can be used in the ML inference phase in the cloud. However, the current widely used ML and

DL models could be computationally intensive for the regular environment. The computational load for the same operation in TEEs is significantly larger than in an untrusted environment with increasing execution time and latency. Thus, to take advantage of the benefits of TEE while keeping the computation load at an acceptable level, the ML/FL process must be optimized for TEEs. An example of such optimization in a given context represents split learning (Section 2.2.2).

The work (Narra et al, 2019) proposed the inference framework for the DL model, where the DL model is divided into two parts, one is executed in the TEE and the other in the regular environment. In practice, the framework uses input obfuscation to cooperatively execute between an enclave and an untrusted Central Processing Unit (CPU) in the early layers. This approach optimizes the overall performance of inference while securing the computation in TEE using secure enclaves of Intel Software Guard Extensions (SGX) (Intel, 2023), which is a set of special instruction codes and APIs to improve security built into Intel chips. Another example is the SCONE platform (Scontain, 2023) to build and run secure applications with the help of Intel SGX.

TEE can be divided into hardware and software environments. Secure Execution Environment (SEE) is a related concept to TEE, which mainly highlights security measures to safeguard the execution environment, such as access control, encryption, isolation, and integrity checks (Mo et al, 2021). Although there may be some differences, it should be noted that the terms SEE and TEE are often used interchangeably and that their precise definitions may be influenced by specific context and industry practice. Finally, both terms refer to protected and isolated environments with the goal of ensuring secure execution and security of software with an emphasis on security and trustworthiness.

In this context, Confidential Computing (CC) (Sardar and Fetzer, 2023) is built on the basis of hardware-assisted TEE and focuses on protecting data during processing. CC also provides remote attestation that enables users to verify the integrity and confidentiality of code and data in the execution environment. SEE in CC is often implemented as virtual machines that reduce the surface of attacks, simplify implementation, and allow users to execute native codes in CC without modification. The confidential container (Yang et al, 2021) is defined by the Cloud Native Computing Foundation (CNCF) as an open source project aiming to standardize confidential computing at the container level and simplify its consumption in Kubernetes.

2.5 Distributed Learning with Sensitive Data Protection

FL democratizes data-driven insights by training ML models on distributed data without sharing sensitive data across devices or servers. As ML applications become increasingly ubiquitous, concerns about data privacy and security have also grown. PPML safeguards data confidentiality in compliance with data protection regulations. The aim is to develop models that can operate on encrypted or anonymized data, preventing unauthorized access to sensitive information. By combining decentralized data access for ML training and

PETs, PPML enables organizations to build, deploy, and manage ML models reliably and securely (Mothukuri et al, 2021).

Table 4 Examples of real-world case studies with applied Federated Learning

Work	**Description**	**FL Framework**
(Aditya et al, 2018)	Digital forensics investigation	TensorFlow Privacy
(Taïk and Cherkaoui, 2020)	Electrical load forecasting	TensorFlow Federated
(Hong et al, 2020)	Privacy-preserving ML on genomic data	TensorFlow Encrypted
(Firouzi et al, 2021)	Software-defined networking and edge computing	TensorFlow Federated
(Budrionis et al, 2021)	Predicting in-hospital mortality	OpenMined PySyft
(Novikova et al, 2022)	Intrusion detection in the critical infrastructures	FATE
(Shi et al, 2022)	Tumor segmentation challenge	FEDML NVIDIA Flare
(Chowdhury et al, 2023)	Covid-19 detection using chest X-ray images	Flower
(Lazzarini et al, 2023)	IoT intrusion detection	Flower
(Urmonov et al, 2024)	Object detection for intelligent vehicles	OpenMined PySyft
(Dasari and Kaluri, 2024)	Privacy preserving in financial sectors	Flower

The most prominent distributed learning architecture is FL (Section 2.2.1), which addresses critical issues such as:

- Data privacy, for example, medical records that cannot leave hospitals (Chaddad et al, 2023), (Lakhan et al, 2023);
- Data security, for example, monitoring data that cannot leave the site (Al Ogaili et al, 2023);
- Data privacy and security, for example, bank data cannot leave bank institutes (Alazab et al, 2021);
- Data access rights for heterogeneous distributed data (Ali and Mohammed, 2024).

The list of FL application case studies is getting longer (Li et al, 2020a) (Li et al, 2021) with more and more use cases than in Table 4. The more complex comparative overview of methods and applications with proposed frameworks

and tools can be found in (Silva et al, 2023) and (Riedel et al, 2024), which provides context and demonstrates the progress made.

FL stands out for its ability to train powerful ML models while protecting sensitive data. This ability of FL is achieved through the combination of distributed learning with PETS through proper frameworks and tools (Truong et al, 2021) as follows.

- Data minimization including purpose minimization and storage minimization: FL avoids transferring raw data itself. Model training is performed on distributed clients (or devices), where only processed updates or model parameters are shared with a central server. This significantly reduces the risk of exposing sensitive information compared to traditional centralized data storage and learning.
- Differential privacy (DP) adds noise during the training process or even to the model updates or parameters before sharing them. This adds another layer of privacy protection. DP guarantees that the model does not leak information about a specific individual, even if an attacker compromises the updates or parameters.
- Secure computation: sensitive data often need additional protection even as processed updates or parameters. FL can integrate cryptographic techniques (such as HE, SMPC or SS) to perform calculations on encrypted data without decrypting them. This ensures privacy even from the central server.
- Trusted and secure environments including TEE/SEE are crucial for FL on distributed data at all stages: data at rest, data in transit and data in use.

As mentioned in the Introduction, AI and its core ML are already changing the way business is done. However, there are also other potential risks associated with these technologies when addressing areas where controls or processes are lacking or inadequate, such as data disclosure, content control, bias mitigation, interpretability of decisions, and lack of explainability. The main objective of PPML is to harness the power of ML while protecting the privacy of the data used to train and deploy these models. It aims to achieve a balance between the acquisition of valuable data insights and ensuring that data owners retain control over their right to privacy (Kaissis et al, 2020).

The privacy protection aspect in PPML can be achieved through:

- Prevent identification: PPML techniques aim to prevent anyone, including those involved in training or using the model, from identifying individuals in the data used. This can be achieved through methods such as anonymization and differential privacy.
- Limit data exposure: PPML minimizes the amount of sensitive data that is exposed during model training and deployment. This can involve using only the necessary data features for FL (where models are trained on local devices) and the use of HE that allows computations on encrypted data.
- Control over data privacy and security: PPML techniques can enable individuals to decide what data are used, for what purpose, and for how long in a proper setting.

The concept of decentralized data access for ML can be seen through two main approaches to ensure data privacy and security:

- FL involves a network of devices or servers holding their own private data while training a global model iteratively (more details in Section 2.2.1, Fig. 2). From the data structure point of view and based on (Yang et al, 2019), the FL architecture can be divided into horizontal FL (HFL), vertical FL (VFL) and transfer federated learning (TFL).
- FL with cryptography techniques such as HE involves more parties with their own private datasets to train a model collaboratively without revealing their individual data to each other. HE ensures that the data remain encrypted throughout the process (Section 2.4.3). The model is trained on the combined encrypted data.

From a data security point of view, in HFL, only the server can compromise the privacy of data participants. HFL combines data from entities, institutions, or data owners with the same features but different samples. An example of HLF is the collaboration of two medical institutions. Since their data are very similar, the feature space could be the same (Nasr et al, 2019). VFL assumes semi-honest behavior of the participants; that is, adversaries can only learn from corrupted parties, but not the data from other parties. VFL combines data from entities with the same sample IDs but distinct features. An example of a VFL is two organizations located in one area, such as a bank and an e-commerce company (Luo et al, 2021), which can have data associated with the same individuals, but with different features and records. TFL comes into a place in situations where the two above-mentioned scenarios are not suitable. TFL involves the use of a previously trained model on a similar task to improve the performance of a new model on a new task, such as fine-tuning. TFL can be applied for both VFL and HFL (Ahmadzai and Nguyen, 2023).

The ML effectiveness aspect in PPML is achieved through:

- Maintain model accuracy: PETs should not significantly compromise the accuracy or performance of ML models. Reaching a balance between privacy and accuracy is a key challenge in PPML research.
- Enable diverse data sources: PPML should allow the use of data from multiple sources while protecting data privacy. This is crucial for building robust and generalizable models.
- Scalability and practicality: PPML techniques should be scalable to handle large datasets and be practically implementable in real-world applications.

In general, PPML seeks to build trust in ML by demonstrating that valuable insights can be extracted from data while respecting individual privacy. This is becoming increasingly important as data privacy protection regulations, such as GDPR and CCPA become more stringent, and individuals become more aware of their privacy rights. The principles discussed above work together to create a PPML approach. It is important to note that FL

is promising, fast evolving and has high research and development dynamics. This provides a number of advances, as well as obstacles and challenges at various levels of implementation (Section 3.2). Recent notable trends in FL research and implementation include advancements in secure aggregation techniques to enhance the efficiency of model updates and exploration of FL applications across diverse industry domains.

2.6 Landscape Summary

The evolution of privacy-aware ML involves complicated tasks, such as managing the increasing size of collected data, along with strict privacy and security regulations. The use of distributed data maintains its demand, encouraging the development of federated learning approaches from distributed data sources with modern frameworks and tools (Section 3.2) accompanied by distributed data protection and secure computation requirements in the global regularization context.

From a practical point of view, data masking and data perturbation provide an upper layer for large-scale data mining with privacy preservation. Here is the need to balance data utility against privacy. The more data you perturb to protect privacy, for example, using various differential privacy frameworks and tools (Section 2.4.2), the less accurate your ML algorithm might be. Then, it is crucial to find a point where the data is still useful for analysis without compromising individual privacy.

There are also approaches to hardening data that focus primarily on controlling and limiting data access or acquiring trusted entities to execute computations. Both approaches do not face the problem directly and have their disadvantages, such as who verifies the trusted entity and social engineering to overcome access control.

With the growing importance of digitization in almost all areas of life, it is no wonder that companies around the world are investing in it at a dizzying pace. ML is witnessing significant adoption rates in the day-to-day operations of organizations with ML adoption. This is in part due to the best ML frameworks that have been developed so far (Hari, 2023). In Section 3, we present continuous details since (Nguyen et al, 2019), on the recent evolution of the ML/DL frameworks towards privacy-preserving FL ones in light of technologies presented in Section 2.

3 Evolution of Machine Learning Frameworks

This section analyzes the evolution of the different Python frameworks to move from centralized ML/DL to PPML with a special focus on FL. It also briefly updates the state-of-the-art of ML and DL frameworks with respect to our previous work carried out in (Nguyen et al, 2019), before moving on to privacy-aware FL frameworks and supporting tools.

