Communication Efficiency in Decentralized Federated Learning: A Taxonomy of Core Components and Optimization Techniques*

David Weissteiner[a,b,c,*], Lea Demelius[b,a] and Andreas Trügler[b,d]

[a]Graz University of Technology, Rechbauerstrasse 12, Graz, 8010, Styria, Austria
[b]Know Center Research GmbH, Sandgasse 34, Graz, 8010, Styria, Austria
[c]BIFOLD, Technische Universität Berlin, Franklinstrasse 28/29, Berlin, 10587, Berlin, Germany
[d]Institute of Geography and Regional Science, University of Graz, Heinrichstrasse 36, Graz, 8010, Styria, Austria

ARTICLE INFO

Keywords:
decentralized federated learning (DFL)
communication efficiency
resource-constrained distributed optimization
communication optimization

ABSTRACT

Federated Learning enables collaborative machine learning in distributed data environments without the consolidation of raw data. To avoid the communication bottleneck and the single point of failure found in Centralized Federated Learning, Decentralized Federated Learning (DFL) eliminates the coordinating central server and distributes its responsibilities among the participants. However, the challenge of communication efficiency persists in DFL, resulting in a trade-off between communication overhead and convergence rate. Despite efforts already invested in improving communication efficiency, the existing literature lacks a comprehensive overview of communication aspects in DFL. To fill this gap, this work systemizes recent DFL publications with a particular focus on communication cost of core constituents in horizontal DFL. We establish a taxonomy of essential DFL components that influence communication by identifying classes of strategies and linking them to implementations in the current literature. Next, we examine popular techniques to optimize the communication cost in DFL and categorize these methods into sub-classes. To demonstrate the applicability of our taxonomy, we then compare the methods of recent DFL publications by taking these components and optimization techniques into account. This paper provides an examination and categorization of key communication aspects in DFL, valuable for both researchers and practitioners engaged in resource-constrained distributed optimization.

1. Introduction

Federated Learning (FL) is a machine learning (ML) paradigm that enables model training on private, decentralized data partitions by sharing model parameter updates instead of raw data [134]. In contrast to distributed ML [182], the data resides undisclosed with their respective producing devices instead of being collected, split into partitions, and distributed across devices. Due to its privacy-preserving capabilities, FL has become a common technique in many domains. Beyond privacy preservation, FL addresses the performance bottleneck of sending large data files over the network in distributed ML. Although extensive research on FL has been conducted in recent years, we still face significant challenges, including communication efficiency, system heterogeneity, data heterogeneity, and privacy concerns [103, 80].

Traditional FL (also known as *centralized FL*, CFL) [22] is based on a central server that orchestrates the learning process and aggregates model updates. However, this setting induces a single point of communication for all participating devices, which eventually causes a major communication bottleneck at the server. Additionally, the server composes a single point of failure in the network. For that reason, *decentralized FL* (DFL) [19, 64] emerges to address these problems by removing the central server from CFL, thus introducing complete decentralization. Under this changed setting, the participating devices exchange model updates with their neighbors and aggregate them autonomously to come up with a collaborative model.

Although DFL tries to eliminate the communication bottleneck at the central node by removing the server and outsourcing its responsibilities to the individual devices (actors), communication bottlenecks can still occur at the actors themselves. In the simplest DFL strategy, the network consists of fully-connected actors, where each actor sends its model update to every other actor. Because every actor must receive all peers' updates, the communication bottleneck moves from the CFL server to every actor. Hence, just eliminating the server does not suffice to resolve the communication bottleneck. Nevertheless, we can alleviate the communication bottleneck in DFL by reducing the communication cost through dedicated configurations of model aggregation techniques, synchronization methods, and network topologies. In addition to these general DFL configurations, we can apply techniques such as compression, local computing, and partial device participation to further reduce communication overhead.

Contributions In this work, we revisit DFL with a special focus on the open research challenge of communication

*This work was supported by the "PRO'k'RESS" project (grant No. 60155996) and the "QUICHE" project (grant No. 54741739), both funded by the Austrian Research Promotion Agency (FFG).

*Corresponding author
david.weissteiner@alumni.tugraz.at (D. Weissteiner); ldemelius@know-center.at (L. Demelius); andreas.truegler@uni-graz.at (A. Trügler)
ywcb00.ywcb.org (D. Weissteiner); leakatharina24.github.io (L. Demelius); atruegler.at (A. Trügler)
ORCID(s): 0009-0004-2809-8342 (D. Weissteiner); 0009-0000-7449-0397 (L. Demelius); 0000-0003-2254-7894 (A. Trügler)

efficiency[1] and discuss the latest advances in this field. In contrast to existing work, we explicitly elaborate on the communication efficiency of key system constituents that characterize communication in the DFL infrastructure. Thus, we provide a guideline with important considerations for constructing a DFL system in bandwidth-limited environments. To our knowledge, the current literature does not include surveys that elaborate on the communication efficiency of the entire DFL system. An overview of existing surveys and their coverage of certain communication aspects is listed in Table 1. Communication efficiency constitutes a critical aspect across all areas of DFL, including robust DFL and personalized DFL. In this paper, we refrain from examining these sub-fields individually and instead concentrate on the communication efficiency challenge inherent to core DFL methodologies.

