Artificial Intelligence Applications And Innovations Vol. 1 No. 3 (2024)

Article history:
Received 24 February 2024
Revised 21 April 2024
Accepted 30 May 2024
Published online 01 July 2024

PPFL: Privacy-Preserving Techniques in Federated Learning

Khalil. Jahani[1*], Behzad. Moshiri[2], Babak. Hossein Khalaj[3]

[1] Department of Computer science, kish International Campus, University of Tehran, Tehran, Iran
[2] School of ECE, College of Engineering, University of Tehran, Tehran, Iran
[3.] Department of Electrical Engineering, Sharif University of Technology, Tehran, Iran

*** Corresponding author email address**: jahanii@ut.ac.ir

A r t i c l e I n f o

Article type:
Review Article

How to cite this article:
Jahani, K., Moshiri, B., & Khalaj, B. H. (2024). PPFL: Privacy-Preserving Techniques in Federated Learning. *Artificial Intelligence Applications and Innovations*, *1*(3), 49-68.
https://doi.org/10.61838/jaiai.1.3.6

A B S T R A C T

Federated Learning is a distributed machine learning paradigm designed to preserve user privacy on decentralized devices without transferring raw data to a central server. Protecting data privacy in FL involves determining permissible operations and how they can be executed. This review provides an in-depth exploration of privacy threat models within FL, distinguishing between scenarios where the central server is either trusted or untrusted, and identifying appropriate defensive tools and technologies for these settings. The review covers secure computational techniques, including MPC, HE, and TEEs, as well as privacy-preserving mechanisms such as DP, LDP, and DDP models. It also examines hybrid approaches that combine multiple privacy models to enhance efficiency and robustness. The effectiveness of these methods is analysed across different scenarios involving both honest and potentially malicious servers and users. The findings reveal that while privacy-preserving methods mitigate risks, challenges persist in trade off privacy, communication efficiency, and model accuracy. This review highlights open research directions and serves as a comprehensive reference for researchers and practitioners seeking to implement robust privacy measures in federated learning systems.

Keywords: *Federated Learning, Privacy Preservation, Differential Privacy, Homomorphic Encryption, Multi-Party Computation, Trusted Execution Environments*.

1. Introduction

The rapid expansion of data-driven technologies has placed immense emphasis on machine learning (ML) algorithms, particularly in sectors such as healthcare, finance, and industrial applications. Federated Learning (FL), an emerging paradigm, has gained significant attention as it enables decentralized model training by aggregating local updates from devices or nodes, without transferring the data itself. This process reduces communication overhead and mitigates privacy concerns, making it particularly valuable in industries that handle sensitive information. However, despite its potential, privacy preservation in Federated Learning remains a

fundamental challenge, especially as the scale and complexity of data continue to increase.

Privacy preservation in FL is critical due to the inherent risks associated with sharing data across decentralized networks. In many applications, such as healthcare or aviation, data may be subject to strict privacy regulations, and mishandling can lead to severe ethical and legal implications. The decentralized nature of FL makes it an attractive solution, as data is never shared in raw form, thus offering a certain level of confidentiality. Nonetheless, securing the privacy of the data while maintaining the performance of the model is a delicate balance that requires innovative techniques.

This paper offers a comprehensive review of recent advancements in Federated Learning, focusing specifically on privacy preservation methods and their application in various domains. Unlike previous reviews, which may have provided a general overview of FL or focused on specific industries, our review delves into the nuanced challenges and solutions related to privacy in FL. We also highlight emerging research opportunities, particularly in the context of integrating advanced cryptographic techniques, differential privacy, and secure multi-party computation into FL frameworks. Furthermore, we explore the potential of hybrid models that combine Federated Learning with edge computing, offering new avenues for data privacy while optimizing computational resources.

The main contributions of this paper are as follows:

1. A detailed survey of privacy-preserving techniques in Federated Learning, with an emphasis on their application in real-world scenarios.
2. A discussion on the existing gaps in FL research, particularly around scalability, heterogeneity, and security.
3. A roadmap for future research that suggests potential solutions for overcoming these challenges and advancing the practical implementation of FL in privacy-sensitive industries.

By offering a novel perspective on Federated Learning with an exclusive focus on privacy preservation, this review aims to provide both academics and practitioners with a deeper understanding of the state-of-the-art techniques and identify promising research directions for the future.

As illustrated in Figure 1: Horizontal Federated Learning, Vertical Federated Learning, and Federated Transfer Learning, each designed to handle different data configurations across collaborating clients [1].

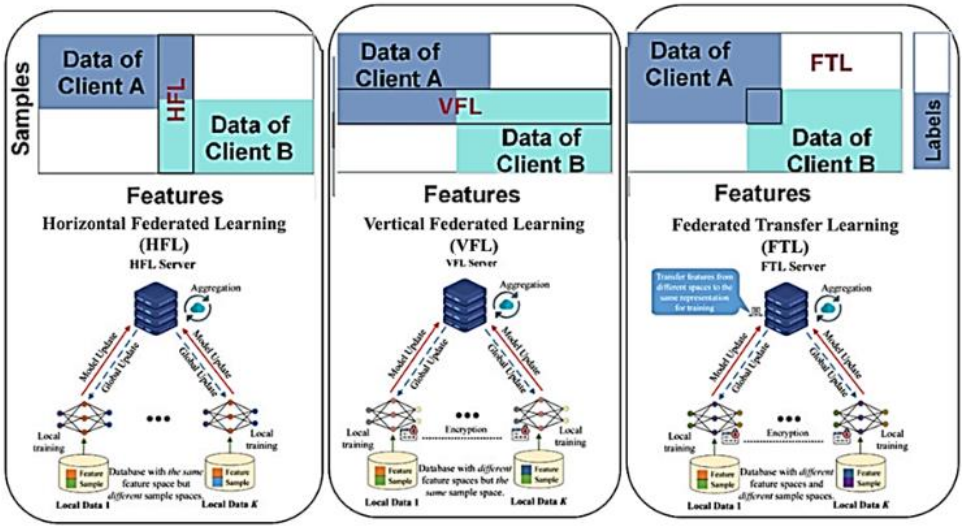

Figure 1. Three distinct types of FL approaches **[1]**

1.1. Horizontal Federated Learning (HFL) (left panel):

HFL is applicable when multiple clients (e.g., Client A and Client B) share the same feature space but have different sample spaces. In other words, each client has data on different individuals but records similar attributes for each individual. Local training occurs on each client's data, and only the model updates (e.g., gradients) are sent to a central server for aggregation. This setup allows for model improvement across distributed datasets without

requiring the raw data to leave each client's device, thus preserving data privacy.

1.2. Vertical Federated Learning (VFL) (middle panel):

VFL is suitable when clients (e.g., Client A and Client B) have different feature sets on the same sample group, meaning they have different information on the same individuals. This scenario typically occurs between organizations holding complementary data about the same set of users. Since each client has distinct features, the data cannot be directly combined for model training. Instead, VFL involves encrypting model updates and exchanging only encrypted gradients and intermediate outputs with the central server, allowing secure aggregation and collaborative learning.

1.3. Federated Transfer Learning (FTL) (right panel):

FTL addresses situations where both the feature and sample spaces differ across clients, but some form of knowledge transfer can still be beneficial. This approach is particularly useful when one client has a limited amount of labeled data that could benefit from the knowledge in other clients' datasets. By transferring features from one domain to another and then aggregating model updates on a central server, FTL supports learning across different feature distributions, enabling effective model training despite limited data availability.

Each method is visualized with an indication of how data is distributed between Client A and Client B and how model updates are aggregated at a central server, showcasing the adaptability of federated learning to different data-sharing scenarios.

Beyond privacy concerns, FL systems also face other security threats. For instance, attackers may attempt to disrupt the training process, prevent effective learning, or manipulate the model to produce outcomes that serve their interests. Addressing these challenges requires a comprehensive understanding of potential attack vectors and the development of defensive strategies to mitigate these risks.

The aim of this review is to provide a comprehensive overview of privacy-preserving techniques in FL. It examines various threat models associated with the deployment of FL, distinguishing between scenarios involving trusted and untrusted servers. Additionally, this review explores key tools and technologies, such as secure computational methods, differential privacy, and hybrid approaches that integrate multiple privacy mechanisms. By analyzing the strengths and limitations of existing methods, it highlights current challenges and identifies promising research directions. This review equips researchers and practitioners with a comprehensive understanding of existing privacy strategies in FL. It also highlights future advancements necessary to address emerging threats effectively.

The subsequent sections of this paper systematically address these concerns. Section 2 explores different threat models relevant to FL, with a focus on the development of defensive mechanisms. Section 3 introduces key tools and technologies that can be employed to establish robust privacy protections against the identified threats. Section 4 addresses challenges under the assumption of a secure server, examining issues related to malicious users and adversarial analysts, and discussing scenarios in which server security is compromised. Finally, Section 5 outlines future directions and presents open questions that require further exploration.

2. Threat Models

To comprehensively understand privacy risks in FL, it is essential to analyze the threat models arising from the various actors involved in the FL ecosystem. This methodology outlines the approach adopted for classifying and examining these privacy threats, with a focus on their potential impact and attack mechanisms across different scenarios.

Privacy is not a binary attribute or a simple scalar variable; rather, it is a multifaceted concept encompassing multiple dimensions. A comprehensive understanding of privacy necessitates a precise identification of the various stakeholders and their respective roles in shaping relevant threat models. Therefore, it is crucial to distinguish between the perspectives of server administrators and analysts who utilize trained models. A system engineered to provide robust privacy protections against malicious analysts may prove inadequate in addressing threats posed by malicious servers. These distinctions align with the threat models delineated in the existing literature. For instance, in the work of [2], the term "encoder" corresponds to users, "shuffler" refers to servers, and "analyzer" denotes analysts or servers engaged in data analysis post-processing.