3.1 Machine Learning Frameworks and Tools

In the following, the most popular and most used frameworks and tools for centralized ML and DL are presented, with the dominance of TensorFlow (in general) and PyTorch (in research) (Table 5 and Fig. 4).

Table 5 Frameworks and tools for centralized ML and DL. GitHub star as of August 14, 2024.

ML/DL Framework	GitHub Stars	ML	DL	Licence and Notes
TensorFlow	185.0K	✓	✓	Apache 2.0
PyTorch	81.3K	✓	✓	a specific open-source license for each module.
Keras	61.4K		✓	Apache 2.0
Scikit-Learn	59.2K	✓		BSD 3-clause
Colossal-AI	38.5K		✓	Apache 2.0, parallelism wrapper for Big DL models rapid growing one
JAX	29.5K	✓	✓	Apache 2.0, numerical computation for ML research
XGBoost	26.0K	✓		Apache 2.0 ensemble learning
fast.ai	26.0K		✓	Apache 2.0 wrapper for PyTorch
PaddlePaddle/Paddle	22.0K		✓	Apache 2.0
deeplearning4j	13.6K		✓	Apache 2.0 DL for JVM
H2O.ai/H2O-3	6.8K	✓	✓	Apache 2.0
MindSpore	4.2K		✓	Apache 2.0 new and growing one

- *TensorFlow* (Tensorflow, 2023a) is an end-to-end open-source platform for ML and DNN. It has a comprehensive and flexible ecosystem of tools, libraries, and community resources that allows building and deploying ML-powered applications. It was originally developed by researchers and engineers working on the Google Brain team to conduct ML/DNN research. Today, TensorFlow is the most widely used DL framework, which allows the creation of optimized static graphs with eager execution to achieve more dynamic behavior.

- *PyTorch* (PyTorch, 2023) follows the motto: *Tensors and Dynamic neural networks in Python with strong GPU acceleration* with its most notable adoption of a Dynamic Computational Graph (DCG). The growing popularity of PyTorch (specially in research) can be clearly seen in Fig. 5, which presents the trend of the different frameworks used in the research implementations listed on the portal *Papers with Code* (PapersWithCode, 2023), focusing on TensorFlow and PyTorch.
- *Keras* (Chollet, 2023) follows the motto: *Deep Learning for humans*. Keras follows best practices for reducing cognitive load: it offers consistent and simple APIs; provides clear and actionable error messages. Keras also has extensive documentation and developer guides. It declares as an exascale ML. Currently built on top of TensorFlow 2, Keras is an industry-strength framework that can scale to large clusters of GPUs or an entire TPU pod. Recently, Keras 3 (Keras3, 2023) is a complete rewrite of Keras that enables the run of Keras workflows on top of JAX, TensorFlow, or PyTorch and that unlocks brand new large-scale model training and deployment capabilities.
- *Scikit-Learn* (Cournapeau, 2023) is an open-source, simple, and efficient tool for predictive data analysis. It is reusable in various contexts. Built on NumPy, SciPy, and Matplotlib for ML. The project was started in 2007 and since then many volunteers have contributed. It is currently maintained by a team of volunteers.
- *Colossal-AI* (Li et al, 2023) provides a tool for writing distributed Big DL models in a unified way. Its motto is: *Making large AI models cheaper, faster, and more accessible*. This framework provides the users with a collection of parallel components aiming to provide tools for training distributed DL models. The framework has rapidly grown the number of GitHub stars in the recent year 2023, which was the year of LLMs.
- *JAX* (GoogleJAX, 2023) is Autograd (HIPS-Autograd, 2023) (maintained, no longer actively developed) and XLA (Accelerated Linear Algebra) (TensorFlow-XLA, 2024) in combination for *high-performance numerical computing, including large-scale ML research*. Despite its increasing popularity, JAX has declined as a research project, still not an official Google product. With its updated version of Autograd, JAX can automatically differentiate native Python and NumPy functions and supports reverse mode differentiation (backpropagation) as well as forward mode differentiation. These two kinds of differentiation can be composed arbitrarily in any order.
- *XGBoost* (XGBoost, 2023) is an optimized distributed gradient boosting library designed to be highly efficient, flexible, and portable. It implements ML algorithms under the Gradient Boosting framework. XGBoost provides parallel tree boosting (also known as GBDT, GBM) that solves many data science problems in a fast and accurate way. The same code runs on major distributed environments (Kubernetes, Hadoop, SGE, MPI, Dask) and can solve problems beyond billions of examples. XGBoost has been developed and used by a group of active community members.

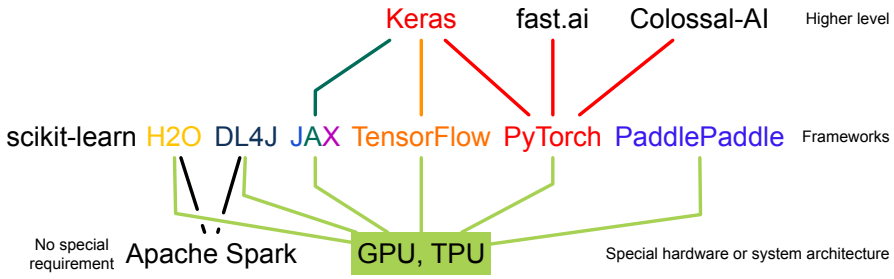

Fig. 4 Machine learning and deep learning frameworks and libraries (2024).

- *PaddlePaddle* (PaddlePaddle, 2023) is an open source Python library (since 2016) that allows DL. It is the first independent DL research and development platform from China. It includes more than 200 pre-trained models, which can help to accelerate development in industrial applications. As in other ML/DL frameworks, it uses tensors to represent data and can be used in distributed learning tasks.
- *Deeplearning4J* (DL4J) (DL4J, 2023) belongs to the Eclipse ecosystem, which is a set of projects designed to support all the needs of a Java Virtual Machine (JVM) based DL application. This means starting with the raw data, loading and preprocessing it, and building and tuning a wide variety of simple and complex DL networks. Because Deeplearning4J runs on the JVM, it can be used within a wide variety of JVM based languages other than Java, such as Scala, Kotlin, Clojure, etc. The DL4J motto is: *Suite of tools for deploying and training DL models using the JVM.*
- *H2O* (H2O, 2023) is a memory platform for distributed scalable ML. It uses familiar interfaces like Python, R, Scala, Java, JSON and the Flow notebook/web interface, and works seamlessly with big data technologies like Hadoop and Spark. H2O provides implementations of many popular ML algorithms, including ensemble learning (XGBoost, Random Forests), as well as DNN algorithms. H2O also provides a fully automatic ML algorithm (H2O AutoML), i.e. unified interfaces to a variety of ML algorithms in H2O.

Fig. 4 presents a core connectivity of the most notable ML/DL frameworks and tools (Table 5) with respect to their hardware and/or system architecture requirements. The lines and their corresponding colors present the support connection from the lower layer to the upper ones.

The landscape of large-scale ML/DL frameworks and tools is dynamic and complex. In addition to the software products mentioned above, many other ML/DL frameworks and tools are listed, for example, *fast.ai* (fast.ai, 2023), which makes DL easier to use or *MindSpore* (MindSpore, 2023), which is a new open-source DL training / inference framework with growing popularity. MindSpore can be used for mobile, edge and cloud scenarios.

On the other hand, we also mention popular DL frameworks such as *CNTK* (CNTK, 2023), which is no longer developed; *Caffe2* (Caffe2, 2023), which was developed by Facebook and now is in the archive; *Chainer* (Chainer,

2023), which is in the maintenance phase with limited development; or *MXNet* (MXNet, 2023a), which has retired recently at the end of 2023 (MXNet, 2023b). MXNet was a flexible and efficient library for DL.

In short summary, the number of ML/DL frameworks and tools is high and is in a dynamic evolving state with the dominance of production grade TensorFlow with Keras built on top of it and PyTorch continues its growing trend as a popular research framework (see Figure 5).

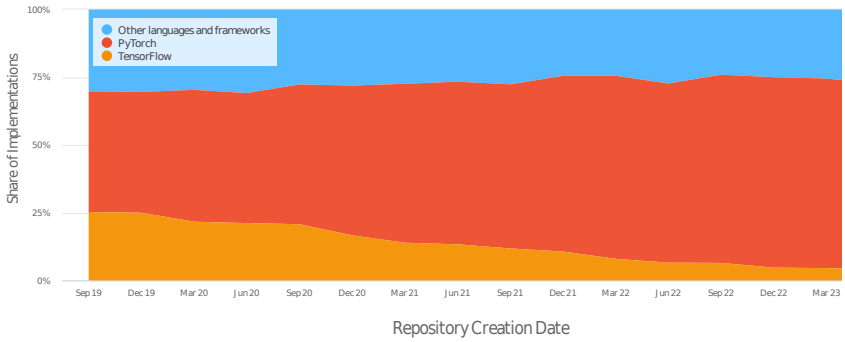

Fig. 5 Research paper implementation grouped by framework (focusing on Tensorflow and PyTorch). Extracted from Papers with code (PapersWithCode, 2023)

In the following sections, we expand the list of ML/DL frameworks from the traditional centralized ones to those supporting distributed and FL, highlighting the main libraries involved. All of these frameworks are privacy-aware and privacy-preserving ML. However, their privacy implementations are on various realization levels, such as planned, partial, or full. Several of them provide more assurance and protection, like secured/encrypted mechanisms or peer-to-peer networking among data owners and data scientists. The landscape is highly dynamic, with development changing, new or planned feature integration, product merging, and so on.

3.2 Privacy-Aware Federated Learning Frameworks

The field of tools and frameworks to implement and deploy privacy-aware machine learning architectures in a distributed and collaborative way is extensive, as summarized in Fig 6, Table 6 and Table 7. The recent focus on privacy preservation in the ML and DL contexts makes this field in continuous evolution. In this section, we present different state-of-the-art frameworks related to Python implementations of PPML techniques. These open-source frameworks enable FL to train models collaboratively in a distributed way without sharing raw data. However, they differ in their focus and insight implementation, as will be exposed in the following.

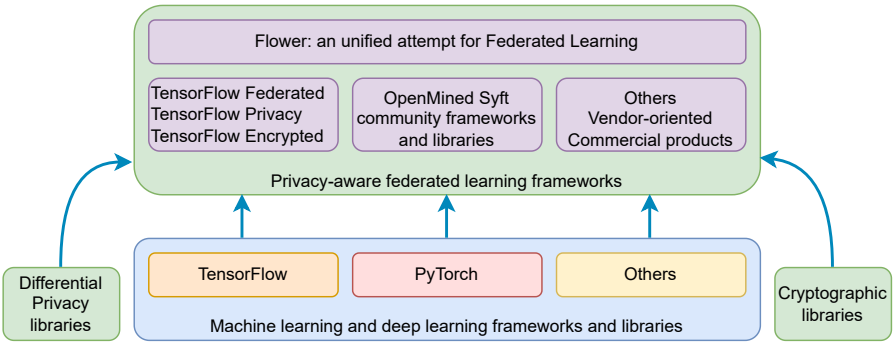

Fig. 6 Summary of the landscape of Privacy-Aware Federated Learning frameworks.