Our key contributions are as follows:

- We systemize DFL methods from the literature with a particular focus on communication cost.
- We identify three essential DFL components with a major impact on communication cost and discuss three popular techniques to optimize communication efficiency in DFL approaches.
- We establish taxonomies within the three DFL components and within the three optimization techniques, and unveil their impact on communication.
- We apply our taxonomy on DFL approaches from recent literature to demonstrate its applicability and reveal combinations that have already been addressed by research.

Search Process The publications discussed in this paper have been collected in a search process using the search engines from Semantic Scholar (sorted by Relevance), Google Scholar, ACM Digital Library, and IEEE Xplore (sorted by Relevance), employing basic search within the time frame 2018 to 2025 with the following search queries: *Communication-efficient Decentralized Federated Learning*, *Communication Decentralized Federated Learning*, *Bandwidth Decentralized Federated Learning*, *Resource-constrained Decentralized Federated Learning*, *Fully Decentralized Federated Learning*, *Communication Peer-to-Peer Federated Learning*, *Communication-efficient P2P Federated Learning*, *Communication Distributed Federated Learning*. Our scope is confined to non-blockchain approaches for horizontal DFL[2], reflecting the predominant focus of existing research and ensuring consistency of setting throughout this document. We selected 13 approaches in such a way as to include the purest forms of the underlying, communication-efficient techniques from recent literature [86, 98, 188, 173, 184, 119, 236, 190, 239, 115, 57, 113, 161]. We provide a list with reasons for not including publications in Appendix A.2.

Structure We organize this paper in the following structure: In Section 2, we provide background information on centralized FL and decentralized FL, and explore existing surveys in the field; In Section 3, we elaborate on different implementations of essential components in DFL based on recent literature and discuss their impact on communication efficiency; Section 4 reviews supplementary techniques to optimize communication cost in DFL and discusses their respective impact; Section 5 analyzes DFL approaches with regard to their composition using the taxonomy presented in Section 3 and Section 4, to draw comparisons and to reveal combinations covered by research; Section 7 discusses open challenges to be addressed in future work; Section 8 summarizes the insights from the previous sections and concludes our findings.

2. Background and Related Work

2.1. Centralized Federated Learning

The expanding field of machine learning (ML), with its constant desire for access to more data, is drawing increasing attention to data privacy considerations. Privacy constraints and regulations that arise from these considerations often prevent data transfer from edge devices or enterprise locations to a central location where traditional ML is performed. Therefore, FL has been proposed to overcome these restrictions without violating privacy constraints [134].

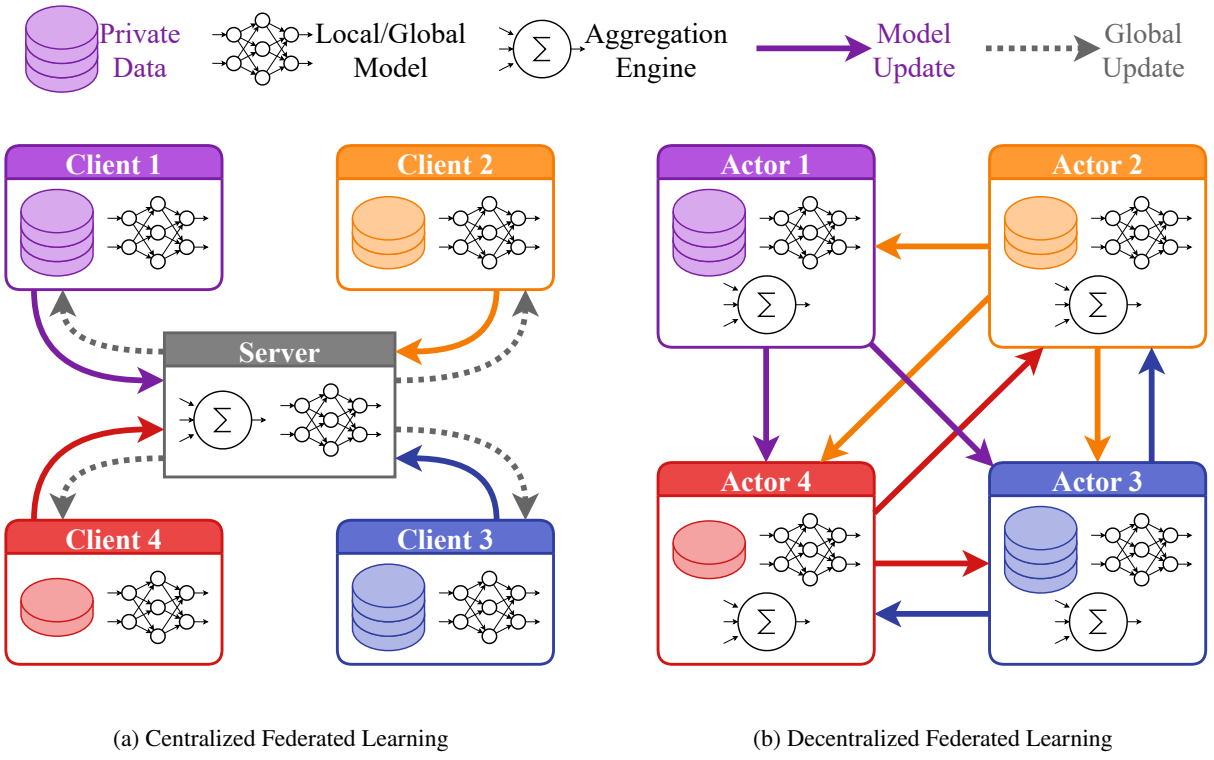
Centralized federated learning (CFL)—as the primal variant of FL—enables model training at a centralized node (server) on decentralized data from multiple client devices (federated workers) with privacy constraints. This is achieved by utilizing the computing resources of the respective federated workers to compute pre-aggregates from the data which do not allow for the recreation of the source data. Subsequently, the federated workers send their individual pre-aggregates to the centralized node (server), where they are aggregated and finalized to a global model, which is then returned to the federated workers (see Figure 1 bottom left). CFL thereby addresses concerns regarding privacy, ownership, and locality of data [22].

