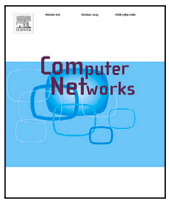
Computer Networks 285 (2026) 112402

Contents lists available at ScienceDirect

Computer Networks

journal homepage: www.elsevier.com/locate/comnet



Survey paper

Unlocking distributed intelligence: A comprehensive survey on federated split learning's evolution, challenges, and future frontiers☆

David Naseh [a],*, Arash Bozorgchenani [b], Swapnil Sadashiv Shinde [c], Daniele Tarchi [d]

[a] School of Electronic and Electrical Engineering, University of Leeds, Leeds, LS2 9JT, United Kingdom
[b] School of Computer Science, University of Leeds, Leeds, LS2 9JT, United Kingdom
[c] School of Electrical Engineering and Computer Science, KTH Royal Institute of Technology, Stockholm, 11428, Sweden
[d] Department of Information Engineering, University of Florence, Florence, 50139, Italy

ARTICLE INFO

Keywords:
Federated split learning
Distributed machine learning
Edge computing
Internet of Things
Wireless networks
Healthcare analytics
Vehicular networks
Large language models
6G networks
Privacy preservation

ABSTRACT

Federated Split Learning (FSL) has emerged as a transformative paradigm that synergizes the parallel processing and scalability of Federated Learning (FL) with the computational efficiency and enhanced privacy of Split Learning (SL). This comprehensive survey provides a systematic exploration of FSL, beginning with a detailed taxonomy of Distributed Machine Learning (DML) paradigms, tracing the progression from foundational concepts to advanced frameworks such as Federated Split Transfer Learning (FSTL) and Generalized Federated Split Transfer Learning (GFSTL). It then delves into the core challenges inherent to FSL, including privacy and security risks, system and data heterogeneity, computational and system constraints, communication overhead, and model optimization complexities. The heart of the survey presents a detailed categorization and analysis of FSL's diverse applications across key domains, including the Internet of Things (IoT) and Edge Computing (EC), wireless networks, healthcare, vehicular networks, Large Language Models (LLMs), and Earth Observation (EO). To ground this research, the survey further discusses standardized evaluation methodologies and implementation frameworks, followed by a quantitative visualization of survey data and research trends. The work concludes by synthesizing critical research gaps and outlining promising future directions. Unlike existing surveys that focus on isolated aspects, this work uniquely bridges challenges, solutions, and applications through a unified framework, incorporating network-specific analysis and quantitative trend visualization. By synthesizing insights from over 100 recent research articles and providing a critical analysis of evaluation methodologies, this survey offers an essential roadmap for researchers and practitioners developing scalable, efficient, and privacy-aware distributed intelligence for the 6G and AI era.

1. Introduction

The emergence of Beyond 5G (B5G) networks and the ongoing development of 6G technologies are intensifying the demand for smart, efficient, and privacy-conscious data handling. Machine Learning (ML), a driving force behind advancements in autonomous systems, healthcare, and smart cities [1], will not merely complement these future networks but serve as a cornerstone. B5G and 6G in particular are anticipated to be inherently self-aware, self-optimizing, and extremely intelligent systems, facilitating capabilities that far surpass those of existing mobile communications. These capabilities include immersive eXtended Reality (XR), pervasive intelligent surfaces, the haptic internet, and highly precise autonomous systems. ML algorithms, particularly in the realms of Deep Learning (DL) and Reinforcement Learning (RL), will be essential to handle the vast complexity and evolving nature of these networks. They will enable intelligent resource distribution, predictive maintenance, dynamic spectrum allocation, interference control, and real-time anomaly detection for enhanced security. In essence, by tapping into ML, B5G and 6G will transcend mere connectivity, cultivating a "connected intelligence" that seamlessly adapts to a wide range of user needs and environmental changes, offering a genuinely transformative experience.

However, traditional Centralized Machine Learning (CML) methods face significant limitations in this evolving landscape. They introduce high latency, considerable computational overhead, and critical

☆ This work was partially supported by the European Union - Next Generation EU under the Italian National Recovery and Resilience Plan (NRRP), Mission 4, Component 2, Investment 1.3, CUP B83C22004870007, partnership on "Telecommunications of the Future" (PE00000001 - program "RESTART").

* Corresponding author.
E-mail addresses: d.naseh@leeds.ac.uk (D. Naseh), a.bozorgchenani@leeds.ac.uk (A. Bozorgchenani), ssshinde@kth.se (S.S. Shinde), daniele.tarchi@unifi.it (D. Tarchi).

https://doi.org/10.1016/j.comnet.2026.112402
Received 17 October 2025; Received in revised form 6 March 2026; Accepted 17 May 2026
Available online 22 May 2026

privacy concerns, especially in distributed environments where the consolidation of raw data is often infeasible [2].

The advent of B5G and 6G networks introduces an era of unprecedented connectivity, demanding ultra-high data rates, near-zero latency, immense device density, and an inherently intelligent, autonomous infrastructure. These future networks will integrate a vast array of interconnected devices, from advanced Internet of Things (IoT) sensors to holographic communication systems and autonomous vehicles, all generating enormous volumes of heterogeneous data at the network edge. Consequently, CML approaches, which depend on transmitting all data to a central cloud for processing, become increasingly inefficient and impractical due to the sheer data scale, stringent latency demands, and escalating concerns over data privacy and security.

Distributed Machine Learning (DML) has become vital in addressing the challenges of training ML models collaboratively across decentralized network nodes such as edge devices and local servers without aggregating raw data [3]. This architectural shift is essential to meet the efficiency and privacy demands of B5G and 6G networks [4]. Within the context of B5G and 6G, various DML algorithms solve the limitations of CML by enabling model training without raw data aggregation. This approach reduces latency, prevents computational bottlenecks, and maintains privacy. DML paradigms are generally categorized based on how tasks and model parameters are distributed and synchronized. Notably, Federated Learning (FL), Split Learning (SL), and Federated Split Learning (FSL) are significant for future networks.

FL represents a decentralized training approach where numerous clients jointly contribute to training a common model without transferring their data off-device, attracting considerable attention within the DML sphere [5]. This approach is particularly advantageous in areas sensitive to privacy, such as healthcare, as it aligns with data privacy regulations, including the General Data Protection Regulation (GDPR) and the Health Insurance Portability and Accountability Act (HIPAA) [6]. Nevertheless, FL encounters challenges including substantial communication expenses, device diversity, and susceptibility to adversarial threats [7]. To mitigate these concerns, researchers have proposed optimization strategies, including Differential Privacy (DP), quantization, and adaptive aggregation [8].

Another method gaining popularity in the field of DML is SL, which segments DL models across clients and servers, thereby alleviating the computational load on devices with limited resources [9]. Instead of transferring raw input data, SL transmits only intermediate activations, also referred to as smashed data, which enhances data privacy [10]. Nevertheless, SL introduces its own set of challenges, such as sequential training delays, heightened latency [11], increased communication overhead, and vulnerability to feature inversion attacks [12]. Recent studies have been directed toward augmenting the efficiency and security of SL through techniques like dynamic layer partitioning, asynchronous updates, and generative activation strategies [13].

FSL merges the benefits of FL and SL to establish a hybrid model that achieves a balance between computational efficiency and privacy protection [14]. In this configuration, clients perform partial training of their neural networks locally and then transmit intermediate data representations to a central server for additional processing [15]. However, despite its strengths, FSL encounters challenges, including privacy risks linked to activation transmission, diverse device capabilities, and difficulties in model optimization [16]. Researchers are fervently developing solutions, such as the adaptive selection of cut-layers, model pruning, and parallelized training, to address these challenges [17].

FSL provides significant benefits for devices with limited resources in domains such as IoT, Edge Computing (EC), and Vehicular Edge Computing (VEC) [18]. Furthermore, FSL has been successfully applied across various fields, including healthcare, wireless networks [19], and Earth Observation (EO) [20]. In the healthcare sector, FSL allows for the collaborative training of models on sensitive medical information while ensuring compliance with rigorous privacy laws [21]. In IoT and EC, FSL reduces the computational load on limited-capacity devices while preserving data locality [14]. It also improves communication efficiency in wireless networks and supports high-speed applications like autonomous vehicles and smart city infrastructures [22]. For VEC, FSL optimizes AI-driven analytics in vehicle systems, aiding in real-time decision processes and resource allocation [18]. In the realm of remote area surveillance, FSL benefits distributed training of satellite-based EO systems, enhancing data precision and scalability in difficult environments, while addressing privacy and communication issues [23].

Despite the substantial progress made in FSL research, significant gaps remain in fully understanding its challenges and practical applications [24]. Existing studies often focus on theoretical advances or domain-specific implementations, underscoring the need for a unified discussion that bridges challenges and applications in FSL [25]. This survey addresses this gap by offering a structured review of key challenges, optimization techniques, and real-world applications of FSL, with a specific focus on the intersection of FSL and computer networks. We analyze how network architectures, protocols, and constraints shape the performance and feasibility of FSL, and conversely, how FSL creates new opportunities for network intelligence.

1.1. Gap analysis and comparison with related works

Existing surveys on distributed learning, while valuable, are constrained in scope. Hu et al. [26] provide a comprehensive review and experimental evaluation of SL paradigms, offering engineering insights, but their work does not extend to the advanced DML family, such as Federated Split Transfer Learning (FSTL), and Generalized Federated Split Transfer Learning (GFSTL) or explore cross-domain applications beyond standard benchmarks. Zhang et al. [27] present a thorough analysis of DML for next-generation communication networks, systematically comparing FL and SL with a three-dimensional framework for privacy, fairness, and efficiency trade-offs; however, their survey does not delve into hybrid FSL or its advanced variants. Duan et al. [28] provide a focused review of FL and SL in edge–cloud IoT but lack quantitative trend analysis and coverage of FSTL/GFSTL. Hukkeri et al. [29] examine Split-Fed Learning with experimental comparisons, yet their scope excludes domains such as vehicular networks, Earth observation, and large language models. Tajabadi et al. [30] concentrate on biomedical applications, while Alsharif et al. [31] survey federated learning broadly without delving into split learning intricacies. Naseh et al. [32] introduce hierarchical DML frameworks (FSTL and GFSTL) for 6G intelligent transportation systems, but their work is a research contribution rather than a comprehensive survey. Consequently, none of these works provide a unified treatment that integrates challenges, solutions, and applications across domains, nor do they explicitly analyze the interplay between FSL and computer network architectures, quantitative research trends, or standardization efforts.

To systematically illustrate these gaps, Table 1 compares this survey with prior reviews across key dimensions: scope, taxonomy granularity, challenge coverage, application domains, network focus, inclusion of standardization, coverage of advanced frameworks (FSTL/GFSTL), and quantitative analysis. The comparison includes the surveys discussed above as well as the research contribution on hierarchical DML frameworks [32] to highlight that while FSTL and GFSTL have been proposed, a comprehensive survey synthesizing the entire FSL ecosystem remains absent.

As evident from Table 1, existing works either focus on specific paradigms (SL, FL), target particular application domains, or lack comprehensive coverage of advanced FSL variants (FSTL, GFSTL). Moreover, none provide a systematic integration of challenges, solutions, and applications with quantitative trend analysis and standardization discussion. This survey uniquely fills these gaps by offering a holistic, structured, and up-to-date review of the entire FSL ecosystem, making it a valuable resource for researchers and practitioners alike.

Table 1
Comparison of our survey with existing surveys and related works on FSL and distributed learning.

Survey	Scope	Taxonomy granularity	Challenge coverage	Application domains	Network focus	Standardization	FSTL/GFSTL	Quantitative analysis
Hu et al. [26]	SL paradigms (experimental focus)	High (SL variants)	Computation, communication	General ML benchmarks	No	No	No	Partial (experimental)
Zhang et al. [27]	DML (FL & SL) for communication networks	High (3D trade-off framework)	Privacy, fairness, efficiency	6G, IoT, edge computing	Yes	No	No	No
Duan et al. [28]	FL & SL in edge–cloud IoT	Medium	Privacy, efficiency	IoT, edge computing	Limited	No	No	No
Hukkeri et al. [29]	Split-fed learning	High	Communication, computation	General ML benchmarks	No	No	No	Partial (experimental)
Tajabadi et al. [30]	Decentralized learning in biomedicine	Medium	Privacy, heterogeneity	Healthcare	No	No	No	No
Alsharif et al. [31]	Federated learning	High	Privacy, security, efficiency	IoT, healthcare, smart cities	No	No	No	No
Naseh et al. [32]	FSTL/GFSTL for 6G ITS (research paper)	Medium (hierarchical frameworks)	Communication, scalability	ITS, Earth observation	Yes	No	Yes	Yes (simulation)
This survey	**FSL, FSTL, GFSTL**	**High (architectures, challenges, solutions, applications)**	**Privacy, security, heterogeneity, computation, communication, model optimization, networking**	**IoT/edge, wireless, healthcare, vehicular, LLMs, Earth observation**	**Yes (dedicated networking challenges, co-design)**	**Yes (3GPP, ITU, IETF, IEEE, ETSI, O-RAN)**	**Yes**	**Yes (trend charts, dataset analysis, publication statistics)**

1.2. Key contributions of our survey

This survey makes the following key contributions:

- **Systematic taxonomy of DML paradigms:** We trace the evolution from foundational FL and SL to advanced frameworks, including FSTL and GFSTL, and provide a latency analysis (Table 4) and comparative table of advantages and challenges (Table 5).
- **Comprehensive challenge–solution mapping:** For each core challenge (privacy, heterogeneity, resource constraints, communication overhead, model optimization, networking), we present a structured analysis comprising problem formulation, taxonomy of solution strategies, representative methods, and a comparative table summarizing pros, cons, overhead, and assumptions (Section 3).
- **Cross-domain application analysis:** We categorize FSL applications by deployment context (edge–IoT continuum, high-mobility networks, regulated domains, large-scale training, remote sensing) and derive cross-cutting design patterns, supported by summary tables (Section 4).
- **Quantitative research trend visualization:** Based on 64 core papers, we provide charts depicting distribution by application, publication year, evaluation metrics, challenges, techniques, and datasets, offering a data-driven overview of the field (Section 5.3).
- **Standardization landscape and future directions:** We review relevant activities in Third Generation Partnership Project (3GPP), International Telecommunication Union – Telecommunication Standardization Sector (ITU-T), Internet Engineering Task Force (IETF), IEEE, European Telecommunications Standards Institute (ETSI), Open Radio Access Network Alliance (O-RAN), and outline eight future research avenues, including adaptive partitioning, integration with foundation models, hardware–software co-design, and GFSTL for 6G ITS (Sections 3.8 and 6).
- **Evaluation framework and design rubric:** We synthesize best practices from recent optimization strategies into a comparative rubric that guides researchers in selecting appropriate FSL variants based on application constraints (Sections 5.1 and 5.2).

1.3. Research questions

To guide our systematic exploration and highlight the scientific contributions, this survey addresses the following Research Questions (RQs):

1. **RQ1 (Evolution):** How do FSL architectures evolve from foundational FL and SL to advanced frameworks like FSTL and GFSTL, and what are their comparative advantages and limitations?
2. **RQ2 (Challenges and Solutions):** What are the core challenges in FSL, and what solution strategies have been proposed to address privacy, heterogeneity, resource constraints, communication overhead, model optimization, and networking issues? How do these strategies compare in terms of effectiveness, overhead, and assumptions?
3. **RQ3 (Applications and Patterns):** How is FSL applied across different domains (IoT/edge, wireless, healthcare, vehicular, LLMs, Earth observation), and what cross-cutting patterns and design guidelines emerge?
4. **RQ4 (Evaluation):** What evaluation methodologies, metrics, benchmarks, and implementation frameworks are used to assess FSL systems, and how can they be standardized to enable fair comparison?
5. **RQ5 (Trends and Gaps):** What are the current research trends in FSL (publication volume, application focus, evaluation practices), and what critical gaps remain?
6. **RQ6 (Future Directions):** What are the most promising future research directions for FSL, particularly in the context of 6G, foundation models, and emerging network paradigms?

Each subsequent section begins with a brief statement indicating which RQs it addresses, thereby providing a coherent thread throughout the paper.

1.4. Reader guide

To help readers navigate this survey and understand the relationships between sections, Fig. 1 provides a visual roadmap. The survey

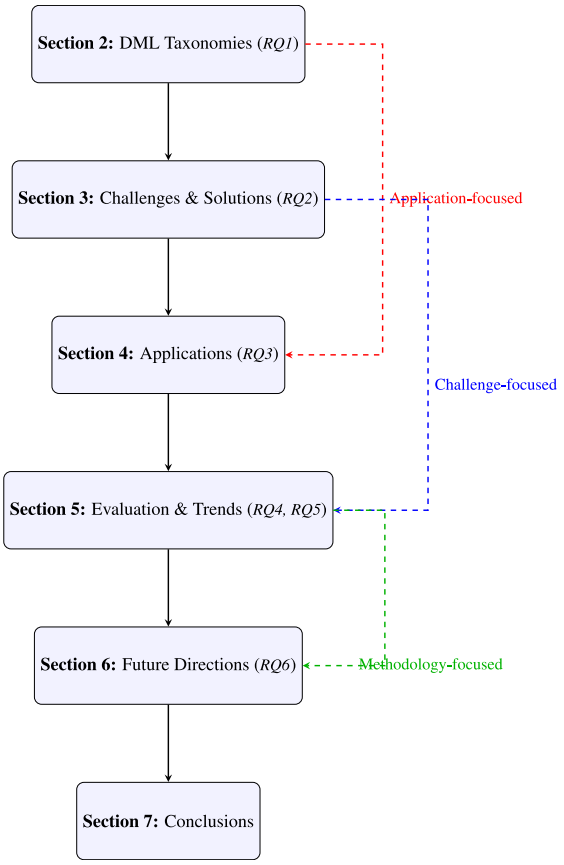


Fig. 1. Reader guide: relationships between sections and corresponding research questions. Dashed arrows indicate alternative reading paths.

is organized into six main parts, each addressing specific research questions (RQs) as indicated. Table 2 provides a quick reference to the content and objectives of each section. Readers seeking a quick overview may proceed directly to Section 5.3 for quantitative trends, Section 4 for application-specific insights, or Section 6 for future directions. For a comprehensive understanding, we recommend reading the sections in sequential order.

1.5. Methodology of inclusion and exclusion

To ensure the rigor, transparency, and reproducibility of this survey, we followed a systematic methodology for paper selection and analysis, inspired by PRISMA (Preferred Reporting Items for Systematic Reviews and Meta-Analyses) guidelines [33]. The process comprised four stages: identification, screening, eligibility, and inclusion.

Identification:. We conducted a comprehensive search of major scientific databases, including IEEE Xplore, ACM Digital Library, Science Direct, Elsevier, and arXiv. The search strings combined terms related to federated split learning and its variants: "`federated split learning`", "`split federated learning`", "SplitFed", "`federated split transfer learning`", "FSTL", and "`generalized federated split transfer learning`". The initial search yielded 337 papers.

Screening:. Titles and abstracts were screened to remove duplicates and exclude papers that were clearly outside the scope. Papers were retained if they: (i) addressed FSL, FL, SL, or their hybrid variants; (ii) were written in English; (iii) were published as journal articles, conference proceedings, or preprints from reputable sources. After screening, 122 papers remained.

Table 2
Section overview and mapping to research questions.

Section	Title	Key Content and RQs Addressed
Section 2	Distributed machine learning paradigms	Taxonomy of FL, SL, FSL, FSTL, GFSTL; latency analysis; comparative advantages (RQ1)
Section 3	Challenges in FSL	In-depth analysis of privacy, heterogeneity, resource constraints, communication, optimization, networking; solution strategies and trade-offs (RQ2)
Section 4	FSL applications	Cross-domain analysis (IoT/edge, wireless, healthcare, vehicular, LLMs, EO); design patterns and lessons learned (RQ3)
Section 5	Evaluation, frameworks, and trends	Performance metrics, benchmarks, implementation tools; quantitative visualization of publication trends (RQ4, RQ5)
Section 6	Future directions	Eight key research avenues, including 6G, foundation models, and standardization (RQ6)
Section 7	Conclusion	Summary of contributions and key insights

Eligibility:. Full-text articles were assessed against the following inclusion criteria:

- **Relevance to FSL:** The paper must directly address FSL or its integration with FL and SL, rather than discussing FL or SL in isolation.
- **Publication date:** To capture recent advances, we focused on papers published in the last six years (2020–2025), though seminal older works were considered for foundational context.
- **Impact and quality:** Preference was given to papers from high-impact journals (e.g., IEEE Transactions, Elsevier journals) and top-tier conferences (e.g., AAAI, INFOCOM, MobiCom). Preprints from arXiv were included only if they represented significant novel contributions not yet published elsewhere.

This stage resulted in 113 papers meeting the eligibility criteria.

Inclusion:. The remaining papers were categorized into two main groups based on their primary contribution:

- **Challenges-focused papers (81 papers):** Includes works that analyze or propose solutions to specific FSL challenges, such as privacy, heterogeneity, resource constraints, communication efficiency, and model optimization.
- **Applications-focused papers (64 papers):** Includes works that apply FSL to concrete domains, including IoT/edge computing, wireless networks, healthcare, vehicular networks, large language models, and Earth observation.

Note that some papers contributed to both categories and were counted in both, which explains why the sum exceeds the total unique papers. Table 3 summarizes the number of papers at each stage.

Flow diagram:. Fig. 2 illustrates the paper selection process as a PRISMA-style flow diagram, providing a visual summary of the screening and inclusion stages.

Data extraction and analysis:. For each included paper, we extracted key information: (i) bibliographic details, (ii) addressed challenges, (iii) proposed techniques/solutions, (iv) evaluation metrics and datasets, (v) application domain, and (vi) key findings. This structured data formed the basis for our taxonomy, comparative tables, and quantitative trend analysis presented in Sections 3–5.

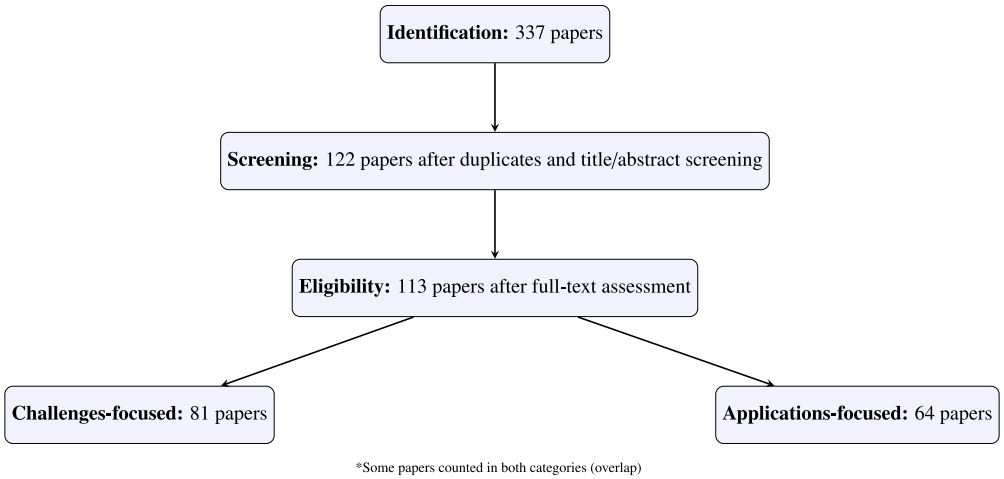


Fig. 2. PRISMA-style flow diagram of the paper selection process.

Table 3
Summary of paper inclusion and exclusion stages.

Criteria	Number of papers	Remarks
Initial pool	337	Papers identified through database searching
Relevance to FSL	122	Papers directly addressing FSL, FL, or SL after title/abstract screening
Publication date	113	Papers published in the last 6 years (2020–2025)
Impact factor	82	Papers from high-impact journals or top-tier conferences
Challenges-focused	81	Papers dealing with challenges (privacy, heterogeneity, constraints, communication, optimization)
Applications-focused	64	Papers focusing on specific applications (IoT/edge, wireless, healthcare, vehicular, LLMs, EO)

1.6. Organization of the survey

The remainder of this paper is organized as follows: Section 2 provides a taxonomy of DML paradigms, introducing FL, SL, and FSL, along with a comparative analysis of their key differences, advantages, and disadvantages. Section 3 discusses the challenges associated with FSL, focusing on privacy, heterogeneity, resource constraints, communication overhead, and model optimization. Section 4 categorizes and examines recent advances and applications of FSL in various domains, including VEC, wireless networks, IoT, EC, LLM, and healthcare. Section 5 delves into the critical aspects of evaluating FSL systems, discussing standardized performance metrics, benchmarks, and the software frameworks and simulators used for implementation and testing. Then we present several charts that quantitatively summarize the distribution of our survey papers by application, publication year, evaluation metrics reported, challenges health with, model or technique trends, and dataset or benchmarks usage. Section 6 outlines future research directions, emphasizing the role of FSL in next-generation AI-driven systems and 6G networks. Finally, Section 7 concludes the paper by summarizing key insights and contributions.

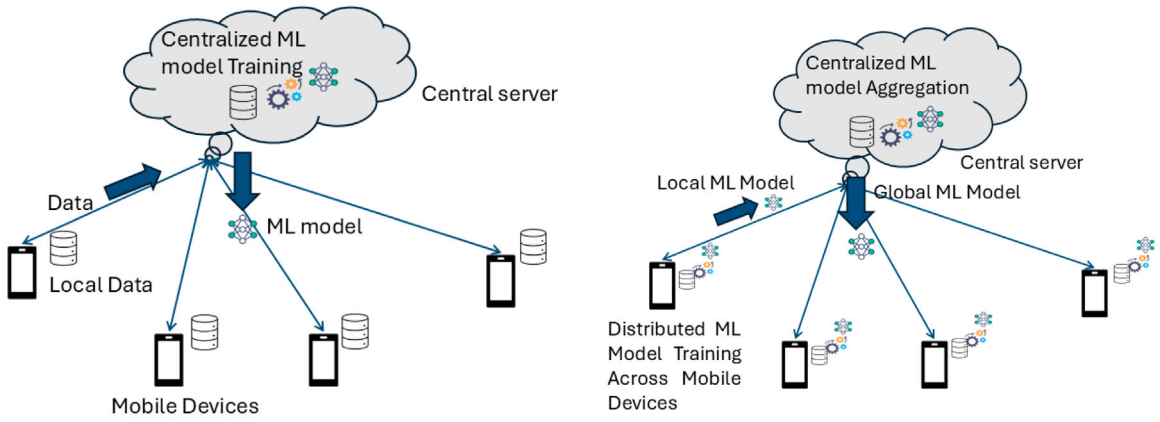
2. Distributed machine learning paradigms

This section delves into the foundational concepts and advanced paradigms within distributed ML, setting the stage for a comprehensive understanding of various collaborative learning approaches. We begin by exploring DML, examining its fundamental principles and the motivations behind its development. Subsequently, we provide a detailed analysis of FL and SL, dissecting their architectural frameworks, advantages, and inherent challenges. Building upon these, we introduce FSL, a hybrid approach that integrates the strengths of both FL and SL. Furthermore, we explore the role of Transfer Learning (TL) in enhancing DML systems, leading to the advanced Federated Split Transfer Learning (FSTL) framework. This discussion culminates with Generalized Federated Split Transfer Learning (GFSTL), an innovative paradigm designed for large-scale, heterogeneous environments. Following individual expositions, we present a latency analysis and comparison of these diverse DML methods and conclude with a consolidated overview of the lessons learned, highlighting their comparative advantages and challenges.

2.1. DML

DML is a transformative paradigm that enables collaborative model training across multiple devices or nodes. This approach avoids centralizing raw data, making it particularly well-suited for applications in EC, VEC, the IoT, healthcare, and with LLMs [31]. By allowing devices to process data locally and share only model updates, DML inherently addresses critical challenges in distributed environments: ensuring data privacy, reducing communication overhead, and mitigating bandwidth limitations [34]. This is especially vital in sectors like healthcare, where sensitive patient data must remain localized due to stringent privacy regulations [5], and within the IoT, where the sheer volume of data generated by billions of devices simply cannot be efficiently centralized due to resource constraints [35]. Key DML frameworks, including FL, SL, and FSL, have emerged to tackle the complexities of these distributed data environments.

Within DML, communication patterns are often characterized by frequent and substantial data exchanges among computing nodes. This typically involves sharing crucial elements like model parameters, gradients, or intermediate activations. In strategies such as data parallelism, each node processes a subset of the data and then synchronizes its gradients, requiring significant inter-node communication. Similarly, model parallelism, where different parts of a model reside on different nodes, necessitates extensive communication for computations. Consequently, networking overhead in DML can be substantial due to the sheer volume of data transferred and stringent synchronization requirements. Collective communication operations, like “all-reduce” for gradient aggregation, are common and can become a significant bottleneck, especially in networks with high latency or limited bandwidth. To optimize communication in DML, techniques such as gradient compression, sparsity-aware communication, and overlapping communication



(a) Traditional centralized machine learning.

(b) Distributed machine learning.

Fig. 3. Comparison between (a) centralized machine learning (raw data transferred to server) and (b) distributed machine learning (clients exchange model parameters or activations with a central server; no direct client–client communication).

with computation are frequently employed to minimize idle network time and improve overall efficiency. Fig. 3 compares the centralized and distributed learning technologies. In the case of CML, distributed data from the device layer must be transferred to a powerful centralized server, where it is processed to generate an ML model. Based on the inference mechanisms, the model can then be sent back to the devices, either in full or partial form, or kept on the server. In the case of DML, devices can collaboratively process local data themselves and with the help of a server node to harness intelligence, and the resulting machine learning models can be utilized to enable distributed intelligence. As highlighted in Fig. 3, the key distinction between centralized and distributed learning lies in the nature of exchanged information: raw data versus model-related parameters. Importantly, in distributed paradigms such as FL, SL, and FSL, clients communicate exclusively with the server; there are no direct client-to-client interactions. This clarification addresses a common misconception that DML involves peer-to-peer exchanges, which are characteristic of fully decentralized gossip-based learning rather than the server-mediated architectures surveyed here.

2.1.1. Server-less and peer-to-peer DML

While server-based DML paradigms dominate the literature, server-less or peer-to-peer (P2P) distributed learning has emerged as an alternative for fully decentralized environments. In P2P DML, nodes communicate directly with each other without a central coordinator, exchanging model updates through gossip protocols or consensus mechanisms [36,37]. This approach eliminates single points of failure and enhances scalability, but introduces challenges in convergence, communication efficiency, and consistency. Techniques such as gossip averaging [38] and blockchain-based coordination [39] have been explored to enable robust P2P learning. Although FSL is typically defined in server-based settings, elements of P2P learning can complement FSL by enabling decentralized aggregation in GFSTL groups or resilient communication in dynamic environments [34]. For a comprehensive overview, we refer readers to surveys on decentralized learning [36, 40].

2.2. FL

FL is a decentralized ML paradigm that enables multiple clients (e.g., vehicles in an Intelligent Transportation System (ITS)) to collaboratively train a shared model without sharing their raw data [41]. Instead of centralizing data, FL aggregates locally computed model updates from distributed clients, thereby preserving data privacy while enabling large-scale learning [42]. This approach is particularly valuable in privacy-sensitive domains such as healthcare and ITS [31].

FL's networking aspects are distinct, primarily focusing on privacy-preserving collaborative model training without centralizing raw data. The communication pattern typically involves clients (e.g., mobile devices, edge servers) training models locally on their private datasets and then sending only model updates (e.g., gradients, updated weights) to a central server for aggregation. This reduces the communication volume compared to DML, as raw data is not transferred. However, the overhead still arises from the repetitive exchange of model updates across potentially many, heterogeneous, and intermittently connected client devices. Challenges include dealing with unreliable network connections, bandwidth limitations of individual devices, and the need for frequent communication rounds to ensure model convergence, all of which contribute to overall training time and resource consumption. Communication optimization in FL often involves client selection strategies, model compression techniques (e.g., quantization, sparsification), and asynchronous update schemes to mitigate these network constraints. Fig. 4 highlights the key elements of the FL training process. In this approach, a federation of devices locally processes the collected data to generate local FL models. These models are then sent to a centralized FL server, where they are aggregated to form a global model. The resulting global FL model is then sent back to the devices for further processing.

2.2.1. Architectural framework of FL

Consider a system with N distributed users defined through the set $\mathcal{U} = \{u_1, \ldots, u_N\}$, where each user i possesses a local dataset $D_i = \{(\mathbf{x}_k, y_k)\}$ containing K_i labeled samples. The feature vector $\mathbf{x}_k \in \mathbb{R}^N$ represents the kth sample, while y_k denotes its corresponding label. The objective is to train a global model W^p by minimizing a global loss function L^p across all clients while maintaining data locality, where W^p is the global model's parameters (weights) and L^p the global loss function calculated at the pth iteration or communication round.

The FL process operates through iterative communication rounds:

1. The server distributes the current global model W_t^p to participating clients
2. Each client i updates its local model by minimizing $L_i^p(D_i, W_t^p)$ on its private data
3. Clients transmit their updated parameters $W_{i,t}^p$ to the server
4. The server aggregates these updates using schemes like Federated Averaging (FedAvg) [43]:

$$W_{t+1} = \frac{1}{N} \sum_{i=1}^{N} W_{i,t}^p \tag{1}$$

This process continues until model convergence or meeting a predefined stopping criterion.

