A Comprehensive Survey of Privacy-preserving Federated Learning: A Taxonomy, Review, and Future Directions

XUEFEI YIN, School of Engineering and Information Technology, University of New South Wales
YANMING ZHU, School of Computer Science and Engineering, University of New South Wales
JIANKUN HU, School of Engineering and Information Technology, University of New South Wales

The past four years have witnessed the rapid development of federated learning (FL). However, new privacy concerns have also emerged during the aggregation of the distributed intermediate results. The emerging privacy-preserving FL (PPFL) has been heralded as a solution to generic privacy-preserving machine learning. However, the challenge of protecting data privacy while maintaining the data utility through machine learning still remains. In this article, we present a comprehensive and systematic survey on the PPFL based on our proposed 5W-scenario-based taxonomy. We analyze the privacy leakage risks in the FL from five aspects, summarize existing methods, and identify future research directions.

CCS Concepts: • **General and reference** → **Surveys and overviews**; • **Computing methodologies** → **Machine learning**; • **Security and privacy** → **Privacy-preserving protocols;**

Additional Key Words and Phrases: Privacy-preserving federated learning, data privacy, horizontal federated learning, vertical federated learning, federated transfer learning, cryptographic encryption, perturbation techniques, anonymization techniques

ACM Reference format:
Xuefei Yin, Yanming Zhu, and Jiankun Hu. 2021. A Comprehensive Survey of Privacy-preserving Federated Learning: A Taxonomy, Review, and Future Directions. *ACM Comput. Surv.* 54, 6, Article 131 (July 2021), 36 pages.
https://doi.org/10.1145/3460427

1 INTRODUCTION

1.1 Background

The concept of **federated learning (FL)** was first introduced in 2016 [121]. Its core idea is to train machine learning models on separate datasets that are distributed across different devices or parties, which can preserve the local data privacy to a certain extent. Since then, FL has achieved a rapid development and become a hot research topic in the field of artificial intelligence [9, 131]. The

This research is supported by ARC Discovery Grant with project ID DP190103660, DP200103207 and ARC Linkage Grant with project ID LP180100663.

Authors' addresses: X. Yin and J. Hu (corresponding author), School of Engineering and Information Technology, University of New South Wales, Northcott Drive, Canberra, ACT, Australia, 2602; emails: xuefei.yin@unsw.edu.au, J.Hu@adfa.edu.au; Y. Zhu, School of Computer Science and Engineering, University of New South Wales, Anzac Parade, Sydney, NSW, Australia, 2052; email: yanming.zhu@unsw.edu.au.

0360-0300/2021/07-ART131 $15.00
https://doi.org/10.1145/3460427

development mainly benefits from the following three facts: (1) the wide successful applications of machine learning technologies, (2) the explosive growth of big data, and (3) the legal regulations for data privacy protection worldwide.

The widespread and successful applications of machine learning technologies are a primary driver of FL development. Over the past decades, machine learning technologies have achieved a remarkable success in a number of applications across various fields, such as language processing [36], image processing [229], and biometrics [208]. One of the most famous applications is AlphaGo [161]. In 2016, AlphaGo successfully beat a 9-dan professional player with a score of 4:1; in 2017, it continued to successfully beat the world's top-ranked Go player at the time with a score of 3:0; and now, its successor, the self-taught AlphaZero, is considered the best Go player in the world. In addition, many other applications have been widely commercialized, including face-recognition systems applied in various electronic products and access control systems. These successful machine learning applications have paved the way for the development of FL.

The explosive growth of big data has pushed development of FL. Every day, huge amounts of data are generated from social networks, the Internet of things, smart grids, e-commerce, hospitals, bank systems, and other areas [77]. This trend has promoted the development of machine learning, and on the other, it has also posed significant challenges to conventional machine learning. Because big data are usually stored in separate devices by various organizations, learning a global model while addressing its associated privacy concerns is becoming increasingly challenging. Therefore, conventional machine learning approaches are becoming less effective and FL has been heralded as an emerging solution.

The legal regulations for data privacy protection have boosted rapid development of FL. In recent years, many data breaches have significantly threatened the data privacy of users. For example, in 2019, more than 540 million records on Facebook users on Amazon's cloud service were exposed,[1] which caused serious social and legal issues. Therefore, several legal regulations have been established to protect private user data, such as the *General Data Protection Regulation* in European Union [179], the *Singapore Personal Data Protection Act* in Singapore [28], and the *California Privacy Rights Act*[2] in the United States. Such regulations have significantly promoted the development of FL, particularly the **privacy-preserving FL (PPFL)**.

1.2 Motivation

To illustrate our motivation, we provide a summary of related surveys in Table 1. Surveys [93, 170, 182, 217] reviewed conventional machine learning approaches and some privacy issues. Lyu et al. [117] reviewed some privacy issues regarding FL but did not present privacy-preserving mechanisms that could be used in the FL for privacy preservation. In addition, Lim et al. [105] provided a survey of the FL in mobile edge networks in terms of communication costs, resource allocation, privacy, and security, and Li et al. [97, 99] provided a survey of the FL covering expensive communication, systems heterogeneity, statistical heterogeneity, and privacy concerns. These three surveys [97, 99, 105] emphasized the efficiency and effectiveness of FL instead of privacy preservation. Li et al. [101] briefly reviewed privacy-preserving FL methods based on the homomorphic encryption, secure multiparty computing, and differential privacy. Kaissis et al. [84] investigated the use of FL and several privacy-preserving techniques in medical imaging. Yang et al. [204] provided a survey of FL in terms of data partitioning and architectures. None of these three surveys [84, 101, 204] analyzed the privacy issues related to FL. Niknam et al. [132] discussed

[1] https://www.upguard.com/breaches/facebook-user-data-leak.
[2] https://oag.ca.gov/privacy/ccpa.

Table 1. A Brief Summary of Related Surveys on Privacy-preserving Federated Learning

References	Federated Learning					Privacy-preserving mechanisms					Privacy leakage scenarios					Privacy-preserving scenarios & Risk assessment
	Data partitioning			Architecture		Protection techniques				Protection metrics						
	A1	A2	A3	B1	B2	C1	C2	C3	C4		D1	D2	D3	D4	D5	
Lyu et al. [117]	✓	✓	✓								✓		✓		✓	
Li et al. [101]				✓		✓	✓								✓	
Niknam et al. [132]				✓												
Kaissis et al. [84]						✓	✓	✓							✓	
Lim et al. [105]				✓		✓	✓									
Li et al. [99]				✓	✓	✓	✓							✓		
Yang et al. [204]	✓	✓	✓	✓	✓	✓	✓									
Tanuwidjaja et al. [170]						✓	✓									
Li et al. [97]	✓	✓	✓	✓	✓	✓	✓							✓		
Kairouz et al. [82]				✓	✓	✓	✓				✓					
Yang et al. [205]	✓	✓	✓	✓	✓	✓	✓								✓	
Zhang et al. [217]				✓		✓	✓				✓		✓			
Wang et al. [182]						✓	✓									
Li et al. [93]	✓				✓											
Our work	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓

A1- Horizontal A2- Vertical A3- Hybrid
B1- Client-Server B2- Peer-to-Peer
C1- Cryptographic Techniques C2- Perturbative Techniques C3- Anonymization Techniques C4- Hybrid Techniques
D1- Internal/External Attackers D2- Passive/Active Attacks D3- Training/Inference Phase
D4- Weight/Gradient/The Final Model D5- Inference Attacks

FL applications in 5G networks and challenges of FL for wireless communications. In addition, Yang et al. [205] provided an FL overview, covering distributed machine learning and its privacy concerns, a categorization of the FL architecture, the FL design, and FL applications. It placed more emphasis on the categorization and security analysis of FL systems. Kairouz et al. [82] provided a survey of FL in terms of efficiency, effectiveness, privacy, robustness, and fairness concerns. This survey only analyzed the privacy concerns from the aspects of external malicious actors and an adversarial server. In summary, the related surveys did not provide a comprehensive review on privacy-preserving aspects of FL.

Given the rapid development of FL (Section 1.1) and the aforementioned limits of existing related surveys, it is necessary to conduct a comprehensive survey on PPFL to review the latest findings, point out existing gaps, and indicate the future directions of PPFL research. Therefore, in this article, a systematic survey focusing on PPFL is presented to cover privacy leakage risks in the FL and PPFL methods. Specifically, we propose a 5W-scenario-based taxonomy to provide a systematic and multi-dimensional image PPFL. To serve as a good reference, we comprehensively analyze the potential privacy leakage risks in FL and divide them into the proposed 5W scenarios, summarize the current PPFL methods based on our well-organized four privacy-preserving schemes, and identify future research directions toward PPFL research. In addition, in this article, an explicit overview of FL and generic privacy-preserving mechanisms are presented to provide the necessary background knowledge, making it easy to understand the different categories of PPFL.

1.3 Main Contributions

As shown in Table 1, our article presents a comprehensive survey on PPFL based on our proposed 5W-scenario-based taxonomy. The proposed taxonomy provides a systematic and multi-dimensional image PPFL. Specifically, we first provide an explicit overview of FL and generic privacy-preserving mechanisms, including a clear categorization of FL based on data partitioning and communication architectures. We then analyze potential privacy leakage risks in FL from five fundamental aspects according to the proposed 5W scenarios, and summarize the current PPFL methods based on our well-organized four privacy-preserving schemes. Finally, we investigate various challenges of applying privacy-preserving mechanisms in the FL frameworks and present open research problems related to PPFL for future research. The main contributions of this study are as follows:

- We propose a 5W-scenario-based taxonomy to systematically analyze potential privacy leakage risks in FL, thereby providing a comprehensive and multi-dimensional image for PPFL. The proposed 5W-scenario-based taxonomy involves five fundamental aspects: “who” (internal and external attackers), “what” (active and passive attacks), “when” (training and inference phases), “where” (weight update, gradient update, and the final model), and “why” (four types of inference attacks).
- We summarize the state-of-the-art PPFL approaches according to our well-categorized four privacy-preserving mechanisms, which provides a significant guidance for future research investigation. The proposed PPFL categories include: encryption-based, perturbation-based, anonymization-based, and hybrid PPFL.
- We provide an explicit overview on FL and generic privacy-preserving mechanisms, paving the way for future research on PPFL and its applications. The categorization of FL is based on the data partitioning and communication architecture. The generic privacy-preserving mechanisms are introduced from two aspects: privacy-preserving techniques and privacy-preserving metrics.
- We discuss the challenges of PPFL, clarify existing gaps, identify open research problems, and indicate future research directions.

The rest of this article is organized as follows. First, we provide an overview of FL in Section 2 and introduce generic privacy-preserving mechanisms in Section 3. In Section 4, we then present our proposed 5W-scenario-based taxonomy on PPFL, and analyze potential privacy leakage risks, and discuss privacy-preserving schemes in FL. Finally, we provide some concluding remarks and offer open research problems for future research directions in Section 5.

2 AN OVERVIEW OF FEDERATED LEARNING

The key idea of FL is to train machine learning models on datasets that are distributed across different devices or organizations, while attempting to preserve data privacy [121]. The focuses of FL research can be roughly divided into three aspects: (1) improving the efficiency and effectiveness of FL [24, 68, 87, 100, 120, 211, 213], (2) improving the security of FL to attacks that aim to undermine the integrity of FL models and degrade the model performance [9, 13, 185, 198], and (3) improving the privacy preservation of FL on private user data and avoiding privacy leakages [116, 137, 144, 193, 199]. In this section, we present an overview of FL by first introducing the concept, and then providing a categorization of FL methods based on the data partitioning and communication architectures. Finally, we briefly compare FL with some related concepts.

2.1 Brief Introduction to FL

FL is a machine learning technique, that aims at collaboratively training a global machine learning model on multiple datasets distributed over separate clients/nodes without explicitly exchanging data samples between the clients [121, 150]. Its basic process is to train local models on local datasets and exchange parameters (e.g., model weights or gradients) between these local clients to achieve a global model.

Definition of Validity [97]. Suppose there are n separate clients, with each client represented by C_i, where $i \in [1, n]$ and C_i possesses a dataset $\mathcal{D}_i$. In a conventional machine learning setting, each client C_i possesses a model $\mathcal{M}_i$ trained only on its local dataset $\mathcal{D}_i$, and the accuracy obtained by $\mathcal{M}_i$ is denoted by f_i. In an FL setting, a global model $\hat{\mathcal{M}}$ is collaboratively trained by all n clients using their local datasets, and the accuracy obtained by $\hat{\mathcal{M}}$ is denoted as $\hat{f}$. The FL model $\hat{\mathcal{M}}$ is called valid if there exists $i \in [1, n]$ such that $\hat{f} > f_i$.

Definition of δ-Accuracy [204]. Assume that $\bar{\mathcal{M}}$ is a conventional model trained on a dataset $\mathcal{D} = \mathcal{D}_1 \cup \cdots \cup \mathcal{D}_n$, and the accuracy of which is represented by $\bar{f}$. The FL model $\hat{\mathcal{M}}$ then has δ-accuracy loss if $|\hat{f} - \bar{f}| < \delta$.

In an FL system, there are generally two main roles: (1) clients that hold their local datasets and (2) a server that orchestrates the entire training process and updates the global model without accessing the client datasets. The number of clients can be exceptionally large whereas there is usually only one server. In a special FL setting, the role of the server will be played by certain clients during the training phase. The FL training process generally consists of three key steps [105]:

- Step 1: *FL initialization on the server side.* First, the server initializes the weights of the global mode and the hyperparameters (e.g., the number of FL rounds, the total number of clients, and the number of clients to be selected during each training round). It then activates the clients, broadcasts the initialized global model $\mathbf{w}_g^0$, and distributes calculation tasks to certain selected clients.
- Step 2: *Local model training and update on the client side.* First, the selected clients receive the current global information (e.g., weights or gradients) sent by the server and update their individual local model parameters $\mathbf{w}_i^t$ using their local datasets, where t denotes the index of the current iteration round. Then, after finishing the local training, they send their local information (e.g., weights or gradients) to the server for model aggregation. During the local training phase, the goal of the selected client i in round t is to obtain the optimal local model parameters $\hat{\mathbf{w}}_i^t$ by minimizing the loss function $f_{loss}(\mathbf{w}_i^t)$, formulated by $\hat{\mathbf{w}}_i^t = \arg\min_{\mathbf{w}_i^t} f_{loss}(\mathbf{w}_i^t)$.
- Step 3: *Global model aggregation and update on the server side.* The server first aggregates the received local information sent by the selected clients, and then sends back the updated information to the clients for the next round of training. The goal is to obtain optimal global model parameters $\hat{\mathbf{w}}_g^t$ by minimizing the global loss function $f_{loss}(\mathbf{w}_g^t)$, formulated by $\hat{\mathbf{w}}_g^t = \arg\min_{\mathbf{w}_i^t} f_{loss}(\mathbf{w}_g^t)$, where $f_{loss}(\mathbf{w}_g^t) \overset{def}{=} \frac{1}{n} \sum_{i=1}^{n} f_{loss}(\mathbf{w}_i^t)$.

When the termination condition is met (e.g., the maximum number of rounds is reached or the accuracy of the global model is greater than the threshold), the server stops the training process, aggregates the updates, and distributes the global model to all clients. The FedAvg method proposed in Reference [121] is a concrete example of an FL framework, as shown in Algorithm 1. As described in the above steps, the server first initializes the model weights and distributes the calculation tasks to m selected clients. Client i then trains the local model by minimizing the local loss function, and returns the local state to the server. Finally, on the server side, the server orchestrates the entire training process and minimizes the global loss function.

In summary, for FL, the model parameters are exchanged, whereas the datasets are not shared among clients during the training phase; and after completing the training, the trained model will be shared among clients and used for inference. The key characteristics of the FL are outlined as follows:

ALGORITHM 1: FedAvg Algorithm [121]

```
Input:
    $R$: Maximum number of rounds, $m$: the number of clients selected in each round, $N_{epoch}$: the number of local epochs, and $\eta$: the local learning rate.
Output:
    Global model $\mathbf{w}_G$
Processing:
    [Server-side]
    Initialize $\mathbf{w}_G^0$
    for each round $t$ from 1 to $R$ do
        $\mathcal{S}_t$ contains $m$ clients randomly selected from the $n$ clients
        for each client $i \in \mathcal{S}_t$ in parallel do
            $\mathbf{w}_i^t, N_i \leftarrow \mathbf{LocalTraining}(i, \mathbf{w}_G^t)$
        end for
        $\mathbf{w}_G^{t+1} = \frac{1}{\sum_{j=1}^{m} N_j} \sum_{i=1}^{m} N_i \mathbf{w}_i^t$
    end for
    [Client-side]
    $\mathbf{LocalTraining}(i, \mathbf{w})$:
    Divide local dataset $\mathcal{D}_i$ into batches; $\mathcal{B}_i$ denotes the set of the batches.
    for each epoch $j$ from 1 to $N_{epoch}$ do
        for each batch $b \in \mathcal{B}_i$ do
            $\mathbf{w} \leftarrow \mathbf{w} - \eta \nabla L(\mathbf{w}; b)$
        end for
    end for
    return the weights $\mathbf{w}$ and $N_i = |\mathcal{D}_i|$
```

- FL systems have two main roles: clients as data holders, and the server/aggregator as the holder of the global model.
- FL trains a global model by sharing the model parameters between the clients and the server.
- FL provides a potential solution for privacy-preserving training owing to its characteristic of not sharing datasets among clients.
- The accuracy of an FL model trained collaboratively on separate datasets should be close to that of a conventional model trained on a dataset containing all of these separate datasets.

