Review | Open Access

Intelligent Networking and Computing

Chen *et al. Intell. Netw. Comput.* **2026**, *1*, 3 **DOI:**10.20517/inect.2026.04

Resource-efficient federated large language models: challenges and mechanisms

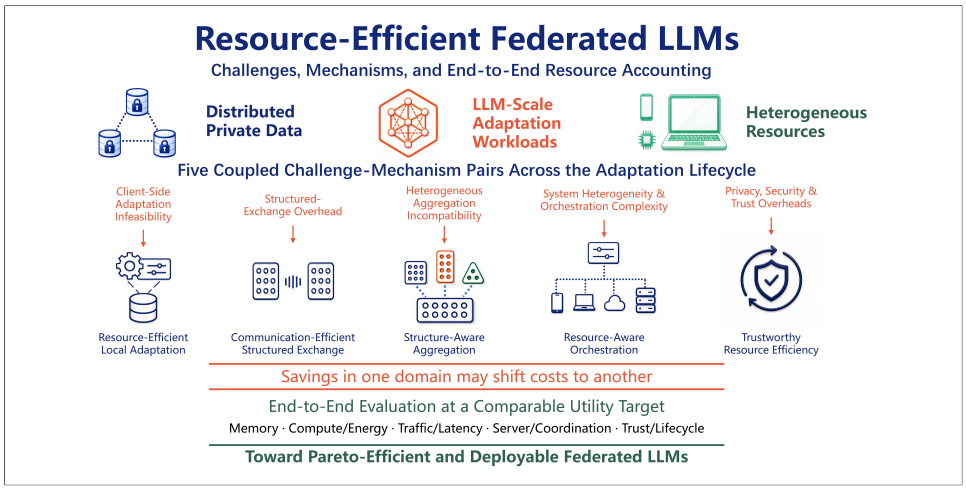
Xiaojing Chen[1], Chenchen Wang[1], Bingcong Li[2], Xin Wang[3]

Keywords:
Federated large language models, parameter-efficient fine-tuning, resource efficiency, heterogeneous aggregation, edge-cloud collaboration, trustworthy federated learning

Citation: Chen, X.; Wang, C.; Li, B.; Wang, X. Resource-efficient federated large language models: challenges and mechanisms. *Intell. Netw. Comput.* **2026**, *1*, 3. https://dx.doi.org/10.20517/inect.2026.04

Received: 17 Jul 2026
First Decision: 6 Aug 2026
Revised: 20 Aug 2026
Accepted: 21 Aug 2026
Published: 23 Sep 2026

Academic Editor: Hongliang Zhang
Copy Editor: Shu-Yuan Duan
Production Editor: Shu-Yuan Duan



Abstract

Adapting large language models (LLMs) over distributed private data is increasingly important for domain specialization, personalization, and alignment. Federated learning (FL) provides a natural paradigm for this goal, but federated LLMs are not simply conventional FL systems with larger models. The pretrained backbone, Transformer execution, activation memory, structured parameter-efficient fine-tuning (PEFT) updates, heterogeneous clients, and protection mechanisms create coupled resource costs across the full adaptation workflow. This review examines federated LLM adaptation from a resource-efficiency perspective. We first identify five recurring challenges: local adaptation feasibility, communication overhead, aggregation compatibility, system orchestration, and trustworthy operation. We then organize existing mechanisms into a taxonomy covering local adaptation and training, structured update and knowledge exchange, structure-aware aggregation, resource-aware orchestration, and trustworthy resource efficiency. The review shows that resource savings should be evaluated across the complete workflow, because reductions in trainable parameters, transmitted objects, or client memory may be accompanied by additional communication, server processing, coordination, or protection costs. Finally, we summarize evaluation requirements and research priorities for deployable

[1]Key Laboratory of Specialty Fiber Optics and Optical Access Networks, Shanghai University, Shanghai 200444, China.
[2]Dept of CS, ETH Zurich, Zurich 8092, Switzerland.
[3]College of Future Information Technology, Fudan University, Shanghai 200433, China.

Correspondence to: Prof. Xin Wang, College of Future Information Technology, Fudan University, Shanghai 200433, China. E-mail: xwang11@fudan.edu.cn

federated LLMs, emphasizing end-to-end accounting, comparable utility targets, transparent resource evidence, realistic heterogeneity, and lifecycle-aware workloads.

INTRODUCTION

Large language models (LLMs) support a wide range of knowledge-intensive applications, but their practical value increasingly depends on post-training adaptation[1-3] for domain specialization, personalization, and alignment. Such adaptation often relies on private, domain-specific, or user-generated data that remain distributed because of regulation, ownership, or confidentiality, or because centralization would impose excessive data-transfer overhead or violate application latency requirements. Federated learning (FL) addresses this access problem by keeping data local while clients optimize and exchange model-related information for collaborative updating. It preserves local control over distributed data and draws on computing resources across organizations and edge devices, making it a natural route to private and domain-specific LLM adaptation[1,4,5]. More broadly, the evolution of edge intelligence toward generative AI places increasing demands on distributed computation, communication, memory, and service orchestration[6], further motivating resource-aware LLM adaptation across edge and cloud infrastructure.

In this context, federated LLMs refer to FL systems that collaboratively adapt a pretrained LLM, or its parameter-efficient adaptation modules, across distributed clients without sharing raw data. The adaptation process may involve full-model fine-tuning, low-rank adaptation (LoRA) or adapter optimization, prompt tuning, intermediate-representation exchange, or personalized component learning. This review focuses on federated post-training adaptation, including fine-tuning, alignment, and personalization, rather than federated pretraining from scratch.

Federated LLMs retain the basic collaborative-learning principle of conventional FL, but substantially change its resource profile. Typical FL systems often assume a common model architecture and directly aggregatable parameter updates, whereas federated LLM adaptation starts from a large pretrained backbone and may exchange LoRA factors, adapters, prompts, intermediate representations, or other structured adaptation objects. Even with parameter-efficient fine-tuning (PEFT), clients may still incur substantial backbone-residency, Transformer-execution, and activation-memory costs. Moreover, clients can differ in adaptation rank, trainable layers, precision, or model-access mode, making communication and aggregation more heterogeneous and potentially requiring alignment, reconstruction, distillation, or personalized fusion.

These characteristics amplify conventional FL challenges, including communication overhead, partial participation, stragglers, non-independent and non-identically distributed (non-IID) data, and privacy risks[4,5,7,8], while creating stronger coupling among computation, communication, aggregation, and system resources. In particular, reducing trainable or transmitted parameters does not necessarily reduce end-to-end cost: Client-side savings may reappear as repeated activation/gradient exchange in split or offloaded adaptation[9], while privacy, verification, and lifecycle-management mechanisms introduce additional computation, communication, or storage overhead[10-12]. More broadly, large-scale AI deployment in communication networks also couples model capabilities with communication, computing, resource allocation, and scalability constraints[13]. Resource efficiency in federated LLMs should therefore be evaluated across the complete adaptation workflow rather than inferred from trainable parameter count or per-round communication payload alone.

To clarify the distinction from existing surveys on federated LLMs[14-17], Table 1 compares representative reviews in terms of their primary emphasis, resource aspects that receive less attention, and the additional perspective provided by this review. Existing surveys provide complementary perspectives on Federated LLM

Table 1. Comparison of representative surveys and the distinctive scope of this review

Review	Primary emphasis	Coverage not emphasized	Added by this review
Hu *et al.*[14]	Federated LLM solutions, challenges, and future directions	Systematic resource accounting across different stages of federated adaptation	Workflow-level resource taxonomy and cross-stage cost analysis
Wen *et al.*[15]	Federated PEFT mechanisms for LLMs	Resource costs beyond parameter-efficient adaptation, including aggregation, orchestration, and trust	End-to-end analysis beyond trainable-parameter and update efficiency
Ren *et al.*[16]	Federated foundation models, architectures, and open challenges	LLM-specific post-training resource accounting and quantitative cross-stage evaluation	Focused analysis of resource-efficient federated LLM adaptation with quantitative resource evidence
Yan *et al.*[17]	Comparison of FedLLM, KD-FedLLM, and Split-FedLLM frameworks	Aggregation heterogeneity, system orchestration, trust/lifecycle costs, and full-workflow resource accounting	Unified analysis of local execution, communication, aggregation, orchestration, and trust costs
Adhikari *et al.*[18]	Robustness, privacy, trustworthiness, and edge deployment	Resource interactions among adaptation, aggregation, orchestration, and protection mechanisms	Trust and protection incorporated into end-to-end resource accounting
Piccialli *et al.*[21]	Federated and edge learning, efficient LLM training and deployment	Federated post-training-specific aggregation, orchestration, and cross-stage resource shifting	Resource analysis specialized to federated LLM post-training adaptation
This review	Resource-efficient federated LLM post-training adaptation	-	Workflow-level resource taxonomy, cross-stage cost-shift analysis, representative quantitative resource evidence, and utility-normalized Pareto benchmarking

LLM: Large language model; PEFT: parameter-efficient fine-tuning.

frameworks, PEFT, federated foundation models, privacy and robustness, and edge learning[18-20], whereas this review takes end-to-end resource efficiency in federated LLM post-training adaptation as its organizing perspective. Specifically, it connects local execution, communication, aggregation, orchestration, and trust/lifecycle costs, while emphasizing cross-stage cost analysis, quantitative resource evidence, and utility-normalized Pareto benchmarking.

Related surveys on efficient FL examine communication reduction, client selection, resource allocation, and system optimization, whereas surveys on efficient and edge LLMs mainly discuss compression, memory reduction, inference acceleration, and hardware-aware deployment[7,21]. Low-rank and other parameter-efficient adaptation methods reduce trainable state[2,3] and complement these broader deployment-oriented efficiency techniques. Together, these studies provide important foundations, but they do not fully explain how LLM-scale adaptation costs interact with FL-specific communication, aggregation, orchestration, and trust overheads across the complete federated adaptation workflow.

Accordingly, we make three contributions:

• We develop a workflow-level resource analysis of federated LLM adaptation, identifying five key challenges: local adaptation infeasibility, communication overhead, aggregation incompatibility, orchestration complexity, and privacy, security, and lifecycle-trust overheads.

• We propose a matching resource-efficiency taxonomy covering resource-efficient local adaptation and training, communication-efficient exchange of structured updates and knowledge, structure-aware aggregation and heterogeneous adaptation, resource-aware system orchestration, and trustworthy resource efficiency.

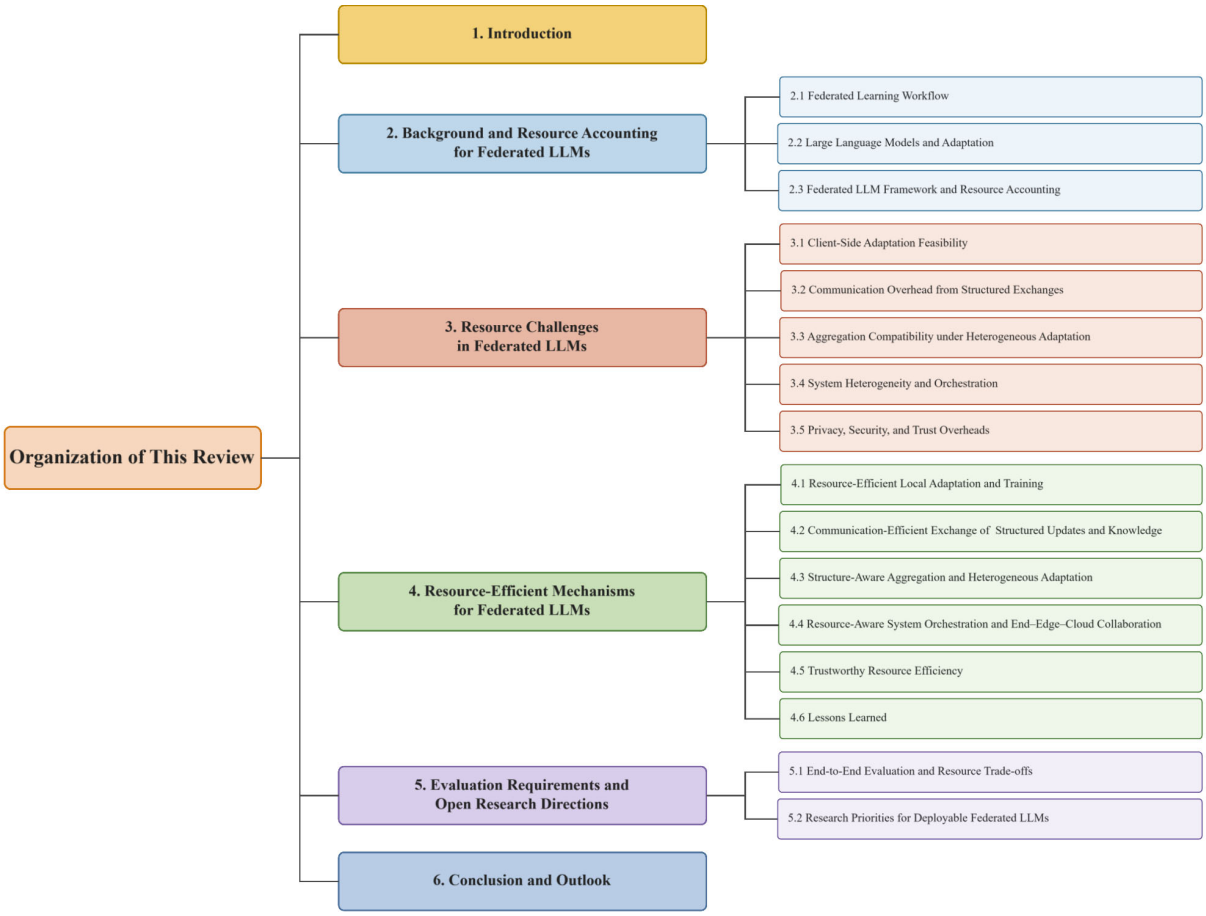


Figure 1. Organization and logical progression of this review. LLM: Large language model.

