Bilecik Şeyh Edebali Üniversitesi Fen Bilimleri Dergisi
Bilecik Seyh Edebali University Journal of Science
e-ISSN: 2458-7575 (https://dergipark.org.tr/tr/pub/bseufbd)

Derleme Makalesi – Review Article

Geliş Tarihi / Received: 01/08/2025

Kabul Tarihi / Accepted: 17/11/2025

Yayın Tarihi / Published: 31/05/2026

Data Privacy in Machine Learning: Challenges and Federated Learning Solutions

Makine Öğreniminde Veri Gizliliği: Zorluklar ve Federe Öğrenme Çözümleri

Goksu Zekiye Ozen[1*], Yunus Ozen[2]

[1*]Kutahya University Department of Software Engineering, Kutahya, Turkey, goksu.ozen@dpu.edu.tr, https://orcid.org/0000-0001-7033-0126

[2]Yalova University Department of Computer Engineering, Yalova, Turkey, yunus.ozen@yalova.edu.tr, https://orcid.org/0000-0003-3225-8797

Ethical Statement: It is declared that scientific and ethical principles were followed during the preparation of this study and that all studies used are stated in the bibliography.

Artificial Intelligence Ethical Statement: The author declares that artificial intelligence was not utilized at any stage of the preparation process of this article and accepts full responsibility in this regard.

Conflicts of Interest: The author(s) has no conflict of interest to declare.

Grant Support: The author(s) acknowledge that they received no external funding to support this research.

Etik Beyan: Bu çalışmanın hazırlanma sürecinde bilimsel ve etik ilkelere uyulduğu ve yararlanılan tüm çalışmaların kaynakçada belirtildiği beyan olunur.

Yapay Zeka Etik Beyanı: Yazar bu makalenin hazırlanma sürecinin hiç bir aşamasında yapay zekadan faydalanılmadığını; bu konuda tüm sorumluluğun kendisine(kendilerine) ait olduğunu beyan etmektedir.

Çıkar Çatışması: Çıkar çatışması beyan edilmemiştir.

Finansman: Bu araştırmayı desteklemek için dış fon kullanılmamıştır.

Data Privacy in Machine Learning: Challenges and Federated Learning Solutions

ABSTRACT

Federated Learning (FL) enables collaborative model training across distributed devices while preserving data locality, making it a promising paradigm for privacy-sensitive applications. This paper presents a structured and comprehensive survey of FL studies with a focus on confidentiality and privacy-preserving mechanisms. This paper first reviews major FL architectures including centralized, decentralized, FedAvg, clustered, asynchronous, and heterogeneous approaches and provides a comparative discussion of their performance, scalability, and implementation complexity. Recent survey literature (2023-2025) is analyzed to highlight evolving challenges related to fairness, heterogeneity, and system-level security. Subsequently, key confidentiality methods are comparatively reviewed, including Differential Privacy (DP), Homomorphic Encryption (HE), Trusted Execution Environments (TEE), Secure Aggregation (SA), and Secure Multi-Party Computation (SMPC). Their relative trade-offs in computation cost, scalability, and protection strength are examined across diverse application domains, such as healthcare, finance, and IoT. The findings indicate that no single mechanism offers complete protection, and effective privacy assurance in FL requires hybrid approaches that balance efficiency with confidentiality. Finally, open research gaps and future directions are identified, emphasizing the need for adaptive, resource-aware, and trust-anchored FL frameworks capable of maintaining privacy guarantees under real-world heterogeneity and dynamic participation.

Keywords- Federated Learning, Data privacy, Machine Learning, Privacy-Preserving Techniques

Highlights

- Federated learning enables model training without centralizing sensitive data.
- Privacy concerns in traditional machine learning are addressed by decentralized learning.
- Key application domains of federated learning are systematically reviewed.
- Technical challenges such as communication cost and heterogeneity are discussed.
- Future research directions for improving federated learning in real-world settings are outlined.

Makine Öğreniminde Veri Gizliliği: Zorluklar ve Federe Öğrenme Çözümleri

ÖZ

Federe Öğrenme (FL), veri yerelliğini koruyarak dağıtık cihazlar arasında iş birliğine dayalı model eğitimi gerçekleştirmekte ve bu yönüyle gizlilik açısından hassas uygulamalar için umut verici bir paradigma sunmaktadır. Bu makale, gizlilik ve mahremiyet koruma mekanizmalarına odaklanan FL çalışmalarına ilişkin yapılandırılmış ve kapsamlı bir inceleme sunmaktadır. Çalışma, öncelikle merkezi, merkezi olmayan, FedAvg, kümelenmiş, asenkron ve heterojen yaklaşımlar dâhil olmak üzere temel FL mimarilerini gözden geçirmekte ve bunların performans, ölçeklenebilirlik ve uygulama karmaşıklığı açısından karşılaştırmalı bir değerlendirmesini yapmaktadır. 2023-2025 dönemine ait güncel literatür analiz edilerek adalet, heterojenlik ve sistem düzeyinde güvenliğe ilişkin gelişen zorluklar vurgulanmaktadır. Ardından, Diferansiyel Gizlilik (DP), Homomorfik Şifreleme (HE), Güvenilir Yürütme Ortamları (TEE), Güvenli Toplama (SA) ve Güvenli Çok Taraflı Hesaplama (SMPC) gibi temel gizlilik yöntemleri karşılaştırmalı olarak incelenmektedir. Bu yöntemlerin hesaplama maliyeti, ölçeklenebilirlik ve koruma gücü açısından göreceli avantaj ve sınırlılıkları, sağlık, finans ve IoT gibi çeşitli uygulama alanları kapsamında tartışılmaktadır. Bulgular, tek bir mekanizmanın tam koruma sağlamadığını ve FL'de etkili gizlilik güvencesinin, verimlilik ile gizlilik arasında denge kuran hibrit yaklaşımlar gerektirdiğini

ortaya koymaktadır. Son olarak, açık araştırma alanları ve gelecek yönelimler belirlenerek, gerçek dünya uygulamalarında heterojenlik ve dinamik katılım koşullarında gizlilik garantilerini sürdürebilen, uyarlanabilir, kaynak farkındalıklı ve güvene dayalı FL çerçevelerine duyulan ihtiyaç vurgulanmaktadır.

Anahtar Kelimeler- Federe Öğrenme, Veri Gizliliği, Makine Öğrenmesi, Veri Gizliliği Koruyucu Teknikler

Öne Çıkanlar

- Federe öğrenme, hassas verileri merkezileştirmeden model eğitimi sağlayarak veri gizliliğini korumaktadır.
- Geleneksel makine öğrenimindeki gizlilik sorunları, dağıtık yaklaşımla ele alınmıştır.
- Federe öğrenmenin farklı yöntemleri (centralized, decentralized, FedAvg, clustered, asynchronous, heterogeneous) sistematik olarak incelenmiş ve güçlü/zayıf yönleri ile uygulama alanları tartışılmıştır.
- İletişim maliyeti, heterojenlik ve cihaz çeşitliliği gibi teknik zorluklar değerlendirilmiş, çözüm yaklaşımları ele alınmıştır.
- Gerçek dünya uygulamaları için federe öğrenmede mevcut araştırma boşlukları belirlenmiş ve gelecek araştırma yönelimleri önerilmiştir.

I. INTRODUCTION

In today's society, Artificial Intelligence (AI) and Machine Learning (ML) technologies are rapidly making their way into every facet of our lives. These technologies offer undeniable benefits in diverse domains, ranging from healthcare to finance and mobile applications. However, as AI and ML rely heavily on large datasets, significant privacy concerns have emerged [1]. The widespread adoption of centralized machine learning has increased the vulnerability of personal data to security breaches, leading to unauthorized access, misuse of information, and compliance challenges with data protection regulations such as the General Data Protection Regulation (GDPR) [2, 3].

To address these pressing concerns, Federated Learning (FL) [4] has emerged as a promising paradigm that enables collaborative model training without requiring raw data transfer to a central server. In this setting, local devices or organizations train models using their own data, and only model updates (e.g., gradients or parameters) are shared. This approach mitigates the risks associated with centralized data collection and offers considerable advantages in sensitive areas where privacy is paramount.

Despite these advantages, FL faces multiple technical and practical challenges that may limit its effectiveness. Issues such as data heterogeneity, scalability, communication efficiency, and security threats (e.g., poisoning, backdoor, or inference attacks) are widely surveyed in the literature [5-19]. Accordingly, several defense mechanisms such as differential privacy, secure multi-party computation (SMPC), trusted execution environments, and secure aggregation have been proposed to mitigate these risks. While these methods provide varying levels of protection, they also introduce trade-offs in performance, implementation complexity, and computational cost.