2.2 Categorization of Current FL Methods

This section summarizes the current FL methods according to a categorization based on data partitioning and communication architectures.

Let matrix $\mathcal{D}_i$ denote the dataset stored in client i and vector $\mathbf{d} \in \mathcal{D}_i$ denote a data sample of $\mathcal{D}_i$, which is represented by $\mathbf{d} = (\mathbf{d}_{ID}, \mathbf{d}_{feature}, \mathbf{d}_{label})$, where $\mathbf{d}_{ID} \in \mathcal{X}_{ID}^i$, $\mathbf{d}_{feature} \in \mathcal{X}_{feature}^i$, and $\mathbf{d}_{label} \in \mathcal{X}_{label}^i$ denote the sample ID, feature, and label, respectively, and $\mathcal{X}_{ID}^i$, $\mathcal{X}_{feature}^i$, and $\mathcal{X}_{label}^i$ denote the ID space, feature space, and label space, respectively. According to the distributions of the data samples in the ID and feature spaces, there are three data partitioning situations [119]:

S_1) Horizontal Data Partitioning. The datasets held by clients possess an identical feature space but different ID spaces, that is, $\mathcal{X}_{feature}^i = \mathcal{X}_{feature}^j$ but $\mathcal{X}_{ID}^i \neq \mathcal{X}_{ID}^j$. This situation usually occurs in the same field. For example, student records from different universities tend to have the same feature space but different ID space.

S_2) Vertical Data Partitioning. The datasets possess an identical ID space but different feature spaces, that is, $\mathcal{X}_{ID}^{i} = \mathcal{X}_{ID}^{j}$ but $\mathcal{X}_{feature}^{i} \neq \mathcal{X}_{feature}^{j}$. This situation usually occurs in different fields. For example, a citizen may have different data records in bank systems, hospitals, e-commerce markets, and so on.

S_3) Hybrid Data Partitioning. The datasets possess different ID spaces and different feature spaces, that is, $\mathcal{X}_{ID}^{i} \neq \mathcal{X}_{ID}^{j}$ and $\mathcal{X}_{feature}^{i} \neq \mathcal{X}_{feature}^{j}$.

Therefore, based on the data partitioning, Yang et al. [204] introduced FL methods according to the following three classes: (1) horizontal FL, (2) vertical FL, and (3) federated transfer learning.

2.2.1 Horizontal FL. Horizontal FL is suitable for training machine learning models using datasets having distributions as in situation S_1, that is, the datasets possess an identical feature space but different ID spaces, which can be formulated as follows:

$$Horizontal\ FL := \mathcal{X}_{feature}^{i} = \mathcal{X}_{feature}^{j}, \mathcal{X}_{ID}^{i} \neq \mathcal{X}_{ID}^{j}, \forall \mathcal{D}_i, \mathcal{D}_j, i \neq j.$$

McMahan et al. [121] proposed a horizontal FL framework based on mobile phone clients. In this framework, a client locally updates model weights and sends the local weights to a server for model aggregation, and collaboratively trains a global model together with other clients. Li et al. [96] proposed a horizontal FL framework for gradient boosting decision trees to improve the model efficiency and accuracy. Phong et al. [137] applied the additively homomorphic encryption to a horizontal FL framework to protect the gradients. Smith et al. [163] proposed a framework for multi-task learning that allows multiple clients to train different tasks, whose advantage is considering issues of high communication costs and stragglers during the training phase.

In horizontal FL, there are mainly two communication architectures: client-server and peer-to-peer architectures. The client-server architecture uses centralized computing, because there is a central server used for orchestrating the entire training process. The peer-to-peer architecture uses decentralized computing, because there is no central server and a client will be randomly chosen as the server in each training round.

Client-Server Architecture [205]. This architecture is also known as centralized FL. In a typical client-server architecture of a horizontal FL system, n clients collaboratively train a machine learning model with the help of the server, and its underlying assumption is that the clients are honest and the server is honest-but-curious. Therefore, the prevention of information leakage in this FL architecture focuses on the model parameter exchange between clients and the server, and the training process consists of five main steps:

- Step 1: The server initializes the model parameters and hyper-parameters, and distributes the calculation tasks to selected clients.
- Step 2: The selected clients train their local models and use privacy-preserving techniques to process the trained model parameters, and then send these parameters to the server.
- Step 3: The server performs secure aggregation by adopting a weighted average, for example.
- Step 4: The server sends the aggregated parameters back to the clients.
- Step 5: The clients decrypt the received parameters and update their local models.

During this training process, the exchanged model parameter has two types: a model weight or gradient. For the model weight, the clients send the locally calculated weights to the server, and the server aggregates the received weights and sends them back to the clients [121, 138, 209]. As the advantages of this type of method, they do not require frequent synchronization and have a tolerance to the update loss. A disadvantage is no guarantee of convergence. For the model gradient, the clients send the locally calculated gradients to the server and the server aggregates the received gradients and sends them back to the clients [63, 91, 210]. The advantages of this type of methods are their accurate gradient information and guaranteed convergence. The disadvantages are a communication cost and the need for connection needs a reliable communication.

Peer-to-Peer Architecture [205]. This architecture is also known as a decentralized FL. Compared to the client-server architecture, there is no central server [151, 214]. In this architecture, each client locally trains a machine learning model using its local dataset and updates its model using the model information received from other clients; the client then sends the updated model information to other clients. Therefore, the prevention of information leakage in this FL architecture focuses on the secure communication between clients, which can be achieved by adopting security techniques such as an encryption scheme based on public keys. In addition, because there is no central server, a protocol should be provided in advance to orchestrate the training process. There are two protocols: a cyclic transfer and a random transfer.

- Cyclic transfer. In this protocol, the clients are organized into a circular chain $\{C_1, C_2, \ldots, C_n\}$. Client C_1 sends its current model updates to client C_2. Client C_2 receives model information from C_1 and updates the received model information using its local dataset, and then sends the updated model information to its next client C_3. When the termination condition is met, the training process stops.
- Random transfer. In this protocol, a client C_k randomly selects a client C_i with equal probability and sends its model information to C_i, which receives model information from C_k and updates the received model information using its local dataset, and then randomly selects a client C_j with equal probability and sends the updated model information to C_j. This process is carried out simultaneously among n clients until the termination condition is met.

2.2.2 Vertical FL. Vertical FL is suitable for training machine learning models using datasets having distributions as in situation S_2, that is, the datasets possess an identical ID space but different feature spaces, which can be formulated by the following:

$$Vertical\ FL := \mathcal{X}^i_{ID} = \mathcal{X}^j_{ID}, \mathcal{X}^i_{feature} \neq \mathcal{X}^j_{feature}, \forall \mathcal{D}_i, \mathcal{D}_j, i \neq j.$$

A vertical FL scheme was proposed to train a privacy-preserving logistic regression model in Reference [71]. This scheme is used to study the effect of the entity resolution on the learning performance, and a Taylor approximation is applied to the loss and gradient functions such that a homomorphic encryption can be adopted for privacy-preserving computations. Yang et al. [203] proposed a quasi-Newton method-based vertical FL framework for logistic regression, whose advantage is that it reduces the communication costs. However, these two methods only focus on binary classification tasks with two clients in the vertical FL setting. To extend the vertical FL, Feng et al. [51] proposed a multi-participant multi-class vertical FL framework. Yang et al. [206] proposed a vertical FL framework for a logistic regress model without a third-party coordinator. Cheng et al. [27] proposed a lossless vertical FL method, which could enable clients to train gradient boosting decision trees in a collaborative manner. Liu et al. [112] proposed a vertical FL framework based on a block coordinate gradient descent algorithm, in which each client locally conducts more than one gradient updates before sending the local model information to the other clients. With this method, the impact of the number of local rounds for local updates is analyzed, and a global convergence with a proper choice of the number of local rounds is shown. As the advantage of this method, it reduces the communication overhead. In addition, Wang et al. [184] proposed measurements based on group instance deletion and group Shapley values to calculate the contribution of each client for vertical FL.

For vertical FL, there are mainly two communication architectures: an architecture with a third-party coordinator and an architecture without a third-party coordinator.

Architecture with Third-party Coordinator [204]. Suppose that clients C_1 and C_2 collaboratively train a machine learning model using their local datasets, and client C_1 has label data used for training the global model. Clients C_1 and C_2 are honest-but-curious to each other. To guarantee the data privacy during the training process, an honest third-party coordinator C_3 is involved. This is a reasonable assumption, because C_3 can be authorities such as governments. Therefore, such a vertical FL system consists of five main steps:

- Step 1: ID Alignment. Because there are different IDs in the two datasets of C_1 and C_2, the vertical FL system first needs to use encryption-based ID alignment techniques such as those in References [104, 154] to confirm the common IDs without exposing the private data of C_1 and C_2. These common data instances are then used to train the vertical FL model.
- Step 2: C_3 generates an encryption key pair and sends the public key to C_1 and C_2.
- Step 3: C_1 and C_2 encrypt their intermediate results and exchange this information.
- Step 4: C_1 and C_2 each calculate encrypted gradients and add a mask. C_1 also calculates an encrypted loss. Then, C_1 and C_2 send the encrypted results to C_3.
- Step 5: C_3 decrypts the received results and sends the decrypted gradients and loss back to C_1 and C_2. Then, C_1 and C_2 unmask the gradients and update their model parameters.

Architecture without Third-party Coordinator [206]. Suppose that clients C_1 and C_2 collaboratively train a machine learning model using their local datasets, and C_1 has label data used for training the global model. C_1 and C_2 are honest-but-curious to each other. To prevent privacy leakage, such a vertical FL system needs to consist of following seven main steps:

- Step 1: ID alignment. An ID alignment technique such as in Reference [104] is first used to confirm the common IDs between C_1 and C_2. Then, their common data instances are used to train a vertical FL model.
- Step 2: C_1 generates an encryption key pair and sends the public key to C_2.
- Step 3: C_1 and C_2 initialize their model weights.
- Step 4: C_1 and C_2 each calculate their partial linear predictors, and C_2 sends its predictor result to C_1.
- Step 5: C_1 calculates the model residual, encrypts the residual, and sends it to C_2.
- Step 6: C_2 calculates the encrypted gradient and sends the masked gradient to C_1.
- Step 7: C_1 decrypts the masked gradient and sends it back to C_2. Then, C_1 and C_2 update their model locally.

2.2.3 *Federated Transfer Learning.*

Federated Transfer Learning (FTL) is suitable for training machine learning models using datasets having distributions as in situation S_3, that is, the datasets possess different ID spaces and feature spaces or only have a few co-occurrence instances, which can be formulated by

$$FTL := \mathcal{X}_{ID}^{i} \neq \mathcal{X}_{ID}^{j}, \mathcal{X}_{feature}^{i} \neq \mathcal{X}_{feature}^{j}, \forall \mathcal{D}_i, \mathcal{D}_j, i \neq j.$$

Yang et al. [202] proposed a secure FTL framework, FedSteg, to train a personalized and distributed model for a secure image steganalysis. As its advantage, FedSteg is a general framework suitable for general network structures in privacy-preserving machine learning. Gao et al. [59] proposed a heterogeneous FTL framework, which provides an end-to-end learning protocol for heterogeneous feature space training among multiple clients. Its experiments show that this framework outperforms local training schemes and homogeneous FL schemes. Liu et al. [111] proposed a general privacy-preserving FTL framework to extend the scope of the existing secure FL to a wider range of practical applications. Compared with some secure deep learning methods that suffer from accuracy loss, the FTL can achieve the same accuracy as non-privacy-preserving methods and a higher accuracy than non-federated self-learning methods. Sharma et al. [157] proposed a secure and efficient FTL framework based on a multi-party computation. This allows clients to train a transfer learning model while keeping their datasets private against adversaries. In addition, compared with the homomorphic encryption approaches, it reduces the running time and communication cost.

Based on the categorization of transfer learning [136], FTL methods can be divided into three categories [205]: instance-based FTL, feature-based FTL, and parameter-based FTL.

- *Instance-based FTL* assumes that some labeled instances in the dataset in the source domain can be reweighted and reused for training in the target domain [11, 136]. For the horizontal FL, the datasets of different clients may have different distributions, which may result in an accuracy degradation of machine learning models trained on these datasets. One solution is to relieve the distribution difference by reweighting some of the selected data instances and then reuse them to train the model [35]. For the vertical FL, the objectives of different clients may be different and the ID alignment in the vertical FL may result in a negative impact on the FTL, which is known as negative transfer [136]. One solution is to use importance sampling to relieve a negative transfer [143].
- *Feature-based FTL* aims to minimize domain divergence and learn a "good" feature representation for the target domain, and thus it can effectively encode the transformation knowledge from the source domain to the target domain [90, 183]. For the horizontal FL, the feature representation can be obtained by minimizing the maximum mean discrepancy among different datasets of clients [135]. For the vertical FL, the feature representation can be obtained by minimizing the distance between features of aligned instances in different datasets.
- *Parameter-based FTL* aims to exploit shared parameters or prior distributions of hyper-parameters between the source domain and target domain models to effectively encode the transformation knowledge [60, 74, 136]. For the horizontal FL, a shared global model is firstly trained based on datasets of different clients. Each client can then fine-tune its local model using the pre-trained global model on its local dataset [121]. For the vertical FL, predictive models trained on the aligned instances can be first used to infer missing features or labels for unaligned data instances of the clients. Then, a more accurate model can be obtained by training on the expanded datasets.

2.3 Related Concepts

In this section, we introduce some related concepts and provide a brief comparison with the FL, as summarized in Table 2.

2.3.1 *Distributed Machine Learning.*

Distributed machine learning is a combination of distributed computing and machine learning, which aims at accelerating the training process on large-scale datasets [58]. There are typically two distribution schemes: data parallel and model parallel [178]. In the data-parallel scheme, the training data are first divided into many partitions, with the number of partitions being equal to the number of local nodes. The local nodes act as computing resources, train the same model on the individual dataset, and then send the local model parameters to a parameter aggregator. Compared to FL, the local nodes hold the same machine learning model. In the model-parallel scheme, a machine model is first divided into a few partitions and distributed to multiple local nodes. Each local node then trains a part of the model on a copy of an entire dataset. A aggregator is deployed to allocate computing tasks and aggregate all model parts [76].

Table 2. A Brief Summary of Related Concepts and Comparison with FL

Related concepts	**Primary features**	**Comparison with FL**
Distributed Machine Learning	• data-parallel scheme • model-parallel scheme • multiple local nodes & an aggregator • acceleration for large-scale datasets • homogeneous data	◇ large-scale datasets ◇ distributed computing ⋇ privacy-preserving ⋇ may no central aggregators ⋇ no exchange of datasets ⋇ heterogeneous/homogeneous data
Mobile Edge Computing	• three-layer architecture: end-users, edge servers & a cloud server • low-latency (e.g., augmented reality) • IoT devices (e.g., smartphones)	◇ privacy-preserving ◇ large-scale datasets ◇ distributed computing ⋇ may no servers ⋇ no exchange of datasets
Split Learning	• model splitting design • collaborative learning • clients & a central server • communication efficiency	◇ large-scale datasets ◇ no exchange of datasets ⋇ entire model training ⋇ may no central aggregators ⋇ privacy-preserving
Privacy-Preserving Machine Learning	• conventional machine learning • usually centralized computing • a combination of privacy-preserving techniques and machine learning	◇ privacy-preserving ◇ neural network ⋇ distributed computing ⋇ multiple participants ⋇ multiple private datasets ⋇ large-scale datasets

◇: Main common features. ⋇: Main different features.

2.3.2 *Mobile Edge Computing.* Mobile edge computing, also known as multi-access edge computing, is a network architecture concept designed to enable a cloud computing service at the edge clients of the cellular network [2, 5]. Technical standards for the mobile edge computing were developed by the European Telecommunications Standards and Institute Industry Specification Group.[3] The key idea of the mobile edge computing is to reduce the network congestion by conducting computation tasks closer to the edge clients, and thus it can enable the rapid deployment of new services or applications for customers. Therefore, mobile edge computing emphasizes the development of network communication techniques and can benefit the development of FL. Recently, several research studies on FL in mobile edge computing have been proposed [49, 105, 116, 140, 189, 190, 201, 220].

2.3.3 *Split Learning.* Split learning is a distributed learning concept, which enables collaborative machine learning without the exchange of datasets with a central server [67, 177]. First, each client trains a model up to a specific layer, which is referred to as a cut layer. Then, the intermediate parameters are transmitted to the central server to complete the training of the remaining layers. Finally, the aggregated gradients are back propagated to the cut layer and sent back to the clients to complete the local training. By contrast, FL typically involves the communication of a whole model's parameters or gradients. An empirical comparison of the communication efficiencies between the split learning and FL can be found in Reference [162]. This study shows that when the model is large or there are numerous clients, the communication efficiency of the split learning is better than that of FL. This is because the clients of split learning do not transmit all model parameters to the central server. However, because the clients and central server in FL run the same global model, FL is easier to implement than split learning.

[3] https://portal.etsi.org/TB-SiteMap/MEC/MEC-White-Papers.