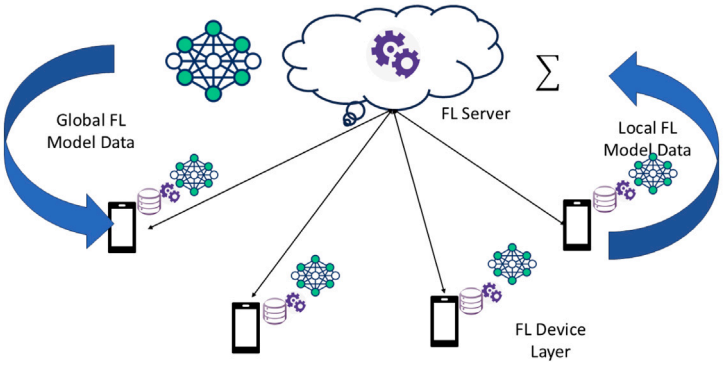


Fig. 4. Federated learning.

2.2.2. Advantages of FL

FL offers several advantages:

- **Enhanced Privacy:** Raw data remains on client devices, complying with strict privacy regulations [5].
- **Reduced Communication:** Only model parameters (not raw data) are exchanged, minimizing bandwidth usage [44].
- **Adaptability:** Models can continuously improve using real-time client data, making FL suitable for dynamic environments.

2.2.3. Challenges of FL

However, FL presents several challenges:

- **Communication Overhead:** Multiple synchronization rounds increase latency, particularly in bandwidth-constrained networks [2].
- **Client Heterogeneity:** Variations in computational resources and data distributions across clients can hinder convergence [45].
- **Model Complexity:** Training DNNs in FL settings often proves inefficient due to computational constraints on edge devices [46].

These limitations have motivated alternative approaches such as SL, which partitions model training between clients and servers to reduce computation/communication costs [12].

2.2.4. Vertical FL and connections to SL

Federated Learning is commonly categorized into horizontal FL (where clients share the same feature space but different samples) and vertical FL (where clients hold different features for the same set of samples). In vertical FL, multiple parties collaboratively train a model by aligning samples with overlapping IDs (e.g., through private set intersection) and exchanging intermediate representations or gradients without revealing raw features [47]. This paradigm shares notable similarities with Split Learning: both involve partitioning the model or features across parties, and both require the exchange of intermediate activations. In fact, SL can be viewed as a special case of vertical FL where one party (the client) holds input features and another (the server) holds the remaining model layers and possibly labels [12]. Conversely, vertical FL can be implemented using a split architecture where each party trains a part of the model and a neutral server aggregates updates [48]. Recent works have begun to combine vertical FL with split learning to handle scenarios where both features and labels are distributed across parties [49,50]. These hybrid approaches inherit the communication efficiency of SL and the privacy guarantees of vertical FL, but also face challenges in feature alignment, security, and scalability.

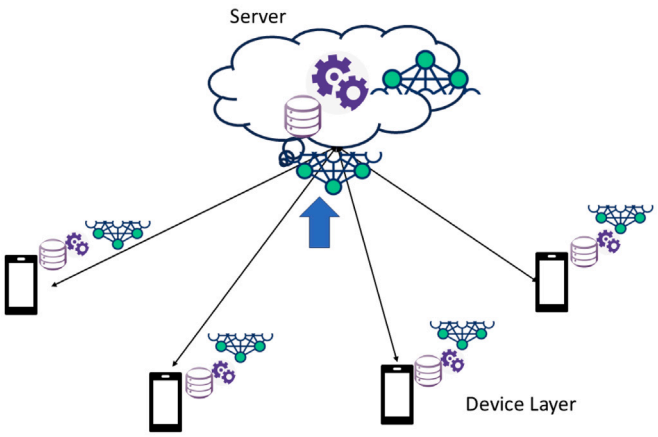


Fig. 5. Split learning.

2.3. SL

SL marks a significant shift in DML by introducing model partitioning to overcome the computational limitations inherent in FL. Unlike FL, which demands each client maintain a complete model replica, SL's innovative architecture allows resource-constrained devices to engage in collaborative training by executing only the initial layers of a neural network [12]. This makes SL especially valuable in EC scenarios, such as smart vehicles and IoT sensor networks, where devices often lack the substantial computational resources needed for full model training [1]. The core idea behind SL is to "split" a DL model into a smaller client-side portion and a larger server-side portion. Clients process only the initial layers on their local data, generating smashed data (intermediate activations) at a designated cut layer. This smashed data is then sent to the server, which completes the forward pass and performs the backward pass. Subsequently, the gradients for the client-side are returned to the client for local weight updates. This approach drastically lessens the computational load on individual devices as they no longer need to train an entire, potentially very large, model. However, a key limitation of vanilla SL is its sequential nature; only one client can interact with the server at a time. This leads to considerable latency and underutilization of server resources, further exacerbating communication delays. The frequent back-and-forth communication of smashed data and gradients for each training iteration also introduces its own form of communication overhead, potentially limiting SL's scalability in large-scale deployments. Fig. 5 shows the SL process, where a complex DL model is trained collaboratively across client and server nodes.

2.3.1. Architectural framework of SL

The SL framework partitions a global model W^p into two components:

- Client-side model $S(\cdot;\theta_S)$: Typically 1–3 shallow layers executed on edge devices
- Server-side model $M(\cdot;\theta_M)$: Remaining deep layers processed on central infrastructure

The training protocol follows an iterative four-phase process:

1. *Client Forward Pass*: Device i computes activations $H_i = S(D_i;\theta_S)$
2. *Server Forward Pass*: Central server generates predictions $\hat{y}_i = M(H_i;\theta_M)$
3. *Server Backward Pass*: Gradients $\nabla\theta_M$ and ∇H_i are calculated
4. *Client Backward Pass*: Device updates θ_S using $\nabla\theta_S = \nabla H_i \cdot \partial S/\partial\theta_S$

2.3.2. Advantages of SL

Three key benefits distinguish SL from conventional FL approaches:

- **Computational Efficiency**: By limiting client-side computation to shallow layers, SL reduces device workload by 10–100× compared to FL [51]. This enables participation from extremely resource-constrained devices like embedded sensors.
- **Enhanced Privacy**: The transmission of intermediate activations rather than raw gradients provides stronger privacy guarantees. As [52] demonstrates, reconstructing input data from smashed data requires solving computationally challenging inverse problems.
- **Bandwidth Optimization**: Empirical studies show smashed data is typically 3–20× smaller than FL's gradient updates [53], significantly reducing communication overhead in bandwidth-constrained networks.

2.3.3. Challenges of SL

Despite its advantages, SL introduces several unique constraints:

- **Sequential Bottleneck**: The alternating client–server computation creates inherent latency, increasing training time by 40%–60% compared to FL [1]. This makes SL less suitable for real-time applications.
- **Privacy-Utility Tradeoff**: While more private than FL, [52] showed that carefully crafted attacks can extract sensitive features from activations, particularly when split layers are shallow.
- **Architectural Rigidity**: Fixed split points often lead to suboptimal performance, with accuracy drops of 5%–15% compared to end-to-end training [54]. This has spurred research into dynamic splitting approaches.

These limitations have motivated hybrid approaches like FSL, which combine SL's computational benefits with FL's parallel processing advantages [1].

2.4. FSL

FSL stands out as a hybrid framework, combining the strengths of FL and SL to overcome their individual limitations, particularly concerning communication and on-device computation [1,19]. FSL tackles the substantial communication and computational burdens often found in FL by adopting SL's model-splitting approach while retaining FL's parallel client training. In FSL, shown in Fig. 6, the DL model is split into client-side and server-side portions. Clients process only the initial layers of the model on their local data, generating intermediate activations (smashed data). This significantly reduces the computational load on individual devices, a major advantage for resource-constrained edge and IoT environments. Unlike the sequential nature of traditional SL, FSL allows multiple clients to train their client-side model portions in parallel. Once their local computations are complete, they transmit their smashed data to a central server. The server then completes the forward pass, performs the backward pass, and aggregates the resulting gradients, either for the server-side model or by averaging client-side contributions. This parallel processing greatly reduces the overall training time compared to sequential SL. While FSL effectively lightens the computational strain on client devices, it still requires the communication of smashed data and corresponding gradients. This can be substantial, prompting ongoing research into communication optimization techniques like compression and sparsification to enhance FSL's efficiency in bandwidth-limited scenarios.

2.4.1. Architectural framework of FSL

The FSL workflow involves three key steps:

1. **Local Forward Pass:** Each client i computes intermediate activations $H_i = S(D_i)$ using their local data $\mathcal{D}_i$ and the client-side submodel S.
2. **Server-Side Aggregation:** The server aggregates these activations using FedAvg:

$$H_{\text{merged}} = \frac{1}{N}\sum_{i=1}^{N} H_i. \tag{2}$$

   The server then processes H_{merged} through the remaining model layers to compute the loss and gradients.
3. **Global Model Update:** The server back propagates gradients, updates the global model, and distributes the refined client-side submodel parameters to participants.

This mechanism decouples compute-intensive operations (server-side) from resource-limited clients while preserving data privacy, as raw data never leaves local devices [14].

2.4.2. pseudocode for a generic FSL training round

To complement the architectural description, Algorithm 1 presents a generic training round for a client participating in FSL. The algorithm highlights the interaction between client and server, the local forward/backward steps, and the aggregation phase. The notation used is as follows: $\mathcal{D}_i$ denotes the local dataset of client i; $S(\cdot;\theta_i)$ is the client-side model with parameters θ_i; $M(\cdot;\phi)$ is the server-side model with parameters ϕ; K is the number of local mini-batch steps per round; η is the learning rate; and Aggregate represents the server-side aggregation rule (e.g., FedAvg). This pseudocode captures the essence of FSL while abstracting away optional details such as compression, encryption, or asynchronous updates, which are discussed later in Section 3.4.

2.4.3. Protocol choices in FSL

The design of an FSL system involves several protocol-level decisions that affect performance, scalability, and reliability:

- **Synchronous vs. Asynchronous Updates:** In synchronous FSL, all clients complete their local steps and transmit updates before the server proceeds to the next round. This ensures consistency but can lead to straggler effects. Asynchronous FSL allows the server to update the global model as soon as a client's update arrives, reducing idle time but potentially causing stale gradients and slower convergence [55,56]. Hybrid approaches that combine synchronous and asynchronous phases are also explored.
- **Client Selection Strategies:** Selecting a subset of clients per round can reduce communication overhead and mitigate heterogeneity. Common strategies include random selection, importance sampling based on data size or loss, and mobility-aware selection in vehicular networks [18,57]. Clustering-based selection groups clients with similar capabilities or data distributions [58,59].

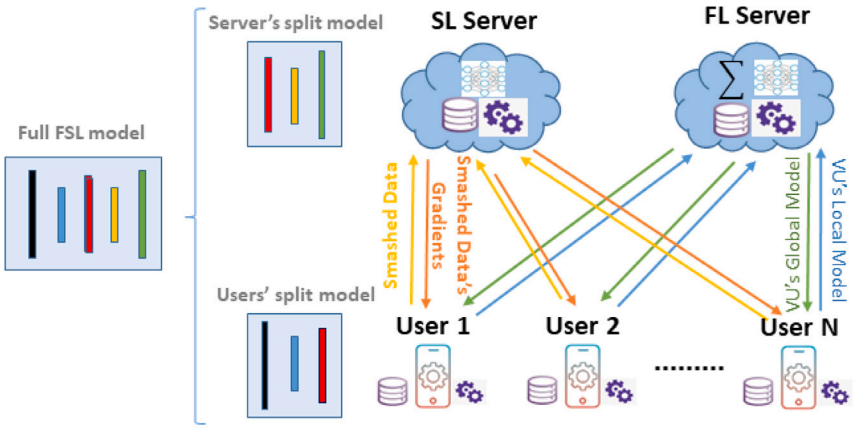


Fig. 6. Federated split learning.

Algorithm 1 Generic FSL Training Round for Client i

Input: Client i with local data $\mathcal{D}_i$, client-side model $S(\cdot;\theta_i)$, server-side model $M(\cdot;\phi)$, number of local steps K
Output: Updated client-side parameters θ_i', server-side parameters ϕ'

Server side: Receive current global parameters $\theta^{(t)}$ from server (or initialize)
Client side: Set $\theta_i \leftarrow \theta^{(t)}$
for $k = 1$ to K **do**
  Sample mini-batch $\mathcal{B} \subset \mathcal{D}_i$
  Compute smashed data $H_i = S(\mathcal{B};\theta_i)$
  Transmit H_i to server
  Server side: Receive H_i from client
  Compute forward pass: $\hat{y} = M(H_i;\phi)$
  Compute loss $\mathcal{L}(\hat{y}, y)$ and backpropagate to obtain gradients $\nabla_\phi \mathcal{L}$ and $\nabla_{H_i}\mathcal{L}$
  Update server parameters: $\phi \leftarrow \phi - \eta\nabla_\phi \mathcal{L}$
  Transmit gradient $\nabla_{H_i}\mathcal{L}$ back to client
  Client side: Receive gradient $\nabla_{H_i}\mathcal{L}$
  Backpropagate through S to obtain $\nabla_{\theta_i}\mathcal{L} = \nabla_{H_i}\mathcal{L} \cdot \frac{\partial H_i}{\partial \theta_i}$
  Update client parameters: $\theta_i \leftarrow \theta_i - \eta\nabla_{\theta_i}\mathcal{L}$
end for
Transmit updated θ_i to server for aggregation
Server side: Aggregate all client updates: $\theta^{(t+1)} = \text{Aggregate}(\{\theta_i\})$ (e.g., FedAvg)
Broadcast $\theta^{(t+1)}$ to all clients
return $\theta^{(t+1)}, \phi$

- **Aggregation Methods:** Federated Averaging (FedAvg) is the most common aggregation rule, but alternatives like FedProx, FedNova, and adaptive weighting can improve convergence under heterogeneity [60,61]. In FSL, aggregation may be applied to client-side model parameters, smashed data, or server-side gradients.
- **Communication Protocols:** The underlying communication framework significantly impacts latency and reliability. gRPC and MQTT are popular choices for real-time client–server interaction in edge deployments [62,63]. For wireless environments, protocols that support intermittent connectivity and quality-of-service guarantees are essential [10].
- **Split Point Adaptation:** The cut layer can be fixed or dynamically adjusted based on network conditions, client capabilities, or task requirements. Dynamic splitting strategies, often guided by reinforcement learning or online optimization, are an active area of research [15,57].

2.4.4. Advantages of FSL

FSL addresses critical limitations of standalone FL and SL:

- **Hybrid Efficiency:** Combines FL's distributed training scalability with SL's communication efficiency, reducing bandwidth usage by transmitting activations instead of full gradients [64].
- **Privacy Enhancement:** Intermediate activations expose less sensitive information than raw data or gradients, though privacy risks (e.g., inversion attacks) persist and require mitigation [51].
- **Client-Side Load Reduction:** Clients only execute a subset of layers, lowering memory and compute requirements compared to full local training in FL [65].

2.4.5. Challenges of FSL

Despite its benefits, FSL faces several challenges:

- **Convergence Speed:** The sequential nature of SL phases (local forward pass, server-side backward pass) can slow convergence, especially with heterogeneous client data or large-scale deployments [66].
- **Resource Imbalance:** Clients must still handle partial model training, which may be infeasible for severely constrained devices.
- **Cut-Layer Optimization:** The choice of where to split the model (the "cut layer") significantly impacts performance; suboptimal splits can lead to bottlenecks or accuracy degradation.

While FSL improves upon both FL and SL, it can still face slow convergence, especially in large-scale deployments [67]. To further optimize the learning process, TL can be integrated into the FSL framework, resulting in the FSTL methodology.

2.5. Transfer learning

TL is a powerful technique that improves model performance by leveraging knowledge gained from previous tasks, enabling efficient training on new but related tasks. TL is especially useful in scenarios with limited data or heterogeneous data distributions among clients, as it reduces the training required for convergence and enhances overall model accuracy [68]. By initializing clients with pre-trained models developed on large datasets, TL allows each client to start training from a more informed state, rather than from random initialization [69]. Fig. 7, shows the TL process, where a complex DL model is trained across local devices with the support of pre-trained model data transfer from a centralized server.

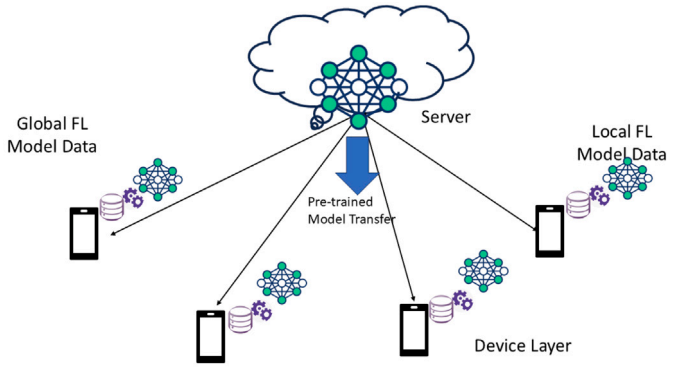


Fig. 7. Transfer learning.

Formally, let $\mathcal{D}_{source} = \{(x_i, y_i)\}_{i=1}^{N_s}$ represent the source domain with N_s samples and associated labels y_i. The objective is to learn a model $W^{p'}$ that minimizes the loss function L^{source} on this domain:

$$L^{source} = \frac{1}{N_s} \sum_{i=1}^{N_s} \mathcal{L}(y_i, f(x_i; W^{p'})) \tag{3}$$

where $\mathcal{L}$ is a chosen loss function (e.g., cross-entropy) and $f(x_i; W^{p'})$ is the model's prediction for input x_i.

Once trained on the source domain, the model is fine-tuned on the target domain $\mathcal{D}_{target} = \{(x_j, y_j)\}_{j=1}^{N_t}$ with N_t samples to minimize the loss function L^{target}:

$$L^{target} = \frac{1}{N_t} \sum_{j=1}^{N_t} \mathcal{L}(y_j, f(x_j; W^{p'})) \tag{4}$$

In the context of FSL, TL provides substantial benefits. By assigning a pre-trained model $W^{p'}$ to each client, the learning process is accelerated and the generalization of the model in varying datasets of clients is improved. This combination is particularly advantageous when client data distributions differ, as the initial pre-trained features and patterns offer a robust starting point for adaptation.

TL offers the following advantages [70]:

- **Accelerated Training:** TL reduces the training time required for model convergence, which is particularly advantageous in real-time or resource-constrained applications.
- **Improved Model Accuracy:** Leveraging learned knowledge from a related domain improves model performance, especially when training data is limited or heterogeneous.
- **Enhanced Generalization:** Pre-trained models offer better adaptability across diverse client datasets, as they can transfer features to new tasks effectively.

In the transition from FSL to FSTL, TL further enhances the learning framework. Integrating TL with FSL maximizes efficiency and scalability while preserving data privacy and the computational benefits of FSL. This synergy allows for faster and more adaptable model training in distributed environments, which is particularly beneficial for diverse applications with unique data characteristics.

2.6. FSTL

FSTL is an advanced collaborative learning paradigm that integrates three powerful techniques: FL, SL, and TL. FSTL extends the FSL framework by leveraging pre-trained models or external knowledge to improve learning efficiency, convergence speed, and generalization in heterogeneous and resource-constrained environments [51].

The motivation behind FSTL lies in the practical limitations encountered in many real-world applications. Devices participating in collaborative learning often have highly non-IID data, limited computational resources, or sporadic network connectivity. FSTL addresses these issues by:

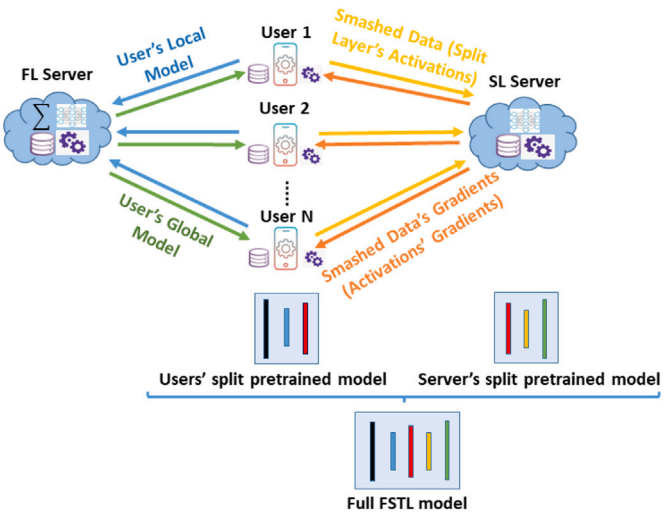


Fig. 8. Federated split transfer learning.

- Splitting deep models between client and server to reduce client-side computation (inherited from SL),
- Aggregating knowledge across clients without requiring raw data sharing (inherited from FL),
- Transferring knowledge from pre-trained models or auxiliary datasets to accelerate and enhance training (adopted from TL).

In a typical FSTL pipeline, shown in Fig. 8, clients (users) begin by initializing the lower portion of their model using pre-trained weights. These lower layers are used to process local data and generate intermediate representations (smashed data), which are then sent to the server for further processing. The server, which may also utilize pre-trained or frozen layers, completes the forward and backward passes and shares the gradients with the clients. A FedAvg or other aggregation strategy is periodically employed to merge updates across clients:

$$\theta^{(global)} = \frac{1}{N} \sum_{i=1}^{N} \theta^{(i)} \tag{5}$$

where $\theta^{(i)}$ represents the parameters of the local ith client model segment (typically the shared cut layer or lower layers).

Benefits of FSTL include:

- **Improved Convergence:** The use of pre-trained models accelerates convergence and reduces training time.
- **Reduced Training Burden:** Clients train only a portion of the model, often with fewer data, reducing the computational and memory load.
- **Adaptability:** Particularly suitable for scenarios involving limited or heterogeneous data, such as IoT, edge AI, and personalized medicine.
- **Cross-Domain Learning:** Facilitates knowledge transfer between domains, allowing models trained in one context to be fine-tuned in another (e.g., from general image datasets to specialized medical datasets).

Challenges of FSTL include:

- **Transfer Source Selection:** The choice of pre-trained models or source domains significantly impacts performance and must be chosen carefully.

- **Model Compatibility:** Ensuring consistency across different devices with varied architectures and initialization points is non-trivial.
- **Communication Bottlenecks:** Although client-side load is reduced, communication costs remain a challenge in low-bandwidth or dynamic networks.
- **Privacy Concerns:** Intermediate representations may leak information unless protected with techniques such as DP or homomorphic encryption.

Remarks on Fine-Tuning in FSTL. A critical design choice in FSTL is whether and how to fine-tune the pre-trained layers. The common practice is to freeze the lower (pre-trained) layers and fine-tune only the task-specific layers (e.g., the top classifier) or small adapter modules [71]. This approach preserves the general knowledge captured in the pre-trained weights while adapting to the target task with minimal computational overhead and privacy risk. However, when the source and target domains differ significantly (e.g., natural images to medical scans), full fine-tuning of all layers may be necessary to achieve acceptable accuracy [72]. Such extensive fine-tuning can increase the computational burden on clients and may expose more information through gradients, potentially weakening privacy guarantees. To mitigate these issues, parameter-efficient fine-tuning techniques like Low-Rank Adaptation (LoRA) [73] or prompt tuning [74] have been proposed. These methods introduce a small number of trainable parameters while keeping the pre-trained weights frozen, thereby balancing adaptation capability with efficiency and privacy. In FSTL, selecting the appropriate fine-tuning strategy involves trade-offs among accuracy, client computation, communication cost, and privacy leakage, and remains an open research area.

FSTL is recommended for applications in personalized services, mobile healthcare, smart manufacturing, and training large-scale models (e.g., LLM) in privacy-preserving, distributed, and resource-aware settings. In particular, Naseh et al. have introduced and practically demonstrated the concept of FSTL, where it was applied in challenging environments of the Non-Terrestrial Network (NTN) to support intelligent vehicular communication [9] and EO tasks [20]. These implementations highlight the flexibility of FSTL in enabling DML even under high-latency, bandwidth-limited, and heterogeneous device conditions, further reinforcing its potential as a scalable and adaptable paradigm for next-generation AI systems.

2.7. GFSTL

GFSTL is an advanced DML paradigm that builds upon the foundations of FSTL to address the limitations of scalability, heterogeneity, and personalization in large-scale, resource-constrained environments. By enabling a multi-group, hierarchical training mechanism, GFSTL introduces flexibility and parallelism in model training and communication processes, making it particularly suitable for scenarios such as intelligent vehicular systems, wireless sensor networks, satellite-based communications, and large-scale DL applications. The original concept and practical implementations were presented by Naseh et al. demonstrating the efficacy of GFSTL in complex and distributed NTN infrastructures, with applications spanning EO [23] and ITS [75].

2.7.1. Architectural framework of GFSTL

The GFSTL architecture enhances the FSTL framework by introducing a layered, group-based topology. As shown in Fig. 9, the user space is partitioned into m independent groups (users batches), where each group hosts its own local SL server. Within each group, users employ an identical model architecture and communicate exclusively with their group's SL server. Training begins with each user receiving a pre-trained local model copy. The forward pass is computed locally, and the resulting smashed data (intermediate outputs) are transmitted to the SL server. The server completes the backward pass, sending gradients back to users for local model updates.

Once all users in a group complete their updates, the group's SL server performs local model aggregation (e.g., via FedAvg). These aggregated submodels are then transmitted to a centralized FL server, which integrates the updates from all groups into a global model. This pipeline supports both horizontal and vertical model splits, accommodates non-Independent and Identically Distributed (non-IID) data, and enables integration with privacy-preserving and secure learning protocols. The framework is inherently modular, allowing seamless extension across domains with high communication or energy constraints.

2.7.2. Advantages of GFSTL

GFSTL offers several key benefits:

- **Hierarchical Scalability:** The multi-group setup distributes computational and communication loads, allowing deployment across vast networks with thousands of participants.
- **Privacy-Conscious Computation:** As with FSTL, user data never leaves the device. Only smashed data is shared, thus maintaining strong privacy guarantees.
- **Parallel Processing:** Independent group operations reduce overall training time and allow real-time model updates in dynamic environments.
- **Cross-Group Knowledge Transfer:** Aggregating submodels at the FL server ensures that knowledge acquired across diverse data groups benefits the global model.
- **Communication Efficiency:** Group-level training and limited message passing minimize bandwidth usage, an essential property for NTN and EC scenarios.

2.7.3. Challenges of GFSTL

Despite its strengths, several challenges persist:

- **Global Coordination Latency:** Synchronization delays between SL and FL servers may affect training speed and convergence.
- **Heterogeneous Hardware Constraints:** Differences in computational power or energy availability across users and groups can introduce imbalance in training dynamics.
- **Information Leakage Risks:** Although GFSTL transmits only smashed data, advanced attacks might still infer private information, requiring robust privacy-preserving techniques.
- **System Deployment Complexity:** Setting up and managing multiple SL servers, especially in remote or bandwidth-constrained areas, can be logistically challenging.
- **Hyperparameter Sensitivity:** Fine-tuning architectural and training parameters becomes more complex due to the hierarchical and distributed nature of GFSTL.

In conclusion, GFSTL provides a promising and versatile architectural solution for distributed AI in environments that demand high scalability, personalization, and privacy. Its modular design, combined with the hierarchical training process, empowers the system to operate efficiently across diverse domains, from space-based sensing to intelligent ground networks. The pioneering works [23,75] have laid the groundwork for future advancements in this area, offering a robust blueprint for real-world deployments of intelligent, distributed systems.

2.8. Unified workflow comparison of FSL variants

To provide a clear visual comparison of the different distributed learning paradigms discussed in this section, Fig. 10 illustrates the information flow in FL, SL, FSL, FSTL, and GFSTL. Each subfigure depicts the key components, clients (or groups) and servers, and the direction of exchanged data (activations, gradients, or model updates). This unified view helps to understand the architectural evolution from basic FL and SL to the more advanced hierarchical GFSTL.

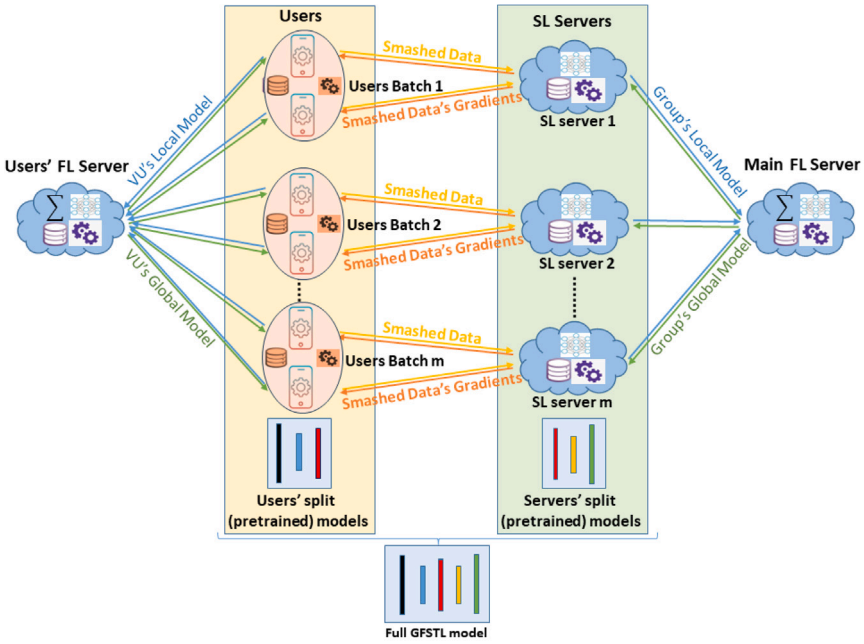


Fig. 9. Generalized federated split transfer learning.

2.9. Latency analysis and comparison

In this section, we analyze the overall latency of different DML methods, such as FL, SL, FSL, FSTL, and GFSTL. The overall latency is represented by the combined computation and communication time. It is important to note that all methods utilize a uniform data distribution. Let us assume that d represents the total size of the data, h denotes the size of the smashed layer, R signifies the data rate based on the communication medium, T indicates the training time required to train the entire Deep Neural Network (DNN) model, while T' and T'' represent the training time in FSTL and GFSTL using a pre-trained TL model. Furthermore, T_{FedAvg}, T_{Merge}, T_{UFL}, and T_{MainFL} represent the time required to perform the aggregation of the full model in FL, the smashed parameters on the user side, and the main FL servers, respectively. Furthermore, q denotes the total number of parameters in the full model and r means the ratio of the size of the user submodel to that of the full models. The summarized results are presented in Table 4. The presence of factor 2 in several terms, such as $2pr$, $\frac{2dh}{n}$, etc., is due to the download and upload of user-side model updates before and after training. Numerical analysis reveals that when there is a large number of clients, SL can become inefficient, since the total time is directly proportional to n, mainly due to the serial training operation, whereas other methods perform the training in parallel. Furthermore, we can see that as n increases, the total time cost increases in this order: $FSTL < FSL < FL < SL$. To be more specific, FSTL is faster than FSL because it uses a pre-trained network ($T'' < T' < T$), and FSL has a lower latency than FL because it aggregates fewer parameters ($T_{Merge} < T_{FedAvg}$). Regarding GFSTL, the total latency depends on the number of users in the jth group, that is, n_j. We used the operator $\max(\cdot)$ since the total communication time depends on the slowest group.

In conclusion, this analysis suggests that both FSTL and GFSTL provide substantial latency improvements, especially when scaling to larger user groups within DML scenarios. These gains stem from pre-trained model usage and efficient parameter aggregation, outperforming conventional FL and SL approaches in latency-sensitive applications.

2.10. Lessons learned: Comparison of advantages and challenges

This section synthesizes the key insights from the comparative analysis of DML paradigms presented in this chapter. As detailed in Table 5, each technique, FL, SL, FSL, TL, FSTL, and GFSTL, exhibits a distinct profile of trade-offs between scalability, privacy, computational efficiency, and communication overhead. FL excels in scalability through parallel client training but imposes a high computational burden on clients and significant communication costs for exchanging full model parameters. In contrast, SL drastically reduces client-side computation and enhances privacy by transmitting only intermediate activations (smashed data), but its sequential training nature severely limits scalability and increases latency. FSL emerges as a hybrid solution, balancing the parallelization benefits of FL with the computational and privacy advantages of SL, though it introduces new challenges such as convergence speed and optimal cut-layer selection. The integration of TL into FSL, resulting in FSTL, further accelerates convergence and improves generalization by leveraging pre-trained models, but requires careful handling of model compatibility and source domain selection. Finally, the GFSTL paradigm introduces a hierarchical, multi-group architecture that offers superior scalability and privacy for large-scale, heterogeneous environments, albeit at the cost of increased system complexity and global coordination latency. This comparative overview underscores that there is no one-size-fits-all solution; the choice of a DML paradigm must be carefully aligned with the specific constraints and objectives of the target application, such as the scale of deployment, device capabilities, network conditions, and stringency of privacy requirements. Having established this foundational taxonomy of DML paradigms, the next section will delve into the specific and pressing challenges that researchers face when implementing FSL in real-world systems.

3. Challenges in FSL

Federated Split Learning (FSL) has emerged as a promising paradigm that blends the privacy-preserving capabilities of FL with the computational efficiency of SL. While this hybrid approach holds immense potential for enabling distributed ML across edge devices, it introduces multifaceted challenges spanning privacy, system heterogeneity, communication efficiency, and computational constraints. These challenges are deeply interconnected, privacy mechanisms influence model utility, device heterogeneity affects communication latency, and optimization strategies must carefully balance performance, security, and efficiency. This section provides a comprehensive analysis of

Table 4
Latency analysis of five DML methods per round.