• We summarize evaluation requirements and research priorities for deployable federated LLMs, emphasizing end-to-end resource accounting, comparable utility targets, evidence transparency, realistic heterogeneity, and lifecycle-aware workloads.

Figure 1 summarizes the organization of this review. Section **BACKGROUND AND RESOURCE ACCOUNTING FOR FEDERATED LLMS** presents the background and resource-accounting framework for federated LLM adaptation. Section **RESOURCE CHALLENGES IN FEDERATED LLMS** analyzes the main resource challenges, and Section **RESOURCE-EFFICIENT MECHANISMS FOR FEDERATED LLMS** reviews the corresponding resource-efficient mechanisms. Section **EVALUATION REQUIREMENTS AND OPEN RESEARCH DIRECTIONS** discusses evaluation requirements and research priorities for deployable federated LLMs. Section **CONCLUSION AND OUTLOOK** concludes the review.

BACKGROUND AND RESOURCE ACCOUNTING FOR FEDERATED LLMS

Federated learning workflow

FL coordinates model optimization across distributed data holders without requiring raw data to be centralized. In a typical server-client workflow, the server initializes and distributes a global model, selected clients perform local optimization on private data, and the server aggregates the returned model-related information to update the global model[4,5]. Federated Averaging (FedAvg) is the canonical instance, where client updates are commonly averaged according to local data volume[4]. This repeated distribution-local update-aggregation cycle makes FL suitable for privacy-preserving collaborative learning, but also introduces resource costs from communication, synchronization, client heterogeneity, and server-side aggregation.

The workflow can be described along three dimensions that are especially relevant to federated LLM adaptation:

• Data organization: Horizontal FL considers different samples in a shared feature space; vertical FL considers overlapping entities with different feature sets; and federated transfer learning addresses limited overlap in both samples and features.

• Coordination architecture: Server-based FL relies on a central aggregator; hierarchical FL introduces intermediate edge servers; and decentralized FL uses peer-to-peer communication and local consensus.

• Synchronization mode: Synchronous FL aggregates updates from a round-specific client set, asynchronous FL incorporates updates as they arrive, and bounded-asynchronous FL limits staleness while retaining partial flexibility[5,8].

These dimensions determine where data remain, how model-related information moves, and when updates are fused. They therefore provide the basic workflow assumptions for analyzing communication, computation, aggregation, and synchronization costs in federated LLM systems.

Large language models and adaptation

LLMs are foundation models trained on large-scale text or multimodal corpora to acquire transferable capabilities for language-centered tasks. Most contemporary LLMs use the Transformer architecture, whose self-attention captures long-range token dependencies and permits parallel processing of token positions within each training layer, although autoregressive generation remains sequential. Through pretraining, typically with next-token prediction, LLMs learn general linguistic, semantic, and task-relevant patterns that support generation, reasoning, coding, and domain-specific text processing[22,23].

Post-training adapts this pretrained foundation to specific domains, users, or preferences through supervised fine-tuning, instruction tuning, domain adaptation, or preference alignment. Full-model fine-tuning updates most or all parameters, whereas PEFT trains only low-rank matrices, adapters, prompts, prefixes, or other small components while freezing most of the backbone[2,3,24]. It aims to preserve adaptation utility while reducing trainable state, optimizer storage, and, in federated settings, update size.

However, PEFT does not remove the main execution burden of LLM adaptation. Clients may still need to host the pretrained backbone, perform forward and backward propagation through Transformer layers, store activations and intermediate states, and process long input sequences. Long-context adaptation further increases attention computation and activation memory[22,23]. As a result, LLM adaptation is not only a model-update problem, but also an execution problem shaped by memory, computation, bandwidth, latency, and energy. This distinction is central to resource accounting in federated LLMs, where fewer trainable parameters do not necessarily imply lower end-to-end adaptation cost.

Federated LLM framework and resource accounting

Although FL can support language-model pretraining, this review focuses on the more common and practically relevant setting of federated post-training adaptation. In this setting, clients collaboratively adapt a pretrained LLM, or an adaptation interface built around it, over distributed private data rather than training a language model from random initialization. Depending on the system design, clients may update the full model, optimize PEFT modules[1,14,17], tune prompts, maintain personalized components, or exchange split-model representations[9], while raw data remain local[25,26].

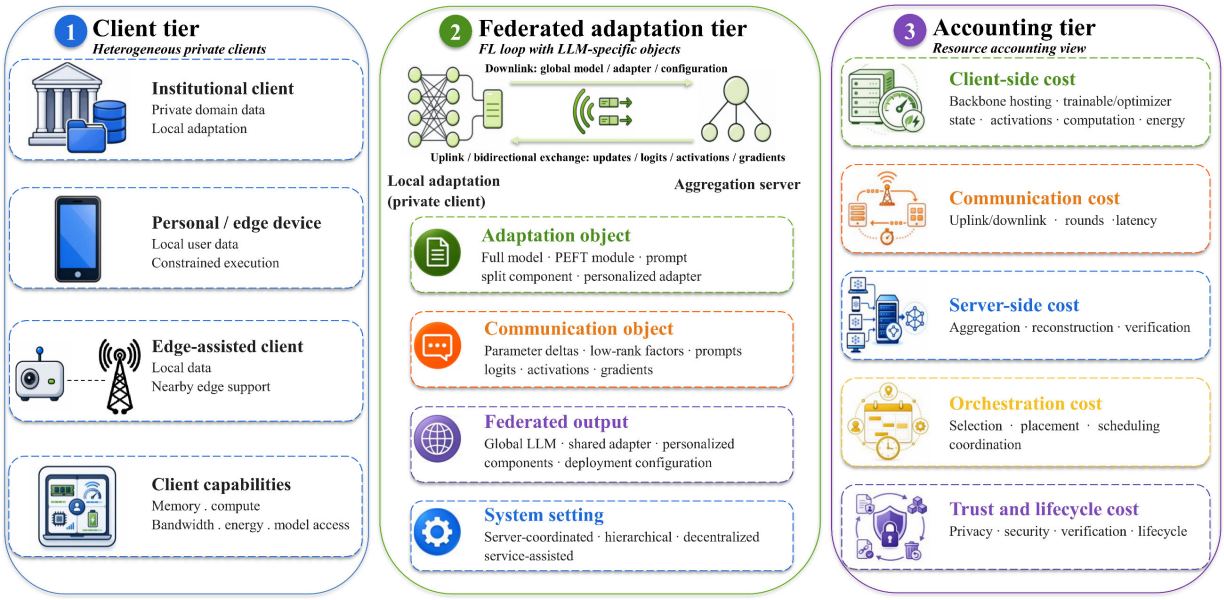


Figure 2. Federated LLM framework and resource-accounting view. Heterogeneous clients adapt LLM-specific objects through communication and aggregation, while total cost is accounted across client-side execution, network exchange, server-side processing, orchestration, and trust/lifecycle operations at a comparable utility target. LLM: Large language model; PEFT: parameter-efficient fine-tuning.

This setting changes three key objects in the federated loop:

• Adaptation object: the full model, a PEFT module, a prompt representation, a split-model component, or a personalized adapter.

• Communication object: compressed parameter deltas, low-rank factors, prompts, logits, activations, gradients, or other task-related signals.

• Federated output: a global LLM, a shared adaptation module, personalized components, or a deployment configuration.

These objects may vary across clients. In server-coordinated, hierarchical, and emerging service-assisted federated LLM systems[27], clients can differ in adaptation configuration, model access, objectives, and available resources. Such differences create two coupled constraints: local feasibility, which determines whether a client can produce an update, and aggregation compatibility, which determines whether heterogeneous updates can be fused into a useful shared or personalized model.

These choices complicate resource accounting in federated LLMs. Unlike conventional FL, model size, computation, and communication can become decoupled: A small PEFT module may still require executing a large pretrained backbone[1,24,28], while compact updates may introduce frequent synchronization, activation transfer, reconstruction, or alignment costs[9]. Trust mechanisms, including privacy protection, verification, robustness, provenance, and unlearning, further add computation, communication, storage, or utility costs[10-12].

Resource efficiency should therefore be assessed across the complete adaptation workflow at a comparable utility target, covering client execution, bidirectional communication, server-side processing, orchestration, and trust-related costs. As summarized in Figure 2, trainable parameter count or per-round payload alone cannot represent the end-to-end cost of federated LLM adaptation.

Table 2. Comparison of federated LLM adaptation with adjacent model-collaboration paradigms

Paradigm	Data/model placement	Collaboration object	Primary resource burden	Appropriate setting/relation to FLLM
Centralized LLM adaptation[2-3,23]	Training data and computation are centralized	Raw data are transferred to a central trainer; checkpoints are later deployed	Centralized data transfer, governance, and compute cost	Preferable when data movement and centralized governance are acceptable; it does not preserve local data control during adaptation
Conventional FL[4,5]	Raw data remain local; clients commonly share a task model	Parameters or gradients in a compatible structure	Communication, local computation, stragglers, and statistical/system heterogeneity	Suitable for manageable, structurally compatible task models; it does not capture the backbone and structured-adaptation costs of LLMs
Split/split-federated learning[9,29,30]	Model execution is partitioned between client and server	Intermediate activations and gradients	Reduced device-side execution, but repeated bidirectional traffic and server-side load	Complementary to FLLM when edge assistance is available; it changes, rather than removes, the resource bottleneck
Large-small/edge-cloud model collaboration[31]	Small models run locally; large models/services reside at edge/cloud	Inputs, intermediate information, knowledge, or service outputs	Local computation versus communication, service latency, and edge/cloud resource use	Primarily deployment-time collaboration; it can complement FLLM after adaptation but does not itself perform collaborative training on local data
Decentralized peer collaboration[32,33]	Raw data remain local; no conventional aggregation server is required	Peer updates, PEFT modules, or knowledge	Network-wide peer exchange, topology/peer selection, consensus, and trust cost	A coordination variant of FLLM when central aggregation is undesirable; it trades server dependence for distributed coordination cost
Federated LLM adaptation[1,14-17,25,26]	Raw data remain local; a pretrained LLM or PEFT interface is adapted across participants	Full deltas, LoRA/adapters, prompts, logits, activations, gradients, or other structured signals	Backbone execution, structured communication, heterogeneous aggregation, orchestration, and trust/lifecycle cost	Distinct when local data control and collaborative shared/personalized adaptation of a common backbone are both required; this is the scope of the review.

FLLM denotes federated LLM. LLM: Large language model; PEFT: parameter-efficient fine-tuning.

Table 2 summarizes the main differences between federated LLM adaptation and related model collaboration paradigms. Federated LLM adaptation occupies a specific design point among these collaboration paradigms. It combines local data control with collaborative adaptation of large pretrained models. Its resource challenge extends beyond conventional FL communication and local computation because LLM-scale execution, structured and potentially heterogeneous adaptation objects, and their aggregation and orchestration must be considered jointly. This distinction motivates the workflow-level resource analysis developed in Sections **RESOURCE CHALLENGES IN FEDERATED LLMS** and **RESOURCE-EFFICIENT MECHANISMS FOR FEDERATED LLMS**.