2.3.4 Privacy-preserving Machine Learning. **Privacy-preserving Machine Learning (PPML)** aims to employ privacy-preserving techniques in machine learning to preserve data privacy. PPFL can be treated as a special case of PPML. The main difference is that PPFL emphasizes a collaborative training with datasets separately stored in different devices. Typically, PPML methods can be divided into three classes: **homomorphic encryption- (HE)** based PPML, **secure multi-party computation– (SMC)** based PPML, and **differential privacy- (DP)** based PPML [18]. HE-based PPML uses encrypted datasets for model training or querying [15]. For example, Gilad-Bachrach et al. [64] proposed a HE-based CNN called CryptoNets for securely exchanging data between clients and a cloud server. As a disadvantage, its accuracy is affected by the number of non-linear layers. To improve the performance, Chabanne et al. [18] proposed combining CryptoNets with a polynomial approximation for an activation function. SMC-based PPML utilizes the SMC technique [65, 207] to ensure data privacy. For example, Mohassel et al. [124] proposed a three-party computation-based framework for PPML. As an advantage, it can be used to train general machine learning models such as logistic regression, linear regression, and neural networks. DP-based PPML utilizes the DP technique to achieve data privacy preservation [66]. DP-based PPML methods can be divided into two categories: privacy-preserving models designed by adding noise to the trained model [22, 85], and privacy-preserving models designed by adding noise to the objective function [21, 219].

3 OVERVIEW OF GENERIC PRIVACY-PRESERVING MECHANISMS

In computer security, privacy is defined to "assure that individuals control or influence what information related to them may be collected and stored and by whom and to whom that information may be disclosed" [166]. Privacy is protected under legal regulations such as the *General Data Protection Regulations* in the European Union [179]. Privacy-preserving mechanisms are designed to realize data utility while ensuring that the original information will not be disclosed to other individuals or groups. In this section, we provide a brief overview of generic privacy-preserving mechanisms covering three types of privacy preservation techniques: cryptographic techniques, perturbative techniques, and anonymization techniques.

3.1 Cryptographic Techniques

The cryptographic techniques widely used in privacy-preserving machine learning primarily consist of homomorphic encryption, secret sharing, and secure multi-party computation.

Homomorphic encryption [148] is a form of encryption. Formally, an encryption algorithm **En** is called homomorphic over an operator $\star$ if fulfilling the criterion $\mathbf{En}(m_1) \star \mathbf{En}(m_2) = \mathbf{En}(m_1 \star m_2), \forall m_1, m_2 \in \mathbb{M}$, where $\mathbb{M}$ denotes a set of plaintext [98]. According to the supported operators, homomorphic encryption methods can be divided into two categories: partially homomorphic encryption [134] and fully homomorphic encryption [176]. Partially homomorphic encryption only supports either additive or multiplicative operations, which are referred to as additively homomorphic encryption [88] and multiplicative homomorphic encryption [46], respectively. Fully homomorphic encryption supports both additive and multiplicative operations [62]. Compared with partially homomorphic encryption, fully homomorphic encryption provides stronger encryption but suffers from computation costs.

Secret sharing [156] is a cryptographic scheme where a secret key consisting of n shares can be reconstructed only if a sufficient number of shares are combined. Formally, a dealer distributes a secret s to n participants, each of whom is allocated one share. Then, the (t, n)-threshold secret sharing is defined by fulfilling the following criteria [98]: (1) For an arbitrary subset of m participants with $m \geqslant t$, the secret s can be reconstructed from m shares, and (2) for an arbitrary subset of m participants with $m < t$, the secret s cannot be reconstructed from m shares. The methods proposed in References [12, 156] are two widely used (t, n)-threshold secret sharing methods.

However, these methods are vulnerable to the dishonest dealer or malicious participants. Therefore, verifiable secret sharing is proposed to prevent these two types of attacks [72, 79].

SMC [207] is a cryptographic scheme that enables distributed participants to collaboratively calculate an objective function without revealing their data. Formally, n participants $\{P_1, P_2, \ldots, P_n\}$ aim to compute a global function $f(\mathcal{D}_1, \mathcal{D}_2, \ldots, \mathcal{D}_n)$ based on each individual's dataset $\mathcal{D}_i$. A protocol is said to be an SMC protocol if meeting the following two requirements [98]: (1) correctness, in which the function $f(\mathcal{D}_1, \mathcal{D}_2, \ldots, \mathcal{D}_n)$ calculated by the protocol must be correct, and (2) privacy, in which the protocol does not reveal any private information of $\mathcal{D}_n$ to any other participants. As its advantages, (1) there is no requirement of trusted third-parties, (2) the tradeoff between data utility and data privacy is eliminated, and (3) a high accuracy is achieved. The disadvantages are computational overhead and high communication costs.

3.2 Perturbation Techniques

The key idea of a perturbation technique is to add noise to the original data, allowing the statistical information calculated from the perturbed data to be statistically indistinguishable from the original data. There are three types of widely used perturbation techniques: differential privacy, additive perturbation, and multiplicative perturbation.

A differential privacy technique is based on probability statistical models to quantify the degree of disclosure of private information of instances in a dataset [41]. Typically, differential privacy techniques can be divided into two categories: global differential and local differential privacy techniques. Global differential privacy technique aims to achieve a goal in which if the effect of substituting an arbitrary sample in a dataset is sufficiently small, the query results cannot be used to explore more information about any samples in the dataset [43]. As an advantage, this technique is more accurate than local differential privacy techniques, because it does not need to add a large amount of noise to the dataset. A local differential privacy technique is introduced to remove the trusted central authority demanded in global differential privacy [34, 102]. Compared with a global differential privacy technique, a local differential privacy technique does not need a trusted third-party [146]. As a disadvantage, the total amount of noise is much larger than that in a global differential privacy technique.

Additive perturbation aims to preserve the privacy of the original data by adding random noise from a certain distribution (e.g., a uniform distribution or Gaussian distribution), which is formulated by $\mathbf{Y} = \mathbf{X} + \delta$, where $\mathbf{Y}$ denotes the perturbed data, $\mathbf{X} \in \mathbb{R}^{d \times n}$ denotes the original data, and $\delta \in \mathbb{R}^{d \times n}$ denotes the random noise [4]. This technique is simple and can preserve the statistical properties [86]. However, it may degrade the data utility and may be vulnerable to a noise reduction.

Multiplicative perturbation aims to multiply the original data using noise from a certain distribution [25]. Instead of adding some random noise to the original data, multiplicative perturbation aims to transform the original data points to a certain space [108]. Compared with the additive perturbation, multiplicative perturbation is more effective, because reconstructing the original data from the perturbed data of the multiplicative perturbation is more difficult [50].

In summary, perturbation techniques are simple, efficient, and usually do not require knowledge of the data distribution. However, perturbed data may be vulnerable to probabilistic attacks, and it is difficult to mitigate such risk without reducing the data utility.

3.3 Anonymization Techniques

Anonymization techniques are mainly used to achieve group-based anonymization by removing the identifiable information while maintaining the utility of the published data. There are three types of widely used anonymization techniques: k-anonymity, l-diversity, and t-closeness. These techniques are primarily developed for the structured data [152] in the form of a table consisting of multiple records, with three main types of attributes: **unique identifiers (UIDs)**, **sensitive attributes (SAs)**, and non-sensitive attributes.

k-anonymity aims to protect the data privacy while maintaining the utility of the published data [152]. A published dataset is said to satisfy the k-anonymity if each sample in the dataset cannot be re-identified from the published data of at least $k - 1$ samples. As an advantage of k-anonymity, it is effective in protecting the privacy of UID-based records [186]. However, it may be vulnerable to inference attacks on the SAs of the records.

In addition, l-diversity, an extension of k-anonymity, was proposed to enhance the protection of the SAs of the records [118]. Compared with k-anonymity, the l-diversity technique adds diversity within a group to the SAs in the anonymization mechanism [171]. However, l-diversity may be vulnerable to attribute linkage attacks, because the SAs may be inferred from the distribution of values.

Thus, t-closeness was proposed to enhance l-diversity by maintaining the distribution of the SAs [95]. As an advantage of t-closeness, it tends to be more effective than l-diversity and k-anonymity for numeric attributes [188]. However, enforcing t-closeness between distributions tends to degrade the utility of the data.

3.4 Privacy-preserving Metrics

There are two types of metrics widely used to assess the performance of privacy-preserving methods: (1) privacy metrics for measuring the loss of privacy of a dataset and (2) utility metrics for measuring the data utility of the protected data for data analysis purposes [171].

Privacy metrics aim to measure the privacy loss of privacy-preserving techniques [180]. Wager et al. [181] thoroughly reviewed more than 80 privacy metrics and discussed these metrics from 4 common aspects: adversary models, data sources, inputs for the computation of the metrics, and output measures. For example, the adversary models analyze the capabilities and goal from the perspective of an adversary, and the data sources introduce privacy leakage issues from the perspective of the data sources. Typically, the privacy metric for a certain case is determined by many aspects, such as the inputs, data source, and adversary models. In most cases, a single metric may not be able to fully assess the complete privacy.

Utility metrics are designed to quantify the utility of the data protected by privacy-preserving techniques for data analysis purposes, i.e., general analysis purposes and specific analysis purposes [55]. For general analysis purposes, information loss metrics are defined to measure the similarity between the original data and the protected data [23]. They are usually measured by the extent to which the protected data retain the statistical information of the original data. For the specific analysis purposes (e.g., machine learning and statistical analysis tasks), the data utility is measured by comparing the evaluation accuracy of the task using the protected data and the original data.

4 PRIVACY-PRESERVING FEDERATED LEARNING: A TAXONOMY AND REVIEW

PPFL is an ingenious combination of FL and privacy-preserving mechanisms. Its key challenge is to balance the tradeoff between data privacy and data utility when applying privacy-preserving mechanisms to FL frameworks. Based on the introduction of FL and the generic privacy-preserving mechanisms in Section 2 and Section 3, in this section, we present a comprehensive and systematic overview of PPFL. First, we present an overview of the proposed 5W-scenario-based taxonomy. Then, we analyze potential privacy leakage risks in FL from five aspects according to the 5W-scenario-based taxonomy. Finally, we investigate and summarize PPFL methods according to four privacy-preserving schemes.

4.1 Proposed 5W-Scenario-Based Taxonomy

As shown in Figure 1, the proposed 5W-scenario-based taxonomy emphasizes two aspects of PPFL: (1) potential privacy leakages and (2) possible privacy-preserving schemes. Concerning the potential privacy leakages, we will discuss the potential privacy risks in FL from five fundamental aspects: “who” (internal and external attackers), “what” (active and passive attacks), “when” (training and inference phases), “where” (weight update, gradient update, and the final model), and “why” (four types of inference attacks). For the possible privacy-preserving schemes, we divide the existing PPFL methods into four categories: encryption-based, perturbation-based, anonymization-based, and hybrid PPFL.

4.2 Potential Privacy Leakage Risks in FL

FL offers a framework to jointly train a global model using datasets stored in separate clients. However, recent studies have shown that FL may not always guarantee sufficient privacy preservation [61]. This is mainly because the model parameters (e.g., weights or gradients) may leak sensitive information to malicious adversaries and cause deep privacy leakage [10, 228]. As reported in [137], a small portion of the original gradients could reveal privacy about local training datasets. Therefore, in this section, we provide a comprehensive analysis of potential privacy leakage risks in FL from five fundamental aspects.

4.2.1 Scenario 1: Who might be a malicious adversary? Insiders and outsiders. In FL, there are two types of actors who can obtain access to the model information: internal actors (participating clients and the central server) and external actors (model consumers and eavesdroppers).

One type of potential malicious adversaries may come from the internal actors: participant clients or the central server. In a typical FL framework, the participant clients collaboratively train a global model using their local datasets with or without the orchestration of a central server, which is required in the client-server architecture of the horizontal FL or in the vertical FL architecture with the third-party coordinator [204]. For example, in the client-server architecture, the central server is designed to aggregate the model updates and orchestrate the entire training process by receiving and sending training updates from and back to the participants [205]. Therefore, either the clients or the server has the opportunity to access the intermediate training updates (e.g., weights and gradients) and the final model. Hence, certain participants and an honest-but-curious server might be the potential internal malicious adversaries whose objective is to gain access to private data.

Another type of potential malicious adversary might come from external actors: model consumers or eavesdroppers. Model consumers usually have two ways to probe the private data. First, they can obtain access to the whole model weights, and second they can obtain access query results provided by a platform API [175]. Therefore, they can probe the private data through the final model or its query results. Compared with the model consumers who can officially and easily access the final model, eavesdroppers can steal the intermediate training updates or the final model by intercepting the communication between the participants and server [187]. However, this eavesdropping process usually needs more effort. In addition, the eavesdroppers may purloin the intermediate weights or gradients transmitted between participants and the aggregator [29]. Hence, certain model consumers and the eavesdroppers might be the potential external adversaries.

Risk Assessment. If a client is malicious, then it may cause privacy leakage under three situations. First, the malicious client can obtain the intermediate training updates from the aggregator to explore private information of other client datasets [10]. Second, the malicious client might send deliberately designed training updates to the aggregator for probing the special private data of other client datasets [123]. Third, a privacy leakage might be caused by the information inference attack launched by the malicious client, for example, by exploiting the final model [129]. Although these risks can be relieved by designing specific training protocols, for example, selecting a portion of clients in each round, FL still faces a substantial risk of privacy leakage from malicious clients.

If the server is malicious, then it will cause high-level risks of privacy leakage [61]. First, because the malicious server maintains the intermediate updates and the final model, the server can utilize those information to reconstruction attacks [223]. Second, the malicious server has the ability to explore the private data of specific clients by differentiating their model updates. Third, the malicious server has the ability to select and regulate the participating clients in each training round to explore the privacy of the training datasets [200]. Given that the role of the server is to aggregate model updates, a simple privacy-preserving solution is to limit and quantify the server's ability to reconstruct the local training datasets of the clients.

Because consumers of a model can easily access the final model or its query results, if they are malicious adversaries or controlled by attackers, the malicious consumers can utilize that information to probe sensitive information through information inference attacks (Section 4.2.5). For example, the malicious consumers can utilize that information to generate representative samples [75, 192], determine whether a sample has been used for model training [123, 130], obtain characteristics of the training data [123], or generate the training samples and labels [223, 228]. Therefore, malicious consumers may cause high-level risks of privacy leakage.

The eavesdroppers attempt to probe sensitive information by stealing the intermediate training updates or the final model transmitted between participants and the server [45, 224]. As a risk, they can utilize this information to explore private information of the training datasets, such as reconstructing the training samples [153]. This will cause a seriously sensitive information leakage and incur a high risk of privacy leakage. For example, research studies [223, 228] have shown that the training samples can be reconstructed by the publicly shared gradients. Compared to malicious consumers, eavesdropping may be defended by cryptographic encryption mechanisms (e.g., homomorphic encryption, secret sharing, or secure multi-party computation).

4.2.2 Scenario 2: What types of privacy attacks? Passive and active attacks. According to RFC 4949 [159], passive attacks are defined as those aimed at using information or learning from a system but not modifying the system, whereas active attacks are defined as those aimed at altering system resources or affecting system operations. With FL, passive attackers only observe computations (e.g., the weights, gradients, and final model) during the training and inference phases [123, 228], whereas active attackers can influence the FL system by manipulating the model parameters to achieve adversarial goals [129, 165].

Passive attacks in FL can be divided into two categories: passive black-box and passive white-box attacks. In a passive black-box attack, for example, in the setting of a service platform, it is assumed that the adversary can only access query results but not the model parameters or intermediate training updates. Nasr et al. [129] studied this type of attack by aiming at the membership inference. In a passive white-box attack, it is assumed that the adversary can access the intermediate training updates, model parameters, and query results [129].

Active attacks in FL aim to actively influence the training process and extract sensitive information about the training datasets [129]. There are diverse ways to carry out this type of attack. For example, Melis et al. [123] designed an active attack in which the adversary uploads a special gradient to the global model to learn separable representations. Song et al. [165] presented an active attack from the perspective of a malicious server by isolating certain clients. In addition, Xu et al. [200] proposed an active attack by controlling certain clients to adjust the training data strategically such that the global model rises or falls with a special pattern.