3.2.1 TensorFlow Community Frameworks and Tools

As already presented, *TensorFlow* (Tensorflow, 2023b) is an open-source end-to-end platform for ML and DL. Today, TensorFlow is one of the most used DL frameworks, together with PyTorch. In this section, we present different TensorFlow frameworks and tools related to privacy, security, and FL implementation (Section 2.2.1).

TensorFlow Federated

or TF Federated or TFF (TF-Federated, 2023) is a framework for implementing FL using the client-server architecture. It is an open-source framework for ML and other computations on decentralized data. TF Federated enables developers to *simulate* the FL algorithms included in their model and data, as well as to experiment with novel algorithms. The building blocks provided by TF Federated can also be used to implement non-learning computations, such as aggregated analytics over decentralized data. TF Federated provides the Federated Core language for federated computation, as well as a set of higher-level Federated Learning API as follows.

- Federated Core API is a set of low-level interfaces for expressing federated algorithms by combining TensorFlow with distributed communication operator within strongly typed functional programming environments.
- Federated Learning API, which is a set of high-level interfaces to apply the included FL and evaluation implementations to existing TensorFlow models.

Insights:

- TF Federated claims to be architecture-agnostic, which means that it has the ability to compile all code into an abstract representation, and as a result, it can be deployed in a diverse environment.
- TF Federated includes numerous user tutorials to better understand the architecture and applicability to support the easy moving from centralized to distributed learning.

– TF Federated includes tutorials on the use of DP in federated learning schemes by changing the basic federated average algorithm (TF-Federated, 2024).
– TF Federated provides users with a collection of data.
– TF Federated is deeply integrated with TensorFlow ecosystem, supports Keras models, and provides good visualization with TensorBoard.

TensorFlow Privacy

or TF Privacy or TFP (TF-Privacy, 2023) is a Python library that allows one to train DL models with different privacy assumptions. For this, one of the available implementations consists, during the training of a neural network, in using stochastic gradient descent with differential privacy (DP-SGD), which is a modification of the standard ML stochastic gradient descent (SGD) algorithm. To do so, for computing the gradient with DP, users need to include the gradient norm bound (for clipping) and the noise scale (Abadi et al, 2016).

Insights:

– TF Privacy provide users with common TensorFlow optimizers implemented with differential privacy
– TF Privacy include different tutorials for better understating of the library, including the use of DP-SGD with widely known datasets.
– For certain Keras models, TF Privacy includes differentially private implementations.

TensorFlow Encrypted

or TF Encrypted or TFE (TF-Encrypted, 2023) is a framework for encrypted ML in TensorFlow. While TensorFlow itself offers an optimized engine to execute local and distributed computations, as well as a high-level interface to express these computations (Dahl et al, 2018), TF Encrypted provides the features presented below.

Insights:

– TF Encrypted provides a basic multi-party computation (MPC) type and passive security under single corruption, which relies on two cryptographic primitives, namely additive secret sharing and a secure channel between all participants.
– TF Encrypted allows the design and implementation of private ML within distributed systems.
– TF Encrypted can be seen as a bridge between TensorFlow and the Microsoft SEAL library.

In short, the TensorFlow community currently has three PPML core pillars:

- TF Federated for federated computation;
- TF Privacy for differential private learning;
- TF Encrypted for encrypted computation.

These libraries could easily benefit from each other, for example, including differential privacy (TF Privacy) during the training of a model in a federated architecture (TF Federated). However, they are developed separately, so there are still no obvious integration points, as expected (TF-Federated-doc, 2024). TF Federated supports differentially private aggregation as well, using adaptive clipping and Gaussian noise.

3.2.2 The OpenMined Syft Community and PyTorch

Syft is OpenMined's open-source stack that provides secure and private Data Science in Python (OpenMined, 2023b). Syft decouples private data from model training using techniques such as FL, differential privacy, and encrypted computation. The OpenMined Syft community of frameworks is the most popular approach compared to the previous TensorFlow community frameworks. The Syft framework community represents a vision of an ecosystem for creating and developing private AI-powered solutions in a distributed and decentralized manner among data owners and data scientists (Section 2.2.4).

PySyft

PySyft (PySyft, 2023) is the flagship of the OpenMined Syft community. PySyft augments DL frameworks for servers and IoT with privacy-preserving capabilities. PySyft has several other co-frameworks for different platforms, such as KotlinSyft (KotlinSyft, 2023) (Kotlin library for Android), SwiftSyft (SwiftSyft, 2023) (Swift library for iOS) or Syft.js (Syft.js, 2023) for web and Node, built in Javascript.

Insights:

- PySyft decouples private data from model training, using FL, differential privacy, and encrypted computation (SMPC and HE) within DL frameworks like PyTorch and TensorFlow (soon).
- PySyft is oriented to be a remote data science platform beyond FL. It uses differential privacy and SMPC for strong privacy guarantees.
- PySyft allows one to perform DL models in a secure and private way, including FL workflows. It can be integrated with DP methods (Ziller et al, 2021).

PySyft has two primary purposes in performing two types of computation:

- *Dynamic computation:* directly computing data that cannot be seen;
- *Static computation:* create static graphs of computation which can be deployed on different machines.

Overall, PySyft is a central part of the Syft ecosystem, which allows one to compute information remotely on machines that you do not have full control over. Syft ecosystem helps to keep the data in original ownership while allowing them to be used privately for computations.

SyMPC

SyMPC (SyMPC, 2023) is a library that extends PySyft with SMPC support.

Insights:

- SyMPC allows computing over encrypted data and to train and evaluate neural networks.
- SyMPC supports different protocols: ABY3, Falcon, FSS and SPDZ.
- SyMPC provides support for Conv2d and linear layers by means of the protocols Falcon and SPDZ for neural networks.

These advantages of SyMPC make it the fastest way to establish ML experiments based on the SMPC approach. However, SyMPC is still a prototype and is not supposed to be used in production environments.

PyDP

PyDP (PyDP, 2023) is an OpenMined wrapper for the Google Differential Privacy project.

Insights:

- PyDP includes several differential privacy algorithms, such as BoundedSum, BoundedMean, Max, Count Above, Percentile, Min, and Median for computing the most common statistics with privacy guarantees by using Laplace noise.
- PyDP includes a curated list of tutorials.

TenSeal

TenSeal (TenSEAL, 2023) is a library for homomorphic encryption operations in tensors, built on top of Microsoft SEAL (SEAL, 2023) to enhance the community software product by cryptographic capacity. TenSeal exploits this property of Microsoft SEAL by providing an easy-to-use Python API, while preserving efficiency by implementing most of the operations using C++.

Insights:

- TenSeal is built on top of Microsoft SEAL for Syft community.
- TenSeal uses Brakerski-Fan-Vercauteren scheme (BFV) for encrypting and decrypting vectors of integers and Cheon-Kim-Kim-Song (CKKS) for real numbers.
- TenSeal is easy to use and is provided with a set of tutorials.

The features of SyMPC and PyDP are planned to progressively integrate into the main PySyft framework.

In this area, we can also see a product such as PySyft-TensorFlow (PySyftTF, 2023), which is a collaboration effort between the Syft community and the TensorFlow community. The framework brings secure, private DL to TensorFlow. However, it will soon be deprecated in favor of PySyft. This fusion-shattering example shows the dynamical evolving state in this applied research area.

3.2.3 Flower: an Unified Approach to Federated Learning

Flower (Flower, 2023) is a Python library that provides a unified approach to FL, analytics, and evaluation with its motto: *An unified approach to federated learning, analytics, and evaluation. Federate any workload, any ML framework, and any programming language.* Flower is an FL framework that offers a stable language and ML framework-agnostic implementation of an FL system. Flower provides an API wrapper for TensorFlow, TensorFlow Lite, PyTorch, PyTorch Lightning, Hugging Face, MXNet, JAX, and Scikit-Learn.

Insights:

- The framework allows for the rapid transition of existing ML training pipelines into an FL setup to evaluate their convergence properties and training time in a federated setting.
- Flower provides support for extending FL implementations to mobile and wireless clients, with heterogeneous compute, memory, and network resources (Beutel et al, 2022).
- Flower federates any workload, any ML framework, and any programming language.

Regarding the integration of PETs and additional privacy preserving measures, Flower supports differential privacy (DP) by means of the DP-FedAvg algorithm. This makes it possible to integrate DP into the model training processes according to the ML or DL frameworks with which Flower is compatible (Flower, 2024).

3.2.4 Other Federated Learning Frameworks and Tools

The list of FL frameworks is long (LF-AI-Data-Landscape, 2023), reflecting the high interest of the research community to address the concerns about security and privacy. Many of them have promising features for federated learning in a fast dynamic development stage. In Table 6 different state-of-the-art FL frameworks are presented, summarized together with the license that covers them and the number of stars they have on GitHub.

More descriptions of selected ones based on their popularity (GitHub stars), except for TensorFlow community, Syft community, and Flower, are as follows.

FATE

FATE (Fate, 2023) (Federated AI Technology Enabler) FATE is an open-source project initiated by Webank's AI Department to provide a secure computing framework to support the federated AI ecosystem. WeBank is the first privately-owned bank and the first digital-only bank in China, which joins the Linux Foundation at the Gold level since 2019 and FATE is now an open-source project hosted by the Linux Foundation.

Insights:

- FATE implements multiple secure computation protocols to enable big data collaboration in compliance with data protection regulations.

Table 6 Notable Federated Learning Frameworks. GitHub star as of August 14, 2024.

Federated Framework	GitHub Stars	Licence and Notes
OpenMined PySyft	9.4K	Apache 2.0
FATE	5.6K	Apache 2.0
Flower	4.7K	Apache 2.0, framework-agnostic
FEDML	4.1K	Apache 2.0
TensorFlow Federated	2.3K	Apache 2.0
SMILELab-FL FedLab	712	Apache 2.0
Intel OpenFL	698	Apache 2.0
NVIDIA Flare	585	Apache 2.0
IBM Federated Learning	493	Specific open-source license

– In addition to FL architectures, FATE supports secure computation for PPML.

FEDML

FEDML (FedML, 2023) stands for Foundational Ecosystem Design for ML. It has the motto: *The unified and scalable ML library for distributed large-scale training, model service, and federated learning.*

Insights:

– FEDML helps developers launch complex model training, deployment, and federated learning anywhere on decentralized GPUs, multiclouds, edge servers, and smartphones, easily, economically, and securely.
– FEDML is backed by FEDML Nexus AI (FedML-Nexus-AI, 2023), which is an enterprise pay-as-you-go all-in-one AI platform for cloud service for LLM and Generative AI.