As illustrated in Table 1, the surveyed papers collectively review the evolution of federated learning (FL) research over the past several years. The earliest surveys primarily focused on foundational aspects such as system architecture, regulatory compliance, aggregation efficiency, and robustness in large-scale deployments. Bonawitz et al. [5] analyzed system design for large-scale FL deployments, emphasizing device diversity and communication efficiency. Zhang et al. [6] offered one of the first comprehensive surveys of federated learning, addressing architectures, optimization strategies, and security vulnerabilities such as data leakage and poisoning, while summarizing defense mechanisms including differential privacy (DP) and SMPC. Truong et al. [7] examined FL from a regulatory perspective, particularly its compliance with GDPR, identifying privacy threats such as membership inference and reconstruction attacks and reviewing countermeasures like DP and SMPC. Kairouz et al. [8] provided a broad technical survey on scalability and robustness, proposing Byzantine-tolerant aggregation as a potential solution.

Subsequent reviews expanded their focus to the growing landscape of security and privacy challenges in FL. Ha and Dang [9] investigated inference attacks, particularly GAN-based threats, highlighting significant privacy leakage and outlining possible defenses against model inversion. Liu et al. [10] presented a focused review on privacy-preserving aggregation in FL, addressing aggregation leakage and collusion attacks through secure aggregation and homomorphic encryption. Zhang et al. [11] reviewed trustworthy FL frameworks, analyzing

adversarial and poisoning attacks while discussing defense strategies such as secure aggregation and anomaly detection.

More recent surveys (2023-2025) have increasingly concentrated on fairness, heterogeneity, and system-level security in complex FL environments. Rafi et al. [12] surveyed fairness- and privacy-preserving mechanisms, identifying issues such as bias and privacy leakage and summarizing approaches like DP, SMPC, and fairness-aware aggregation. Chhetri et al. [13] reviewed blockchain-based FL frameworks, emphasizing decentralized trust management, integrity assurance, and mitigation of malicious updates in collaborative learning. Hasan [14] provided a survey of key privacy and security threats in FL, including GAN-based and poisoning attacks, and discussed blockchain-enabled secure environments for enhanced system resilience. Zhao et al. [15] offered an extensive review of privacy challenges in FL, covering model inversion, poisoning, and adversarial manipulations, while evaluating defenses like gradient leakage prevention and anomaly detection. Li et al. [16] analyzed privacy-preserving techniques for edge IoT-based FL, highlighting the roles of DP, SMPC, and homomorphic encryption in protecting sensitive device data. Rahman [17] discussed privacy-preserving collaborative intelligence in FL, examining data leakage and model inversion risks and emphasizing its deployment within IoT ecosystems. Jimenez-Gutierrez et al. [18] compiled a comprehensive survey of FL security and privacy, reviewing adversarial and inversion attacks, defense frameworks, and open challenges related to scalable and robust deployment. Finally, Feng et al. [19] surveyed security threats such as backdoor, Byzantine, and adversarial attacks, summarizing corresponding defense mechanisms to improve robustness and trustworthiness.

Collectively, these survey papers demonstrate the evolving challenges and growing sophistication of privacy and security research in federated learning.

In light of these studies, this paper aims to provide a structured and comprehensive review of federated learning in the context of data privacy. The main contributions can be outlined as follows:

- A systematic survey of centralized, decentralized, FedAvg, clustered, asynchronous, and heterogeneous approaches, focusing on their strengths, limitations, and practical application domains.
- Identification of open research gaps and future directions, particularly for improving efficiency, robustness, and privacy in large-scale deployments.
- Highlighting the unique contribution of this study through a systematic categorization of FL methods combined with a critical evaluation of privacy-preserving techniques.

The remainder of this paper is organized as follows. Section II introduces FL and its principal methods. Section III provides a comparative analysis of these FL paradigms, evaluating their performance, computational cost, implementation complexity, and privacy-security trade-offs across different application domains. Section IV examines potential privacy breaches and adversarial threats in FL, along with the mechanisms designed to mitigate them. Section V concludes the paper by summarizing the main findings and highlighting the unique contributions of this review. Finally, Section VI presents future research directions and recommendations aimed at enhancing efficiency, robustness, and privacy in large-scale FL deployments.

Table 1. Comparative overview of federated learning survey papers.

Study	Focus / Scope	Main Privacy & Security Concerns	Proposed Solutions	Application Domain	Gaps / Limitations Addressed by This Work
Bonawitz et al. [5])	Large-scale system design.	Communication efficiency, device diversity	Efficient FL protocols, aggregation strategies.	Mobile edge computing, IoT.	Did not address privacy attacks or advanced security threats.
Zhang et al. [6]	Comprehensive survey on federated learning concepts, frameworks, and algorithms	Data leakage, communication overhead, model poisoning	DP, SMPC, model compression, communication optimization	General FL applications	Did not deeply explore heterogeneous clients or detailed comparisons of privacy-preserving mechanisms
Truong et al. [7]	GDPR compliance.	Membership inference, reconstruction attacks.	DP, SMPC	Healthcare, finance	Limited to regulatory compliance; broader system-level issues not addressed
Kairouz et al. [8]	Technical FL challenges.	Scalability, robustness, defective/malicious clients.	Byzantine-tolerant aggregation.	General FL deployments.	Did not deeply analyze advanced privacy attacks or practical deployment constraints.
Ha et al. [9]	Analyzed privacy leakage in FL through GAN-based inference attacks	Data poisoning, model inversion, backdoor attacks	Primarily attack analysis; only suggests differential privacy and gradient perturbation as potential defenses	Experimental FL environments	Did not consider heterogeneous clients, performance-privacy trade-offs, or large-scale multi-device FL environments
Liu et al [10]	Privacy-preserving aggregation in FL	Aggregation leakage, collusion	Secure aggregation, homomorphic encryption	Multi-party collaborative learning	Focused mainly on aggregation; broader attack vectors not considered
Zhang et al. [11]	Trustworthy FL: security, robustness, and privacy	Adversarial attacks, data isolation.	Secure aggregation, anomaly detection	Healthcare, finance.	Did not address scalability or heterogeneous client capabilities
Rafi et al. [12]	Fairness and privacy in FL	Bias, fairness, privacy leakage	Fair aggregation, DP, SMPC	IoT, mobile devices	Did not consider system heterogeneity or communication efficiency
Chhetri et al. [13]	Blockchain-based FL and data privacy	Data integrity, malicious updates	Blockchain-enabled FL, smart contracts	Decentralized FL networks	Did not explore communication efficiency or device-level constraints
Hasan [14]	Review of security & privacy issues in FL	Poisoning, backdoor, membership inference, GAN-based attacks	Blockchain, TEE, homomorphic encryption, secure aggregation, differential privacy	Various ML models in FL	No heterogeneous clients, no performance–privacy trade-offs, not multi-device
Zhao et al. [15]	Privacy attacks evaluation.	Model inversion / reverse-engineering of updates.	Countermeasures for gradient leakage.	Health, IoT	Focused only on attack evaluation; broader system and deployment challenges not covered.
Li et al.[16]	FL in edge IoT networks	Data privacy, model inversion	DP, SMPC, homomorphic encryption	Edge computing, IoT	Did not address fairness or robustness in heterogeneous networks
Rahman [17]	Privacy-preserving collaborative intelligence	Data leakage, model inversion	DP, SMPC, homomorphic encryption	General FL applications, including IoT and edge devices	Did not fully consider heterogeneous clients or performance–privacy trade-offs
Jimenez-Gutierrez et al. [18]	Security and privacy in FL	Adversarial attacks, model inversion	DP, secure aggregation, anomaly detection	General FL applications	Focused on attacks; broader system-level deployment not fully addressed
Feng et al. [19]	Security threats in FL	Backdoor, Byzantine, adversarial attacks	Multiple defense mechanisms	General FL applications	Focused on security threats; performance and fairness in heterogeneous clients not covered

II. FEDERATED LEARNING

FL is a machine-learning approach that enables the training of models using data stored locally on multiple devices or institutions instead of centralizing it on a single server. This approach allows for collaborative training of a model while keeping the sensitive local data secure. The general operation of FL [20] can be visualized in Figure 1 and can be summarized in the following steps:

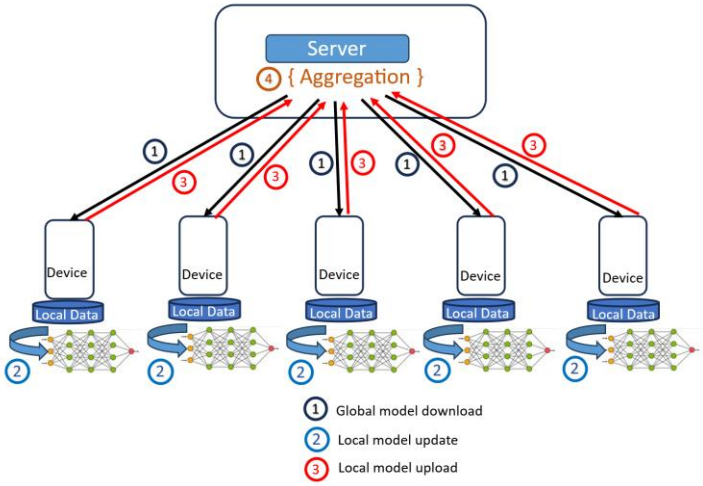

Figure 1. Illustration of the federated learning workflow.