Research in the field of CFL mainly focuses on four central challenges [103, 80]:

1. *Security and Privacy*: Although FL addresses some privacy concerns, research has shown that model updates can still reveal sensitive information about the underlying training data [238, 71, 53]. CFL is therefore often combined with other privacy-enhancing technologies, for example differential privacy [196] or homomorphic encryption [69, 226]. Additionally, security challenges arise such as availability and integrity [30]. The central server represents a single point of failure, making it an attractive target for attacks.

[1] Communication efficiency refers to the minimized utilization of communication resources (e.g., bandwidth, network links, etc.) required to maintain convergence, and is assessed based on communication frequency, information transfer volume, and related metrics.

[2] In horizontal DFL, the individual datasets at the actors share the same feature space [214].



(a) Centralized Federated Learning

(b) Decentralized Federated Learning

Figure 1: Illustration of system architectures; (1a) Clients train their local model on private data and send model updates to the server which aggregates them into one global model update and distributes it to the clients; (1b) Actors train their model on private data, transmit model updates to neighboring actors (arbitrary connections here), and aggregate received model updates to their local model.

2. *Data Heterogeneity*: Data partitions violate the assumptions about independent and identically distributed (I.I.D.) data in distributed optimization. For that reason, traditional CFL methods face the challenge of poor convergence on heterogeneous data. The majority of existing work on preventing such convergence issues relies on techniques from data augmentation, client selection, regularization, meta-learning, and transfer learning [169].
3. *System Heterogeneity*: Federated workers often differ in their computing, communication, and storage resources as well as in their general availability. To mitigate the problems arising from heterogeneous client resources, asynchronous CFL and semi-asynchronous CFL have been proposed [206].
4. *Communication Bottleneck*: All federated workers send their model updates to the same central server. Current approaches for reducing communication cost mainly focus on partial client participation, local updating, and compression techniques [233, 155, 4]. Furthermore, the paradigm of decentralized FL—on which we focus in the present document—aims to overcome the central communication bottleneck.

2.2. Decentralized Federated Learning

Decentralized federated learning (DFL) is a variant of federated learning that enables devices to train collaborative ML models without centralized coordination. In the literature, DFL is also referred to as peer-to-peer FL (P2P FL), gossip FL, or distributed FL. Contrary to CFL, there is no distinction between the server and clients in DFL. During DFL model training, the actors share the same responsibilities of exchanging updates to model parameters with their respective neighbors and incorporating information from the received updates into the training process (depicted in Figure 1 bottom right). That way, DFL eliminates the need for a central entity and, thus, removes the single point of failure and the major communication bottleneck arising from the server in CFL.

Although the high-level objective of DFL remains similar to CFL—namely to train ML models in distributed environments subject to privacy constraints—, eliminating the central coordination entity entails some key conceptual changes. Due to the absence of a single point of communication, DFL requires dedicated strategies to handle the communication of model parameter updates between neighboring actors. In addition, every actor has to aggregate the updates from its neighbors autonomously. However, the neighborhood does not necessarily include all other actors, but only a subset. Therefore, we have to consider model aggregation with a limited set of model parameter updates in DFL.

Besides the absence of a central coordination entity, the system prerequisites in DFL are similar to CFL: Multiple

devices with computing resources and private data are connected via the network. Based on these prerequisites, we can set up a DFL system which is characterized by three core components:

1. *Learning units* to enable model training on the actors using their local data.
2. *Inter-node communication* to exchange updates to model parameters between neighboring actors.
3. *Aggregation engine* to incorporate received updates to model parameters into the learning process.

Together, these components represent the three stages comprising DFL: (1) Local updating, (2) model update exchange, and (3) model aggregation. During training, the sequence of these stages is executed iteratively, with one pass through all three stages being referred to as a communication round. Within these three segments, it is possible to implement further techniques to optimize system behavior and overcome existing challenges.