The design and features of an FL infrastructure—such as network topology, types of algorithms, and the frequency of model updates—significantly impact data minimization and privacy assurance. If users are unaware of the technical details and limitations, they may develop a false sense of security regarding their privacy. Therefore, providing clear and transparent information, along with appropriate education, is crucial to help users understand how their data is utilized.

Differences in users' perceptions of privacy and the varying importance they place on data protection can inform the development of better privacy-preserving mechanisms. This may involve creating methods that adapt federated learning parameters based on individual privacy preferences. Determining who should be responsible for setting these parameters—whether service providers, users, platforms, or policymakers—remains an important topic that requires ongoing discussion.

Moreover, mechanisms such as "privacy-preserving safeguards" [3] can offer privacy guarantees to most users while allowing targeted monitoring to ensure both data confidentiality and privacy goals. These solutions would enable users to select their desired level of privacy, potentially enhancing their sense of security. While the issues at hand are complex, addressing user needs and perceptions is essential for progress in this field.

2.1. Federated Learning's Privacy

The primary objective of FL is to enable analysts or engineers to perform computations on distributed datasets, often involving the training of machine learning models or simpler statistical evaluations. FL improves privacy by keeping raw user data on devices and only sharing model updates, but it doesn't fully guarantee privacy, as model updates can still potentially reveal information. Three key aspects of privacy must be considered:

Computational Privacy: Ensuring that the process of computing functions and any intermediate results do not expose users to potential attacks by malicious servers. Secure computation methods, such as secure multi-party computation (MPC) and trusted execution environments (TEEs), are essential to protect data during the computation process.

Data Privacy: Assessing the extent of information about participating users that might be exposed through computation results. Techniques like differential privacy (DP) are employed to limit the exposure of individual data.

Verifiability: Ensuring that users and servers can verify the integrity of computations without disclosing private data. Techniques such as zero-knowledge proofs (ZKPs) and remote attestation can be used for this purpose [4].

2.2. Definition of Scope and Assumptions

Addressing privacy risks in FL requires a precise understanding of the threat landscape. Privacy is a multi-dimensional concept, influenced by various adversarial capabilities. For this review, each threat model was examined based on the following:

Actor Role: Distinguishing between the roles of the server, users, and external parties, including both colluding and non-colluding entities.

Passive Malicious Server or Client (Honest-But-Curious or Semi-Honest)

Active Malicious Server or Client (Malicious while the Context is Clear)

Trust Level: Analysing scenarios with configurations ranging from fully trusted to partially trusted and untrusted servers to understand the limitations and security requirements of each setup.

2.3. Comparison of Privacy-Preserving Techniques in FL

This section compares several widely used privacy-preserving techniques in the context of FL. The comparison that shows in Table 1, addresses key factors such as threat models, computational complexity, scalability, the impact on model accuracy, and the trade-offs between privacy and performance. The goal is to provide a clear and concise overview of each technique's strengths and weaknesses, enabling a deeper understanding of their applicability in sensitive domains like healthcare, aviation, and finance.

Table 1. Comparison Privacy-Preserving Techniques

Privacy-Preserving Technique	Effectiveness in Threat Models	Computational Complexity	Scalability	Impact on Model Accuracy	Privacy Trade-Offs
Differential Privacy (DP)	Effective against untrusted servers.	Moderate to high. Noise injection requires	Medium scalability. Performance degrades	Significant impact. High noise injection reduces	Strong privacy protection but reduces

	Provides strong privacy guarantees even if the server is adversarial.	significant computation, especially in large-scale models.	with large datasets or high-dimensional data.	model accuracy, especially in sensitive applications.	model performance due to added noise.
Secure Aggregation (SA)	Effective against untrusted servers. Works well with trusted clients.	Low to moderate. Computational complexity primarily depends on the aggregation process and communication overhead.	High scalability. Suitable for large-scale FL setups with many clients.	Minimal impact. Secure aggregation mainly affects communication, leaving model accuracy largely intact.	Balanced trade-off: security with minimal loss of accuracy, but may not provide strong privacy guarantees in all cases.
Homomorphic Encryption (HE)	Effective against both trusted and untrusted servers, ensuring data remains encrypted during computations.	Very high. Homomorphic encryption operations are computationally expensive and often require specialized hardware.	Low scalability. Due to high computational cost, it is less suitable for large-scale deployments.	High impact. The complexity of encryption operations typically results in reduced model accuracy.	Strong privacy guarantees, but significant performance degradation due to computational overhead.
Secure Multi-Party Computation (SMPC)	Effective against untrusted servers. Provides robust privacy by distributing computations across multiple parties.	High computational cost, particularly with large-scale data. Relies on multiple rounds of secure communication.	Low scalability. SMPC is typically not scalable for large federated setups with numerous clients.	Moderate impact. SMPC may cause delays in model training and accuracy loss due to the distributed nature of computation.	Strong privacy protection, but expensive in terms of computation and communication, leading to trade-offs with efficiency.
Federated Learning with Blockchain (FLB)	Effective against both trusted and untrusted servers. Blockchain ensures integrity and transparency in the aggregation process.	High. Blockchain operations (e.g., verification and consensus) introduce additional overhead.	Medium scalability. Blockchain solutions can handle a moderate number of nodes, but verification processes can slow down the system.	Moderate to high. Blockchain increases latency due to consensus mechanisms, negatively affecting real-time applications.	Strong privacy and transparency, but at the cost of high computational and communication overhead.

Table 2 provides a clear and concise comparison of the strengths and weaknesses of each privacy-preserving technique, along with their respective trade-offs in terms of privacy, model accuracy, and scalability.

Table 2. Strengths, Weaknesses, and Trade-Offs of The Privacy-Preserving Techniques

Privacy-Preserving Technique	Strengths	Weaknesses	Trade-offs
Differential Privacy (DP)	Provides strong theoretical privacy guarantees, ensuring individual data cannot be inferred. Well-suited for sensitive domains like aviation, healthcare, and finance.	Significant trade-off between privacy and model accuracy. High noise injection degrades performance, especially at high privacy levels. Computational complexity increases with data size.	Balances strong privacy with potential accuracy loss and higher computational overhead.
Secure Aggregation (SA)	Lightweight solution ensuring the server cannot access individual updates from clients. Efficient when only aggregated updates are needed.	Limited security against advanced attacks (e.g., model inversion). Scalability issues in heterogeneous client environments.	Balances privacy and efficiency, but may lack strong security in complex systems.
Homomorphic Encryption (HE)	Ideal for maintaining encrypted data during training, ensuring data remains private from untrusted servers.	High computational cost. Operations on encrypted data are resource-intensive, leading to high latency and scalability issues.	Strong privacy but sacrifices scalability and model accuracy due to high computational overhead.
Secure Multi-Party Computation (SMPC)	Distributes computation, ensuring no single party accesses the entire dataset. Suitable for sensitive, collaborative applications.	Highly inefficient in large-scale systems. Significant computational cost for secure coordination. Delays in model training due to necessary secure communication.	Strong privacy guarantees but compromises scalability and efficiency in large federated setups.
Federated Learning with Blockchain (FLB)	Enhances transparency and integrity. Verifiable model updates are tamper-proof, providing additional security against adversarial behavior.	High overhead due to consensus mechanisms. Scalability issues due to verification costs, making it unsuitable for large-scale systems.	Adds transparency and security at the cost of increased computational and verification overhead.

2.4. Classification of Threats Based on Actor Capabilities

Privacy threats in FL arise from various sources, including central servers, users, and external adversaries, each with specific objectives and risks. The classification of these threats depends on clear definitions of system assumptions and privacy goals, which enable effective threat modeling. Key parameters—differential privacy (ε), system integrity (σ), and adversarial capabilities (γ)—form

a core framework for assessing and countering these threats. Differential privacy (ε) measures the strength of privacy protection, with lower values indicating greater privacy, while system integrity (σ) reflects the resilience of cryptographic protocols to interference. Adversarial capabilities (γ) define the potential control adversaries have over system interactions, affecting risk levels. Adversary types range from "honest but curious," who observe without altering data, to "malicious," who may compromise user data and system integrity, necessitating robust privacy controls. Incorporating ε, σ, and γ into privacy frameworks allows for a well-rounded defense strategy, although evolving adversarial techniques require ongoing innovation to keep FL systems secure.

Figure 2 presents a classification of threat models according to the capacities of participants involved in model training, with each model elaborated in detail in the subsequent subsections.

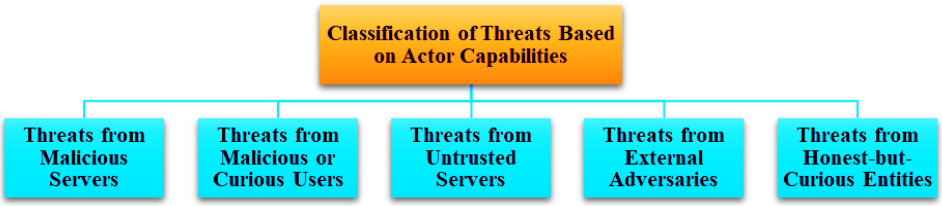

Figure 2. Classification of Threats Based on Actor Capabilities

Threats from Malicious Servers

The central server plays a crucial role in aggregating model updates from user devices. However, if compromised or acting maliciously, significant privacy risks may arise:

Inference Attacks: Malicious servers can analyze the gradients received from devices to infer sensitive information, such as identifying patterns related to health conditions or financial data.

Membership Inference: By examining the updates, servers may determine whether specific data points were part of the training set, potentially leading to privacy breaches.

Example: In a scenario where hospitals utilize FL to train predictive models, a compromised server could infer whether data from a specific patient was part of the training set by closely analyzing updates from a particular hospital, posing a privacy risk.