The review is organized around five resource questions that follow the federated adaptation workflow: Can a client *produce* a useful update within its memory, computation, time, and energy budget? Can the required information be *exchanged* efficiently? Can heterogeneous client contributions be *aggregated* into a meaningful shared or personalized model? Can participation, workload, synchronization, computation placement, and communication resources be *orchestrated* efficiently? Finally, can *privacy, security, and lifecycle trust* be maintained without excessive additional resource cost? These questions cover the principal resource-bearing stages of federated post-training adaptation and explain the one-to-one organization of the challenges in Section **RESOURCE CHALLENGES IN FEDERATED LLMS** and the corresponding mechanisms in Section **RESOURCE CHALLENGES IN FEDERATED LLMS**.

Table 3. Representative edge devices and resource heterogeneity in federated language-model adaptation studies

Device class	Compute and memory profile	Power and network budget	Observed heterogeneity and implication for federated adaptation
Jetson AGX Orin 64 GB[34]	2048-core Ampere GPU; 12-core Arm CPU; 64 GB unified LPDDR5	15-60 W configurable power; 1 Gbit/s link in the edge-LLM study	High-end edge client; supports PEFT of FLAN-T5 models up to 3B parameters, but memory bandwidth and model-update communication remain bottlenecks
Jetson Orin Nano 8 GB[35]	1024-core Ampere GPU; 6-core Arm CPU; 8 GB LPDDR5	7-15 W power modes; evaluated at 15 W and 1 MB/s in FedARA	Mid-range accelerator; lower latency than CPU-only clients, but memory, thermal, and energy budgets favor adaptive rank, precision, and participation
Jetson TX2 8 GB[36]	256-core Pascal GPU; 8 GB LPDDR4	7.5-15 W device class; 1 MB/s default experimental link	0.88 s per BERT batch; communication dominates on the stronger client, making compact adapters and fewer rounds important
Jetson Nano 4 GB[36]	128-core Maxwell GPU; 4 GB LPDDR4	5/10 W power modes; 1 MB/s default experimental link	1.89 s per BERT batch; tighter memory and compute than TX2 increase sensitivity to adapter depth and width
Raspberry Pi 4B[36]	Quad-core Cortex-A72 CPU; 1-8 GB LPDDR4; no CUDA-class GPU	15 W recommended supply; 1 MB/s default experimental link	18.27 s per BERT batch; CPU-bound local training motivates shallow adapters, caching, and selective participation
Raspberry Pi 5[35]	Quad-core Cortex-A76 CPU; 4/8 GB LPDDR4X variants; exact evaluated RAM not reported	27 W recommended supply; 1 MB/s experimental link in FedARA	1.00/2.01 s per DistilBERT/BERT batch; faster than Pi 4B but still markedly slower than Orin devices

GPU: Graphics processing unit; LLM: Large language model; RAM: random access memory; CPU: central processing unit; BERT: Bidirectional Encoder Representations from Transformers.

RESOURCE CHALLENGES IN FEDERATED LLMS

Federated LLMs inherit classical FL challenges, including communication overhead, partial participation, stragglers, non-IID data, and privacy risks, while intensifying them through large-scale model execution and adaptation. Structured updates, heterogeneous modules, multi-tier orchestration, and LLM-specific trust risks add further complexity. Resource efficiency therefore requires adaptation, communication, aggregation, orchestration, and protection to remain jointly feasible under practical resource constraints.

Client-side adaptation feasibility

The reported link rates in Table 3 are experimental configurations adopted by the corresponding studies and should not be interpreted as constant-rate assumptions for practical wireless deployments.

A basic assumption in FL is that each selected client can complete its assigned local update. This assumption becomes fragile in federated LLMs. Representative edge platforms differ in accelerator support, memory capacity, power envelope, network conditions, and measured local-training latency, as summarized in Table 3[34-36] The reported link rates in Table 3 are experimental configurations adopted by the corresponding studies and should not be interpreted as constant-rate assumptions for practical wireless deployments. Even when adaptation starts from a pretrained model, clients may still need to host the backbone, store activations and intermediate states, propagate gradients through Transformer layers, and maintain optimizer states. These requirements can make some local adaptation configurations infeasible, rather than merely slower, on resource-constrained clients[1,37-39]. Local feasibility is also affected by the data used for adaptation. Redundant, noisy, or low-value instructions consume expensive Transformer updates without proportional utility, whereas data-scarce settings may require additional augmentation, privacy-preserving adaptation, or sample-selection mechanisms[40-42].

The resource evidence in existing federated language-model studies spans substantially different model scales. Results obtained with DistilBERT, BERT, and BART are useful for studying device heterogeneity and resource-control mechanisms, but they should not be interpreted as direct feasibility evidence for multi-billion-parameter generative LLMs. The latter introduce a much larger backbone-residency and execution

burden, in addition to activation and optimizer-state memory during training. For example, AssyLLM[43] reports that full fine-tuning of LLaMA-7B with batch size 16 requires more than 40 GB of memory, compared with the 4-16 GB memory available to many of the edge devices considered in its evaluation. Its block-assembly design reduces memory consumption by up to 92%, illustrating both the severity of the memory gap and the need for LLM-specific execution mechanisms.

This creates a gap between parameter efficiency and system feasibility. PEFT methods such as LoRA and adapters reduce the number of trainable parameters[2-3,24,44-46], but producing even a lightweight update may still require substantial memory, computation, and energy because the frozen backbone must be executed. Local feasibility is best assessed through the full adaptation process, where uploaded update size is considered together with model residency, memory footprint, training time, energy use, and data utility.

Communication overhead from structured exchanges

Communication has long been a core bottleneck in FL, but federated LLMs change both the size and the form of exchanged information. Since full-model transmission is often impractical, clients may instead communicate PEFT modules[47,48], dense or compressed parameter deltas[49,50], prompts, logits, activations, gradients, or other intermediate representations[9,51]. These objects are usually much smaller than a complete LLM, but they may be exchanged repeatedly across local batches, communication rounds, or split-model interactions, creating substantial uplink, downlink, and synchronization costs.

Thus, smaller messages do not necessarily imply lower end-to-end communication cost. Compression and low-rank transmission may reduce each payload but introduce extra processing, metadata, or reconstruction costs, and lower-fidelity updates may require more rounds to reach the same utility.

In practical wireless deployments, network conditions may vary substantially over time. Channel fading, interference, link errors and retransmissions, MAC-layer contention, bandwidth fluctuations, and uplink/downlink asymmetry can reduce effective goodput and increase communication-latency variability. This variability directly affects federated synchronization: A temporarily weak client link may become a communication straggler in synchronous FL, whereas asynchronous or partial aggregation can reduce waiting at the cost of increased update staleness. The effect can be particularly pronounced for split-federated LLMs, where activations and gradients may be exchanged repeatedly during local adaptation. Communication efficiency should therefore account not only for nominal link rates, but also for link variability, reliability, tail latency, and their effects on synchronization.

Aggregation compatibility under heterogeneous adaptation

FedAvg assumes that client updates share a common parameter structure and can be directly averaged[4,5]. This assumption weakens in federated LLMs because resource constraints often push clients toward heterogeneous adaptation configurations. A resource-rich client may train a higher-rank LoRA module or more Transformer layers, whereas a constrained client may use lower ranks, fewer trainable layers[52-54], lower-bit precision[37], or a prompt-based interface[55]. These choices improve local feasibility, but they also change the structure and functional meaning of the updates received by the server.

Aggregation therefore becomes a compatibility problem, not merely an averaging operation. Client updates may differ in structure, precision, or training objective. While mild structural differences can be handled through simple alignment[52,53], stronger heterogeneity often requires reconstruction[54], distillation[56], or personalized fusion[55]. Uniform configurations simplify aggregation but may exclude weaker clients, whereas heterogeneous configurations improve participation at the cost of additional fusion complexity. The key challenge is therefore to preserve the functional or semantic content of local adaptations while keeping

aggregation efficient.

System heterogeneity and orchestration

Aggregation compatibility concerns the structure of updates, whereas system heterogeneity concerns the clients and infrastructure that produce them. In federated LLMs, system heterogeneity arises jointly from computing and communication capacities. Computational heterogeneity includes memory and accelerator availability, training speed, numerical-precision support, and energy or thermal constraints, while communication heterogeneity includes bandwidth, uplink/downlink asymmetry, time-varying link quality, and intermittent connectivity. Existing studies adapt quantization, LoRA configuration, or local workload to heterogeneous client capabilities[35,37], while hierarchical and split-federated designs coordinate workload, synchronization, and communication resources across heterogeneous devices and links[29,57,58]. A uniform workload may exclude resource-constrained clients or create stragglers[59,60], whereas edge assistance[61] and asynchronous participation[8] can shift resource burdens toward communication, edge servers, or coordination overhead[57,62]. These effects are tightly coupled: offloading can reduce client computation while increasing communication demand, whereas communication-limited clients may require lighter adaptation workloads or less frequent participation.

System orchestration must therefore assign feasible and useful roles to heterogeneous participants. This involves deciding which clients participate, what adaptation configuration they use, where computation is placed, when updates are synchronized, and how communication and computing resources are allocated. These decisions are coupled: Lowering local computation through edge assistance may increase repeated transfers of activations and gradients[61,62]; relaxing synchronization may reduce waiting time but introduce staleness[8,57]; and favoring capable clients may improve efficiency but reduce data coverage or fairness[55,59]. Effective orchestration should therefore balance local feasibility, communication and computation cost, system latency, and data representativeness.

Privacy, security, and trust overheads

Keeping raw data local provides an important basis for privacy-preserving LLM adaptation, but the exchanged information still requires protection. Privacy mainly concerns leakage from shared updates or intermediate representations, which may reveal sensitive training content or user-specific patterns[63-65]. Security concerns the robustness and integrity of adaptation, where malicious, biased, or backdoored updates may distort the shared model or propagate harmful behavior[66-68]. Trust further concerns whether the adaptation process is verifiable, accountable, and auditable across the model lifecycle.

These protections introduce resource overheads. Privacy mechanisms may reduce utility or increase communication and cryptographic costs[10,18,65]; security mechanisms such as robust aggregation and attack detection require additional computation[66]; and trust mechanisms such as verifiable aggregation, provenance, and unlearning introduce storage, auditing, rollback, or retraining costs[11,12]. Privacy, security, and trust mechanisms should therefore be co-designed with resource optimization, because they jointly shape the attainable balance among utility, efficiency, and trustworthiness.

RESOURCE-EFFICIENT MECHANISMS FOR FEDERATED LLMS

Following the challenges in Section RESOURCE CHALLENGES IN FEDERATED LLMS, Figure 3 organizes resource-efficient mechanisms according to the main stages of the federated adaptation workflow: local adaptation and training, structured exchange, heterogeneous aggregation, system orchestration, and trust-constrained operation. This section examines how these mechanisms reduce resource costs and where their trade-offs arise.