Research in the field of DFL entails many of the same core challenges as CFL. However, due to the difference in setting, we need to address these challenges from a different perspective. Besides the communication efficiency, on which we focus in the present work, these challenges comprise:

- *Security and Privacy*: While DFL eliminates server-related threats (e.g., single point of failure), it inherits all other security and privacy challenges of centralized FL, and often amplifies the respective risks due to lack of trust and the higher complexity of peer-to-peer communication [59]. Key threats include: (1) node failures or dropouts, (2) malicious actors (including model poisoning attacks), and (3) information leakage (including inference attacks on private data). The likelihood of node failures and dropouts is increased due to diverse device capabilities and network conditions, requiring robust fault-tolerant solutions that avoid model version inconsistency. Enforcing trust and validating nodes is also more challenging without a central server, leading to the adoption of cryptographic methods and blockchain technologies to authenticate participants and ensure data integrity. Poisoning attacks are a major concern in FL, as malicious nodes can inject manipulated model updates into the system to degrade performance or introduce backdoors. Without a central aggregator, detecting such poisoned updates is more difficult since each node only observes a subset of updates (with exemption of fully-connected networks). Apart from robust aggregation rules and verification of nodes, anomaly detection is applied to filter out malicious updates (see e.g., [49]). To protect against information leakage, DFL can be combined with privacy-enhancing technologies such as multi-party computation (MPC), differential privacy (DP) and homomorphic encryption (HE), though doing so is generally more challenging than in CFL due to its decentralized, peer-to-peer architecture [59]. Multi-party computation [215] is employed to enable participants to jointly compute aggregates without revealing their individual model updates (see for example [82]). Differential privacy [45, 46] is widely used to prevent inference of individual data points from model updates or from the trained model itself. In DFL, the most prominent approach is local DP, where each node adds carefully calibrated noise to its model updates before sharing them with peers. While well-aligned with decentralized systems, local DP often leads to significant loss of accuracy. To mitigate this, researchers have proposed a novel DP notion called network DP that leverages the properties of DFL to amplify privacy guarantees [39, 40]. Homomorphic encryption enables participants to perform computations on encrypted model updates. While HE is a prominent approach in CFL, its deployment in DFL is more challenging due to the difficulty of coordinating key management, calling for adaptions of standard HE such as multi-key HE (see e.g., [67]).
- *Data Heterogeneity*: Similar to CFL, DFL faces the challenge of heterogeneous data distributions between actors, leading to performance degradation and convergence issues. Approaches to tackle this challenge are divided into two differing objectives: (1) Global convergence or (2) personalized models. Approaches with the goal of global convergence continue to aggregate the diverging models at a higher level to reach convergence of the globally aggregated model. To achieve this, Wang et al. [183] incorporate synthetic data generation to locally re-balance the training datasets. In [33], on the other hand, the authors utilize the techniques of gradient push and neighbor selection to enhance global performance on heterogeneous data. Additionally, several new or modified aggregation schemes evolved to address this challenge, such as model fusion via a model knowledge transfer algorithm [98], self-knowledge distillation [161], a localized primal-dual method [179], and the incorporation of synthetic data from other actors [74]. Personalized models (also called customized models), on the contrary, encourage actors to learn different (i.e., personalized) models while still benefiting from the common basis among the individual models. Some existing approaches tackle this objective by introducing a penalty in the objective function [150], by re-weighting model updates based on their similarity to the individual local models [16], or by employing personalized adaptive masks to customize sparse local models [41].
- *System Heterogeneity*: Due to the heterogeneity of devices in computing resources and connectivity, participating actors differ in (1) computation time to perform local updating, (2) communication reliability due to interrupted connections, and (3) network latency due to differing bandwidths. Therefore, the

DFL system encounters problems of stragglers (i.e., slow devices) that slow down the global learning process. Asynchronous training processes help to overcome this problem by continuing the training without waiting for the other actors—thereby ignoring stragglers in the system and eliminating respective idle times (see Section 3.2) [26, 115, 113]. Besides asynchronous DFL, the challenge of system heterogeneity is also addressed by semi-synchronized strategies, for instance by allowing the actors to continue their local training until the slowest actor completes its local update [236]. With both the asynchronous and the semi-synchronous methods, however, the issue of stale (i.e., outdated) model updates arises, since actors learn at different rates. On top of asynchronous training, Cao et al. [26] propose a dynamic adjustment of local training steps with probability-based neighbor selection to further optimize the learning process with heterogeneous devices. Alternatively, in [115], the authors implement asynchronous training in combination with a model selection strategy that is based on reinforcement learning and dynamic weighting of neighbor models. Apart from asynchronous DFL, research also addresses system heterogeneity in synchronous systems. In [216], the authors explicitly address unreliable communication by enabling the aggregation of partially received models while dynamically optimizing the mixing weights according to the reliability of the individual network link. Similar to approaches in the asynchronous case, Liao et al. [112] employ heterogeneity-aware methods for controlling the amount of local updating steps and for neighbor selection in synchronous DFL.

Underpinning all of these challenges is the cross-cutting problem of maintaining model consistency and accelerating convergence in a fully distributed setting, where no central authority exists to coordinate updates or enforce synchronization. While the challenges outlined above are active and important areas of research, they fall outside the scope of this work. Many of the approaches we discuss may contribute to addressing them, but our focus is squarely on communication efficiency.