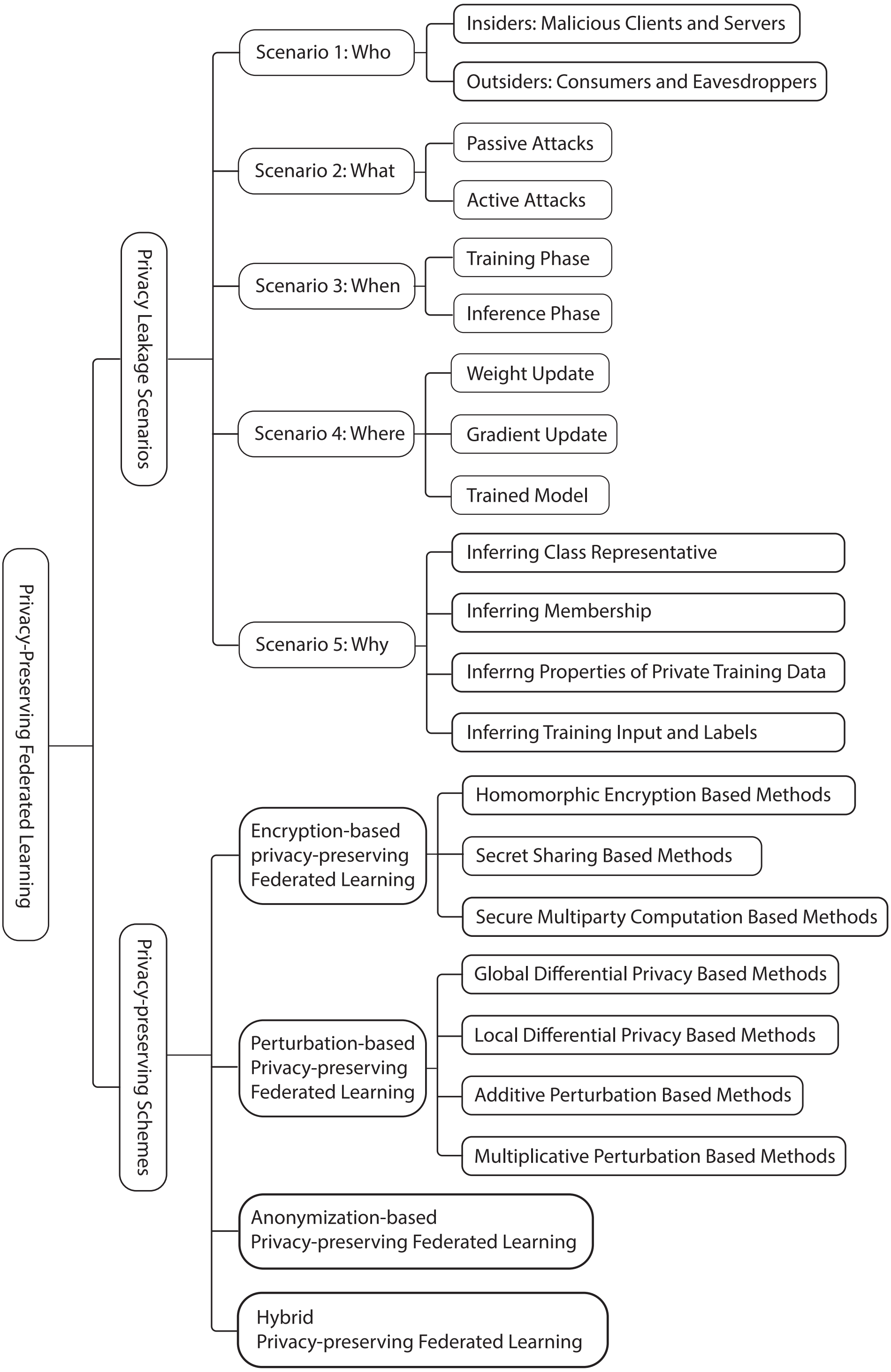
Privacy-Preserving Federated Learning
Privacy Leakage Scenarios
Scenario 1: Who
Insiders: Malicious Clients and Servers
Outsiders: Consumers and Eavesdroppers
Scenario 2: What
Passive Attacks
Active Attacks
Scenario 3: When
Training Phase
Inference Phase
Scenario 4: Where
Weight Update
Gradient Update
Trained Model
Scenario 5: Why
Inferring Class Representative
Inferring Membership
Inferrng Properties of Private Training Data
Inferring Training Input and Labels
Privacy-preserving Schemes
Encryption-based privacy-preserving Federated Learning
Homomorphic Encryption Based Methods
Secret Sharing Based Methods
Secure Multiparty Computation Based Methods
Perturbation-based Privacy-preserving Federated Learning
Global Differential Privacy Based Methods
Local Differential Privacy Based Methods
Additive Perturbation Based Methods
Multiplicative Perturbation Based Methods
Anonymization-based Privacy-preserving Federated Learning
Hybrid Privacy-preserving Federated Learning

Fig. 1. Proposed taxonomy to Privacy-preserving Federated Learning.

In FL systems, an attacker can conduct both active and passive attacks [123, 129]. An active attacker may be a client that maliciously modifies its local uploads, or it may be a central server that maliciously modifies the aggregation parameters. In a passive case, the attacker only observes the updates and conducts inference, without changing anything during the training phase. In an active case, the attacker can affect the model training and conduct more powerful attacks against other participants. More specifically, the attacker can share malicious updates, thereby deceiving the FL model into revealing more sensitive information about the local data of other clients that the attacker is interested in.

Risk Assessment. A passive black-box attack usually has a limited risk, because attackers can only utilize the query results to probe sensitive information [129], whereas a passive white-box attack can usually reveal more sensitive information about training datasets. Nasr et al. [129] investigated a passive white-box attack as well as a passive black-box attack against the membership inferences and showed that the former will be much stronger than the latter in the membership information inference. Passive attacks usually consist of different types of inference attacks, where the adversary only observes the training process and obtains access to model parameters without modifying the learning process. For example, Hitaj et al. [75] studied a passive white-box attack based on a **generative adversarial network (GAN)** to effectively extract sensitive information (clients' face images) of training datasets.

Compared with passive attacks, active attacks are much more powerful, because an adversary can modify the model updates on the client or server side. For example, to extract more private information about the training datasets, gradient updates maliciously modified by a client can mislead the global model into learning specific features without a significant impact on the model performance [123]. Nasr et al. [129] designed an active attack that actively misleads the stochastic gradient descent into leaking more sensitive information about the datasets of other participants. Under this active attack setting, the attacker aims to update the data features towards lifting off the gradients of the global model or local model. Specifically, the stochastic gradient descent algorithm iteratively minimizes the loss function on the training data by decreasing the gradient of the loss. The magnitude of the gradient change depends on the contribution of the training data samples in the loss function. If data samples lead to a large loss of the loss function, then the stochastic gradient descent algorithm will update the model to reduce the loss during the training phase. If data samples do not contribute to a loss of the loss function, then the gradient change on these samples is gradual throughout the training phase. When the attacker lifts off the gradients of the target instances, the gradient norm of the target members will become indistinguishable to that of the non-target member instances. The active gradient ascend attacker will make the target model behave differently between target member instances and non-target member instances. This can render an easier membership inference attack.

4.2.3 Scenario 3: When might a data privacy leakage occur? Training phase and inference phase.

With FL, there are two primary phases: the training phase and the inference phase. The training phase mainly involves computing the local gradient on the client-side, aggregating the global model on the server-side, transmitting intermediate updates between the clients and the aggregator, and releasing the final model to the clients [69, 96]. The inference phase mainly involves a way to provide the query service to consumers [129]. Both phases are vulnerable to a privacy leakage.

Risk Assessment. During the training phase, one risk of privacy leakage is mainly related to model updates. This is because all update information during the training phase is potentially exposed to malicious adversaries [89, 91]. This information includes local gradients, local model weights, aggregated gradients or model weights, and the final model [121, 138]. For example, during each training round, the aggregated gradients or the model weights are available to multiple selected participants [218]. If one of the participants is malicious, then the adversary can gather the aggregated model updates and conduct privacy attacks to infer the data privacy of the other participants. In addition, during the training phase, the adversary may send malicious local updates to mislead the global model into learning specific features [123]. As another risk, the model updates during the training phase are potentially exposed to eavesdroppers. For example, when the selected clients send the local updates (e.g., local gradients or model weights) to the server for the model aggregation or when the server sends the aggregated updates back to the clients, the update information communicated between the participants and the aggregator may be leaked to eavesdroppers [223, 228]. To limit the privacy leakage from the updates or the final model, a user-level differential privacy can be applied during the FL training process [6, 44, 83].

During the reference phase, the risk of a privacy leakage is mainly related to the final model, which is released to the participants or is provided as a service platform. There are two main risks of privacy leakage: (1) attacks based on model parameters and (2) attacks based on model queries. In the first case where the adversaries can access the model parameters and thereby obtain the query outputs as well as all intermediate computations, the adversaries can use the model parameters to conduct inference attacks to extract sensitive information about the training datasets of the participants [53, 129]. For example, Fredrikson et al. [53] presented an inversion method to extract private information during the reference phase. The experiment results showed that this attack can reveal face information of the users. Melis et al. [123] investigated the privacy leakage of the membership during the inference phase and showed that positions of the words in a batch can be revealed from a deep learning model. In the second case where it is assumed that the adversaries can only obtain the model query outputs (for example, the query outputs returned by a machine learning-as-a-service API [175]), a privacy leakage is mainly caused by inference attacks. For example, Shokri et al. [160] studied the privacy leakage during the inference phase by investigating the inference membership attack against the model query results, where an inference model was trained to distinguish between the training and non-training data samples.

4.2.4 Scenario 4: Where might a data privacy leakage occur? Weight update, gradient update, and the final model. With FL, three types of important data need to be transmitted between the participants and the aggregator: the local weight/gradient, an aggregated weight/gradient, and the final model, all of which contain the necessary information that can be utilized to reveal sensitive information about the training datasets [53, 212]. In gradient/weight-update-based FL frameworks, clients send the gradients/weights to the server; the server aggregates the received data and sends them back to the clients for model updating [107, 115]. In deep network models, the gradients are usually calculated by back-propagating the loss of the training datasets through the entire network. Specifically, the gradient of a layer is calculated using the current layer's data features and the error from the upper layer. Similarly, the local model weight is calculated based on the participant's local dataset. Therefore, they (weight update, gradient update and the final model) contain sensitive information of local data [133, 142, 167]. The final model in the FL is collaboratively trained by private datasets of multiple participants and encodes essential information about these datasets [53].

Risk Assessment. In gradient-update-based FL frameworks [91, 210], as the model gradients are derived from the participant's private training dataset, the gradients may cause serious privacy leakage [129, 192]. An adversary (e.g., the participants or eavesdroppers) can conduct privacy attacks by observing, modifying, or eavesdropping gradient updates during the training process to infer private information of the training datasets, such as class representatives [192], membership [160], and training samples and labels [228]. For example, privacy attacks based on the gradient update presented in References [223, 228] have been successfully used to extract training samples and labels. Specifically, in Reference [228], the private training samples and labels are leaked by minimizing the loss between the attack and true gradients.

In weight-update-based FL frameworks [121, 138], the aggregated weight calculated by the server is available to multiple participants during each training round. Therefore, a weight update will leak the privacy of the training data of the participants to adversarial participants or eavesdroppers [200]. Malicious adversaries can save the parameters of the current FL model and conduct a property inference by exploiting the difference between the current FL model and the previous FL model. Xu et al. [200] showed that the model weight can be trained to reveal sensitive information regarding training datasets by controlling some participants during the training phase.

As the final model encodes essential information on all the datasets of the participants. Therefore, attacks based on the final model will cause a serious privacy leakage. Typically, there are two types of privacy leakages related to the final released model: (1) a model-parameter-based attack in which adversaries are assumed to be able to access the model parameters, and (2) a query-based attack in which adversaries are assumed to be able to obtain the query results of the model. Both types of model-based attacks were studied in Reference [129]. The experiment results showed that the model-parameter-based attacks leak much more sensitive information than the query-based attacks. In addition, Ateniese et al. [8] showed that an adversary can use the final model parameter to infer the ethnicity or gender of a user. Fredrikson et al. [53] presented an inversion method to extract sensitive information from trained models. The experiment results showed that this attack can reveal the privacy of the users by effectively reconstructing images of their faces based on the labels. Shokri et al. [160] investigated model-parameter-based inference membership attacks. In this study, predictions obtained by the final model on training data samples and non-training data samples were used to train an inference model.

4.2.5 Scenario 5: Why might a malicious attacker launch an attack? Inference attacks, including inference of class representatives, memberships , properties of training data, and training samples and labels. The purpose of a privacy attack is usually to infer sensitive information regarding training datasets, such as membership and class representatives. The inference attacks can be divided into four categories [47]:

- The inference of class representatives aims to generate representative samples, which are not real data instances of training datasets but can be used to study sensitive information about the training datasets [53, 75, 192].
- The inference of memberships aims to determine whether a data sample has been used for model training [42, 114, 123, 129, 130, 139, 160].
- The inference of properties of the training data aims to infer the property information regarding the training datasets [8, 54, 123].
- The inference of inferring training samples and labels aims to reconstruct the original training data samples and the corresponding labels [153, 223, 228].

Risk Assessment. *The inference of class representatives* attempts to extract class representatives, which are synthesized generic samples rather than the real data in the training datasets [192]. Fredrikson et al. [53] showed that many machine learning classifiers are vulnerable to privacy leakage. For example, facial recognition models can be used to extract sensitive features as inputs and generate face images from the model. If an adversary is one of the clients participating in the FL training phase, then class representatives can be obtained using GANs. When all data samples are similar, the extracted class representatives obtained by GANs will be similar to the training data. For example, the representatives of the handwritten digit "9" generated by the GAN model tend to be similar to any image of the digit and the training samples. In a special facial recognition case in which all images depict the same person, the reconstructed class representatives also tend to be similar to face images of this person. Hitaj et al. [75] designed a class representatives inference attack based on a GAN against the FL framework. The experiment results showed that this attack can effectively generate representative samples of training datasets. Wang et al. [192] proposed to extract data representatives by a GAN-based framework with a multi-task discriminator, which is used to simultaneously discriminate client identity and category of data samples.

The inference of a membership aims to determine whether a data sample was included in the training dataset [103, 114]. For instance, it can be used to infer whether the records of a specific patient have been used to train a classifier related to a certain disease. Shokri et al. [160] investigated the inference membership attack against the privacy leakage of machine learning. An inference model was trained to infer whether a data sample is in the training dataset. Truex et al. [175] extended this membership attack presented in Reference [160] to a more general setting and showed that membership inference attacks are data-driven and largely transferable. Melis et al. [123] investigated the membership privacy leakage from two aspects: embedding layers and gradients. It was shown that the non-zero gradients of the embedding layer of a deep learning model can reveal the positions of the words in a training batch. This enables an adversary to conduct a membership inference attack. Hayes et al. [73] presented an evaluation of membership inference attacks against generative models to detect an overfitting and recognize training inputs. The evaluation results showed that many models based on a deep convolutional GAN or a boundary equilibrium GAN are susceptible to privacy leakage.

The inference of the properties of training data aims to infer the property information on the training datasets [128, 158, 192]. In particular, these properties might be irrelevant to the main training task. For example, when the main task is to train a model for race or gender recognition, the property inference attack may intend to infer whether people in the training images wear glasses or infer race. Naseri et al. [128] evaluated a property inference attack proposed in Reference [158] on a face dataset to infer the race of a target client. Melis et al. [123] investigated passive and active property inference attacks to infer uncorrelated properties of the training data. In the passive property inference attack setting, the adversary is assumed to have auxiliary data labeled with the target property, and to be able to eavesdrop on the model updates, but unable to maliciously modify the training process. The adversary first uses the global model parameters to iteratively calculate positive gradient updates from the data with the target property and calculate negative gradient updates from the data without the target property. The labeled gradient updates are then used to train a binary classifier for the property inference attack. In the active property inference attack setting in Reference [123], the adversary follows the FL protocol but uploads specific local gradients to lead the global model to learn separable property without maliciously modifying the training process. The goal is to guide the global model to learn separable representations of the data with and without the target property. However, these property inference attacks have an assumption in that there are auxiliary training datasets labeled with the target property. For general cases, it is easy to obtain the auxiliary datasets. For example, the auxiliary data for a gender inference can be any datasets containing images with males and females. For specific cases, however, the auxiliary data may not be easily available, for example, the datasets used to infer the authorship for training handwritten texts.

The inference of training samples and labels aims to generate the original training data samples and the corresponding labels. Recent studies [194, 228] have shown that the training samples and labels can be obtained from the publicly shared model gradients. Zhu et al. [228] designed an optimization algorithm and successfully obtained the training inputs and labels within a few training rounds. In this setting, a dummy dataset consisting of dummy samples and labels is first randomly generated. Then, dummy gradients are obtained through normal forward and backward computations based on the dummy dataset. The algorithm iteratively optimizes the dummy samples and labels by minimizing the loss between the dummy gradients and real gradients. Finally, the attacker can obtain the training samples and labels. This indicates that this type of attack can be used to recover original images with pixelwise accuracy and tokenwise matching of original texts and is much stronger than previous shallow leakages [75, 123]. As an advantage, this type of attack does not require prior knowledge regarding the training datasets. As a disadvantage, defense strategies (e.g., perturbation-based schemes) can be effectively applied to model gradients to prevent a gradient leakage. Zhao et al. [223] improved the attack proposed in Reference [228] and proposed a stronger attack by exploiting the relationship between the ground-truth labels and the signs of the gradients. As an advantage, this method is suitable for any differentiable model trained with cross-entropy loss on one-hot labels. In addition, Sannai [153] introduced an attack framework based on deep neural network and showed that this approach can reconstruct training samples from loss functions.

4.3 PPFL Methods

In this section, we discuss different privacy-preserving schemes adopted in FL. According to the privacy-preserving techniques used in FL, the PPFL methods can be dividend into four categories: (1) encryption-based, (2) perturbation-based, (3) anonymization-based, and (4) hybrid PPFL methods.

4.3.1 Encryption-based PPFL. Encryption-based PPFL methods are mainly using cryptographic techniques for privacy preservation, which can be divided into three categories: (1) homomorphic encryption-based, (2) secret sharing–based, and (3) secure multiparty computation-based PPFL methods.