- A FL system has multiple clients. These are usually devices or organizations. Each client has its local data. Model training is done without exporting the data from the device or the organization.

- At the start of the training process, a common global model is created by a central server. This global model is distributed to all participating clients. However, the data remains with the clients and is not sent outside.

- Each client trains the received global model with its local data. At the end of this process, each client updates the global model with its local data. That is, each device or organization improves the global model with its data.

- Clients do not share their local data. Instead, they send only certain information of the model updated with local data (for example, model parameters or gradients) back to the central server.

- The server collects and combines model parameters from clients. Using these parameters, the global model is updated. That is, each client's local training contributes to the global model.

- The updated global model is again distributed to clients, and the process continues as a new round. In each round, clients train the model once more with their local data and send it back to the server. This cycle continues until the model reaches the desired accuracy.

In FL systems, the interplay of heterogeneous data structures, varying device computational power, and diverse sectoral privacy standards plays a critical role in determining the most suitable learning algorithm and optimization strategy. Considering these dynamics, the commonly employed methods in FL include.

A. Centralized Federated learning (CeFL)

In centralized federated learning (CeFL), each client device trains the model on its local dataset and sends model updates such as weights or parameters to a central server. The server consolidates these updates and distributes the improved global model back to the clients [21]. This centralized consolidation enables large-scale coordination and simplifies system management, as the server functions as the main orchestrator of communication and aggregation. As Bonawitz et al. [5] highlight, such centralized protocols are well-suited for mobile edge computing and IoT environments due to their efficient communication strategies and adaptability to large-scale deployments. Moreover, the existence of a central aggregator often provides more stable convergence and faster training compared to decentralized setups, since the model updates are synchronized in a consistent manner.

However, this architecture also introduces critical vulnerabilities. The central server acts as a pivotal component of the system, and its failure can disrupt the entire learning process, posing a significant security threat [22]. In scenarios involving extensive datasets and numerous devices, the server may also become overwhelmed, causing degraded performance and delays. Additionally, when all updates are collected in one place, data privacy

risks increase. Zhao et al. [15] show that model updates can be exploited through inversion or reconstruction attacks, potentially exposing sensitive client data. Similarly, Hasan [14] points out that centralized systems are prone to poisoning and backdoor attacks unless combined with additional mechanisms such as blockchain-based secure FL. As noted in studies on CeFL, reliance on a single server can constrain system scalability and flexibility, as the computational and communication capacity of the server directly limits the number of participating clients and the volume of updates that can be efficiently processed simultaneously [5,6,8].

Despite these challenges, CeFL has been successfully deployed in real-world applications. For instance, Xu et al. [23] demonstrate its use in mobile applications like Google's Gboard, where millions of devices collaboratively train language models while keeping sensitive user data on local devices. In the healthcare domain, Truong et al. [7] discuss how CeFL supports GDPR-compliant collaboration across hospitals, enabling model training without direct sharing of raw patient data. Finally, Bonawitz et al. [5] emphasize the suitability of CeFL for large-scale deployments, including mobile edge and IoT environments, by addressing communication efficiency and device heterogeneity.

B. Decentralized Federated Learning (DFL)

Decentralized federated learning (DFL) operates without reliance on a central server, adopting a peer-to-peer architecture in which client devices directly exchange and merge model updates [24]. This structure eliminates the single point of failure inherent in centralized systems, ensuring that the network remains operational even if individual nodes fail. The decentralized framework also facilitates scalability, allowing new devices to join the network dynamically and enabling the collaborative training of larger and more heterogeneous models [25].

Despite these advantages, DFL introduces several technical challenges. Continuous device-to-device communication increases bandwidth consumption and latency, particularly in environments with limited connectivity [26]. Direct peer-to-peer interactions also expose the system to security threats, such as malicious participants manipulating model updates or intercepting communication channels [14, 26]. Moreover, heterogeneity in device hardware, software, and local data distributions can lead to imbalanced learning, potentially affecting convergence speed and overall model performance [12, 16].

Several studies have examined these benefits and limitations in detail. Surveys on DFL highlight that removing the central server can reduce communication bottlenecks and improve resilience but also emphasize unresolved challenges in communication protocols and trust management among clients [13, 17]. Techniques such as segmented gossip or neighborhood-based topologies have been proposed to optimize information propagation while maintaining robustness, yet these methods remain sensitive to device heterogeneity and network instability [18].

In practice, DFL is particularly relevant in scenarios where centralized aggregation is infeasible or undesirable. For example, IoT networks with geographically distributed sensors can collaboratively train models without routing all updates through a central server, thereby reducing communication overhead and improving fault tolerance [12]. In domains such as healthcare or cross-border collaborations, DFL facilitates direct exchange of model parameters between institutions while ensuring compliance with data privacy regulations such as GDPR, as discussed by Truong et al. [7]. Similarly, Zhang et al. [11] emphasize that decentralized coordination can enhance trust and robustness in federated settings by reducing reliance on a single central aggregator. Edge computing applications also benefit from this paradigm, particularly when central coordination is too costly or slow, making peer-to-peer model sharing more efficient and adaptive.

Nonetheless, several research gaps remain. Communication efficiency and optimal synchronization strategies in DFL are still under active investigation [5]. Security and privacy threats, including poisoning, adversarial participants, and gradient manipulation, require stronger mitigation mechanisms [14, 26]. Finally, effectively managing heterogeneous clients to ensure fair contribution and consistent model quality remains a key open problem [12, 16]. Addressing these challenges aligns with the objectives of this study to systematically evaluate FL methods and identify future research directions for robust, privacy-preserving DFL.

C. Federated Averaging (FedAvg)

Federated Averaging (FedAvg) is one of the most widely adopted algorithms in federated learning due to its simplicity and scalability. In this approach, each client device trains a local model on its private dataset for a predefined number of iterations before transmitting the updated model parameters to a central server [27]. The server then performs a weighted averaging of these parameters to produce an updated global model, which is subsequently redistributed to all participating clients. This iterative process continues until the global model reaches a satisfactory level of convergence [28].

The primary advantage of FedAvg lies in its ease of implementation and compatibility with large-scale systems, making it suitable for environments with a high number of heterogeneous devices [8]. By allowing

multiple clients to train concurrently and asynchronously, FedAvg can achieve substantial improvements in computational efficiency and reduce the communication overhead compared to approaches that require synchronized global updates [17]. Moreover, its straightforward aggregation mechanism facilitates integration with privacy-preserving techniques such as secure multiparty computation or differential privacy, which have been explored in healthcare, finance, and IoT applications [7, 17].

Despite these strengths, FedAvg has notable limitations. Performance can degrade when local datasets are not independent and identically distributed (non-IID) data, leading to unbalanced or biased global models [29]. Significant variability in local data distributions across devices may result in slower convergence, reduced accuracy, or unfair representation of certain client subgroups [12, 16]. Additionally, FedAvg is vulnerable to adversarial attacks, including model poisoning and gradient manipulation, particularly in large-scale deployments with potentially untrusted clients [14, 18].

In terms of applications, FedAvg has been widely utilized in mobile and edge computing scenarios, such as predictive text systems where millions of devices collaboratively improve language models without sharing sensitive user data [5]. It is also applied in healthcare, where hospitals jointly train diagnostic models while preserving patient privacy [7], and in IoT networks, where edge devices collaboratively improve anomaly detection models without centralizing raw sensor data [12, 16]. These practical deployments demonstrate that FedAvg effectively balances scalability, simplicity, and privacy, although careful consideration of data heterogeneity and client trustworthiness remains essential.

D. Clustered Federated Learning (CluFL)

Clustered Federated Learning (CluFL) is designed to tackle the challenges posed by heterogeneous data in federated learning by partitioning clients into clusters according to the similarity of their local data distributions [30]. Unlike conventional FL, which relies on a single global model for all devices, Clustered FL enables each cluster to train its own model, tailored to its specific data characteristics [31].