The emerging field of robust DFL addresses these challenges by enhancing resilience against adversarial threats (security and privacy), unreliable participants (system heterogeneity), and inconsistent data (data heterogeneity) [180]. In certain applications (e.g., satellite networks [47, 199, 211, 213, 222] and vehicles [12, 20, 62, 141, 146, 170, 204, 205]), robust DFL extends to scenarios with completely unresponsive participants due to harsh channel conditions or unreliable participants. This relates closely to the technique of *partial device participation* (see Section 4.3), which addresses scenarios where only a fraction of the actors participate in the learning process. This is particularly critical in DFL, where the absence of a central coordinating server means that each participant must independently assess the trustworthiness of its peers. Additionally, robustness in wireless networks becomes an increasingly important topic, yet most associated challenges arise from wireless signal properties rather than the learning paradigm [216]. There exists a trade-off between communication cost and robustness [37], which stems from the underlying trade-offs among the core DFL challenges described above and the challenge of communication efficiency. We discuss these trade-offs individually in Section 6.

Another particularly promising direction is personalized DFL, where each device trains its own model tailored to local data characteristics while still incorporating shared knowledge from the network. This is especially relevant in scenarios with significant data heterogeneity or divergent learning tasks across devices [224, 181, 220, 102]. Personalization is typically introduced in DFL systems by modifying the network topology (see Section 4) or through weightings in the model aggregation method (see Section 3.1).

In this paper, communication efficiency refers to minimizing communication cost while maintaining convergence. We define communication cost as the total amount of information exchanged among the DFL participants, measured by the cumulative size of all transmitted messages across the network. This definition accounts solely for the volume of data communicated between actors during training and explicitly excludes considerations of latency, transmission delays, or other time-dependent factors.

2.3. Related Work

The rapid growth of FL and the increasing number of respective publications gave rise to numerous surveys summarizing the advances. Although the majority of these surveys focus on CFL, significant efforts have also been devoted to reviewing DFL, both in addition to CFL [200, 31, 85, 218] and exclusively [19, 52, 175]. In this context, the authors in [19] analyze DFL in terms of federated architectures, network topologies, communication mechanisms, security and privacy approaches, and key performance indicators. Furthermore, they explore existing optimization mechanisms of DFL and provide a list of trends, lessons learned, and open challenges. In survey [200], the authors elaborate on FL solutions that focus on network topologies for both CFL and DFL by analyzing their advantages and disadvantages, and discuss the remaining challenges and future work for applying FL in specific network topologies. Similarly, the authors of [31] survey CFL and DFL approaches and classify the existing work based on their network topology. They discuss the general techniques behind FL from the perspectives of theory and application, and frame the current application problems—also providing the existing attack scenarios and defense methods. In [85], the authors focus on synchronization strategies, consolidation methods, and network topologies, assessing and contrasting the efficacy and constraints of methodologies and providing insights for future progression. In [218], the authors introduce a systematic and detailed perspective on DFL, including iteration order, communication protocols, network topologies, paradigm proposals, and temporal variability. Moreover, they propose categorizations

Survey	Communication-Influencing Components			Communication Optimization		
	Model Aggregation (Section 3.1)	Synchronization Method (Section 3.2)	Network Topology (Section 3.3)	Compression (Section 4.1)	Local Computation (Section 4.2)	Partial Device Participation (Section 4.3)
Beltrán et al. [19]	∼	✓	✓	∼	×	×
Wu et al. [200]	×	✓	✓	×	×	✓
Chen et al. [31]	✓	×	×	✓	×	×
Khan et al. [85]	×	∼	✓	×	×	×
Yuan et al. [218]	✓	∼	∼	∼	×	✓
Gabrielli et al. [52]	×	×	∼	∼	×	✓
Thabet et al. [175]	×	∼	∼	✓	×	∼
Ours	✓	✓	✓	✓	✓	✓

Table 1
Coverage of communication aspects for DFL constituents in existing surveys on DFL; ×: does not mention anything regarding the aspect of communication efficiency; ∼: briefly mentions the aspect of network communication without providing insights; ✓: elaborates on the aspect of communication efficiency and provides insights.

within this context, and they discuss possible solutions and future research directions. The authors in [52] review existing approaches for traditional and blockchain-based DFL, identify emerging challenges, and discuss future research directions. In survey [175], the authors examine DFL approaches that try to optimize performance and efficiency in terms of memory usage, communication cost, convergence under data heterogeneity, and computational effort. In addition, they categorize the approaches based on their method to address system heterogeneity and data heterogeneity, and the authors provide some application scenarios of DFL.

Existing reviews on DFL cover a significant share of the recent approaches from the literature. Analogous as it is the case for DFL approaches, some survey papers focus on certain challenges such as security and privacy [19, 31], data heterogeneity [175], and system heterogeneity [175]. A fraction of surveys also discuss communication aspects for selected methodologies. However, to the best of our knowledge, the current literature does not comprise a review paper that specifically focuses on the communication cost in the DFL system. Table 1 provides an overview of the aspects that are addressed in terms of communication efficiency by the individual survey papers.

3. Communication-Influencing Components

Communication efficiency in FL constitutes an ongoing research challenge. Albeit DFL addresses the communication bottleneck of CFL, there still exists a high demand for further improvements. In particular for systems with bandwidth limitations, exchanging a massive amount of model updates induces a major bottleneck in the DFL infrastructure. Therefore, reducing the communication cost brings substantial advancement in time performance.