Threats from Malicious or Curious Users

FL involves collaborative participation, where user devices contribute to the training process. Some users may attempt to extract more information than intended:

Data Poisoning: Malicious users can introduce incorrect or adversarial updates aimed at corrupting the global model or inserting backdoors that trigger specific behaviors.

Model Manipulation and Reverse Engineering: Users may try to reverse-engineer the aggregated model, potentially inferring information about other participants' data.

Example: In a federated language model, a user may analyze updates to infer specific phrases, revealing sensitive information shared by other participants.

This classification framework facilitates a deeper understanding of the diverse privacy threats in FL, guiding the development of effective countermeasures.

Threats from Untrusted Servers

In scenarios where the central server cannot be fully trusted, additional privacy concerns arise:

Collusion Attacks: A server may collude with selected users to isolate and analyze data contributions from others, undermining the privacy of individual users.

Sybil Attacks: Malicious entities may create fake identities to exert undue influence over the training process, thereby increasing the risk of data exposure.

Example: In a multi-organizational FL setup, an organization could deploy fake user devices to manipulate the model or weaken privacy guarantees, enabling it to extract sensitive data from other organizations.

Threats from External Adversaries

Privacy risks may also originate from external entities attempting to interfere with FL operations:

Eavesdropping: External attackers may intercept communications between users and the server, potentially extracting sensitive information.

Man-in-the-Middle Attacks: Adversaries can position themselves between users and the server to alter or redirect data packets, compromising both the privacy and integrity of the model.

Example: In a mobile FL application, an attacker intercepting data updates may infer a user's browsing history or app usage, leading to a privacy breach.

Threats from Honest-but-Curious Entities

Even entities adhering to standard protocols may unintentionally compromise privacy:

Gradient Leakage: Shared gradients can reveal sufficient information to reconstruct local data, posing a significant risk even in the absence of malicious intent.

Data Reconstruction: Analyzing aggregated updates may allow for the reconstruction of original data, especially if updates are not sufficiently obfuscated.

Example: In FL systems for smart home devices, gradient updates may inadvertently reveal patterns that indicate users' daily routines or presence at home.

2.5. Integration of Multiple Privacy-Preserving Strategies

Achieving comprehensive privacy in FL often necessitates the integration of multiple tools and technologies. This approach involves examining the following:

Layered Approaches: Implementing strategies such as MPC within TEEs to enhance security.

"Privacy in Depth": Principles wherein multiple privacy mechanisms, such as MPC, DP and TEEs, are combined to ensure that even if one layer fails, others can maintain security [5].

3. Techniques to enhance privacy in FL

Figure 3 shows the various tools and technologies that can be classified to implement privacy protection against identified threats in section 2 These mechanisms are designed to increase data security in federated learning systems by reducing vulnerabilities and addressing potential privacy risks. Each tool contributes to a robust privacy framework, ensuring data confidentiality and integrity throughout the federated learning process.

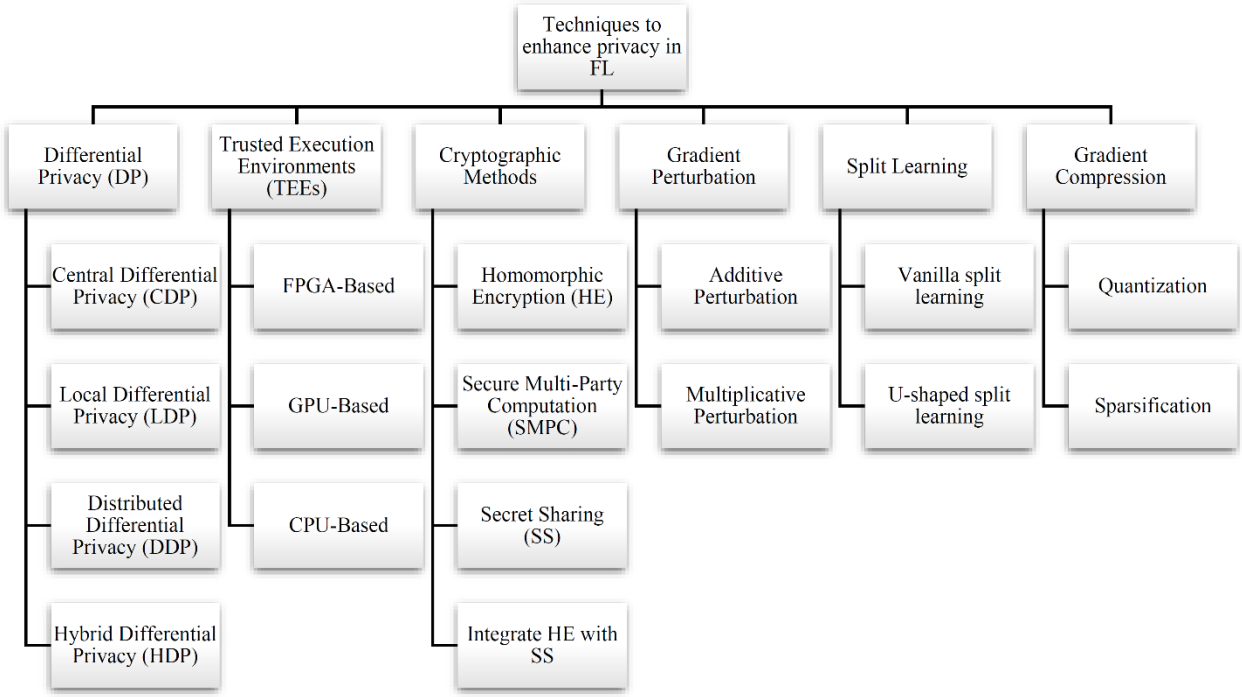

Figure 3. Tools and technologies to enhance privacy in FL

3.1. Differential Privacy (DP)

Over the past decade [6, 7] a wide range of techniques has been developed for differentially private data analysis, primarily under the assumption of a centralized setting, where raw data is collected by a trusted entity that applies the necessary perturbations to ensure privacy. In the context of federated learning, the coordinating server typically assumes the role of the trusted implementer of the

DP mechanism, ensuring that only privatized outputs are provided to model developers or analysts. However, minimizing the reliance on a trusted party remains a key objective. Types of Differential Privacy:

Central Differential Privacy (CDP)

In this model, a trusted server collects raw data and applies noise to the outputs to preserve privacy. This approach assumes the server is reliable and can securely implement the differential privacy mechanisms. DP introduces controlled randomness to prevent the disclosure of individual data. In federated learning, DP is typically implemented at the user level, ensuring that individual user contributions remain indistinguishable.

Local Differential Privacy (LDP)

LDP [8] is a privacy-preserving framework where users add noise to their data before sharing it, minimizing the reliance on a central server's security. This approach enhances user privacy by reducing central points of vulnerability. However, ensuring that the data remains useful while being private is a significant challenge under LDP. Tech giants like Google and Apple have adopted LDP for secure data collection, acknowledging its potential in safeguarding user information [9]. The main challenge lies in achieving privacy without sacrificing the utility of the data. This involves a delicate balance between the amount of noise added, the computational resources required, and the overhead in communication. Efficient implementation of LDP is crucial for its success, as it must maintain the integrity and usefulness of the data while providing robust privacy protections. The trade-offs between privacy, utility, and efficiency are at the core of LDP's challenges and continue to be an area of active research and development.

Distributed Differential Privacy (DDP)

This model combines elements of both Central and Local DP, employing techniques such as secure aggregation and shuffling to minimize reliance on a central server while preserving data utility. DDP often leverages MPC or trusted execution environments (TEE) to enable privacy-preserving computations through secure data processing [10].

Hybrid Differential Privacy (HDP)

Integrates different trust models to optimize privacy and efficiency. By allowing multiple privacy models to coexist, hybrid mechanisms achieve greater flexibility compared to purely local or centralized approaches [11].

This methodology integrates disparate trust paradigms by categorizing users based on their preferences for trust frameworks. The pioneering HDP-VFL [12] represents the inaugural hybrid DP architecture for VFL, which facilitates the collaborative training of a generalized linear model (GLM) within the VFL environment at a nominal expense relative to the hypothetical non-private VFL. It employs mechanisms such as Secure Multi-party Computation (SMC) and Homomorphic Encryption (HE) to ensure the preservation of privacy.

Distributed DP via Secure Aggregation: Ensures that only aggregated results are accessible, preserving privacy without exposing intermediate data [13].

Secure Shuffling: Secure shuffling involves collecting multiple messages from users and ensuring that the server can only observe an unordered set of messages, thereby preserving anonymity. The methods employed in secure shuffling often overlap with those used in secure aggregation. Secure mixers, which are frequently discussed within the field of multi-party secure computing, are sometimes referred to as hybrid networks. Trusted computation techniques have also been explored in this context, with large-scale hybrid networks implemented in systems like the Tor network [14].

Distributed DP via Secure Shuffling: Involves randomizing inputs to prevent the server from linking data to individual users, enhancing privacy [12].

3.2. Trusted Execution Environments (TEEs)

TEEs support secure execution of certain components in FL by operating within a trusted enclave, which ensures confidentiality, integrity, and verifiable execution. Different TEE implementations, such as Intel SGX-enabled CPUs [15] and Sanctum for RISC-V architectures [16], vary in their security capabilities. TEEs face limitations in memory and computational resources, prompting research to extend TEE functionalities, such as integrating them with GPUs for improved machine learning performance [17]. Key features of TEEs include confidentiality, where execution states are kept private unless intentionally disclosed; integrity, where execution remains unaffected except by defined inputs; and attestation, where a TEE can prove to remote parties the exact code and initial conditions running within it. TEEs are particularly valuable in cases where kernel-level OS vulnerabilities are present, providing robust privacy protections via hardware-level assurances [18]. As hardware-based machine learning solutions gain

popularity, systems leveraging TEEs, such as Intel SGX, are becoming essential for privacy-preserving FL. Solutions like ShuffleFL [19] and FLASH [20] utilize hardware-optimized structures to enhance gradient preservation and accelerate cross-silo FL, while platforms like FLATEE [21] reduce training and communication times by implementing TEEs. Furthermore, hardware acceleration using field-programmable gate arrays (FPGAs) [22], GPUs [23], and CPUs [24] is increasingly crucial to addressing the computational demands of privacy-preserving federated learning (PPFL), especially for homomorphic encryption tasks, which has greatly advanced the efficiency and practical implementation of FL systems.