FEDML itself is an open-source library, and FEDML Nexus AI provides various advanced plans or enterprise services. The company is also actively involved in the construction of an academic ecosystem through the sponsorship of AI-related academic conferences.

In short, on the presented FL frameworks, the evolution has started to move from classic and centralized ML/DL frameworks to FL ones. However, they are currently in a highly dynamic continuous development state, i.e., fast changing with many improvements and incompatibility issues. The list of existing FL frameworks and tools is longer than our list presented in Section 3.2. However, their number of GitHub stars is not very high, probably because of the novelty and their recent appearance, but also because of the continuous development of emerging technologies and frameworks in the field. We briefly present the rest of the ones listed in Table 6 below.

- *OpenFL* (OpenFL, 2023) is an Intel Python 3 framework for FL without sharing sensitive information. It is designed to be a flexible, extensible and easily learnable tool for data scientists. OpenFL is hosted by Intel, aims to be community-driven, and welcomes contributions back to the project. It supports aggregation algorithms such as FedAvg, FedProx, FedOpt, and FedCurv ((Shoham et al, 2019), (Li et al, 2020b), (Reddi et al, 2021)), with TensorFlow, PyTorch implementations, and with other DL frameworks.
- *FedLab* (FedLab, 2023) has motto: *A flexible FL framework based on PyTorch, which simplifies FL research.* It is from the SMILELab-FL laboratory, providing the necessary modules for FL simulation, including communication, compression, model optimization, data partition, and other functional modules avoiding traditional direct data-sharing behavior.
- *NVIDIA Flare* (NVFlare) is an extensible, open-source domain-independent SDK that enables users to adapt existing ML/DL workflows to a federated paradigm (FLARE, 2024a). It is a framework powered by NVIDIA Federated Learning Application Runtime Environment, which allows one to perform research concerning FL applications, but also real-world production deployments. This library implements commonly used privacy protection filters and allows the use of differential privacy to model weights before performing the aggregation on the server side (FLARE, 2024b).
- *IBM Federated Learning* (IBM-FL, 2023) provides tools for multiple remote parties to collaboratively train a single ML model without sharing data. Each party trains a local model with a private data set. Only the local model is sent to the aggregator to improve the quality of the global model that benefits all parties. IBM Federated Learning library is an open-source Python framework for FL in an enterprise environment.

For a more complete review, we can list *Sherpa.ai FL* (Rodríguez-Barroso et al, 2020) a framework for FL and DP includes both DNN and classical ML approaches; Microsoft *Flute* (Flute, 2023), which is a PyTorch-based orchestration environment that enables GPU or CPU-based FL simulations with the primary goal to enable researchers to rapidly prototype and validate their ideas; or Google *FedJAX* (FedJAX, 2023), which is an open-source library based on JAX for FL simulations that emphasizes ease-of-use in research.

3.3 Cryptographic Libraries

As presented in Section 2.4.3, cryptography is used to ensure data security and consequently data privacy. Related to Homomorphic Encryption (HE) and in addition to the TF Encrypted (TF-Encrypted, 2023) presented in Section 3.2.1, there are other cryptography libraries that implement techniques for HE or other cryptographic protocols. Other notable HE libraries in the FL context (from Table 7) are presented as follows.

- *SEAL* (SEAL, 2023): Microsoft SEAL is an easy-to-use open-source (MIT licensed) HE library developed by the Cryptography and Privacy Research Group at Microsoft. The library is written in C++ and is easy to compile and

run in many different environments. The TenSeal (TenSEAL, 2023) library from OpenMined which was exposed in Section 3.2.2 is built on SEAL.

- *HElib* (HElib, 2023) is an open-source software library that implements homomorphic encryption. It supports the Brakerski-Gentry Vaikuntanathan FHE scheme (BGV) with bootstrapping and the approximate number Cheon-Kim-Kim-Song homomorphic encryption scheme (CKKS). HElib also includes optimizations for efficient homomorphic evaluation, focusing on effective use of ciphertext packing techniques and on Gentry-Halevi-Smart optimizations.
- *SecretFlow* (SecretFlow, 2023) is a unified framework for privacy-preserving data intelligence and ML. To achieve this goal, it provides: (1) An abstract device layer consisting of plain devices and secret devices which encapsulate various cryptographic protocols; (2) A device flow layer modeling higher algorithms such as device object flow and DAG; (3) An algorithm layer to perform data analysis and ML with horizontal or vertical partitioned data; (4) A workflow layer that seamlessly integrates data processing, model training, and hyperparameter tuning.
- *Paillier* (Paillier, 2023) is a Python library that implements Paillier Partially Homomorphic Encryption. The homomorphic properties of the Paillier cryptosystem are: (1) Encrypted numbers can be multiplied by a non-encrypted scalar; (2) Encrypted numbers can be added together; (3) Encrypted numbers can be added to non-encrypted scalars.
- *OpenFHE* (OpenFHE, 2023) is an open-source C++ library for performing fully homomorphic encryption. It includes efficient implementations for the most common FHE schemes, such as the Brakerski-Gentry-Vaikuntanathan (BGV) and Brakerski-Fan-Vercauteren scheme (BFV) for integer arithmetic and Cheon-Kim-Kim-Song (CKKS) for real-number arithmetic, among others.

Cryptographic libraries are essential tools for implementing secure communication and data protection at various application levels. They provide pre-built, tested, and optimized functions for performing different cryptographic operations, simplifying the process for developers and ensuring the use of robust and secure algorithms.

3.4 Differential Privacy Libraries

As presented in Section 2.4.2, Differential Privacy (DP) is used in the FL context to effectively prevent information leakage (OpenMined, 2023a). In addition to *TF Privacy* (TF-Privacy, 2023) presented in Section 3.2.1 and *PyDP* (PyDP, 2023) presented in Section 3.2.2, we also highlight other notable DP libraries (from Table 7) as follows.

- *Google's Differential Privacy* library (GoogleDP, 2023) allows users to implement algorithms with ϵ-differential privacy an (ϵ, δ)-differential privacy. Both Laplace and Gaussian mechanisms are implemented. The most

Table 7 Homomorphic Encryption and Differential Privacy Libraries. GitHub star as of August 14, 2024.

Framework	GitHub Stars	HE	DP	Licence and Notes
Microsoft SEAL	3.5K	✓		MIT
HElib	3.1K	✓		Apache 2.0
SecretFlow	2.3K	✓		Apache 2.0
TF Encrypted	1.2K	✓		Apache 2.0
OpenMined TenSEAL	790	✓		Apache 2.0
Paillier	594	✓		GPLv3
OpenFHE	684	✓		BSD-2-Clause
Google's Differential Privacy	3K		✓	Apache 2.0
TF Privacy	1.9K		✓	Apache 2.0
Opacus	1.7K		✓	Apache 2.0
Diffprivlib	804		✓	MIT
OpenMined PyDP	495		✓	Apache 2.0
OpenDP	303		✓	MIT

common statistics are implemented by using DP, such as mean, standard deviation, quantiles, etc.

- *Opacus* (MetaPlatforms, 2023) is a library that allows PyTorch models to be trained with differential privacy. It supports training with minimal code changes required on the client, has little impact on training performance, and allows the client to online track the privacy budget expended at any given moment.
- *IBM Diffprivlib* (IBM-Diffprivlib, 2023) is a general purpose library for experimenting, investigating, and developing applications in differential privacy. It comprises four major components: Mechanisms (with little or no default settings, and are intended for experts to apply their own models); Models (includes ML models with DP such as clustering, classification, regression, dimensionality reduction and pre-processing); Tools (a number of tools for DP data analysis); and Accountant (used to track privacy budget and estimate the privacy loss).
- *OpenDP* (HarvardUniversity, 2023) library is a community effort to build open-source software tools for sensitive private data analysis. The tool is part of the larger OpenDP project of Harvard University that expresses privacy-aware computations. It is implemented in Rust, with bindings for easy use from Python and R. It is under continuous development, with work in progress.

DP libraries are specifically designed to help developers implement differential privacy within their applications. This is a privacy-preserving technique that is used to share data insights while protecting the privacy of individual participants. The combination or integration of DP libraries with FL frameworks and tools is the subject of concrete implementation at various production levels. We can mention the popular combination of Flower, Opacus, and PyTorch as an example.

3.5 Evolution Key Findings

The ML landscape is constantly evolving with the rapid evolution of frameworks and tools. Privacy-aware FL frameworks and tools are driving exciting advancements:

- FL frameworks like TensorFlow Federated, PySyft, Flower and FATE are expanding capabilities and popularity. While TensorFlow Federated is implemented to allow TensorFlow users to perform FL training, PySyft is oriented to remote data science and PyTorch. Flower is comfortable for quick experimentation and ease of use. FATE has strong security features, but less community support compared to other open-source frameworks.
- Some FL frameworks integrate the differential privacy approach, secure aggregation methods, and homomorphic encryption to protect sensitive data during training.
- FL frameworks also focus on efficiency and scalability, that is, optimizing communication overhead, managing heterogeneous devices, and enabling efficient model aggregation for large-scale deployments.

This is a highly dynamic and evolving environment with many frameworks and tools, indicating intensive research interests in the PPML aiming to address siloed and unstructured data, including privacy and worldwide regulations of data sharing (such as GDPR and CCPA) and incentive models for data-transparent ecosystems.

Theoretically, with an FL architecture we can perform all centralized ML tasks like supervised learning, unsupervised learning, or reinforcement learning, including neural network implementation. Practically, current implementations of FL libraries and frameworks are at various levels of quality and realization compared to the implementation of classical ML and DL ones. We can note that all FL frameworks and tools in the current state do not yet implement all the features mentioned in Section 2.5.

The highly evolving state of FL frameworks and tools leads to unstable development of intelligent software. The most common obstacle is inconsistency among library versions, which can break up or slow down the development phase of artificial intelligence models. The appreciated use of the containerization approach may help overcome this obstacle.

It is important to note that, in some cases, the PETs presented involve high computational power requirements and high memory consumption because of the underlying cryptographic and secure technologies. This is the case for

homomorphic encryption and trusted execution environments. In the special case of FHE the requirement on computational power and memory consumption is exceptionally high, which leads to, in many cases, impossible realization of secured distributed model training and also a lot of difficulties in inference. The reason is the huge overhead of FHE schemes and the lack of access to such high computational power, which enables the realization of secure AI research within AI communities of practitioners (Section 2.4.3).

The requirements for secure assurance for sensitive data protection in distributed computing environments are for all three data states: data at rest, data in motion, and data in use (Section 2.3). The concrete requirements are different from application to application. Some of them require the assurance of all states, others can require certain states based on their need.