One of the main advantages of CluFL is its ability to mitigate the negative effects of non-IID data. By grouping devices with similar data distributions, CluFL improves model accuracy and convergence rates, allowing the system to achieve better generalization across heterogeneous clients [30]. This approach also reduces communication overhead relative to fully centralized FL, as model updates can be aggregated within clusters before being merged at higher levels [31]. Furthermore, CluFL facilitates personalized model training in privacy-sensitive domains such as healthcare and finance, where institutions require collaborative learning without sharing raw data. Moreover, in privacy-sensitive domains like healthcare and finance, clustered FL frameworks such as PCBFL support collaborative learning without sharing raw patient or customer data [32, 33].

Despite these benefits, CluFL presents several limitations and challenges. The clustering process itself is computationally intensive and requires sophisticated methods to assign clients accurately based on their data similarity. Aggregating models across clusters can also be complex, since naive averaging may dilute the cluster-specific knowledge that provides CluFL its advantages [31]. Device heterogeneity-including differences in computational power, network connectivity, and storage-can further complicate cluster management and training efficiency. Additionally, as with other FL approaches, CluFL remains vulnerable to adversarial attacks and unreliable clients, highlighting the need for robust defense mechanisms [15, 34].

In terms of application scenarios, CluFL is particularly suited for environments where client data is highly heterogeneous and sensitive. Examples include cross-hospital collaborations, multi-institution financial modeling, and geographically distributed IoT networks. In these contexts, CluFL enables privacy-preserving, cluster-specific model training, achieving better performance than conventional FL while maintaining data confidentiality [30, 31]. Techniques such as FedClust [30] and cluster-based aggregation frameworks [31] demonstrate that CluFL can effectively handle non-IID distributions, providing more accurate and robust models in practice. In addition, alternative methods such as FedRDS [29] address similar heterogeneity challenges through regularization and limited data sharing, offering complementary strategies to clustering-based approaches.

In summary, CluFL offers substantial advantages in terms of performance and generalization under heterogeneous data conditions, but it introduces complexities and limitations in clustering, aggregation, and client heterogeneity. Its applicability in privacy-sensitive and distributed domains underscores its practical relevance, while ongoing research aims to address communication efficiency, security, and fairness across clusters.

E. Asynchronous Federated Learning (AFL)

Asynchronous Federated Learning (AFL) allows devices to update the global model independently, without waiting for all other clients to synchronize. In this framework, each device sends its local model updates to the central server as soon as they become available, which is particularly advantageous for clients with intermittent network connectivity or limited computational resources [35]. By removing the requirement for

synchronous updates, AFL reduces idle times for faster devices and enables more flexible participation across heterogeneous edge clients, thereby improving the overall efficiency and scalability of federated learning systems [36].

One of the key advantages of AFL is its ability to accommodate diverse client capabilities and network conditions, making it well-suited for dynamic environments such as mobile edge computing, IoT networks, and distributed healthcare systems [36]. As updates are processed asynchronously, devices are not constrained by the slowest clients, which can accelerate convergence compared to fully synchronous FL approaches [37]. Furthermore, asynchronous aggregation can be combined with privacy-preserving techniques such as differential privacy and secure multiparty computation, maintaining the confidentiality of sensitive local data while allowing large-scale collaborative training [38]. Recent studies also introduce semi-asynchronous frameworks that strike a balance between synchronization and flexibility to enhance robustness against straggler clients and fluctuating client availability, which is especially beneficial in dynamic infrastructures such as drone-based networks or edge–IoT systems [39].

However, AFL also introduces limitations and challenges. Because updates are applied as they arrive, some local models may be stale, potentially slowing convergence or affecting global model accuracy [36]. Aggregation strategies must account for these asynchronous contributions to prevent bias or instability in the global model [37]. Additionally, asynchronous operations can increase the complexity of system management, requiring mechanisms to handle device failures, malicious updates, and heterogeneous client behavior. Security vulnerabilities such as poisoning attacks or gradient manipulation remain a concern, necessitating robust defenses within the AFL framework [40].

In terms of application scenarios, AFL has been effectively applied in domains with heterogeneous and sensitive data. For instance, in edge IoT networks, devices with varying connectivity and computational capabilities can collaboratively train models such as anomaly detection systems without waiting for all participants to synchronize, thereby improving responsiveness and scalability [40]. Similarly, in sectors like healthcare and finance, AFL enables multiple institutions to jointly train predictive or diagnostic models while complying with strict privacy and data-sharing regulations, accommodating differences in data volume, access frequency, and processing speed across organizations [41]. Semi-asynchronous extensions have also been explored for drone networks and other time-critical applications, where resilience to stragglers and delayed updates is essential for maintaining robust and accurate global models [42].

Overall, AFL provides a flexible and scalable alternative to synchronous FL, particularly in environments with heterogeneous clients and variable network conditions. While it improves efficiency and supports privacy-preserving collaborative learning, careful design of aggregation mechanisms and robust defense strategies is required to address challenges related to staleness, model accuracy, and adversarial threats [36, 40].

F. Heterogeneous Federated Learning (HFL)

Heterogeneous Federated Learning (HFL) is designed for environments where client devices exhibit diverse computational capabilities, network conditions, and data characteristics, necessitating customized learning strategies for optimal performance [43]. In this paradigm, the central server coordinates or assists in the generation of personalized models tailored to each client, while clients perform local updates based on their unique data and hardware constraints [44].

By aligning model architectures and training processes with device-specific attributes, HFL significantly enhances learning efficiency and predictive accuracy compared to one-size-fits-all global models. The advantages of this approach are evident in its ability to accommodate device diversity and heterogeneous datasets [45]. Personalized models reduce the risk of performance degradation caused by non-IID data distributions and enable resource-constrained devices to participate more effectively without being overburdened by models tailored for high-capacity clients [46]. Moreover, since local data never leaves the device, such approaches maintain data privacy while contributing to a federated intelligence framework [43].

Despite these benefits, HFL introduces several complexities and limitations. Coordinating model updates from diverse clients and merging them into a coherent global system is challenging, requiring sophisticated aggregation mechanisms and careful synchronization to prevent bias or instability [43, 47]. Additionally, ensuring fairness among clients with varying computational power and data quality remains an open research problem, as devices with limited resources or noisy data may contribute less to the global model unless explicitly compensated through adaptive weighting or fairness-aware strategies [48].

In terms of application scenarios, HFL has proven particularly relevant in multi-device IoT networks, edge computing, personalized healthcare analytics, and financial modeling across institutions with varying computational infrastructure [41, 43]. By providing customized and resource-adaptive models, HFL maximizes the utility of each device's contribution while maintaining both privacy and training efficiency [46, 47]. Its

importance continues to grow as federated learning evolves toward large-scale, real-world deployments involving highly diverse clients, where uniform global models often underperform due to differences in data, computation, and connectivity [43].

In summary, HFL enables personalized and device-specific model training, improving both performance and efficiency in diverse client environments [43, 46]. While offering significant advantages in addressing system and data heterogeneity as well as preserving privacy, HFL requires careful design of aggregation mechanisms and fairness-aware strategies to manage system complexity and ensure equitable contribution across all participating devices [47, 48].

III. DISCUSSION AND COMPARATIVE REVIEW OF FL METHODS

Following the detailed discussion of individual FL methods, Table 2 presents a comparative overview of their performance, application domains, implementation complexity, computational cost, and privacy-security characteristics.

CeFL demonstrates high performance owing to synchronized updates coordinated by a central server, which enables rapid convergence and stable model aggregation. Compared with decentralized schemes, CeFL provides stronger consistency and easier orchestration but at the cost of higher dependency on a single node. This central reliance introduces potential bottlenecks and privacy vulnerabilities, though its stable coordination makes it well suited to domains demanding reliability-such as healthcare, finance, and mobile applications.

In contrast, DFL eliminates the central aggregator, enhancing privacy and resilience against single point of failures. However, this improvement comes with trade-offs: communication overhead and coordination costs rise sharply as nodes must exchange updates directly with peers. Relative to CeFL, DFL offers greater autonomy but lower scalability, particularly in IoT or smart-city environments where heterogeneous connectivity and dynamic participation impede synchronization.

The FedAvg algorithm balances efficiency and simplicity, achieving strong scalability through lightweight aggregation. Compared with DFL, it reduces communication overhead via centralized coordination yet still faces information leakage risks from gradient sharing in untrusted settings. Its wide adoption in mobile and healthcare systems reflects an effective compromise between ease of deployment and moderate privacy protection.