Homomorphic encryption-based PPFL methods. Homomorphic encryption allows a calculation directly to be conducted on a ciphertext to generate an encrypted result such that the decrypted result is the same as the result calculated on the corresponding plaintext. This scheme is an effective way to protect data privacy when exchanging intermediate parameters during the FL training process, and has been widely used in many FL methods [7, 26, 39, 69, 71, 137, 215, 221, 224]. For example, Phong et al. [137] proposed using homomorphic encryption to protect gradient updates during the FL training process. With this method, each participant calculates its local gradient using its local dataset, and then encrypts the local gradient, and sends the encrypted gradient to the server for aggregation. Chen et al. [26] developed an FL framework for wearable healthcare using homomorphic encryption. With this framework, additively homomorphic encryption is used to encrypt the local model of each participant, which is sent to the server for model aggregation. Each participant then decrypts the encrypted model sent by the server and applies a local update. Hao et al. [69] proposed an FL framework based on the fully homomorphic encryption [14]. However, the homomorphic encryption utilized in these studies brings about a large communication and computational overhead. Zhang et al. [215] presented a batch encryption-based FL framework to reduce the computational costs. Specifically, a batch of gradients is encoded into a long integer data type instead of a double data type. Zhang et al. [218] proposed a PPFL approach based on homomorphic encryption by encrypting local gradients. As an advantage, this approach utilized a distributed selective stochastic gradient descent procedure to reduce the computation costs.

Secret sharing–based PPFL methods. Secret sharing [156] is a cryptographic technique guaranteeing that a secret consisting of n shares can be reconstructed only when a sufficient number of shares are combined. Secret sharing has been used in many FL frameworks to achieve privacy preservation [38, 59, 113, 126, 157, 169, 198, 224]. For example, Bonawitz et al. [13] proposed a practical and secure framework for the FL based on the secret sharing. As an advantage, this framework allows the server to securely aggregate the model updates of the participants. Gao et al. [59] proposed a heterogeneous FTL framework based on the secret sharing. With this method, secret sharing was used for secure model updating. However, these methods are vulnerable to either a dishonest server or malicious participants. Xu et al. [198] proposed a PPFL framework supporting verification during the FL training process. With this method, the secret sharing [156] and key agreement protocol [37, 92] are used to protect the privacy of the local gradients of the participants during the model updating.

SMC-based PPFL methods. A secure multiparty computation is a cryptographic scheme that enables distributed participants to collaboratively calculate an objective function without revealing their own data. It has been widely used in many PPFL methods [13, 125, 127, 144, 157, 164, 199, 227]. Bonawitz et al. [13] introduced a federated deep learning approach that adopts an SMC-based secure aggregation protocol to protect individual model updates. With this approach, the central server cannot explicitly access any local updates but can still observe the exact aggregated results during each round. As an advantage, it can retain the original accuracy and achieve a high privacy guarantee. However, the secure aggregation protocol used in this approach incurs significant communication costs. A challenge of SMC-based FL methods is to improve the computational efficiency, because significant computational resources are required to complete a training round in FL frameworks [147].

4.3.2 Perturbation-based PPFL methods. This type of method can be divided into four categories: (1) global differential privacy-based, (2) local differential privacy-based, (3) additive perturbation-based, and (4) multiplicative perturbation-based PPFL methods.

Global differential privacy-based PPFL methods. The global differential privacy scheme has been widely used in many FL methods [3, 31, 40, 63, 69, 70, 78, 94, 128, 149, 172, 193]. Geyer et al. [63] presented an FL framework based on global differential privacy by incorporating the Gaussian mechanism to protect client datasets. Specifically, during each training round, the server selects a random number of participants to train the global model, and the participants update their local models and send weights back to the server. The server then aggregates the global model by adding random Gaussian noise. In this way, malicious participants cannot infer the information of other participants from the shared global model. However, this framework is vulnerable to a malicious server, because the server can obtain "clean" model updates from the clients. Compared with a method [63] of adding noise to the aggregation update, Hao et al. [70] proposed adding noise to the local gradients. McMahan et al. [122] proposed a user-level privacy-preserving language model based on the differential privacy. In particular, the differential privacy technique is applied to the federated averaging algorithm to protect user privacy through large step updates from user-level data. As an advantage, this method can achieve a comparative predictive accuracy with privacy guarantees. However, this method inevitably incurs a significant computational overhead because of the tradeoff between data privacy and utility. In summary, as an advantage of the global differential privacy-based FL methods, they provide good accuracy while preserving data privacy. That is, because this scheme is applied to an entire dataset allowing it to a good statistic distribution by adding a limited amount of noise.

Local differential privacy-based PPFL methods. A local differential privacy scheme has been widely used in many FL methods [10, 16, 109, 116, 128, 155, 168, 174, 191, 195, 225, 226]. Abadi et al. [1] proposed using a local differential privacy scheme to train deep networks. The privacy preservation in this method is achieved by two operations: (1) limiting the sensitivity of each data sample by clipping the norm of its gradient, and (2) adding noise to the gradient of a batch. However, they did not apply this method to the FL systems. Zheng et al. [226] compared the FL and the local differential privacy in terms of the efficiency and privacy loss. However, they did not investigate the performance of applying local differential privacy to the FL. Bhowmick et al. [10] proposed an FL method based on a local differential privacy to defend against reconstruction attacks. In a client-side computation, the local differential privacy is used to protect the privacy of each sample; in addition, in the server-side computation, the local differential privacy was used to guarantee the privacy preservation of the global model update. Lu et al. [116] proposed a differentially private asynchronous FL for the mobile edge computing and used the local differential privacy for local model updates. Cao et al. [16] developed an FL framework based on the local differential privacy to achieve the privacy-preserving in the Internet-of-Things [141]. Truex et al. [174] proposed an FL framework with local differential privacy to large-scale network training. Wang et al. [191] introduced a local differential privacy FL framework for industrial-grade text mining and showed that it can guarantee data privacy and model accuracy. In summary, compared with global differential privacy-based FL methods, the local differential privacy-based FL methods provide a stronger privacy guarantee.

Additive perturbation-based PPFL methods. Additive perturbation-based FL methods aim to preserve privacy by adding random noise to weight updates or gradient updates [19, 52, 63, 69, 70, 78, 110, 172, 193, 198]. In some methods [52, 78, 193], random noise was added to weight updates to achieve privacy-preserving in the training process, whereas in other methods [69, 70, 172, 198], random noise was added to the gradient updates. Compared with the above methods, Chamikara et al. [19] introduced a data perturbation method for a dataset perturbation and showed that the perturbation does not degrade the accuracy of the FL while preserving the privacy of the datasets. Hu et al. [78] proposed a personalized FL method by adding noise to the intermediate updates. Liu et al. [110] proposed an adaptive PPFL method by injecting adaptive noise to data attributes. In summary, as an advantage, this scheme is simple, can preserve the statistical properties, and does not require knowledge of the original data distribution. As a disadvantage, the data perturbation may degrade the data utility and may be vulnerable to a noise reduction [86].

Multiplicative perturbation-based PPFL methods. Instead of adding random noise to data, a multiplicative perturbation transforms the original data into another space [19, 20, 52, 57, 81, 145, 216]. Gade et al. [57] proposed a PPFL method by obfuscating the stochastic gradient by using multiplicative perturbations to protect the gradients from the curious server. In methods in References [20, 216], multiplicative weight update–based FL frameworks are sued to apply a local weight update to prevent the leakage of the gradient information to curious servers. Multiplicative perturbation-based FL methods [19, 81] were proposed for the Internet-of-Things to preserve the data privacy. Jiang et al. [81] proposed a PPFL method for IoT objects by applying an independent Gaussian random projection to obfuscate the data of each IoT object. Chamikara et al. [19] applied a multiplicative perturbation mechanism to fog devices. However, this method is vulnerable to an honest-but-curious server, because the perturbation mechanism is controlled by global perturbation parameters generated by the central server. In summary, compared with additive perturbation-based FL methods, multiplicative perturbation-based FL methods tend to be more effective, because the reconstruction of the original data values is more difficult.

4.3.3 Anonymization-based PPFL.

Although perturbation-based PPFL methods can provide a strong guarantee for privacy preservation, they also incur a degradation of the data utility [31, 63]. Therefore, anonymization-based FL methods have been proposed for privacy preservation [32, 56, 165, 197, 222]. For example, Song et al. [165] investigated anonymization schemes against a user-level privacy attack and proposed a GAN-based FL framework with a multi-task discriminator. Choudhury et al. [32] proposed a syntactic method for PPFL. This method was proposed to improve the data utility and the model performance while providing a defensible level of data privacy that adheres to the legal regulations (such as the EU General Data Protection Regulation and the US Health Insurance Portability and Accountability Act). Specifically, this method consists of two core components. The first core component is to apply the k-anonymity scheme [33] on the original private data on the client side and utilize the anonymized data to collaboratively train a global model. Because the anonymization operation is performed on data records composed of relational and transactional parts, it can protect data privacy from attacks of malicious adversaries who may have knowledge regarding these two data parts. The second core component is a global anonymization mapping process used for the prediction of the FL global model. The results showed that this method achieves better privacy preservation and model performance than differential privacy-based FL methods.

4.3.4 Hybrid Privacy-preserving Federated Learning. Because cryptographic technique-based FL methods tend to suffer from computation and communication overhead, and perturbation-based FL methods tend to degrade the data utility, in recent years, hybrid PPFL methods have been proposed to balance the tradeoff between data privacy and data utility [30, 59, 69, 70, 127, 173, 198, 199, 221, 224]. For example, the hybrid PPFL methods in References [59, 198, 224] are based on homomorphic encryption and secret sharing. The PPFL methods in References [69, 70] are developed by integrating homomorphic encryption with differential privacy. The method proposed in References [30] combines homomorphic encryption and secure multiparty computation. The FL methods presented in References [127, 173, 199] are based on secure multiparty computation and differential privacy. In Reference [173], SMC is used to guarantee that the proposed FL framework will not reveal updates exchanged with a differential privacy protection, and thus private information will not be leaked during the training process. As an advantage, this method can reduce the amount of the injected noise while guaranteeing privacy and maintaining a pre-defined rate of trust. In Reference [198], secure sharing and additive perturbation are deployed in a FL method to protect the local gradients in the training process.

4.4 Summary of the Relationship between the Taxonomy and Latest Techniques

The proposed 5W-scenario-based taxonomy in Figure 1 emphasizes two aspects of the PPFL: (1) potential privacy leakage risks under the five scenarios and (2) privacy-preserving schemes used in the existing PPFL. In relation to the first aspect, in Section 4.2, we reviewed the existing literature related to privacy leakage in FL and discussed the potential privacy leakage risks from five different perspectives: "who" (internal and external attackers), "what" (active and passive attacks), "when" (training phase and inference phase), "where" (weight update, gradient update, and the final model), and "why" (four types of inference attacks). From the five perspectives, we discussed who might carry out what types of attacks by manipulating what parameters (where) and for what purposes (why). We believe that this provides useful clues for researchers to design efficient PPFL methods. In relation to the second aspect, in Section 4.3, we reviewed the existing PPFL methods. According to the privacy-preserving schemes adopted in such FL methods, we divided them into four categories: encryption-based, perturbation-based, anonymization-based, and hybrid schemes. In each category, we reviewed the specific privacy-preserving techniques utilized in such PPFL methods and discussed the advantages and disadvantages related to them. Finally, we summarize the relationship between the taxonomy and some existing approaches in Table 3.

5 SUMMARY AND OPEN RESEARCH DIRECTIONS

In this article, we presented a comprehensive and systematic survey on the latest developments of the PPFL based on our proposed 5W-scenario-based taxonomy. First, we provided an explicit overview of FL and generic privacy-preserving mechanisms. For FL, we provided a clear

Table 3. A Summary of the Relationship between the Proposed Taxonomy and the Existing Methods

		Privacy attack	Encryption-based FL			Perturbation-based FL				Anonymization-based FL	Hybrid-based FL
			CA1	CA2	CA3	CB1	CB2	CB3	CB4		
WHO	RA1	[61, 196, 228]	[26, 137, 218]	[157, 169, 198]	[144, 164, 199]	[63, 70, 193]	[16, 174, 191]	[78, 110, 228]	[52, 81, 216]	[32, 56, 222]	[30, 127, 173]
	RA2	[114, 175, 187]	[215, 221, 224]	[29, 111, 113]	[13, 199]	[3, 10, 40]	[109, 155, 168]	[45, 172, 193]	[20, 52, 57]	[56, 165, 197]	[173, 199, 224]
WHAT	RB1	[75, 160, 192]	[215, 218, 221]	[29, 113, 157]	[144, 199, 227]	[63, 70, 193]	[16, 109, 225]	[69, 78, 110]	[20, 52, 57]	[32, 165, 197]	[198, 199, 224]
	RB2	[129, 165, 196]	[26, 111, 221]	[157, 169, 224]	[164, 227]	[40, 78, 193]	[116, 168, 174]	[69, 172, 193]	[20, 81, 216]	[56, 165, 197]	[69, 127, 221]
WHEN	RC1	[96, 192, 200]	[215, 218, 221]	[59, 198, 224]	[13, 127, 199]	[3, 70, 78]	[116, 155, 225]	[69, 110, 193]	[20, 57, 216]	[56, 197, 222]	[30, 173, 198]
	RC2	[53, 123, 196]	[7, 39, 111]	[111, 113, 126]	[144, 164]	[80, 123, 128]	[128, 195]	[45, 194]	[20, 52]	[32, 222]	[127, 173, 224]
WHERE	RD1	[121, 138, 139]	[26, 71]	[29, 169]	[13, 127, 199]	[3, 63, 193]	[16, 168, 191]	[52, 78, 193]	[20, 52, 216]	[197, 199]	[127, 199]
	RD2	[61, 129, 194]	[39, 137, 215]	[157, 198]	[125, 164]	[10, 70, 172]	[109, 155, 174]	[110, 172, 228]	[57, 81]	[56, 165, 222]	[30, 198, 224]
	RD3	[8, 53, 160]	[111, 137]	[111, 169, 224]	[125, 144]	[3, 80, 123]	[123, 128]	[19, 69]	[52, 81]	[32, 56]	[30, 69, 173]
WHY	RE1	[53, 75, 192]	[111, 221]	[59, 224]	[164, 227]	[63, 123, 172]	[116, 195]	[110, 194]	[57, 81, 216]	[32, 165, 222]	[127, 173, 198]
	RE2	[103, 129, 160]	[137, 215]	[126, 157, 169]	[125, 144, 199]	[70, 128]	[123, 128, 168]	[63, 70, 78]	[20, 52]	[56, 197]	[30, 127, 173]
	RE3	[53, 123, 192]	[7, 111]	[59, 224]	[127, 227]	[123, 128, 172]	[225, 226]	[110, 193, 228]	[52, 145]	[32, 165, 222]	[127, 173, 198]
	RE4	[194, 223, 228]	[39, 221]	[29, 198]	[13, 144, 164]	[3, 40, 128]	[10, 155, 174]	[194, 198, 228]	[19, 57, 81]	[165, 197]	[30, 69, 221]

CA1- Homomorphic encryption-based FL CA2- Secret sharing–based FL CA3- Secure multiparty computation-based FL
CB1- Global differential privacy-based FL CB2- Local differential privacy based FL CB3- Additive perturbation based FL
CB4- Multiplicative perturbation-based FL RA1- Internal attacker RA2- External attacker
RB1- Passive attack RB2- Active attack RC1- Training phase
RC2- Inference phase RD1- Weight update RD2- Gradient update
RD3- Final model RE1- Inferring class representative RE2- Inferring membership
RE3- Inferring properties RE4- Inferring input & label

categorization based on the data partitioning and communication architectures. For the generic privacy-preserving mechanisms, we introduced three types of privacy-preserving techniques, two privacy-preserving metrics, and a brief comparison with the related concepts. We then analyzed potential privacy leakage risks in FL from five aspects based on the proposed 5W scenarios including who (internal and external attackers), what (active and passive attacks), when (training and inference phases), where (weight update, gradient update, and the final model), and why (four types of inference attacks). In addition, we investigated and summarized the current PPFL methods based on our well-organized four privacy-preserving schemes, including encryption-based PPFL, perturbation-based PPFL, anonymization-based PPFL, and hybrid PPFL.

Despite the rapid development of the PPFL in recent years, this research field is still a challenge and room remains for improving the existing frameworks and developing novel methods to enhance both data privacy and data utility. Some potential open research problems and directions are listed below:

- A way to effectively apply the privacy-preserving mechanisms described in Section 3 to FL frameworks for privacy preservation should be determined. The concept of FL provides a basic framework for privacy-preserving model learning, which allows participants to collaboratively train a global model using their respective datasets. However, there is no privacy guarantee in the basic framework [61, 123, 137]. To protect data privacy, privacy-preserving mechanisms have been widely developed in a variety of FL frameworks (Section 4.3). However, the privacy-preserving mechanisms adopted in such FL frameworks tend to degrade the accuracy or efficiency. Therefore, it is necessary to balance the tradeoff between data utility and data privacy when introducing privacy-preserving mechanisms to the FL. The metrics presented in Reference [181] may provide a good guidance for data utility and data privacy.
- In the FL, the intermediate weights or gradients communicated between participants and the aggregator may reveal sensitive information about the participants' training datasets, as shown in References [96, 137, 198, 212]. In general, there are two ways to protect such data privacy. One is to protect the communication channel from eavesdropping by utilizing encryption techniques (Section 3.1), such as homomorphic encryption, secret sharing, and

secure multi-party computation. However, encryption-based FL tends to suffer from computation and communication overhead. Therefore, it is necessary to develop an efficient way to balance the tradeoff between them. The other is to protect weight or gradient updates by utilizing perturbation techniques (Section 3.2). As an advantage, such techniques can protect the intermediate updates and achieve privacy preservation. However, because perturbation techniques usually need to add noise to the weights or gradients, the adoption of a perturbation will simultaneously degrade the model accuracy and cause computational overhead. It is therefore necessary to find a way to provide a good balance between these two conflicting performances.