FPGA-Based

Field-Programmable Gate Arrays (FPGAs) [25] are adaptable semiconductor circuits that can be reprogrammed post-manufacturing to accelerate a variety of tasks, including conventional machine learning and cryptography [26, 27]. Recently, FPGAs have been applied to enhance the efficiency of VFL with Homomorphic Encryption (HE), particularly by offloading computationally intensive modular multiplication operations used in the Paillier cryptosystem [28]. Yang et al. [29] introduced an FPGA-based framework for VFL that streamlines this process, while FLASH [20] expanded this approach to support a wider range of cryptographic functions within the Paillier system. By enabling custom circuit design with fine-grained control and ample on-chip memory, FPGAs optimize HE operations through efficient pipelining and large data storage, significantly improving the computational performance of HE in cross-silo federated learning setups.

GPU-Based

GPUs, known for their parallel processing capabilities, are instrumental in expediting the training of machine learning models. Specifically, in the realm of HE applied to VFL, a GPU-accelerated approach named HAFLO has been introduced for logistic regression task [23]. This method enhances the efficiency of homomorphic operations, reducing the computational burden of the Paillier cryptosystem and improving data handling on GPUs. Further advancements in GPU utilization for HE-based Privacy-Preserving Federated Learning (PPFL) are also noted, with significant improvements in processing speed being achieved through memory optimizations [30, 31]. Despite these advancements, GPUs face challenges in handling large numerical operations required for HE, which hinders their potential for high-performance HE computations [32]. Nevertheless, the development of GPU-optimized schemes like CARM [33], intended for IoT applications, indicates a positive trajectory for the integration of HE in PPFL systems [34], although adaptation to federated settings remains a work in progress.

CPU-Based

In 2021, Intel introduced the Intel Homomorphic Encryption acceleration library (HEXL) [35], which marked a significant shift from the reliance on specialized hardware like GPUs and FPGAs for homomorphic encryption. HEXL utilizes the SIMD capabilities and Intel AVX-512 instructions of Intel CPUs to accelerate Fully Homomorphic Encryption (FHE), making it more accessible and efficient. Specifically, HEXL has optimized operations such as modular exponentiation within Partial Homomorphic Encryption (PHE) in the context of federated learning. The survey discussed highlights inefficiencies in both PHE and FHE, with PHE depending on modular operations and FHE on more complex polynomial-based operations [36]. While algorithmic techniques like FFT and NTT can improve the speed of polynomial operations, challenges in accelerating these due to computational complexity, memory demands, and limited generalizability remain. HEXL's introduction is a step towards addressing these inefficiencies by leveraging widely available CPU instructions for homomorphic encryption acceleration.

3.3. Cryptographic Methods

Secure Multi-Party Computation (SMPC) [37]

The goal of secure computation is to perform calculations on distributed data while revealing only the results to authorized parties. MPC, a field in cryptography, allows multiple participants to jointly compute a function over their private inputs without revealing the inputs themselves. Although originally a theoretical concept, MPC has evolved into a practical technology that enables secure machine learning operations [38]. Cryptographic solutions frequently operate within restricted data ranges, which presents difficulties when working with real numbers. To address this, techniques such as normalization and precise quantization are required to adapt machine learning models for secure environments. Specialized protocols are being developed to optimize operations for particular applications, including linear regression and neural network training. MPC protocols have been employed in cross-silo

environments to facilitate secure collaborative computation. Additionally, reviews of homomorphic encryption software libraries, along with criteria and features to consider when selecting an appropriate library, are available in the literature [39].

Two or more participants collaborate to simulate, though cryptography, a fully trusted third party who can:

Compute a function of inputs provided by all the participants;

Reveal the computed value to a chosen subset of the participants, with no party learning anything further.

Private Information Retrieval (PIR): Private Information Retrieval allows users to query databases without revealing the content of the query to the server. PIR can be achieved through computational or information-theoretic approaches, each offering distinct advantages and limitations. Approaches to PIR using MPC are typically divided into two categories: computational PIR (cPIR), where a single party handles the entire server-side protocol, and information-theoretic PIR (itPIR) [40], which requires multiple non-colluding parties to execute the server-side protocol.

Homomorphic Encryption (HE)

HE facilitates computations directly on encrypted data, ensuring that the data remains secure throughout the computational process. HE can be categorized into several types, including fully homomorphic encryption (FHE) [36] and partial schemes, such as ElGamal and Paillier, which support specific operations like addition or multiplication. Approaches proposed by Reyzin et al. and and Roth et al. [41, 42]. have employed these techniques to enable joint computations across multi-device or cross-device environments, utilizing incremental homomorphic schemes. In the context of federated learning, a significant challenge is encryption key management, which can be addressed through distributed key management systems or by relying on non-collusive trusted third parties [43].

Enables a party to compute functions of data to which they do not have plain-text access, by allowing mathematical operations to be performed on ciphertexts without decrypting them. Arbitrarily complicated functions of the data can be computed this way ("Fully Homomorphic Encryption") though at greater computational cost.

Homomorphic Encryption: This method allows computations on encrypted data, ensuring that sensitive information remains confidential during model training [44].

Secret Sharing (SS)

Secret sharing is a cryptographic technique in which a secret is divided into N distinct shares, allowing reconstruction only when a sufficient subset of these shares is combined. This approach has been applied in studies such as [45-47]. The t-out-of-n method [48] illustrates how a data element (or secret) S can be partitioned into n segments, enabling reconstruction from any subset of m parts. Critically, possessing only $m-1$ parts reveals no information about S. This method facilitates the design of robust key management systems within cryptographic frameworks, supporting secure and reliable operational continuity.

Integrate HE with SS

Numerous studies integrate homomorphic encryption (HE) with secure secret sharing (SS) to prevent the leakage of intermediate computational results to unauthorized clients. For example, Pivot [49] combines SS and HE to ensure the confidentiality of intermediate results during the aggregation of general tree models, including random forests (RF) and gradient-boosted decision trees (GBDT). In the Pivot framework, HE is primarily utilized to facilitate local computations on the client side, while SS is employed selectively in cases where HE lacks the necessary functional capacity. This hybrid approach not only enhances security but also significantly improves computational efficiency, particularly in applications involving vertical tree models.

Secure Aggregation: SA enables multiple users to submit values to a central server, which can only access the aggregated result without revealing individual inputs. Techniques utilized for secure aggregation include incremental masking [50], threshold homomorphic encryption [51], and secure multi-party computation [52].

3.4. Gradient Perturbation

Perturbation methods [53] are a set of techniques used in machine learning to enhance data privacy without requiring knowledge of the underlying data distribution. These methods work by adding random noise to the model's parameters, which makes the modified data statistically similar to the original, thus preserving privacy in a manner comparable to DP. An example of this is the Privacy-Encoding based FL (PEFL) model [54], which integrates a long short-term memory-autoencoder with perturbation

encoding to create a federated learning model for detecting network intrusions. This approach is effective in preventing attacks that aim to reconstruct the original data or infer membership information by introducing randomness into the model's parameters, much like DP. However, unlike DP, perturbation methods have the advantage of preserving the accuracy of the learning process, as the noise can be subtracted out by the server to recover the true gradients. Despite their effectiveness, these methods are not immune to probabilistic privacy attacks. The three main perturbation techniques currently in use are Differential Privacy detailed in subsection 3.1, Additive Perturbation, and Multiplicative Perturbation, each providing a different mechanism for securing data privacy while enabling accurate machine learning models.

Additive Perturbation

Additive perturbation techniques [55] are straightforward, efficient methods for ensuring data privacy without the need for understanding the data's distribution. They work by adding random noise, drawn from specific distributions like uniform or Gaussian, to the original data. This process helps to maintain the statistical properties of the data while protecting its privacy. Such techniques are used in privacy-preserving data mining to add random disturbances to data, thereby preserving its probabilistic characteristics. The goal is to achieve a balance between data privacy and integrity, introducing enough noise to protect the data but not so much that it obscures the original signal, ensuring patterns can still be accurately estimated. While additive perturbation methods are cost-effective, easy to implement, and can be applied to individual data points, they do have drawbacks. Specifically, they can reduce the overall utility of the data and may be susceptible to techniques that aim to reduce noise and potentially compromise privacy.

Multiplicative Perturbation

A multiplicative perturbation approach [56] employs models resistant to transformation, such as rotation and translation, allowing for the direct use of altered data without additional random noise. This technique effectively shifts the original data into a new domain while maintaining critical information pertinent to the task and model, ensuring the model's precision. Traditional methods that solely rely on additive perturbation often fall short; however, integrating both additive and multiplicative perturbations overcomes these shortcomings, as evidenced in recent studies [57]. The POLARSGD protocol, which stands for Private Optimization and Learning Algorithm with Stochastic Gradient Descent, incorporates this dual perturbation strategy to conceal client gradients before they are sent to various servers for model development.