Learning method	Training + Aggregation time	Communications per user/server	Total communications/server	Total communication time	Total latency
FL	$T + T_{\text{FedAvg}}$	$2q$	$2nq$	$\frac{2q}{R}$	$T + T_{\text{FedAvg}} + \frac{2q}{R}$
SL	T	$\frac{2dh}{n} + 2qr$	$2dh + 2nqr$	$\frac{2dh}{R} + \frac{2nqr}{R}$	$T + \frac{2dh}{R} + \frac{2nqr}{R}$
FSL	$T + T_{\text{Merge}}$	$\frac{2dh}{n} + 2qr$	$2dh + 2nqr$	$\frac{2dh}{nR} + \frac{2qr}{R}$	$T + T_{\text{Merge}} + \frac{2dh}{nR} + \frac{2qr}{R}$
FSTL	$T' + T_{\text{Merge}}$	$\frac{2dh}{n} + 2qr$	$2dh + 2nqr$	$\frac{2dh}{nR} + \frac{2qr}{R}$	$T' + T_{\text{Merge}} + \frac{2dh}{nR} + \frac{2qr}{R}$
GFSTL	$T'' + T_{\text{UFL}} + T_{\text{MainFL}}$	$\frac{2dh}{n_j} + 2qr$	$2dh + 2n_j qr$	$\max\left\{\frac{2dh}{n_j R}\right\} + \frac{2qr}{R}$	$T'' + T_{\text{UFL}} + T_{\text{MainFL}} + \max\left\{\frac{2dh}{n_j R}\right\} + \frac{2qr}{R}$

Table 5
Comparison of distributed machine learning techniques: Advantages and disadvantages.

Technique	Scalability	Privacy	Computational efficiency	Communication overhead
FL	+ High scalability due to parallel model training across clients. – Limited by full model update aggregation on the server.	+ Data remains on local devices (enhances privacy). – Vulnerable to model inversion and poisoning attacks.	– High computation requirement due to full model training on each client.	– High communication overhead with full model parameter exchange.
SL	+ Suitable for scenarios with fewer clients. – Limited scalability due to serialized client training.	+ Raw data remains private (smashed data shared only). – Some data leakage can occur via smashed data.	+ Reduced computational burden on client side. – Server-side computation burden remains high.	+ Lower communication overhead (only smashed data is sent). – High latency due to serial model updates.
FSL	+ More scalable than SL with parallelized FL elements. – Limited scalability compared to pure FL.	+ Enhanced privacy through combined FL and SL benefits. – Privacy concerns due to server–client split.	+ Reduced client-side computational load. – Server-side computational load remains significant.	+ Lower communication overhead compared to FL (smashed data only). – Higher latency than FL due to split training.
TL	+ Applicable for diverse domains; increases scalability in heterogeneous data settings.	+ Pre-trained model can mask some sensitive information.	+ Reduces computational load by leveraging pre-trained models. – Limited by domain compatibility of source and target tasks.	+ Lower communication overhead due to pre-trained model usage.
FSTL	+ More scalable than FSL, due to faster model convergence using TL.	+ High data privacy with TL masking and combined FL–SL protections.	+ Enhanced efficiency by utilizing pre-trained model splits.	+ Lower communication overhead due to smaller model splits. + Improved latency by combining parallel FL updates and TL.
GFSTL	+ Highly scalable due to hierarchical multi-server structure with multiple client groups.	+ High data privacy preservation via group separation and SL protections.	+ Efficient model convergence with pre-trained models and reduced client computation.	+ Reduced latency by parallel updates within groups and lower bandwidth requirements per user.

the primary challenges in FSL, organizing them into six categories: privacy and security, heterogeneity, computational and system constraints, communication overhead, model optimization, and networking challenges. For each challenge, we present a structured analysis comprising problem formulation, taxonomy of solution strategies, representative methods, a comparative table summarizing key approaches, and a discussion of trade-offs and open issues.

3.1. Privacy and security

The distributed nature of FSL exacerbates existing security concerns, making privacy-preserving techniques a critical research area. The unique architectural design of FSL presents novel attack surfaces, including data reconstruction, gradient leakage, and inference attacks. We organize privacy and security challenges into three categories: privacy leakage, security threats, and overheads introduced by countermeasures.

3.1.1. Problem formulation

In FSL, clients share intermediate activations (smashed data) with the server rather than raw data or full gradients. While this provides stronger privacy guarantees than FL in principle, several attack vectors remain:

- **Data reconstruction attacks:** Adversaries attempt to recover input data from shared intermediate activations.
- **Label inference attacks:** Attackers deduce class labels from activation patterns, particularly in non-IID settings.
- **Gradient inversion attacks:** Even partial gradient information can be used to reconstruct training samples.
- **Model poisoning and backdoor attacks:** Malicious clients inject poisoned updates to corrupt the global model.
- **Membership inference:** Adversaries determine whether a specific data point was used in training.

The challenge lies in designing defense mechanisms that provide strong privacy guarantees while minimizing computational and communication overhead and maintaining model accuracy.

3.1.2. Taxonomy of solution strategies

Privacy-preserving techniques for FSL can be categorized into four main families:

1. **Differential Privacy (DP):** Injecting calibrated noise into data, activations, or gradients to provide mathematical privacy guarantees. Sub-categories include:
    - Input-level DP: Noise added to raw data before processing
    - Activation-level DP: Noise added to smashed data before transmission
    - Gradient-level DP: Noise added to gradients during backpropagation
    - Label-level DP: Noise added to labels to protect against label inference
2. **Cryptographic Techniques:** Using encryption to protect data during computation and transmission:

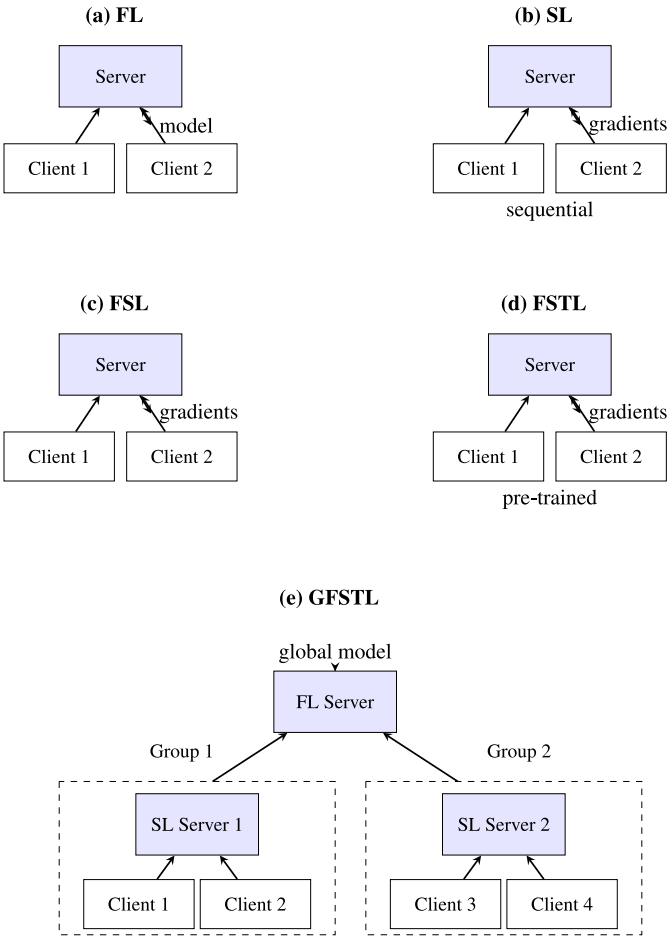


Fig. 10. Unified workflow comparison of distributed learning paradigms: (a) Federated learning, (b) Split learning with sequential client participation, (c) Federated split learning, (d) Federated split transfer learning, and (e) Generalized federated split transfer learning. In FL, FSL, and FSTL, multiple clients communicate in parallel with the server; SL is sequential (only one client interacts at a time); GFSTL organizes clients into groups, each with its own SL server, and aggregates submodels at a central FL server.

- Homomorphic Encryption (HE): Enables computation on encrypted data
- Secure Multi-Party Computation (SMPC): Distributes computation across multiple parties
- Secret Sharing: Splits sensitive information into shares

3. **Architectural Defenses:** Leveraging the split architecture itself for privacy:

   - Deeper split points: Moving more layers to the client side to reduce exposed information
   - Activation obfuscation: Transforming activations before transmission
   - Feature compression: Reducing information content of shared data

4. **Hybrid Approaches:** Combining multiple techniques for enhanced protection:

   - DP + HE: Encrypted computation with formal privacy guarantees
   - DP + secret sharing: Distributed noise injection
   - Split architecture + trusted execution environments

3.1.3. Representative methods

- **Data Reconstruction and Activation Leakage:** Yang et al. [56] introduce the GAS framework, demonstrating how activation sharing in asynchronous FSL can be exploited by adversaries to infer private data. Their proposed buffering mechanism reduces bias in staggered client updates but does not entirely mitigate inference risks, suggesting the need for stronger cryptographic protections such as homomorphic encryption or SMPC [64].
- **Label Inference in Non-IID Settings:** The SCALA framework [13] investigates information leakage pathways in concatenated activation schemes, revealing that adversaries can deduce class distributions even when raw data is obfuscated. Their logit adjustment technique balances model performance with privacy protection.
- **Gradient Inversion Attacks:** Zhao et al. [76] highlight that LLMs trained within FSL systems are vulnerable to gradient inversion attacks, where sophisticated reconstruction techniques can recover high-dimensional input features even from partial gradient updates. Their proposed Low-Rank Adaptation (LoRA) reduces attack vectors by lowering parameter exposure.
- **Differential Privacy Integration:** Ndeko et al. [25] introduce FSL-DP for Human Activity Recognition (HAR), injecting calibrated noise into smashed activations. Thapa et al. [1] integrate DP and PixelDP within the SplitFed architecture. Shu et al. [77] propose a GAN-powered multi-objective optimization for privacy-preserved FSL in IIoT.
- **Robustness to Network Imperfections:** Shiranthika et al. [78] evaluate SplitFed resilience to packet loss, finding that deeper split points yield statistically significant accuracy advantages under lossy conditions. Kafshgari et al. [79] propose smart averaging to maintain accuracy under noisy channels.
- **Label Privacy:** Zhang et al. [80] introduce Edge-assisted U-shaped SFL (EUSFL) with Label-Differential-Privacy (LabelDP) to protect label information against reconstruction attacks.

3.1.4. Comparative analysis

Table 6 compares representative privacy-preserving techniques for FSL across key dimensions.

3.1.5. Key trade-offs and open issues

The analysis reveals several fundamental trade-offs:

- **Privacy vs. Accuracy:** Stronger privacy guarantees (e.g., larger DP noise, encryption) typically degrade model accuracy. Adaptive mechanisms that tune privacy budgets based on data sensitivity show promise.
- **Privacy vs. Efficiency:** Cryptographic techniques provide strong protection but introduce significant computational and communication overhead, limiting their applicability in resource-constrained edge deployments.
- **Architectural choices:** Deeper split points improve privacy by exposing less information but increase client-side computation, potentially negating FSL's computational benefits.
- **Threat model coverage:** Most existing techniques address specific attack vectors rather than providing comprehensive protection. Hybrid approaches combining multiple mechanisms are needed but increase system complexity.

Open research directions include lightweight cryptographic protocols tailored for FSL, adaptive privacy mechanisms that respond to real-time threat detection, and formal privacy guarantees for split architectures.

3.2. Heterogeneity

The diversity of client devices, data distributions, and network conditions in FSL leads to performance degradation unless adaptive mechanisms are employed to mitigate hardware and software disparities.

Table 6
Comparison of privacy-preserving techniques in FSL.

Technique	Privacy mechanism	Computational overhead	Communication overhead	Accuracy impact	Threat model
FSL-DP [25]	Activation-level DP	Low	Low	Moderate (tunable)	Data reconstruction
SplitFed + DP [1]	Gradient-level DP	Low	Low	Low–Moderate	Multiple
GAS [56]	Buffering + obfuscation	Low	Low	Minimal	Activation leakage
SCALA [13]	Logit adjustment	Low	None	Improves	Label inference
EUSFL [80]	LabelDP + U-shaped split	Low	Low	Minimal	Label leakage
HE-based FSL [64]	Homomorphic encryption	High	High	None	All passive
SMPC-based FSL [64]	Secure multi-party computation	High	High	None	Malicious
GAN-GA DP [77]	Adaptive DP + GAN	Moderate	Low	Tunable	Multiple

3.2.1. Problem formulation

Heterogeneity in FSL manifests in three primary forms:

- **System heterogeneity:** Clients have varying computational capabilities, memory constraints, energy budgets, and communication bandwidth.
- **Statistical heterogeneity:** Data distributions across clients are non-IID, with variations in class balance, feature distributions, and sample sizes.
- **Network heterogeneity:** Clients experience diverse network conditions, including varying latency, bandwidth, and reliability.

These heterogeneities can lead to straggler effects, where slow clients delay global aggregation, biased model updates, and poor convergence.

3.2.2. Taxonomy of solution strategies

Approaches to handling heterogeneity in FSL can be categorized as:

1. **Dynamic Architecture Adaptation:** Adjusting model split points and client-side submodel sizes based on client capabilities.
2. **Client Clustering and Grouping:** Partitioning clients into homogeneous groups for efficient training.
3. **Resource-Aware Scheduling:** Selecting clients and allocating resources based on their capabilities and network conditions.
4. **Personalization:** Adapting global models to local data distributions through fine-tuning or personalized layers.

3.2.3. Representative methods

- **Dynamic Architecture Adaptation:** Sun et al. [15] introduce Heterogeneous Split Federated Learning (HSFL), which utilizes dynamic cut-layer selection to optimize resource allocation across heterogeneous devices. Their hybrid optimization pipeline combines Sample Average Approximation, Genetic Algorithm, and Lagrangian Relaxation.
- **Client Clustering:** Liao et al. [58] propose ParallelSFL, which handles both statistical and system heterogeneity by clustering clients based on data distribution (KL-divergence) and device capability, achieving 21% traffic reduction and 1.36× speedup. Samikwa et al. [3] introduce DFL, dynamically clustering clients based on submodel similarity and resource profiles.
- **Resource-Aware Scheduling:** Ao et al. [19] formulate network-aware optimization using Mixed-Integer Non-Linear Programming (MINLP) to balance communication and computation. Wazzeh et al. [81] propose CRSFL, clustering devices by capability and using genetic algorithms for optimal client selection.
- **Personalized Approaches:** Han et al. [14] present a hybrid FSL framework balancing personalization and generalization. Pervej and Molisch [82] propose Personalized Hierarchical SFL (PHSFL) with shared feature extractors and client-specific classifiers.
- **Memory-Aware Adaptation:** Tian et al. [83] develop SmartSplit, a hierarchical memory optimization framework achieving 94% latency reduction and 100× memory reduction on mobile SoCs.

3.2.4. Comparative analysis

Table 7 compares representative heterogeneity-handling techniques.

3.2.5. Key trade-offs and open issues

- **Adaptation granularity:** Coarse-grained adaptation (e.g., per-round split selection) has lower overhead but may not capture fine-grained heterogeneity. Fine-grained adaptation (e.g., per-client dynamic splits) increases complexity.
- **Clustering stability:** Dynamic clustering improves adaptability but may cause instability if cluster assignments change frequently.
- **Personalization vs. generalization:** Personalized approaches improve per-client performance but may reduce global model coherence.

Open issues include handling heterogeneity in federated settings with massive client populations, adapting to concept drift in non-stationary environments, and theoretical guarantees for convergence under heterogeneous conditions.

3.3. Computational and system constraints

Edge devices participating in FSL often face stringent resource limitations, necessitating innovative techniques for memory management, efficient computation, and workload balancing.

3.3.1. Problem formulation

Resource constraints in FSL include:

- **Memory limitations:** Many edge devices have limited RAM (e.g., 192 kB in smart meters [84]), making it impossible to store full models.
- **Computational capacity:** CPU/GPU constraints limit the number of operations per second.
- **Energy budgets:** Battery-powered devices require energy-efficient computation.
- **Storage constraints:** Limited flash storage for model parameters and intermediate data.

Table 7
Comparison of heterogeneity-handling techniques in FSL.

Technique	Approach	Heterogeneity type	Performance gain	Overhead	Assumptions
HSFL [15]	Dynamic cut-layer	System	Latency↓37%, accuracy↑	Moderate	Known client profiles
ParallelSFL [58]	Clustering + grouping	Statistical + system	Traffic↓21%, speed↑1.36×	Low	Stable data distributions
DFL [3]	Dynamic clustering	Data + resource	Time↓48%, energy↓62.8%	Moderate	Real-time profiling
CRSFL [81]	Capability clustering + GA	System	Overhead↓, accuracy maintained	Low	Authentication tasks
SmartSplit [83]	Memory-aware partitioning	Memory constraints	Latency↓94%, memory↓100×	Low	Mobile SoCs
PHSFL [82]	Personalized heads	Statistical	Personalization↑	Low	Label availability

Table 8
Comparison of resource-optimization techniques in FSL.

Technique	Optimization	Resource Saved	Accuracy Impact	Deployment	Scalability
MP-FSL [66]	Channel pruning	Storage↓32%, compute↓27%	+1.35%	Real FL platform	Moderate
Smart meter FSL [84]	End–edge–cloud split	Memory↓95.5%, time↓94.8%	Maintained	Smart meters	Low (specific)
SmartSplit [83]	Hierarchical memory	Memory↓100×, latency↓94%	+1.5%–57%	Mobile SoCs	Moderate
AFSL [55]	Asynchronous updates	Latency↓	Maintained	Edge devices	High
CHEESE [59]	D2D clustering + load balancing	Delay↓	Maintained	Edge networks	High

3.3.2. Taxonomy of solution strategies

1. **Model Pruning and Compression:** Reducing model size before or during training.
2. **Hierarchical Offloading:** Distributing computation across end–edge–cloud tiers.
3. **Memory-Efficient Execution:** Techniques like checkpointing and activation caching.
4. **Asynchronous Training:** Reducing synchronization overhead.

3.3.3. Representative methods

- **Model Pruning:** Jia et al. [66] present MP-FSL, integrating adaptive channel pruning to compress DNN channels by 32.2% in storage and 26.73% in computation before splitting.
- **Hierarchical Frameworks:** Li et al. [84] propose an end–edge–cloud FSL framework for smart meters, achieving 95.5% memory reduction and 94.8% training time reduction. Khan et al. [85] develop a hierarchical split FL framework with multi-level aggregation.
- **Memory Optimization:** Tian et al. [83] enable deep models on resource-constrained devices through hierarchical memory management and cost-aware checkpointing.
- **Asynchronous Learning:** Albuquerque et al. [55] propose AFSL, an asynchronous FSL framework incorporating parameter decay to maintain consistency despite device heterogeneity.

3.3.4. Comparative analysis

Table 8 compares resource-optimization techniques.

3.3.5. Key trade-offs and open issues

- **Pruning vs. accuracy:** Aggressive pruning reduces resource usage but may degrade accuracy; adaptive pruning strategies help.
- **Hierarchical offloading vs. latency:** Multi-tier computation reduces device load but introduces additional communication latency.
- **Asynchrony vs. convergence:** Asynchronous updates improve efficiency but may cause model drift.

Open issues include lightweight memory management for ultra-constrained devices, energy-aware scheduling, and hardware–software co-design for FSL.

3.4. Communication overhead

Minimizing communication costs remains a key challenge in FSL, particularly for mobile and bandwidth-limited environments.

3.4.1. Problem formulation

Communication overhead in FSL stems from:

- **Smashed data transmission:** Intermediate activations sent from clients to server.
- **Gradient transmission:** Gradients sent from server back to clients.
- **Model updates:** Aggregated model parameters distributed to clients.
- **Control messages:** Synchronization and coordination overhead.

3.4.2. Taxonomy of solution strategies

1. **Gradient Compression:** Quantization, sparsification, and pruning of gradients.
2. **Activation Compression:** Reducing size of smashed data through encoding.
3. **Communication-Efficient Aggregation:** Last-layer aggregation, selective transmission.
4. **Asynchronous Protocols:** Reducing synchronization rounds.
5. **Caching and Reuse:** Storing intermediate results to avoid retransmission.

Table 9
Comparison of communication optimization techniques in FSL.

Technique	Approach	Communication reduction	Accuracy impact	Computation overhead	Applicability
EPSL [10]	Last-layer aggregation + resource opt.	Significant (latency↓)	Maintained	Low	Wireless edge
FedSL [87]	Layer-group aggregation	42%–70%	Maintained	Low	CNN-based
ESL [86]	Activation caching	3.92×	+35%	Moderate	IoT
CSE-FSL [88]	Auxiliary networks + sparse updates	High	Maintained	Moderate	General
SFETEC [89]	Thing-edge–cloud split	Marked reduction	Maintained	Low	Edge–cloud

3.4.3. Representative methods

- **Gradient Compression:** Lin et al. [10] propose EPSL, leveraging last-layer gradient aggregation and joint optimization of wireless resources, achieving significant latency reduction.
- **Activation Compression:** Chawla et al. [86] develop ESL with key–value store for activation caching, achieving 3.92× communication reduction and 35% accuracy improvement.
- **Layer-wise Aggregation:** Zhang et al. [87] propose FedSL with layer-group aggregation, reducing communication by 42% (VGG) and 70% (ResNet-9).
- **Storage-Efficient Communication:** Mu and Shen [88] introduce CSE-FSL with auxiliary networks, transmitting smashed data only in designated epochs.
- **Mobility-Aware Communication:** Zheng et al. [57] optimize FSL for vehicular networks with mobility-aware selection.

3.4.4. Comparative analysis

Table 9 compares communication optimization techniques.

3.4.5. Key trade-offs and open issues

- **Compression vs. accuracy:** Aggressive compression may lose information; adaptive compression rates help.
- **Caching vs. freshness:** Cached activations may become stale; cache invalidation strategies needed.
- **Asynchrony vs. consistency:** Reduced communication rounds may slow convergence.

Open issues include optimal compression for heterogeneous networks, integration with network slicing, and theoretical bounds on communication efficiency.

3.5. Model optimization

Effective model optimization in FSL systems requires careful attention to architecture design, parameter efficiency, and training dynamics.

3.5.1. Problem formulation

Key optimization challenges include:

- **Cut-layer selection:** Choosing optimal split points balancing client–server load.
- **Convergence speed:** Slow convergence due to sequential phases and heterogeneity.
- **Parameter efficiency:** Reducing model size while maintaining accuracy.
- **Personalization:** Adapting global models to local data distributions.

3.5.2. Taxonomy of solution strategies

1. **Dynamic Cut-Layer Selection:** Adapting split points during training.
2. **Model Pruning and Compression:** Reducing parameter count.
3. **Parallel Training Techniques:** Client clustering and parallelization.
4. **Personalized Layers:** Client-specific adaptations.
5. **Decentralized Consensus:** Server-less aggregation.

3.5.3. Representative methods

- **Dynamic Cut-Layer:** Sun et al. [15] (HSFL) and Zheng et al. [57] (MSFTL) demonstrate adaptive split selection.
- **Model Pruning:** Jia et al. [66] (MP-FSL) achieves significant size reduction.
- **Parallel Training:** Liao et al. [58] (ParallelSFL) and Liao et al. [90] (MergeSFL) achieve speedups through client clustering and feature merging.
- **Personalized Optimization:** Han et al. [91] (SplitGP) embed Gaussian Process modules for personalization. Qin et al. [92] (SpFedRec) use SE blocks for recommendation.
- **Decentralized Consensus:** Tedeschini et al. [11] (SCFL) abolish central servers using MPNN-inspired message passing.

3.5.4. Comparative analysis

Table 10 compares model optimization techniques.

3.5.5. Key trade-offs and open issues

- **Dynamic splits vs. stability:** Frequent split changes may cause instability; optimal adaptation frequency unknown.
- **Pruning vs. representation power:** Aggressive pruning may limit model capacity.
- **Personalization vs. generalization:** Local adaptations may diverge from global objectives.

Open issues include theoretical convergence guarantees for dynamic architectures, automated neural architecture search for FSL, and optimization for foundation models.

3.6. Networking challenges in FSL

The distributed nature of FSL introduces unique networking challenges that significantly impact performance and scalability. These challenges go beyond the simple communication of model updates and require careful consideration of the underlying network infrastructure, especially in wireless and mobile environments.

3.6.1. Problem formulation

Networking challenges in FSL can be categorized into four main areas:

Table 10
Comparison of model optimization techniques in FSL.

Technique	Optimization strategy	Performance gain	Computation overhead	Communication impact	Best suited for
HSFL [15]	Dynamic cut-layer	Latency↓37%, accuracy↑	Low	Reduced	Heterogeneous edge
MP-FSL [66]	Channel pruning	Size↓32%, accuracy↑1.35%	Low	Reduced	Resource-constrained
ParallelSFL [58]	Client clustering	Speed↑1.36×, traffic↓21%	Moderate	Reduced	Edge clusters
MergeSFL [90]	Feature merging	Speed↑1.39–4.14×, accuracy↑5.8%–26%	Moderate	Reduced	Non-IID data
SplitGP [91]	Gaussian Process	Inference time↓, OOD accuracy↑	Moderate	Unchanged	Personalized FL
SCFL [11]	Decentralized consensus	Central server eliminated	Moderate	Increased	Distributed inference

- **Communication Bottlenecks:** FSL involves multiple rounds of smashed data and gradient transmissions between clients and servers. In large-scale deployments with thousands of devices, the aggregated communication load can saturate network links, leading to increased latency and reduced throughput [10,93].
- **Network Heterogeneity:** Clients connect via diverse links (e.g., 5G, Wi-Fi, wired) with vastly different characteristics in terms of bandwidth, latency, jitter, and reliability. This heterogeneity can cause straggler effects, where a few slow clients delay global synchronization [19,85].
- **Mobility and Intermittent Connectivity:** In vehicular networks, mobile health, and other dynamic scenarios, clients frequently change their point of attachment or experience temporary disconnections. This mobility disrupts ongoing training and requires mechanisms for state migration and session continuity [9,57,94].
- **Network Security:** The communication channels are vulnerable to various attacks, including man-in-the-middle, eavesdropping, denial-of-service (DoS), and session hijacking. These attacks can compromise data privacy, model integrity, and system availability [64].

3.6.2. Taxonomy of solution strategies

Existing solutions to networking challenges in FSL can be grouped into the following categories:

1. **Joint Communication and Computation Optimization:** Formulating optimization problems that jointly consider model split points, resource allocation (bandwidth, power), and client scheduling to minimize latency or energy [10,93,95].
2. **Mobility-Aware Protocols:** Designing handover mechanisms that preserve training state, predictive scheduling based on mobility patterns, and location-aware client selection [57,94].
3. **Adaptive Scheduling and Load Balancing:** Dynamically selecting clients and balancing workloads based on real-time network conditions, device capabilities, and data importance [19,59,96].
4. **Resilient Communication:** Employing techniques such as redundant transmissions, erasure coding, and fast retransmission to cope with packet loss and disconnections [78,79].
5. **Secure Communication Protocols:** Integrating encryption, authentication, and intrusion detection to protect the communication fabric [64].

3.6.3. Representative methods

- **Joint Resource Allocation:** Zhu et al. [95] propose ESFL, which jointly optimizes model partition and wireless resource allocation (subchannel, power) to minimize per-round latency. Xu et al. [93] formulate a non-convex optimization problem for split-point selection and bandwidth allocation, solved via alternating optimization. Lin et al. [10] introduce EPSL, which parallelizes client training and aggregates only last-layer gradients while jointly optimizing wireless resources.
- **Mobility Management:** Zheng et al. [57] develop a mobility-aware split-federated framework (MSFTL) for vehicular semantic communication, using Stackelberg games for resource allocation based on vehicle residence time. Moon and Lim [94] propose a dual-cut scheme with state migration, allowing seamless handover between edge servers in vehicular edge computing.
- **Adaptive Scheduling:** Zhang et al. [96] propose Group-based SFL (GSFL), which organizes clients into groups for parallel training and sequential group updates, reducing overall latency. Cheng et al. [59] introduce CHEESE, a D2D clustering-based framework that uses potential game theory for cluster formation and bandwidth allocation.
- **Resilience to Packet Loss:** Shiranthika et al. [78] evaluate SplitFed resilience to packet loss, demonstrating that deeper split points maintain higher accuracy under lossy conditions. Kafshgari et al. [79] propose smart averaging to mitigate the impact of noisy channels.
- **Secure Communication:** Zhang et al. [64] analyze privacy and efficiency trade-offs in FSL communications and discuss the use of homomorphic encryption and SMPC to secure data in transit.

3.6.4. Key trade-offs and open issues

The analysis reveals several fundamental trade-offs:

- **Communication vs. Computation:** Joint optimization often trades off communication and computation; minimizing one may increase the other. Finding the Pareto-optimal balance is an ongoing challenge.
- **Mobility Handling vs. Overhead:** State migration ensures training continuity but introduces additional signaling and data transfer overhead. Lightweight handover protocols are needed.
- **Security vs. Performance:** Strong encryption and authentication increase latency and computational load, which may be unacceptable for real-time applications. Lightweight cryptographic primitives tailored for FSL are required.
- **Adaptability vs. Stability:** Highly adaptive scheduling can react to network fluctuations but may cause instability in training convergence. Online learning-based scheduling policies show promise.

Open research directions include: (i) network-aware neural architecture search that considers communication constraints, (ii) integration with 5G/6G network slicing to guarantee quality of service for FSL traffic, (iii) theoretical performance bounds for mobile FSL, and (iv) cross-layer optimization that spans the physical, MAC, and application layers.

3.7. Standardization landscape and implications for FSL

As FSL matures toward practical deployment, standardization efforts across various bodies are becoming increasingly relevant. Standardization ensures interoperability, security, and performance benchmarks, facilitating widespread adoption in real-world systems, particularly in telecommunications and edge computing environments.

3.7.1. Overview of standardization activities

Several international standards organizations are actively working on aspects relevant to FSL:

- **3GPP (3rd Generation Partnership Project):** Working groups SA1, SA3, and SA6 are addressing federated learning and distributed AI for 5G-Advanced and 6G networks. Key documents include:
  - TR 23.700-91: Study on enablers for network automation for 5G [97].
  - TS 23.288: Architecture for enabling edge applications [98].
  - SA3 work on security for network automation (TR 33.818) [99].
- **ITU-T (International Telecommunication Union – Telecommunication Standardization Sector):** Focus Groups on Machine Learning for Future Networks (FG-ML5G) and AI for Energy Efficiency (FG-AI4EE) have produced recommendations:
  - ITU-T Y.3172: Architectural framework for machine learning in future networks including IMT-2020 [100].
  - ITU-T Y.3173: Framework for evaluating intelligence levels of future networks [101].
- **IEEE:** Standard P3652.1 (Guide for Architectural Framework and Application of Federated Machine Learning) provides guidelines for FL architecture, data management, and security [102].
- **ETSI (European Telecommunications Standards Institute):** Industry Specification Groups on Experiential Networked Intelligence (ISG ENI) and Multi-access Edge Computing (ISG MEC) are exploring ML integration:
  - ETSI GR ENI 007: Experiential Networked Intelligence (ENI) – Terminology for Main Concepts in ENI [103].
  - ETSI GR MEC 017: MEC Support for AI/ML [104].
- **O-RAN Alliance:** The RAN Intelligent Controller (RIC) and xApps/rApps frameworks enable ML-based optimization in Open RAN, creating opportunities for FSL deployment in radio access networks. Relevant specifications include O-RAN.WG1.EADP-v02.00 [105].
- **IETF (Internet Engineering Task Force):** The draft "draft-irtf-nmrg-ai-challenges" addresses challenges of integrating AI into network management, including federated learning considerations [106].

3.7.2. Mapping FSL components to standards

Table 11 maps key FSL components to relevant standardization activities.

Table 11
Mapping of FSL components to standardization bodies and activities.

FSL component	Relevant standards bodies	Standardization activities
Client–server communication	3GPP (SA2, SA3), IETF	5G/6G protocols, security, QoS (TR 23.700-91, TS 23.288)
Privacy-preserving mechanisms	IEEE P3652.1, ITU-T (SG17)	Federated learning architecture, privacy guidelines (IEEE P3652.1)
Model aggregation	3GPP (SA6), ETSI MEC	Edge computing APIs, service orchestration (ETSI MEC 017)
Split point selection	IEEE, O-RAN Alliance	RAN intelligent controllers, xApps (O-RAN.WG1)
Security and trust	3GPP (SA3), IETF (IRTF)	Authentication, encryption, threat mitigation (TR 33.818)
Performance benchmarks	ITU-T (FG-ML5G), IEEE	ML performance metrics, evaluation frameworks (Y.3172, Y.3173)
Interoperability	All bodies	Cross-vendor compatibility, API standardization

3.7.3. Implications for FSL deployment

Standardization has several implications for FSL research and deployment:

- **Interoperability:** Standards ensure that FSL implementations from different vendors can work together, crucial for multi-vendor network deployments and edge–cloud federation.
- **Security and Privacy:** Standardized security protocols (e.g., 3GPP's authentication frameworks) and privacy mechanisms (e.g., IEEE P3652.1's differential privacy guidelines) provide baseline protection and facilitate regulatory compliance (GDPR, HIPAA).
- **Performance Benchmarks:** Common metrics and evaluation frameworks (e.g., ITU-T Y.3173) enable fair comparison of different FSL approaches and accelerate technology maturation.
- **Deployment Acceleration:** Standards reduce integration effort, allowing developers to focus on algorithmic innovations rather than low-level protocol design.
- **Regulatory Alignment:** Standards help align FSL with emerging regulations through certified privacy-preserving techniques and accountability frameworks.