- Studies [75, 160, 228] on inference attacks have shown that sensitive information can be extracted from the final model or an inference API if the final model or the query results are not protected properly. It is necessary to develop efficient solutions to defend the final model against such attacks. Here, we come up with two possible directions: (1) utilizing the encryption or perturbation techniques introduced in Section 3.1 and 3.2 to protect the final model against external attackers and (2) using the splitting techniques described in Section 2.3.3 to personalize the model for each participant by splitting the global model, for instance, allowing the participants to privately hold some special layers for their individual goals [48].
- Data memorization should be efficiently handled in the PPFL to prevent a privacy leakage. This is mainly because neural network models might unintentionally memorize sensitive information of the training data [17]. In general, there are two ways to deal with this concern: (1) anonymizing the training datasets or (2) anonymizing the training process. There have been a few studies that have recently focused on this research topic [32, 56, 106, 165, 197, 222]. Concerning the anonymization of the training datasets, it is clear that all privacy leakage risks originate from the sensitive information of the training datasets. Therefore, we believe that a method for anonymizing the training datasets while retaining their utility is one fundamental direction for privacy preservation in PPFL. The anonymization techniques introduced in Section 3.3 provide possible guides for researchers. For anonymization during the training process, researchers [32, 222] have proposed some solutions. The key idea is how to effectively anonymize the sensitive information of the participants during the training process and simultaneously guarantee model accuracy.
- Privacy-preserving mechanisms may differ in terms of the effectiveness and computation cost when applied in FL for privacy defense. It is necessary to study how to optimize the deployment of defense mechanisms or measurements. The study in Reference [181] presents diverse metrics for measuring the data utility and data privacy, and is a useful guide to conduct a comprehensive investigation on this research topic. In addition, most research studies have focused on PPFL frameworks with a central server. It is necessary to investigate whether existing privacy attacks against these frameworks are still effective against the FL frameworks without a central server.
- We believe it will be promising to develop PPFL frameworks based on hybrid privacy-preserving techniques, for example, combining different encryption techniques or combining encryption techniques with perturbation techniques. Because different privacy-preserving techniques possess dominant advantages in distinct aspects, it is reasonable to combine their advantages to develop effective PPFL frameworks [59, 70, 224]. For example, anonymization techniques are usually simple and computationally efficient but tend to degrade the accuracy, whereas encryption techniques tend to be accurate but computationally expensive. Thus, the correct way to develop PPFL might be to combine several types of techniques or combine different techniques within the same category.

REFERENCES

[1] M. Abadi, A. Chu, I. Goodfellow, H. B. McMahan, I. Mironov, K. Talwar, and L. Zhang. 2016. Deep learning with differential privacy. In *Proceedings of the ACM SIGSAC Conference on Computer and Communications Security*. 308–318.

[2] N. Abbas, Y. Zhang, A. Taherkordi, and T. Skeie. 2018. Mobile edge computing: A survey. *IEEE IoT J.* 5, 1 (2018), 450–465.

[3] N. Agarwal, A. T. Suresh, F. X. Yu, S. Kumar, and B. McMahan. 2018. cpSGD: Communication-efficient and differentially-private distributed SGD. In *Advances in Neural Information Processing Systems*, Vol. 31. 7564–7575.

[4] D. Agrawal and C. Aggarwal. 2001. On the design and quantification of privacy preserving data mining algorithms. In *Proceedings of the ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems*. 247–255.

[5] A. Ahmed and E. Ahmed. 2016. A survey on mobile edge computing. In *Proceedings of the International Conference on Intelligent Systems and Control*. 1–8.

[6] K. Amin, A. Kulesza, A. Munoz, and S. Vassilvtiskii. 2019. Bounding user contributions: A bias-variance trade-off in differential privacy. In *Proceedings of the International Conference on Machine Learning*, Vol. 97. 263–271.

[7] Muhammad Asad, Ahmed Moustafa, and Takayuki Ito. 2020. FedOpt: Towards communication efficiency and privacy preservation in federated learning. *Appl. Sci.* 10, 8 (2020), 2864.

[8] G. Ateniese, L. Mancini, A. Spognardi, et al. 2015. Hacking smart machines with smarter ones: How to extract meaningful data from machine learning classifiers. *Int. J. Secur. Netw.* 10, 3 (2015), 137–150.

[9] A. N. Bhagoji, S. Chakraborty, P. Mittal, and S. Calo. 2019. Analyzing federated learning through an adversarial lens. In *Proceedings of the International Conference on Machine Learning*, Vol. 97. 634–643.

[10] A. Bhowmick, J. Duchi, J. Freudiger, G. Kapoor, and R. Rogers. 2019. Protection against reconstruction and its applications in private federated learning. arXiv:1812.00984. Retrieved from https://arxiv.org/abs/1812.00984.

[11] S. Bickel, M. Brückner, and T. Scheffer. 2007. Discriminative learning for differing training and test distributions. In *Proceedings of the International Conference on Machine Learning*. 81–88.

[12] G. R. Blakley. 1979. Safeguarding cryptographic keys. In *Proceedings of the International Workshop on Managing Requirements Knowledge*. 313–318.

[13] K. Bonawitz, V. Ivanov, B. Kreuter, et al. 2017. Practical secure aggregation for privacy-preserving machine learning. In *Proceedings of the ACM SIGSAC Conference on Computer and Communications Security*. 1175–1191.

[14] Z. Brakerski, C. Gentry, and V. Vaikuntanathan. 2014. (Leveled) fully homomorphic encryption without bootstrapping. *ACM Trans. Comput. Theory* 6, 3 (2014), 1–36.

[15] Alon Brutzkus, Ran Gilad-Bachrach, and Oren Elisha. 2019. Low latency privacy preserving inference. In *Proceedings of the International Conference on Machine Learning*, Vol. 97. 812–821.

[16] H. Cao, S. Liu, R. Zhao, and X. Xiong. 2020. IFed: A novel federated learning framework for local differential privacy in power internet of things. *Int. J. Distrib. Sens. Netw.* 16, 5 (2020), 1550147720919698.

[17] N. Carlini, C. Liu, Ú. Erlingsson, J. Kos, and D. Song. 2019. The secret sharer: Evaluating and testing unintended memorization in neural networks. In *Proceedings of the USENIX Security Symposium*. 267–284.

[18] H. Chabanne, A. De Wargny, J. Milgram, C. Morel, and E. Prouff. 2017. Privacy-preserving classification on deep neural network. *IACR Cryptol. ePrint Arch.* 2017, 35 (2017).

[19] M. Chamikara, P. Bertok, I. Khalil, D. Liu, and S. Camtepe. 2021. Privacy preserving distributed machine learning with federated learning. *Computer Communications* 171, 1 (2021), 112–125.

[20] H. Chang, V. Shejwalkar, R. Shokri, and A. Houmansadr. 2019. Cronus: Robust and heterogeneous collaborative learning with black-box knowledge transfer. arXiv:1912.11279. Retrieved from https://arxiv.org/abs/1912.11279.

[21] K. Chaudhuri and C. Monteleoni. 2009. Privacy-preserving logistic regression. In *Advances in Neural Information Processing Systems*, Vol. 22. 289–296.

[22] K. Chaudhuri, A. Sarwate, and K. Sinha. 2012. Near-optimal differentially private principal components. In *Advances in Neural Information Processing Systems*, Vol. 25. 989–997.

[23] S. Chawla, C. Dwork, F. McSherry, A. Smith, and H. Wee. 2005. Toward privacy in public databases. In *Theory of Cryptography*. 363–385.

[24] Hong-You Chen and Wei-Lun Chao. 2021. FedBE: Making bayesian model ensemble applicable to federated learning. In *Proceedings of the International Conference on Learning Representations*.

[25] K. Chen and L. Liu. 2008. *A Survey of Multiplicative Perturbation for Privacy-Preserving Data Mining*. Springer, 157–181.

[26] Y. Chen, X. Qin, J. Wang, C. Yu, and W. Gao. 2020. Fedhealth: A Federated Transfer Learning Framework for Wearable Healthcare. *IEEE Intell. Syst.* 4 (2020).

[27] K. Cheng, T. Fan, Y. Jin, et al. 2021. Secureboost: A lossless federated learning framework. *IEEE Intelligent Systems* (2021). DOI: 10.1109/MIS.2021.3082561

[28] W. Chik. 2013. The singapore personal data protection act and an assessment of future trends in data privacy reform. *Comput. Law Secur. Rev.* 29 (2013), 554–575.

[29] Beongjun Choi, Jy yong Sohn, Dong-Jun Han, and Jaekyun Moon. 2020. Communication-computation efficient secure aggregation for federated learning. arXiv:2012.05433. Retrieved from https://arxiv.org/abs/2012.05433.

[30] C. Choquette-Choo, N. Dullerud, A. Dziedzic, et al. 2021. CaPC learning: Confidential and private collaborative learning. In *Proceedings of the International Conference on Learning Representations*.

[31] Olivia Choudhury, Aris Gkoulalas-Divanis, Theodoros Salonidis, et al. 2019. Differential privacy-enabled federated learning for sensitive health data. In *Proceedings of the NeurIPS Workshop on Machine Learning for Health*.

[32] O. Choudhury, A. Gkoulalas-Divanis, T. Salonidis, I. Sylla, Y. Park, G. Hsu, and A. Das. 2020. A syntactic approach for privacy-preserving federated learning. In *Proceedings of the European Conference on Artificial Intelligence*.

[33] V. Ciriani, S. Di Vimercati, S. Foresti, and P. Samarati. 2008. *K-Anonymous Data Mining: A Survey*. Springer, 105–136.

[34] G. Cormode, S. Jha, T. Kulkarni, N. Li, D. Srivastava, and T. Wang. 2018. Privacy at Scale: Local differential privacy in practice. In *Proceedings of the International Conference on Management of Data*. 1655–1658.

[35] W. Dai, Q. Yang, G. Xue, and Y. Yu. 2007. Boosting for transfer learning. In *Proceedings of the International Conference on Machine Learning*. 193–200.

[36] J. Devlin, M. Chang, K. Lee, and K. Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. arXiv:1810.04805. Retrieved from https://arxiv.org/abs/1810.04805.

[37] W. Diffie and M. Hellman. 1976. New directions in cryptography. *IEEE Trans. Inf. Theory* 22, 6 (1976), 644–654.

[38] Y. Dong, X. Chen, L. Shen, and D. Wang. 2019. Privacy-preserving distributed machine learning based on secret sharing. In *Proceedings of the International Conference on Information and Communications Security*. 684–702.

[39] Ye Dong, Xiaojun Chen, Liyan Shen, and Dakui Wang. 2020. EaSTFLy: Efficient and secure ternary federated learning. *Comput. Secur.* 94, 1 (2020), 101824.

[40] Abhimanyu Dubey and Alex Pentland. 2020. Differentially-private federated linear bandits. In *Advances in Neural Information Processing Systems*, Vol. 33. 6003–6014.

[41] C. Dwork. 2011. A firm foundation for private data analysis. *Commun. ACM* 54, 1 (2011), 86–95.

[42] C. Dwork and M. Naor. 2010. On the difficulties of disclosure prevention in statistical databases or the case for differential privacy. *J. Priv. Confident.* 2, 1 (2010).

[43] C. Dwork and A. Roth. 2014. The Algorithmic foundations of differential privacy. *Found. Trends Theor. Comput. Sci.* 9, 3–4 (2014), 211–407.

[44] C. Dwork, G. N. Rothblum, and S. Vadhan. 2010. Boosting and differential privacy. In *Proceedings of the IEEE Annual Symposium on Foundations of Computer Science*. 51–60.

[45] A. Elgabli, J. Park, C. Ben Issaid, and M. Bennis. 2021. Harnessing wireless channels for scalable and privacy-preserving federated learning. *IEEE Transactions on Communications* (2021).

[46] T. ElGamal. 1985. A public key cryptosystem and a signature scheme based on discrete logarithms. *IEEE Trans. Inf. Theory* 31, 4 (1985), 469–472.

[47] D. Enthoven and Z. Al-Ars. 2020. An overview of federated deep learning privacy attacks and eefensive strategies. arXiv:2004.04676. Retrieved from https://arxiv.org/abs/2004.04676.

[48] A. Fallah, A. Mokhtari, and A. Ozdaglar. 2020. Personalized federated learning with theoretical guarantees: A model-agnostic meta-learning approach. In *Advances in Neural Information Processing Systems*, Vol. 33. 3557–3568.

[49] R. Fantacci and B. Picano. 2020. Federated learning framework for mobile edge computing networks. *CAAI Trans. Intell. Technol.* 5, 1 (2020), 15–21.

[50] Aamir Farooq and Mahvish Samar. 2020. Multiplicative perturbation bounds for the block cholesky downdating problem. *Int. J. Comput. Math.* 97, 12 (2020), 2421–2435.

[51] S. Feng and H. Yu. 2020. Multi-participant multi-class vertical federated learning. arXiv:2001.11154. Retrieved from https://arxiv.org/abs/2001.11154.

[52] Y. Feng, X. Yang, W. Fang, S. Xia, and X. Tang. 2020. Practical and bilateral privacy-preserving federated learning. arXiv:2002.09843. Retrieved from https://arxiv.org/abs/2002.09843.

[53] M. Fredrikson, S. Jha, and T. Ristenpart. 2015. Model inversion attacks that exploit confidence information and basic countermeasures. In *Proceedings of the ACM SIGSAC Conference on Computer and Communications Security*. 1322–1333.

[54] M. Fredrikson, E. Lantz, S. Jha, S. Lin, D. Page, and T. Ristenpart. 2014. Privacy in Pharmacogenetics: an end-to-end case study of personalized warfarin dosing. In *Proceedings of the USENIX Security Symposium*. 17–32.

[55] B. C. Fung, K. Wang, R. Chen, and P. Yu. 2010. Privacy-preserving data publishing: a survey of recent developments. *Comput. Surv.* 42, 4 (2010), 1–53.

[56] Clement Fung, Jamie Koerner, Stewart Grant, and Ivan Beschastnikh. 2019. Dancing in the dark: Private multi-party machine learning in an untrusted setting. arXiv:1811.09712. Retrieved from https://arxiv.org/abs/1811.09712.

[57] S. Gade and N. Vaidya. 2018. Privacy-Preserving distributed learning via obfuscated stochastic gradients. In *Proceedings of the IEEE Conference on Decision and Control*. 184–191.

[58] A. Galakatos, A. Crotty, and T. Kraska. 2018. *Distributed Machine Learning*. Springer, New York, 1196–1201.

[59] D. Gao, Y. Liu, A. Huang, C. Ju, H. Yu, and Q. Yang. 2019. Privacy-preserving heterogeneous federated transfer learning. In *Proceedings of the IEEE International Conference on Big Data*. 2552–2559.

[60] J. Gao, W. Fan, J. Jiang, and J. Han. 2008. Knowledge transfer via multiple model local structure mapping. In *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. 283–291.

[61] Jonas Geiping, Hartmut Bauermeister, Hannah Dröge, and Michael Moeller. 2020. Inverting gradients-how easy is it to break privacy in federated learning. In *Advances in Neural Information Processing Systems*, Vol. 33. 16937–16947.

[62] C. Gentry, A. Sahai, and B. Waters. 2013. Homomorphic encryption from learning with errors: Conceptually-simpler, asymptotically-faster, attribute-based. In *Proceedings of the Annual Cryptology Conference*. 75–92.

[63] Robin C. Geyer, Tassilo Klein, and Moin Nabi. 2018. Differentially private federated learning: A client level perspective. arXiv:1712.07557. Retrieved from https://arxiv.org/abs/1712.07557.

[64] R. Gilad-Bachrach, N. Dowlin, K. Laine, et al. 2016. CryptoNets: Applying neural networks to encrypted data with high throughput and accuracy. In *Proceedings of the International Conference on Machine Learning*. 201–210.

[65] O. Goldreich, S. Micali, and A. Wigderson. 1987. How to play any mental game. In *Proceedings of the ACM Symposium on Theory of Computing*. 218–229.

[66] M. Gong, Y. Xie, K. Pan, K. Feng, and A. K. Qin. 2020. A survey on differentially private machine learning. *IEEE Comput. Intell. Mag.* 15, 2 (2020), 49–64.

[67] O. Gupta and R. Raskar. 2018. Distributed learning of deep neural network over multiple agents. *J. Netw. Comput. Appl.* 116, 1 (2018), 1–8.

[68] Jenny Hamer, Mehryar Mohri, and Ananda Theertha Suresh. 2020. FedBoost: A communication-efficient algorithm for federated learning. In *Proceedings of the International Conference on Machine Learning*, Vol. 119. 3973–3983.

[69] M. Hao, H. Li, X. Luo, G. Xu, H. Yang, and S. Liu. 2020. Efficient and privacy-enhanced federated learning for industrial artificial intelligence. *IEEE Trans. Industr. Inf.* 16, 10 (2020), 6532–6542.

[70] M. Hao, H. Li, G. Xu, S. Liu, and H. Yang. 2019. Towards efficient and privacy-preserving federated deep learning. In *Proceedings of the IEEE International Conference on Communications*. 1–6.

[71] S. Hardy, W. Henecka, H. Ivey-Law, R. Nock, G. Patrini, G. Smith, and B. Thorne. 2017. Private federated learning on vertically partitioned data via entity resolution and additively homomorphic encryption. arXiv:1711.10677. Retrieved from https://arxiv.org/abs/1711.10677.