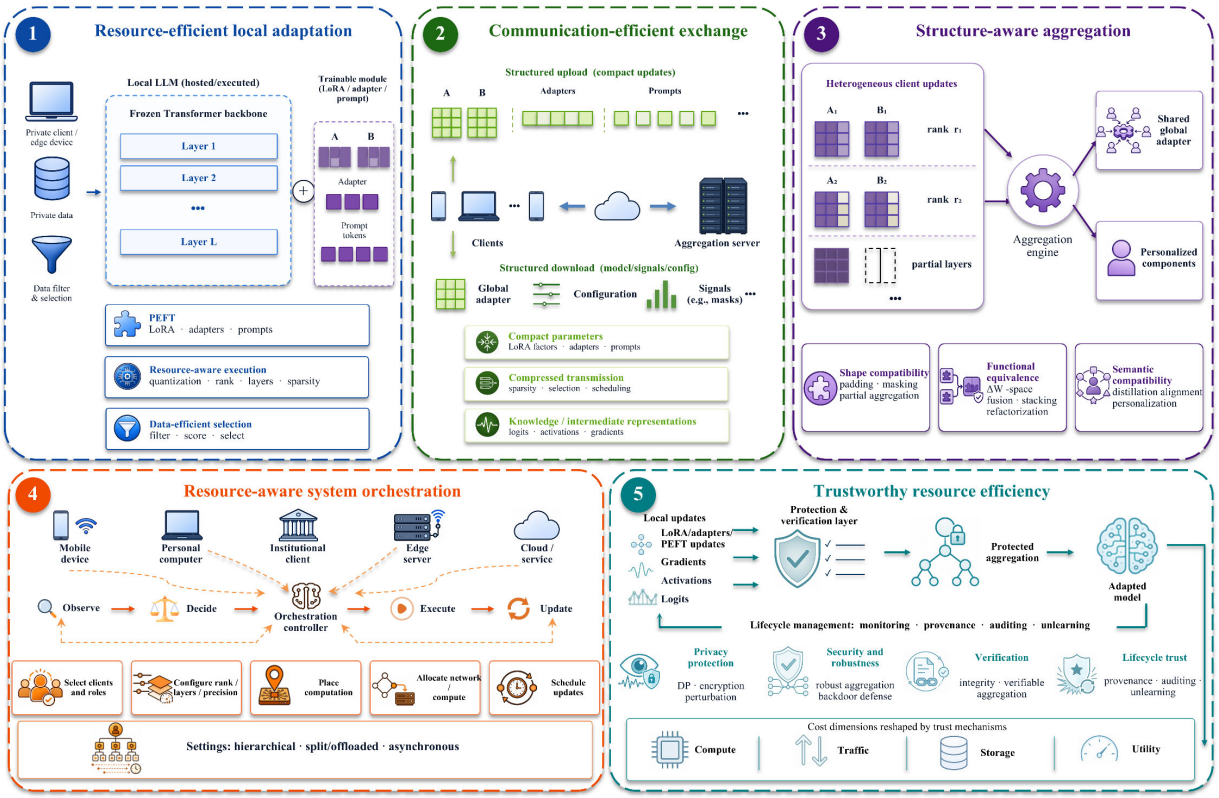


Figure 3. Taxonomy of resource-efficient mechanisms for federated LLM adaptation. LLM: Large language model; PEFT: parameter-efficient fine-tuning.

Resource-efficient local adaptation and training

Local adaptation is the first step at which resource feasibility is tested. A selected client must be able to produce a useful update within its memory, computation, time, and energy budget. Existing methods improve local feasibility by reducing trainable state, lowering execution cost, or selecting the data that justify expensive Transformer updates.

Parameter-efficient adaptation. PEFT adapts a pretrained LLM by freezing most backbone parameters and updating only a small set of trainable components. Many federated LLM systems therefore replace full fine-tuning with PEFT methods such as LoRA, adapters, prefix tuning, and prompt tuning[1,44-46]. For LLaMA-7B, FederatedScope-LLM[1] reports per-round messages of approximately 12,852 MB for full-model exchange, 21.40 MB for LoRA, and 0.17 MB for prompt tuning, while model-only memory remains about 13.4 GB under the study's accounting. PEFT therefore provides its most direct savings in trainable and transmitted state, while backbone hosting and execution remain important parts of local resource accounting.

Resource-aware local execution. Beyond the choice of trainable modules, resource-aware execution controls the cost of producing each local update. Existing methods adjust numerical precision[37], LoRA rank, trainable layers, or active modules[39] according to client capability and module sensitivity[35,52,69,70]. FedARA[35], for example, dynamically allocates and prunes low-rank modules and reports an average 2.40-fold improvement in communication efficiency, up to 48.90% shorter total training time, and 46.95% lower energy consumption on Orin Nano. It also reports a 31.67% reduction in average peak GPU memory usage per round relative to FedLoRA. These results are obtained mainly with DistilBERT, BERT, and BART and therefore provide PLM-scale evidence for adaptive execution rather than direct feasibility evidence for multi-billion-parameter LLMs.

Recent generative-LLM studies have addressed the larger memory constraint more directly. AssyLLM[43] assembles and adapts selected pretrained blocks to reduce the memory required for federated LLaMA-7B fine-tuning, while FedBiOT[71] enables federated adaptation of LLaMA-2 without requiring clients to hold the complete model. Together with the LLaMA-7B evidence reported above for FederatedScope-LLM[1], these studies show that reducing trainable parameters alone is insufficient; backbone residency, activation memory, and Transformer execution remain first-order constraints for edge-side generative-LLM adaptation.

Data-efficient local training. Local cost also depends on how many examples undergo expensive Transformer updates. Data-efficient methods reduce this cost by selecting higher-value training samples or filtering low-quality instructions. FedDQC[40] filters low-quality instructions through a scoring stage that consumes roughly 1% of training time, whereas FedHDS[41] retains less than 1.5% of the available samples and reports 6.66- to 48.8-fold wall-clock speedups across different settings. These results extend local efficiency from choosing trainable parameters to choosing training data.

In summary, PEFT reduces what is updated, resource-aware execution reduces the cost of each update, and data-efficient training reduces the number of updates performed on low-value samples. End-to-end deployability depends on combining these mechanisms with communication, aggregation, orchestration, and trust-aware design across the full federated adaptation workflow.

Communication-efficient exchange of structured updates and knowledge

Communication efficiency in federated LLMs depends on what is exchanged, how it is encoded, and how often information is exchanged. Existing methods improve communication efficiency by exchanging compact adaptation parameters, selectively transmitting structured updates, or replacing parameter exchange with knowledge or intermediate representations.

Compact structured parameter exchange. The most direct approach is to exchange trainable adaptation parameters instead of full model weights. LoRA modules, adapters, and prompts preserve the conventional server-client training loop while substantially reducing the communicated object[1,44]. More compact parameterizations further redesign the adaptation object itself. FedTT and FedTT+[47] represent tensor-train-based federated adaptation, with FedTT+ further freezing selected factors. These approaches are most effective when clients share a pretrained backbone and compatible adaptation structures. Their end-to-end benefit should be assessed together with model initialization, downlink broadcasts, metadata, and convergence rounds.

Selective and compressed transmission. These methods reduce the amount of structured information sent in each exchange while keeping the adaptation representation largely unchanged. Typical strategies include low-precision encoding, sparsity, component selection, and partial transmission. EcoLoRA[48], for example, retains the LoRA structure while rotating transmitted segments, exploiting matrix-specific sparsity, and compactly encoding indices. Other methods transmit selected low-rank components or active update portions[69,72]. These methods mainly optimize the scheduling and encoding of structured updates rather than redesigning the adaptation representation. Their gains depend on the balance among payload reduction, update fidelity, and the number of rounds required to reach a target utility.

Knowledge and intermediate-representation exchange. Instead of transmitting model parameters, these methods use task-level signals or internal representations as the communication object. Black-box language and vision-language settings may exchange discrete prompts or compact task signals[73-75], while split systems transmit activations and gradients across model partitions[9,51]. These designs can support heterogeneous

access modes and reduce parameter transmission, but they may introduce repeated queries, proxy-data dependence, distillation overhead, or frequent bidirectional communication. TITANIC[9] provides a useful example of this cost shift. Under its reported representation size and exchange pattern, cumulative activation and gradient traffic can exceed a compact LoRA update after only a few batches.

A simple break-even condition helps clarify when split adaptation is preferable to direct PEFT exchange. For one local adaptation round, let D_P denote the total bidirectional PEFT-update traffic, D_S the bidirectional activation/gradient traffic per split interaction, and K the number of such interactions. With effective network throughput B_{eff}, the first-order completion times of PEFT and split adaptation are

$$T_{\mathrm{P}} = T_{\mathrm{local}} + \frac{D_{\mathrm{P}}}{B_{\mathrm{eff}}}, \quad T_{\mathrm{S}} = T_{\mathrm{c}} + T_{\mathrm{e}} + \frac{KD_{\mathrm{S}}}{B_{\mathrm{eff}}}, \tag{1}$$

where T_{local} is the local PEFT execution time, and T_c and T_e are the client-side and edge-side execution times under splitting, respectively. Defining the relative split-computation ratio as

$$\rho = \frac{T_{\mathrm{c}} + T_{\mathrm{e}}}{T_{\mathrm{local}}}, \tag{2}$$

when $\rho < 1$ and $KD_S > D_P$, the break-even throughput is

$$B^{*} = \frac{KD_{\mathrm{S}} - D_{\mathrm{P}}}{(1-\rho)T_{\mathrm{local}}}, \tag{3}$$

and split adaptation reduces latency when $B_{eff} > B^*$. Thus, greater computation offloading lowers the required bandwidth, whereas larger or more frequent activation/gradient exchanges raise it. The threshold varies with the split point, sequence length, batch size, compression, hardware capability, and network conditions, motivating joint model-partitioning and communication-resource optimization[29,58].

Overall, communication efficiency should therefore be evaluated at the level of the complete exchange process and its computation-communication trade-off. In addition to the size of each transmitted object, assessment should include total bidirectional traffic, the number and pattern of interactions, encoding and reconstruction overhead, and the time or rounds needed to reach a comparable utility target.

Structure-aware aggregation and heterogeneous adaptation

Aggregation becomes a resource problem when clients adopt different adaptation configurations. FedAvg is straightforward when updates share the same structure, but uniform configurations may overburden weak clients or underuse capable ones. Heterogeneous adaptation improves local feasibility by allowing clients to use different ranks, trainable layers, precisions, or adaptation modules, while introducing additional aggregation complexity. Existing methods address this problem at three levels: shape compatibility, functional equivalence, and semantic compatibility.

Shape compatibility. Shape-compatible methods handle heterogeneous updates through layer allocation[52], zero-padding and sparsity-weighted aggregation[53], padding- or knowledge-distillation-based alignment[76], masking, or partial aggregation[72]. HETLORA[53], for example, combines local rank self-pruning with server-side zero-padding, sparsity-weighted aggregation, and rank-specific truncation. These operations make updates structurally aggregatable, while functional consistency still depends on how the adapted parameters affect the model.

Functional equivalence. For LoRA-based adaptation, the update produced by client i is not represented by a single matrix, but by the product of two low-rank factors, i.e., $\Delta W_i = B_i A_i$. Therefore, the desired global update

is the weighted average of the actual updates, $\Delta W_{\text{global}} = \sum_{i=1}^{K} p_i B_i A_i$, where p_i denotes the normalized aggregation weight. A naïve parameter-wise extension of FedAvg would instead average the factors separately, but this does not generally recover the desired global update[54]. The product of the averaged factors introduces cross-client combinations that were not optimized by any client[46,54]. This creates a functional mismatch: The aggregated LoRA factors may have the right shape but fail to represent the intended average model update. Methods such as FLoRA[54], FedOTAB[77], and FedPipe[70] address this issue by preserving or reconstructing LoRA updates in a functionally meaningful space rather than relying on naïve factor averaging: FLoRA[54] stacks factors to preserve the weighted update, FedOTAB[77] alternates the optimized factor to reduce communication, and FedPipe[70] aggregates in the ΔW space before refactorization.

Semantic compatibility and personalization. When clients differ in backbone, adapter type, task, or output space, parameter-level alignment may be insufficient. Related federated foundation-model methods use distillation, representation alignment, or common-subspace mapping to connect heterogeneous client knowledge[76,78,79]. Structural-bias-aware partial-layer tuning addresses a related layer-coverage heterogeneity problem[56]. Personalized designs, including dual adapters[80], clustered aggregation[79], and Rest-of-World LoRA[81], further separate globally transferable knowledge from client-specific behavior[82,83]. These methods improve flexibility but may introduce proxy data, server inference, or multi-module management costs.

Structure-aware aggregation therefore determines whether heterogeneous local adaptations can be made shape-compatible, functionally meaningful, and semantically useful. Effective aggregation should not only align update dimensions, but also preserve the model behavior contributed by different clients and support either a coherent global model or personalized local components.

Resource-aware system orchestration and end-edge-cloud collaboration

System orchestration determines how federated LLM adaptation is mapped onto heterogeneous clients, networks, and edge/cloud infrastructure. It controls which clients participate, what adaptation workload they receive, where computation is executed, when updates are incorporated, and how communication and computing resources are allocated.

Participation and role assignment. Client participation can be guided by data value, resource availability, energy status, and expected completion time. Resource-aware selection should assign training roles to clients that can provide useful updates with manageable delay, energy use, or carbon emissions, while maintaining sufficient data coverage and participation fairness[59]. FedSustain[59] shows how energy-aware participation can affect both learning and system cost. Its renewable-energy-aware configuration achieved 73.8% accuracy with estimated 13 kWh energy use and 9 kg CO_2 emissions, compared with 71.2%, 21 kWh, and 14 kg CO_2 under random selection. This result shows that sustainability-aware participation can jointly improve learning utility and resource efficiency when client selection is aligned with energy and deployment conditions.