3.7.4. Open standardization challenges

Despite progress, several standardization challenges remain:

- **Lack of FSL-Specific Standards:** Most existing standards address FL broadly; FSL-specific aspects like split point selection, cut-layer optimization, and hybrid client–server coordination lack standardized interfaces and protocols.
- **Cross-Domain Coordination:** FSL spans telecommunications, edge computing, and AI domains, requiring coordination across multiple SDOs (3GPP, ITU-T, IEEE, ETSI, O-RAN) that often work in silos.
- **Rapid Evolution:** The fast pace of FSL research makes it difficult for standards to keep up with innovations; a balance between stability and flexibility is needed.
- **Certification and Compliance:** Mechanisms for certifying FSL implementations against standards are underdeveloped, hindering trust in commercial deployments.
- **Testbeds and Conformance:** Lack of standardized testbeds and conformance testing procedures makes it hard to validate compliance.

3.7.5. Future directions

Future work should focus on:

- Developing FSL-specific extensions to existing FL standards (e.g., IEEE P3652.1 revision).
- Establishing cross-SDO liaisons to harmonize FSL-related activities.
- Creating open-source reference implementations aligned with standards to accelerate adoption.

- Engaging the research community with standards bodies to ensure that innovations translate into deployable solutions.

3.8. Lessons learned

FSL has made significant progress in balancing privacy, efficiency, and scalability, but critical challenges remain. Key lessons from the literature include:

- **Privacy-security trade-offs:** While privacy-preserving techniques like DP and homomorphic encryption enhance security, they often introduce computational overhead that hampers real-world deployment. Hybrid approaches combining lightweight cryptography with architectural defenses show promise.
- **Heterogeneity demands adaptability:** System and data heterogeneity require adaptive solutions, current approaches like dynamic cut-layer selection and client clustering improve resource allocation but struggle in highly dynamic environments.
- **Resource constraints drive innovation:** Memory-efficient execution, model pruning, and hierarchical offloading enable FSL on highly constrained devices, with demonstrated improvements of 100× memory reduction and 160× faster load distribution.
- **Communication efficiency is paramount:** Techniques like layer-wise aggregation, activation caching, and GAN-driven allocation achieve communication reductions of 42–1623× while preserving accuracy.
- **Standardization gaps:** While standards for FL are emerging, FSL-specific aspects lack coverage, creating opportunities for community input to standards bodies.

Looking ahead, the field must prioritize lightweight cryptographic hybrids that maintain security without sacrificing performance, RL-driven adaptation for real-time adjustments, decentralized aggregation protocols for scalability, and active engagement with standards bodies. By addressing these gaps, FSL can evolve into a robust framework capable of supporting next-generation distributed AI applications.

4. FSL applications

FSL has emerged as a potent paradigm for collaboratively training DL models while preserving data privacy and addressing the computational constraints of edge devices. Over the past few years, a diverse array of research studies have explored and extended FSL to various application domains. To provide a structured overview, we categorize these applications by deployment context rather than traditional domain labels. This organization highlights common challenges and design patterns across similar environments. We identify five primary contexts: the Edge–IoT continuum (resource-constrained, wireless edge), high-mobility networks (vehicular, UAV), regulated domains (healthcare), large-scale model training (LLMs, foundation models), and remote sensing (Earth observation, non-terrestrial networks). Table 12 summarizes the mapping from original application domains to these deployment contexts and highlights the dominant constraints in each setting. Tables provided in the next subsections summarize the key features, challenges addressed, techniques employed, evaluation settings, and publication details for each area.

4.1. Edge–IoT Continuum

The Edge–IoT continuum represents a class of deployments where resource-constrained devices (sensors, wearables, smart meters) are connected via wireless networks to edge servers or cloud infrastructure. These environments are characterized by limited computational capacity, memory constraints, energy budgets, and variable wireless channel conditions. FSL is particularly well-suited for this context because it offloads the bulk of model computation to the edge server while keeping data local, preserving privacy and reducing the on-device footprint. This subsection synthesizes works presented in Sections 4.1.1 (IoT and Edge Computing) and 4.1.2 (Wireless Networks), as they share common challenges in resource optimization, communication efficiency, and heterogeneity management.

Table 12
Mapping of original application domains to deployment contexts.

Original domain	Deployment context	Dominant constraints
IoT and Edge computing	Edge–IoT continuum	Resource limitations, wireless connectivity, latency sensitivity
Wireless networks	Edge–IoT continuum	Bandwidth variability, channel impairments, device mobility
Healthcare	Regulated/Sensitive domains	Strict privacy laws, heterogeneous medical data, interpretability
Vehicular networks	High-Mobility networks	Rapid topology changes, intermittent connectivity, safety requirements
Large language models	Large-Scale model training	Massive parameter counts, communication overhead, memory constraints
Earth observation	Remote sensing and NTN	High data volumes, long-range communication, energy constraints

4.1.1. IoT and Edge Computing

FSL has emerged as a powerful paradigm to address the unique challenges of deploying collaborative ML in resource-constrained and privacy-sensitive IoT and edge environments. By partitioning models between client (resource-limited devices) and server (edge or cloud), FSL combines the strengths of FL and SL to reduce raw data exchange, lower computational burden on devices, and enhance privacy. In the following subsections, we reorganize and discuss thirty-seven recent papers according to the primary challenge each addresses: (1) Privacy and Security, (2) Heterogeneity, (3) Computational and System Constraints, (4) Communication Overhead, and (5) Model Optimization. Table 13 summarizes the key features, challenges addressed, techniques employed, evaluation settings, and publication details for all works in this area.

Privacy and Security

Authors in [25] introduce the FSL with DP (FSL-DP) framework for Human Activity Recognition (HAR) on edge networks. FSL-DP splits the HAR model between clients (wearable devices capturing accelerometer and gyroscope data) and a server, injecting calibrated noise into the smashed activations to satisfy DP constraints. Experimental comparisons against standard FL show that FSL-DP achieves faster communication rounds and improved accuracy and loss metrics, while the authors explore the privacy–utility trade-off across multiple data settings on a real HAR dataset.

Thapa et al. introduce SplitFed (SFL) [1], a hybrid of FL and SL that preserves raw-data privacy via model partitioning, while leveraging relay-based training across clients to parallelize computation. They further embed DP and PixelDP mechanisms to harden model robustness. Empirical results on multiple image-classification benchmarks demonstrate that pure SFL matches SplitNN's accuracy and communication efficiency but reduces per-epoch computation time by over 30%, and its privacy-enhanced variant maintains these gains under stricter DP budgets.

Focusing on industrial IoT applications, [77] proposes a privacy-preserved FSL architecture that applies a differential-privacy-based perturbation mechanism to intermediate activations. They formalize a multi-objective optimization balancing privacy budgets and model

Table 13
Summary of papers on FSL in IoT and edge computing.

Ref #	Title	Short description	Challenges & measures	Model/ Technique	Datasets/ Benchmarks	Eval. metrics	Scalability & deployment	Type/Year
[15]	*Split Federated Learning Over Heterogeneous Edge Devices: Algorithm and Optimization*	Parallelizes training across heterogeneous IoT clients.	Latency minimization (37%↓), convergence boost (22%↑).	SAA, GA, Lagrangian Relaxation, B&B	Synthetic non-IID	Latency, convergence, accuracy	Simulations for heterogeneous devices	Conference, 2025
[66]	*Model Pruning-enabled Federated Split Learning for Resource-constrained Devices*	Prunes DNN channels before split to save resources.	Storage (32.2%↓) and computation (26.73%↓) reduction.	Adaptive channel pruning, heterogeneous aggregation	Real FL platform	Accuracy, resource usage	Real-world FL deployment	Journal, 2024
[19]	*Federated Split Learning for Distributed Intelligence with Resource-Constrained Devices*	Jointly optimizes device scheduling, power, split layer.	Training delay and energy consumption minimization.	Mixed-integer nonlinear programming, alternating optimization	Wireless IoT simulation	Delay, energy, throughput	Edge–cloud testbed simulations	Journal, 2024
[1]	*SplitFed: When Federated Learning Meets Split Learning*	Combines FL and SL with DP and PixelDP.	Compute time per epoch↓, communication efficiency↑.	SplitFed architecture, DP	Multiple image datasets	Accuracy, time per epoch	Multi-client SL/FL prototype	Conference, 2022
[53]	*End-to-End Evaluation of Federated Learning and Split Learning for Internet of Things*	Empirical FL vs. SplitNN on Raspberry Pis.	Comm. overhead↓ for FL; accuracy trade-offs under non-IID.	FL, SplitNN benchmarking	Raspberry Pi, various datasets	Training time, comm. traffic, power	Physical IoT deployment	Conference, 2020
[87]	*FedSL: A Communication Efficient Federated Learning With Split Layer Aggregation*	Layer-group aggregation to cut traffic.	42%↓ comm. (VGG), 70%↓ (ResNet-9).	FedSL algorithm, Max–Min selection	CNN benchmarks	Comm. cost, accuracy	Edge cluster experiments	Journal, 2024
[59]	*CHEESE: Distributed Clustering-Based Hybrid Federated Split Learning*	D2D-clustered hybrid FL/SL for delay minimization.	Training delay cost↓ via clustering, MSA, bandwidth alloc.	Potential game theory, heuristic MSA, convex alloc.	Practical models	Delay, throughput	Large-scale edge network testbed	Journal, 2023
[25]	*Federated Split Learning for HAR with Differential Privacy*	Applies DP to split activations for HAR.	Comm. time↓ per round; privacy-utility trade-off.	FSL-DP framework	Real HAR dataset	Accuracy, loss, comm. time	Edge network deployment	Journal, 2025
[84]	*Introducing Edge Intelligence to Smart Meters via FSL*	End–edge–cloud split for memory-limited meters.	Memory (95.5%↓), time (94.8%↓), comm. (50%↓).	Hierarchical FSL partitioning	Real hardware (192 kB SRAM)	Footprint, time, accuracy	Smart meter prototype	Journal, 2024
[83]	*Breaking the Memory Wall for Heterogeneous FL via Model Splitting*	Hierarchical SmartSplit with adaptive cut-layer.	Latency (94%↓), memory (100×↓), accuracy (1.49–57.18%↑)	Hierarchical manager, cost-aware check-pointing	Mobile SoC testbeds	Latency, memory, accuracy	Commercial edge hardware	Journal, 2024
[58]	*ParallelSFL: A Novel Split Federated Learning Framework*	Clustered split training with utility-based grouping.	Traffic (21%↓), speed (1.36×↑), accuracy (5%↑).	Clustering, submodel splits	80-node Jetson cluster	Comm., time, accuracy	Physical edge cluster	Conference, 2024
[89]	*SFETEC: Split-Federated Scheme for Thing-Edge-Cloud*	Tiered base/core split across client–edge–cloud	Comm.↓, training time↓, privacy↑.	SFETEC split strategy	Preliminary experiments	Time, traffic	Edge–cloud prototype	Conference, 2024
[77]	*Privacy-preserved Federated Split Learning in IIoT*	GAN-GA-powered DP perturbation in FSL.	Balance privacy budget vs. accuracy on Cifar-10.	Intermediate DP, multi-objective GAN-GA	Cifar-10	Accuracy, privacy metrics	IIoT simulation	Conference, 2024
[28]	*Edge–Cloud Computing and FS Learning in IoT*	Special issue survey of FL/SL in edge–cloud IoT.	Coverage of privacy, efficiency, and application use cases.	Literature compilation	Twelve research articles	N/A	Academic special issue	Journal, 2024
[107]	*Split-FL: An Efficient Online FL Framework with Constrained Computation and Streaming Data*	Adaptive pruning for streaming data in edge FL.	Computation and comm. trade-off optimized online.	Online learning, adaptive pruning	Streaming multimedia datasets	Latency, accuracy	Edge networks simulation	Journal, 2024

(*continued on next page*)

Table 13 (*continued*).

[108]	*A deep cut into Split Federated Self-supervised Learning*	Aligns momentum and online client models for privacy	Efficiency.	Comm. overhead↓ while preserving accuracy. and Contrastive alignment	Vision datasets	Accuracy, comm. rounds	Distributed training emulation	Conference, 2024
[81]	*CRSFL: Cluster-based Resource-aware Split FL in Resource-constrained Edge Networks*	Clusters by device capability for continuous authentication.	Overhead burden↓, accuracy maintained.	Genetic-optimized client selection	UMDAA-02-FD	Accuracy, latency	Smartphone testbed	Journal, 2024
[109]	*AdaptSFL: Adaptive Split FL in Edge Networks*	Theoretically controls split and aggregation adaptively.	Time–accuracy trade-off↓, latency↓.	Convergence analysis, adaptive control	Standard FL benchmarks	Time, accuracy	Edge network simulations	Journal, 2024
[86]	*Beyond Federated Learning for IoT: Efficient Split Learning with Caching*	Caches activations and personalizes models for SL efficiency.	Compute (1623×↓), comm. (3.92×↓), accuracy (35%↑).	Key–value store, model customization	Image and 3D segmentation	Compute, comm., accuracy	Resource-constrained devices	Preprint, 2024
[11]	*Split Consensus Federated Learning: An Approach for Distributed Training and Inference*	Decentralized consensus via MPNN-inspired message passing.	No central server; convergence of global FL.	Gossip-like message passing	Cooperative positioning use case	Accuracy, convergence	Real-world networked clients	Journal, 2024
[3]	*DFL: Dynamic Federated Split Learning in Heterogeneous IoT*	Dynamic clustering and resource-aware splitting.	Training time (48%↓), accuracy (32%↑), energy (62.8%↓).	Resource-aware split, dynamic clustering	Real IoT testbed	Time, accuracy, energy	Physical IoT deployment	Journal, 2024
[110]	*Have Your Cake and Eat It Too: Toward Efficient and Accurate Split Federated Learning*	Adaptive sliding split and data-balanced batching.	Inference accuracy (16.5%↑), training speed (3.54×↑).	Sliding split strategy	Image benchmarks	Accuracy, time	AIoT prototype setups	Conference, 2023
[111]	*RingSFL: An Adaptive Split FL Toward Taming Client Heterogeneity*	Ring-topology split propagation to tame stragglers.	Convergence↑, privacy↑ under non-IID.	Ring propagation	IID/non-IID benchmarks	Convergence, privacy risk	Simulation and prototype	Journal, 2023
[92]	*A Split-Federated Learning and Edge–Cloud based Efficient and Privacy-preserving Large-scale Item Recommendation Model*	Two-tower split with SE blocks and secret sharing.	Compute (23%↓), comm. (49%↓).	Squeeze-Excitation, secret sharing	Real recommendation datasets	Time, accuracy, privacy	Cloud–edge experimental setup	Journal, 2023
[112]	*FedSplitX: Federated Split Learning for Computationally-Constrained Heterogeneous Clients*	Multi-point splitting for FM client heterogeneity.	Comm.↓, performance↑ vs. size-limited baselines.	Auxiliary networks	Foundation model tasks	Accuracy, comm.	Server–client testbed	Journal, 2023
[88]	*Communication and Storage Efficient Federated Split Learning*	Single global model, auxiliary updates, sparse epochs.	Comm. and storage↓ with convergence guarantees.	Auxiliary network, sparse transmission	FL benchmarks	Comm., storage, accuracy	Simulated FL tasks	Journal, 2023
[113]	*A Joint Communication and Learning Framework for Hierarchical Split Federated Learning*	Two-tier HSFL with RLA-aware resource allocation.	Local accuracy, latency, energy jointly optimized.	MINLP decomposition and BSUM	IoT sensor data	Accuracy, latency, energy	Edge hierarchy simulation	Journal, 2023
[114]	*Federated Split Learning with Only Positive Labels for Resource-constrained IoT Environment*	Random shuffling and local BN for positive-label IoT.	Convergence improvements (32.6×–51.5×) on CIFAR variants.	SFPL algorithm	CIFAR-10/100	Accuracy, convergence rate	IoT device emulation	Journal, 2023

(*continued on next page*)

performance, and solve it using a generative adversarial network–powered genetic algorithm (GAN-GA). Cifar-10 experiments demonstrate that their approach maintains high classification accuracy while ensuring strong privacy guarantees, with privacy budgets tuned to the deployment scenario.

Under unreliable network conditions, [78] evaluate the robustness of SplitFed Learning (SFL) against packet loss by comparing shallow and deep model split points in a medical image segmentation task. Their findings indicate that deeper split points, where more of the model resides on the client, yield statistically significant accuracy advantages under various packet-loss rates, offering practical guidance for privacy-sensitive deployments in lossy IoT networks.

In [92], the authors develop SpFedRec, a split-federated recommendation framework for large-scale item recommendation systems. By migrating the item-tower computation to cloud servers and compressing transmitted gradients via secret sharing, SpFedRec reduces client

Table 13 (*continued*).

[115]	*Energy and Latency-aware Computation Load Distribution of Hybrid Split and Federated Learning on IoT Devices*	Clustering-based load distribution for hybrid SL+FL.	Search time (160×↓), balanced energy/latency.	Adaptive clustering	Simulation	Time, energy	Large-scale IoT simulation	Conference, 2023
[116]	*Split Federated Learning and Reinforcement-based Codec Switching in Edge Platform*	Optimal split	RL-based codec switching for video offload.	Training time↓ under Wi-Fi/LTE fluctuations., O-SFL, RLCS	Video datasets	Latency, throughput	Edge video testbed	Conference, 2023
[117]	*splitDyn: Federated Split Neural Network for Distributed Edge AI Applications*	Round-Robin client scheduling, Hungarian layer assignment.	Segmentation accuracy↑, processing time↓.	splitDyn scheduling	Incident prediction datasets	Accuracy, time	Edge AI prototype	Journal, 2022
[118]	*Training Latency Minimization for Model-Splitting Allowed Federated Edge Learning*	Regression-aided cut-layer modeling, alt. optimization.	Round latency↓ without accuracy loss.	Regression modeling, alternating optimization	Efficient-NetV2 MNIST	Latency, accuracy	Edge learning experiments	Preprint, 2025
[119]	*Predictive GAN-Powered Multi-Objective Optimization for Hybrid Federated Split Learning*	GAN-driven Pareto front for split, bandwidth, compute.	Time–energy trade-off optimized.	GAN-based multi-objective optimizer	Wireless testbed	Time, energy, Pareto optimality	Simulated wireless networks	Conference, 2023
[91]	*SplitGP: Achieving Both Generalization and Personalization in Federated Learning*	Client-side personalization, server-side GP generalization.	Inference time↓, OOD accuracy↑.	Gaussian Process modules	Vision benchmarks	Latency, accuracy	Edge-AI scenarios	Journal, 2023
[78]	*SplitFed Resilience to Packet Loss: Where to Split, that is the Question*	Shallow vs. deep split comparison under loss.	Deep split accuracy↑ under packet-loss.	SplitFed variants	Medical image segmentation	Accuracy, robustness	Network emulation	Conference, 2023
[80]	*EUSFL: U-Shaped Edge-Assisted Split FL*	Three-segment U-shaped split with LabelDP noise.	Training time↓, privacy↑ in IoT.	Label-DP mechanism	Heterogeneous IoT datasets	Time, accuracy, privacy	Edge-assisted IoT deployment	Journal, 2025

computation time by 23% and communication cost by 49% compared to conventional two-tower federated recommenders, all while leveraging Squeeze-and-Excitation blocks to enhance feature representation without exposing user data.

Lastly, [80] introduce Edge-assisted U-shaped SFL (EUSFL), which partitions neural networks into three segments, client head, server body, and client tail, and adds a novel Label-Differential-Privacy (LabelDP) noise mechanism to protect label information against reconstruction attacks. Benchmarking on heterogeneous IoT devices shows that EUSFL reduces overall training time and on-device computation overhead substantially, without compromising model accuracy.

Heterogeneity

Sun et al. in [15] address device-level heterogeneity through Heterogeneous SFL (HSFL). HSFL allows clients with varying computational resources to train different portions of the model in parallel by selecting distinct cut layers per device. The authors formulate a joint latency minimization problem over computation and communication resources, solved via a hybrid Sample Average Approximation, Genetic Algorithm, Lagrangian Relaxation, and Branch-and-Bound pipeline. Simulations under non-IID data demonstrate that HSFL converges faster and achieves higher accuracy than baseline FL and SL methods.

Liao et al. propose Parallel SFL (ParallelSFL) in [58] to handle both statistical and system heterogeneity in edge clusters. ParallelSFL splits the global model into base and top submodels, groups clients into clusters based on data distribution and device capability, and assigns cluster-specific update frequencies. Experiments on 80 NVIDIA Jetson boards show at least 21% reduction in network traffic, a 1.36× speedup in training time, and a minimum 5% boost in accuracy over standard FedAvg.

Dynamic FSL (DFL), introduced in [3], jointly tackles data and resource heterogeneity by dynamically clustering clients based on submodel similarity and resource profiles, and adaptively distributing split model segments. Real-world IoT testbed results reveal that DFL speeds up training by up to 48%, improves accuracy by up to 32%, and cuts energy consumption by up to 62.8% compared to classic FL and static FSL.

Yan et al. in [110] present Sliding SFL (S2FL) for AIoT systems. S2FL dynamically adjusts the split point for each device according to its computation power and uses data-balance batching to mitigate label imbalance across clients. On benchmark image datasets, S2FL achieves a 16.5% improvement in inference accuracy and a 3.54× acceleration in training compared to conventional SFL.

Shen et al. propose Ring-based SFL (RingSFL) in [111], which arranges clients in a logical ring and propagates split model segments sequentially. By tuning propagation lengths, RingSFL alleviates the straggler effect and blends local updates for enhanced privacy. Experiments on IID and non-IID datasets confirm improved convergence rates and stronger resistance to model inversion attacks.

Shin et al. introduce FedSplitX in [112] to accommodate large Foundation Models (FMs) among heterogeneous clients. FedSplitX enables multi-point splitting: clients choose cut layers matching their resource constraints, training only lightweight submodels while offloading heavy computations to the server. Auxiliary networks at each split point further reduce communication overhead. The approach outperforms size-limited baselines in both accuracy and resource utilization.

Tian et al. in [83] develop SmartSplit, a memory-aware FL framework that hierarchically manages split selection. A central server selects participants and cut layers based on memory budgets and data profiles, edge managers refine splitting via real-time statistical importance, and

on-device managers employ cost-aware checkpointing. On commercial mobile SoCs, SmartSplit cuts latency by up to 94%, reduces memory use 100×, and improves accuracy by 1.49%–57.18%.

Finally, Cheng et al. propose CHEESE in [59], a clustering-based hybrid FSL framework for edge networks. Devices form D2D clusters, split the model via a Model Splitting and Allocation (MSA) strategy, and collaboratively train segments. A designated transmission client uploads the merged model, while potential game theory, heuristic MSA, and greedy bandwidth allocation jointly minimize training delay. Large-scale experiments validate significant delay reductions in heterogeneous deployments.

Computational and System Constraints

In [28], Duan et al. provide a comprehensive survey on deploying Federated and Split Learning frameworks atop edge–cloud IoT infrastructures. They cover methods for jointly optimizing computation offloading, data privacy, and system security across heterogeneous edge and cloud layers. They highlight trends in resource-aware orchestration, privacy-performance trade-offs, and end-to-end deployment architectures, setting the stage for future practical implementations.

Jia et al. in [66] present Model Pruning-enabled FSL (MP-FSL), which integrates adaptive channel pruning into the FSL pipeline to compress DNN channels by up to 32.2% in storage and 26.73% in computation before splitting. MP-FSL also introduces a tailored aggregation algorithm that reconciles the heterogeneous, pruned client models. Deployed on a real FL platform, MP-FSL achieves up to 1.35% higher accuracy than state-of-the-art baselines while substantially lowering resource consumption on client devices.

Li et al. propose an end–edge–cloud FSL framework in [84] to enable intelligent load forecasting on memory-limited smart meters with only 192 kB of SRAM. By jointly optimizing model partitioning across end devices, edge servers, and the cloud, their approach achieves a 95.5% reduction in memory footprint, a 94.8% cut in training time, and a 50% decrease in communication overhead. Experimental validation on real hardware demonstrates forecasting accuracy equivalent to high-capacity centralized models.

Ao et al. in [19] develop FedSL, a novel framework for resource-constrained IoT devices that jointly optimizes power allocation, device scheduling, and split-layer selection via a mixed-integer nonlinear programming formulation. An alternating optimization algorithm tackles this complex problem with low computational overhead, yielding significant reductions in both training delay and energy consumption under realistic wireless network conditions.

Wazzeh et al. introduce Cluster-based Resource-aware SFL (CRSFL) in [81] to support continuous authentication on smartphones and wearables. CRSFL first clusters devices by computational capability, then concurrently trains split model segments across each cluster while a genetic algorithm selects the optimal participant set. Evaluations on the UMDAA-02-FD facial authentication dataset demonstrate that CRSFL preserves high authentication accuracy while reducing computational and communication burdens compared to centralized and standard federated approaches.

Lin et al. propose AdaptSFL in [109], offering a comprehensive convergence analysis that quantifies the effects of split-layer depth and client-side aggregation frequency on learning performance. AdaptSFL dynamically adapts these parameters based on real-time resource availability and latency constraints, demonstrating faster attainment of target accuracies and lower end-to-end training times than fixed-split baselines across multiple EC scenarios.

Hossan et al. in [115] address hybrid SL+FL computation load distribution for heterogeneous IoT devices by leveraging adaptive clustering. Their method groups devices with similar resources, determines optimal split points via scalable cluster-based analysis, and reduces load distribution search time by an average factor of 160× over brute-force techniques while balancing energy consumption and latency.

Khoa et al. present splitDyn in [117], a dynamic SL solution that applies Round-Robin scheduling to client selection and the Hungarian algorithm for optimal layer-to-client assignment. splitDyn dynamically adjusts model partitions per round, achieving improvements in segmentation accuracy and reduced processing time compared to static split schemes in edge AI applications.

Wen et al. in [118] propose a training latency minimization technique for Model-Splitting Allowed Federated Edge Learning (SFL). By fitting regression models to characterize cut-layer effects on computation and communication parameters, and then alternatingly optimizing cut-layer selection and server resource allocation, the method significantly lowers per-round latency on EfficientNetV2/MNIST without degrading final model accuracy.

Karjee et al. in [116] combine Optimal SFL (O-SFL) with RL-based Codec Switching (RLCS) to handle fluctuating network conditions during video frame offloading. O-SFL determines split points that minimize training time under current bandwidth, while RLCS proactively switches video codecs using a learned policy, yielding reduced end-to-end training latency on both Wi-Fi and LTE networks.

Finally, Liu et al. introduce Split-FL in [107], an online FL framework tailored for streaming data scenarios. Split-FL balances computational load and communication overhead through adaptive pruning of model parameters and an online update procedure, demonstrating robust convergence and high efficiency in non-IID edge deployments.

Communication Overhead

Gao et al. in [53] perform an end-to-end empirical evaluation comparing FL and Split Neural Networks (SplitNN) on Raspberry Pi clusters. They measure not only learning performance under imbalanced and non-IID data distributions but also real-world overheads, training time, communication traffic, power draw, and memory usage. Their results reveal that FL incurs significantly lower communication overhead in bandwidth-constrained IoT settings, while SplitNN can outperform FL in accuracy under moderately imbalanced data.

Zhang et al. propose FedSL in [87], a split-layer aggregation algorithm that groups layers of convolutional neural networks by depth and employs a Max–Min client selection strategy for each group. By transmitting only selected layer parameters, FedSL reduces communication costs by 42% on VGG-style architectures and 70% on ResNet-9, while matching the accuracy of FedAvg, FedProx, and contrastive methods like MOON.

Mu et al. introduce Communication and Storage Efficient FSL (CSE-FSL) in [88], targeting both high communication overhead and server-side model bloat. CSE-FSL maintains a single global model at the server, uses an auxiliary network for client updates, and transmits smashed data only in designated epochs. Their theoretical analysis guarantees convergence under non-convex losses, and experiments confirm large reductions in communication and storage with state-of-the-art performance.

Chawla et al. in [86] develop Efficient SL (ESL), incorporating a key–value store to cache and share intermediate activations, customized model architectures for the SL context, and personalized fine-tuning for each client. ESL demonstrates up to 1623× reduction in client computation (image classification) and 23.9× for 3D segmentation, while cutting communication traffic by 3.92× and 1.3× respectively, and boosting average accuracy by 35% and 31%.

Le et al.'s SFETEC in [89] tailors SFL to the Thing–Edge–Cloud continuum by partitioning models into base components on clients and core models on edge servers, then aggregating at the cloud. Preliminary results show SFETEC markedly reduces communication volume and training time compared to FedAvg-oriented schemes, while also mitigating non-IID label skew and enhancing privacy.

Przewikezlikowski et al. propose MonAcoSFL in [108], a momentum-aligned contrastive SFL approach. By aligning online and momentum encoder models across clients, MonAcoSFL maintains high accuracy even under deep split configurations while significantly reducing the volume of gradient and activation exchanges.

Yin et al. in [119] present a hybrid FSL (HFSL) framework augmented with a predictive Generative Adversarial Network (GAN) for

multi-objective optimization. Their algorithm jointly determines split decisions, bandwidth allocations, and compute resource distributions to approximate the Pareto front of time–energy trade-offs. Wireless network experiments show that HFSL dominates conventional FL in both metrics.

Karjee's RL-based Codec Switching (RLCS) within [116] further demonstrates that proactively adapting video codecs in response to network fluctuations can stabilize throughput and reduce retransmissions, thus cutting communication spikes in split deployments that involve multimedia data.

Duan et al. survey edge–cloud computing integrations with federated and split learning in [28], compiling techniques for communication reduction, from model compression and coded transmissions to hierarchical splits, across diverse IoT applications such as smart meters, continuous authentication, and recommendation systems.

Model Optimization

Tedeschini et al. in [11] propose Split Consensus FL (SCFL), a novel decentralized training and inference protocol inspired by Message Passing Neural Networks (MPNN). SCFL abolishes the central parameter server by enabling clients to iteratively exchange and merge partial model updates along network links. Experiments in cooperative positioning tasks show SCFL achieves convergence accuracy comparable to centralized FL without the single point of failure.

Khan et al. develop a Joint Communication and Learning Framework for Hierarchical SFL (HSFL) in [113]. Their hierarchical structure groups devices under edge servers; within each group, partial models are trained locally and aggregated at the edge, and subsequently across edges at the cloud. They decompose the overarching mixed-integer nonlinear programming formulation into convex edge-resource allocation and block successive upper-bound minimization subproblems. Evaluations demonstrate balanced trade-offs among relative local accuracy, latency, and energy consumption.

Han et al. introduce SplitGP in [91], which embeds a Gaussian Process (GP) module into SL to capture uncertainty. The client-side GP personalized layers learn to adapt to local data distributions, while the server-side GP layers generalize across all clients. Analytical bounds guarantee asymptotic convergence to stationary points, and empirical tests confirm that SplitGP reduces inference time and improves out-of-distribution accuracy compared to monolithic and federated baselines.

Liao et al. propose MergeSFL in [90], which enhances SFL through feature merging and batch size regulation. Client activations are merged into a mixed feature sequence approximating IID data, and batch sizes are tuned per device to mitigate stragglers. On an 80-node Jetson cluster, MergeSFL yields 5.82%–26.22% accuracy improvements and 1.39×–4.14× training speedups over benchmark SFL methods.

Qin et al.'s SpFedRec [92] also contributes to model-level optimization by deploying Squeeze-and-Excitation (SE) blocks within the split towers, enhancing dominant feature representation for large-scale recommendation while preserving privacy and reducing communication.

Yin et al. in [119] leverage a predictive GAN to approximate Pareto-optimal split configurations and resource allocations for hybrid FSL. The GAN's discriminator guides the generator to propose high-quality solutions along the time–energy frontier, yielding deployments that dominate standard FL and SL baselines in both latency and energy efficiency.

Shen et al.'s RingSFL [111] implicitly regularizes the global model by cyclically blending split segments across clients in a ring topology. This approach fosters model robustness and privacy without additional communication overhead, as demonstrated by accelerated convergence and resistance to inversion attacks in both IID and non-IID setups.

Lessons Learned: From the reviewed literature, several key insights and lessons can be drawn about the application of FSL in IoT and EC:

- **Privacy & Security**: Techniques such as DP (FSL-DP, LabelDP), secret-sharing chains (SpFedRec), and deeper split points (SplitFed resilience) can reduce communication and protect data, achieving up to 49% communication savings and statistically significant accuracy retention under lossy channels. Deeper split points also prove more resilient to packet loss [78].
- **Heterogeneity**: Parallel and adaptive split frameworks (HSFL, ParallelSFL, DFL, S2FL, FedSplitX, SmartSplit, CHEESE) effectively mitigate device and data heterogeneity, delivering speedups up to 3.54×, energy savings up to 62.8%, and accuracy gains of up to 57% in real-world IoT testbeds.
- **Computational & System Constraints**: Pruning (MP-FSL), hierarchical orchestration (SmartSplit), adaptive aggregation (AdaptSFL), clustering (CRSFL), and dynamic scheduling (split-Dyn) reduce memory use by 100×, cut load-distribution time by 160×, and lower latency by >90%, enabling FSL on highly constrained devices.
- **Communication Overhead**: Layer-wise aggregation (FedSL), activation caching (ESL), auxiliary update networks (CSE-FSL), and GAN-driven allocation (Predictive FSL) achieve communication reductions ranging from 42% to 1623×, all while preserving or improving model accuracy on IoT hardware.
- **Model Optimization**: Decentralized consensus (SCFL), GP-based personalization (SplitGP), feature merging with batch regulation (MergeSFL), and SE-enhanced towers (SpFedRec) enable simultaneous gains in personalization and generalization, yielding faster inference and robust out-of-distribution performance without central coordination.