Regarding the *limitation of the work* the depth of the analysis stands out, which aims to achieve the commitment of a clear presentation of a broad theoretical background of combined research fields, as well as a wide number of practical implementations of frameworks and tools. Our work presents a recent comprehensive analysis of the evolution landscape, which is rapidly evolving and changing with the fusion-shattering characteristic.

4 Conclusion and Future Work

Recent years have been marked by technological resilience and vibrancy, where the machine learning landscape is evolving rapidly, driven by advancements in technology, data availability, and the growing demand for AI-powered solutions. The key emerging trends that shape the future of machine learning development and deployment are federated learning, privacy-preserving, and machine learning lifecycle management.

This research presents a broad theoretical landscape concerning machine learning evolution, first in relation to Privacy Preserving Machine Learning (PPML) and secondly in the field of Privacy Enhancing Technologies (PETs). The evolution of ML and DL applied to centralized data has been carefully analyzed to move on to the application of these models with architectures acting on distributed data. In this sense, different state-of-the-art approaches have been presented, from well-known federated learning to split learning, gossip learning, and other decentralized architectures. In relation to PETs, different paradigms have been extensively studied and analyzed, including data masking, data perturbation, and differential privacy, homomorphic encryption, secure multi-party computation, and trusted and secure execution environments. Furthermore, the work in this paper provides a comprehensive evolution landscape of the most relevant technologies available in the field in the form of ML frameworks and tools with clear and concise outlines given in Section 2.6 and Section 3.5. Specifically, Section 3.5 carefully analyzes a battery of openly available frameworks and tools, first, in relation to classic ML and DL, and then moves to privacy-aware machine learning, with a special focus on federated learning. In addition, to complete the landscape of tools that, when used

in combination, allow building privacy-preserving machine learning applications, different cryptographic tools with emphasis on covering the HE domain, as well as libraries for differential privacy, are described and analyzed.

This detailed and comprehensive analysis of the landscape allows users and researchers to have a clear reference to make decisions regarding the tools and techniques needed to build their ML/DL applications with special attention to privacy. Specifically, practical information is concisely detailed on the use, availability, and functionality of different frameworks that make up the state-of-the-art in the area. However, it should be noted that the field of privacy-preserving techniques is a booming sector, with numerous tools emerging and continually being updated. For example, in relation to HE techniques, one of the main disadvantages falls from the point of view of computational cost, which makes it a field of study with a promising development horizon.

Concerning future work, we are interested in studying the implementation of the different distributed architectures mentioned, especially gossip learning and split learning, as open-source frameworks and tools. Although the scope of the frameworks that implement FL is wide, this is still not the case with these other techniques. In relation to PETs, future work should begin by delving into the implications related to their incorporation in distributed architectures, both from the point of view of usability, interpretability, and computational cost. Finally, regarding HE techniques and their inclusion in distributed architectures, further research on MKHE or SS schemes should be performed. The integration of MKHE in an FL architecture will allow the server to perform aggregation with the encrypted models with each client using different private keys.

The impact of new computing resources and techniques combined with an increasing avalanche of large datasets is transforming many areas of our lives. This dynamic evolution has many different faces, components, and contexts, and will continue to be shaped by the needs of society and technological advancements as they occur. In addition, there is the importance of responsibly building AI systems that are trustworthy, reliable, and human-centric with respect to data protection and privacy, as well as the importance of auditing AI systems to ensure that they operate as intended.

Funding

This work is funded by the European Union through the Horizon Europe AI4EOSC project under grant number 101058593.

References

Abadi M, Chu A, Goodfellow I, et al (2016) Deep learning with differential privacy. In: Proceedings of the 2016 ACM SIGSAC conference on computer and communications security, pp 308–318, https://doi.org/10.1145/2976749.2978318

AbdulRahman S, Tout H, Ould-Slimane H, et al (2020) A survey on federated learning: The journey from centralized to distributed on-site learning and beyond. IEEE Internet of Things Journal 8(7):5476–5497. https://doi.org/10.1109/JIOT.2020.3030072

Acar A, Aksu H, Uluagac AS, et al (2018) A survey on homomorphic encryption schemes: Theory and implementation. ACM Computing Surveys (Csur) 51(4):1–35. https://doi.org/10.1145/3214303

Aditya K, Grzonkowski S, Lekhac NA (2018) Enabling trust in deep learning models: A digital forensics case study. In: 2018 17th IEEE International Conference On Trust, Security And Privacy In Computing And Communications/12th IEEE International Conference On Big Data Science And Engineering (TrustCom/BigDataSE), IEEE, pp 1250–1255, https://doi.org/10.1109/TrustCom/BigDataSE.2018.00172

Ahmadzai M, Nguyen G (2023) Data partitioning effects in federated learning. Proceedings http://ceur-ws org ISSN 1613:0073. URL https://ceur-ws.org/Vol-3588/p13.pdf

Ahmadzai M, Nguyen G (2024) Differential private federated learning in geographically distributed public administration processes. Future Internet 16(7):220. https://doi.org/10.3390/fi16070220

AI-Act (2024) European Artificial Intelligence Act comes into force. URL https://ec.europa.eu/commission/presscorner/detail/en/ip_24_4123, accessed on 01.08.2024, European Commission

Al Ogaili RRN, Alomari ES, Alkorani MBM, et al (2023) Malware cyberattacks detection using a novel feature selection method based on a modified whale optimization algorithm. Wireless Networks pp 1–17. https://doi.org/10.1007/s11276-023-03606-z

Alazab M, RM SP, Parimala M, et al (2021) Federated learning for cybersecurity: Concepts, challenges, and future directions. IEEE Transactions on Industrial Informatics 18(5):3501–3509. https://doi.org/10.1109/TII.2021.3119038

Ali AM, Mohammed MA (2024) A comprehensive review of artificial intelligence approaches in omics data processing: Evaluating progress and challenges. International Journal of Mathematics, Statistics, and Computer Science 2:114–167. https://doi.org/10.59543/ijmscs.v2i.8703

APPI (2019) Japan's data protection law, the Act on the Protection of Personal Information. URL https://www.ppc.go.jp/files/pdf/Act_on_the_Protection_of_Personal_Information.pdf, accessed on 04.12.2023, Personal Information Protection Commission, Japan

Arrieta AB, Díaz-Rodríguez N, Del Ser J, et al (2020) Explainable artificial intelligence (xai): Concepts, taxonomies, opportunities and challenges toward responsible ai. Information fusion 58:82–115. https://doi.org/10.1016/j.inffus.2019.12.012

ARX (2024) ARX data anonymization tool. URL https://arx.deidentifier.org/, accessed on 11.01.2024

Asad M, Shaukat S, Javanmardi E, et al (2023) A comprehensive survey on privacy-preserving techniques in federated recommendation systems. Applied Sciences 13(10):6201. https://doi.org/10.3390/app13106201

Bauer LA, Bindschaedler V (2020) Towards realistic membership inferences: The case of survey data. In: Annual Computer Security Applications Conference, pp 116–128, https://doi.org/10.1145/3427228.3427282

Bell JH, Bonawitz KA, Gascón A, et al (2020) Secure single-server aggregation with (poly) logarithmic overhead. In: Proceedings of the 2020 ACM SIGSAC Conference on Computer and Communications Security, pp 1253–1269, https://doi.org/10.1145/3372297.3417885

Beltrán ETM, Pérez MQ, Sánchez PMS, et al (2023) Decentralized federated learning: Fundamentals, state of the art, frameworks, trends, and challenges. IEEE Communications Surveys & Tutorials https://doi.org/10.1109/COMST.2023.3315746

Bertino E (2016) Data security and privacy: Concepts, approaches, and research directions. In: 2016 IEEE 40th Annual computer Software and Applications conference (cOMPSAc), IEEE, pp 400–407, https://doi.org/https://doi.org/10.1109/COMPSAC.2016.89

Beutel DJ, Topal T, Mathur A, et al (2022) Flower: A friendly federated learning research framework. 2007.14390

Blot M, Picard D, Cord M, et al (2016) Gossip training for deep learning. CoRR abs/1611.09726. URL http://arxiv.org/abs/1611.09726, 1611.09726

Bonawitz K, Ivanov V, Kreuter B, et al (2017) Practical secure aggregation for privacy-preserving machine learning. In: proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security, pp 1175–1191, https://doi.org/10.1145/3133956.3133982

Brown T, Mann B, Ryder N, et al (2020) Language models are few-shot learners. Advances in neural information processing systems 33:1877–1901. URL https://proceedings.neurips.cc/paper_files/paper/2020/file/1457c0d6bfcb4967418bfb8ac142f64a-Paper.pdf

Budrionis A, Miara M, Miara P, et al (2021) Benchmarking pysyft federated learning framework on mimic-iii dataset. IEEE Access 9:116869–116878. https://doi.org/10.1109/ACCESS.2021.3105929

Caffe2 (2023) Caffe2 is a lightweight, modular, and scalable deep learning framework. URL https://github.com/facebookarchive/caffe2, source code now lives in the PyTorch repository https://github.com/pytorch/pytorch/, Accessed on 12.12.2023

CCPA (2020) California Consumer Privacy Act. URL https://oag.ca.gov/privacy/ccpa, accessed on 04.12.2023, State of California Department of Justice

Chaddad A, Wu Y, Desrosiers C (2023) Federated learning for healthcare applications. IEEE Internet of Things Journal https://doi.org/10.1109/JIOT.2023.3325822

Chainer (2023) Chainer - A flexible framework of neural networks for deep learning. URL https://github.com/chainer/chainer, accessed on 12.12.2023

Chatterjee S, Hanawal MK (2022) Federated learning for intrusion detection in iot security: a hybrid ensemble approach. International Journal of Internet of Things and Cyber-Assurance 2(1):62–86. https://doi.org/10.1504/IJITCA.2022.124372

Chen H, Chillotti I, Song Y (2019) Multi-key homomorphic encryption from tfhe. In: Galbraith SD, Moriai S (eds) Advances in Cryptology – ASIACRYPT 2019. Springer International Publishing, Cham, pp 446–472

Chen Y, Luo F, Li T, et al (2020) A training-integrity privacy-preserving federated learning scheme with trusted execution environment. Information Sciences 522:69–79. https://doi.org/10.1016/j.ins.2020.02.037

Cheng Y, Liu Y, Chen T, et al (2020) Federated learning for privacy-preserving ai. Communications of the ACM 63(12):33–36. https://doi.org/10.1145/3387107

Chollet F (2023) Keras - Deep Learning for humans. URL https://github.com/keras-team/keras, accessed on 12.12.2023

Chowdhury D, Banerjee S, Sannigrahi M, et al (2023) Federated learning based covid-19 detection. Expert Systems 40(5):e13173