[72] Lein Harn and Changlu Lin. 2010. Strong (n, t, n) verifiable secret sharing scheme. *Inf. Sci.* 180, 16 (2010), 3059–3064.

[73] J. Hayes, L. Melis, G. Danezis, and E. De Cristofaro. 2019. LOGAN: Membership inference attacks against generative models. In *Proceedings of the Conference on Privacy Enhancing Technologies*. 133–152.

[74] Chaoyang He, Murali Annavaram, and Salman Avestimehr. 2020. Group knowledge transfer: Federated learning of large cnns at the edge. In *Advances in Neural Information Processing Systems*, Vol. 33. 14068–14080.

[75] B. Hitaj, G. Ateniese, and F. Perez-Cruz. 2017. Deep Models under the GAN: Information leakage from collaborative deep learning. In *Proceedings of the ACM SIGSAC Conference on Computer and Communications Security*. 603–618.

[76] Q. Ho, J. Cipar, H. Cui, et al. 2013. More effective distributed ml via a stale synchronous parallel parameter server. In *Advances in Neural Information Processing Systems*, Vol. 26. 1223–1231.

[77] J. Hu and A. V. Vasilakos. 2016. Energy big data analytics and security: Challenges and opportunities. *IEEE Trans. Smart Grid* 7, 5 (2016), 2423–2436.

[78] R. Hu, Y. Guo, H. Li, Q. Pei, and Y. Gong. 2020. Personalized federated learning with differential privacy. *IEEE IoT J.* 10 (2020), 9530–9539.

[79] S. Janbaz, R. Asghari, B. Bagherpour, and A. Zaghian. 2020. A fast non-interactive publicly verifiable secret sharing scheme. In *Proceedings of the International ISC Conference on Information Security and Cryptology*. 7–13.

[80] B. Jayaraman, L. Wang, D. Evans, and Q. Gu. 2018. Distributed learning without distress: Privacy-preserving empirical risk minimization. In *Advances in Neural Information Processing Systems*, Vol. 32. 6346–6357.

[81] L. Jiang, R. Tan, X. Lou, and G. Lin. 2019. On lightweight privacy-preserving collaborative learning for internet-of-things objects. In *Proceedings of the International Conference on Internet of Things Design and Implementation*. 70–81.

[82] P. Kairouz, H. B. McMahan, B. Avent, et al. 2019. Advances and open problems in federated learning. *Foundations and Trends in Machine Learning* 14, 1 (2021).

[83] P. Kairouz, S. Oh, and P. Viswanath. 2017. The composition theorem for differential privacy. *IEEE Trans. Inf. Theory* 63, 6 (2017), 4037–4049.

[84] G. A. Kaissis, M. R. Makowski, D. Rückert, and R. F. Braren. 2020. Secure, privacy-preserving and federated machine learning in medical imaging. *Nat. Mach. Intell.* 2, 6 (2020), 305–311.

[85] M. Kapralov and K. Talwar. 2013. On differentially private low rank approximation. In *Proceedings of the ACM-SIAM Symposium on Discrete Algorithms*. 1395–1414.

[86] H. Kargupta, S. Datta, Q. Wang, and K. Sivakumar. 2003. On the privacy preserving properties of random data perturbation techniques. In *Proceedings of the IEEE International Conference on Data Mining*. 99–106.

[87] S. P. Karimireddy, S. Kale, M. Mohri, S. Reddi, S. Stich, and A. T. Suresh. 2020. SCAFFOLD: Stochastic controlled averaging for federated learning. In *Proceedings of the International Conference on Machine Learning*, Vol. 119. 5132–5143.

[88] A. Kawachi, K. Tanaka, and K. Xagawa. 2007. Multi-bit cryptosystems based on lattice problems. In *Proceedings of the International Workshop on Public Key Cryptography*. 315–329.

[89] A. Lalitha, O. C. Kilinc, T. Javidi, and F. Koushanfar. 2019. Peer-to-Peer federated learning on graphs. arXiv:1901.11173. Retrieved from https://arxiv.org/abs/1901.11173.

[90] S. Lee, V. Chatalbashev, D. Vickrey, and D. Koller. 2007. Learning a meta-level prior for feature relevance from multiple related tasks. In *Proceedings of the International Conference on Machine Learning*. 489–496.

[91] H. Li and T. Han. 2019. An end-to-end encrypted neural network for gradient updates transmission in federated learning. In *Proceedings of the Data Compression Conference*. 589–589.

[92] H. Li, D. Liu, Y. Dai, T. Luan, and X. Shen. 2014. Enabling efficient multi-keyword ranked search over encrypted mobile cloud data through blind storage. *IEEE Trans. Emerg. Top. Comput.* 3, 1 (2014), 127–138.

[93] J. Li. 2018. Cyber security meets artificial intelligence: A survey. *Front. Inf. Technol. Electr. Eng.* 19, 12 (2018), 1462–1474.

[94] J. Li, M. Khodak, S. Caldas, and A. Talwalkar. 2019. Differentially private meta-learning. In *Proceedings of the International Conference on Learning Representations*.

[95] N. Li, T. Li, and S. Venkatasubramanian. 2007. t-closeness: Privacy Beyond k-anonymity and l-diversity. In *Proceedings of the IEEE International Conference on Data Engineering*. 106–115.

[96] Q. Li, Z. Wen, and B. He. 2020. Practical federated gradient boosting decision trees. In *Proceedings of the AAAI Conference on Artificial Intelligence*. 4642–4649.

[97] Q. Li, Z. Wen, Z. Wu, S. Hu, N. Wang, and B. He. 2021. A survey on federated learning systems: Vision, hype and reality for data privacy and protection. arXiv:1907.09693. Retrieved from https://arxiv.org/abs/1907.09693.

[98] R. Li, Y. Xiao, C. Zhang, T. Song, and C. Hu. 2018. Cryptographic algorithms for privacy-preserving online applications. *Math. Found. Comput.* 1, 4 (2018), 311.

[99] T. Li, A. K. Sahu, A. Talwalkar, and V. Smith. 2020. Federated Learning: Challenges, methods, and future directions. *IEEE Sign. Process. Mag.* 37, 3 (2020), 50–60.

[100] Xiaoxiao Li, Meirui Jiang, Xiaofei Zhang, Michael Kamp, and Qi Dou. 2021. FedBN: Federated learning on non-iid features via local batch normalization. In *Proceedings of the International Conference on Learning Representations*.

[101] Z. Li, V. Sharma, and S. P. Mohanty. 2020. Preserving data privacy via federated learning: Challenges and solutions. *IEEE Cons. Electr. Mag.* 9, 6 (2020), 8–16.

[102] Z. Li, T. Wang, M. Lopuhaä-Zwakenberg, N. Li, and B. Škoric. 2020. Estimating numerical distributions under local differential privacy. In *Proceedings of the ACM SIGMOD International Conference on Management of Data*. 621–635.

[103] Z. Li and Y. Zhang. 2021. Label-leaks: Membership inference attack with label. arXiv:2007.15528. Retrieved from https://arxiv.org/abs/2007.15528.

[104] G. Liang and S. Chawathe. 2004. Privacy-preserving inter-database operations. In *Proceedings of the International Conference on Intelligence and Security Informatics*. 66–82.

[105] W. Y. B. Lim, N. C. Luong, D. T. Hoang, Y. Jiao, Y. C. Liang, Q. Yang, D. Niyato, and C. Miao. 2020. Federated learning in mobile edge networks: A comprehensive survey. *IEEE Commun. Surv. Tutor.* 3 (2020), 2031–2063.

[106] Tao Lin, Lingjing Kong, Sebastian U. Stich, and Martin Jaggi. 2020. Ensemble distillation for robust model fusion in federated learning. In *Advances in Neural Information Processing Systems*, Vol. 33. 2351–2363.

[107] D. Liu, T. Miller, R. Sayeed, and K. Mandl. 2018. FADL: Federated-autonomous deep learning for distributed electronic health record. arXiv:1811.11400. Retrieved from https://arxiv.org/abs/1811.11400.

[108] Na Liu, Wei Luo, and Qingxiang Xu. 2018. New multiplicative perturbation bounds for the generalized polar decomposition. *Appl. Math. Comput.* 339, C (2018), 259–271.

[109] R. Liu, Y. Cao, M. Yoshikawa, and H. Chen. 2020. FedSel: Federated SGD under Local Differential privacy with top-k dimension selection. In *Proceedings of the International Conference on Database Systems for Advanced Applications*.

[110] X. Liu, H. Li, G. Xu, R. Lu, and M. He. 2020. Adaptive privacy-preserving federated learning. *Peer-to-Peer Netw. Appl.* 6 (2020), 2356–2366.

[111] Y. Liu, Y. Kang, C. P. Xing, T. J. Chen, and Q. Yang. 2020. A secure federated transfer learning framework. *IEEE Intell. Syst.* 35, 4 (2020), 70–82.

[112] Y. Liu, Y. Kang, X. Zhang, et al. 2019. A communication efficient vertical federated learning framework. arXiv:1912.11187. Retrieved from https://arxiv.org/abs/1912.11187.

[113] Y. Liu, Z. Ma, Z. Yan, Z. Wang, X. Liu, and J. Ma. 2020. Privacy-preserving federated k-means for proactive caching in next generation cellular networks. *Inf. Sci.* 521, C (2020), 14–31.

[114] H. Lu, C. Liu, T. He, S. Wang, and K. Chan. 2020. Sharing models or coresets: A study based on membership inference attack. In *Proceedings of the International Workshop on Federated Learning for User Privacy and Data Confidentiality.*

[115] S. Lu, Y. Zhang, and Y. Wang. 2020. Decentralized federated learning for electronic health records. In *Proceedings of the Annual Conference on Information Sciences and Systems.* 1–5.

[116] Y. Lu, X. Huang, Y. Dai, S. Maharjan, and Y. Zhang. 2019. Differentially private asynchronous federated learning for mobile edge computing in urban informatics. *IEEE Trans. Industr. Inf.* 16, 3 (2019), 2134–2143.

[117] L. Lyu, H. Yu, and Q. Yang. 2020. Threats to federated learning: A survey. arXiv:2003.02133. Retrieved from https://arxiv.org/abs/2003.02133.

[118] A. Machanavajjhala, D. Kifer, J. Gehrke, and M. Venkitasubramaniam. 2007. l-Diversity: Privacy beyond k-anonymity. *ACM Trans. Knowl. Discov. Data* 1, 1 (2007), Article 3.

[119] M. S. Mahmud, J. Z. Huang, S. Salloum, T. Z. Emara, and K. Sadatdiynov. 2020. A survey of data partitioning and sampling methods to support big data analysis. *Big Data Min. Analyt.* 3, 2 (2020), 85–101.

[120] G. Malinovskiy, D. Kovalev, E. Gasanov, L. Condat, and P. Richtarik. 2020. From local sgd to local fixed-point methods for federated learning. In *Proceedings of the International Conference on Machine Learning*, Vol. 119. 6692–6701.

[121] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas. 2017. Communication-efficient learning of deep networks from decentralized data. In *Artificial Intelligence and Statistics.* 1273–1282.

[122] H. B. McMahan, D. Ramage, K. Talwar, and L. Zhang. 2018. Learning differentially private recurrent language models. In *Proceedings of the International Conference on Learning Representations.*

[123] L. Melis, C. Song, E. De Cristofaro, and V. Shmatikov. 2019. Exploiting unintended feature leakage in collaborative learning. In *Proceedings of the IEEE Symposium on Security and Privacy.* 691–706.

[124] P. Mohassel and P. Rindal. 2018. ABY^3: A mixed protocol framework for machine learning. In *Proceedings of the ACM Conference on Computer and Communications Security.* 35–52.

[125] P. Mohassel and Y. Zhang. 2017. SecureML: A system for scalable privacy-preserving machine learning. In *Proceedings of the IEEE Symposium on Security and Privacy.* 19–38.

[126] Vaikkunth Mugunthan, Anton Peraire-Bueno, and Lalana Kagal. 2020. PrivacyFL: A simulator for privacy-preserving and secure federated learning. In *Proceedings of the ACM International Conference on Information & Knowledge Management.* 3085–3092.

[127] Vaikkunth Mugunthan, Antigoni Polychroniadou, David Byrd, and Tucker Hybinette Balch. 2019. Smpai: Secure multi-party computation for federated learning. In *Proceedings of the NeurIPS 2019 Workshop on Robust AI in Financial Services.*

[128] M. Naseri, J. Hayes, and E. De Cristofaro. 2021. Toward robustness and privacy in federated learning: Experimenting with local and central differential privacy. arXiv:2009.03561. Retrieved from https://arxiv.org/abs/2009.03561.

[129] M. Nasr, R. Shokri, et al. 2019. Comprehensive privacy analysis of deep learning: Passive and active white-box inference attacks against centralized and federated learning. In *Proceedings of the IEEE Symposium on Security and Privacy.* 739–753.

[130] M. Nasr, R. Shokri, and A. Houmansadr. 2018. Machine learning with membership privacy using adversarial regularization. In *Proceedings of the ACM SIGSAC Conference on Computer and Communications Security.* 634–646.

[131] Kang Loon Ng, Zichen Chen, Zelei Liu, Han Yu, Yang Liu, and Qiang Yang. 2020. A multi-player game for studying federated learning incentive schemes. In *Proceedings of the International Joint Conference on Artificial Intelligence.* 5279–5281.

[132] S. Niknam, H. Dhillon, and J. Reed. 2020. Federated learning for wireless communications: Motivation, opportunities, and challenges. *IEEE Commun. Mag.* 58, 6 (2020), 46–51.

[133] T. Orekondy, S. Oh, Y. Zhang, et al. 2019. Gradient-leaks: Understanding and controlling deanonymization in federated learning. In *Proceedings of the NeurIPS Workshop on Federated Learning for Data Privacy and Confidentiality.*

[134] P. Paillier. 1999. Public-Key cryptosystems based on composite degree residuosity classes. In *Proceedings of the International Conference on the Theory and Applications of Cryptographic Techniques.* 223–238.

[135] S. J. Pan, I. Tsang, J. Kwok, and Q. Yang. 2010. Domain adaptation via transfer component analysis. *IEEE Trans. Neur. Netw.* 22, 2 (2010), 199–210.

[136] S. J. Pan and Q. A. Yang. 2010. A survey on transfer learning. *IEEE Trans. Knowl. Data Eng.* 22, 10 (2010), 1345–1359.

[137] L. T. Phong, Y. Aono, T. Hayashi, L. H. Wang, and S. Moriai. 2018. Privacy-preserving deep learning via additively homomorphic encryption. *IEEE Trans. Inf. Forens. Secur.* 13, 5 (2018), 1333–1345.

[138] L. T. Phong and T. T. Phuong. 2019. Privacy-preserving deep learning via weight transmission. *IEEE Trans. Inf. Forens. Secur.* 14, 11 (2019), 3003–3015.

[139] Anastasia Pustozerova and Rudolf Mayer. 2020. Information leaks in federated learning. In *Proceedings of the Workshop on Decentralized IoT Systems and Security.*

[140] Y. Qian, L. Hu, J. Chen, X. Guan, M. M. Hassan, and A. Alelaiwi. 2019. Privacy-aware service placement for mobile edge computing via federated learning. *Inf. Sci.* 505, 1 (2019), 562–570.

[141] J. Qiu, Z. Tian, C. Du, Q. Zuo, S. Su, and B. Fang. 2020. A survey on access control in the age of internet of things. *IEEE IoT J.* 7 (2020), 4682–4696.

[142] Y. Qu, L. Gao, T. H. Luan, Y. Xiang, S. Yu, B. Li, and G. Zheng. 2020. Decentralized privacy using blockchain-enabled federated learning in fog computing. *IEEE IoT J.* 7, 6 (2020), 5171–5183.

[143] J. Quionero-Candela, M. Sugiyama, A. Schwaighofer, and N. Lawrence. 2009. *Dataset Shift in Machine Learning*. The MIT Press.

[144] D. Reich, A. Todoki, R. Dowsley, et al. 2019. Privacy-preserving classification of personal text messages with secure multi-party computation. In *Advances in Neural Information Processing Systems*, Vol. 32. 3757–3769.

[145] Amirhossein Reisizadeh, Farzan Farnia, Ramtin Pedarsani, and Ali Jadbabaie. 2020. Robust federated learning: The case of affine distribution shifts. In *Advances in Neural Information Processing Systems*, Vol. 33. 21554–21565.

[146] X. Ren, C. Yu, W. Yu, et al. 2018. LoPub: High-dimensional crowdsourced data publication with local differential privacy. *IEEE Trans. Inf. Forens. Secur.* 13, 9 (2018), 2151–2166.

[147] M. S. Riazi, K. Laine, B. Pelton, and W. Dai. 2020. HEAX: An architecture for computing on encrypted data. In *Proceedings of the International Conference on Architectural Support for Programming Languages and Operating Systems*. 1295–1309.

[148] R. L. Rivest, L. Adleman, and M. L. Dertouzos. 1978. On data banks and privacy homomorphisms. *Found. Sec. Comput.* 11, 4 (1978), 169–179.

[149] N. Rodríguez-Barroso et al. 2020. Federated learning and differential privacy: Software tools analysis, the sherpa.ai fl framework and methodological guidelines for preserving data privacy. *Inf. Fus.* 1 (2020), 270–292.

[150] D. Rothchild, A. Panda, E. Ullah, et al. 2020. FetchSGD: Communication-efficient federated learning with sketching. In *Proceedings of the International Conference on Machine Learning*, Vol. 119. 8253–8265.