Overall, these studies demonstrate that judicious model partitioning, resource-aware orchestration, and privacy-preserving mechanisms can deliver single-digit to orders-of-magnitude improvements in latency, communication overhead, memory usage, energy efficiency, and accuracy, thereby charting a clear path for deploying FSL in practical IoT and EC scenarios.

4.1.2. Wireless networks

FSL merges FL and SL to enable collaborative model training on distributed devices without sharing raw data. This is particularly appealing in wireless networks, where devices are resource-constrained and privacy-sensitive. FSL allows devices to train only a portion of a model (client side) and offload the rest to edge/cloud servers, reducing computation on devices. However, wireless channels introduce new challenges: limited bandwidth, heterogeneous device capabilities, and communication delays. Recent works in wireless communications apply FSL by jointly optimizing model partitioning and communication resources to balance latency, accuracy, and energy. The following subsections review nine representative studies, each addressing key challenges in FSL over wireless systems. Table 14 summarizes the key features, challenges addressed, techniques employed, evaluation settings, and publication details for all works on FSL in Wireless Communications.

Lin et al. [10] tackle the high latency of SL in edge networks. They propose Efficient Parallel Split Learning (EPSL), which parallelizes client-side training and aggregates only the last-layer gradients on the server to reduce both computation and communication overhead. EPSL also jointly optimizes wireless subchannel allocation, power control, and cut-layer selection to minimize per-round latency. In simulations with ResNet-18 on MNIST and HAM10000 (under IID and non-IID data), EPSL achieves accuracy comparable to vanilla SL, SFL and PSL while dramatically reducing training latency (time to reach a target accuracy). This shows that EPSL can sustain high accuracy with significantly less delay by leveraging parallelization and gradient aggregation.

Han et al. [14] present a hybrid FSL framework that simultaneously optimizes for personalization and generalization during inference. By applying adaptive loss aggregation and fine-tuning at edge nodes, the model balances global performance with user-specific adaptation, reducing the personalization gap across devices. Evaluation over wireless datasets validates improved accuracy and latency performance in dynamic edge settings.

Table 14
Summary of papers on FSL in wireless communications.

Ref #	Title	Short description	Challenges & measures	Model/ Technique	Datasets/ Benchmarks	Eval. metrics	Scalability & deployment	Type/Year
[10]	*Efficient Parallel Split Learning over Resource-constrained Wireless Edge Networks*	Enables parallel split training in wireless edge.	Latency (↓), throughput (↑) under intermittent connectivity.	Parallel SL	Wireless simulation	Latency, throughput, accuracy	Simulation	Journal, 2024
[14]	*Federated Split Learning With Joint Personalization–Generalization for Inference-Stage Optimization in Wireless Edge Networks*	Balances personalization and generalization.	Personalization gap (↓) with minimal accuracy loss.	Joint personalization–generalization	IoT/wireless datasets	Latency, accuracy, personalization gap	Simulation	Journal, 2024
[82]	*Personalized Hierarchical Split Federated Learning in Wireless Networks*	Hierarchical, personalized split FL framework.	Accuracy (↑), personalization (↑) via hierarchy.	Hierarchical SL	Wireless benchmarks	Latency, accuracy, personalization	Simulation	Journal, 2024
[120]	*Impact of Neural Network Depth on Split Federated Learning Performance in Low-Resource UAV Networks*	Studies NN depth impact in UAVs.	Latency (↓), energy (↓) with minimal accuracy change using shallower networks.	Depth analysis	UAV simulation	Accuracy, latency, energy	Simulation	Conference, 2024
[95]	*ESFL: Efficient Split Federated Learning over Resource-Constrained Heterogeneous Wireless Devices*	Optimizes FSL for heterogeneous wireless devices.	Latency (↓), energy (↓) with minimal accuracy loss.	Efficient split FL framework	Wireless device datasets	Latency, energy, accuracy	Simulation	Journal, 2024
[93]	*Accelerating Split Federated Learning over Wireless Communication Networks*	Introduces accelerated communication protocols.	Training time (↓) under unreliable channels.	Accelerated communication	Wireless simulation	Latency, throughput, accuracy	Simulation	Journal, 2023
[96]	*Split Federated Learning: Speed up Model Training in Resource-Limited Wireless Networks*	Speeds up model training in wireless.	Training time (↓) with maintained accuracy.	Speed-focused SL	Wireless simulation	Training speed, accuracy	Simulation	Conference, 2023
[121]	*Resource Optimized Hierarchical Split Federated Learning for Wireless Networks*	Hierarchical model for resource allocation.	Energy efficiency (↑), latency (↓).	Hierarchical SL	Wireless simulation	Latency, accuracy, energy efficiency	Simulation	Proceedings, 2023
[49]	*Core Network Traffic Prediction Based on Vertical FL and SL*	Hybrid vertical–split FL for traffic prediction.	Prediction accuracy (↑) under heterogeneous data.	Hybrid vertical–split FL	Network traffic datasets	Prediction accuracy, latency	Simulation	Journal, 2024

Pervej and Molisch [82] propose Personalized Hierarchical SFL (PHSFL) to handle client heterogeneity. In PHSFL, the global model is split into a shared feature extractor (body) and client-specific classifiers (heads). During global training, only the body is trained (the classifiers are left untrained), after which each client fine-tunes its own classifier on local data. The authors provide a theoretical analysis on how the split and hierarchical aggregation affect convergence. Experiments on CIFAR images with Dirichlet data splits show that the globally trained model achieves accuracy similar to standard FL/SFL, but the local fine-tuning yields significantly improved personalized accuracy for each client. This approach effectively decouples global feature learning from local specialization, improving performance in non-IID settings.

Hafi et al. [120] investigate the impact of neural network depth and split strategy on SFL performance in Unmanned Aerial Vehicle (UAV) networks. They construct an 8-layer Long Short-Term Memory (LSTM) model to predict UAV power consumption and evaluate five splitting variants (different numbers of layers on client vs. server). Their results show that SFL dramatically reduces client computation and memory usage while maintaining prediction accuracy and overall training time. They find that arbitrary or equal splits are not always optimal: splitting must consider device/server capabilities. For example, the configuration with the smallest client-side model (M1) best suits low-memory UAV devices. The study highlights that effective splitting depends on hardware and network conditions, and that FSL can enable heavy models on light devices by shifting computation server-side.

Zhu et al. [95] introduce Efficient SFL (ESFL) for heterogeneous wireless devices. ESFL jointly optimizes the model partition and resource allocation: it formulates a mixed-integer program to balance user workload and server computing power under device heterogeneity. The NP-hard problem is solved via an iterative approximation. In CIFAR-10 experiments using multiple VGG networks, ESFL achieves much faster convergence than FL, SL, or SplitFed under the same settings. In particular, ESFL attains comparable accuracy while greatly reducing training and communication time. By leveraging powerful edge servers and adjusting the split based on device resources, ESFL efficiently scales to large user counts and diverse DNN models.

Xu et al. [93] focus on latency reduction in SFL. They consider heterogeneous devices with individual split points on a neural network and jointly optimize each device's model cut-layer and its wireless bandwidth allocation. The goal is to minimize total system latency per round. The authors decompose the non-convex problem via alternating optimization, solving split-point selection and bandwidth allocation iteratively. Using MNIST and CIFAR-10 benchmarks, they demonstrate that this joint scheme significantly reduces training latency compared to fixed or naive splits. By adapting split layers to device conditions, their approach also achieves modest accuracy improvements under constrained communication, highlighting the benefit of resource-aware splitting.

Zhang et al. [96] propose Group-based SFL (GSFL) to accelerate training. In GSFL, an access point splits the model and distributes the

client-side parts to devices. Clients are organized into groups: within each group, clients run SL training in parallel (sharing a server-side model), while groups update sequentially. This "split-then-federated" approach allows parallel client training while limiting the number of server-side models needed. Simulations show that GSFL substantially reduces overall training latency compared to vanilla SL or FL, because it avoids SL's strictly sequential client updates. Notably, GSFL achieves this acceleration without degrading accuracy, demonstrating that group-based parallelism can make SL practical in wireless networks.

Khan et al. [121] introduce a Hierarchical SFL (HSFL) framework for IoT. They note that SFL may suffer from slow convergence and fairness issues when devices have varying data and resources. HSFL structures clients in a hierarchy (e.g. clusters under edge servers) and performs distributed aggregations at multiple levels to avoid single points of failure. They formulate a cost function combining local accuracy, transmit power, and resource allocation, and solve it via block successive upper bound minimization. Numerical results (on simulated IoT networks) show that HSFL can improve convergence speed and fairness compared to flat SFL: Hierarchical design and optimized resource scheduling lead to more balanced training across devices.

Li et al. [49] develop an end–edge–cloud FSL framework for smart meter load forecasting. Smart meters have only 192 kB of RAM, too little to train modern models. The proposed scheme splits the model so that smart meters train a tiny client-side network, while edge servers train the larger part. A three-phase protocol (model splitting, joint training between meters and the edge, and aggregation) is implemented on real hardware. The results are striking: relative to conventional centralized training, the FSL approach reduces meter memory footprint by 95.5%, training time by 94.8%, and communication by 50%, while achieving comparable or better forecasting accuracy. This work demonstrates that FSL can bring "core" AI intelligence to highly resource-limited IoT devices by leveraging edge/cloud resources.

Lessons Learned: Research on FSL in wireless networks underscores that its efficiency is dependent on joint learning-network optimization. A common theme across studies is the co-design of model partitioning with wireless resource allocation (e.g., bandwidth, power, device scheduling) to minimize latency and energy [10,93,95]. Techniques such as parallel client training and last-layer gradient aggregation (EPSL [10]) are pivotal in overcoming the sequential bottleneck of vanilla SL, enabling high accuracy with significantly reduced delay. Furthermore, personalization strategies (for example, PHSFL [82]) effectively decouple global feature learning from local specialization, which proves crucial for handling non-IID data distributions across devices. Although FSL consistently demonstrates the ability to enable complex models on resource-constrained hardware by offloading computation to servers, the communication of smashed data can still be substantial, especially under channel variability. This highlights the need for continued innovation in compression and adaptive strategies. Privacy, often addressed via DP, frequently involves a trade-off with model accuracy. Future work must focus on dynamic splitting, integration with advanced architectures like transformers, and robust validation in large-scale, real-world wireless testbeds to fully realize FSL's potential for wireless edge intelligence.

4.2. Regulated and sensitive data domains

Domains such as healthcare, finance, and government services are subject to strict data protection regulations (e.g., GDPR, HIPAA) that prohibit the sharing of raw data. FSL's inherent privacy preservation, transmitting only intermediate activations, makes it an attractive paradigm for collaborative learning across institutions. This subsection focuses on applications in healthcare, but the principles extend to other regulated sectors. Key challenges include label privacy, handling noisy annotations, and ensuring robustness to distribution shifts.

4.2.1. Healthcare

FSL merges the privacy benefits of FL with the resource efficiency of SL to address core challenges in medical AI. Healthcare data is highly sensitive and often siloed; FSL enables collaborative model training *without* exchanging raw patient records. Unlike traditional FL that requires each client to train a full model (impractical for low-powered devices), FSL splits the neural network between client and server. This approach greatly reduces on-device computation and storage burdens [122]. At the same time, FSL retains the privacy guarantee of FL by transmitting only intermediate features or model updates (not raw data) to a central aggregator [123]. These properties are particularly crucial in healthcare, where data privacy laws and heterogeneous edge devices (e.g., Internet of Medical Things (IoMT) wearables, hospital servers) impose severe constraints. In this subsection, we review recent FSL works in healthcare, each tackling problems such as data scarcity, device heterogeneity, and data privacy with novel methods. Table 15 summarizes the key characteristics, challenges addressed, techniques employed, evaluation settings, and publication details for all works on FSL in healthcare.

In [124] Karnati et al. apply FSL to a classic tabular dataset of breast cancer diagnostics (Wisconsin Diagnostic Breast Cancer, with 30 features) to demonstrate that decentralized training can match or exceed centralized accuracy while preserving privacy. Using a simple neural architecture (few fully-connected layers with ReLU and dropout), the authors implement both FL (via FedAvg) and SL on distributed clients, each holding a portion of the data. In their SL setup, each client runs a *client-model* (a linear layer + ReLU + dropout) that outputs smashed data to a central *server-model*, which completes the classification. The federated setup alternates full-model training on each client with global averaging. On the WDBC dataset, both FSL methods achieve *very high* diagnostic accuracy (98% for FL, 99% for SL), illustrating their superiority over a naive centralized model. Training metrics like loss and accuracy curves show rapid convergence and high performance across clients. Crucially, these results are obtained *without ever sharing raw patient features*, thus fully respecting data privacy. While the work uses a relatively small dataset, its key finding is that FSL can deliver near-perfect predictive performance on medical data *and* maintain decentralization, suggesting practical viability for clinical decision support. The study also notes that SL was slightly more accurate than FL here, hinting that in some settings splitting the model can even boost accuracy (perhaps by reducing overfitting in the central layers).

In [122] Ni et al. propose FedSL, a federated split framework tailored to resource-constrained wearable medical devices. The challenge is that small IoMT sensors (like patch monitors or smartwatches) generate useful health data but cannot host large neural nets. FedSL splits the model so that each IoMT device holds a lightweight "head" of the model (several initial layers) and an edge server holds the remaining "tail" layers. Training proceeds in two stages: in multiuser SL, each device feeds local mini-batches through its head and sends the smashed activations to the server, which completes the forward and backward pass; then in FedAvg, all devices upload their updated head weights to a central node for aggregation. This hybrid pipeline allows even low-end devices to participate in training. The authors evaluate FedSL on medical imaging tasks (e.g. classifying chest X-rays and OCT scans) under both IID and non-IID label distributions. Metrics include accuracy and convergence over training rounds. FedSL's performance closely matches or slightly outperforms both CML and standard FL benchmarks, especially in non-IID settings: for example, after $\tilde{5}0$ rounds, the gap in accuracy between FedSL and fully central/FL is negligible. This indicates robustness to data heterogeneity. Importantly, FedSL preserves privacy (only intermediate features are communicated), and by offloading most computation to the server, it greatly reduces the hardware burden on devices. The framework scales to many devices because each client only trains a shallow network; experiments show stable accuracy gains even as the number of clients increases. The

Table 15
Summary of papers on FSL in healthcare.

Ref #	Title	Short description	Challenges & measures	Model/ technique	Datasets /Benchmarks	Eval. metrics	Scalability & deployment	Type/year
[17]	*Federated Split Learning with HyperNetworks for Medical Image Classification*	Uses hypernetworks for medical images.	Classification accuracy (↑), privacy preserved.	Hypernetwork-integrated split FL	Chest X-ray, MRI	Classification accuracy, privacy leakage	Simulation	Conference, 2024
[122]	*FedSL: Collaborative Healthcare Analytics on Wearables*	Enables split FL on low-end IoMT devices.	Resource usage (↓), convergence speed (↑).	ResNet-18 split	Chest X-ray (pneumonia), OCT	Accuracy, convergence speed	Simulation	Journal, 2024
[124]	*Federated and Split Learning for Enhanced Breast Cancer Diagnosis*	Hybrid FL–SL for breast cancer.	Diagnostic accuracy (↑), AUC (↑), sensitivity (↑).	Hybrid FL–SL framework	Breast cancer imaging	Diagnostic accuracy, AUC, sensitivity	Simulation, prototype	Conference, 2023
[17]	*Federated Split Learning with Hypernetworks for Medical Image Classification*	Integrates dynamic hypernetworks to adapt client models and cut-layer in real-time.	Addresses data heterogeneity, enhances model adaptation, improves accuracy	Hypernetwork integrated split FL	Pneumoni-aMNIST (Chest X-rays)	Classification Accuracy, Training time	Simulation	Conference, 2024
[123]	*FeSViBS: Federated Split Learning of Vision Transformer with Block Sampling*	Integrates block sampling in ViT.	Model complexity (↓), accuracy maintained.	Block sampling with ViT	Image benchmarks	Accuracy, sampling efficiency	Simulation	Journal, 2023
[125]	*Smart Split-Federated Learning over Noisy Channels for Embryo Image Segmentation*	Addresses noisy channels in embryo segmentation.	Segmentation accuracy (maintained) despite noise.	Noisy channel robust SL	Embryo image datasets	Segmentation accuracy, noise robustness	Simulation, testbed	Open Access, 2023
[126]	*Quality-Adaptive Split-Federated Learning for Segmenting Medical Images with Inaccurate Annotations*	Handles noisy annotations in segmentation.	Segmentation accuracy (↑) under annotation noise.	Quality-adaptive SL	Medical segmentation benchmarks	Segmentation accuracy, quality metrics	Simulation	Conference, 2023
[127]	*Robust Split Federated Learning for U-shaped Medical Image Networks*	Robust FSL for U-shaped models.	Segmentation accuracy (↑), robustness (↑).	Robust SL	Medical imaging benchmarks	Segmentation accuracy, robustness	Simulation	Journal, 2022

authors also highlight FedSL's deployment feasibility: it requires only one coordinating server and can easily be added atop existing FL infrastructure. Overall, this work demonstrates FSL's promise for wearable health analytics, blending privacy with efficiency for large-scale IoMT deployments.

In [17] Sindhura et al. tackle the common FSL pitfalls of data heterogeneity and model adaptation by integrating dynamic hypernetworks into the split learning pipeline. The key idea is that the server-side hypernetwork dynamically generates or adjusts the client's model parameters (or the cut-layer location) in real-time based on the incoming data features. This adaptive mechanism aims to optimize the "cut" between client and server during training and to handle varying client computational budgets. The authors apply their method to a medical image classification task using the PneumoniaMNIST dataset (5856 pediatric chest X-rays, binary pneumonia detection). All experiments use the same baseline CNN architecture across methods for fair comparison. Evaluation metrics include classification accuracy and training time. The results are striking: standard SL (with a fixed cut) only achieved $\tilde{5}3\%$ accuracy, FL achieved $\tilde{7}3\%$, whereas the proposed SL+Hypernetwork approach reached 86% accuracy (higher even than centralized training at 80%). They attribute this gain to the hypernetwork's ability to tune model features to each client's data. Additionally, the hypernetwork optimization significantly reduces total training time (e.g. from 6.39 min to 4.24 min), which is critical for practical deployments. In summary, this paper's contribution is a method for FSL that *adapts* to client heterogeneity on-the-fly. It shows that advanced techniques (hypernetworks) can substantially improve both accuracy and efficiency. One limitation is that the method was demonstrated on a single dataset, but the concept of dynamic cut-layer selection could generalize to other medical domains (e.g. radiology, histopathology) and edge scenarios. This work highlights that carefully designing the split and augmentation can lead to better generalization on non-IID healthcare data.

In [123] Almalik et al. propose FeSViBS, a novel FSL framework for medical image tasks using Vision Transformers (ViT). The core innovation is a block sampling module at the server: instead of always forwarding only the class token from the ViT's final block (standard SL), FeSViBS randomly selects one intermediate transformer block's patch embeddings, aggregates their information into a pseudo-class token, and sends this enriched token back to the client. This serves as a powerful form of feature augmentation. The split model has a small ViT "head" on the client that extracts patches and a larger ViT "body" on the server that processes them. FeSViBS was evaluated on three different medical imaging benchmarks, HAM10000 (dermoscopic skin lesion images), BloodMNIST (microscopic blood cell images) and a federated version of ISIC2019 (skin lesion segmentation), both in IID and non-IID settings. Compared to plain SL, FL, and CML, FeSViBS consistently achieved higher classification accuracy on all datasets by virtue of its pseudo-token augmentation. Qualitatively, intermediate features carry richer information to the server, improving generalization. The authors also analyze communication cost and convergence: By only sampling one block, FeSViBS reduces overhead compared to sending multiple partial activations, making the federated split of a heavy ViT model more feasible. In effect, FeSViBS scales ViT-based models to federated healthcare settings. This work demonstrates that FSL can leverage state-of-the-art architectures (Vision Transformers) and still ensure data privacy. The scalability is highlighted by the fact that FeSViBS works well in non-IID regimes – essential for real-world hospitals – and by optimizations that keep the client-side model lightweight.

Both the hypernetwork approach and FeSViBS share a focus on enhancing intermediate representations during FSL. Sindhura et al. and Almalik et al. show that adding intelligence at the split interface (via dynamic hypernetworks or block sampling) yields notable performance gains. In practice, both methods augment what is sent from client to server: Sindhura's hypernetwork refines model weights for each client,

while FeSViBS distills multiple patch embeddings into richer tokens. The key lesson is that more sophisticated feature fusion can mitigate the loss of accuracy inherent in strict data partitioning. Both works report that their techniques yield higher accuracy (86% vs. 53% for SL in Sindhura et al. and improvements across three image datasets in Almalik et al.) and do so without compromising privacy. They also demonstrate efficient deployment: Sindhura et al. cut training time significantly, and Almalik et al. communicate only one sampled token per round. These advances suggest a trend: integrating AI methods (hypernetworks, transformer modules) into FSL pipelines can tailor the shared model to heterogeneous medical data and devices, making FSL scalable to large models and datasets.

In [125], Kafshgari et al. examine the impact of communication noise on a SplitFed learning system applied to embryo image segmentation. They observe that SplitFed requires frequent transmission of activations and gradients between clients and the server, which can be corrupted by channel noise. To address this, the authors propose a novel "smart averaging" strategy that dynamically adjusts the aggregation of model updates to mitigate noise effects. In experiments with a SplitFed U-Net for embryo segmentation, this noise-aware aggregation significantly outperforms conventional averaging, maintaining model accuracy even under extremely noisy channels. This work is one of the first to explicitly incorporate channel impairment into the SplitFed framework; prior FSL studies generally assumed ideal communication, thus it extends SplitFed learning to more realistic, unreliable network conditions.

Authors in [126] tackle a complementary problem: inconsistent label quality across clients. Medical image segmentation clients often annotate data with varying expertise, introducing bias and errors in ground truth. The authors introduce QA-SplitFed, a data quality-adaptive aggregation scheme that weights each client's model update based on validation scores that reflect the reliability of the annotation. They evaluated QA-SplitFed on the embryo segmentation task by artificially corrupting the labels of some clients. The results show that QA-SplitFed dramatically outperforms five standard aggregation baselines: while all baselines collapse in accuracy as more clients have poor labels, the QA-SplitFed approach maintains high segmentation accuracy as long as at least one client remains uncorrupted. This finding highlights a novel contribution to the SplitFed literature, since previous work had not directly addressed annotation noise. By integrating label-quality information into the FedAvg step, this method enhances robustness against noisy annotations, complementing other FSL advances that handle data heterogeneity or privacy.

Lessons Learned: The application of FSL in healthcare reinforces several critical design principles. First, architectural innovation at the split point is a significant performance driver, as demonstrated by dynamic hypernetworks [17] and Vision Transformer block sampling (FeSViBS [123]), which recoup accuracy losses from data partitioning and enhance feature representation. Second, efficiency for edge devices is non-negotiable; frameworks like FedSL [122] are explicitly designed to operate on low-end IoMT devices by offloading most computations to the server. Third, robustness to real-world imperfections is essential, with studies proposing noise-aware aggregation for volatile channels [125] and quality-adaptive mechanisms for inconsistent annotations [126]. Fourth, the feasibility of deployment is carefully considered, with designs that work within single-server constraints and limited bandwidth. Ultimately, these works collectively demonstrate that FSL successfully resolves the tension between data security and model performance in healthcare, provided the learning protocol is meticulously tailored for privacy, efficiency, and heterogeneity.

4.3. High-mobility networks

High-mobility networks, such as vehicular edge computing (VEC) and unmanned aerial vehicle (UAV) swarms, pose unique challenges for distributed learning: rapid topology changes, intermittent connectivity, strict latency requirements, and heterogeneous node capabilities. FSL addresses these challenges by enabling partial model training on mobile clients and leveraging edge infrastructure (roadside units, base stations) for aggregation and server-side computation.

4.3.1. Vehicular networks

Vehicular networks are increasingly employing DML due to the proliferation of onboard data and the need for privacy-preserving on-device intelligence. Traditional FL allows vehicles to collaboratively train models without sharing raw data, but it struggles with vehicular heterogeneity, non-IID data and scenarios where data and labels are held by different entities [128]. SL can offload part of the model to edge servers, mitigating model weight leakage and local computation burden. FSL merges these paradigms by splitting models between vehicles and edge infrastructure and aggregating gradients, enabling parallel training across many vehicles. This approach inherently reduces communication overhead and latency in vehicular edge scenarios [18]. As the authors of this paper note, an adaptive parallel FSL scheme can exploit vehicle mobility and network conditions to optimize the cut layer and resource allocation. Thus, FSL is well-suited for Vehicular Edge Intelligence (VEI), addressing mobility and connectivity dynamics while preserving data privacy [18,128]. Table 16 summarizes the key features, challenges addressed, techniques employed, evaluation settings, and publication details for all works on FSL in Vehicular Networks.

In [18] by Qiang et al. the authors propose ASFV, an adaptive FSL scheme for VEC. They observe that traditional sequential SL causes high communication delay in vehicular contexts, so ASFV parallelizes training among vehicles while adaptively splitting the model at an optimal cut layer. The scheme incorporates a vehicle selection algorithm based on vehicle speed and communication range, and formulates a joint delay-and-energy minimization problem under vehicle heterogeneity and channel instability. This NP-hard problem is decomposed into subproblems (cut-layer selection, transmit power allocation, and bandwidth assignment) solved via block-coordinate descent and KKT/SCA methods. Extensive simulations on non-IID vehicle datasets show that ASFV significantly reduces training latency and energy consumption compared to benchmarks by adapting the data rate and power allocation to the fluctuating V2I channel quality.

The paper "Mobility-Aware Split-Federated With transfer learning for Vehicular Semantic Communication Networks" by Zheng et al. introduces MSFTL, a framework tailored for high-mobility semantic communications [57]. Here the semantic encoder is split into four parts across the vehicle and edge, so each vehicle only needs to train a portion of the encoder, reducing its computation. They integrate TL by reusing parts of a pre-trained semantic encoder to accelerate convergence and improve accuracy under limited training samples. To handle rapid mobility, the authors formulate a Stackelberg game for resource and data allocation: the mechanism fairly allocates training data based on vehicle residence time, computational load, and QoS requirements. Simulations show MSFTL achieves faster convergence and higher accuracy in vehicular semantic tasks than conventional FL or SL methods.

In "Federated Split Learning With Data and Label Privacy Preservation in Vehicular Networks", Wu et al. consider a vehicular setting where vehicles hold raw sensor data but no labels, while Road-Side Units (RSUs) hold labels [128]. They propose an FSL paradigm where the deep model is split into a vehicle-side submodel and an RSU-side submodel. Vehicles and RSUs exchange only intermediate (smashed) data and cut-layer gradients; gradients are aggregated at a server. The authors analyze three privacy attacks showing that attackers could recover private data or labels from the exchanged intermediate information. To defend against these, they introduce a DP mechanism that injects noise into the intermediate data. Furthermore, they design a contract-theoretic incentive scheme that rewards vehicles and RSUs for improving model performance while respecting privacy budgets.

Table 16
Summary of papers on FSL in vehicular networks.

Ref #	Title	Short description	Challenges & Measures	Model/ Technique	Datasets/ Benchmarks	Eval. metrics	Scalability & deployment	Type/Year
[18]	*Adaptive and Parallel Split Federated Learning in Vehicular EC*	Adaptive parallel FSL for vehicles.	Latency (↓), energy (↓), accuracy (↑).	Adaptive parallel split FL	Vehicular network datasets	Latency, energy, accuracy	Simulation	Preprint, 2024
[57]	*Mobility-Aware Split-Federated With Transfer Learning for Vehicular Semantic Communication Networks*	Integrates mobility awareness and TL.	Semantic accuracy (↑) under dynamic mobility.	Mobility-aware split FL + TL	Vehicular network datasets	Latency, semantic accuracy	Simulation	Journal, 2024
[128]	*Federated Split Learning With Data and Label Privacy Preservation in Vehicular Networks*	Data/label privacy via DP + contract theory.	Privacy risk (↓), utility (↑).	DP + contract theory	Vehicular datasets	Privacy risk, accuracy, utility	Simulation	Journal, 2024
[129]	*Split Federated Learning Empowered Vehicular Edge Intelligence: Concept, Adaptive Design, and Future Directions*	ASFL with dynamic cut-layer selection.	Accuracy (↑), comm. overhead (↓), convergence (↑).	Dynamic cut-layer + federated aggregation	Vehicular datasets	Accuracy, communication overhead, convergence	Simulation	Journal, 2025
[94]	*Split and Federated Learning with Mobility in Vehicular EC*	Handles vehicular mobility in FSL.	Accuracy (maintained), connectivity (maintained).	Integrated split-FL framework	Vehicular datasets	Latency, accuracy, connectivity	Simulation	Journal, 2023
[9]	*Enabling Intelligent Vehicular Networks Through Distributed Learning in the Non-Terrestrial Networks 6G Vision*	FSTL for 6G NTN vehicular networks.	Latency (↓), accuracy (↑).	Multilayer FSTL	ResNet on MNIST	Accuracy, latency	Simulation	Conference, 2024
[75]	*Multi-Layer Distributed Learning for ITS in 6G Aerial–Ground Integrated Networks*	GFSTL for aerial–ground ITS.	Scalability (↑), accuracy (↑), latency (↓).	GFSTL on multilayer networks	ResNet on vehicular datasets	Scalability, accuracy, latency	Simulation	Conference, 2024

The paper "Split Federated Learning Empowered Vehicular Edge Intelligence: Adaptive Parallel Design and Future Directions" by Qiang et al. takes a broader view, discussing how SFL can enable VEI [129]. They outline an adaptive, parallel SFL scheme that leverages vehicular computation and edge resources to train AI models. The authors argue that by distributing model training across vehicles and edge servers, SFL can reduce data leakage and latency in ITS applications. They provide a performance analysis of the proposed SFL design and highlight open research directions, such as scalable deployment under dynamic vehicular environments and efficient resource management for SL.

In [94] Moon and Lim study FSL in a highly mobile VEC scenario. They propose a two-cut scheme where the neural network has an *early* and a *late* split point: vehicles always perform computation up to an early cut (minimizing initial data transfer), while the remaining layers are processed at the server using a later cut. This dual-cut approach balances the load between vehicle and edge and avoids extreme splits that would overload one side. Crucially, when a vehicle hands over from one edge server to another, the "SplitFed learning with mobility" method migrates the current smashed data and model state to the target server, allowing training to continue seamlessly. The authors evaluate this on an object recognition task and show that SplitFed-M maintains classification accuracy while substantially reducing transmission volume and latency under realistic mobility. This work highlights that carefully choosing split points and enabling model migration are effective techniques to sustain FSL performance as vehicles move across edge domains.

Across these works, a few lessons emerge. First, model split strategies must adapt to vehicular dynamics: optimal cut layers or TL schemes can mitigate latency and account for vehicles' compute limits and channel quality [18,57]. Second, privacy is not automatically solved by splitting: intermediate representations may still leak sensitive data, so techniques like DP or encryption are often needed [128]. Third, resource allocation is critical; game-theoretic or optimization frameworks (vehicle selection, bandwidth and power allocation) are commonly used to balance latency, energy and fairness in moving networks. Finally, incentives and TL can further improve participation and convergence in non-IID settings. In summary, FSL's combination of parallel learning and privacy preservation shows great promise for vehicular intelligence, but its effectiveness hinges on carefully addressing heterogeneity, mobility and security.

Naseh et al. [9] first introduced FSTL for highly mobile Vehicular Users (VUs) and demonstrated its benefits in terms of privacy preservation and reduced backhaul usage. Building on that foundation, in [75] the multilayer GFSTL architecture couples RSUs on the ground with High-Altitude Platforms (HAPs) to form a unified Aerial–Ground Integrated Network (AGIN). This arrangement improves coverage, mitigates intermittent connectivity, and shortens model-update latency for safety-critical vehicular applications.

Fig. 11 depicts the workflow. Locally, each VU executes a shallow "client" model and transmits smashed data to its serving RSU, which aggregates updates for its cluster. Completed FSTL models from all RSUs are then forwarded to the HAP, which plays the role of the main FL server, performing final aggregation to obtain the global model, thus realizing the four-step loop: *local compute → RSU aggregation → HAP aggregation → model redistribution*.

In summary, vehicular GFSTL yields scalable learning while ensuring that raw sensor data never leaves the vehicle. Key challenges remain: (i) fast hand-overs between RSUs under high mobility, (ii) efficient scheduling of limited HAP backhaul, and (iii) robustness to non-IID data caused by diverse driving conditions [130].