In DFL, numerous system components determine the overall communication cost. Therefore, when calculating the communication overhead, we have to consider the question "*What* gets communicated *with whom* and *how often*?". This question is composed of three aspects: *What*, *with whom*, and *how often*. The "*what*" (i.e., the information transfer volume) is mainly influenced by the model aggregation technique (Section 3.1), as it decides the type of exchanged information (e.g., model parameters, gradient, etc.). The network topology (Section 3.3) affects the "*with whom*" (i.e., the network links for exchange) by specifying the neighborhood of an actor. Lastly, the question of "*how often*" (i.e., the communication frequency) is influenced by the synchronization method (Section 3.2), as it determines the system behavior in terms of waiting times and potential interruptions of communication rounds. Given these considerations, we identify the communication-influencing DFL components to be the Model Aggregation (Section 3.1), the Synchronization Method (Section 3.2), and the Network Topology (Section 3.3). In this section, we discuss the communication-influencing components in more detail by categorizing common implementations and summarizing the way and the extent to which they influence communication. The components are depicted in Figure 2 with their respective subcategories. Furthermore, Table 2 links the individual publications from common approaches to the corresponding subcategory of their implementation.

3.1. Model Aggregation

The model aggregation scheme is a central part of DFL, as it defines the method for incorporating received model parameter updates into the training process. As a core part of DFL, it plays an important role in communication efficiency, since it determines the type of information that actors need to exchange. In other words, it specifies the content of a model parameter update. Such an update could, for instance, contain one (or several) complete set(s) of model parameters, a gradient to previous models, or reduced model parameters. Furthermore, it may also contain supplementary information regarding model performance, network connectivity, or system state to enable dynamic adaptation to evolving conditions. As a natural consequence, information transmission is generally classified into two distinct categories: the control plane, which transmits auxiliary information to control the learning process, and the data plane, which transmits the

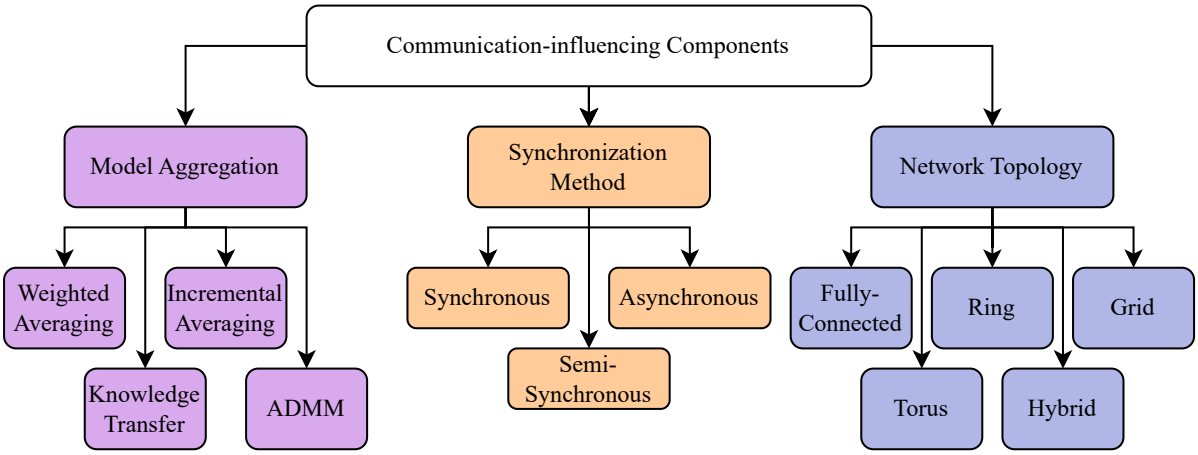


Figure 2: Taxonomy with the communication-influencing essential components of DFL and their prevailing categories.

Reference	Model Aggregation	Synchronization Method	Network Topology
Koloskova et al. [86]	wAvg	Synchronous	Fully-connected, Ring, Torus
Li et al. [98]	KT/KD	Synchronous	Fully-connected
Wang et al. [188]	wAvg	Synchronous	Random
Tang et al. [173]	wAvg	Synchronous	Fully-connected
Wang et al. [184]	wAvg	Synchronous	Fully-connected, Random, Other
Liu et al. [119]	wAvg	Synchronous	N/A
Zhou et al. [236]	ADMM	Semi-synchronous	Random
Wang et al. [190]	wAvg	Synchronous	Fully-connected, Ring
Zong et al. [239]	wAvg	Synchronous	Torus
Liu et al. [115]	wAvg	Asynchronous	Other
Gupta et al. [57]	incAvg	Synchronous	Other
Liao et al. [113]	wAvg	Asynchronous	Random
Soltani et al. [161]	wAvg	Synchronous	Grid, Hybrid

Table 2
Publications discussed in this paper ("Reference") with the corresponding categories of the communication-influencing components: Model Aggregation ("wAvg": weighted averaging; "incAvg": incremental averaging; "ADMM": alternating direction method of multipliers; "KT/KD": knowledge transfer/distillation), Synchronization Method, and Network Topology.

actual model parameter updates. Overall, choosing an appropriate model aggregation scheme is crucial to achieve communication efficiency, as well as to ensure the convergence of the training process.