Clarke R (2019) Principles and business processes for responsible ai. Computer Law & Security Review 35(4):410–422. https://doi.org/10.1016/j.clsr.2019.04.007

CNTK (2023) CNTK - Microsoft Cognitive Toolkit (CNTK), an open source deep-learning toolkit. URL https://github.com/microsoft/CNTK., accessed on 12.12.2023

Cournapeau D (2023) scikit-learn: machine learning in Python. URL https://github.com/scikit-learn/scikit-learn, accessed on 12.12.2023

Dahl M, Mancuso J, Dupis Y, et al (2018) Private machine learning in tensorflow using secure computation. 1810.08130

Dasari S, Kaluri R (2024) 2p3fl: A novel approach for privacy preserving in financial sectors using flower federated learning. CMES-Computer Modeling in Engineering and Sciences 140(2):2035–2051. https://doi.org/10.32604/cmes.2024.049152

DL4J (2023) deeplearning4j (DL4J) Suite of tools for deploying and training deep learning models using the JVM. URL https://github.com/deeplearning4j/deeplearning4j, accessed on 12.12.2023

Dua S, Du X (2016) Data mining and machine learning in cybersecurity. CRC press

Dwork C, Roth A, et al (2014) The algorithmic foundations of differential privacy. Foundations and Trends® in Theoretical Computer Science 9(3–4):211–407. https://doi.org/10.1561/0400000042

El Ouadrhiri A, Abdelhadi A (2022) Differential privacy for deep and federated learning: A survey. IEEE access 10:22359–22380. https://doi.org/10.1109/ACCESS.2022.3151670

Evans D, Kolesnikov V, Rosulek M, et al (2018) A pragmatic introduction to secure multi-party computation. Foundations and Trends® in Privacy and Security 2(2-3):70–246. https://doi.org/10.1561/3300000019

Faridoon A, Kechadi MT (2020) Data privacy in its three forms–a systematic review. In: Future Data and Security Engineering. Big Data, Security and Privacy, Smart City and Industry 4.0 Applications: 7th International Conference, FDSE 2020, Springer, pp 425–433, https://doi.org/10.1007/978-981-33-4370-2_30

fast.ai (2023) The fastai deep learning library. URL https://github.com/fastai/fastai., accessed on 12.12.2023

Fate (2023) FATE (Federated AI Technology Enabler). URL https://github.com/FederatedAI/FATE, accessed on 12.12.2023

FedJAX (2023) FedJAX is a JAX-based open source library for Federated Learning simulations that emphasizes ease-of-use in research. URL https://github.com/google/fedjax, accessed on 12.12.2023

FedLab (2023) FedLab - A flexible Federated Learning Framework based on PyTorch, simplifying your Federated Learning research. URL https://github.com/SMILELab-FL/FedLab, accessed on 12.12.2023

FedML (2023) FedML, Federated Learning/Analytics and Edge AI Platform. URL https://fedml.ai/, accessed on 12.12.2023

FedML-Nexus-AI (2023) FEDML Nexus AI: Next-Gen Cloud Services for LLMs and Generative AI. URL https://nexus.fedml.ai/, accessed on 12.12.2023

Firouzi R, Rahmani R, Kanter T (2021) Federated learning for distributed reasoning on edge computing. Procedia Computer Science 184:419–427. https://doi.org/10.1016/j.procs.2021.03.053

FLARE N (2024a) NVIDIA Federated Learning Application Runtime Environment. URL https://github.com/NVIDIA/NVFlare, accessed on 04.01.2024

FLARE N (2024b) NVIDIA FLARE - Data Privacy Protection. URL https://nvflare.readthedocs.io/en/main/user_guide/security/data_privacy_protection.html, accessed on 02.02.2024

Flower (2023) Flower - A Friendly Federated Learning Framework. URL https://github.com/adap/flower, accessed on 12.12.2023

Flower (2024) Flower - Differential privacy wrapper classes. URL https://flower.dev/docs/framework/explanation-differential-privacy.html, accessed on 02.02.2024

Flute (2023) FLUTE, Federated Learning Utilities and Tools for Experimentation. URL https://github.com/microsoft/msrflute, accessed on 12.12.2023

Gadekallu TR, Pham QV, Huynh-The T, et al (2021) Federated learning for big data: A survey on opportunities, applications, and future directions. 2110.04160

GDPR (2018) General Data Protection Regulation. URL https://eur-lex.europa.eu/eli/reg/2016/679/oj, accessed on 04.12.2023, European Union

Gentry C (2009) A fully homomorphic encryption scheme. URL https://www.proquest.com/docview/305003863?pq-origsite=gscholar&fromopenview=true, proQuest Dissertations Publishing, 3382729. Stanford University. Accessed on 12.12.2023

Giaretta L, Girdzijauskas Š (2019) Gossip learning: Off the beaten path. In: 2019 IEEE International Conference on Big Data (Big Data), IEEE, pp 1117–1124, https://doi.org/10.1109/BigData47090.2019.9006216

Gong X, Sharma A, Karanam S, et al (2022) Preserving privacy in federated learning with ensemble cross-domain knowledge distillation. In: Proceedings of the AAAI Conference on Artificial Intelligence, pp 11891–11899, URL https://ojs.aaai.org/index.php/AAAI/article/download/21446/21195

GoogleDP (2023) Google's differential privacy libraries.. URL https://github.com/google/differential-privacy, accessed on 12.12.2023

GoogleJAX (2023) JAX - Composable transformations of Python+NumPy programs: differentiate, vectorize, JIT to GPU/TPU, and more. URL https://github.com/google/jax, accessed on 11.01.2024

Gupta O, Raskar R (2018) Distributed learning of deep neural network over multiple agents. Journal of Network and Computer Applications 116:1–8. https://doi.org/10.1016/j.jnca.2018.05.003

H2O (2023) H2O is an Open Source, Distributed, Fast and Scalable Machine Learning Platform. URL https://github.com/h2oai/h2o-3, accessed on 12.12.2023

Hari S (2023) Best Machine Learning Frameworks(ML) for Experts in 2023. URL https://hackr.io/blog/machine-learning-frameworks, accessed on 12.12.2023

HarvardUniversity (2023) Open Differential Privacy. URL https://github.com/opendp/opendp, accessed on 04.02.2024

Hegedűs I, Danner G, Jelasity M (2019) Gossip learning as a decentralized alternative to federated learning. In: Distributed Applications and Interoperable Systems: 19th IFIP WG 6.1 International Conference, DAIS 2019, Springer, pp 74–90, https://doi.org/10.1007/978-3-030-22496-7_5

Hegedűs I, Danner G, Jelasity M (2021) Decentralized learning works: An empirical comparison of gossip learning and federated learning. Journal of Parallel and Distributed Computing 148:109–124. https://doi.org/10.1016/j.jpdc.2020.10.006

HElib (2023) Open-source software library that implements homomorphic encryption (HE). URL https://github.com/homenc/HElib, accessed on 12.12.2023

Hergenrother L, Park S (2021) Fully Homomorphic Encryption on IBM Cloud Hyper Protect Virtual Servers. URL https://www.proquest.com/

docview/305003863?pq-origsite=gscholar&fromopenview=true, accessed on 12.12.2023

Heurix J, Zimmermann P, Neubauer T, et al (2015) A taxonomy for privacy enhancing technologies. Computers & Security 53:1–17. https://doi.org/10.1016/j.cose.2015.05.002

HIPS-Autograd (2023) Autograd - Efficiently computes derivatives of numpy code. URL https://github.com/hips/autograd, accessed on 12.01.2024

Hong C, Huang Z, Lu Wj, et al (2020) Privacy-preserving collaborative machine learning on genomic data using tensorflow. In: Proceedings of the ACM Turing Celebration Conference-China, pp 39–44, https://doi.org/10.1145/3393527.3393535

IBM (2015) The IBM Analytics Solutions Unified Method for Data Mining/Predictive Ana- lytics (ASUM-DM). URL ftp://ftp.software.ibm.com/software/data/sw-library/services/ASUM.pdf,, accessed on 12.12.2023

IBM-Diffprivlib (2023) Diffprivlib is a general-purpose library for experimenting with, investigating and developing applications in, differential privacy. URL https://github.com/IBM/differential-privacy-library, accessed on 12.12.2023

IBM-FL (2023) IBM Federated Learning. URL https://github.com/IBM/federated-learning-lib, accessed on 12.12.2023

Intel (2023) Intel Software Guard Extensions (Intel SGX). URL https://www.intel.com/content/www/us/en/architecture-and-technology/software-guard-extensions.html, accessed on 21.12.2023

Jiang Y, Gu H, Lu Y, et al (2020) 2d-hra: Two-dimensional hierarchical ring-based all-reduce algorithm in large-scale distributed machine learning. IEEE Access 8:183488–183494. https://doi.org/10.1109/ACCESS.2020.3028367

Kadhe S, Rajaraman N, Koyluoglu OO, et al (2020) Fastsecagg: Scalable secure aggregation for privacy-preserving federated learning. arXiv preprint arXiv:200911248 https://doi.org/10.48550/arXiv.2009.11248

Kairouz P, McMahan HB, Avent B, et al (2021) Advances and open problems in federated learning. Foundations and Trends® in Machine Learning 14(1–2):1–210. https://doi.org/10.1561/2200000083

Kaissis GA, Makowski MR, Rückert D, et al (2020) Secure, privacy-preserving and federated machine learning in medical imaging. Nature Machine Intelligence 2(6):305–311. https://doi.org/10.1038/s42256-020-0186-1

Keras3 (2023) Introducing Keras 3.0. URL https://keras.io/keras_3/, accessed on 12.12.2023

Khalifa S, Martin P, Young R (2019) Label-aware distributed ensemble learning: A simplified distributed classifier training model for big data. Big Data Research 15:1–11. https://doi.org/10.1016/j.bdr.2018.11.001

KotlinSyft (2023) OpenMined/KotlinSyft - The official Syft worker for secure on-device machine learning. URL https://github.com/OpenMined/KotlinSyft, accessed on 12.12.2023

Lakhan A, Mohammed MA, Kadry S, et al (2021) Federated learning enables intelligent reflecting surface in fog-cloud enabled cellular network. PeerJ Computer Science 7:e758. https://doi.org/10.7717/peerj-cs.758

Lakhan A, Mohammed MA, Abdulkareem KH, et al (2023) Autism spectrum disorder detection framework for children based on federated learning integrated cnn-lstm. Computers in Biology and Medicine 166:107539. https://doi.org/10.1016/j.compbiomed.2023.107539

Lakhan A, Hamouda H, Abdulkareem KH, et al (2024) Digital healthcare framework for patients with disabilities based on deep federated learning schemes. Computers in Biology and Medicine 169:107845. https://doi.org/10.1016/j.compbiomed.2023.107845