CluFL extends FedAvg by grouping clients with similar data distributions, thereby improving accuracy and adaptability under non-IID conditions. Unlike FedAvg's uniform model aggregation, CluFL's intra-cluster training increases computational complexity and latency. Thus, while CluFL achieves higher model performance on heterogeneous data, it demands more coordination effort and resource allocation.

AFL relaxes synchronization constraints, enabling clients with intermittent connectivity to contribute updates independently. This improves scalability and device participation compared with synchronous schemes such as CeFL or FedAvg. Yet, AFL's flexibility introduces temporal inconsistency-stale updates can degrade model accuracy if not properly controlled-making it suitable primarily for dynamic IoT and edge deployments rather than latency-sensitive tasks.

Finally, HFL adapts architectures and aggregation rules to device-specific capabilities and data diversity. In comparison with the more uniform frameworks above, HFL prioritizes fairness and personalized optimization at the expense of higher implementation complexity. Its adaptive aggregation enhances efficiency across diverse participants, though the required alignment mechanisms and fairness control significantly increase orchestration overhead.

In practice, these paradigms are frequently combined into hybrid frameworks to leverage complementary strengths. For example, cluster-based FL can be paired with FedAvg within clusters to handle heterogeneity efficiently, while asynchronous updates can be integrated into HFL to accommodate variability in compute speed and connectivity. Such hybrids are particularly effective in IoT, mobile, and healthcare applications, where balancing device diversity and operational variability is essential for robust and scalable FL systems.

IV. POTENTIAL PRIVACY BREACHES IN FL

As discussed in Section II, FL enables collaborative model training across multiple clients without transferring raw data to a central server, offering a substantial privacy advantage over traditional centralized approaches. Nevertheless, the absence of direct data sharing does not fully eliminate privacy vulnerabilities. Privacy threats in FL are generally categorized into passive attacks, in which adversaries infer private information by analyzing shared parameters, and active attacks, in which adversaries deliberately manipulate model updates to extract or corrupt data [1].

In passive attacks, adversaries do not directly interfere with the training process but instead exploit exchanged information-such as model weights and gradients-to infer private data. A malicious central server or a compromised client can reconstruct sensitive user information from these updates, a process known as a reconstruction attack [49].

Conversely, active attacks involve direct manipulation of the federated learning process. Adversaries may inject poisoned updates, tamper with model parameters, or introduce malicious gradients to degrade model performance, bias learning outcomes, or expose confidential client information. Such model poisoning attacks can critically undermine the integrity and trustworthiness of the federated system [50].

To mitigate privacy vulnerabilities in FL, a diverse set of defensive strategies has been proposed in recent studies. These approaches aim to enhance the confidentiality and resilience of FL systems by reducing the risk of sensitive information leakage or manipulation during model training and communication. Instead of relying solely on data isolation, contemporary solutions emphasize securing model updates, aggregation processes, and local computations. Such mechanisms not only mitigate both passive and active attacks but also strengthen trust, transparency, and accountability among distributed participants. The following subsections detail the key methods developed to ensure data confidentiality and system integrity within FL environments [5, 8, 50].

A. Differential Privacy (DP)

DP has emerged as one of the most widely adopted methods to safeguard individual data in FL. By adding calibrated random noise to model updates before they are shared with the central server, DP ensures that the contribution of any single user cannot be inferred, thereby preventing privacy breaches such as membership inference or reconstruction attacks [51]. This mechanism provides formal and quantifiable privacy guarantees, expressed by the parameters (ϵ, δ), which define the balance between privacy protection and model utility. Because the noise is introduced locally on each client, DP integrates seamlessly into existing FL architectures and can scale efficiently across large, distributed networks [51, 52].

The principal advantage of DP lies in its mathematically grounded privacy guarantees and its compatibility with diverse FL applications. DP is particularly valuable in sensitive domains such as healthcare, finance, and personalized recommendation systems, where data confidentiality and regulatory compliance (e.g., GDPR, HIPAA) are paramount [53, 54].

However, DP also introduces several limitations. The added noise, while crucial for privacy, can degrade model accuracy, particularly in cases of small or highly imbalanced datasets or when strong privacy guarantees (small ϵ values) are required [51]. Additionally, determining optimal privacy budgets remains challenging, as excessive noise may hinder convergence, whereas insufficient noise may expose private data. This trade-off becomes even more complex in heterogeneous or resource-constrained FL settings. Consequently, DP is often combined with complementary techniques such as secure aggregation or homomorphic encryption to achieve a balanced trade-off between privacy and model performance [51, 54].

B. Homomorphic Encryption (HE)

Homomorphic Encryption (HE) enables arithmetic over ciphertexts, allowing the server to aggregate client updates without ever decrypting them. In a typical HE-enabled FL pipeline, each client encrypts its local model update and transmits the ciphertext to the server; the server then performs homomorphic aggregation (e.g., summation/averaging under an approximate HE scheme) and broadcasts the updated global model-never accessing plaintext gradients or parameters [52]. By ensuring that computation occurs entirely in the encrypted domain, HE preserves the confidentiality of sensitive information throughout the training lifecycle.

A principal advantage of HE is its end-to-end confidentiality. Because the server manipulates only ciphertexts, even a compromised or honest-but-curious aggregator learns no client-level plaintext updates. This property reduces reliance on fully trusted aggregators and facilitates collaboration among participants with imperfect trust relationships (e.g., cross-institutional settings). Accordingly, HE has been widely considered for high-sensitivity domains including healthcare, finance, and industrial IoT where disclosure risks carry significant legal and ethical implications [52-54].

These privacy gains come with non-trivial costs. Homomorphic operations are substantially more expensive than their plaintext counterparts, and ciphertext expansion increases communication load, potentially stressing bandwidth-constrained environments. Moreover, correct parameterization (e.g., polynomial modulus degree, ciphertext modulus chain, multiplicative depth/precision in CKKS) requires expertise to balance privacy, numerical accuracy, and computational efficiency-challenges that are amplified in large-scale or resource-constrained FL deployments [52].

Recent work proposes mitigations. FedSHE introduces adaptive segmented CKKS, partitioning encrypted payloads to lower computational and communication overhead while maintaining strong privacy guarantees in

federated settings [52]. Beyond such optimizations, hybrid designs can further improve practicality, for example, combining HE with secure aggregation to shrink trust assumptions and reduce plaintext exposure during aggregation, or leveraging trusted execution environments (TEE) to offload parts of the computation while preserving confidentiality constraints [54, 55]. In practice, these hybrids allow system designers to tune the privacy-utility-efficiency trade-off to application needs.

C. Trusted Execution Environment (TEE)

TEE provides a hardware-backed isolated context in which code and data can be processed with confidentiality and integrity guarantees, even in the presence of a privileged but honest-but-curious system operator. TEE can execute sensitive stages-such as gradient aggregation, secure preprocessing, or even full model training inside a secure enclave, preventing the hosting server (and other processes on the same machine) from directly accessing plaintext model updates or intermediate values. This hardware-software co-design is particularly attractive when stringent privacy and security requirements preclude exposing data in use [54].

The principal advantage of TEE is strong in-use data protection: computations and keys confined to the enclave are isolated from the rest of the system, supporting confidentiality and integrity during training/aggregation. TEE also enables remote attestation, allowing clients to verify enclave identity and code load before releasing updates, which is valuable in cross-institutional deployments (e.g., healthcare, finance, industrial IoT) where participants may not fully trust the aggregator. Moreover, TEE can complement cryptographic techniques, for example, running lightweight secure aggregation inside the enclave or selectively combining HE outside/inside the enclave-to reduce plaintext exposure while improving efficiency relative to purely cryptographic designs [53, 54] (cf. HE as a complementary technique [52]).

These benefits come with limitations. First, hardware availability and cost constrain deployment; not all devices or clouds expose a production-grade TEE. Second, systems integration is non-trivial: enclave memory size, I/O boundaries, and enclave transitions can introduce engineering complexity and performance overheads. Third, while TEE protects data in use, it does not inherently prevent model-level attacks (e.g., model poisoning/backdoors) and must be combined with robust aggregation or anomaly detection to address malicious updates. Finally, side-channel vulnerabilities (e.g., cache/timing/speculative-execution leaks) have been demonstrated against several TEE implementations; without careful hardening, these channels can undermine enclave confidentiality despite the isolation boundary [49, 50].

TEE-enabled FL has been demonstrated in security-critical scenarios such as confidential medical analytics, financial transaction modeling, and multi-institution collaborative training showing that enclave-based aggregation and preprocessing can increase trust and resilience while keeping operational complexity within acceptable bounds for many deployments [54].