Workload and synchronization control. Orchestration determines how much work each selected client performs and how this workload is coordinated with available communication resources. LoRA rank, trainable layers, numerical precision, sparsity[37,52,69], local training time, and participation[62] can be adapted jointly with communication and computing resources[60,84-86]. HierFedLoRA[57] provides a representative example by combining near-IID grouping, group-specific aggregation frequency, and fine-tuning depth across 80 Jetson devices, optimizing time-to-accuracy rather than single-round efficiency. Dynamic network conditions further couple these decisions with synchronization. Channel fading, interference, and bandwidth variation can change client communication latency over time, causing communication stragglers in

synchronous FL or increased update staleness in asynchronous FL. Accordingly, client participation, bandwidth allocation, adaptation workload, and synchronization frequency can be adjusted according to both device and channel states. Wang *et al.*[58] consider device scheduling and bandwidth allocation under time-varying wireless conditions; Pang *et al.*[84] jointly optimize client-specific pruning and bandwidth allocation for low-latency federated LLM fine-tuning, and Zhang *et al.*[87] adapt the communicated knowledge according to available channel resources. More broadly, hierarchical and asynchronous coordination can reduce cloud traffic or waiting time, but their benefits depend on how network variability, staleness, grouping overhead, and controller complexity are managed.

Computation placement and end-edge-cloud collaboration. Split and offloaded training move part of the adaptation workload from devices to edge or cloud servers. By placing selected Transformer blocks or backward computation outside the client, these methods can reduce client-side memory and computation, while shifting part of the cost to activation exchange, edge/cloud load, and coordination overhead[61,88-90].

Related wireless split-learning and cloud-edge FL studies provide broader evidence that computation placement and representation design can reshape communication and computing costs across heterogeneous networks[30,91]. In wireless split-federated LLM systems, these trade-offs become directly coupled with LLM adaptation and communication-resource control. Existing studies jointly optimize split points, LoRA ranks, bandwidth, transmit power, subchannel allocation, or multiple-access schemes[61,88,89], and some designs further incorporate anti-jamming, sensing assistance, or privacy constraints[92-94].

Decentralized and multi-participant collaboration. Federated LLM adaptation can extend beyond conventional server-coordinated aggregation to direct collaboration among multiple clients or model instances. Dec-LoRA[32] coordinates decentralized LoRA updates through topology-based peer interaction, while Balija *et al.*[33] consider asynchronous peer-to-peer model exchange. Such decentralized designs can reduce dependence on a central aggregator and alleviate communication congestion at the central node, but shift resource costs toward repeated peer exchange, heterogeneous Device-to-Device (D2D) links, topology management, synchronization, and distributed coordination. Compact PEFT exchange can reduce peer-transfer payloads, while topology-aware peer selection and clustered communication can limit unnecessary transfers. Asynchronous coordination can further reduce waiting for slow or intermittently connected peers, although update staleness must be controlled.

Beyond client-level decentralized federation, broader multi-participant training can involve collaboration among multiple model agents. MAPoRL[95], for example, studies multi-agent post-training in which multiple LLMs generate, exchange, and discuss responses during joint optimization. Such collaboration introduces additional interaction rounds, token processing, peer communication, and coordination overhead.

Resource-aware orchestration therefore goes beyond client selection by jointly designing participation, workload configuration, synchronization, computation placement, and resource allocation across the end-edge-cloud continuum.

Trustworthy resource efficiency

Privacy[96,97], security, and lifecycle trust constraints reshape the resource-efficiency problem in federated LLM adaptation. Protection mechanisms[98,99] improve the reliability of distributed adaptation[100-102], but they also consume the same computation, communication, storage, and utility budgets required for training and aggregation[103].

Privacy-aware resource efficiency. Privacy-aware mechanisms reduce leakage from shared updates and intermediate representations. Differential privacy protects updates through clipping and perturbation[10], while encrypted or homomorphic aggregation strengthens confidentiality at the cost of additional communication and cryptographic processing[65,98]. Selective perturbation can reduce utility loss by protecting sensitive components[96,97,99], and reconstruction risk depends on what information the protocol exposes, such as gradients, adapters, activations, or split representations[63,64]. The resource challenge is to provide sufficient privacy protection while controlling communication, computation, and convergence overhead.

Security-aware resource efficiency. Security-aware mechanisms address malicious or biased contributions that may distort the shared model, introduce backdoors, or weaken robustness[67,68,104]. Robust aggregation, poisoning detection, and adversarially robust prompt-tuning can improve resilience, while adding server-side computation, screening latency, or extra optimization stages[66,105,106]. Related security-aware resource-allocation studies in broader cloud-edge-terminal systems also show that security requirements can be coupled with communication and computing-resource allocation[107]. This provides a complementary system-level perspective on security-resource co-design for federated LLMs. In the Shakespeare next-token-prediction experiment, GPT-2 Medium was evaluated with 100 clients and 10 selected per round; aggregation required approximately 3.9 s for a perturbation-based defense, 27.03 s for Multi-Krum, and 63.20 s for FLAME. The sentence-trigger Neurotoxin variant could nevertheless evade both screening methods[66]. Security-aware resource efficiency requires robustness improvements to be evaluated alongside aggregation latency, server-side computation, and learning utility.

Trust- and lifecycle-aware resource efficiency. Trust- and lifecycle-aware mechanisms focus on whether the federated adaptation process is verifiable, accountable, and auditable over time. Verifiable aggregation strengthens confidence that structured updates are processed correctly, but introduces additional computation and verification overhead. FLAGuard[11], for example, improves the efficiency of LoRA verification by more than two orders of magnitude relative to prior schemes, with approximately 8% overhead over unverified FLoRA. Provenance tracking and federated unlearning extend trust beyond the training round by requiring historical states, audit trails, rollback, retraining, or deletion verification[12]. The resource challenge is to integrate verification, provenance, and unlearning with compact federated adaptation, so that accountability can be supported without excessive storage or retraining cost.

Privacy, security, and lifecycle trust protect different aspects of federated LLM adaptation, but they draw on the same resource budgets as training and communication. Resource-efficient design should therefore optimize protection strength, learning utility, and system cost jointly.

Lessons learned

Across these mechanism classes, local adaptation reduces the cost of producing client updates; structured exchange reduces communication payload; structure-aware aggregation preserves the meaning of heterogeneous updates; orchestration matches workloads to available resources; and privacy, security, and trust mechanisms define the protection cost required for reliable adaptation. Resource savings should therefore be assessed over the full workflow. A reduction in trainable parameters, message size, or client memory is valuable only when it also improves end-to-end efficiency at a comparable utility target. Table 4 summarizes representative methods according to their mechanism, primary efficiency contribution, and main limitation.

EVALUATION REQUIREMENTS AND OPEN RESEARCH DIRECTIONS

Current evidence on resource-efficient federated LLMs remains fragmented across model scales, hardware platforms, network assumptions, and reporting conventions. Progress requires end-to-end evaluation at comparable utility targets, together with research that addresses adaptive system control, realistic

Table 4. Comparison of representative mechanisms for resource-efficient federated LLM adaptation

Mechanism class	Representative method	Mechanism	Reported method-specific efficiency evidence	Remaining limitation
Local adaptation	FederatedScope-LLM[1]	LoRA/prompt update exchange	Serialized-adapter message size for one server-client communication: 21.40 MB with LoRA and 0.17 MB with prompt tuning (LLaMA-7B)	Approximately 13.4 GB model-only memory; backbone execution remains
	FedARA[35]	Dynamic rank allocation and rank-based module pruning	2.40× average communication-efficiency improvement; up to 48.90% shorter total training time relative to FedLoRA; on Orin Nano (15 W), 46.95% lower energy consumption relative to FedLoRA over 100 rounds with 10 clients per round	Rank-based module pruning reduces average peak GPU memory usage per round by 31.67% relative to FedLoRA; based on DistilBERT, BERT, and BART
	FedDQC/FedHDS[40-41]	Quality and representativeness selection	FedDQC scoring consumes approximately 1% of training time; FedHDS variants use less than 1.5% of the data with up to 48.8× speedup	Coverage and robustness depend on the quality-scoring and subset-selection criteria
Structured exchange	FedTT/FedTT+[47]	Tensor-train factorization and adaptive factor freezing	Approximately 10× lower communication overhead for FedTT and 30× for FedTT+ in the reported LLaMA2-13B cross-silo setting	Compatible backbone and adapter structures are required; evidence remains specific to tensor-train parameterization
	EcoLoRA[48]	Rotating sparse segments and indices	Up to 79% less communication time and 65% less total training time under the reported 1/5 Mbps uplink/downlink setting	Potential staleness and fidelity-round trade-off; limited real-device evidence
	TITANIC[9]	Activation/gradient partitioning	Lower client residency and execution burden	Frequent bidirectional traffic; batch- and placement-sensitive cost
Structure-aware aggregation	HETLORA[53]	Rank self-pruning, zero-padding, and sparsity-weighted aggregation	Faster convergence with reduced computation and communication	Shape compatibility does not ensure functional equivalence
	FLoRA[54]	Function-preserving factor stacking	Preserves the weighted ΔW update	Global rank grows with participating ranks
	FedPipe/FedOTAB[70,77]	ΔW-space refactorization or alternating one-factor optimization/transmission	Function-preserving aggregation; half-LoRA transmission with FedOTAB	Additional server reconstruction or optimization
System orchestration	FedSustain[59]	Utility- and renewable-aware scheduling	Up to 38% lower energy use and 46% lower CO_2 emissions	Deployment-, grid-, and renewable-dependent gains
	HierFedLoRA[57]	Adaptive grouping, depth, and frequency	At least 2.1× faster fine-tuning and 1.6%-4.2% higher final accuracy	Grouping, staleness, and controller overhead
	Split-federated LLMs[61,88,89]	Split-point, rank, and/or radio-resource co-design	For FedsLLM, approximately 47.63% lower training delay on average than the BA strategy (which optimizes neither η nor bandwidth) in the reported MATLAB simulation	Repeated transfers of activations and gradients, server load, synchronization, and idealized channels
Trustworthy resource efficiency	DP and privacy-aware adaptation[10,96-97]	Calibrated perturbation, selective protection, and privacy-aware alignment	Formal DP guarantees or targeted leakage mitigation	Utility loss, extra rounds, or narrower formal coverage
	FLAGuard/FedHE[11,65]	LoRA verification or CKKS encryption	More than 100× faster verification with FLAGuard; encrypted aggregation with FedHE	Cryptographic overhead; FedHE is limited to compact BERT
	Federated TrustChain[12]	Provenance and unlearning	Auditable training and unlearning workflows	Ledger, rollback, and retraining costs; formal deletion guarantees remain open

BERT: Bidirectional Encoder Representations from Transformers; CKKS: Cheon-Kim-Kim-Song homomorphic encryption scheme; CO_2: carbon dioxide; DP: differential privacy; GPU: graphics processing unit; ΔW: model-weight update; LLM: large language model.

heterogeneity, and emerging LLM workloads.

End-to-end evaluation and resource trade-offs

Resource-efficient federated LLM methods should be evaluated by the total cost required to reach a comparable utility target. A complete accounting should include backbone distribution, client-side memory and computation[1], bidirectional traffic, server-side processing, orchestration and synchronization overhead[57,59], and privacy/security/trust costs[11,65]. Studies should report time, energy, and communication cost at comparable utility levels, together with task performance, personalization quality, tail-client performance, fairness, privacy, and robustness.

Transparent reporting is also needed for fair comparison. Key information includes the backbone model, adaptation object, client participation pattern, heterogeneity setting[35,37,52], hardware platform, network assumptions, peak memory, transferred bytes, wall-clock time, and energy measurement or estimation method[57,70]. Privacy and security studies should further specify the threat model, server-visible information, and protected object. Resource claims should be labeled as directly measured, modeled or estimated, proxy-based, or unevaluated, while formal guarantees should be reported separately from empirical efficiency results.

Wireless evaluations should additionally report the adopted channel model or network trace, uplink and downlink variability, reliability and retransmission assumptions, synchronization protocol, and mean and tail communication latency. For asynchronous or partial aggregation, update staleness and acceptance or timeout policies should also be specified. These factors are necessary to distinguish performance under nominal link configurations from resource efficiency under realistic time-varying communication conditions.