Lambert M, Schuster T, Kessel M, et al (2023) Robustness analysis of machine learning models using domain-specific test data perturbation. In: EPIA Conference on Artificial Intelligence, Springer, pp 158–170, https://doi.org/10.1007/978-3-031-49008-8_13

Lazzarini R, Tianfield H, Charissis V (2023) Federated learning for iot intrusion detection. Ai 4(3):509–530. https://doi.org/10.3390/ai4030028

LF-AI-Data-Landscape (2023) LF AI & Data Foundation Interactive Landscape. URL https://landscape.lfai.foundation/, accessed on 12.12.2023

Li L, Fan Y, Tse M, et al (2020a) A review of applications in federated learning. Computers & Industrial Engineering 149:106854. https://doi.org/10.1016/j.cie.2020.106854

Li N, Li T, Venkatasubramanian S (2006) t-closeness: Privacy beyond k-anonymity and l-diversity. In: 2007 IEEE 23rd international conference on data engineering, IEEE, pp 106–115, https://doi.org/10.1109/ICDE.2007.367856

Li Q, Wen Z, Wu Z, et al (2021) A survey on federated learning systems: Vision, hype and reality for data privacy and protection. IEEE Transactions

on Knowledge and Data Engineering 35(4):3347–3366. https://doi.org/10.1109/TKDE.2021.3124599

Li S, Liu H, Bian Z, et al (2023) Colossal-ai: A unified deep learning system for large-scale parallel training. In: Proceedings of the 52nd International Conference on Parallel Processing. Association for Computing Machinery, New York, NY, USA, ICPP '23, p 766–775, https://doi.org/10.1145/3605573.3605613

Li T, Sahu AK, Zaheer M, et al (2020b) Federated optimization in heterogeneous networks. Proceedings of Machine learning and systems 2:429–450. URL https://proceedings.mlsys.org/paper_files/paper/2020/file/1f5fe83998a09396ebe6477d9475ba0c-Paper.pdf

Lindell Y (2020) Secure multiparty computation. Communications of the ACM 64(1):86–96. https://doi.org/10.1145/3387108

López-Alt A, Tromer E, Vaikuntanathan V (2012) On-the-fly multiparty computation on the cloud via multikey fully homomorphic encryption. In: Proceedings of the forty-fourth annual ACM symposium on Theory of computing, pp 1219–1234

Luo X, Wu Y, Xiao X, et al (2021) Feature inference attack on model predictions in vertical federated learning. In: 2021 IEEE 37th International Conference on Data Engineering (ICDE), IEEE, pp 181–192, https://doi.org/10.1109/ICDE51399.2021.00023

Lytvyn O, Nguyen G (2023a) Efficiency and security trade-offs of secure multi-party computation for machine learning. Procedia Computer Science 225:655–664. https://doi.org/10.1016/j.procs.2023.10.051

Lytvyn O, Nguyen G (2023b) Secure multi-party computation for magnetic resonance imaging classification. Procedia Computer Science 220:24–31. https://doi.org/10.1016/j.procs.2023.03.006

Machanavajjhala A, Kifer D, Gehrke J, et al (2007) l-diversity: Privacy beyond k-anonymity. ACM Transactions on Knowledge Discovery from Data (TKDD) 1(1):3–es. https://doi.org/10.1145/1217299.1217302

Mahato GK, Chakraborty SK (2023) A comparative review on homomorphic encryption for cloud security. IETE Journal of Research 69(8):5124–5133. https://doi.org/10.1080/03772063.2021.1965918

Majeed A, Lee S (2020) Anonymization techniques for privacy preserving data publishing: A comprehensive survey. IEEE access 9:8512–8545. https://doi.org/10.1109/ACCESS.2020.3045700

May R, Denecke K (2022) Security, privacy, and healthcare-related conversational agents: a scoping review. Informatics for Health and Social Care 47(2):194–210. https://doi.org/10.1080/17538157.2021.1983578

McMahan B, Moore E, Ramage D, et al (2017) Communication-efficient learning of deep networks from decentralized data. In: Artificial intelligence and statistics, PMLR, pp 1273–1282, URL http://proceedings.mlr.press/v54/mcmahan17a/mcmahan17a.pdf

MetaPlatforms (2023) Opacus a library that enables training PyTorch models with differential privacy. URL https://github.com/pytorch/opacus, accessed on 12.12.2023

Microsoft (2016) the Microsoft Team Data Science Process (TDSP). URL https://learn.microsoft.com/en-us/azure/architecture/data-science-process/overview, accessed on 12.12.2023

MindSpore (2023) MindSpore is a new open source deep learning training/inference framework that could be used for mobile, edge and cloud scenarios. URL https://github.com/mindspore-ai/mindspore, accessed on 12.12.2023

Mironov I, Talwar K, Zhang L (2019) Rényi differential privacy of the sampled gaussian mechanism. 1908.10530

MIT (2023) MIT Media Lab's Split Learning: Distributed and collaborative learning - Distributed deep learning and inference without sharing raw data. URL https://ai-infrastructure.org/ai-infrastructure-landscape/, accessed on 04.12.2023

Mo F, Haddadi H, Katevas K, et al (2021) Ppfl: privacy-preserving federated learning with trusted execution environments. In: Proceedings of the 19th annual international conference on mobile systems, applications, and services, pp 94–108, https://doi.org/10.1145/3458864.3466628

Mohammed MA, Lakhan A, Abdulkareem KH, et al (2023) Multi-objectives reinforcement federated learning blockchain enabled internet of things and fog-cloud infrastructure for transport data. Heliyon 9(11). https://doi.org/10.1016/j.heliyon.2023.e21639

Mothukuri V, Parizi RM, Pouriyeh S, et al (2021) A survey on security and privacy of federated learning. Future Generation Computer Systems 115:619–640. https://doi.org/10.1016/j.future.2020.10.007

Mouchet C, Troncoso-Pastoriza J, Bossuat JP, et al (2021) Multiparty homomorphic encryption from ring-learning-with-errors. Proceedings on Privacy Enhancing Technologies 2021(4):291–311

Munjal K, Bhatia R (2023) A systematic review of homomorphic encryption and its contributions in healthcare industry. Complex & Intelligent Systems 9(4):3759–3786. https://doi.org/10.1007/s40747-022-00756-z

MXNet (2023a) MXNet - Lightweight, Portable, Flexible Distributed/Mobile Deep Learning with Dynamic, Mutation-aware Dataflow Dep Scheduler; for Python, R, Julia, Scala, Go, Javascript and more. URL https://github.com/keras-team/keras, accessed on 12.01.2024

MXNet (2023b) MXNet - This project has retired. For details please refer to its Attic page. URL https://mxnet.apache.org/versions/1.9.1/, accessed on 12.01.2024

Narra KG, Lin Z, Wang Y, et al (2019) Privacy-preserving inference in machine learning services using trusted execution environments. 1912.03485

Nasr M, Shokri R, Houmansadr A (2019) Comprehensive privacy analysis of deep learning: Passive and active white-box inference attacks against centralized and federated learning. In: 2019 IEEE symposium on security and privacy (SP), IEEE, pp 739–753, https://doi.org/10.1109/SP.2019.00065

Nguyen G (2022) Introduction to Data Science. Spektrum STU Publishing, URL https://elvira.fiit.stuba.sk, the Edition of University Textbooks on Informatics and Information Technologies

Nguyen G, Dlugolinsky S, Bobák M, et al (2019) Machine learning and deep learning frameworks and libraries for large-scale data mining: a survey. Artificial Intelligence Review 52(1):77–124. https://doi.org/10.1007/s10462-018-09679-z

Nguyen G, Dlugolinsky S, Tran V, et al (2020) Deep learning for proactive network monitoring and security protection. ieee Access 8:19696–19716. https://doi.org/10.1109/ACCESS.2020.2968718

Nguyen G, Dlugolinsky S, Tran V, et al (2024) Network security aiops for online stream data monitoring. Neural Computing and Applications pp 1–25. https://doi.org/10.1007/s00521-024-09863-z

Novikova E, Doynikova E, Golubev S (2022) Federated learning for intrusion detection in the critical infrastructures: Vertically partitioned data use case. Algorithms 15(4):104. https://doi.org/10.3390/a15040104

OpenFHE (2023) Open-Source Fully Homomorphic Encryption Library. URL https://github.com/openfheorg/openfhe-development, accessed on 12.12.2023

OpenFL (2023) Open Federated Learning (OpenFL) - An Open-Source Framework For Federated Learning. URL https://github.com/intel/openfl, accessed on 12.12.2023

OpenMined (2023a) A survey of differential privacy framework. URL https://blog.openmined.org/a-survey-of-differential-privacy-frameworks/, accessed on 12.12.2023

OpenMined (2023b) OpenMined - A world where every good question is answered. URL https://www.openmined.org/, accessed on 12.12.2023

OpenMinedDP (2023) Open Mined - Use Cases of Differential Privacy. URL https://blog.openmined.org/use-cases-of-differential-privacy/, accessed on 12.12.2023

PaddlePaddle (2023) PArallel Distributed Deep LEarning: Machine Learning Framework from Industrial Practice. URL https://github.com/PaddlePaddle/Paddle, accessed on 12.12.2023

Paillier (2023) Paillier - A library for Partially Homomorphic Encryption in Python. URL https://github.com/data61/python-paillier, accessed on 12.12.2023

PapersWithCode (2023) Papers with code. Trends on the paper implementations grouped by framework. . URL https://paperswithcode.com/trends, accessed on 11.12.2023

Patarasuk P, Yuan X (2009) Bandwidth optimal all-reduce algorithms for clusters of workstations. Journal of Parallel and Distributed Computing 69(2):117–124. https://doi.org/https://doi.org/10.1016/j.jpdc.2008.09.002

Ponomareva N, Hazimeh H, Kurakin A, et al (2023) How to dp-fy ml: A practical guide to machine learning with differential privacy. Journal of Artificial Intelligence Research 77:1113–1201. https://doi.org/10.1613/jair.1.14649

PyDP (2023) PyDP The Python Differential Privacy Library. URL https://github.com/OpenMined/PyDP, accessed on 12.12.2023

PySyft (2023) OpenMined/PySyft - Data science on data without acquiring a copy. URL https://github.com/OpenMined/PySyft, accessed on 12.12.2023

PySyftTF (2023) PySyft - Tensorflow. URL https://github.com/OpenMined/PySyft-TensorFlow, accessed on 12.12.2023

PyTorch (2023) PyTorch - Tensors and Dynamic neural networks in Python with strong GPU acceleration. URL https://github.com/pytorch/pytorch, accessed on 12.12.2023