Based on DFL literature, we focus on four general categories of model aggregation that are prominent in DFL: (1) weighted model averaging (wAvg), (2) incremental model averaging (incAvg), (3) alternating direction method of multipliers (ADMM), and (4) knowledge transfer and distillation. The authors in [218] introduce two learning paradigms that affect model aggregation: `Continual` and `Aggregate`. In `Continual` (also called continual federated learning or incremental federated learning), the actor resumes training directly on top of the previous actor's model parameters. In the `Aggregate` paradigm, on the other hand, the actor first aggregates several received model updates and then progresses training based on the aggregated model parameters. Since the `Aggregate` class covers a broad spectrum of aggregation techniques, we further categorize the model aggregation methods into sub-classes. In our context, incAvg falls into the `Continual` paradigm while the other three categories belong to the `Aggregate` paradigm.

wAvg The most popular class of aggregation schemes in DFL is the class of wAvg aggregation strategies, which implement an average over the set of received model parameter updates to obtain an aggregated state of parameter update in each round. Within this averaging process, neighbor-specific weights determine the relative influence of the individual contributions received from neighboring actors. Thanks to the ability to aggregate multiple updates, we can parallelize the training of the individual actors. Equation 1 shows the general form of an aggregation step for wAvg schemes in epoch k at actor i. To obtain the aggregated model $m_{k+1,i}$, actor i averages the model parameter updates

$\Delta_{k,j}$ of its neighboring actors $j \in \mathcal{N}_i$, weighted by their respective mixing weight $W_{i,j}$. Note that with this notation, the set of neighboring actors $\mathcal{N}_i$ also contains the actor $i \in \mathcal{N}_i$ and that the corresponding mixing weights sum up to $\sum_{j \in \mathcal{N}_i} W_{i,j} = 1$. The model parameter update $\Delta_{k,j}$ refers to actor j's locally updated model parameters, for instance, by performing a local epoch of stochastic gradient descent (SGD). The simplicity of wAvg schemes and their suitability for parameter aggregation led to the development of numerous variants of wAvg-based aggregation schemes in DFL.

$$m_{k+1,i} = \sum_{j \in \mathcal{N}_i} (W_{i,j} \Delta_{k,j}) \tag{1}$$

The wAvg-based variants for DFL differ in three aspects: (1) the mixing weights $W_{i,j}$, (2) the set of neighbors' model updates considered for aggregation $\mathcal{N}_i^* \subseteq \mathcal{N}_i$, and (3) the content of a transmitted model update, consisting of the locally updated parameters $\Delta_{k,j}$ and potential supplementary information. We summarize the individual variants of common approaches regarding these three aspects in Table 3. Supplementary information in addition to the main content of a model update might include connectivity information, system configurations, training statistics, or other relevant information, which is transmitted either once for initialization or in every communication round, as listed in Table 4.

The **mixing weights** determine the impact of the individual model updates in the aggregation. The simplest strategy is to specify the mixing weights equally among all actors, giving each actor the same magnitude of impact [86, 173, 161]. A slight variation of this is to assign an individual weight to the aggregating actor while uniformly distributing the weights of the neighboring actors, thus adjusting the influence of the local model update separately [188]. Another strategy sets the mixing weights of the individual actors proportional to their respective data cardinality, prioritizing the model updates from actors with more data [239]. The choice of mixing weights does not directly influence communication in DFL; however, strategies that accelerate convergence, for example by setting the mixing weights according to network properties [119, 190] or dynamically adjusting them during training to capture time-varying connectivity [113] or model-specific behavior [184, 115], may reduce the number of required communication rounds.

The **selection of a subset of model updates** $\mathcal{N}_i^*$ after they have been received from the neighbors (i.e., model update selection) is rarely implemented in DFL, as it only brings a small reduction in computational effort and usually the contribution of all neighbors is valuable to some extent. However, the authors in [115] maintain a cache of model parameter updates from previous epochs and include these updates in the aggregation. They employ a model update selection technique based on reinforcement learning to prevent the consideration of stale updates. Most DFL approaches, however, leverage the optimization technique of partial device participation (Section 4.3) to reduce the amount of model updates involved in training. In contrast to the model update selection, partial device participation determines the subset of neighboring actors before the transmission step, avoiding the redundant transmission of not considered model updates and, thus, reducing communication cost. Note that both the selection of model updates and the technique of partial device participation can also be expressed in terms of the mixing weights by setting the respective weight to 0. Since the selection of model updates occurs after they have been received, this does not directly affect the communication efficiency.