D. Secure Aggregation (SA)

Secure aggregation (SA) is a cryptographic protocol that protects the confidentiality of client updates during FL by ensuring that the server learns only an aggregate (e.g., a sum) of client contributions and never any individual update in plaintext. Concretely, clients mask or encrypt their local model updates and transmit the protected values to the server; the protocol guarantees that only the aggregated update is revealed once sufficient clients participate, while individual updates remain hidden even from an honest-but-curious server [55]. This provides strong protection against passive adversaries attempting to infer private information from per-client gradients.

The main advantage of SA is that it preserves update confidentiality while retaining the ability to learn an accurate global model: the server learns only an aggregate (e.g., a sum) of client contributions and never an individual update in plaintext. Because only aggregates are revealed, SA fosters trust among participants and can be combined with complementary privacy mechanisms (e.g., differential privacy, homomorphic encryption, or TEEs) when additional protections are required. SA has also been deployed at scale in commercial FL systems, for example, in Google's mobile keyboard application-where user typing signals are aggregated while protecting individual contributions. [5, 55, 56]

SA introduces limitations. First, masking/cryptographic operations and coordination of keys or masks introduce computational and communication overhead, which becomes salient as the number of clients grows. Second, many SA protocols require some synchronization (e.g., a minimum quorum per round) and explicit dropout handling, which complicate cross-device deployments with churn and heterogeneous resources. Although modern SA designs incorporate dropout resilience, client failures can still disrupt progress without fault-tolerant orchestration. Finally, SA protects confidentiality during aggregation but does not by itself mitigate active threats such as model poisoning or backdoor attacks; it should be paired with robust aggregation or anomaly detection for comprehensive security. [5, 50, 55]

SA is a cornerstone primitive for privacy-preserving FL: it hides individual updates, supports accurate global learning, and scales to production. Ongoing work on more efficient and dropout-tolerant variants continues to improve practicality in real-world systems, especially when combined with DP, HE, or TEE to address distinct facets of the privacy-security-utility trade-off [5, 55, 56]

E. Protection Against Model Poisoning and Backdoor Attacks

FL is vulnerable to model poisoning and backdoor attacks because it aggregates updates from potentially untrusted clients. In poisoning, adversaries craft local updates to degrade accuracy or bias predictions; in backdoors, they implant a trigger so that specific inputs elicit attacker-chosen outputs while the model appears benign on clean data [50, 61, 62]. These threats undermine reliability, fairness, and integrity, making principled defenses essential.

To mitigate these risks, the literature proposes robust aggregation and security auditing. Robust aggregation replaces simple averaging with Byzantine-resilient estimators [58], coordinate-wise median/trimmed mean [59], and that limit the influence of corrupted updates [60]. Auditing layers include gradient clipping, anomaly/outlier detection, and reputation-/history-based filtering that down-weight or exclude suspicious clients [50, 63]. Empirical studies show that adaptive local-model poisoning can bypass a single defensive layer unless auditing and aggregation are jointly tuned [61]. In peer-to-peer DeFL, the lack of a central curator further elevates backdoor risk and motivates continuous auditing/topology-aware defenses [62]. Trust-aware schemes (e.g., FLTrust, or reputation-driven filters) leverage historical behavior or small trusted seeds to improve resilience in open settings [63].

These mechanisms enhance resilience without accessing raw data, preserving privacy while protecting model integrity; they can also be composed with orthogonal privacy tools-such as DP and SA-to form hybrid defenses addressing distinct risks (statistical disclosure vs. update confidentiality) [50, 64]. Nonetheless, defenses introduce computational and communication overheads and often require synchronization or quorum assumptions that are costly under client churn or asynchronous participation. Robust rules may yield false positives, discarding benign yet atypical updates and harming accuracy; many methods also assume a majority of honest clients, an assumption that may fail in open or peer-to-peer regimes [50, 62, 63]. Despite these trade-offs, protection against poisoning and backdoors remains indispensable for safety-critical applications (e.g., autonomous driving, healthcare, finance), and current studies continue on adaptive and explainable defenses that track evolving attack patterns [50, 61, 62].

F. Comparative Review of Confidentiality Methods

Following the detailed examination of individual privacy-preserving mechanisms, Table 3 presents a comparative overview of their core characteristics, including data privacy level, computational cost, ease of implementation, impact on performance, and overall security. This comparative analysis synthesizes the essential insights from the preceding discussion, enabling a clearer understanding of how these techniques differ in practical deployment. By contrasting their relative strengths, weaknesses, and trade-offs, the table highlights that no single method is universally optimal; rather, the selection of an appropriate confidentiality-preserving technique in FL depends on the specific application context, resource constraints, and required level of privacy protection.

As shown in Table 3, each method offers distinct advantages tailored to specific operational and security requirements. DP stands out for its ease of implementation and low computational overhead, making it suitable for large-scale and resource-constrained FL systems. However, the random noise added to preserve privacy can reduce model accuracy, particularly in sensitive domains where even minor performance degradation is undesirable. In what follows, we discuss each technique individually, detailing practical trade-offs and recommended deployment scenarios.

HE provides a high level of server-side data confidentiality by allowing computations directly on encrypted data, thereby avoiding plaintext exposure to the aggregator. This benefit comes with substantial computational complexity and latency due to heavy cryptographic operations. Compared with TEE and SA, HE achieves the strongest confidentiality guarantees but introduces the highest computational cost and latency, making it suitable mainly for scenarios where maximum security outweighs efficiency concerns.

TEE bridges hardware and software security by ensuring that computations occur within hardware-backed isolated enclaves. In contrast to HE, TEE provides in-use protection without full cryptographic processing, thereby reducing encryption overhead but depending heavily on hardware availability and secure configuration. Compared with SA, TEE offers stronger integrity assurances and lower computation overhead but faces scalability limitations and higher integration complexity, particularly in heterogeneous environments.

SA techniques ensure that individual client updates remain undisclosed while still enabling collective model training. Unlike HE, SA avoids expensive ciphertext computation, resulting in higher scalability and lower latency, though with weaker protection during computation. Compared with TEE, SA achieves broader applicability and easier deployment but offers less comprehensive integrity guarantees. It therefore provides a balanced trade-off

between confidentiality and efficiency, incurring moderate coordination overhead and synchronization challenges in asynchronous or cross-device settings.

Finally, defenses against model poisoning and backdoor attacks primarily target the integrity and reliability of the global model rather than direct data privacy. Although these defenses enhance robustness by detecting and mitigating malicious updates, they typically impose additional auditing and verification costs, and their success may depend on accurate anomaly detection and honest-majority assumptions among participants.

No single FL method simultaneously satisfies privacy, efficiency, and scalability. Lightweight techniques such as DP and SA complement more resource-intensive solutions like HE and TEE, while poisoning/backdoor defenses ensure model integrity. These observations motivate the adoption of hybrid approaches to achieve a balanced trade-off among privacy, performance, and robustness in practical federated learning scenarios.

Table 2. Comparison of federated learning methods

Method	Performance	Application Area	Implementation Difficulty	Calculation Cost	Privacy	Security
Centralized FL	High; efficient update with centralized server.	Health, finance, mobile apps.	**Medium**; requires centralized server management.	**Medium**; centralized server resource utilization.	**Medium**; central server risk**.**	**Medium**; the central server can be the target.
Decentralized FL	**Medium;** communication can affect costs**.**	IoT, smart city applications, edge computing**.**	**High**; inter-device management is complex.	**High**; requires constant communication.	**High**; user data remains on the device**.**	**Medium**; communication security risks.
FedAvg	**High;** works well with large datasets**.**	General machine learning applications.	**Easy**; common and well documented.	**Low**; each device operates on its own dataset.	**Medium**; model updates are sent; some information may leak.	**Medium**; updates can be targeted.
Cluster-based FL	**High**; good performance on heterogeneous data distribution.	Fields with heterogeneous data structure.	**Difficult**; clustering and model merging processes are complex.	**Medium**; clustering and model merging can be complex**.**	**High;** inter-cluster data security is ensured.	**High**; model safety is increased in different clusters**.**
Asynchronous FL	**Medium**; lack of synchronization may affect performance.	Applications with poor connection quality.	**Easy**; allows flexible operation even with weak connections.	**Medium**; no need to wait for updates.	**Medium**; updates are safe but there are risks.	**Medium**; lack of synchronization can increase security risks.
Heterogeneous FL	**High**; models that match the specifications of the devices.	Scenarios where different types of devices are used together.	**Difficult**; managing multiple devices and models is complex.	**High**; managing different models is complex.	**High**; privacy is ensured with customized models.	**High;** protection methods tailored to the characteristics of each device.

Table 3. Federated learning comparison table of methods used to ensure confidentiality.