In response to the inherent challenges, the study presents DISTPAB, a distributed perturbation approach aimed at bolstering privacy within Hierarchical Federated Learning frameworks. DISTPAB capitalizes on the uneven distribution of resources in such networks to decentralize the privacy protection process, thus reducing computational strain. It achieves this by employing independent Gaussian random projections to mask data on each Internet of Things (IoT) device, allowing for the training of a Deep Neural Network model at the server level with data from IoT clients [58]. This method primarily assigns the computational load to the coordinator, which is typically resource-rich, thereby lessening the burden on IoT devices. Comparative studies, including those involving support vector machines and additive noise for differential privacy, have shown that this method is more effective than other privacy-preserving techniques.

3.5. Split Learning

The split learning configuration is illustrated in Figure 4(a), which presents the standard Vanilla split learning setup. An alternative configuration, where the labels are kept private, is demonstrated in Figure 4(b), referred to as U-shaped split learning. Additionally, Figure 4(c) displays the arrangement for split learning with vertically partitioned data. Each figure provides a unique setup to facilitate different aspects of split learning while ensuring data privacy and integrity.

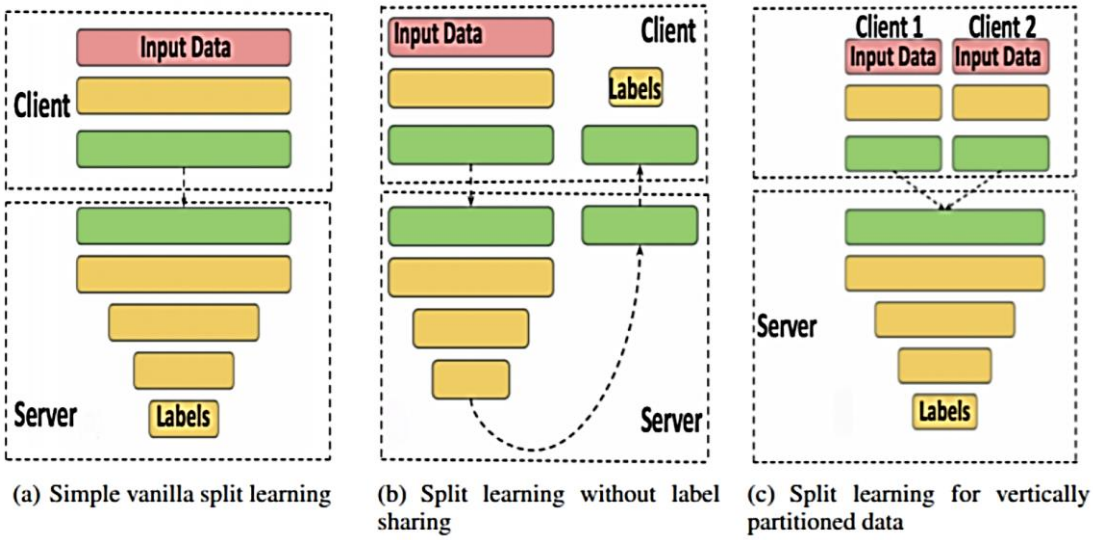

Figure 4. Split Learning [59]

Vanilla split learning

Split learning presents a new approach to collaborative machine learning, distinct from traditional data partitioning and communication-focused methods. Instead of each client solely working on separate model components and communicating results, split learning distributes the model itself between clients and a server. Each client processes data through a deep neural network up to a predetermined point, known as the "cut layer." At this layer, outputs, or "smashed data," are passed on to the server (or occasionally another client), which completes the forward computation. This setup not only facilitates distributed model training and inference but also enhances privacy by minimizing the need to share raw data. During backpropagation, gradients calculated from the server's end are sent back to the cut layer, allowing clients to handle backpropagation locally. This iterative process continues until convergence, enabling privacy-preserving training as clients avoid direct data exchange [60].

U-shaped split learning

Beyond its primary configuration, split learning can adapt to unique needs. For example, in scenarios where label privacy is essential, a U-shaped configuration allows training without label sharing. In this setup, clients process the forward pass up to the cut layer, after which the server completes computation but returns the output back to the client, avoiding any exchange of sensitive label information. This method, particularly advantageous for privacy-sensitive tasks like medical diagnosis, ensures confidentiality by localizing gradient generation and backpropagation on the client's side [59].

Vertically partitioned data for split learning

The flexibility of split learning is further demonstrated in configurations with vertically partitioned data, where multiple clients contribute distinct features for model training. Each client trains a portion of the model up to its respective cut layer, and the server combines outputs to complete the forward pass. This approach, called splitNN [61], performs collaborative model training without sharing raw data across clients or with the server. Additionally, split learning facilitates the matching of client-side and server-side model components to optimize model selection, as seen in ExpertMatcher [62]. To address this, methods like NoPeek SplitNN [63] reduce data correlation between inputs and transmitted outputs, enhancing privacy while maintaining model performance. Further techniques, such as selective pruning of client-side activations, have been proposed to limit information exchange. However, ensuring formal privacy guarantees for split learning remains a challenge.

3.6. Gradient Compression Methods

In federated learning, client devices often face limitations in bandwidth and energy, creating challenges in communication efficiency. Adapting centralized compressed communication schemes to decentralized environments without affecting convergence is a significant research focus, along with developing algorithms that naturally produce sparse updates. Communication remains

a primary bottleneck, with consumer network speeds often lower and less reliable than datacenter connections. Efforts to reduce communication requirements through Federated Averaging [64] combined with sparsification or quantization have shown promising results, but questions remain on achieving optimal trade-offs between communication and accuracy. Recent theoretical studies [65] explore the minimum possible rates for accuracy under communication constraints, but practical applications remain elusive due to the complex influence of optimization algorithms. Compression efforts target three main areas: reducing client-server communication, minimizing model broadcast sizes, and optimizing local training computation.

Compression objectives, informed by the resource limitations of contemporary devices in computation, memory, and communication, span several goals:

Gradient compression – minimizing the data communicated from clients to the server, which is used to update the global model.

Model broadcast compression – reducing the size of the model broadcast from the server to clients, which serves as the starting point for local training.

Local computation reduction – modifying the training algorithm to enhance the computational efficiency of local training.

These objectives collectively aim to address the resource constraints prevalent in federated learning scenarios.

Quantization

Existing noise addition mechanisms assume adding real-valued Gaussian or Laplacian noise on each client, and this is not compatible with standard quantization methods used to reduce communication.

Sparsification

To address privacy concerns, anonymization techniques that remove identifiable information from datasets are employed, which helps protect user privacy while supporting effective model training.

Anonymization: Removing identifiable information from datasets helps protect user privacy while still allowing for effective model training [66].

These tools and technologies facilitate secure, efficient, and privacy-preserving computation within federated learning frameworks. They ensure the protection of sensitive data while supporting robust machine learning and statistical analysis, thereby enabling collaborative data processing without compromising privacy.

3.7. Verification in Federated Learning

Zero-Knowledge Proofs (ZKPs)

ZKPs enable one party to prove to another that a computation was performed correctly without revealing the underlying data. This ensures verifiable and secure computations, which is crucial in federated learning environments. Xie et al [67] introduced a ZKP system named Libra, which achieves linear complexity for the prover by increasing the size of the proof and the time required for verification.

Remote Attestation and TEEs

TEEs can demonstrate that computations are executed securely as expected within a protected environment. This ensures that data is processed without tampering, enhancing trust in federated learning systems. In addition to ZKPs, TEEs support remote verification by ensuring that computations are conducted as expected within a secure, isolated environment. This guarantees that data is processed without tampering, thereby enhancing trust in federated learning systems [59].

4. Challenges in Privacy Preservation and Open Problems

Despite advances, challenges remain, including balancing privacy, utility, and communication efficiency, developing robust privacy guarantees against sophisticated attacks, and ensuring graceful degradation of privacy when parts of the system fail.

Inference Attacks: Attackers can exploit shared model parameters to reconstruct sensitive data, necessitating robust defenses.

Malicious Server or Client Risks: A compromised server can manipulate aggregated data, leading to privacy breaches.

Fairness vs. Privacy: Balancing individual model quality and privacy remains a challenge, as traditional aggregation methods may expose sensitive information [68].

4.1. Protection Against External Malicious Entities

This section assumes the presence of a trusted server and explores various challenges and open issues related to privacy assurance against external malicious entities. These entities include hostile users, analysts, and devices that may use trained models or any combination thereof. Adversarial users can inspect all messages received from the server

during periods of participation, including model iterations. Adversarial analysts can scrutinize the data sequences across multiple training sessions with varying hyperparameters, and in cross-device federated learning environments, malicious devices can access the final model as a white-box or black-box entity. Therefore, to effectively guard against external adversaries, it is essential first to assess what information can be inferred from intermediate iterations and the final model.

Final Model Adoption

In federated learning, assessing model sensitivity to attacks is crucial due to the broader attack surface, as adversaries may gain access to the model through the server. The server, however, can regulate adversarial access at different training stages, potentially controlling their impact. For traditional (non-federated) models, understanding attack sensitivity involves simulating attacks on a proxy dataset that mimics user data. Although useful, this approach often relies on impractical assumptions, and setting a worst-case sensitivity bound remains challenging [69]. Current research [70] seeks theoretical guarantees that would indicate if simulated attacks fail to reveal privacy breaches, stronger attacks are unlikely to succeed, though further work is needed in this area.

Federated learning introduces unique opportunities for both measuring and defending against attacks. Server-controlled access during training allows for innovative methods to test model sensitivity. Such methods could inform adaptive defenses that adjust protections in real-time to mitigate adversarial influence effectively while preserving model efficiency.

Training with Central Differential Privacy

User-level differential privacy can be applied during the iterative training process in FL to limit or eliminate what can be learned about individuals from the model iterations (or the final model). This approach involves the server clipping individual updates, aggregating them, and adding Gaussian noise to ensure the model does not align closely with any single user's updates. Advanced composition theorems or methods like the moments accountant can be used to track the overall privacy budget across rounds, particularly when employing subsampled Gaussian mechanisms. However, adding noise may reduce model accuracy, especially when data is sparse, leading to a trade-off between privacy and accuracy. Solutions to this trade-off include aggregating more private data, designing privacy-preserving model architectures, or incorporating priors that impact the domain of private data.