Lessons Learned: The application of FSL in vehicular networks yields several critical insights. First, adaptive model partitioning is paramount. Work such as ASFV [18] and MSFTL [57] demonstrates that dynamically selecting the optimal cut layer based on vehicle

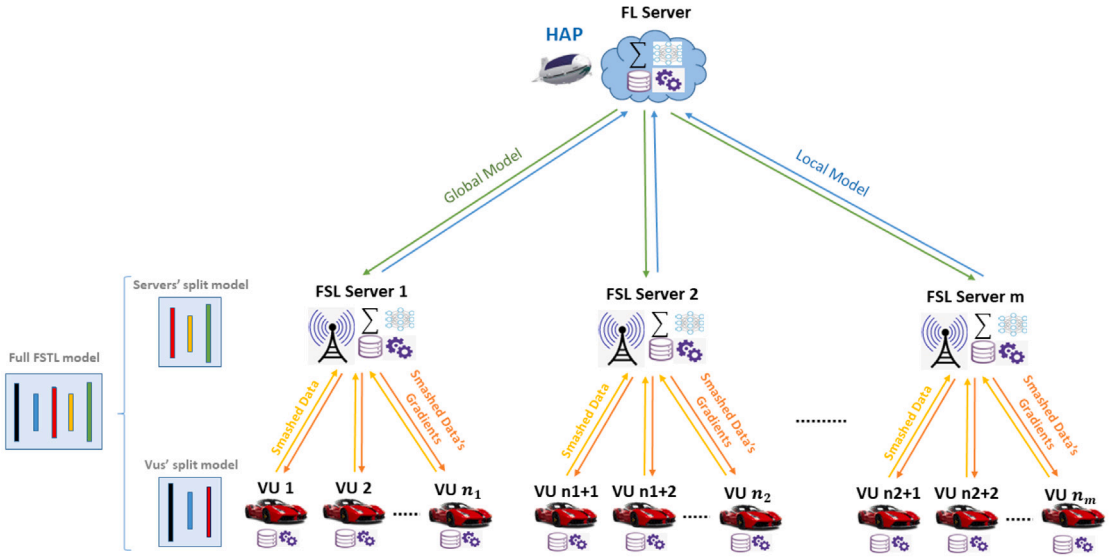


Fig. 11. GFSTL structure for vehicular AGIN.

computational capability, channel quality, and mobility patterns is crucial for minimizing latency and energy consumption. This adaptability allows the system to effectively balance the computational load between the vehicles and the edge infrastructure. Second, privacy is not automatically guaranteed by model splitting. As highlighted in [128], intermediate activations (smashed data) and gradients transmitted between vehicles and RSUs can be vulnerable to reconstruction attacks, potentially leaking sensitive sensor data or labels. Consequently, integrating privacy-preserving techniques such as DP is often necessary to fortify the framework. Third, managing high mobility and intermittent connectivity is a unique challenge in VEC. Solutions such as the dual-cut scheme with state migration in [94] and mobility-aware client selection in [18] are crucial to maintaining training continuity and consistency as vehicles hand over between RSUs. Fourth, incentive mechanisms and TL play a vital role. Using Stackelberg games for resource allocation [57] or contract theory for participant incentivization [128] ensures fair and efficient collaboration. Furthermore, leveraging TL, as in MSFTL [57], accelerates convergence and improves model performance, which is particularly beneficial given the non-IID nature of vehicular data across different routes and environments. Finally, hierarchical architectures like GFSTL [75] show immense promise for scalable VEC, enabling model aggregation across aerial and ground networks to enhance coverage and reduce latency. In summary, while FSL effectively brings distributed intelligence to vehicular networks, its success depends on a tightly integrated approach that combines adaptive splitting, robust privacy measures, mobility-aware protocols, and strategic incentive design.

4.4. Large language models and transformer architectures

The emergence of large language models (LLMs) and Transformer architectures [131] has revolutionized natural language processing and extended to vision (Vision Transformers, ViT [132]) and multimodal domains. However, training such massive models is computationally intensive and raises significant privacy concerns when data is distributed across multiple parties. FSL offers a promising solution by partitioning these large architectures across clients and servers, enabling collaborative training without exposing raw data. In this subsection, we review recent FSL works targeting large language models and Transformer architectures. Table 17 provides a concise summary of these papers.

4.4.1. FSL for large language models

Training LLMs in federated settings faces unique challenges: the sheer number of parameters (often billions) makes transmitting full gradients or model updates infeasible, and client devices (e.g., mobile phones) lack the memory to store the entire model. FSL addresses these issues by splitting the Transformer into a client-side portion (e.g., embedding layers and a few initial Transformer blocks) and a server-side portion (the remaining blocks and task head). Clients transmit only intermediate activations (smashed data) to the server, drastically reducing communication and memory footprint.

Zhao et al. [76] introduce FedSLLM, a framework that adapts split learning to LLMs by incorporating Low-Rank Adaptation (LoRA) [73] to further reduce parameter exposure. Their approach divides the LLM into client and server subnetworks, and by applying LoRA, they optimize the computation–communication trade-off, reducing training delays by an average of 47.63% while maintaining competitive perplexity on the WikiText benchmark.

Shi et al. [133] explore heterogeneous federated learning with split language models, focusing on Vision Transformers (ViTs). They leverage pre-trained ViTs and employ zeroth-order optimization for gradient approximation, enabling efficient training on resource-constrained edge devices. Their method shows resilience to gradient inversion attacks and achieves strong accuracy on CIFAR-10/100 and Tiny-ImageNet under non-IID data distributions.

4.4.2. Challenges specific to transformer architectures

Applying FSL to Transformers introduces additional complexities:

- **Activation size explosion:** Intermediate activations in Transformers, especially the multi-head attention outputs, can be large (e.g., sequence length × hidden dimension). Transmitting these activations may negate the communication benefits of FSL. Techniques such as activation compression, quantization, and selective transmission are critical [10].
- **Attention mechanism partitioning:** Splitting attention layers across client and server is non-trivial because self-attention requires global context. One approach is to place early attention blocks on the client and later blocks on the server, but this may limit the client's ability to capture long-range dependencies. Alternative strategies include using cross-attention or approximated attention [134].

Table 17
Summary of papers on FSL in large language models and transformer architectures.

Ref #	Title	Short Description	Challenges & Measures	Model/ Technique	Datasets/ Benchmarks	Eval. metrics	Scalability & deployment	Type/Year
[76]	*FedsLLM: Federated Split Learning for Large Language Models over Communication Networks*	Adapts split FL with LoRA for efficient LLM training.	Communication cost (↓), training delay (↓47.63%), perplexity (competitive).	Transformer-based split FL + LoRA	WikiText-2	Perplexity, communication cost	Simulation, wireless network emulation	Journal, 2024
[133]	*Heterogeneous Federated Learning with Split Language Model*	Gradient approximation for distributed LLM training; focuses on Vision Transformers.	Convergence (↑), resilience to gradient inversion attacks, accuracy under non-IID data.	ViT, zeroth-order optimization, split learning	CIFAR-10, CIFAR-100, Tiny-ImageNet	Accuracy, convergence rate	Simulation, edge device emulation	Preprint, 2024
[123]	*FeSViBS: Federated Split Learning of Vision Transformer with Block Sampling*	Integrates block sampling in ViT for medical imaging; reduces complexity while maintaining accuracy.	Model complexity (↓), communication overhead (↓), accuracy (↑).	Vision Transformer with block sampling, split FL	HAM10000, BloodMNIST, ISIC2019	Classification accuracy, communication rounds	Simulation, federated healthcare settings	Journal, 2023

- **Memory constraints on clients:** Even with splitting, the client-side portion of a Transformer (e.g., several layers) may exceed the memory of edge devices. Model pruning, quantization, and knowledge distillation are being explored to create lightweight client sub-networks [135].

4.4.3. Vision Transformers and FSL

The success of Vision Transformers (ViT) in computer vision has prompted their integration into FSL for medical imaging and remote sensing. Almalik et al. [123] propose FeSViBS, a federated split learning framework for ViT that introduces a block sampling module at the server. Instead of forwarding only the class token, FeSViBS randomly selects an intermediate transformer block's patch embeddings, aggregates them into a pseudo-class token, and sends this enriched token back to the client. This augmentation improves feature representation and boosts accuracy on medical imaging datasets (HAM10000, BloodMNIST) while keeping the client-side model lightweight. The block sampling mechanism also reduces communication overhead compared to sending multiple partial activations.

4.4.4. Cross-modal and multimodal extensions

FSL's ability to handle heterogeneous data makes it attractive for cross-modal learning, where models must align representations from different modalities (e.g., text–image, text–speech). For instance, in medical diagnostics, combining imaging data (X-rays) with electronic health records (text) could improve accuracy, but data silos and privacy regulations hinder centralized training. FSL can enable collaborative training of multimodal Transformers by splitting modality-specific encoders across clients and a fusion module on the server. Early works in this direction include split learning for vision–language models [136] and federated multimodal learning [137].

4.4.5. Future directions

The integration of FSL with foundation models opens several research avenues:

- **Efficient fine-tuning:** Parameter-efficient fine-tuning methods (LoRA, adapters, prompt tuning) can be seamlessly combined with FSL to adapt large pre-trained models to downstream tasks with minimal client computation and communication.
- **Dynamic splitting for Transformers:** Developing algorithms that dynamically adjust the split point based on input sequence length, client capability, and network conditions could optimize the trade-off between accuracy and efficiency.
- **Privacy-preserving attention:** Designing attention mechanisms that operate on encrypted or noisy activations to protect sensitive information while maintaining model performance.
- **Multimodal foundation models:** Extending FSL to large multimodal models (e.g., CLIP, Flamingo) to enable privacy-preserving collaborative learning across diverse data types.

4.4.6. Lessons learned

From the reviewed literature, several key insights emerge:

- Pre-trained models significantly reduce the computational burden on clients, making LLM training feasible in federated settings.
- Communication efficiency is paramount; techniques like LoRA, gradient approximation, and activation compression are essential.
- Transformer-specific challenges (activation size, attention partitioning) require novel solutions beyond those developed for CNNs.
- Vision Transformers benefit from architectural innovations at the split interface (e.g., block sampling) that enrich intermediate representations.

Future work should focus on refining these techniques, developing standardized benchmarks for FSL with Transformers, and exploring cross-modal applications in privacy-sensitive domains like healthcare and finance.

4.5. Remote sensing and non-terrestrial networks

Earth observation (EO) and non-terrestrial networks (NTNs), including satellites, high-altitude platforms (HAPs), and UAVs, generate vast amounts of geospatial data that must be processed efficiently despite limited bandwidth and intermittent connectivity. FSL's hierarchical architecture (GFSTL) naturally aligns with the multi-layer structure of NTNs, enabling in-situ processing at lower layers (UAVs, HAPs) and global aggregation at higher layers (satellites).

4.5.1. Earth observation

EO involves the collection and analysis of data about the Earth's physical, chemical, and biological systems using remote sensing technologies, typically through satellites and UAVs. These data are crucial for a wide range of applications, including environmental monitoring, natural disaster detection, agricultural assessment, climate change analysis, and urban planning. However, the sheer volume and distributed nature of EO data present significant challenges in terms of transmission bandwidth, processing efficiency, and data privacy. To address these issues, the multilayer GFSTL framework is adopted, enabling hierarchical DML across different layers of the NTN infrastructure [20]. By processing data closer to the source and coordinating learning tasks across Low-Earth Orbit (LEO) satellites, UAVs, and ground stations, GFSTL facilitates timely, resource-efficient, and privacy-aware EO analytics. Table 18 summarizes the key features, challenges addressed, techniques employed, evaluation settings, and publication details for all works on FSL in EO.

To address the bottlenecks of satellite-only EO systems, the authors in [23] extend GFSTL to a three-tier NTN that synergistically employs

Table 18
Summary of papers on FSL in earth observation.

Ref #	Title	Short description	Challenges & Measures	Model/ Technique	Datasets/ Benchmarks	Eval. metrics	Scalability & deployment	Type/Year
[20]	*Distributed Learning Framework for Earth Observation on Multilayer Non-Terrestrial Networks*	GFSTL for multilayer NTN in EO.	Privacy – accuracy trade-off analysis.	Multilayer NTN GFSTL	N/A	N/A	Architectural proposal	Proceedings, 2024
[23]	*Distributed Intelligent Framework for Remote Area Observation on Multilayer Non-Terrestrial Networks*	Distributed learning for remote EO.	Latency (↓), accuracy (↑).	Multilayer GFSTL	ResNet on MNIST	Accuracy, latency	Simulation	Conference, 2024

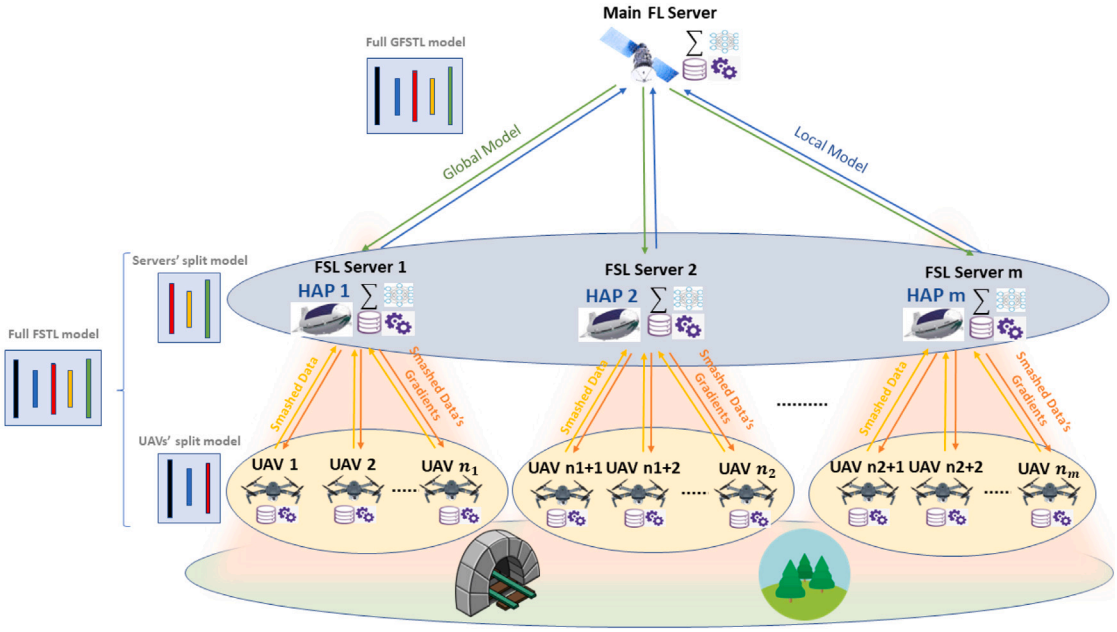


Fig. 12. GFSTL structure for Earth observation using non-terrestrial layers.

UAVs, HAPs, and LEO satellites. The resulting architecture enables fine-grained, in-situ data capture while preserving the global vantage and compute muscle of higher layers, ultimately improving image resolution and decision timeliness for climate monitoring and disaster response.

As illustrated in Fig. 12, UAVs keep raw imagery onboard and send only smashed features to the overlying HAP, which acts as an FSL server. Multiple HAP–UAV groups periodically upload their composite models to a LEO satellite that performs FedAvg, after which the refined global model is pushed back down. The numbered arrows in the figure show the iterative steps in order.

Lessons Learned: The multilayer EO pipeline boosts accuracy, shortens convergence via TL, and respects data-sovereignty rules. Outstanding issues include selecting pre-trained backbones that minimize domain shift, coping with intermittent air–space links, and scaling the protocol as the fleet of UAV–HAP clusters grows.

4.6. Lessons learned

The research on FSL has provided numerous insights into the feasibility and practical implementation of this learning paradigm. Several lessons have been drawn from experimental evaluations, real-world applications, and theoretical analysis:

Privacy-Preserving Techniques Need Further Refinement: While FSL mitigates privacy concerns by splitting model training, the transmission of intermediate activations still poses risks. Studies suggest that homomorphic encryption and DP should be further integrated to enhance security.

Adaptability to Heterogeneous Environments is Key: The success of FSL depends on its ability to accommodate devices with varying computational power and connectivity constraints. Solutions such as adaptive cut-layer selection and hierarchical federated aggregation have shown promise in optimizing training efficiency under dynamic conditions.

Communication Overhead Remains a Challenge: Although activation compression and bandwidth-aware optimizations have been explored, reducing communication overhead without sacrificing model accuracy remains an ongoing area of research.

Parallel and Asynchronous FSL Frameworks Improve Scalability: Traditional sequential SL models can cause delays due to dependencies between client and server updates. Emerging approaches incorporating parallelization and asynchronous updates have demonstrated superior scalability and training speed.

By incorporating these insights into future FSL frameworks, researchers can enhance the robustness, efficiency, and scalability of FSL models for real-world applications.

5. Evaluation methodologies, implementation frameworks, and research trends

The advancement of FSL as a research field depends on the adoption of rigorous, standardized, and reproducible evaluation practices.

A comprehensive assessment of any FSL contribution requires not only measuring the final model's accuracy but also quantifying the associated communication and computational costs that define its real-world viability. This section provides an exploration of the essential methodologies for evaluating FSL systems, from performance metrics and standard benchmarks to the software frameworks and simulation tools that form the backbone of modern research in DML.

5.1. Performance metrics and benchmarks for comprehensive evaluation

Evaluating an FSL system is a multi-objective optimization problem where model efficacy must be constantly balanced against system efficiency. A robust evaluation framework, therefore, must incorporate a diverse set of metrics that together paint a complete picture of the system's strengths and weaknesses.

- **Model Performance and Efficacy:** These metrics quantify the quality of the final trained model.
  - Task-Specific Metrics: The primary measure of success is performance on the intended task. For classification, this includes accuracy, precision, recall, and the F1-score. The Area Under the Receiver Operating Characteristic (AUC-ROC) curve is also crucial for evaluating binary classifiers, especially in medical diagnoses where the trade-off between true positives and false positives is critical. In multi-class settings with imbalanced data, the Matthews Correlation Coefficient (MCC) or class-weighted F1-scores provide a more nuanced view than simple accuracy. For image segmentation tasks, which are common in FSL's medical applications, the Dice Similarity Coefficient (DSC) and Intersection over Union (IoU) are indispensable for measuring spatial overlap.
  - Robustness to Heterogeneity: A key challenge in any federated system is non-IID data. An FSL system must be evaluated on its ability to converge to a high-quality model even when client data distributions vary significantly. A thorough analysis should report not just the average performance of the final global model but also its performance variance across different client subsets to assess fairness and generalization.
- **System Efficiency and Resource Cost:** These metrics determine the practicality and scalability of an FSL algorithm.
  - Communication Cost: This remains the most significant bottleneck in distributed learning. A detailed evaluation must go beyond a single number and should report:
    - Total Data Transmitted: The cumulative volume (in MB or GB) of all data exchanged between clients and the server(s). This should be broken down into uplink (client-to-server) and downlink (server-to-client) traffic, as these can have different costs in asymmetric networks (e.g., cellular networks). This includes smashed data, gradients, and model parameters.
    - Number of Communication Rounds: The total number of synchronous rounds needed to reach convergence. Since each round incurs network latency, minimizing this count is vital.
  - Computational Cost: FSL's value proposition is tied to reducing the load on edge devices. Therefore, a granular analysis of the computation is essential:
    - Client-Side Load: Measured in FLOPs (Floating Point Operations), wall-clock time per local epoch, or peak memory usage. This directly reflects the feasibility of deploying the model on resource-constrained devices such as smartphones or IoT sensors.
    - Server-Side Load: The computational resources required by the server(s) to train the server-side model portions and aggregate client contributions.
  - Convergence Analysis: The rate of convergence is a critical indicator of overall efficiency. This is often visualized by plotting model accuracy against the number of communication rounds or the elapsed real time (wall-clock time). Time-to-accuracy, which measures the total time needed to achieve a predefined performance target (e.g., 90% accuracy), is a powerful holistic metric that implicitly combines both communication and computation efficiency.
- **Standardized Benchmarks and Datasets:** To ground research and enable fair comparisons, the community relies on a set of standard benchmarks.
  - Foundational Benchmarks: For algorithm development and theoretical analysis, image datasets such as MNIST, Fashion-MNIST, CIFAR-10, and CIFAR-100 are ubiquitous. Their relatively small size allows for rapid prototyping and testing of new ideas.
  - Real-World Application Benchmarks: To demonstrate practical utility, researchers turn to more complex domain-specific datasets. In healthcare, this includes medical imaging archives such as ChestX-ray14 [138], the ADNI dataset [139] for Alzheimer's research and the BraTS (Brain Tumor Segmentation) challenge dataset [140–144]. In natural language processing, the Sentiment140 dataset [145] is often used for federated sentiment analysis.
  - Simulation Environments: For applications in dynamic environments such as vehicular networks, evaluation often depends on data generated from sophisticated simulators. SUMO (Simulation of Urban MObility) [146] and CARLA [147] are open-source platforms that can model realistic traffic flows, vehicle mobility, and communication patterns, providing a rich source of data for testing DML protocols under realistic network conditions.

5.2. Implementation tools: Frameworks and simulators

The practical exploration and deployment of FSL is enabled by a growing ecosystem of open-source software. These tools provide the building blocks for communication, computation, and simulation, allowing researchers to focus on algorithmic innovation.

- **Software Libraries and Frameworks:**
  - *FedML* [148]: This is one of the most comprehensive open-source libraries for FL and DML. Its highly modular design and support for diverse communication backends (e.g., MQTT, gRPC) make it an excellent choice for FSL research. Researchers can leverage its existing infrastructure for managing clients and servers while implementing custom split model architectures and training loops. Its extensive benchmark suite also facilitates comparison with established baselines.
  - *Flower* [149]: A popular and lightweight framework designed to be agnostic to the specific machine learning library being used (e.g., TensorFlow, PyTorch). Its core strength is its simplicity and flexibility. Although primarily designed for FL, its abstractions for client and server logic can be easily adapted to implement the two-part computational flow of FSL. A researcher would typically define the client-side logic to execute the first part of the model and the server-side logic (or "strategy" in Flower's terminology) to handle the second part and the aggregation.

- *PySyft* [150]: A powerful library focused on private and secure DL. PySyft's main differentiator is its native support for advanced Privacy-Enhancing Technologies (PETs), including Multi-Party Computation (MPC), Homomorphic Encryption (HE), and Differential Privacy (DP). For researchers focusing on the security and privacy aspects of FSL, PySyft provides the tools to build systems where intermediate data remain encrypted even when processed by the server.
- *TensorFlow Federated (TFF)* [151]: Developed by Google, TFF is a highly scalable framework for decentralized data computations. Provides a lower-level expressive language for implementing novel federated algorithms. While this gives researchers immense control, it also comes with a steeper learning curve compared to frameworks like Flower. TFF is well-suited for researchers who need to design entirely new communication patterns or aggregation schemes for their FSL systems.

- **Network Simulators for High-Fidelity Evaluation:** To truly understand how an FSL algorithm will perform in the wild, particularly in wireless and mobile contexts, network simulators are essential. They allow for controlled and repeatable experiments under realistic network conditions that are impossible to replicate in simple lab setups.
  - *NS-3 (Network Simulator 3)* [152]: A powerful and widely used discrete-event network simulator. Provide detailed models for a wide array of networking technologies, from Wi-Fi and LTE/5G to vehicular communication protocols (DSRC/C-V2X). By integrating FSL code with NS-3, researchers can simulate a complete system, modeling the impact of network-layer phenomena such as packet loss, channel fading, signal interference, and client mobility on the performance of the learning algorithm.
  - *OMNeT++* [153]: A modular and extensible simulation framework. Its component-based architecture makes it particularly well-suited for modeling complex hierarchical systems. For example, a researcher could use OMNeT++ to simulate a large-scale FSL deployment with thousands of clients, multiple edge servers, and a central cloud aggregator, modeling the different network links and latencies at each level of the hierarchy. This enables the study of the scalability and performance of hierarchical aggregation strategies.

5.3. Quantitative analysis of research trends

This section presents several charts that quantitatively summarize our survey of 64 unique FSL papers.

5.3.1. Distribution of papers by application

The pie chart in Fig. 13 illustrates the relative proportion of papers across six application domains. Out of 64 papers, 37 (57.8%) focus on IoT and EC, followed by Wireless Communications (9 papers, 14.1%), Healthcare (7 papers, 10.9%), Vehicular Networks (7 papers, 10.9%), Earth Observation (2 papers, 3.1%), and Large Language Models (2 papers, 3.1%). This distribution shows that IoT and EC remains the primary driver of FSL research, while EO and LLMs represent smaller but emerging subfields that may benefit from increased attention.

5.3.2. Distribution of papers by publication year

The bar chart in Fig. 14 illustrates the number of FSL papers published per year from 2020 to 2025. We observe a gradual rise in 2022 (3 papers), a notable jump in 2023 (20 papers), and a peak in 2024 (35 papers). Five additional papers appeared in early 2025. This trend underscores the rapid growth of FSL research, particularly in 2023–2024, driven by expanding interest in privacy-preserving, resource-efficient DML.

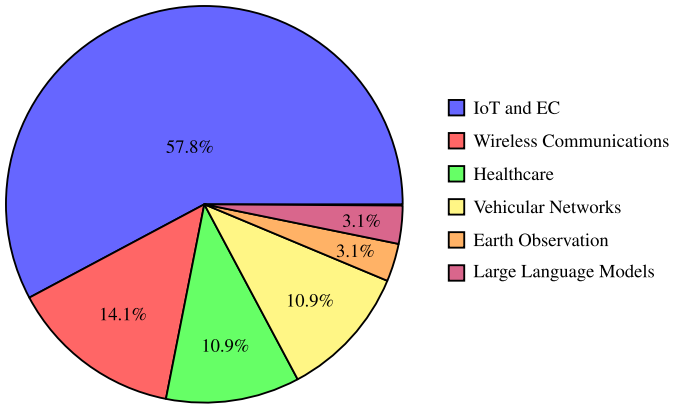


Fig. 13. Distribution of papers by application domains.

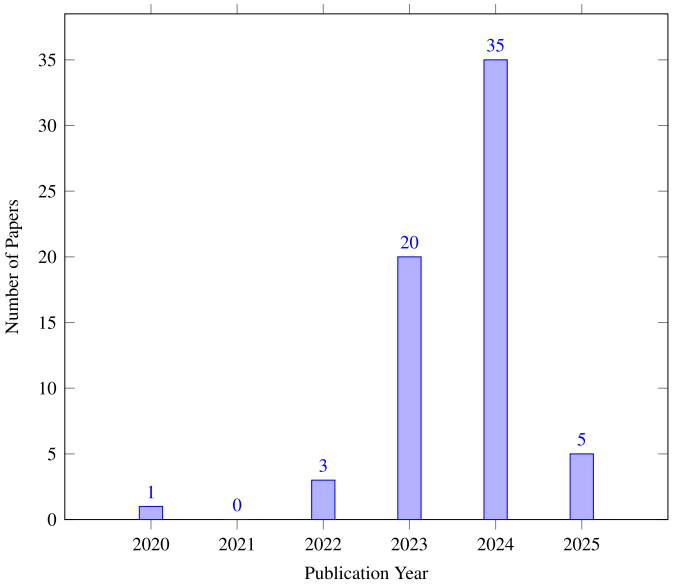


Fig. 14. Distribution of papers by publication year.

5.3.3. Frequency of evaluation metrics reported

The bar chart in Fig. 15 details how often common evaluation metrics appear in the surveyed literature. Among 64 papers, **Accuracy** is reported by 61 papers (≈95.3%), **Latency** (including convergence or training time) by 45 papers (≈70.3%), **Energy** (or energy efficiency) by 15 papers (≈23.4%), and **Privacy** (leakage/risk metrics) by 6 papers (≈9.4%). These counts highlight the community's strong emphasis on model accuracy and system efficiency, with fewer works explicitly measuring privacy leakage or energy consumption.

5.3.4. Challenges dealt with in FSL research

Each paper was classified according to the primary challenge it addresses. The pie chart in Fig. 16 shows that **Computational/System Constraints (resource limitations)** appear in 24 papers (37.5%), **Privacy/Security** in 11 papers (17.2%), **Heterogeneity** in 10 papers (15.6%), **Model Optimization** in 10 papers (15.6%), and **Communication Overhead** in 9 papers (14.1%). This breakdown reveals that while resource constraints are the most frequently tackled issue, privacy, heterogeneity, and model optimization also receive substantial attention.

5.3.5. Model/technique trends

The bar chart in Fig. 17 categorizes the primary modeling strategies and techniques employed across the 64 FSL papers. To focus on

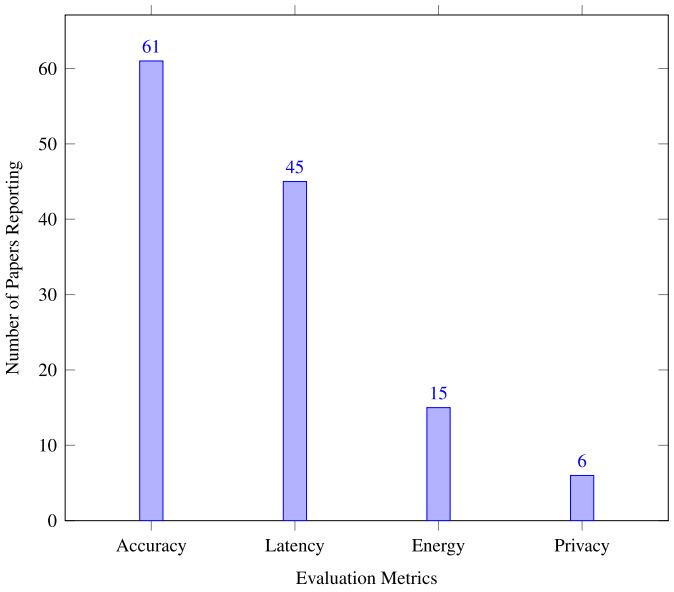


Fig. 15. Frequency of evaluation metrics reported.

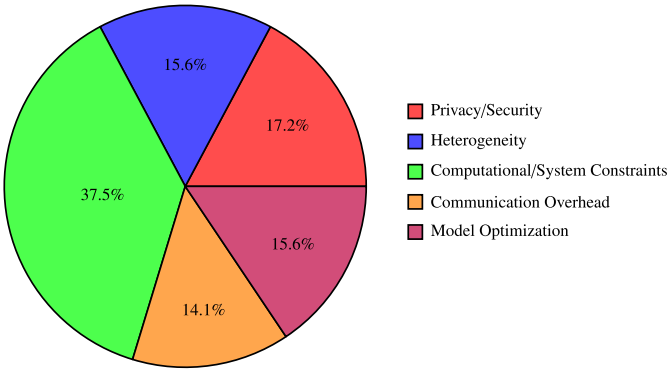


Fig. 16. Challenges dealt with in FSL research.

recurring themes, we list categories that appear in more than one paper as separate entries; all techniques that occur exactly once are grouped under "Other". The categories and their counts reveal the following:

- **Hierarchical/Multilayer (7 papers):** Approaches that introduce multi-tier or hierarchical splitting frameworks, often combining edge, fog, and cloud tiers, to balance computation and communication across different network layers.
- **Optimization-Based (7 papers):** Methods that rely on explicit mathematical optimization (e.g., KKT, SCA, BCD) to determine optimal split points, resource allocation, or convergence criteria.
- **Hybrid FL–SL Frameworks (5 papers):** Techniques that merge FL and SL paradigms, aiming to leverage the privacy benefits of SL with the aggregation mechanics of FL.
- **Adaptive/Dynamic Split (5 papers):** Algorithms that adjust where to "cut" the model during training in real time, based on device heterogeneity or network fluctuations.
- **Parallel/Edge–Cloud (5 papers):** Schemes that enable parallelism by distributing sub-models across multiple edge or cloud nodes, reducing training latency and balancing system load.
- **Privacy-Preserving (3 papers):** Designs that incorporate DP, homomorphic encryption, or secure aggregation specifically at the split boundary to protect sensitive activations.
- **Clustering/Consensus (3 papers):** Approaches that form clusters of clients based on similarity (e.g., data distribution or resource profile) and use consensus algorithms to synchronize split model updates within each cluster.
- **Model Splitting/Layering (3 papers):** Techniques focusing on how to partition neural network layers into client- and server-side segments, often evaluating different cut-layer configurations for accuracy and efficiency.
- **Communication/Storage Optimization (2 papers):** Methods dedicated to reducing communication overhead or intermediate data storage, using compression, quantization, or caching strategies to minimize bandwidth usage.
- **Generator–Personalization (Gen–Pers) (2 papers):** Schemes that combine global generation of shared representations (e.g., via GANs) with local personalization layers, balancing generalization and client-specific customization.
- **Block Sampling/Transformer (2 papers):** Studies that integrate transformer-based architectures or block sampling into split federated setups for large-scale vision tasks or medical imaging.
- **Robustness (3 papers):** Solutions that explicitly target robustness against noisy channels, packet loss, or adversarial attacks at the split boundary.
- **Other (17 papers):** This group aggregates all singleton-frequency categories, including but not limited to model pruning, self-supervised feature learning, cache-enabled SL, U-shaped network architectures, quality-adaptive segmentation, positive-label training, hypernetwork integration, gradient approximation, resource-aware codec switching via RL, and comparative evaluation studies. Each of these appears once in the surveyed corpus.

Together, these counts demonstrate that FSL research employs a wide variety of techniques. The prominence of hierarchical, optimization-based, hybrid, adaptive, and parallel frameworks (each with multiple papers) indicates that balancing system efficiency, privacy, and model performance remains the community's chief focus. Meanwhile, the large "Other" category underscores the diversity of emerging ideas, suggesting ongoing innovation in specialized or niche FSL methods.