Method	Data Privacy Level	Calculation Cost	Ease of Implementation	Impact on Performance	Security
DP	**High**; makes it difficult to identify individuals.	**Low**; easy to add noise.	**Easy** to integrate into existing systems.	**Limited**; added noise may affect model accuracy	**High**; however, some data can be back extracted.
HE	**Highest**; data is processed encrypted**.**	**High;** encrypted transactions take time**.**	**Difficult**; requires technical knowledge	**Important**; encrypted operations may degrade performance.	**Highest**; data is never traded unencrypted
TEE	**High;** (in-use) via hardware-backed isolation**.**	**Low**; there are additional hardware costs.	**Difficult**; requires hardware that supports TEE.	Typically near-native; overhead varies with hardware and enclave integration	**High**; provides software and hardware security.
SA	**High**; for per-client updates (server sees only aggregate)	**Medium**; requires encryption and concatenation operations.	**Medium**; requires secure communication protocols.	**Medium**; Typically neutral on accuracy, communication costs may increase.	**High;** no access to the individual content of the data**.**
Protection Against Model Poisoning and Backdoor Attacks	**Low / N/A** integrity-focused, not privacy.	**Medium**; security audits may incur additional costs.	**Medium;** requires audit processes.	**Variable;** audits/robust rules can reduce accuracy (false positives) and add overhead.	**High**; provides resilience against malicious attacks**.**

V. CONCLUSION

FL reduces the privacy risks of centralized data pooling by enabling on-device training, yet it remains vulnerable to passive leakage (membership inference, reconstruction) and active manipulation (poisoning, backdoors). This necessitates complementary privacy and security layers in FL pipelines.

Across system designs-CeFL, DeFL, FedAvg, CluFL, AFL, HFL-there are clear trade-offs in performance, complexity, privacy posture, and scalability. CeFL can be efficient but concentrates risk; DeFL and HFL improve adaptability at higher protocol and engineering cost. CluFL and AFL help with heterogeneity and connectivity but introduce coordination overhead and potential staleness. In practice, hybrid compositions are needed to balance these factors for concrete deployments and regulatory contexts.

Privacy/robustness mechanisms offer complementary strengths. DP and SA are comparatively lightweight and broadly applicable though DP entails an accuracy-privacy trade-off and SA does not ensure integrity by itself. HE and TEE provide stronger confidentiality (server-side and in-use, respectively) but incur nontrivial computational or deployment costs and likewise do not neutralize model-level attacks alone. Robust aggregation and auditing remain essential for integrity. Effective FL therefore layers these tools to cover statistical disclosure, update confidentiality, in-use protection, and adversarial robustness while preserving utility.

Despite progress, important gaps persist: navigating efficiency-privacy and accuracy-privacy trade-offs at scale; fault-tolerant orchestration under heterogeneity, churn, and partial participation; and understanding the cumulative effects of hybrid privacy/robustness stacks in resource-constrained settings. Further gaps include benchmarking and evaluation (multi-metric reporting across privacy-utility-cost-scalability), reproducibility (standard configs and attack/defense suites), and governance/compliance (DP budgets, auditing, and incident response integrated into FL MLOps). Addressing these is key to FL systems that are deployable, resilient, and privacy-compliant in sensitive domains (healthcare, finance, edge/IoT).

Against this backdrop, our survey contributes the following. We unify system paradigms and privacy/robustness mechanisms in a single comparative framework; we add practical hybrid recipes (e.g., DP+SA, TEE-assisted aggregation, selective HE) and evaluation criteria (privacy-utility-cost-scalability) often missing in prior work; and we highlight domain-specific guidance for deploying FL under heterogeneity and regulatory constraints. This integrated, defense-in-depth perspective aims to guide future research and practice toward efficient, secure, and privacy-compliant FL, with an emphasis on standardized benchmarks, transparent reporting, and real-world readiness.

VI. FUTURE STUDIES AND SUGGESTIONS

To further strengthen the security posture of FL, future work should concentrate on advancing privacy-preserving methods while reducing their system overheads. In particular, the accuracy-privacy and efficiency-privacy trade-offs of DP and HE ought to be minimized through adaptive privacy budgeting, precision-aware encryption parameterization, and task-sensitive scheduling, thereby improving their practicality in large-scale deployments. In parallel, enhancing TEEs and SA with robust hardware support including attestation, enclave hardening, and efficient mask/dropout handling will be critical to limiting data leakage during computation and aggregation. For distributed/decentralized FL, the design of stronger, Byzantine-resilient consensus protocols and topology-aware communication schemes can help secure inter-client exchanges and improve global model consistency under churn and heterogeneous connectivity. Finally, the field would benefit from systematic, repeatable security assessments that evaluate extensibility and scalability covering privacy leakage, robustness to adaptive attacks, utility, and systems cost, so that proposed techniques can be compared on a common, real-world-oriented basis.

REFERENCES

[1] Yin X., Zhu Y., and Hu J. (2021). A comprehensive survey of privacy-preserving federated learning: a taxonomy, review, and future directions, *ACM Computing Surveys*, 54, (6), 1-36.

[2] Peng L. and Qiu M. (2024). AI in healthcare data privacy-preserving: enhanced trade-off between security and utility. *Proceedings of Knowledge Science, Engineering and Management (KSEM)*, 349-360.

[3] Paracha A, Arshad J, Farah M. B., and Ismail K. (2024). Machine learning security and privacy: a review of threats and countermeasures, *EURASIP Journal on Information Security*, 2024, (1).

[4] Ma S., Cao Y., and Xiong L. (2021). Transparent contribution evaluation for secure federated learning on blockchain. *37th International Conference on Data Engineering Workshops (ICDEW)*, IEEE, 88-91.

[5] Bonawitz K., Eichner H., Grieskamp W., Huba D., Ingerman A., Ivanov V., Kiddon C., Konečný J., Mazzocchi S., McMahan H. B., Van Overveldt T., Petrou D., Ramage D., and Roselander J. (2019). Towards federated learning at scale: system design, *arXiv:1910.06664*.

[6] Zhang C., Xie Y., Bai H., Yu B., Li W., and Gao Y. (2021). A survey on federated learning, *Knowledge-Based Systems*, (216), 106775.