In cross-device FL, training samples can vary significantly across devices, making it important to discover adaptive methods to limit user participation and adjust model parameters. Unlike record-level DP, the trade-off between accuracy and privacy for user-level DP is not well understood, especially when contributions vary greatly across users. Recent advances have described this trade-off for learning discrete distributions under user-level DP, but further research is needed to fully comprehend it.

Distinguishing between adversarial users who can observe intermediate iterations and adversarial analysts who can only view the final model is important. Even if central DP can protect against both, it may require different privacy parameters, with stronger guarantees for analysts, as adversarial users have access to more information. This approach, known as "privacy amplification through iteration," has been studied by Feldman et al [71], especially in convex optimization contexts, although its application in non-convex scenarios remains uncertain.

Enhancing Privacy in cross-devices FL setting

- Providing formal privacy guarantees (ϵ,δ) in a cross-device FL system is challenging because:
- The set of eligible users (the underlying dataset) is dynamic and unknown in advance.

Participants in FL can opt out of the protocol at any time.

Therefore, protocols must be designed to be self-accounting and reliant on local participation decisions without assuming the server is aware of which users are online. They must also balance privacy with efficiency. While recent research suggests that these constraints are manageable, building an end-to-end protocol that operates effectively in production FL systems remains a significant challenge.

Concealing Iterations

In conventional FL systems, models are iteratively updated after each training session, with the updated versions visible to multiple actors, including the server and users. However, tools such as TEEs can potentially conceal these iterations from users. The server can verify that the expected FL code has been executed within the TEE, ensuring confidentiality and sending an encrypted model to be decrypted only within the TEE. Unfortunately, TEEs may not be widely available, particularly in end-user devices like smartphones, and might lack the power to

handle the computational needs of training. However, as TEE capabilities improve, methods for partial computation offloading while maintaining integrity may mitigate these limitations.

Another approach within the MPC framework involves encrypting model parameters before sending them to users. Thanks to homomorphic encryption, users can perform updates on the encrypted models without decrypting the parameters. The challenges here include ensuring that servers do not decrypt the data before aggregation and improving performance, as advanced systems still require high computational resources.

Iterative Analysis of Dynamic Data

Analysts often need to analyze data streams and provide dynamically updated models that incorporate past data while accurately predicting future data. Without privacy concerns, this can be achieved by retraining models as new data arrives. However, privacy considerations mean that updates must be frequent enough to maintain both privacy and accuracy. Extending differential privacy techniques for dynamic databases to FL environments to enable private learning over time-series data remains an ongoing challenge.

Preventing Model Theft and Misuse

Developers may wish to restrict access to their machine learning models to prevent misuse or theft. Techniques for protecting models during inference often mirror those used to conceal them during training, with TEEs and MPC being key tools. However, challenges include balancing the benefits of on-device inference with ensuring that cryptographic technique keys do not compromise security. Furthermore, research indicates that adversaries can sometimes reconstruct models solely through access to inference APIs, raising further concerns about protecting models deployed across millions of devices.

4.2. Protection against Untrusted server

In the previous section, we assumed that there is a trusted server that can coordinate the training process. In this section, we explore a scenario where we protect a potentially hostile server. In particular, we begin by examining the challenges related to these settings and studying existing works, and then outline open issues and how to use the methods discussed.

Communication channels

In a cross-device FL configuration, we are dealing with a server that has significant computing resources and a large number of users that (1) can only communicate with the server (such as a star network topology) and (2) may be limited in connectivity and bandwidth. be limited, we face; These conditions create specific requirements for the implementation of a specific trust model. In particular, users without dependence on the server do not have an obvious way to establish secure channels between themselves, as shown by Reyzin et al [41]. In practical settings that assume honest (or at least semi-honest) behavior on the part of the server in the key distribution phase/phase in scenarios requiring private channels between users, cryptographic solutions based on MPC methods are included. An alternative assumption could be to include an additional party or a public bulletin in a model that is known by users and is sure not to collude with the server.

Sybil Attacks

Instead of trusting the server in private communication channels, participants in cross-device FL must trust the server to form groups of users fairly and honestly. An active malicious user in control of the server can clone a large number of fake user devices (a Sybil attack) or can select previously vulnerable devices from the pool of available devices. Either way, an adversary can have more control over the participants in a federated learning round than the number of devices expected to be in the population. This makes it much easier to break the usual assumption in secure multiparty computing that at least a certain number of devices are true, thereby weakening the security of the protocol. Even if the security of the protocol itself remains intact (for example, if its security is based on a different source of trust, such as a secure enclosure), there is a risk that if a large number of users' model updates are known or controlled by an adversary. That is, the privacy of the remaining users' updates is undermined. Note that these considerations can also apply to TEE contexts, for example, a TEE-based mixer can also be vulnerable to a Sybil attack. If an honest user's input is combined with known inputs from fake users, the adversary can directly identify the honest user's value in the combined output.

Note that in some cases, it may be possible to prove in one round between users that they are all running the correct protocol, for example, if there are secure fences on user devices and users can verify each other remotely. In these cases, it may be possible to establish privacy for all honest participants in the round, for example by verifying that secure multilateral computing protocols are strictly

followed, distributed differential privacy contributions are cryptographically and correctly added, etc. Even if the model updates themselves are known or controlled by the adversary.

Limiting the ability of the server

Considering that the goal of FL is to build a model of general patterns in users' data, one of the main goals of privacy protection is to reduce and limit the server's ability to reconstruct a particular user's input data. This includes a formal definition of (a) the representation of user data that is provided to the server as a result of FL execution, and (b) the extent to which privacy information is leaked from such representation. In FL, the server can collect user logs, while somehow hiding each user's contribution. This can be done in a number of ways, usually involving the use of some differential privacy concept. There is a wide range of these methods, each with its own weaknesses, particularly in FL, where central DP is limited by the need to access a trusted central server.

5. Future Directions

Privacy Requirements for Specific Applications: In Federated Learning, user data can be complex and multidimensional, often necessitating significant noise to ensure differential privacy. However, if users do not prioritize protection against all possible inferences, privacy constraints can be relaxed to reduce the level of noise added. For instance, data from a smart thermostat programmed to turn on or off based on occupancy patterns could reveal sensitive information, such as typical return times of residents. Conversely, data indicating whether residents were asleep during specific hours might be less sensitive.

The Pufferfish privacy framework [72] allows analysts to specify which inferences should be protected with differential privacy, while less sensitive inferences may not require such stringent protection. To ensure that this approach provides adequate privacy guarantees, analysts must first understand users' privacy preferences by gathering and analyzing relevant data. The FL framework can be adapted to enable users to specify which inferences are permissible, with these preferences processed locally on devices or incorporated into the aggregation process. Further research is needed to develop tools that integrate user preferences into the FL model and meaningfully extract preferences from users.

Updating Models with New Data: How should analysts privately update an FL model with new data, and to what extent can a model trained on dataset D generalize to a similar dataset?

On the other hand, how well can a model trained privately using FL on a dataset called D generalize to another dataset that is guaranteed to be similar to D on a certain criterion? Given that in FL, samples taken online do not cause overfitting, it is likely that such a model can still perform well on a new dataset.

One way to circumvent the problem of privacy fusion is to generate synthetic data that can be used indefinitely without harming privacy. This is the result of differential post-processing privacy guarantees. Augenstein et al. [73] have investigated the generation of synthetic data in a federated manner. In a dynamic data setting, synthetic data can be used repeatedly until it becomes "out of date" with new data and needs to be updated. Even after synthetic data is federated, updates must be made privately and federated.

Adapting Differential Privacy Approaches: Can differential privacy techniques for dynamic datasets or time-series data be adapted for federated learning?

First, how to query time series data in a federated model? By design, the same users are not asked regularly and multiple times for updated data points, so it is difficult to gather accurate within-subject estimates of the evolution of individual data over time. Conventional tools for statistical sampling of time series data may be used here, but should be used in conjunction with privacy-preserving and federated tools. Other approaches include reformulating the questions so that each sub-question within a topic can be fully answered on the device.

User perception Behavior to Derive Privacy Settings: Any strategy that requires users to define their privacy standards must include behavioral and field studies to ensure users are aware of their privacy preferences. This approach should involve educating users and assessing their understanding of privacy practices and data usage details. For federated educational applications, it is essential to confirm that average users can meaningfully grasp the privacy assurances provided by these privacy-preserving learning processes. Once this understanding is achieved, researchers can begin to extract users' privacy preferences.

This process can be conducted across various settings, including behavioral labs, large-scale field experiments, or small focus groups. Care must be taken to ensure that

participants providing data about their preferences are both informed and representative of the broader population. While behavioral research has shown that individuals often behave differently in public versus private settings, there is limited research on how to elicit privacy preferences. Advancing this area is vital for broader adoption of private federated learning in the future.

6. Conclusion

Protecting user data privacy requires a dual focus: understanding *what* operations are performed on the data and *how* these operations are conducted, particularly regarding who has access to the data or can influence it. Addressing the "what" involves techniques such as data minimization and differential privacy, which aim to reduce the exposure of sensitive information. However, applying these techniques in real-world scenarios remains challenging, especially when training machine learning models on diverse and dynamic datasets managed by independent actors.

To tackle the "how," approaches such as SMPC, HE, and TEEs are employed. While MPC has seen practical implementation, it can still be resource-intensive, and establishing secure and reliable TEE environments continues to pose significant challenges. Effective privacy-preserving solutions must integrate these methods to ensure that privacy is maintained, even if one of the components fails.