5.3.6. Datasets/benchmarks usage

The bar chart in Fig. 18 shows how often different types of datasets and benchmarks appear in evaluations across the 64 surveyed papers. Categories with more than one paper are listed separately; all singleton-frequency datasets are grouped under "Other". The breakdown is as follows:

- **Synthetic or Simulated Data (28 papers, 43.8%)**: Papers that rely on artificially generated or simulated datasets to evaluate FSL algorithms, often for scalability tests or controlled experiments.
- **Standard Benchmarks (11 papers, 17.2%)**: Well-known datasets such as MNIST, CIFAR-10, and Tiny-ImageNet used to measure model performance under FSL constraints.
- **IoT Sensor Data (6 papers, 9.4%)**: Real-world IoT benchmarks (e.g., UCI HAR, smart meter readings, industrial Internet of Things datasets) reflecting common edge-computing scenarios.
- **Vehicular Data (6 papers, 9.4%)**: Datasets related to vehicular networks, such as traffic flow, semantic communication frames from UAVs, or RSU data for connected cars.
- **Medical Imaging (5 papers, 7.8%)**: Clinical and biomedical image datasets including chest X-rays, MRI scans, embryo segmentation images, and other medical imaging benchmarks.
- **Network Traffic Data (2 papers, 3.1%)**: Realistic network traffic traces or datasets used for predictive tasks in core network and non-terrestrial network scenarios.
- **Other (4 papers, 6.25%)**: Singleton-frequency datasets, such as continuous authentication logs, recommendation (e.g., MovieLens), language modeling corpora (e.g., WikiText), or papers where dataset usage is not applicable (architectural proposals without empirical benchmarks).

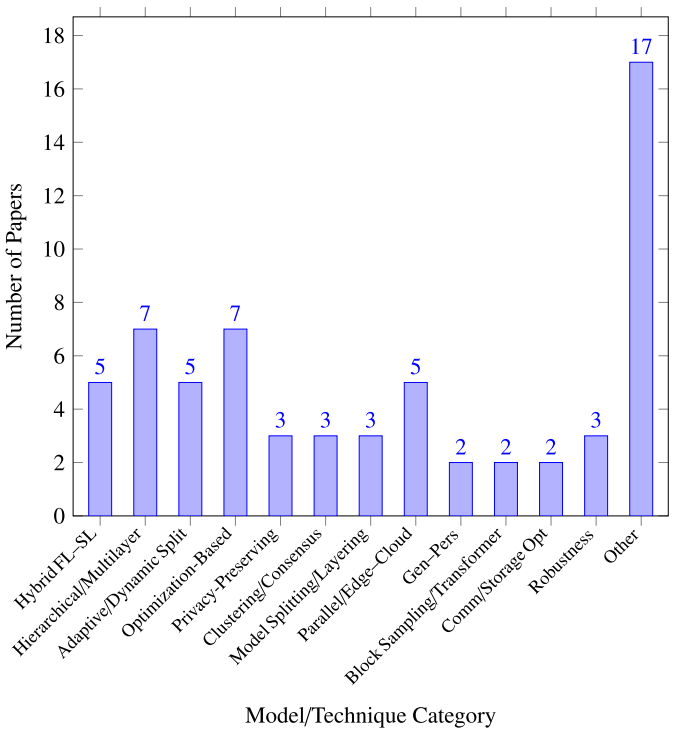


Fig. 17. Ditribution of papers by model and techniques used.

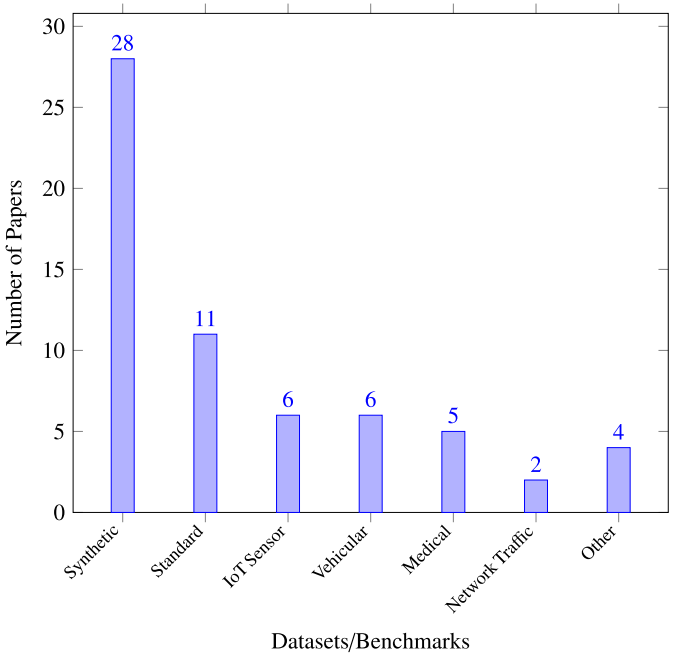


Fig. 18. Ditribution of papers by datasets and benchmarks used.

These counts reveal that almost half of the studies use synthetic or simulated data, facilitating controlled scalability experiments. Standard vision benchmarks remain popular for baseline comparisons, while domain-specific real-world datasets (IoT, vehicular, medical, network traffic) collectively account for over one-third of the evaluations. The relatively small "Other" category highlights that niche or highly specialized datasets appear infrequently but still contribute to validating FSL in diverse contexts.

5.4. Lessons learned

These visualizations provide a quantitative overview of current FSL research. The dominance of IoT and EC applications, the surge in publications during 2023–2024, and the prevalence of accuracy and latency metrics illustrate where the community's interests have concentrated. Likewise, resource constraints and model optimization appear as primary challenges, while techniques remain highly diverse, with many bespoke or hybrid solutions. Finally, although synthetic simulations are common, there is a growing usage of real-world benchmarks, especially in the medical imaging and IoT sensor domains, indicating a trend toward practical deployments. Together, these charts highlight key research trends, gaps, and opportunities for future work in FSL.

6. Future directions

As FSL continues to mature, several open avenues merit further investigation to fully realize its potential across diverse application domains. Building on the challenges highlighted in this survey, ranging from privacy preservation and system heterogeneity to resource constraints and communication bottlenecks, future research must explore adaptive, scalable, and robust solutions that can be seamlessly integrated into real-world environments. In particular, innovations in dynamic layer partitioning, hardware–software co-design, and advanced privacy mechanisms hold promise for enhancing FSL's efficiency and trustworthiness. Moreover, the integration of FSL with emerging paradigms such as foundation models, decentralized AI, and next-generation networks (e.g., 6G) can unlock new capabilities for distributed intelligence.

6.1. Integration with foundation models and large-scale architectures

The rise of large pre-trained models, such as Large Language Models (LLMs) [131], Vision Transformers (ViTs) [132], and multimodal architectures (e.g., CLIP, Flamingo), offers transformative opportunities for FSL. Future work should investigate how to leverage these foundation models in split learning scenarios, enabling resource-constrained clients to collaboratively fine-tune or utilize large architectures without exposing raw data. Key research directions include:

- **Efficient fine-tuning in split settings:** Parameter-efficient methods (LoRA [73], adapters [71], prompt tuning [74]) can be naturally integrated into FSL to adapt foundation models to downstream tasks while keeping client-side computation and communication low. The challenge lies in optimally placing trainable parameters and determining whether to freeze or partially fine-tune the pre-trained layers.
- **Dynamic splitting for Transformers:** The activation size of Transformer blocks grows with sequence length, making naive splitting inefficient. Algorithms that dynamically adjust the split point based on input characteristics (e.g., sequence length, attention head dimensionality) and network conditions are needed. Early ideas appear in block sampling [123] and adaptive compression [10].
- **Privacy-preserving attention mechanisms:** Self-attention requires global context, which may leak information when activations are shared. Designing attention variants that operate on encrypted or noised activations, or that use differential privacy at the attention output, is an open problem.
- **Multimodal foundation models:** Extending FSL to multimodal architectures (e.g., vision–language, vision–audio) can enable privacy-preserving collaborative learning across different data types. For example, in medical diagnostics, combining imaging data (X-rays) with electronic health records (text) could improve accuracy, but data silos hinder centralized training. FSL can split modality-specific encoders across clients and a fusion module on the server [136,137].

6.2. Adaptive layer partitioning and personalization

While dynamic cut-layer selection has been explored (e.g., HSFL [15], MSFTL [57]), future research should move toward fully adaptive partitioning guided by reinforcement learning or meta-learning. Such strategies could optimize the division of computational load in real time, accounting for fluctuating network conditions, device energy budgets, and task-specific accuracy requirements. Personalization techniques (e.g., SplitGP [91], PHSFL [82]) can be extended to ensure that each client's model segment aligns with its local data distribution while still contributing to a coherent global model. The trade-off between personalization and generalization remains a central challenge.

6.3. Hardware–software co-design and efficient computation

Co-optimization of hardware and software can greatly accelerate FSL. Dedicated accelerators (GPUs, TPUs, FPGAs) and efficient communication protocols (e.g., gRPC, MQTT) should be co-designed with FSL algorithms to meet latency, energy, and scalability targets. Joint design approaches can align model partitioning with hardware capabilities, improving throughput on edge devices and reducing overall system overhead. For ultra-low-power devices, techniques such as in-memory computing and near-data processing could be integrated with split architectures.

6.4. Deployment, benchmarking, and standardization

To transition FSL from theory to practice, real-world deployment and comprehensive benchmarking are essential. Establishing standardized datasets and evaluation frameworks across domains (IoT, healthcare, vehicular) will facilitate reproducible research and fair comparisons. Investigating deployment scenarios in industrial or clinical settings can uncover practical constraints and validate the benefits of FSL in production environments. Open-source platforms (FedML [62], Flower [63]) and testbeds can accelerate community-driven development. Standardization efforts (see Section 3.7) must evolve to cover FSL-specific aspects, such as split point negotiation and cross-vendor aggregation protocols.

6.5. Emerging network paradigms: Decentralized AI and 6G

Next-generation networks offer fertile ground for FSL innovation. Decentralized AI paradigms (e.g., peer-to-peer federated learning, gossip-based aggregation) can be combined with FSL to eliminate single points of failure and improve resilience in dynamic environments. 6G's ultra-high bandwidth, low latency, and native support for AI will enable new FSL applications, such as real-time collaborative perception among autonomous vehicles. Synergies with edge caching, network slicing, and aerial/space networks (satellites, UAVs, HAPs) represent further research avenues. The GFSTL architecture, with its hierarchical group-based design, is particularly well-suited for the multi-layer structure of 6G non-terrestrial networks.

6.6. Cross-domain and multi-modal extensions

Investigating FSL in multi-domain and multi-modal contexts is promising. For instance, collaborative learning across healthcare institutions combining imaging and genomic data, or joint IoT and language applications, can leverage split architectures to handle diverse data types. Domain adaptation and transfer learning techniques in FSL settings can enhance model generalizability when local data distributions vary widely among clients.

6.7. Advanced privacy, security, and robustness

Emerging privacy-preserving methods, homomorphic encryption, secure aggregation, differential privacy, must be tailored for FSL to strengthen trust in collaborative learning. Future research should design lightweight cryptographic protocols for split interactions and explore formal privacy guarantees for cut-layer activations. The robustness of FSL against adversarial attacks (poisoning, backdoors, evasion) and inference risks at split boundaries warrants thorough investigation. Hybrid defenses combining multiple mechanisms (e.g., DP + secure enclaves) are a promising direction.

6.8. Resource-efficient and green FSL

As sustainability becomes increasingly important, designing energy-efficient FSL systems is crucial. Research into model compression, pruning, and sparsity in split federated contexts can significantly reduce energy consumption. Algorithms that adaptively manage computational load and communication based on real-time resource availability will make FSL more viable for battery-powered and ultra-low-power devices.

6.9. GFSTL for 6G ITS over joint T/NTNs

Looking toward 6G, every component of an intelligent transportation system (ITS), vehicles, drones, roadside cameras, sensors, and control centers, will be networked. Drawing on the terrestrial vehicular and non-terrestrial network blueprints [34], a hierarchical GFSTL design can coordinate vehicles (VUs), UAVs, and fixed cameras with edge FSL servers (RSUs or base stations) and a HAP performing top-level FL aggregation. The goal is to fuse heterogeneous multimodal traffic data while guaranteeing millisecond-level responsiveness for autonomous-driving services.

Fig. 19 shows the end-to-end loop. Clients transmit smashed features to their nearest RSU/BS; those edge servers back-propagate and periodically relay their updated models to the HAP, which creates the global model and redistributes it. GFSTL equips 6G ITS with privacy-aware learning, graceful scalability, and resilience to node failures. Remaining hurdles include fine-grained resource orchestration across air–ground tiers, harmonizing highly heterogeneous data streams, and tuning hyper-parameters to guarantee stable convergence under real-time constraints.

6.10. Summary of future research directions

Table 19 summarizes the key future directions and their associated challenges. By addressing these directions, FSL can evolve into a robust, versatile framework capable of supporting the next generation of distributed, privacy-aware intelligent systems.

7. Summary and conclusions

This survey has provided a comprehensive and systematic exploration of Federated Split Learning (FSL), a transformative paradigm that synergizes the parallel processing capabilities of Federated Learning (FL) with the computational efficiency and privacy enhancements of Split Learning (SL). Through an in-depth analysis of over 100 recent research articles, we have traced the evolution of distributed machine learning from foundational paradigms to advanced frameworks, including Federated Split Transfer Learning (FSTL) and Generalized Federated Split Transfer Learning (GFSTL). Our structured taxonomy, accompanied by latency analysis and comparative tables, clarifies the trade-offs among scalability, privacy, computational load, and communication overhead that guide the selection of appropriate DML paradigms for specific applications.

A key contribution of this work is the systematic mapping of challenges inherent to FSL, privacy and security, heterogeneity, computational constraints, communication overhead, model optimization, and

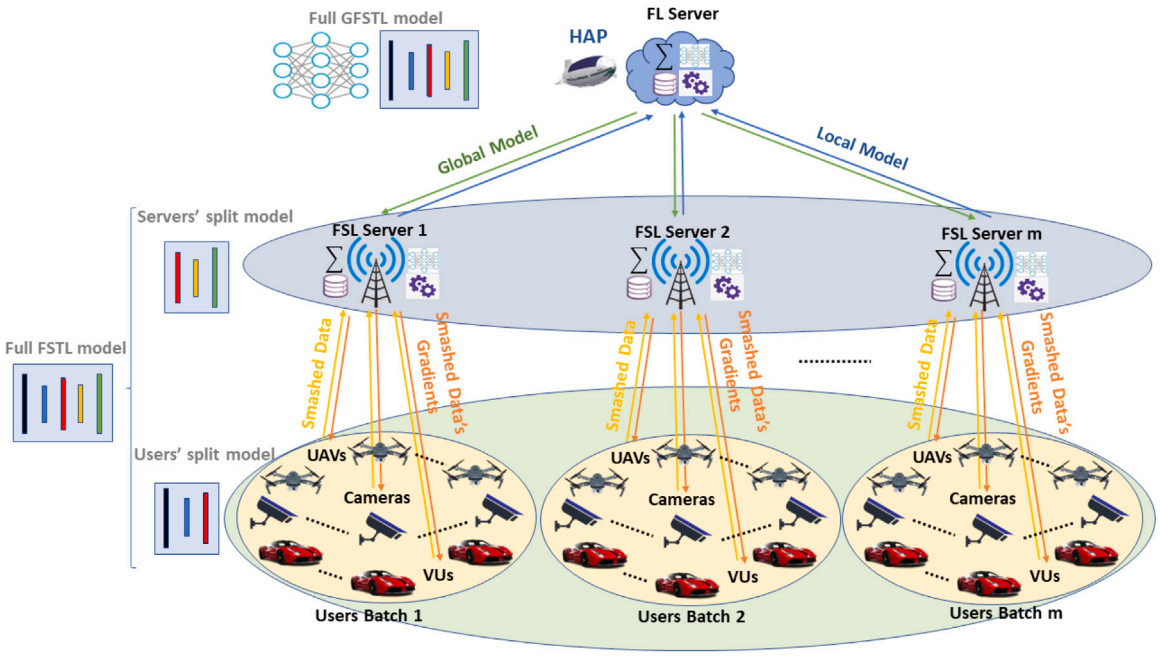


Fig. 19. Multilayer GFSTL architecture for 6G ITS.

Table 19
Summary of future research directions for FSL.

Direction	Key research questions	Enabling techniques
Foundation models	How to split and fine-tune LLMs/ViTs efficiently with privacy?	LoRA, adapters, block sampling, dynamic splitting
Adaptive partitioning	Real-time split point optimization under heterogeneity	RL, meta-learning, online convex optimization
Hardware–software co-design	Accelerators tailored for split execution; communication-aware partitioning	FPGA/TPU integration, in-memory computing
Standardization	FSL-specific protocols, cross-vendor interoperability	3GPP, IEEE P3652.1 extensions
6G	Ultra-low latency, massive connectivity, NTN integration	Network slicing, edge–cloud continuum, GFSTL
Multimodal learning	Combining vision, language, sensor data across silos	Modality-specific encoders, cross-attention
Privacy	Lightweight cryptography, formal guarantees for split data	HE, SMPC, DP, trusted execution environments
Green FSL	Energy-aware scheduling, sparsity, model pruning	Adaptive compression, event-driven computation

networking issues, to corresponding solution strategies. For each challenge, we have presented a problem formulation, a taxonomy of approaches, representative methods, and comparative tables that summarize pros, cons, overhead, and assumptions. This challenge–solution mapping serves as a practical reference for researchers designing robust FSL systems. The heart of the survey lies in its detailed, cross-domain analysis of FSL applications, reorganizing the literature according to deployment contexts: the Edge–IoT continuum, high-mobility networks, regulated domains such as healthcare, large-scale model training encompassing LLMs and Transformers, and remote sensing for Earth observation and non-terrestrial networks. This organization highlights common patterns, dominant constraints, and design guidelines that transcend traditional domain boundaries, with summary tables consolidating key features, techniques, and evaluation metrics for each context.

To ground this research in practice, we have discussed standardized evaluation methodologies, performance metrics, and implementation frameworks including FedML, Flower, PySyft, NS-3, and OMNeT++. A quantitative visualization of publication trends reveals the explosive growth of FSL research, with IoT and Edge Computing emerging as the dominant application area at 57.8% of papers, followed by wireless networks at 14.1% and healthcare at 10.9%. The analysis also shows that while accuracy and latency are reported in 95.3% and 70.3% of works respectively, explicit privacy quantification remains rare at only 9.4%, pointing to an important gap for future investigation. Looking forward, we have outlined key research directions including integration with foundation models and parameter-efficient fine-tuning, adaptive layer partitioning guided by reinforcement learning, hardware–software co-design for energy-efficient execution, deployment and standardization efforts, synergies with emerging 6G and non-terrestrial networks, cross-domain and multimodal extensions, advanced privacy and security mechanisms tailored for split architectures, and resource-efficient green FSL for sustainable edge intelligence.

In conclusion, FSL has matured into a versatile and powerful framework for distributed, privacy-aware machine learning. Its ability to balance computational efficiency with strong privacy guarantees makes it particularly attractive for resource-constrained edge environments, high-mobility scenarios, and regulated domains. The hierarchical GFSTL architecture, in particular, offers a scalable solution for next-generation intelligent systems, such as 6G-enabled intelligent transportation and Earth observation networks. By synthesizing the state of the art, identifying critical gaps, and charting future research avenues, this survey aims to serve as an essential roadmap for researchers and practitioners developing the next wave of distributed intelligence for the 6G and AI era.

List of abbreviations

Abbreviation	Full form
3GPP	3rd Generation Partnership Project
5G	Fifth Generation
6G	Sixth Generation
ADNI	Alzheimer's Disease Neuroimaging Initiative
AGIN	Aerial–Ground Integrated Network
AI	Artificial Intelligence
ASFV	Adaptive and Parallel Split Federated Learning for VEC
AUC-ROC	Area Under the Receiver Operating Characteristic
B5G	Beyond 5G
BraTS	Brain Tumor Segmentation
CML	Centralized Machine Learning
CNN	Convolutional Neural Network
D2D	Device-to-Device
DFL	Dynamic Federated Split Learning
DL	Deep Learning
DML	Distributed Machine Learning
DP	Differential Privacy
DSC	Dice Similarity Coefficient
EC	Edge Computing
EO	Earth Observation
EPSL	Efficient Parallel Split Learning
FedAvg	Federated Averaging
FL	Federated Learning
FM	Foundation Model
FPGA	Field-Programmable Gate Array
FSL	Federated Split Learning
FSTL	Federated Split Transfer Learning
GAN	Generative Adversarial Network
GDPR	General Data Protection Regulation
GFSTL	Generalized Federated Split Transfer Learning
GP	Gaussian Process
GPU	Graphics Processing Unit
HAP	High-Altitude Platform
HAR	Human Activity Recognition
HE	Homomorphic Encryption
HIPAA	Health Insurance Portability and Accountability Act
HSFL	Heterogeneous/Hierarchical Split Federated Learning
IETF	Internet Engineering Task Force
IIoT	Industrial Internet of Things
IoMT	Internet of Medical Things
IoT	Internet of Things
ITS	Intelligent Transportation System
ITU-T	International Telecom Union – Telecom Standardization Sector
KL	Kullback–Leibler
KKT	Karush–Kuhn–Tucker
LEO	Low Earth Orbit
LLM	Large Language Model
LoRA	Low-Rank Adaptation
LSTM	Long Short-Term Memory
MCC	Matthews Correlation Coefficient
MINLP	Mixed-Integer Non-Linear Programming
ML	Machine Learning
MPC	Multi-Party Computation
MPNN	Message Passing Neural Network
MSFTL	Mobility-Aware Split-Federated with Transfer Learning
NN	Neural Network
non-IID	non-Independent and Identically Distributed
NS-3	Network Simulator 3
NTN	Non-Terrestrial Network
O-RAN	Open Radio Access Network
PDF	Probability Density Function
PET	Privacy-Enhancing Technology
PRISMA	Preferred Reporting Items for Systematic Reviews & Meta-Analyses
RL	Reinforcement Learning
RSU	Road-Side Unit
SCA	Successive Convex Approximation
SL	Split Learning
SMPC	Secure Multi-Party Computation
SoC	System-on-Chip
SUMO	Simulation of Urban MObility
T/NTN	Terrestrial/Non-Terrestrial Network
TFF	TensorFlow Federated
TL	Transfer Learning
TPU	Tensor Processing Unit
UAV	Unmanned Aerial Vehicle
V2I	Vehicle-to-Infrastructure
VEC	Vehicular Edge Computing
VEI	Vehicular Edge Intelligence
ViT	Vision Transformer
XR	eXtended Reality
ZO	Zeroth-Order Optimization

CRediT authorship contribution statement

David Naseh: Writing – review & editing, Writing – original draft, Visualization, Validation, Software, Methodology, Investigation, Formal analysis, Data curation, Conceptualization. **Arash Bozorgchenani:** Writing – review & editing, Writing – original draft, Supervision, Resources, Investigation, Conceptualization, Methodology. **Swapnil Sadashiv Shinde:** Writing – review & editing, Conceptualization, Visualization. **Daniele Tarchi:** Writing – review & editing, Writing – original draft, Supervision, Resources, Project administration, Investigation, Funding acquisition, Conceptualization.

Declaration of competing interest

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

Data availability

No data was used for the research described in the article.

References

[1] C. Thapa, P.C. Mahawaga Arachchige, S. Camtepe, L. Sun, SplitFed: When federated learning meets split learning, Proc. AAAI Conf. Artif. Intell. 36 (8) (2022) 8485–8493, http://dx.doi.org/10.1609/aaai.v36i8.20825.

[2] P. Kairouz, H.B. McMahan, B. Avent, A. Bellet, M. Bennis, A.N. Bhagoji, K. Bonawitz, Z. Charles, G. Cormode, R. Cummings, R.G.L. D'Oliveira, H. Eichner, S.E. Rouayheb, D. Evans, J. Gardner, Z. Garrett, A. Gascón, B. Ghazi, P.B. Gibbons, M. Gruteser, Z. Harchaoui, C. He, L. He, Z. Huo, B. Hutchinson, J. Hsu, M. Jaggi, T. Javidi, G. Joshi, M. Khodak, J. Konecný, A. Korolova, F. Koushanfar, S. Koyejo, T. Lepoint, Y. Liu, P. Mittal, M. Mohri, R. Nock, A. Özgür, R. Pagh, H. Qi, D. Ramage, R. Raskar, M. Raykova, D. Song, W. Song, S.U. Stich, Z. Sun, A.T. Suresh, F. Tramèr, P. Vepakomma, J. Wang, L. Xiong, Z. Xu, Q. Yang, F.X. Yu, H. Yu, S. Zhao, Advances and open problems in federated learning, Found. Trends® Mach. Learn. 14 (1–2) (2021) 1–210, http://dx.doi.org/10.1561/2200000083.

[3] E. Samikwa, A. Di Maio, T. Braun, DFL: Dynamic federated split learning in heterogeneous IoT, IEEE Trans. Mach. Learn. Commun. Netw. 2 (2024) 733–752, http://dx.doi.org/10.1109/TMLCN.2024.3409205.

[4] A. Fallah, A. Mokhtari, A. Ozdaglar, Personalized federated learning: A meta-learning approach, 2020, arXiv preprint arXiv:2002.07948.

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

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

[7] E. Bagdasaryan, A. Veit, Y. Hua, D. Estrin, V. Shmatikov, How to backdoor federated learning, in: International Conference on Artificial Intelligence and Statistics, PMLR, 2020, pp. 2938–2948.

[8] X. Zhang, X. Zhu, J. Wang, H. Yan, H. Chen, W. Bao, Federated learning with adaptive communication compression under dynamic bandwidth and unreliable networks, Inform. Sci. 540 (2020) 242–262.

[9] D. Naseh, S.S. Shinde, D. Tarchi, Enabling intelligent vehicular networks through distributed learning in the non-terrestrial networks 6G vision, in: European Wireless 2023; 28th European Wireless Conference, VDE, 2023, pp. 136–141.

[10] Z. Lin, G. Zhu, Y. Deng, X. Chen, Y. Gao, K. Huang, Y. Fang, Efficient parallel split learning over resource-constrained wireless edge networks, IEEE Trans. Mob. Comput. 23 (10) (2024) 9224–9239, http://dx.doi.org/10.1109/TMC.2024.3359040.

[11] B. Camajori Tedeschini, M. Brambilla, M. Nicoli, Split consensus federated learning: An approach for distributed training and inference, IEEE Access 12 (2024) 119535–119549, http://dx.doi.org/10.1109/ACCESS.2024.3446577.

[12] P. Vepakomma, O. Gupta, T. Swedish, R. Raskar, Split learning for health: Distributed deep learning without sharing raw patient data, 2018, arXiv preprint arXiv:1812.00564.

[13] J. Yang, Y. Liu, SCALA: Split federated learning with concatenated activations and logit adjustments, 2024, arXiv preprint arXiv:2405.04875.

[14] D.-J. Han, D.-Y. Kim, M. Choi, D. Nickel, J. Moon, M. Chiang, C.G. Brinton, Federated split learning with joint personalization-generalization for inference-stage optimization in wireless edge networks, IEEE Trans. Mob. Comput. 23 (6) (2024) 7048–7065, http://dx.doi.org/10.1109/TMC.2023.3331690.

[15] Y. Sun, G. Hu, Y. Teng, D. Cai, Split federated learning over heterogeneous edge devices: Algorithm and optimization, in: 2025 IEEE Wireless Communications and Networking Conference, WCNC, 2025, pp. 01–06, http://dx.doi.org/10.1109/WCNC61545.2025.10978241.

[16] L. Luo, X. Zhang, Federated split learning via mutual knowledge distillation, IEEE Trans. Netw. Sci. Eng. 11 (3) (2024) 2729–2741, http://dx.doi.org/10.1109/TNSE.2023.3348461.

[17] S. Sindhura, B. Annappa, D.N. Sachin, Federated split learning with hypernetworks for medical image classification, in: 2024 15th International Conference on Computing Communication and Networking Technologies, ICCCNT, 2024, pp. 1–6, http://dx.doi.org/10.1109/ICCCNT61001.2024.10725552.

[18] X. Qiang, Z. Chang, Y. Hu, L. Liu, T. Hämäläinen, Adaptive and parallel split federated learning in vehicular edge computing, IEEE Internet Things J. 12 (5) (2025) 4591–4604, http://dx.doi.org/10.1109/JIOT.2024.3479158.

[19] H. Ao, H. Tian, W. Ni, Federated split learning for distributed intelligence with resource-constrained devices, in: 2024 IEEE International Conference on Communications Workshops, ICC Workshops, 2024, pp. 798–803, http://dx.doi.org/10.1109/ICCWorkshops59551.2024.10615309.

[20] D. Naseh, S.S. Shinde, D. Tarchi, Distributed learning framework for earth observation on multilayer non-terrestrial networks, in: 2024 IEEE International Conference on Machine Learning for Communication and Networking, ICMLCN, 2024, pp. 1–2, http://dx.doi.org/10.1109/ICMLCN59089.2024.10625007.

[21] S. Li, M. Iqbal, A.K. Bashir, W. Xinheng, Privacy preservation in artificial intelligence-enabled healthcare analytics, Hum.-Centric Comput. Inf. Sci. (2021).

[22] H. Hafi, B. Brik, P.A. Frangoudis, A. Ksentini, M. Bagaa, Split federated learning for 6G enabled-networks: Requirements, challenges, and future directions, IEEE Access 12 (2024) 9890–9930, http://dx.doi.org/10.1109/ACCESS.2024.3351600.

[23] D. Naseh, S.S. Shinde, D. Tarchi, T. DeCola, Distributed intelligent framework for remote area observation on multilayer non-terrestrial networks, in: 2024 IEEE International Mediterranean Conference on Communications and Networking, MeditCom, 2024, pp. 1–6, http://dx.doi.org/10.1109/MeditCom61057.2024.10621222.

[24] X. Liang, Y. Xu, Y.-e. Lin, C. Zhang, Federated split learning via dynamic aggregation and homomorphic encryption on non-IID data, J. Supercomput. 81 (1) (2025) 63.

[25] J. Ndeko, S. Shaon, A. Beal, A. Sahoo, D.C. Nguyen, Federated split learning for human activity recognition with differential privacy, in: 2025 IEEE 22nd Consumer Communications & Networking Conference, CCNC, 2025, pp. 1–2, http://dx.doi.org/10.1109/CCNC54725.2025.10976058.

[26] Z. Hu, T. Zhou, B. Wu, C. Chen, Y. Wang, A review and experimental evaluation on split learning, Future Internet 17 (2) (2025) http://dx.doi.org/10.3390/fi17020087, URL https://www.mdpi.com/1999-5903/17/2/87.

[27] W. Zhang, D. Zhan, H. Yu, L. Zhang, B. Zhao, X. Du, Z. Tian, Distributed machine learning for next-generation communication networks: A survey on privacy, fairness, efficiency, and trade-offs, Inf. Fusion 126 (2026) 103657, http://dx.doi.org/10.1016/j.inffus.2025.103657, URL https://www.sciencedirect.com/science/article/pii/S1566253525007298.

[28] Q. Duan, S. Hu, R. Deng, Z. Lu, Combined federated and split learning in edge computing for ubiquitous intelligence in internet of things: State-of-the-art and future directions, Sensors 22 (16) (2022) http://dx.doi.org/10.3390/s22165983, URL https://www.mdpi.com/1424-8220/22/16/5983.

[29] G.S. Hukkeri, R.H. Goudar, G.M. Dhananjaya, V.N. Rathod, S. Ankalaki, A comprehensive survey on split-fed learning: Methods, innovations, and future directions, IEEE Access 13 (2025) 46312–46333, http://dx.doi.org/10.1109/ACCESS.2025.3547641.

[30] M. Tajabadi, R. Martin, D. Heider, Privacy-preserving decentralized learning methods for biomedical applications, Comput. Struct. Biotechnol. J. 23 (2024) 3281–3287, http://dx.doi.org/10.1016/j.csbj.2024.08.024, URL https://www.sciencedirect.com/science/article/pii/S2001037024002824.

[31] M.H. Alsharif, R. Kannadasan, W. Wei, K.S. Nisar, A.-H. Abdel-Aty, A contemporary survey of recent advances in federated learning: Taxonomies, applications, and challenges, Internet Things 27 (2024) 101251, http://dx.doi.org/10.1016/j.iot.2024.101251.

[32] D. Naseh, A. Bozorgchenani, S.S. Shinde, D. Tarchi, Unified distributed machine learning for 6G intelligent transportation systems: A hierarchical approach for terrestrial and non-terrestrial networks, Network 5 (3) (2025) http://dx.doi.org/10.3390/network5030041, URL https://www.mdpi.com/2673-8732/5/3/41.

[33] M.J. Page, J.E. McKenzie, P.M. Bossuyt, I. Boutron, T.C. Hoffmann, C.D. Mulrow, L. Shamseer, J.M. Tetzlaff, E.A. Akl, S.E. Brennan, R. Chou, J. Glanville, J.M. Grimshaw, A. Hróbjartsson, M.M. Lalu, T. Li, E.W. Loder, E. Mayo-Wilson, S. McDonald, L.A. McGuinness, L.A. Stewart, J. Thomas, A.C. Tricco, V.A. Welch, P. Whiting, D. Moher, The PRISMA 2020 statement: an updated guideline for reporting systematic reviews, BMJ 372 (2021) http://dx.doi.org/10.1136/bmj.n71, arXiv:https://www.bmj.com/content/372/bmj.n71.full.pdf. URL https://www.bmj.com/content/372/bmj.n71.