Utility-normalized Pareto benchmarking. Motivated by multi-objective resource evaluation in FL[108], we propose a utility-normalized Pareto framework. Let U^* denote a task-specific target utility and T_i^* the wall-clock time required by method i to first reach this target. For each method that reaches U^* within the evaluation budget, we define its resource vector at the target utility as

$$R_i\left(U^*\right)=\left(T_i^*, E_{i,\text{ client}}, E_{i,\text{ server}}, C_{i,\text{ up}}, C_{\text{i, down}}, M_{\text{i, peak}}\right), \tag{4}$$

where $E_{i,client}$ and $E_{i,server}$ are the accumulated client- and server/edge-side energy, respectively; $C_{i,up}$ and $C_{i,down}$ are the cumulative uplink and downlink traffic, respectively; and $M_{i,peak}$ is the peak memory, all measured up to T_i^*. At the same U^*, method A Pareto-dominates method B if it is no worse in every reported resource dimension and strictly better in at least one. Methods that do not reach the target should instead report their terminal utility and budget-end resource vector. Fair comparisons should use the same task, backbone, data partition, client-participation setting, hardware/network scenario, and resource-accounting boundary. This framework avoids labeling a method resource-efficient when savings in one dimension are mainly obtained by shifting cost to another.

Several trade-offs recur across the literature. Lower ranks, lower bit widths, or smaller selected training subsets reduce local cost but may affect utility or convergence. Heterogeneous configurations improve participation flexibility but complicate functionally correct aggregation. Privacy, security, and lifecycle-trust mechanisms improve protection while consuming bandwidth, computation, storage, or utility. Favoring fast or well-provisioned clients can improve average efficiency, but may reduce data coverage and participation fairness[55,59]. Methods should therefore be compared through utility-normalized and Pareto-aware evaluation rather than a single favorable metric.

Research priorities for deployable federated LLMs

Adaptive cross-layer control. Federated LLM systems require joint control across model, communication, system, and trust layers. Future methods should adapt participation[60,85], PEFT configuration, exchange strategy, aggregation rule[35,37,69], computation placement[60,85], and protection level[94] according to changing memory, bandwidth, energy, data value, and risk conditions. Such cross-layer control is important because savings in one stage may otherwise reappear as additional rounds, server reconstruction, communication traffic, or protection overhead.

Realistic-scale validation and joint heterogeneity. Current evidence remains concentrated on BERT-family models, small client populations, homogeneous backbones, and simulated network settings. Future validation should extend to larger generative LLMs, heterogeneous devices[35], dynamic client availability, variable bandwidth[61,88,89], asynchronous participation, and explicit client- and server-side energy accounting. Aggregation also remains challenging when clients differ simultaneously in backbone, LoRA rank, layer coverage, adapter type, precision, or model-access mode[53-54,56,78]. Addressing such joint heterogeneity is essential for moving from controlled experiments to deployable federated LLM systems.

Lifecycle-aware and emerging post-training settings. Future federated LLM systems will increasingly face diverse post-training workloads and lifecycle constraints. Multimodal and agentic workloads add visual representations[109-111], persistent memory, planning, and repeated interaction. In multi-model or multi-agent deployment, additional collaboration costs arise from agent participation, repeated model interaction, token processing, dialogue-history management, and service latency. Wang *et al.*[112], for example, study agent-level collaboration strategies and their accuracy-token-cost trade-offs, while large-small model collaboration[31] illustrates cooperation between models with different capabilities across edge resources. Personalization[81], provenance and unlearning[12], and renewable-energy-aware operation[59,113] further extend resource costs beyond a single adaptation round. Evaluation should therefore consider not only federated adaptation but also the downstream collaboration and operational costs over the lifetime of the adapted model.

CONCLUSION AND OUTLOOK

Federated LLM adaptation changes the resource profile of FL by altering what is trained, what is exchanged, how updates are aggregated, where computation is placed, and which protection mechanisms are required. This review shows that resource efficiency cannot be inferred from trainable parameter count or per-round payload alone. PEFT, structured exchange, heterogeneous aggregation, system orchestration, and privacy/security/trust mechanisms reduce different parts of the federated adaptation cost, but their gains may also shift costs to additional communication rounds, repeated transfer of intermediate activations and gradients, server-side processing, coordination, or protection.

Current evidence remains uneven across model scales, client populations, hardware platforms, network settings, and reporting conventions. Future research should therefore emphasize utility-normalized end-to-end accounting, function-preserving aggregation under joint model and resource heterogeneity, adaptive cross-layer control, and realistic evaluation of energy, privacy, security, and lifecycle costs. As federated LLMs move toward larger generative models, multimodal and agentic workloads, and continuously personalized services, resource efficiency should be assessed over the full operational lifetime of the system rather than within a single training round.

DECLARATIONS

Authors' contributions

Made substantial contributions to the conception and design of the review, literature search and screening, analysis and synthesis of the literature, and manuscript drafting: Chen, X.; Wang, C.; Li, B.
Provided supervision, critical revision of the manuscript, and administrative, technical, and material support: Wang, X.

Availability of data and materials

Not applicable.

AI and AI-assisted tools statement

During the preparation of this manuscript, the AI tool ChatGPT (version GPT-5.4, released 2026-03-05) was used for language editing and to assist in generating illustrative icons for Figures 1-3 based on textual descriptions provided by the authors. In the Graphical Abstract, the AI-assisted icons include those representing distributed private data, LLM-scale adaptation workloads, heterogeneous resources, local adaptation, structured exchange, heterogeneous aggregation, system orchestration, and trustworthy resource efficiency. The scientific concepts, taxonomy, text, logical relationships, and overall layout were developed by the authors, who subsequently selected, edited, and arranged the illustrative icons. The tool did not influence the study design, data collection, analysis, interpretation, or the scientific content of the work. All authors take full responsibility for the accuracy, integrity, and final content of the manuscript.

Financial support and sponsorship

This work was supported by the National Key R&D Program of China (No. 2022YFB2902303) and Shanghai Municipal Science and Technology Commission Foundation (No. 25DP1500300).

Conflicts of interest

All authors declared that there are no conflicts of interest.

Ethical approval and consent to participate

Not applicable.

Consent for publication

Not applicable.

REFERENCES

1. Kuang, W.; Qian, B.; Li, Z.; et al. FederatedScope-LLM: A comprehensive package for fine-tuning large language models in federated learning. KDD '24: The 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining; Barcelona Spain. New York, NY, USA: ACM; 2024. pp. 5260-71. DOI
2. Hu, E. J.; Shen, Y.; Wallis, P.; et al. LoRA: Low-rank adaptation of large language models. The International Conference on Learning Representations; 2022 April 25-29; https://openreview.net/forum?id=nZeVKeeFYf9 (accessed 2026-09-14).
3. Lialin, V.; Deshpande, V.; Yao, X.; Rumshisky, A. Scaling down to scale up: a guide to parameter-efficient fine-tuning. arXiv 2023, arXiv:2303.15647. Available online: https://arxiv.org/abs/2303.15647.
4. McMahan, B.; Moore, E.; Ramage, D.; Hampson, S.; Agüera y Arcas, B. Communication-efficient learning of deep networks from decentralized data. In Proceedings of the 20th International Conference on Artificial Intelligence and Statistics; 2017 April 20-22; Ft. Lauderdale, FL USA. ACM ICPS; 2017. pp 1273-82. https://proceedings.mlr.press/v54/mcmahan17a.html (accessed 2026-09-14).
5. Kairouz, P.; Mcmahan, H. B. Advances and open problems in federated learning. *Found. Trends. Mach. Learn.* **2021**, *14*, 1-210. DOI
6. Zhang, X.; Xie, G.; Huang, Y.; et al. Edge intelligence in the generative artificial intelligence era. *IEEE. Wireless. Commun.* **2025**, *32*, 60-8. DOI
7. Lim, W. Y. B.; Luong, N. C.; Hoang, D. T.; et al. Federated learning in mobile edge networks: a comprehensive survey. *IEEE. Commun. Surv. Tutorials.* **2020**, *22*, 2031-63. DOI
8. Nguyen, J.; Malik, K.; Zhan, H.; et al. Federated learning with buffered asynchronous aggregation. 25th International Conference on Artificial Intelligence and Statistics; 2022 Mar 28-30; Valencia, Spain. 2022, pp 3581-607. https://proceedings.mlr.press/v151/nguyen22b.html (accessed 2026-09-14).
9. Su, N.; Hu, C.; Li, B.; Li, B. Titanic: towards production federated learning with large language models. IEEE INFOCOM 2024 - IEEE Conference on Computer Communications; 2024 May 20-23; Vancouver, BC, Canada. IEEE; 2024. pp. 611-20. DOI
10. Liu, X.; Zhu, R.; Zha, D.; et al. Differentially private low-rank adaptation of large language model using federated learning. *ACM. Trans. Manage. Inf. Syst.* **2025**, *16*, 1-24. DOI

11. Zhang, T.; Yu, H.; Chen, Y.; Wang, S.; Yang, Z. FLAGuard: efficient verifiable federated LoRA of large language models. *IEEE. Trans. Mobile. Comput.* **2026**, *25*, 7182-95. DOI
12. Zuo, X.; Wang, M.; Zhu, T.; et al. Federated TrustChain: blockchain-enhanced LLM training and unlearning. *IEEE. Trans. Dependable. and. Secure. Comput.* **2026**, *23*, 6457-73. DOI
13. Shahid; A. ; Kliks, A.; Al-Tahmeesschi, A.; et al. Large-scale AI in telecom: charting the roadmap for innovation, scalability, and enhanced digital experiences. arXiv 2025, arXiv:2503.04184. Available online: https://doi.org/10.48550/arXiv.2503.04184.
14. Hu, J.; Wang, D.; Wang, Z.; et al. Federated large language model: solutions, challenges and future directions. *IEEE. Wireless. Commun.* **2025**, *32*, 82-9. DOI
15. Wen, Q.; Zhang, X.; Xiang, N.; Chen, J.; Wang, X.; Zhang, J. A survey on federated parameter-efficient fine-tuning for large language models. 2025 11th International Conference on Big Data and Information Analytics (BigDIA); 2025 Nov 8-11; Nha Trang, Vietnam. IEEE; 2025. pp. 637-42. DOI
16. Ren, C.; Yu, H.; Peng, H.; et al. Advances and open challenges in federated foundation models. *IEEE. Commun. Surv. Tutorials.* **2026**, *28*, 2087-126. DOI
17. Yan, N.; Su, Y.; Deng, Y.; Schober, R. Federated fine-tuning of LLMs: framework comparison and research directions. *IEEE. Commun. Mag.* **2025**, *63*, 52-8. DOI
18. Adhikari, D.; Ullah, I.; Khadim, M.; et al. A comprehensive survey on robustness and privacy in federated learning meets large language model at edge. *J. Reliab. Secur. Comput.* **2026**, *2*, 111-55. DOI
19. Akhmetov, A.; Ala'Anzy, M. A.; Ibraheem, A. Federated learning strategies for fine-tuning large language models: a systematic literature review. 2026 11th International Conference on Information and Network Technologies (ICINT); 2026 Mar 6-8; Sydney, Australia. IEEE; 2026. pp. 27-32. DOI
20. Wu, X.; Zhang, C.; Lin, W. Federated learning with large language models. 2025 IEEE Cyber Science and Technology Congress (CyberSciTech); 2025 Oct 21-24; Hakodate, Japan. IEEE; 2025. pp. 620-5. DOI
21. Piccialli, F.; Chiaro, D.; Qi, P.; Bellandi, V.; Damiani, E. Federated and edge learning for large language models. *Information. Fusion.* **2025**, *117*, 102840. DOI
22. Vaswani, A.; Shazeer, N.; Parmar, N.; et al. Attention is all you need. 31st International Conference on Neural Information Processing Systems; 2017 Dec 4-9; Long Beach, California, USA; Curran Associates Inc.; 2017. pp 5998-6008. https://papers.nips.cc/paper/7181-attention-is-all-you-need (accessed 2026-09-14). DOI
23. Zhao, W. X.; Zhou, K.; Li, J.; et al. A survey of large language models. *Front. Comput. Sci.* **2026**, *20*, 2012627. DOI
24. Sun, G.; Khalid, U.; Mendieta, M.; Wang, P.; Chen, C. Exploring parameter-efficient fine-tuning to enable foundation models in federated learning. 2024 IEEE International Conference on Big Data (BigData); 2024 Dec 15-18; Washington, DC, USA. IEEE; 2024. pp. 8015-24. DOI
25. Chen, J.; Cai, Z.; Chen, W.; Wang, W.; Zheng, Z.; Yu, P. S. A federated adaptive large language model fine-tuning framework for software development. *IEEE. Trans. Serv. Comput.* **2026**, *19*, 32-43. DOI
26. Baali, F. A.; Ait-Mlouk, A.; Agouti, T. Federated instruction tuning with DeepSeek: towards scalable and private LLM adaptation. 2025 3rd International Conference on Federated Learning Technologies and Applications (FLTA); 2025 Oct 14-17; Dubrovnik, Croatia. IEEE; 2025. pp. 373-9. DOI
27. Yuan, W.; Yang, C.; Ye, G.; Chen, T.; Nguyen, Q. V. H.; Yin, H. FELLAS: Enhancing federated sequential recommendation with LLM as external services. *ACM. Trans. Inf. Syst.* **2025**, *43*, 1-24. DOI
28. Wang, Z.; Li, Z.; Guo, Y.; Tang, J. An investigation of parameter efficient federated learning with foundation model. 2025 10th International Conference on Machine Learning Technologies (ICMLT); 2025 May 23-25; Helsinki, Finland. IEEE; 2025. pp. 233-9. DOI
29. Zhang, S.; Cheng, G.; Wu, W.; Huang, X.; Song, L.; Shen, X. Split fine-tuning for large language models in wireless networks. *IEEE. J. Sel. Top. Signal. Process.* **2025**, *19*, 1376-91. DOI
30. Wu, W.; Li, M.; Qu, K.; et al. Split learning over wireless networks: parallel design and resource management. *IEEE. J. Select. Areas. Commun.* **2023**, *41*, 1051-66. DOI
31. Cheng, L.; Zhang, S.; Zhang, H.; et al. Large-small model collaboration in mobile edge networks with heterogeneous computational resources. *IEEE. J. Sel. Areas. Commun.* **2026**, *44*, 2733-49. DOI
32. Ghiasvand, S.; Alizadeh, M.; Pedarsani, R. Decentralized low-rank fine-tuning of large language models. 1st Workshop for Research on Agent Language Models (REALM 2025); 2025 Jun; Vienna, Austria. Stroudsburg, PA, USA: Association for Computational Linguistics; 2025. pp. 334-45. DOI
33. Balija, S. B.; Nanda, A.; Sahoo, D. Building communication efficient asynchronous peer-to-peer federated LLMs with blockchain. *AAAI-SS.* **2024**, *3*, 288-92. DOI