Distributed differential privacy combines both the "what" and "how" strategies, offering a balanced approach that ensures high accuracy while protecting data privacy, even against honest-but-curious servers. Additionally, bidirectional verifiability, enabled through techniques such as zero-knowledge proofs and TEEs, allows participants to verify the correctness of computations, thereby enhancing trust. However, addressing risks posed by potentially malicious servers remains a critical area for ongoing development.

This paper tried to present a structured approach to analyze privacy threats in federated learning. By categorizing threats based on the roles and capabilities of different actors, strong defense mechanisms can be designed and implemented. Addressing these privacy concerns requires a combination of technical measures, including secure computing, differential privacy, and encryption, to ensure that FL systems can be securely deployed in a variety of applications. Understanding these threat models enables researchers and practitioners to develop comprehensive strategies to protect user data and support the safe and effective use of federated learning in various domains.

Authors' Contributions

All authors equally contributed to this study.

Declaration

In order to correct and improve the academic writing of our paper, we have used the language model ChatGPT.

Transparency Statement

Data are available for research purposes upon reasonable request to the corresponding author.

Acknowledgments

We would like to express our gratitude to all individuals helped us to do the project.

Declaration of Interest

The authors declare that they have no conflict of interest. The authors also declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Funding

According to the authors, this article has no financial support.

Ethical Considerations

The study placed a high emphasis on ethical considerations. Informed consent obtained from all participants, ensuring they are fully aware of the nature of the study and their role in it. Confidentiality strictly maintained, with data anonymized to protect individual privacy. The study adhered to the ethical guidelines for research with human subjects as outlined in the Declaration of Helsinki.

References

[1] K. Jahani, B. Moshiri, and B. Hossein Khalaj, "A survey on data distribution challenges and solutions in vertical and

horizontal federated learning," *Journal of Artificial Intelligence, Applications and Innovations*, vol. 1, no. 2, pp. 55–71, 2024, doi: 10.61838/jaiai.1.2.5.

[2] A. Bittau *et al.*, "Prochlo: Strong privacy for analytics in the crowd," in *Proceedings of the 26th Symposium on Operating Systems Principles*, 2017: ACM, pp. 441–459, doi: 10.1145/3132747.3132769.

[3] E. Batista, M. A. Moncusi, P. López-Aguilar, A. Martínez-Ballesté, and A. Solanas, "Sensors for context-aware smart healthcare: A security perspective," *Sensors*, vol. 21, no. 20, p. 6886, 2021, doi: 10.3390/s21206886.

[4] A. Francillon, Q. Nguyen, K. B. Rasmussen, and G. Tsudik, "A minimalist approach to remote attestation," in *Design, Automation, and Test in Europe (DATE)*, 2014, pp. 1–6.

[5] S. Saha, A. Hota, and A. K. Chattopadhyay, "A multifaceted survey on privacy preservation of federated learning: Progress, challenges, and opportunities," *Artificial Intelligence Review*, vol. 57, p. 184, 2024, doi: 10.1007/s10462-024-10766-7.

[6] C. Dwork, F. McSherry, K. Nissim, and A. D. Smith, "Calibrating noise to sensitivity in private data analysis," in *Theory of Cryptography Conference*, 2006: Springer, pp. 265–284, doi: 10.1007/11681878_14.

[7] C. Dwork and A. Roth, "The algorithmic foundations of differential privacy," *Foundations and Trends in Theoretical Computer Science*, vol. 9, no. 3-4, pp. 211–407, 2014.

[8] S. P. Kasiviswanathan, H. K. Lee, K. Nissim, S. Raskhodnikova, and A. D. Smith, "What can we learn privately?," *SIAM Journal on Computing*, vol. 40, no. 3, pp. 793–826, 2011.

[9] B. Ding, J. Kulkarni, and S. Yekhanin, "Collecting telemetry data privately," in *Advances in Neural Information Processing Systems 30*, 2017.

[10] A. Cheu, A. Smith, J. Ullman, D. Zeber, and M. Zhilyaev, "Distributed differential privacy via shuffling," in *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, 2019: Springer, pp. 375–403.

[11] B. Avent, A. Korolova, D. Zeber, T. Hovden, and B. Livshits, "BLENDER: Enabling local search with a hybrid differential privacy model," in *26th USENIX Security Symposium*, 2017: USENIX Association, pp. 747–764.

[12] C. Wang, J. Liang, M. Huang, B. Bai, K. Bai, and H. Li, "Hybrid differentially private federated learning on vertically partitioned data," in "arXiv," 2020.

[13] V. Rastogi and S. Nath, "Differentially private aggregation of distributed time-series with transformation and encryption," in *Proceedings of the 2010 ACM SIGMOD International Conference on Management of Data*, 2010: ACM, pp. 735–746, doi: 10.1145/1807167.1807247.

[14] R. H. Humza Ikram, Muaz Ali, Zartash Afzal Uzmi, "VaulTor: Putting the TEE in Tor," 2024, doi: https://doi.org/10.48550/arXiv.2412.16064.

[15] V. Costan and S. Devadas, "Intel SGX explained," in "IACR Cryptology ePrint Archive," 2016, vol. 2016. [Online]. Available: https://ia.cr/2016/086

[16] V. Costan, I. Lebedev, and S. Devadas, "Sanctum: Minimal hardware extensions for strong software isolation," in *25th USENIX Security Symposium*, 2016: USENIX Association, pp. 857–874.

[17] F. Tramer and D. Boneh, "Slalom: Fast, verifiable and private execution of neural networks in trusted hardware," in "arXiv," 2019. [Online]. Available: https://arxiv.org/abs/1806.03287

[18] P. Subramanyan, R. Sinha, I. Lebedev, S. Devadas, and S. A. Seshia, "A formal foundation for secure remote execution of enclaves," in *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, 2017: ACM, pp. 2435–2450, doi: 10.1145/3133956.3134098.

[19] Y. Zhang, Z. Wang, J. Cao, R. Hou, and D. Meng, "ShuffleFL: Gradient-preserving federated learning using trusted execution environment," in *Proceedings of the 18th ACM International Conference on Computing Frontiers*, 2021: ACM, pp. 161–168, doi: 10.1145/3457388.3458665.

[20] J. Zhang, X. Cheng, W. Wang, L. Yang, J. Hu, and K. Chen, "FLASH: Towards a high-performance hardware acceleration architecture for cross-silo federated learning," in *20th USENIX Symposium on Networked Systems Design and Implementation*, 2023, pp. 1057–1079.

[21] A. Mondal, Y. More, R. H. Rooparaghunath, and D. Gupta, "Poster: Flatee: Federated learning across trusted execution environments," in *2021 IEEE European Symposium on Security and Privacy*, 2021: IEEE, pp. 707–709.

[22] Z. Wang, X. Li, Y. Chen, and J. Liu, "PipeFL: Hardware/software co-design of an FPGA accelerator for federated learning," *IEEE Access*, vol. 10, pp. 98649–98661, 2022.

[23] X. Cheng, W. Lu, X. Huang, S. Hu, and K. Chen, "HAFLO: GPU-based acceleration for federated logistic regression," in "arXiv," 2021. [Online]. Available: https://arxiv.org/abs/2107.13797

[24] WeBank and Intel, "Accelerating secure computing for federated learning," *Intel Spotlight Story*, 2024.

[25] R. Fang, S. Jiang, H. W. Chen, W. Ding, and M. S. Chen, "Dual triangular QR decomposition with global acceleration and partially Q-rotation skipping," in *Proceedings of the ICFPT*, 2022, pp. 1–4.

[26] C. Zhang, P. Li, G. Sun, Y. Guan, B. Xiao, and J. Cong, "Optimizing FPGA-based accelerator design for deep convolutional neural networks," in *Proceedings of the ACM/SIGDA International Symposium on Field-Programmable Gate Arrays*, 2015, pp. 161–170.

[27] M. S. Riazi, K. Laine, B. Pelton, and W. Dai, "HEAX: An architecture for computing on encrypted data," in *Proceedings of the 25th International Conference on Architectural Support for Programming Languages and Operating Systems*, 2019, pp. 1295–1309.

[28] B. Che, Wang, Zixiao, Chen, Ying, Guo, Liang, Liu, Yuan, Tian, Yuan, Zhao, Jizhuang, "UniFL: Accelerating Federated Learning Using Heterogeneous Hardware Under a Unified Framework," *IEEE Access*, PP. 1-1. 10.1109/ACCESS.2023.334752, 2024.

[29] Z. Yang, S. Hu, and K. Chen, "FPGA-based hardware accelerator of homomorphic encryption for efficient federated learning," in "arXiv," 2020. [Online]. Available: https://arxiv.org/abs/2007.10560

[30] J. Zhang, X. Cheng, L. Yang, J. Hu, X. Liu, and K. Chen, "SoK: Fully homomorphic encryption accelerators," in "arXiv," 2022. [Online]. Available: https://arxiv.org/abs/2212.01713

[31] W. Jung, S. Kim, J. H. Ahn, J. H. Cheon, and Y. Lee, "Over 100× faster bootstrapping in fully homomorphic encryption through memory-centric optimization with GPUs," *IACR Transactions on Cryptographic Hardware and Embedded Systems*, vol. 2021, no. 4, pp. 114–148, 2021.

[32] M. Kim, Y. Song, S. Wang, Y. Xia, and X. Jiang, "Secure logistic regression based on homomorphic encryption: Design and evaluation," *JMIR Medical Informatics*, vol. 6, no. 2, p. e8805, 2018.