[151] A. G. Roy, S. Siddiqui, et al. 2019. Braintorrent: A peer-to-peer environment for decentralized federated learning. arXiv:1905.06731. Retrieved from https://arxiv.org/abs/1905.06731.

[152] P. Samarati and L. Sweeney. 1998. *Protecting Privacy when Disclosing Information: k-Anonymity and its Enforcement through Generalization and Suppression*. Technical Report. SRI International Computer Science Laboratory.

[153] A. Sannai. 2018. Reconstruction of training samples from loss functions. arXiv:1805.07337. Retrieved from https://arxiv.org/abs/1805.07337.

[154] M. Scannapieco, I. Figotin, E. Bertino, and A. K. Elmagarmid. 2007. Privacy preserving schema and data matching. In *Proceedings of the ACM SIGMOD International Conference on Management of Data*. 653–664.

[155] Mohamed Seif, Ravi Tandon, and Ming Li. 2020. Wireless federated learning with local differential privacy. In *Proceedings of the IEEE International Symposium on Information Theory*. https://doi.org/10.1109/ISIT44484.2020.9174426

[156] A. Shamir. 1979. How to share a secret. *Commun. ACM* 22, 11 (1979), 612–613.

[157] S. Sharma, C. Xing, Y. Liu, and Y. Kang. 2019. Secure and efficient federated transfer learning. In *Proceedings of the IEEE International Conference on Big Data*. 2569–2576.

[158] M. Shen, H. Wang, B. Zhang, et al. 2021. Exploiting unintended property leakage in blockchain-assisted federated learning for intelligent edge computing. *IEEE IoT J.* 8, 4 (2021), 2265–2275.

[159] R. Shirey. 2007. *Internet Security Glossary, Version 2*. Technical Report. RFC 4949, August.

[160] R. Shokri, M. Stronati, C. Song, and V. Shmatikov. 2017. Membership inference attacks against machine learning models. In *Proceedings of the IEEE Symposium on Security and Privacy*. 3–18.

[161] David Silver, Julian Schrittwieser, Karen Simonyan, et al. 2017. Mastering the Game of Go without human knowledge. *Nature* 550, 7676 (2017), 354–359.

[162] A. Singh, P. Vepakomma, O. Gupta, and R. Raskar. 2019. Detailed comparison of communication efficiency of split learning and federated learning. arXiv:1909.09145. Retrieved from https://arxiv.org/abs/1909.09145.

[163] V. Smith, C. K. Chiang, M. Sanjabi, and A. Talwalkar. 2017. Federated multi-task learning. In *Advances in Neural Information Processing Systems*, Vol. 30. 4424–4434.

[164] Jinhyun So, Basak Guler, and A. Salman Avestimehr. 2020. A scalable approach for privacy-preserving collaborative machine learning. In *Advances in Neural Information Processing Systems*, Vol. 33. 8054–8066.

[165] M. Song, Z. Wang, Z. Zhang, Y. Song, Q. Wang, J. Ren, and H. Qi. 2020. Analyzing user-level privacy attack against federated learning. *IEEE J. Select. Areas Commun.* 10 (2020), 2430–2444.

[166] W. Stallings. 2017. *Cryptography and Network Security Principles and Practices* (7th ed.). Pearson Education, Inc.

[167] Lili Su and Jiaming Xu. 2019. Securing distributed gradient descent in high dimensional statistical learning. *ACM Meas. Anal. Comput. Syst.* 3, 1 (2019), Article 12.

[168] L. Sun, J. Qian, X. Chen, and P. Yu. 2020. LDP-FL: Practical private aggregation in federated learning with local differential privacy. arXiv:2007.15789. Retrieved from https://arxiv.org/abs/2007.15789.

[169] T. Szatmari, M. Petersen, M. Korzepa, and T. Giannetsos. 2020. Modelling audiological preferences using federated learning. In *Proceedings of the ACM Conference on User Modeling, Adaptation and Personalization*. 187–190.

[170] H. Tanuwidjaja, R. Choi, and K. Kim. 2019. A survey on deep learning techniques for privacy-preserving. In *Proceedings of the International Conference on Machine Learning for Cyber Security.* 29–46.

[171] H. Tran and J. Hu. 2019. Privacy-preserving big data analytics a comprehensive survey. *J. Parallel Distrib. Comput.* 134, 1 (2019), 207–218.

[172] A. Triastcyn and B. Faltings. 2019. Federated learning with bayesian differential privacy. In *Proceedings of the IEEE International Conference on Big Data.* 2587–2596.

[173] S. Truex, N. Baracaldo, A. Anwar, T. Steinke, H. Ludwig, R. Zhang, and Y. Zhou. 2019. A hybrid approach to privacy-preserving federated learning. In *Proceedings of the ACM Workshop on Artificial Intelligence and Security.* 1–11.

[174] S. Truex, L. Liu, K. Chow, M. Gursoy, and W. Wei. 2020. LDP-Fed: Federated learning with local differential privacy. In *Proceedings of the ACM International Workshop on Edge Systems, Analytics and Networking.* 61–66.

[175] S. Truex, L. Liu, M. E. Gursoy, L. Yu, and W. Wei. 2019. Demystifying Membership Inference Attacks in Machine Learning as a Service. *IEEE Trans. Serv. Comput.* (2019). https://doi.org/10.1109/TSC.2019.2897554

[176] M. Van Dijk, C. Gentry, S. Halevi, and V. Vaikuntanathan. 2010. Fully homomorphic encryption over the integers. In *Proceedings of the Annual International Conference on the Theory and Applications of Cryptographic Techniques.* 24–43.

[177] P. Vepakomma, O. Gupta, T. Swedish, and R. Raskar. 2018. Split learning for health: Distributed deep learning without sharing raw patient data. arXiv:1812.00564. Retrieved from https://arxiv.org/abs/1812.00564.

[178] J. Verbraeken, M. Wolting, J. Katzy, J. Kloppenburg, T. Verbelen, and J. S. Rellermeyer. 2020. A survey on distributed machine learning. *Comput. Surv.* 53, 2 (2020), 1–33.

[179] P. Voigt and A. Von dem Bussche. 2017. *The EU General Data Protection Regulation (GDPR).* Springer International Publishing. https://doi.org/10.1007/978-3-319-57959-7

[180] Isabel Wagner. 2017. Evaluating the strength of genomic privacy metrics. *ACM Trans. Priv. Secur.* 20, 1 (2017), Article 2.

[181] I. Wagner and D. Eckhoff. 2018. Technical privacy metrics: A systematic survey. *ACM Comput. Surv.* 51, 3 (2018), Article 57.

[182] A. Wang, C. Wang, M. Bi, and J. Xu. 2018. A Review of privacy-preserving machine learning classification. In *Cloud Computing and Security.* 671–682.

[183] C. Wang and S. Mahadevan. 2008. manifold alignment using procrustes analysis. In *Proceedings of the International Conference on Machine Learning.* 1120–1127.

[184] G. Wang, C. X. Dang, and Z. Zhou. 2019. Measure contribution of participants in federated learning. In *Proceedings of the IEEE International Conference on Big Data.* 2597–2604.

[185] H. Wang, K. Sreenivasan, S. Rajput, et al. 2020. Attack of the tails: Yes, you really can backdoor federated learning. In *Advances in Neural Information Processing Systems*, Vol. 33. 16070–16084.

[186] J. Wang, Z. Cai, and J. Yu. 2020. Achieving personalized k-anonymity-based content privacy for autonomous vehicles in CPS. *IEEE Trans. Industr. Inf.* 16, 6 (2020), 4242–4251.

[187] Lixu Wang, Shichao Xu, Xiao Wang, and Qi Zhu. 2019. Eavesdrop the composition proportion of training labels in federated learning. arXiv:1910.06044. Retrieved from https://arxiv.org/abs/1910.06044.

[188] Rong Wang, Yan Zhu, Tung-Shou Chen, and Chin-Chen Chang. 2018. Privacy-preserving algorithms for multiple sensitive attributes satisfying t-closeness. *J. Comput. Sci. Technol.* 33, 6 (2018), 1231–1242.

[189] S. Wang, T. Tuor, T. Salonidis, et al. 2019. Adaptive federated learning in resource constrained edge computing systems. *IEEE J. Select. Areas Commun.* 37, 6 (2019), 1205–1221.

[190] X. Wang, Y. Han, C. Wang, Q. Zhao, X. Chen, and M. Chen. 2019. In-Edge AI: Intelligentizing mobile edge computing, caching and communication by federated learning. *IEEE Netw.* 33, 5 (2019), 156–165.

[191] Y. Wang, Y. Tong, and D. Shi. 2020. Federated latent dirichlet allocation: A local differential privacy based framework. In *Proceedings of the AAAI Conference on Artificial Intelligence.* 6283–6290.

[192] Z. Wang, M. Song, Z. Zhang, Y. Song, Q. Wang, and H. Qi. 2019. Beyond inferring class representatives: User-level privacy leakage from federated learning. In *Proceedings of the IEEE Conference on Computer Communications.* 2512–2520.

[193] Kang Wei, Jun Li, Ming Ding, et al. 2020. Federated learning with differential privacy: Algorithms and performance analysis. *IEEE Trans. Inf. Forens. Secur.* 15, 1 (2020), 3454–3469.

[194] W. Wei, L. Liu, M. Loper, K. Chow, M. Gursoy, S. Truex, and Y. Wu. 2020. A framework for evaluating client privacy leakages in federated learning. In *Proceedings of the European Symposium on Research in Computer Security.* 545–566.

[195] M. Wu, D. Ye, J. Ding, et al. 2021. Incentivizing differentially private federated learning: A multi-dimensional contract approach. *IEEE IoT J.* (2021). https://doi.org/10.1109/JIOT.2021.3050163

[196] Chulin Xie, Keli Huang, Pin-Yu Chen, and Bo Li. 2020. DBA: Distributed backdoor attacks against federated learning. In *Proceedings of the International Conference on Learning Representations.*

[197] Cong Xie, Oluwasanmi Koyejo, and Indranil Gupta. 2020. SLSGD: Secure and efficient distributed on-device machine learning. In *Proceedings of the Joint European Conference on Machine Learning and Knowledge Discovery in Databases.* 213–228.

[198] G. Xu, H. Li, S. Liu, K. Yang, and X. Lin. 2020. Verifynet: Secure and verifiable federated learning. *IEEE Trans. Inf. Forens. Secur.* 15, 1 (2020), 911–926.

[199] R. Xu, N. Baracaldo, Y. Zhou, A. Anwar, and H. Ludwig. 2019. Hybridalpha: An efficient approach for privacy-preserving federated learning. In *Proceedings of the ACM Workshop on Artificial Intelligence and Security*. 13–23.

[200] X. Xu, J. Wu, M. Yang, et al. 2020. Information leakage by model weights on federated learning. In *Proceedings of the Workshop on Privacy-Preserving Machine Learning in Practice*. 31–36.

[201] H. Yang, A. Arafa, T. Quek, and H. Poor. 2020. Age-based scheduling policy for federated learning in mobile edge networks. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing*. 8743–8747.

[202] H. Yang, H. He, W. Zhang, and X. Cao. 2020. FedSteg: A federated transfer learning framework for secure image steganalysis. *IEEE Trans. Netw. Sci. Eng.* (2020). https://doi.org/10.1109/TNSE.2020.2996612

[203] K. Yang, T. Fan, T. Chen, Y. Shi, and Q. Yang. 2019. A quasi-Newton method based vertical federated learning framework for logistic regression. arXiv:1912.00513. Retrieved from https://arxiv.org/abs/1912.00513.

[204] Q. Yang, Y. Liu, T. Chen, and Y. Tong. 2019. Federated machine learning: concept and applications. *ACM Trans. Intell. Syst. Technol.* 10, 2 (2019), 1–19.

[205] Q. Yang, Y. Liu, Y. Cheng, Y. Kang, T. Chen, and H. Yu. 2019. Federated learning. *Synth. Lect. Artif. Intell. Mach. Learn.* 13 (2019), 1–207.

[206] S. Yang et al. 2019. Parallel distributed logistic regression for vertical federated learning without third-party coordinator. In *Proceedings of the IJCAI'19 Workshop on Federated Machine Learning for User Privacy and Data Confidentiality.*

[207] A. C. Yao. 1982. Protocols for secure computations. In *Proceedings of the Annual Symposium on Foundations of Computer Science*. 160–164.

[208] X. Yin et al. 2021. 3D fingerprint recognition based on ridge-valley-guided 3D reconstruction and 3D topology polymer feature extraction. *IEEE Trans. Pattern Anal. Mach. Intell.* 43, 3 (2021), 1085–1091.

[209] Tehrim Yoon, Sumin Shin, Sung Ju Hwang, and Eunho Yang. 2021. Fedmix: Approximation of mixup under mean augmented federated learning. In *Proceedings of the International Conference on Learning Representations.*

[210] Chen Yu, Hanlin Tang, Cedric Renggli, Simon Kassing, Ankit Singla, Dan Alistarh, Ce Zhang, and Ji Liu. 2019. Distributed learning over unreliable networks. In *Proceedings of the International Conference on Machine Learning*. 7202–7212.

[211] Felix Yu, Ankit Singh Rawat, Aditya Menon, and Sanjiv Kumar. 2020. Federated learning with only positive labels. In *Proceedings of the International Conference on Machine Learning*. 10946–10956.

[212] H. Yu et al. 2019. Parallel restarted sgd with faster convergence and less communication: demystifying why model averaging works for deep learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 33. 5693–5700.

[213] Honglin Yuan and Tengyu Ma. 2020. Federated accelerated stochastic gradient descent. In *Advances in Neural Information Processing Systems*, Vol. 33. 5332–5344.

[214] V. Zantedeschi, A. Bellet, and M. Tommasi. 2020. Fully Decentralized joint learning of personalized models and collaboration graphs. In *Proceedings of the International Conference on Artificial Intelligence and Statistics*. 864–874.

[215] C. Zhang, S. Li, J. Xia, W. Wang, F. Yan, and Y. Liu. 2020. Batchcrypt: Efficient homomorphic encryption for cross-silo federated learning. In *Proceedings of the USENIX Annual Technical Conference*. 493–506.

[216] Chi Zhang, Yu Liu, Le Wang, Yuehu Liu, Li Li, and Nanning Zheng. 2020. Joint intelligence ranking by federated multiplicative update. *IEEE Intell. Syst.* 35, 4 (2020), 15–24.

[217] D. Zhang, X. Chen, D. Wang, and J. Shi. 2018. A survey on collaborative deep learning and privacy-preserving. In *Proceedings of the IEEE 3rd International Conference on Data Science in Cyberspace*. 652–658.

[218] J. Zhang, B. Chen, S. Yu, and H. Deng. 2019. PEFL: A privacy-enhanced federated learning scheme for big data analytics. In *Proceedings of the IEEE Global Communications Conference*. 1–6.

[219] J. Zhang, Z. Zhang, X. Xiao, Y. Yang, and M. Winslett. 2012. Functional mechanism: Regression analysis under differential privacy. In *Proceedings of the International Conference on Very Large Data Bases*. 1364–1375.

[220] J. Zhang, Y. Zhao, J. Wang, and B. Chen. 2020. Fedmec: Improving efficiency of differentially private federated learning via mobile edge computing. *Mobile Netw. Appl.* 6 (2020), 1–13.

[221] X. Zhang, A. Fu, H. Wang, C. Zhou, and Z. Chen. 2020. A privacy-preserving and verifiable federated learning scheme. In *Proceedings of the IEEE International Conference on Communications*. 1–6.

[222] B. Zhao, K. Fan, K. Yang, Z. Wang, H. Li, and Y. Yang. 2021. Anonymous and privacy-preserving federated learning with industrial big data. *IEEE Trans. Industr. Inf.* (2021). https://doi.org/10.1109/TII.2021.3052183

[223] B. Zhao, K. R. Mopuri, and H. Bilen. 2020. iDLG: Improved deep leakage from gradients. arXiv:2001.02610. Retrieved from https://arxiv.org/abs/2001.02610.

[224] K. Zhao, W. Xi, Z. Wang, R. Wang, Z. Jiang, and J. Zhao. 2020. SMSS: Secure member selection strategy in federated learning. *IEEE Intell. Syst.* 35, 4 (2020), 37–49.

[225] Yang Zhao, Jun Zhao, Mengmeng Yang, et al. 2020. Local differential privacy based federated learning for internet of things. *IEEE IoT J.* 11, 8 (2020), 8836–8853. https://doi.org/10.1109/JIOT.2020.3037194

[226] H. D. Zheng, H. B. Hu, and Z. Y. Han. 2020. Preserving user privacy for machine learning: Local differential privacy or federated machine learning. *IEEE Intell. Syst.* 35, 4 (2020), 5–14.

[227] H. Zhu, Z. Li, M. Cheah, and M. Goh. 2020. Privacy-preserving weighted federated learning within oracle-aided MPC framework. arXiv:2003.07630. Retrieved from https://arxiv.org/abs/2003.07630.

[228] L. Zhu, Z. Liu, and S. Han. 2019. Deep leakage from gradients. In *Advances in Neural Information Processing Systems*, Vol. 32. 14774–14784.

[229] Y. Zhu and E. Meijering. 2020. neural architecture search for microscopy cell segmentation. In *Proceedings of the International Workshop on Machine Learning in Medical Imaging*. 542–551.

Received August 2020; revised March 2021; accepted April 2021