34. Woisetschläger, H.; Erben, A.; Wang, S.; Mayer, R.; Jacobsen, H. Federated fine-tuning of LLMs on the very edge: the good, the bad, the ugly. SIGMOD/PODS '24: International Conference on Management of Data; Santiago AA Chile. New York, NY, USA: ACM; 2024. pp. 39-50. DOI
35. Wu, F.; Hu, J.; Min, G.; Wang, S. Adaptive rank allocation for federated parameter-efficient fine-tuning of language models. *IEEE. Trans. Comput.* **2026**, *75*, 1650-63. DOI
36. Cai, D.; Wu, Y.; Wang, S.; Lin, F. X.; Xu, M. Efficient federated learning for modern NLP. ACM MobiCom '23: 29th Annual International Conference on Mobile Computing and Networking; Madrid Spain. New York, NY, USA: ACM; 2023. pp. 1-16. DOI
37. Gao, Z.; Zhang, Z.; Guo, Y.; Gong, Y. Federated adaptive fine-tuning of large language models with heterogeneous quantization and LoRA. IEEE INFOCOM 2025 - IEEE Conference on Computer Communications; 2025 May 19-22; London, United Kingdom. IEEE; 2025. pp. 1-10. DOI
38. Cai, D. Federated LLM pre-training on mobile phones. MobiSys '25: 23rd Annual International Conference on Mobile Systems, Applications and Services; Hilton Anaheim Anaheim CA USA. New York, NY, USA: ACM; 2025. pp. 657-8. DOI
39. Bai, G.; Li, Y.; Li, Z.; Zhao, L.; Kim, K. FedSpaLLM: federated pruning of large language models. Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers); 2025 Mar; Albuquerque, New Mexico. Stroudsburg, PA, USA: Association for Computational Linguistics; 2025. pp. 8361-73. DOI
40. Du, Y.; Ye, R.; Yuchi, F.; et al. FedDQC: Data quality control in federated instruction-tuning of large language models. Findings of the Association for Computational Linguistics: ACL 2025; 2025 Jun; Vienna, Austria. Stroudsburg, PA, USA: Association for Computational Linguistics; 2025. pp. 15267-91. DOI
41. Qin, Z.; Wu, Z.; He, B.; Deng, S. Federated data-efficient instruction tuning for large language models. Findings of the Association for Computational Linguistics: ACL 2025; 2025 Jun; Vienna, Austria. Stroudsburg, PA, USA: Association for Computational Linguistics; 2025. pp. 15550-68. DOI
42. Zhang, Z.; Zhang, J.; Huang, J.; et al. PPFedIT: towards privacy-preserving federated instruction tuning with few-shot local examples. *ACM. Trans. Intell. Syst. Technol.* **2026**, *17*, 1-23. DOI
43. Zhan, S.; Li, L.; Xu, C. AssyLLM: Efficient federated fine-tuning of LLMs via assembling pre-trained blocks. 2025 USENIX Annual Technical Conference; 2025; pp 1677-91. Available online: https://www.usenix.org/conference/atc25/presentation/zhan (accessed 2026-09-14).
44. Che, T.; Liu, J.; Zhou, Y.; et al. Federated learning of large language models with parameter-efficient prompt tuning and adaptive optimization. 2023 Conference on Empirical Methods in Natural Language Processing; 2023 Nov; Singapore. Stroudsburg, PA, USA: Association for Computational Linguistics; 2023. pp. 7871-88. DOI
45. Kim, G.; Yoo, J.; Kang, S. Efficient federated learning with pre-trained large language model using several adapter mechanisms. *Mathematics.* **2023**, *11*, 4479. DOI
46. Saadati, N.; Jiang, Z.; Balu, A.; Liu, C.; Hegde, C.; Sarkar, S. Foundation model efficient fine-tuning in centralized and federated settings. 2025 IEEE International Conference on Big Data (BigData); 2025 Dec 8-11; Macau, China. IEEE; 2025. pp. 1857-66. DOI
47. Ghiasvand, S.; Yang, Y.; Xue, Z.; Alizadeh, M.; Zhang, Z.; Pedarsani, R. Communication-efficient and tensorized federated fine-tuning of large language models. Findings of the Association for Computational Linguistics: ACL 2025; 2025 Jun; Vienna, Austria. Stroudsburg, PA, USA: Association for Computational Linguistics; 2025. pp. 24192-207. DOI
48. Liu, H.; Wen, R.; Nair, S.; et al. EcoLoRA: communication-efficient federated fine-tuning of large language models. 2025 Conference on Empirical Methods in Natural Language Processing; 2025 Oct; Suzhou, China. Stroudsburg, PA, USA: Association for Computational Linguistics; 2025. pp. 20743-57. DOI
49. Liang, P.; Guo, J.; Zhao, M. Federated fine-tuning large language models with LoRA and non-orthogonal transmission. GLOBECOM 2025 - 2025 IEEE Global Communications Conference; 2025 Dec 8-12; Taipei, Taiwan. IEEE; 2025. pp. 405-10. DOI
50. Faiyaz, A.; Salman, T. GradualDiff-Fed: A federated learning specialized framework for large language model. 2025 IEEE 4th International Conference on Computing and Machine Intelligence (ICMI); 2025 Apr 5-6; MI, USA. IEEE; 2025. pp. 1-5. DOI
51. Rahman, S.; Rahman, R. Semantic communication-aware federated fine-tuning of large language models. *IEEE. Commun. Lett.* **2025**, *29*, 2974-7. DOI
52. Zhang, Z.; Liu, P.; Xu, J.; Hu, R. Fed-HeLLo: Efficient federated foundation model fine-tuning with heterogeneous LoRA allocation. *IEEE. Trans. Neural. Netw. Learning. Syst.* **2025**, *36*, 17556-69. DOI
53. Cho, Y. J.; Liu, L.; Xu, Z.; Fahrezi, A.; Joshi, G. Heterogeneous LoRA for federated fine-tuning of on-device foundation models. 2024 Conference on Empirical Methods in Natural Language Processing; 2024 Oct; Miami, Florida, USA. Stroudsburg, PA, USA: Association for Computational Linguistics; 2024. pp. 12903-13. DOI

54. Wang, Z.; Shen, Z.; He, Y.; et al. FLoRA: federated fine-tuning large language models with heterogeneous low-rank adaptations. Advances in Neural Information Processing Systems 37; 2024 Dec 10-15; Vancouver, BC, Canada. San Diego, California, USA: Neural Information Processing Systems Foundation, Inc. (NeurIPS); 2024. pp. 22513-33. DOI
55. Jiang, Y.; Li, Z.; Song, B. Fine-tuning large language models in federated learning with fairness-aware prompt selection. *Neural. Netw.* **2026**, *194*, 108160. DOI
56. Zhang, Y.; Wang, X.; Sun, W.; Chen, J.; Wang, F. FedBRICK: structural bias aware heterogeneous foundation model federated tuning. *AAAI.* **2026**, *40*, 28528-36. DOI
57. Liu, J.; Liao, Y.; Xu, H.; Xu, Y. Tackling data heterogeneity in parameter-efficient federated fine-tuning of large language models. ICASSP 2026 - 2026 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP); 2026 May 3-8; Barcelona, Spain. IEEE; 2026. pp. 17202-6. DOI
58. Wang, Z.; Zhou, Y.; Shi, Y.; Letaief, K. B. Federated fine-tuning for pre-trained foundation models over wireless networks. *IEEE. Trans. Wireless. Commun.* **2025**, *24*, 3450-64. DOI
59. Iftikhar, S.; Alsamhi, S. H.; Davy, S. Enhancing sustainability in LLM training: leveraging federated learning and parameter-efficient fine-tuning. *IEEE. Trans. Sustain. Comput.* **2025**, *10*, 1158-72. DOI
60. Wang, Z.; Zhou, Y.; Shi, Y.; Letaief, K. B. Federated low-rank adaptation for large language model fine-tuning over wireless networks. GLOBECOM 2024 - 2024 IEEE Global Communications Conference; 2024 Dec 8-12; Cape Town, South Africa. IEEE; 2024. pp. 3063-8. DOI
61. Zhao, K.; Zhu, C.; Chen, M.; Huang, C.; Yang, Z.; Zhang, Z. SflLLM: efficient split federated learning for large language model over wireless networks. GLOBECOM 2025 - 2025 IEEE Global Communications Conference; 2025 Dec 8-12; Taipei, Taiwan. IEEE; 2025. pp. 1835-40. DOI
62. Otoum, Y.; Danish, S. M.; Ahmad, I.; Alkhrijah, Y.; Asad, A. Efficient federated LLM framework for intelligent IoMT management. *IEEE. Trans. Consumer. Electron.* **2026**, *72*, 5832-47. DOI
63. Wang, F.; Li, B. Data reconstruction and protection in federated learning for fine-tuning large language models. *IEEE. Trans. Big. Data.* **2024**, 1-13. DOI
64. Zheng, J.; Zhang, H.; Wang, L.; Qiu, W.; Zheng, H.; Zheng, Z. Safely learning with private data: a federated learning framework for large language model. 2024 Conference on Empirical Methods in Natural Language Processing; 2024 Oct; Miami, Florida, USA. Stroudsburg, PA, USA: Association for Computational Linguistics; 2024. pp. 5293-306. DOI
65. Pontes, M. F.; Pedrosa, R. C.; Lopes, P. H.; Luz, E. J. Evaluating federated learning with homomorphic encryption for medical named entity recognition using compact BERT models. Simpósio Brasileiro de Tecnologia da Informação e da Linguagem Humana; Brasil. Sociedade Brasileira de Computação; 2024. pp. 48-56. DOI
66. Kim, S.; Lim, C.; Ryu, G.; Kim, H. How robust are language models against backdoors in federated learning? *CMES.* **2025**, *145*, 2617-30. DOI
67. Zhao, J.; Fang, M.; Zhong, M.; Zheng, S.; Chen, L.; Pechenizkiy, M. Investigating social bias propagation in federated fine-tuning of large language models. *AAAI.* **2026**, *40*, 39637-45. DOI
68. Li, X.; Wu, C.; Wang, J. Foundation models in federated learning: assessing backdoor vulnerabilities. 2025 International Joint Conference on Neural Networks (IJCNN); 2025 Jun 30-Jul 5; Rome, Italy. IEEE; 2025. pp. 1-8. DOI
69. Xie, S.; Wen, D.; You, C.; Chen, Q.; Bennis, M.; Huang, K. FedLoDrop: Federated LoRA with dropout for generalized LLM fine-tuning. *IEEE. J. Sel. Areas. Commun.* **2026**, *44*, 3541-56. DOI
70. Fang, Z.; Lin, Z.; Chen, Z.; Chen, X.; Gao, Y.; Fang, Y. Automated federated pipeline for parameter-efficient fine-tuning of large language models. *IEEE. Trans. on. Mobile. Comput.* **2026**, *25*, 8782-97. DOI
71. Wu, F.; Li, Z.; Li, Y.; Ding, B.; Gao, J. FedBiOT: LLM local fine-tuning in federated learning without full model. KDD '24: The 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining; Barcelona Spain. New York, NY, USA: ACM; 2024. pp. 3345-55. DOI
72. Su, Y.; Yan, N.; Deng, Y. Federated LLMs fine-tuned with adaptive importance-aware LoRA. ICC 2025 - IEEE International Conference on Communications; 2025 Jun 8-12; Montreal, QC, Canada. IEEE; 2025. pp. 6112-7. DOI
73. Li, Y.; Sun, J.; Liu, Y.; et al. Federated black-box prompt tuning system for large language models on the edge. ACM MobiCom '24: 30th Annual International Conference on Mobile Computing and Networking; Washington D.C. DC USA. New York, NY, USA: ACM; 2024. pp. 1775-7. DOI
74. Tanimura, T.; Nakano, W.; Kitagawa, Y.; Takase, M. Federated discrete prompt tuning for language models using synthetic examples. 2025 IEEE 22nd Consumer Communications & Networking Conference (CCNC); 2025 Jan 10-13; Las Vegas, NV, USA. IEEE; 2025. pp. 1-4. DOI
75. Wu, J.; Chen, S.; Tang, J.; et al. FDPT: Federated discrete prompt tuning for black-box visual-language models. 2025 IEEE/CVF International Conference on Computer Vision (ICCV); 2025 Oct 19-25; Honolulu, HI, USA. IEEE; 2025. pp. 1-10. DOI