[33] S. Y. Shen, H. Yang, Y. Liu, Z. Liu, and Y. Zhao, "CARM: CUDA-accelerated RNS multiplication in word-wise

homomorphic encryption schemes for Internet of Things," *IEEE Transactions on Computers*, vol. 72, no. 7, pp. 1999–2010, 2023.
[34] Z. Zhao, N. Ling, N. Guan, and G. Xing, "Aaron: Compile-time kernel adaptation for multi-DNN inference acceleration on edge GPUs," in *Proceedings of the 20th ACM Conference on Embedded Networked Sensor Systems*, 2022, pp. 802–803.
[35] F. Boemer, S. Kim, G. Seifu, F. D. M. de Souza, and V. Gopal, "Intel HEXL: Accelerating homomorphic encryption with Intel AVX-512 IFMA52," in *Proceedings of the 9th Workshop on Encrypted Computing and Applied Homomorphic Cryptography*, 2021, pp. 57–62.
[36] C. Gentry, "Fully homomorphic encryption using ideal lattices," in *Proceedings of the 41st Annual ACM Symposium on Theory of Computing*, 2009, pp. 169–178.
[37] A. Kwon, D. Lazar, S. Devadas, and B. Ford, "Riffle," *Proceedings on Privacy Enhancing Technologies*, vol. 2016, no. 2, pp. 115–134, 2016.
[38] M. a. K. Ion, Ben and Nergiz, Ahmet Erhan and Patel, Sarvar and Saxena, Shobhit and Seth, Karn and Raykova, Mariana and Shanahan, David and Yung, Moti, "On Deploying Secure Computing: Private Intersection-Sum-with-Cardinality," *2020 IEEE European Symposium on Security and Privacy (EuroS&P)*, pp. 370–389, 2020, doi: 10.1109/EuroSP48549.2020.00031.
[39] S. Sathya, P. Vepakomma, R. Raskar, and R. Ramachandra, "A review of homomorphic encryption libraries for secure computation," in "arXiv," 2018. [Online]. Available: https://arxiv.org/abs/1812.02428
[40] E. Kushilevitz and R. Ostrovsky, "Replication is not needed: Single database, computationally-private information retrieval," in *Proceedings of the 38th Annual IEEE Symposium on Foundations of Computer Science*, 1997, pp. 364–373.
[41] L. Reyzin, A. D. Smith, and S. Yakoubov, "Turning HATE into LOVE: Homomorphic ad hoc threshold encryption for scalable MPC," in "IACR Cryptology ePrint Archive," 2018, vol. 2018.
[42] E. Roth, D. Noble, B. H. Falk, and A. Haeberlen, "Honeycrisp: Large-scale differentially private aggregation without a trusted core," in *Proceedings of the ACM Symposium on Operating Systems Principles*, 2019, pp. 196–210.
[43] L. Lyu, H. Yu, and Q. Yang, "Threats to federated learning: A survey," in "arXiv," 2020. [Online]. Available: https://arxiv.org/abs/2003.02133
[44] a. M. S. D. Janis Adamek, "Privacy-preserving gradient-based fair federated learning," *arXiv:2407.13881v1*, 2024.
[45] Y. Liu, Z. Ma, Z. Yan, Z. Wang, X. Liu, and J. Ma, "Privacy-preserving federated k-means for proactive caching in next-generation cellular networks," *Information Sciences*, vol. 521, pp. 14–31, 2020.
[46] G. Xu, H. Li, S. Liu, K. Yang, and X. Lin, "Verifynet: Secure and verifiable federated learning," *IEEE Transactions on Information Forensics and Security*, vol. 15, pp. 911–926, 2020.
[47] D. Gao, Y. Liu, A. Huang, C. Ju, H. Yu, and Q. Yang, "Privacy-preserving heterogeneous federated transfer learning," in *2019 IEEE International Conference on Big Data*, 2019: IEEE, pp. 2552–2559.
[48] A. Shamir, "How to share a secret," *Association for Computing Machinery*, vol. 22, 11, pp. 612–613, 1979, doi: https://doi.org/10.1145/359168.359176.
[49] Y. Wu, S. Cai, X. Xiao, G. Chen, and B. C. Ooi, "Privacy-preserving vertical federated learning for tree-based models," *Proceedings of the VLDB Endowment*, vol. 13, no. 12, pp. 2090–2103, 2020.
[50] J. So, B. Guler, and S. A. Avestimehr, "Turbo-aggregate: Breaking the quadratic aggregation barrier in secure federated learning," in "arXiv," 2020. [Online]. Available: https://arxiv.org/abs/2002.04156
[51] T. H. H. Chan, Shi, E., Song, D. , "Privacy-Preserving Stream Aggregation with Fault Tolerance," *Springer*, vol. 7397, Keromytis, A.D. (eds) Financial Cryptography and Data Security. FC 2012. Lecture Notes in Computer Science, 2012, doi: https://doi.org/10.1007/978-3-642-32946-3_15.
[52] A. Lapets, N. Volgushev, A. Bestavros, F. Jansen, and M. Varia, "Secure MPC for analytics as a web application," in *IEEE Secure Development Conference*, 2016, pp. 73–74.
[53] X. Yang *et al.*, "An accuracy-lossless perturbation method for defending privacy attacks in federated learning," in "arXiv," 2021. [Online]. Available: https://arxiv.org/abs/2107.04511
[54] P. Kumar, G. P. Gupta, and R. Tripathi, "PEFL: Deep privacy-encoding-based federated learning framework for smart agriculture," *IEEE Micro*, vol. 42, no. 1, pp. 33–40, 2022.
[55] J. Liao, Z. Chen, and E. G. Larsson, "Over-the-air federated learning with privacy protection via correlated additive perturbations," in *2022 58th Annual Allerton Conference on Communication, Control, and Computing*, 2022: IEEE, pp. 1–8.
[56] X. Yang and S. Ji, "Learning with multiplicative perturbations," in *2020 25th International Conference on Pattern Recognition*, 2021: IEEE, pp. 1321–1328.
[57] M. A. P. Chamikara, P. Bertok, I. Khalil, D. Liu, and S. Camtepe, "Privacy-preserving distributed machine learning with federated learning," *Computer Communications*, vol. 171, pp. 112–125, 2021.
[58] T. Liu, X. Hu, H. Xu, T. Shu, and D. N. Nguyen, "High-accuracy low-cost privacy-preserving federated learning in IoT systems via adaptive perturbation," *Journal of Information Security and Applications*, vol. 70, p. 103309, 2022.
[59] P. Kairouz, H. B. McMahan, B. Avent, A. Bellet, and M. Bennis, "Advances and open problems in federated learning," *Foundations and Trends in Machine Learning*, vol. 14, no. 1-2, pp. 1–210, 2021. [Online]. Available: https://arxiv.org/abs/1912.04977.
[60] A. Singh, P. Vepakomma, O. Gupta, and R. Raskar, "Detailed comparison of communication efficiency of split learning and federated learning," in "arXiv," 2019. [Online]. Available: https://arxiv.org/abs/1909.09145
[61] I. Ceballos *et al.*, "SplitNN-driven vertical partitioning," in "arXiv," 2020. [Online]. Available: https://arxiv.org/abs/2008.04137
[62] V. Sharma, P. Vepakomma, T. Swedish, K. Chang, J. Kalpathy-Cramer, and R. Raskar, "ExpertMatcher: Automating ML model selection for clients using hidden representations," in "arXiv," 2019. [Online]. Available: https://arxiv.org/abs/1910.03731
[63] P. Vepakomma, O. Singh, A. Gupta, and R. Raskar, "NoPeek: Information leakage reduction to share activations in distributed deep learning," in "arXiv," 2020. [Online]. Available: https://arxiv.org/abs/2008.09161
[64] H. B. McMahan and D. Ramage, "Federated learning: Collaborative machine learning without centralized training data," in *Google AI Blog*, ed, 2017.

[65] A. Koloskova, T. Lin, S. U. Stich, and M. Jaggi, "Decentralized deep learning with arbitrary communication compression," in *International Conference on Learning Representations*, 2020.
[66] L. G. a. Y. L. a. H. L. a. L. T. a. Z. Wang, "A review of privacy-preserving research on federated graph neural networks," *Neurocomputing*, vol. 600, p. 128166, 2024, doi: https://doi.org/10.1016/j.neucom.2024.128166.
[67] T. Xie, J. Zhang, Y. Zhang, C. Papamanthou, and D. Song, "Libra: Succinct zero-knowledge proofs with optimal prover computation," in *Proceedings of the 39th Annual International Cryptology Conference (CRYPTO)*, 2019, pp. 733–764.
[68] T. H. Rafi, F. A. Noor, T. Hussain, and D. K. Chae, "Fairness and privacy-preserving in federated learning: A survey," in "arXiv," 2023. [Online]. Available: https://arxiv.org/abs/2306.08402
[69] N. Carlini, F. Tramer, E. Wallace, M. Jagielski, and A. Herbert-Voss, "Extracting training data from large language models," in "arXiv," 2020. [Online]. Available: https://arxiv.org/abs/2012.07805
[70] S. Yeom, I. Giacomelli, M. Fredrikson, and S. Jha, "Privacy risk in machine learning: Analyzing the connection to overfitting," in *2018 IEEE 31st Computer Security Foundations Symposium (CSF)*, 2018: IEEE, pp. 268–282.
[71] V. Feldman, I. Mironov, K. Talwar, and A. Thakurta, "Privacy amplification by iteration," in *2018 IEEE 59th Annual Symposium on Foundations of Computer Science (FOCS)*, 2018: IEEE, pp. 521–532.
[72] D. Kifer and A. Machanavajjhala, "Pufferfish: A framework for mathematical privacy definitions," *ACM Transactions on Database Systems*, vol. 39, no. 1, pp. 3:1–3:36, 2014, doi: 10.1145/251468.
[73] S. Augenstein, H. B. McMahan, D. Ramage, S. Ramaswamy, and P. Kairouz, "Generative models for effective ML on private, decentralized datasets," in "arXiv," 2019. [Online]. Available: https://arxiv.org/abs/1911.06679