[34] D. Naseh, S.S. Shinde, D. Tarchi, Network sliced distributed learning-as-a-service for internet of vehicles applications in 6G non-terrestrial network scenarios, J. Sens. Actuator Netw. 13 (1) (2024) 14, http://dx.doi.org/10.3390/jsan13010014.

[35] J. Gubbi, R. Buyya, S. Marusic, M. Palaniswami, Internet of Things (IoT): A vision, architectural elements, and future directions, Future Gener. Comput. Syst. 29 (7) (2013) 1645–1660, http://dx.doi.org/10.1016/j.future.2013.01.010.

[36] A. Liotta, et al., A survey on decentralized federated learning, 2019, arXiv preprint arXiv:1908.09655.

[37] A. Bellet, et al., Decentralized machine learning: A survey, Found. Trends Mach. Learn. 14 (4) (2021) 345–480.

[38] S. Boyd, et al., Randomized gossip algorithms, IEEE Trans. Inform. Theory 52 (6) (2006) 2508–2530, http://dx.doi.org/10.1109/TIT.2006.874516.

[39] H. Kim, et al., Blockchain for federated learning: A survey, in: IEEE International Conference on Blockchain, 2020, pp. 123–130.

[40] J. Verbraeken, et al., A survey on distributed machine learning, ACM Comput. Surv. 53 (2) (2020) 1–41.

[41] L. Ridolfi, D. Naseh, S.S. Shinde, D. Tarchi, Implementation and evaluation of a federated learning framework on raspberry PI platforms for IoT 6G applications, Future Internet 15 (11) (2023) 358, http://dx.doi.org/10.3390/fi15110358.

[42] D.C. Nguyen, M. Ding, P.N. Pathirana, A. Seneviratne, J. Li, H.V. Poor, Federated learning for internet of things: A comprehensive survey, IEEE Commun. Surv. Tutor. 23 (3) (2021) 1622–1658.

[43] P. Qi, D. Chiaro, A. Guzzo, M. Ianni, G. Fortino, F. Piccialli, Model aggregation techniques in federated learning: A comprehensive survey, Future Gener. Comput. Syst. 150 (2024) 272–293, http://dx.doi.org/10.1016/j.future.2023.09.008.

[44] W.Y.B. Lim, N.C. Luong, D.T. Hoang, Y. Jiao, Y.-C. Liang, Q. Yang, D. Niyato, C. Miao, Federated learning in mobile edge networks: A comprehensive survey, IEEE Commun. Surv. Tutor. 22 (3) (2020) 2031–2063.

[45] D.-J. Han, M. Choi, J. Park, J. Moon, FedMes: Speeding up federated learning with multiple edge servers, IEEE J. Sel. Areas Commun. 39 (12) (2021) 3870–3885.

[46] N. Obi, A review of split learning and federated learning: Challenges and synergies, EngrXiv (2024) http://dx.doi.org/10.31224/3848, Preprint. URL https://engrxiv.org/preprint/view/3848.

[47] K. Cheng, et al., Vertical federated learning: Concepts, advances, and challenges, IEEE Trans. Knowl. Data Eng. (2021).

[48] L. Romand, et al., Vertical federated learning: A structured literature review, 2023, arXiv preprint arXiv:2304.08862.

[49] P. Li, C. Guo, Y. Xing, Y. Shi, L. Feng, F. Zhou, Core network traffic prediction based on vertical federated learning and split learning, Sci. Rep. 14 (1) (2024) 4663, http://dx.doi.org/10.1038/s41598-024-53193-y.

[50] M. Wu, et al., Federated split learning with data and label privacy preservation in vehicular networks, IEEE Trans. Veh. Technol. 73 (1) (2024) 1223–1238, http://dx.doi.org/10.1109/TVT.2023.3304176.

[51] O. Gupta, R. Raskar, Distributed learning of deep neural network over multiple agents, J. Netw. Comput. Appl. 116 (2018) 1–8, http://dx.doi.org/10.1016/j.jnca.2018.05.003, URL https://www.sciencedirect.com/science/article/pii/S1084804518301590.

[52] E. Erdoğan, A. Küpçü, A.E. Çiçek, UnSplit: Data-oblivious model inversion, model stealing, and label inference attacks against split learning, in: Proceedings of the 21st Workshop on Privacy in the Electronic Society, WPES '22, Association for Computing Machinery, New York, NY, USA, 2022, pp. 115–124, http://dx.doi.org/10.1145/3559613.3563201.

[53] Y. Gao, M. Kim, S. Abuadbba, Y. Kim, C. Thapa, K. Kim, S.A. Camtep, H. Kim, S. Nepal, End-to-end evaluation of federated learning and split learning for internet of things, in: 2020 International Symposium on Reliable Distributed Systems, SRDS, 2020, pp. 91–100, http://dx.doi.org/10.1109/SRDS51746.2020.00017.

[54] J. Liu, H. Xu, L. Wang, Y. Xu, C. Qian, J. Huang, H. Huang, Adaptive asynchronous federated learning in resource-constrained edge computing, IEEE Trans. Mob. Comput. 22 (2) (2021) 674–690.

[55] R.A. Albuquerque, L.P. Dias, M. Ziazet, K. Vandikas, S. Ickin, B. Jaumard, C. Natalino, L. Wosinska, P. Monti, E. Wong, Asynchronous federated split learning, in: 2024 IEEE 8th International Conference on Fog and Edge Computing, ICFEC, 2024, pp. 11–18, http://dx.doi.org/10.1109/ICFEC61590.2024.00010.

[56] J. Yang, Y. Liu, GAS: Generative activation-aided asynchronous split federated learning, 2024, arXiv preprint arXiv:2409.01251.

[57] G. Zheng, Q. Ni, K. Navaie, H. Pervaiz, G. Min, A. Kaushik, C. Zarakovitis, Mobility-aware split-federated with transfer learning for vehicular semantic communication networks, IEEE Internet Things J. 11 (10) (2024) 17237–17248, http://dx.doi.org/10.1109/JIOT.2024.3360230.

[58] Y. Liao, Y. Xu, H. Xu, Z. Yao, L. Huang, C. Qiao, Parallelsfl: A novel split federated learning framework tackling heterogeneity issues, in: Proceedings of the 30th Annual International Conference on Mobile Computing and Networking, 2024, pp. 845–860, http://dx.doi.org/10.1145/3636534.3690665.

[59] Z. Cheng, X. Xia, M. Liwang, X. Fan, Y. Sun, X. Wang, L. Huang, CHEESE: Distributed clustering-based hybrid federated split learning over edge networks, IEEE Trans. Parallel Distrib. Syst. 34 (12) (2023) 3174–3191, http://dx.doi.org/10.1109/TPDS.2023.3322755.

[60] T. Li, A.K. Sahu, M. Zaheer, M. Sanjabi, A. Talwalkar, V. Smith, Federated optimization in heterogeneous networks, 2020, arXiv:1812.06127. URL https://arxiv.org/abs/1812.06127.

[61] J. Wang, et al., Tackling the objective inconsistency problem in heterogeneous federated optimization, Adv. Neural Inf. Process. Syst. (NeurIPS) 33 (2020) 7611–7623.

[62] C. He, et al., FedML: A research library and benchmark for federated machine learning, 2020, arXiv preprint arXiv:2007.13518.

[63] D.J. Beutel, et al., Flower: A friendly federated learning research framework, 2022, arXiv preprint arXiv:2007.14390.

[64] Z. Zhang, A. Pinto, V. Turina, F. Esposito, I. Matta, Privacy and efficiency of communications in federated split learning, IEEE Trans. Big Data 9 (5) (2023) 1380–1391, http://dx.doi.org/10.1109/TBDATA.2023.3280405.

[65] T. Qin, G. Cheng, Y. Wei, Z. Yao, Hier-SFL: Client-edge-cloud collaborative traffic classification framework based on hierarchical federated split learning, Future Gener. Comput. Syst. 149 (2023) 12–24.

[66] Y. Jia, B. Liu, X. Zhang, F. Dai, A. Khan, L. Qi, W. Dou, Model pruning-enabled federated split learning for resource-constrained devices in artificial intelligence empowered edge computing environment, ACM Trans. Sens. Netw. (2024) http://dx.doi.org/10.1145/3687478.

[67] Y. Liang, P. Kortoçi, P. Zhou, L.-H. Lee, A. Mehrabi, P. Hui, S. Tarkoma, J. Crowcroft, Federated split GANs for collaborative training with heterogeneous devices, Softw. Impacts 14 (2022) 100436, http://dx.doi.org/10.1016/j.simpa.2022.100436.

[68] D. Naseh, M. Abdollahpour, D. Tarchi, Real-world implementation and performance analysis of distributed learning frameworks for 6G IoT applications, Information 15 (4) (2024) 190, http://dx.doi.org/10.3390/info15040190.

[69] Z. Zhao, L. Alzubaidi, J. Zhang, Y. Duan, Y. Gu, A comparison review of transfer learning and self-supervised learning: Definitions, applications, advantages and limitations, Expert Syst. Appl. 242 (2024) 122807, http://dx.doi.org/10.1016/j.eswa.2023.122807.

[70] N. Girelli Consolaro, S.S. Shinde, D. Naseh, D. Tarchi, Analysis and performance evaluation of transfer learning algorithms for 6G wireless networks, Electronics 12 (15) (2023) 3327, http://dx.doi.org/10.3390/electronics12153327.

[71] N. Houlsby, et al., Parameter-efficient transfer learning for NLP, in: International Conference on Machine Learning, ICML, 2019, pp. 2790–2799.

[72] J. Yosinski, et al., How transferable are features in deep neural networks? in: Advances in Neural Information Processing Systems, NeurIPS, 2014, pp. 3320–3328.

[73] J.E. Hu, Y. Shen, P. Wallis, Z. Allen-Zhu, Y. Li, S. Wang, W. Chen, LoRA: Low-rank adaptation of large language models, 2021, ArXiv abs/2106.09685. URL https://api.semanticscholar.org/CorpusID:235458009.

[74] B. Lester, et al., The power of scale for parameter-efficient prompt tuning, in: Conference on Empirical Methods in Natural Language Processing, EMNLP, 2021, pp. 3045–3059.

[75] D. Naseh, S.S. Shinde, D. Tarchi, Multi-layer distributed learning for intelligent transportation systems in 6G aerial-ground integrated networks, in: 2024 Joint European Conference on Networks and Communications & 6G Summit (EuCNC/6G Summit), 2024, pp. 711–716, http://dx.doi.org/10.1109/EuCNC/6GSummit60053.2024.10597130.

[76] K. Zhao, Z. Yang, C. Huang, X. Chen, Z. Zhang, FedsLLM: Federated split learning for large language models over communication networks, in: 2024 International Conference on Ubiquitous Communication, Ucom, 2024, pp. 438–443, http://dx.doi.org/10.1109/Ucom62433.2024.10695888.

[77] Z. Shu, H. Zhao, B. Xu, S. Qiao, Y. Xu, J. Sun, Q. Wang, Privacy-preserved federated split learning in industrial internet of things, in: 2024 IEEE/CIC International Conference on Communications in China, ICCC, 2024, pp. 633–638, http://dx.doi.org/10.1109/ICCC62479.2024.10681937.

[78] C. Shiranthika, Z.H. Kafshgari, P. Saeedi, I.V. Bajić, SplitFed resilience to packet loss: Where to split, that is the question, in: International Conference on Medical Image Computing and Computer-Assisted Intervention, Springer, 2023, pp. 367–377, http://dx.doi.org/10.1007/978-3-031-47401-9_35.

[79] Z.H. Kafshgari, I.V. Bajic, P. Saeedi, Smart split-federated learning over noisy channels for embryo image segmentation, in: ICASSP 2023 - 2023 IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP, 2023, pp. 1–5, http://dx.doi.org/10.1109/ICASSP49357.2023.10095067.

[80] S. Zhang, Z. Zhao, D. Liu, Y. Cao, H. Tang, S. You, Edge-assisted U-shaped split federated learning with privacy-preserving for Internet of Things, Expert Syst. Appl. 262 (2025) 125494, http://dx.doi.org/10.1016/j.eswa.2024.125494, URL https://www.sciencedirect.com/science/article/pii/S0957417424023613.

[81] M. Wazzeh, M. Arafeh, H. Sami, H. Ould-Slimane, C. Talhi, A. Mourad, H. Otrok, CRSFL: Cluster-based resource-aware split federated learning for continuous authentication, J. Netw. Comput. Appl. 231 (2024) 103987, http://dx.doi.org/10.1016/j.jnca.2024.103987, URL https://www.sciencedirect.com/science/article/pii/S1084804524001644.

[82] M.-F. Pervej, A.F. Molisch, Personalized hierarchical split federated learning in wireless networks, 2025, arXiv:2411.06042. URL https://arxiv.org/abs/2411.06042.

[83] C. Tian, L. Li, K. Tam, Y. Wu, C.-Z. Xu, Breaking the memory wall for heterogeneous federated learning via model splitting, IEEE Trans. Parallel Distrib. Syst. 35 (12) (2024) 2513–2526, http://dx.doi.org/10.1109/TPDS.2024.3480115.

[84] Y. Li, D. Qin, H.V. Poor, et al., Introducing edge intelligence to smart meters via federated split learning, Nat. Commun. 15 (1) (2024) 9044, http://dx.doi.org/10.1038/s41467-024-53352-9.

[85] L.U. Khan, M. Guizani, A. Al-Fuqaha, C.S. Hong, D. Niyato, Z. Han, A joint communication and learning framework for hierarchical split federated learning, IEEE Internet Things J. 11 (1) (2024) 268–282, http://dx.doi.org/10.1109/JIOT.2023.3315673.

[86] M. Chawla, G.R. Gupta, S. Gaddam, M. Wadhwa, Beyond federated learning for IoT: Efficient split learning with caching and model customization, IEEE Internet Things J. 11 (20) (2024) 32617–32630, http://dx.doi.org/10.1109/JIOT.2024.3424660.

[87] W. Zhang, T. Zhou, Q. Lu, Y. Yuan, A. Tolba, W. Said, FedSL: A communication-efficient federated learning with split layer aggregation, IEEE Internet Things J. 11 (9) (2024) 15587–15601, http://dx.doi.org/10.1109/JIOT.2024.3350241.

[88] Y. Mu, C. Shen, Communication and storage efficient federated split learning, in: ICC 2023 - IEEE International Conference on Communications, 2023, pp. 2976–2981, http://dx.doi.org/10.1109/ICC45041.2023.10278891.

[89] V.A. Le, J. Haga, Y. Tanimura, et al., SFETEC: Split-federated learning scheme optimized for thing-edge-cloud environment, IEEE e-Sci. (2024) http://dx.doi.org/10.1109/e-science62913.2024.10678725.

[90] Y. Liao, Y. Xu, H. Xu, L. Wang, Z. Yao, C. Qiao, MergeSFL: Split federated learning with feature merging and batch size regulation, in: 2024 IEEE 40th International Conference on Data Engineering, ICDE, 2024, pp. 2054–2067, http://dx.doi.org/10.1109/ICDE60146.2024.00164.

[91] D.-J. Han, D.-Y. Kim, M. Choi, C.G. Brinton, J. Moon, SplitGP: Achieving both generalization and personalization in federated learning, in: IEEE INFOCOM 2023 - IEEE Conference on Computer Communications, 2023, pp. 1–10, http://dx.doi.org/10.1109/INFOCOM53939.2023.10229027.

[92] J. Qin, X. Zhang, B. Liu, J. Qian, A split-federated learning and edge-cloud based efficient and privacy-preserving large-scale item recommendation model, J. Cloud Comput. 12 (1) (2023) 57, http://dx.doi.org/10.1186/s13677-023-00435-5.

[93] C. Xu, J. Li, Y. Liu, Y. Ling, M. Wen, Accelerating split federated learning over wireless communication networks, IEEE Trans. Wirel. Commun. 23 (6) (2024) 5587–5599, http://dx.doi.org/10.1109/TWC.2023.3327372.

[94] S. Moon, Y. Lim, Split and federated learning with mobility in vehicular edge computing, in: 2023 IEEE/ACIS 21st International Conference on Software Engineering Research, Management and Applications, SERA, 2023, pp. 35–38, http://dx.doi.org/10.1109/SERA57763.2023.10197801.

[95] G. Zhu, Y. Deng, X. Chen, H. Zhang, Y. Fang, T.F. Wong, ESFL: Efficient split federated learning over resource-constrained heterogeneous wireless devices, IEEE Internet Things J. 11 (16) (2024) 27153–27166, http://dx.doi.org/10.1109/JIOT.2024.3397677.

[96] S. Zhang, W. Wu, P. Hu, S. Li, N. Zhang, Split federated learning: Speed up model training in resource-limited wireless networks, in: 2023 IEEE 43rd International Conference on Distributed Computing Systems, ICDCS, 2023, pp. 985–986, http://dx.doi.org/10.1109/ICDCS57875.2023.00096.

[97] 3GPP, 3rd Generation Partnership Project; Technical Specification Group Services and System Aspects; Study on Enablers for Network Automation for 5G (Release 17), Tech. Rep. TR 23.700-91 V17.0.0, 3GPP, 2022, Available: https://portal.3gpp.org/desktopmodules/Specifications/SpecificationDetails.aspx?specificationId=3589.

[98] 3GPP, 3rd Generation Partnership Project; Technical Specification Group Services and System Aspects; Architecture for Enabling Edge Applications (Release 17), Tech. Rep. TS 23.288 V17.4.0, 3GPP, 2022, Available: https://portal.3gpp.org/desktopmodules/Specifications/SpecificationDetails.aspx?specificationId=3579.

[99] 3GPP, 3rd Generation Partnership Project; Technical Specification Group Services and System Aspects; Study on Security Aspects of Enablers for Network Automation for 5G (Release 16), Tech. Rep. TR 33.818 V16.0.0, 3GPP, 2020, Available: https://portal.3gpp.org/desktopmodules/Specifications/SpecificationDetails.aspx?specificationId=3512.

[100] ITU-T, Architectural Framework for Machine Learning in Future Networks Including IMT-2020, Tech. Rep. Recommendation ITU-T Y.3172, International Telecommunication Union, 2019, Available: https://www.itu.int/rec/T-REC-Y.3172-201906-I.

[101] ITU-T, Framework for Evaluating Intelligence Levels of Future Networks Including IMT-2020, Tech. Rep. Recommendation ITU-T Y.3173, International Telecommunication Union, 2020, Available: https://www.itu.int/rec/T-REC-Y.3173-202008-I.

[102] IEEE, IEEE Guide for Architectural Framework and Application of Federated Machine Learning, Tech. Rep. IEEE Std 3652.1-2022, IEEE, 2022, http://dx.doi.org/10.1109/IEEESTD.2022.9944213.

[103] ETSI, Experiential Networked Intelligence (ENI); Terminology for Main Concepts in ENI, Tech. Rep. ETSI GR ENI 007 V1.1.1, European Telecommunications Standards Institute, 2021, Available: https://www.etsi.org/deliver/etsi_gr/ENI/001_099/007/01.01.01_60/gr_ENI007v010101p.pdf.

[104] ETSI, Multi-Access Edge Computing (MEC); MEC Support for AI/ML, Tech. Rep. ETSI GR MEC 017 V3.1.1, European Telecommunications Standards Institute, 2022, Available: https://www.etsi.org/deliver/etsi_gr/MEC/001_099/017/03.01.01_60/gr_MEC017v030101p.pdf.

[105] O-RAN Alliance, O-RAN Architecture Description, Tech. Rep. O-RAN.WG1.EADP-v02.00, O-RAN Alliance, 2023, Available: https://www.o-ran.org/specifications.

[106] Q. Wang, A. Clemm, C.J. Bernardos, et al., Challenges in the Integration of Artificial Intelligence into Network Management, Tech. Rep. draft-irtf-nmrg-ai-challenges-02, IETF, 2023, in preparation. Available: https://datatracker.ietf.org/doc/draft-irtf-nmrg-ai-challenges/.

[107] X. Liu, L. Xu, X. Wu, S. Zhang, L. Wang, Split-FL: An efficient online federated learning framework with constrained computation and streaming data, in: 2024 IEEE International Conference on Communications Workshops, ICC Workshops, 2024, pp. 661–666, http://dx.doi.org/10.1109/ICCWorkshops59551.2024.10615347.

[108] M. Przewięźlikowski, M. Osial, B. Zieliński, M. Śmieja, A deep cut into split federated self-supervised learning, in: A. Bifet, J. Davis, T. Krilavičius, M. Kull, E. Ntoutsi, I. Žliobaitė (Eds.), Machine Learning and Knowledge Discovery in Databases. Research Track, Springer Nature Switzerland, Cham, 2024, pp. 444–459.

[109] Z. Lin, G. Qu, W. Wei, X. Chen, K.K. Leung, Adaptsfl: Adaptive split federated learning in resource-constrained edge networks, 2024, arXiv preprint arXiv:2403.13101.

[110] D. Yan, M. Hu, Z. Xia, Y. Yang, J. Xia, X. Xie, M. Chen, Have your cake and eat it too: Toward efficient and accurate split federated learning, 2023, arXiv preprint arXiv:2311.13163.

[111] J. Shen, N. Cheng, X. Wang, F. Lyu, W. Xu, Z. Liu, K. Aldubaikhy, X. Shen, RingSFL: An adaptive split federated learning towards taming client heterogeneity, IEEE Trans. Mob. Comput. 23 (5) (2024) 5462–5478, http://dx.doi.org/10.1109/TMC.2023.3309633.

[112] J. Shin, J. Ahn, H. Kang, J. Kang, Fedsplitx: Federated split learning for computationally-constrained heterogeneous clients, 2023, arXiv preprint arXiv:2310.14579.

[113] L.U. Khan, M. Guizani, A. Al-Fuqaha, C.S. Hong, D. Niyato, Z. Han, A joint communication and learning framework for hierarchical split federated learning, IEEE Internet Things J. 11 (1) (2024) 268–282, http://dx.doi.org/10.1109/JIOT.2023.3315673.

[114] P. Joshi, C. Thapa, M. Hasanuzzaman, T. Scully, H. Afli, Federated split learning with only positive labels for resource-constrained IoT environment, 2023, arXiv preprint arXiv:2307.13266.

[115] S. Hossan, F. Mahmud, P. Roy, M.A. Razzaque, M.M. Rahman, Energy and latency-aware computation load distribution of hybrid split and federated learning on IoT devices, in: Proceedings of the 10th International Conference on Networking, Systems and Security, NSysS '23, Association for Computing Machinery, New York, NY, USA, 2023, pp. 61–68, http://dx.doi.org/10.1145/3629188.3629201.

[116] J. Karjee, P.S. Naik, N. Srinidhi, Split federated learning and reinforcement based codec switching in edge platform, in: 2023 IEEE International Conference on Consumer Electronics, ICCE, 2023, pp. 1–6, http://dx.doi.org/10.1109/ICCE56470.2023.10043412.

[117] T.A. Khoa, D.-V. Nguyen, M.-S. Dao, K. Zettsu, splitDyn: Federated split neural network for distributed edge AI applications, in: 2022 IEEE International Conference on Big Data, Big Data, 2022, pp. 6066–6073, http://dx.doi.org/10.1109/BigData55660.2022.10020803.

[118] Y. Wen, G. Zhang, K. Wang, K. Yang, Training latency minimization for model-splitting allowed federated edge learning, IEEE Trans. Netw. Sci. Eng. 12 (3) (2025) 2081–2092, http://dx.doi.org/10.1109/TNSE.2025.3544313.

[119] B. Yin, Z. Chen, M. Tao, Predictive GAN-powered multi-objective optimization for hybrid federated split learning, IEEE Trans. Commun. 71 (8) (2023) 4544–4560, http://dx.doi.org/10.1109/TCOMM.2023.3277878.

[120] H. Hafi, B. Brik, M. Bagaa, A. Ksentini, Impact of neural network depth on split federated learning performance in low-resource UAV networks, in: 2024 International Wireless Communications and Mobile Computing, IWCMC, 2024, pp. 1290–1295, http://dx.doi.org/10.1109/IWCMC61514.2024.10592346.

[121] L.U. Khan, M. Guizani, C.S. Hong, Resource optimized hierarchical split federated learning for wireless networks, in: Proceedings of Cyber-Physical Systems and Internet of Things Week 2023, in: CPS-IoT Week '23, Association for Computing Machinery, New York, NY, USA, 2023, pp. 254–259, http://dx.doi.org/10.1145/3576914.3590148.

[122] W. Ni, H. Ao, H. Tian, Y.C. Eldar, D. Niyato, FedSL: Federated split learning for collaborative healthcare analytics on resource-constrained wearable IoMT devices, IEEE Internet Things J. 11 (10) (2024) 18934–18935, http://dx.doi.org/10.1109/JIOT.2024.3370985.

[123] F. Almalik, N. Alkhunaizi, I. Almakky, K. Nandakumar, FeSViBS: Federated split learning of vision transformer with block sampling, 2023, http://dx.doi.org/10.48550/arXiv.2306.14638, arXiv preprint arXiv:2306.14638.

[124] H. Karnati, B.B. V, Federated and split learning for enhanced breast cancer diagnosis, in: 2023 IEEE Engineering Informatics, 2023, pp. 1–7, http://dx.doi.org/10.1109/IEEECONF58110.2023.10520444.

[125] Z.H. Kafshgari, I.V. Bajić, P. Saeedi, Smart split-federated learning over noisy channels for embryo image segmentation, in: ICASSP 2023 - 2023 IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP, 2023, pp. 1–5, http://dx.doi.org/10.1109/ICASSP49357.2023.10095067.

[126] Z.H. Kafshgari, C. Shiranthika, P. Saeedi, I.V. Bajić, Quality-adaptive split-federated learning for segmenting medical images with inaccurate annotations, in: 2023 IEEE 20th International Symposium on Biomedical Imaging, ISBI, 2023, pp. 1–5, http://dx.doi.org/10.1109/ISBI53787.2023.10230484.

[127] Z. Yang, Y. Chen, H. Huangfu, M. Ran, H. Wang, X. Li, Y. Zhang, Robust split federated learning for U-shaped medical image networks, 2022, arXiv:2212.06378. URL https://arxiv.org/abs/2212.06378.

[128] M. Wu, G. Cheng, D. Ye, J. Kang, R. Yu, Y. Wu, M. Pan, Federated split learning with data and label privacy preservation in vehicular networks, IEEE Trans. Veh. Technol. 73 (1) (2024) 1223–1238, http://dx.doi.org/10.1109/TVT.2023.3304176.

[129] X. Qiang, Z. Chang, C. Ye, T. Hamalainen, G. Min, Split federated learning empowered vehicular edge intelligence: Concept, adaptive design, and future directions, IEEE Wirel. Commun. (2025) 1–8, http://dx.doi.org/10.1109/MWC.009.2400219.

[130] D. Naseh, Distributed Machine Learning for 6G Intelligent Vehicular Communication Networks (Ph.D. thesis), alma, 2025, http://dx.doi.org/10.48676/unibo/amsdottorato/12512, URL https://amsdottorato.unibo.it/id/eprint/12512/.

[131] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A.N. Gomez, Ł. Kaiser, I. Polosukhin, Attention is all you need, in: Advances in Neural Information Processing Systems, NeurIPS, 2017, pp. 5998–6008.

[132] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, J. Uszkoreit, N. Houlsby, An image is worth 16x16 words: Transformers for image recognition at scale, in: International Conference on Learning Representations, ICLR, 2021.

[133] Y. Shi, Y. Zhang, Z. Huang, et al., Heterogeneous federated learning with splited language model, 2024, http://dx.doi.org/10.48550/arxiv.2403.16050, arXiv preprint arXiv:2403.16050.

[134] S. Wang, B.Z. Li, M. Khabsa, H. Fang, H. Ma, Linformer: Self-attention with linear complexity, 2020, arXiv preprint arXiv:2006.04768.

[135] V. Sanh, L. Debut, J. Chaumond, T. Wolf, DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter, 2019, arXiv preprint arXiv:1910.01108.

[136] J. Lu, et al., Split learning for vision-language models, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR, 2022.

[137] Y. Chen, et al., Federated multimodal learning: A survey, 2022, arXiv preprint arXiv:2205.11031.

[138] X. Wang, Y. Peng, L. Lu, Z. Lu, M. Bagheri, R.M. Summers, ChestX-Ray8: Hospital-scale chest X-Ray database and benchmarks on weakly-supervised classification and localization of common thorax diseases, in: 2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR, IEEE, 2017, pp. 3462–3471, http://dx.doi.org/10.1109/cvpr.2017.369.

[139] Alzheimer's Disease Neuroimaging Initiative (ADNI), ADNI data, 2025, Data retrieved from the Alzheimer's Disease Neuroimaging Initiative (ADNI) database. Accessed: 2025-09-26. URL https://adni.loni.usc.edu/data-samples/adni-data/.

[140] B.H. Menze, A. Jakab, S. Bauer, J. Kalpathy-Cramer, K. Farahani, J. Kirby, et al., The multimodal brain tumor image segmentation benchmark (BRATS), IEEE Trans. Med. Imaging 34 (10) (2015) 1993–2024, http://dx.doi.org/10.1109/TMI.2014.2377694.

[141] S. Bakas, M. Reyes, A. Jakab, S. Bauer, M. Rempfler, A. Crimi, et al., Identifying the best machine learning algorithms for brain tumor segmentation, progression assessment, and overall survival prediction in the BRATS challenge, 2018, arXiv:1811.02629.

[142] S. Bakas, H. Akbari, A. Sotiras, M. Bilello, M. Rozycki, J.S. Kirby, et al., Advancing The Cancer Genome Atlas glioma MRI collections with expert segmentation labels and radiomic features, Sci. Data 4 (2017) 170117, http://dx.doi.org/10.1038/sdata.2017.117.

[143] S. Bakas, H. Akbari, A. Sotiras, M. Bilello, J. Rozycki, J. Kirby, et al., Segmentation Labels and Radiomic Features for the Pre-Operative Scans of the TCGA-LGG Collection, The Cancer Imaging Archive, 2017, http://dx.doi.org/10.7937/K9/TCIA.2017.GJQ7R0EF.

[144] S. Bakas, H. Akbari, A. Sotiras, M. Bilello, M. Rozycki, J. Kirby, et al., Segmentation Labels and Radiomic Features for the Pre-Operative Scans of the TCGA-GBM Collection, The Cancer Imaging Archive, 2017, http://dx.doi.org/10.7937/K9/TCIA.2017.KLXWJJ1Q.

[145] A. Go, R. Bhayani, L. Huang, Twitter Sentiment Classification using Distant Supervision, Tech. Rep. CS224N, Stanford University, 2009, Dataset available at: https://www.kaggle.com/datasets/kazanova/sentiment140. URL https://cs.stanford.edu/people/alecmgo/papers/Twitter_Sentiment_Classification_using_Distant_Supervision.pdf.

[146] P.A. Lopez, M. Behrisch, L. Bieker-Walz, J. Erdmann, Y.-P. Flötteröd, R. Hilbrich, L. Lücken, J. Rummel, P. Wagner, E. Wiessner, Microscopic traffic simulation using SUMO, in: 2018 21st International Conference on Intelligent Transportation Systems, ITSC, 2018, pp. 2575–2582, http://dx.doi.org/10.1109/ITSC.2018.8569938.

[147] A. Dosovitskiy, G. Ros, F. Codevilla, A. Lopez, V. Koltun, CARLA: An open urban driving simulator, in: Proceedings of the 1st Annual Conference on Robot Learning, 2017, pp. 1–16.

[148] C. He, S. Li, J. So, X. Zeng, M. Zhang, H. Wang, X. Wang, P. Vepakomma, A. Singh, H. Qiu, X. Zhu, J. Wang, L. Shen, P. Zhao, Y. Kang, Y. Liu, R. Raskar, Q. Yang, M. Annavaram, S. Avestimehr, FedML: A research library and benchmark for federated machine learning, 2020, arXiv:2007.13518. URL https://arxiv.org/abs/2007.13518.

[149] D.J. Beutel, T. Topal, A. Mathur, X. Qiu, J. Fernandez-Marques, Y. Gao, L. Sani, K.H. Li, T. Parcollet, P.P.B. de Gusmão, N.D. Lane, Flower: A friendly federated learning research framework, 2022, arXiv:2007.14390. URL https://arxiv.org/abs/2007.14390.

[150] T. Ryffel, A. Trask, M. Dahl, B. Wagner, J. Mancuso, D. Rueckert, J. Passerat-Palmbach, A generic framework for privacy preserving deep learning, 2018, arXiv:1811.04017. URL https://arxiv.org/abs/1811.04017.

[151] K. Bonawitz, H. Eichner, W. Grieskamp, D. Huba, A. Ingerman, V. Ivanov, C. Kiddon, J. Konečný, S. Mazzocchi, H.B. McMahan, T.V. Overveldt, D. Petrou, D. Ramage, J. Roselander, Towards federated learning at scale: System design, 2019, arXiv:1902.01046. URL https://arxiv.org/abs/1902.01046.

[152] G.F. Riley, T.R. Henderson, The ns-3 network simulator, in: Modeling and Tools for Network Simulation, Springer, 2010, pp. 15–34.

[153] A. Varga, OMNeT++, in: Modeling and Tools for Network Simulation, Springer, 2010, pp. 35–59.