[7] Truong N., Sun K., Wang S., Guitton F., and Guo Y. (2021). Privacy preservation in federated learning: an insightful survey from the GDPR perspective, *Computers & Security*, (110), 102402.
[8] Kairouz P., McMahan H., B., Avent B., Bellet A., Bennis M., and et al. (2021). Advances and open problems in federated learning, *arXiv:1912.04977*.
[9] Ha T., and Dang T. K. (2022). Inference attacks based on GAN in federated learning, *International Journal of Web Information System*, 18, (2-3), 117-136.
[10] Liu Z., Guo J., Yang W., Fan J., and Zhao J. (2022). Privacy-preserving aggregation in federated learning: A survey, *arXiv:2203.17005*.
[11] Zhang Y., Zeng D., Luo J., Xu Z., and King I. (2023). A Survey of trustworthy federated learning with perspectives on security, robustness, and privacy, *arXiv:2302.10637*.
[12] Rafi T., H., Noor F., A.,Hussain T., and Chae D-K. (2023). Fairness and privacy-preserving in federated learning: A survey, *arXiv:2306.08402*.
[13] Chhetri B., Gopali S., Olapojoye R., Dehbash S., and Namin A. S. (2023). A survey on blockchain-based federated learning and data privacy, *arXiv:2306.17338*.
[14] Hasan J. (2023). Security and privacy issues of federated learning, *arXiv:2307.12181*.
[15] Zhao J. C., Bagchi S., Avestimehr S., Chan K. S., Chaterji S., Dimitriadis D., Li J., Li N., Nourian A., and Roth H. R. (2024). Federated learning privacy: attacks, defenses, applications, and policy landscape – a survey, *arXiv:2303.01347*.
[16] Li H., Ge L., and Tian L. (2024). Survey: federated learning data security and privacy-preserving in edge-internet of things, *Artificial Intelligence Review*, 57, 130.
[17] Rahman R. (2025). Federated learning: A survey on privacy-preserving collaborative intelligence, *arXiv: 2504.17703*.
[18] Jimenez-Gutierrez D. M., Falkouskaya Y., Hernandez-Ramos J. L., Anagnostopoulos A., Chatzigiannakis I., and Vitaletti A. (2025). On the security and privacy of federated learning: a survey with attacks, defenses, frameworks, applications, and future directions, *arXiv:2508.13730*.
[19] Feng Y., Guo Y., Hou Y., Wu Y., Lao M., Yu T., and Liu G. (2025). A survey of security threats in federated learning, *Complex & Intelligent Systems*, 11, 165.
[20] Soltani B., Zhou Y., Haghighi V., and Lui J. C. S. (2023). A survey of federated evaluation in federated learning. *Proceedings of the Thirty-Second International Joint Conference on Artificial Intelligence (IJCAI)*, 6769-6777.
[21] Konečný J., McMahan H. B., Ramage D., and Richtárik P. (2016). Federated optimization: distributed machine learning for on-device intelligence, *arXiv:1610.02527*.
[22] AbdulRahman S., Tout H., Ould-Slimane H., Mourad A., Talhi C., and Guizani M. (2021). A survey on federated learning: the journey from centralized to distributed on-site learning and beyond, *IEEE Internet of Things Journal*, 8, (7), 5476-5497.
[23] Xu Z., Zhang Y., Andrew G., Choquette C., Kairouz P., Mcmahan B., Rosenstock J., and Zhang Y. (2023). Federated learning of gboard language models with differential privacy. *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics*, 5, 629-639.
[24] Kasturi A., Sivaraju R., and Hota C. (2022). FedPeer: A peer-to-peer learning framework using federated learning, *Edge Analytics*, 517-525.
[25] Yuan L., Wang Z, Sun L., Yu P. S., and Brinton C. G. (2023). Decentralized federated learning: a survey and perspective, *arXiv preprint arXiv:2306.01603*.
[26] Liu Y., Shi Y., Li Q., Wu B., Wang X., and Shen L. (2024). Decentralized directed collaboration for personalized federated learning, *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 23168-23178.
[27] Zhao Y., Li M., Lai L., Suda N., Civin D., and Chandra V. (2022). Federated learning with non-IID data, *arXiv preprint arXiv:1806.00582*.
[28] Pandey M., Pandey S., and Kumar A. (2022). Introduction to Federated Learning, Springer, Berlin, Germany, 1-17.
[29] Lv Y., Ding H., Wu H., Zhao Y., and Zhang L. (2023). FedRDS: federated learning on non-IID data via regularization and data sharing, *Applied Sciences*, 13, (23), 12962.
[30] Islam M. S., Javaherian S., Xu F., Yuan X., and Tizeng N-F. (2024). FedClust: tackling data heterogeneity in federated learning through weight-driven client clustering. *In Proceedings of the IEEE International Conference on Parallel Processing (ICPP)*, 474-483.
[31] Ghosh A., Chung J., and Yin D. (2021). An efficient framework for clustered federated learning. *arXiv:2006.04088*
[32] Elhussein A., Gursoy G. (2023). Privacy-preserving patient clustering for personalized federated learning, *arXiv:2307.08847*.

[33] Yoo J. H., Son H. M, Jeong H., Jang E. H., Kim A. Y., Yu H. Y., Jeon H. J., and Chung T-M. (2021). Personalized federated learning with clustering: non-iid heart rate variability data application, *arXiv:2108.01903*.
[34] Nguyen T. and Thai M. T. (2023). Preserving privacy and security in federated learning, *IEEE/ACM Transactions on Networking*, 32,(1), 833-843.
[35] Tang B., Xiao Y., Zhang L., Cao B., Tang M., and Yang Q. (2024). AFL-HCS: asynchronous federated learning based on heterogeneous edge client selection, *Cluster Computing*, 27, (5), 6247-6264.
[36] Xie C. (2020). Asynchronous federated optimization, *arXiv:1903.03934*.
[37] Nguyen J., Malik K., Zhan H., Yousefpour, A. Rabbat M., Malek M., and Huba D. (2022). Federated learning with buffered asynchronous aggregation, *arXiv:2106.06639*.
[38] Odeh J. O., Yang X., Nawakanma C. I., and Dhelim S. (2024). Asynchronous privacy-preservation federated learning method for mobile edge network in industrial internet of things ecosystem, *Electronics*, 13, (9), 1610.
[39] Yu L., Sun X., Albelaihi R., and Yi C. (2022). Latency aware semi-synchronous client selection and model aggregation for wireless federated learning, *arXiv:2210.10311*.
[40] Xu C, Qu Y., Xiang Y, and Gao L. (2023). Asynchronous federated learning on heterogeneous devices: a survey, *arXiv:2109.04269*.
[41] Rieke N., Hancox J., Fausto Milletari W L., Roth H., Albarqouni S., Bakas S., and et al. (2021). The future of digital health with federated learning, *arXiv:2003.08119*.
[42] Islam M. S., Panta S., Xu F., Yuan X., Chen L., Tzeng N-F. (2025). SEAFL: enhancing efficiency in semi-asynchronous federated learning through adaptive aggregation and selective training, *arXiv:2503.05755*.
[43] Fan B., Jiang J.,Su X.,Tarkoma S., and Hui P. (2024). A survey on model-heterogeneous federated learning: problems, methods, and prospects, *arXiv:2312.12091*.
[44] Fallah A., Mokhtari A., and Ozdaglar A. (2020). Personalized federated learning: a meta-learning approach, *arXiv:2002.07948*
[45] Tan A. Z., Yu H., Cui L., and Yang Q. (2022). Towards personalized federated learning, *arXiv:2103.00710*.
[46] Gao D., Yao X., and Yang Q. (2022). A survey on heterogeneous federated learning, *arXiv:2210.04505*.
[47] Zeng D., Xu Z., Liu S., Pan Y., Wang Q., and Tang X. (2024). On the power of adaptive weighted aggregation in heterogeneous federated learning and beyond, *arXiv:2310.02702*.
[48] Pfeiffer K., Rapp M., Khalili R., and Henkel J. (2023). Federated learning for computationally-constrained heterogeneous devices: a survey, *arXiv:2307.09182*.
[49] Lyu L. and Chen C. (2021). A Novel Attribute Reconstruction Attack in Federated Learning, *arXiv:2108.06910*.
[50] Liang J., Wang R., Feng C., and Chang C. (2023). A survey on federated learning poisoning attacks and defenses, *arXiv:2306.03397*.
[51] Banse A., Kreischer J., and Oliva i Jürgens X. (2024). Federated learning with differential privacy. *arXiv:2402.02230*.
[52] Pan Y., Chao Z., He W., Jing Y., Hongjia L., and Liming W. (2024). FedSHE: privacy-preserving and efficient federated learning with adaptive segmented CKKS homomorphic encryption, *Cybersecurity*, 7, (1), 40.
[53] Alam T. and Gupta R. (2023). Federated learning and its role in the privacy preservation of IOT devices, *Future Internet*, 14, (9), 246.
[54] Zhang L., Duan B., Li J., Ma Z., and Cao X. (2024). A TEE-based federated privacy protection method: proposal and implementation, *Applied Sciences-Basel*, 14, (8).
[55] Bonawitz K., Ivanov V., Kreuter B., Marcedone A., McMahan H. B., Patel S., Ramage D., Segal A., and Seth K. (2017). Practical secure aggregation for privacy-preserving machine learning. *CCS '17: Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, 1175-1191
[56] Hard A., Rao K., Mathews R., Ramaswamy S., Beaufays F., Augenstein S:, Eichner H:, Kiddon C., and Ramage R. (2019). Federated learning for mobile keyboard prediction. *arXiv:1811.03604*.
[57] Bagdasaryan E., Veit A., Hua Y., Estrin D., and Shmatikov V. (2019). How to backdoor federated learning, *arXiv:1807.00459*.
[58] Blanchard P., El Mhamdi E. M., Guerraoui R., and Stainer J. (2017). Machine Learning with Adversaries: Byzantine Tolerant Gradient Descent, *31st Conference on Neural Information Processing Systems (NIPS 2017)*. 118-128
[59] Yin D., Chen Y., Ramchandran K., and Bartlett P. (2021). Byzantine-robust distributed learning: towards optimal statistical rates. *arXiv:1803.01498*.
[60] El Mhamdi E. H., Guerraoui R:,and Rouault S. (2018). The hidden vulnerability of distributed learning in byzantium, *arXiv:1802.07927*.
[61] Fang M., Cao X., Jia J., and Gong N. (2020). Local model poisoning attacks to byzantine-robust federated learning, *29th USENIX Security Symposium. 1623-1640.*

[62] Yar G., Boboila S., Nita-Rotaru C., and Oprea A. (2023). Backdoor attacks in peer-to-peer federated learning. *arXiv:2301.09732*.
[63] Cao X., Fang M., Liu J., and Gong N. Z. (2022). FLTrust: byzantine-robust federated learning via trust bootstrapping, *arXiv:2012.13995*.
[64] Dwork C., and Roth A. (2014). The algorithmic foundations of differential privacy, *Foundations and Trends in Theoretical Computer Science*, 9, (3-4), 211-407.