76. Fan, B.; Su, X.; Tarkoma, S.; Hui, P. HeLoRA: LoRA-heterogeneous federated fine-tuning for foundation Models. *ACM. Trans. Internet. Technol.* **2025**, *25*, 1-22. DOI
77. Chen, Y.; Li, R.; Shao, J.; et al. Federated LoRA fine-tuning of LLMs with only transmitting matrix A or B. *IEEE. Trans. Mobile. Comput.* **2026**, *25*, 17503-19. DOI
78. Guo, W.; Lu, S.; Tong, Y.; et al. H2Tune: federated foundation model fine-tuning with hybrid heterogeneity. In: Lynce I, Murano N, Vallati M, Villata S, Chesani F, Milano M, Omicini A, Dastani M, Editors. ECAI 2025. IOS Press; 2025. DOI
79. Wang, X.; Qiao, Y.; Wu, D.; Wu, C.; Wang, F. Cluster based heterogeneous federated foundation model adaptation and fine-tuning. *AAAI.* **2025**, *39*, 21269-77. DOI
80. Yang, Y.; Long, G.; Shen, T.; Jiang, J.; Blumenstein, M. Dual-personalizing adapter for federated foundation models. Advances in Neural Information Processing Systems 37; 2024 Dec 10-15; Vancouver, BC, Canada. San Diego, California, USA: Neural Information Processing Systems Foundation, Inc. (NeurIPS); 2024. pp. 39409-33. DOI
81. Bian, J.; Wang, L.; Zhang, L.; Xu, J. FedALT: federated fine-tuning through adaptive local training with rest-of-world LoRA. *AAAI.* **2026**, *40*, 19728-36. DOI
82. Chang, Y.; Shi, X.; Zhao, X.; Chen, Z.; Ma, D. Dual prompt personalized federated learning in foundation models. *Sci. Rep.* **2025**, *15*, 28026. DOI PubMed PMC
83. Su, S.; Yang, M.; Li, B.; Xue, X. Federated adaptive prompt tuning for multi-domain collaborative learning. *AAAI.* **2024**, *38*, 15117-25. DOI
84. Pang, Z.; Wei, K.; Shi, L.; Wang, Z.; Li, J.; Shu, F. Low-latency federated fine-tuning for large language models over wireless networks. *IEEE. Wireless. Commun. Lett.* **2026**, *15*, 2179-83. DOI
85. Chen, H.; Yuan, X.; Li, H. Edge-assisted federated learning for large language models in IoT sensor systems. *IEEE. J. Sel. Areas. Sensors.* **2026**, *3*, 125-38. DOI
86. Chen, X.; Li, Z.; Ni, W.; et al. Toward dynamic resource allocation and client scheduling in hierarchical federated learning: a two-phase deep reinforcement learning approach. *IEEE. Trans. Commun.* **2024**, *72*, 7798-813. DOI
87. Zhang, X.; Yan, N.; Su, Y.; Deng, Y.; Mahmoodi, T. Communication-aware knowledge distillation for federated LLM fine-tuning over wireless networks. GLOBECOM 2025 - 2025 IEEE Global Communications Conference; 2025 Dec 8-12; Taipei, Taiwan. IEEE; 2025. pp. 1956-61. DOI
88. Zhao, K.; Yang, Z.; Huang, C.; Chen, X.; Zhang, Z. FedsLLM: federated split learning for large language models over communication networks. 2024 International Conference on Ubiquitous Communication (Ucom); 2024 Jul 5-7; Xi'an, China. IEEE; 2024. pp. 438-43. DOI
89. Dai, J.; Tang, R.; Jiang, F.; et al. Federated split learning for large language models with RSMA. *IET. Communications.* **2026**, *20*, e70152. DOI
90. Zhao, J.; Wang, W.; Xu, C.; Ng, S.; Chua, T. A federated framework for LLM-based recommendation. Findings of the Association for Computational Linguistics: NAACL 2025; 2025 Mar; Albuquerque, New Mexico. Stroudsburg, PA, USA: Association for Computational Linguistics; 2025. pp. 2852-65. DOI
91. Gao, X.; Hou, L.; Chen, B.; Yao, X.; Suo, Z. Compressive-learning-based federated learning for intelligent IoT with cloud-edge collaboration. *IEEE. Internet. Things. J.* **2025**, *12*, 2291-4. DOI
92. Andrei, V. C.; Djuhera, A.; Li, X.; Mönich, U. J.; Saad, W.; Boche, H. Resilient, Federated large language models over wireless networks: why the PHY matters. GLOBECOM 2024 - 2024 IEEE Global Communications Conference; 2024 Dec 8-12; Cape Town, South Africa. IEEE; 2024. pp. 5211-6. DOI
93. Djuhera, A.; Andrei, V. C.; Li, X.; Mönich, U. J.; Boche, H.; Saad, W. R-SFLLM: jamming resilient framework for split federated learning with large language models. *IEEE. Trans. Inform. Forensic. Secur.* **2025**, *20*, 8296-311. DOI
94. Wang, H.; Yin, Z.; Chen, B.; et al. ROFED-LLM: robust federated learning for large language models in adversarial wireless environments. *IEEE. Trans. Netw. Sci. Eng.* **2026**, *13*, 1084-96. DOI
95. Park, C.; Han, S.; Guo, X.; Ozdaglar, A. E.; Zhang, K.; Kim, J. MAPoRL: Multi-agent post-co-training for collaborative large language models with reinforcement learning. 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers); 2025 Jun; Vienna, Austria. Stroudsburg, PA, USA: Association for Computational Linguistics; 2025. pp. 30215-48. DOI
96. Pan, Q.; Wu, J. Selective privacy-preserving federated learning for large language model fine-tuning. 2025 International Wireless Communications and Mobile Computing (IWCMC); 2025 May 12-16; Abu Dhabi, United Arab Emirates. IEEE; 2025. pp. 1626-31. DOI
97. Wang, S.; Han, S.; Cheng, Z.; Wang, M.; Li, Y. Federated fine-tuning of large language models with privacy preservation and cross-domain semantic alignment. 2025 6th International Conference on Computer Vision and Data Mining (ICCVDM); 2025 Sep 12-14; London, United Kingdom. IEEE; 2025. pp. 494-8. DOI

98. Kumar, G. S.; Vankudothu, B.; Tiwari, A. K.; Pushparathi, V. G.; Sathiya, B.; Prasan, U. D. Privacy-aware federated large language model adaptation using encrypted gradient aggregation. 2026 International Conference on Electronic Systems and Intelligent Computing (ICESIC); 2026 Mar 13-14; Chennai, India. IEEE; 2026. pp. 420-5. DOI
99. Shanmugam, K.; B, N.; K, K. K. Federated instruction-tuning of large language models with privacy and communication efficiency. 2026 Contemporary Computing Innovations Conference (CCIC); 2026 Feb 6-7; Tirupati, India. IEEE; 2026. pp. 1-7. DOI
100. Li, J.; Fu, M.; Zhang, X.; Li, J.; Li, J. Privacy-preserving and efficient aggregation for federated large language models. 2025 International Conference on Artificial Intelligence Security and Governance (ICAISG); 2025 Dec 12-14; Hangzhou, China. IEEE; 2025. pp. 65-9. DOI
101. Zhang, T.; Yu, H.; Yang, Z.; Chen, Y.; Yu, S. LaVFL: efficient verifiable federated learning for large language models. *IEEE. Trans. Dependable. and. Secure. Comput.* **2025**, *22*, 6214-29. DOI
102. Wu, Y.; Ren, Y.; Guo, Z.; Huang, M. Privacy-preserving personalized federated prompt learning for vision-language models. *Neural. Netw.* **2026**, *195*, 108220. DOI
103. Ni, F.; Zhou, Z.; Ni, W.; et al. Scheduling and securing asynchronous federated learning through cooperative jamming. *IEEE. Trans. Cogn. Commun. Netw.* **2026**, *12*, 3209-22. DOI
104. Faiyaz, A.; Olapojoye, R.; Karwa, G.; Salman, T. A study of model poisoning attacks on federated large language models. 2026 IEEE 5th International Conference on Computing and Machine Intelligence (ICMI); 2026 Apr 9-10; Al-Ahsa, Saudi Arabia. IEEE; 2026. pp. 1-6. DOI
105. Zhai, K.; Chen, S.; Ma, X.; Jiang, Y. FedAPT: federated adversarial prompt tuning for vision-language models. MM '25: The 33rd ACM International Conference on Multimedia; Dublin Ireland. New York, NY, USA: ACM; 2025. pp. 4310-8. DOI
106. Nehara, T.; Samaraweera, C. K.; Nettasinghe, O.; et al. DistilGuard - large language models for poisoning detection in federated learning. 2025 IEEE Conference on Communications and Network Security (CNS); 2025 Sep 8-11; Avignon, France. IEEE; 2025. pp. 1-9. DOI
107. Zhang, Y.; Jiang, C.; Zhang, P. Security-aware resource allocation scheme based on DRL in cloud-edge-terminal cooperative vehicular network. *IEEE. Internet. Things. J.* **2024**, *11*, 95-104. DOI
108. Lu, J.; Pan, B.; Yu, J.; Jiang, W.; Han, J.; Ye, Z. Towards energy-efficient and time-sensitive task assignment in cross-silo federated learning. *J. King. Saud. Univ-Com.* **2023**, *35*, 63-74. DOI
109. Xiong, B.; Yang, X.; Song, Y.; Wang, Y.; Xu, C. Pilot: building the federated multimodal instruction tuning framework. *AAAI.* **2025**, *39*, 21716-24. DOI
110. Singha, M.; Roy, S.; Mehrotra, S.; et al. FedMVP: federated multimodal visual prompt tuning for vision-language models. 2025 IEEE/CVF International Conference on Computer Vision (ICCV); 2025 Oct 19-25; Honolulu, HI, USA. IEEE; 2025. pp. 1-10. DOI
111. Bala, A.; Vereshchaka, A. Multimodal LLM using federated visual instruction tuning for visually impaired. ICMI '25: International Conference on Multimodal Interaction; Canberra Australia. New York, NY, USA: ACM; 2025. pp. 191-9. DOI
112. Wang, H.; Zhao, S.; Wang, J.; Qiang, Z.; Qin, B.; Liu, T. Beyond frameworks: unpacking collaboration strategies in multi-agent systems. 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers); 2025 Jun; Vienna, Austria. Stroudsburg, PA, USA: Association for Computational Linguistics; 2025. pp. 21361-75. DOI
113. Chen, X.; Chen, S.; Ni, W.; et al. Optimal two-timescale configuration of mobile edge computing with mixed energy supply. *IEEE. Trans. Smart. Grid.* **2024**, *15*, 4765-78. DOI