Rauniyar A, Hagos DH, Jha D, et al (2023) Federated learning for medical applications: A taxonomy, current trends, challenges, and future research directions. IEEE Internet of Things Journal https://doi.org/10.1109/JIOT.2023.3329061

Reddi S, Charles Z, Zaheer M, et al (2021) Adaptive federated optimization. 2003.00295

Riedel P, Schick L, von Schwerin R, et al (2024) Comparative analysis of open-source federated learning frameworks-a literature-based survey and review. International Journal of Machine Learning and Cybernetics pp 1–22. https://doi.org/10.1007/s13042-024-02234-z

Rodríguez-Barroso N, Stipcich G, Jiménez-López D, et al (2020) Federated learning and differential privacy: Software tools analysis, the sherpa. ai fl framework and methodological guidelines for preserving data privacy. Information Fusion 64:270–292. https://doi.org/10.1016/j.inffus.2020.07.009

Rodríguez N, Stipcich G, Jiménez D, et al (2020) Federated learning and differential privacy: Software tools analysis, the sherpa.ai fl framework and methodological guidelines for preserving data privacy. Information Fusion 64. https://doi.org/10.1016/j.inffus.2020.07.009

Sáinz-Pardo Díaz J, López García Á (2022) A python library to check the level of anonymity of a dataset. Scientific Data 9(1):785. https://doi.org/10.1038/s41597-022-01894-2

Sáinz-Pardo Díaz J, López García Á (2023a) Comparison of machine learning models applied on anonymized data with different techniques. In: 2023 IEEE International Conference on Cyber Security and Resilience (CSR), pp 618–623, https://doi.org/10.1109/CSR57506.2023.10224917

Sáinz-Pardo Díaz J, López García Á (2023b) Study of the performance and scalability of federated learning for medical imaging with intermittent clients. Neurocomputing 518:142–154. https://doi.org/10.1016/j.neucom.2022.11.011

Sardar MU, Fetzer C (2023) Confidential computing and related technologies: a critical review. Cybersecurity 6(1):1–7. https://doi.org/10.1186/s42400-023-00144-1

Scontain (2023) SCONE Confidential Computing - Protect your data, code & secrets. URL https://sconedocs.github.io/, accessed on 21.12.2023

SEAL (2023) Microsoft SEAL is an easy-to-use and powerful homomorphic encryption library. URL https://github.com/microsoft/SEAL, accessed on 12.12.2023

SecretFlow (2023) SecretFlow - A unified framework for privacy-preserving data analysis and machine learning. URL https://github.com/secretflow/secretflow, accessed on 12.12.2023

Shearer C (2000) The crisp-dm model: the new blueprint for data mining. Journal of data warehousing 5(4):13–22

Shi Y, Gao H, Avestimehr S, et al (2022) Experimenting fedml and nvflare for federated tumor segmentation challenge. In: International MICCAI Brainlesion Workshop, Springer, pp 228–240, https://doi.org/10.1007/978-3-031-44153-0_22

Shoham N, Avidor T, Keren A, et al (2019) Overcoming forgetting in federated learning on non-iid data. 1910.07796

Shokri R, Shmatikov V (2015) Privacy-preserving deep learning. In: Proceedings of the 22nd ACM SIGSAC conference on computer and communications security, pp 1310–1321, https://doi.org/10.1145/2810103.2813687

Silva PR, Vinagre J, Gama J (2023) Towards federated learning: An overview of methods and applications. Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery 13(2):e1486. https://doi.org/10.1002/widm.1486

Slijepčević D, Henzl M, Klausner LD, et al (2021) k-anonymity in practice: How generalisation and suppression affect machine learning classifiers. Computers & Security 111:102488. https://doi.org/10.1016/j.cose.2021.102488

So J, He C, Yang CS, et al (2022) Lightsecagg: a lightweight and versatile design for secure aggregation in federated learning. Proceedings of Machine Learning and Systems 4:694–720

Soykan EU, Karaçay L, Karakoç F, et al (2022) A survey and guideline on privacy enhancing technologies for collaborative machine learning. IEEE Access 10:97495–97519. https://doi.org/10.1109/ACCESS.2022.3204037

StanfordVisionLab (2020) ImageNet - image database organized according to the WordNet hierarchy. URL https://www.image-net.org/, accessed on 12.01.2024

Su W, Li L, Liu F, et al (2022) Ai on the edge: a comprehensive review. Artificial Intelligence Review 55(8):6125–6183. https://doi.org/10.1007/s10462-022-10141-4

Sweeney L (2002) k-anonymity: A model for protecting privacy. International journal of uncertainty, fuzziness and knowledge-based systems 10(05):557–570. https://doi.org/10.1142/S0218488502001648

SwiftSyft (2023) OpenMined/SwiftSyft - The official Syft worker for iOS, built in Swift. URL https://github.com/OpenMined/SwiftSyft, accessed on 12.12.2023

Syft.js (2023) OpenMined/Syft.js - The official Syft worker for Web and Node, built in Javascript. URL https://github.com/OpenMined/syft.js/, accessed on 12.12.2023

SyMPC (2023) A SMPC companion library for Syft. URL https://github.com/OpenMined/SyMPC., accessed on 12.12.2023

Taïk A, Cherkaoui S (2020) Electrical load forecasting using edge computing and federated learning. In: ICC 2020-2020 IEEE international conference on communications (ICC), IEEE, pp 1–6, https://doi.org/10.1109/ICC40277.2020.9148937

Taylor R, Kardas M, Cucurull G, et al (2022) Galactica: A large language model for science. 2211.09085

TenSEAL (2023) A library for doing homomorphic encryption operations on tensors. URL https://github.com/OpenMined/TenSEAL, accessed on 12.12.2023

Tensorflow (2023a) Tensorflow - An Open Source Machine Learning Framework for Everyone. URL https://github.com/tensorflow/tensorflow, accessed on 12.12.2023

Tensorflow (2023b) Tensorflow - An Open Source Machine Learning Framework for Everyone. URL https://www.tensorflow.org/, accessed on 27.11.2023

TensorFlow-XLA (2024) XLA (Accelerated Linear Algebra) open-source compiler for machine learning. URL https://www.tensorflow.org/xla, accessed on 12.01.2024

TF-Encrypted (2023) Encrypted Deep Learning in Tensorflow. URL https://tf-encrypted.io/, accessed on 27.11.2023

TF-Federated (2023) Tensorflow Federated: Machine Learning on Decentralized Data. URL https://www.tensorflow.org/federated, accessed on 27.11.2023

TF-Federated (2024) Differential Privacy in TensorFlow Federated. URL https://www.tensorflow.org/federated/tutorials/federated_learning_with_differential_privacy, accessed on 02.02.2024

TF-Federated-doc (2024) Federated Core. URL https://www.tensorflow.org/federated/federated_core, accessed on 04.02.2024

TF-Privacy (2023) Library for training machine learning models with privacy for training data. URL https://github.com/tensorflow/privacy, accessed on 27.11.2023

Thapa C, Arachchige PCM, Camtepe S, et al (2022) Splitfed: When federated learning meets split learning. In: Proceedings of the AAAI Conference on Artificial Intelligence, pp 8485–8493, URL https://ojs.aaai.org/index.php/AAAI/article/download/20825/20584

Truong N, Sun K, Wang S, et al (2021) Privacy preservation in federated learning: An insightful survey from the gdpr perspective. Computers & Security 110:102402. https://doi.org/10.1016/j.cose.2021.102402

Urmonov O, Sajid S, Aziz Z, et al (2024) Federated object detection scenarios for intelligent vehicles: Review, case studies, experiments and discussions. IEEE Transactions on Intelligent Vehicles https://doi.org/10.1109/TIV.2024.3408921

Vepakomma P, Gupta O, Swedish T, et al (2018) Split learning for health: Distributed deep learning without sharing raw patient data. 1812.00564

Verbraeken J, Wolting M, Katzy J, et al (2020) A survey on distributed machine learning. Acm computing surveys (csur) 53(2):1–33. https://doi.org/10.1145/3377454

Wei K, Li J, Ding M, et al (2020) Federated learning with differential privacy: Algorithms and performance analysis. IEEE Transactions on Information Forensics and Security 15:3454–3469. https://doi.org/10.1109/TIFS.2020.2988575

Wen J, Zhang Z, Lan Y, et al (2023) A survey on federated learning: challenges and applications. International Journal of Machine Learning and Cybernetics 14(2):513–535. https://doi.org/10.1007/s13042-022-01647-y

XGBoost (2023) XGBoost - Scalable, Portable and Distributed Gradient Boosting (GBDT, GBRT or GBM) Library, for Python, R, Java, Scala, C++ and more. Runs on single machine, Hadoop, Spark, Dask, Flink and DataFlow. URL https://github.com/apache/mxnet, accessed on 12.12.2023

Xu J, Glicksberg BS, Su C, et al (2021) Federated learning for healthcare informatics. Journal of Healthcare Informatics Research 5:1–19. https://doi.org/10.1007/s41666-020-00082-4

Yang Q, Liu Y, Chen T, et al (2019) Federated machine learning: Concept and applications. ACM Transactions on Intelligent Systems and Technology (TIST) 10(2):1–19. https://doi.org/10.1145/3298981

Yang X (2023) A historical review of collaborative learning and cooperative learning. TechTrends pp 1–11. https://doi.org/0.1007/s11528-022-00823-9

Yang Y, Shen W, Ruan B, et al (2021) Security challenges in the container cloud. In: 2021 Third IEEE International Conference on Trust, Privacy and Security in Intelligent Systems and Applications (TPS-ISA), IEEE, pp 137–145, https://doi.org/10.1109/TPSISA52974.2021.00016

Yi X, Paulet R, Bertino E (2014) Homomorphic encryption. In: Homomorphic Encryption and Applications. Springer, p 27–46, https://doi.org/10.1007/978-3-319-12229-8_2

Yu M, Tian Y, Ji B, et al (2022) Gadget: Online resource optimization for scheduling ring-all-reduce learning jobs. In: IEEE INFOCOM 2022-IEEE Conference on Computer Communications, IEEE, pp 1569–1578, https://doi.org/10.1109/INFOCOM48880.2022.9796785

Yuan L, Wang Z, Sun L, et al (2024) Decentralized federated learning: A survey and perspective. IEEE Internet of Things Journal https://doi.org/10.1109/JIOT.2024.3407584

Zhao C, Zhao S, Zhao M, et al (2019) Secure multi-party computation: theory, practice and applications. Information Sciences 476:357–372. https://doi.org/10.1016/j.ins.2018.10.024

Ziller A, Trask A, Lopardo A, et al (2021) PySyft: A Library for Easy Federated Learning, Springer International Publishing, Cham, pp 111–139. https://doi.org/10.1007/978-3-030-70604-3_5, URL https://doi.org/10.1007/978-3-030-70604-3_5