The **content of a model update**, on the other hand, influences the communication cost in DFL directly, as it determines the size of the transmitted message. The simplest and most prominent approach is the transmission of the model parameters [188, 173, 184, 190, 239, 115, 161]. Other approaches exchange differential model parameters (i.e., model deltas) among the neighbors that are based on a common reference [86, 119], resulting in a generally smaller value range for the elements and, hence, yielding either a reduction in the communicated size or an increase in precision at the same communication cost. In [113], the authors include a scalar weight (i.e., the push-sum weight [137]) in addition to the model parameters with each model update, which is then leveraged to de-bias the local model. Beyond the model update, they communicate the training loss and the local iteration index to all available neighbors—regardless of the neighbors selected by partial device participation (see Section 4.3)—to support the neighbor selection in their implementation of partial device participation. The content of the model update directly influences the communication cost, since it determines the size of the communicated model update. Note that the communication between actors is not necessarily limited to model parameter updates, but can entail additional communication. Here, it is important to distinguish between information that is only communicated in the initialization phase [188, 184, 239] (see Table 4a) or in every communication round [173, 190, 113, 161] (see Table 4b). Also note that in the approaches of references [188, 173, 190], the additional information is communicated between the actors and a lightweight coordinator[3] to support their advanced training strategies from a central perspective. Since this coordinator serves as a shared, central communication hub, it challenges the notion of full decentralization. Consequently, there is an ongoing debate regarding whether such approaches belong to DFL. Divergent perspectives largely stem from the intended objectives of the use-case. When the primary goal is to keep the model updates undisclosed to any central party (privacy), it is generally acceptable. However, when robustness against partial outages (system heterogeneity) is prioritized, these approaches exhibit the same limitations as CFL. In this work, we proceed on the assumption that these approaches are DFL, as claimed in the respective publications. However, we expressly refrain from taking a position in this debate.

[3] A lightweight coordinator is a central node that does not perform model aggregation.

Reference	Mixing Weights	Update Selection	Update Content
Koloskova et al. [86]	Equal	None	Estimated model delta
Wang et al. [188]	Equal (others)	None (PDP)	Model parameters
Tang et al. [173]	Equal (both)	None (PDP)	Model parameters
Wang et al. [184]	Trust-score (dynamic)	None	Model parameters
Liu et al. [119]	Network topology	None	Model delta
Wang et al. [190]	Degree (global)	None (PDP)	Model parameters
Zong et al. [239]	Data size	None	Model parameters
Liu et al. [115]	Staleness (dynamic)	RL-based (cached)	Model parameters
Liao et al. [113]	Degree	None (PDP)	Model parameters, scalar
Soltani et al. [161]	Equal	None (PDP)	Model parameters

Table 3
Strategies of three fundamental aspects of wAvg-based model aggregation: Basis for the specification of Mixing Weights ("both": a total of two model updates; "others": different for the aggregating actor; "dynamic": dynamic re-weighting in every round; "global": consideration on a global scope instead of the neighborhood), technique to perform Update Selection ("PDP": implementing *partial device participation* (see Section 4.3) to select a subset of neighboring actors to consider; "cached": selection of updates from a cache; RL: reinforcement learning), and the Update Content that is exchanged between the actors (model delta: difference in model parameters compared to a common reference).

Reference	Dir.	Additional Communication
Wang et al. [188]	⇀	Matching decompositions, seed
Wang et al. [184]	⇌	Data size, data variance, connectivity
Zong et al. [239]	⇌	Data size

(a) Once for initialization.

Reference	Dir.	Additional Communication
Tang et al. [173]	⇀	Seed, mixing weights matrix, round index
	↼	Bandwidth, loss, accuracy
Wang et al. [190]	⇀	Set of neighbors, compression ratio
	↼	Consensus distance, communication capacity
Liao et al. [113]	⇌	Training loss, local iteration index
Soltani et al. [161]	⇌	Parameters of last layer

(b) Every communication round.

Table 4
Communicated information in addition to the model update content from Table 3; "Dir." indicates the direction of information transmission (⇀ from coordinator to actors; ↼ from actors to coordinator; ⇌ among actors).

incAvg For the case of incAvg aggregation, the model parameters are passed through the network sequentially—similar to a random walk—, with each actor locally optimizing the received parameters before forwarding the enhanced model parameters to the next actor. During training, various states of the global model parameters are available in the network, since the individual actors only receive the global state when it is their turn to perform the local optimization step. The general formula for such a local optimization step on received model parameters m_p using SGD at actor i is given in Equation 2, where η is the learning rate, $f_i(\cdot,\cdot)$ is the loss function of actor i, x_i represents the private data on actor i, and ∇ is the differential operator.

When compared to DFL with wAvg, the incAvg strategy presents a completely different scenario, as it lacks computational parallelism due to its sequential nature. It is related to the field of continual learning and faces similar challenges (e.g., catastrophic forgetting) [91]. Despite increased execution times due to the lack of computational parallelism, incAvg aggregation drastically reduces the communication cost, as only one actor transmits the model parameters to one other actor in each round. In this context, Gupta et al. [57] propose a novel strategy for selecting the path through the network to achieve the best possible accuracy with the least possible communication cost. More precisely, they consider all possible paths within a distance threshold to the shortest path in the network and select the path that offers the highest training accuracy. To do so, the connectivity and bandwidths of the entire network must be known. In their experiments, the authors show a significant reduction in communication cost while achieving similar accuracy as their baselines. In general, incremental training of the collaborative model is a powerful method to significantly reduce communication cost, but it is not suitable for time-critical applications.

$$m_i = m_p - \eta \nabla f_i(m_p, x_i) \tag{2}$$

ADMM First introduced in the 1970s [51], ADMM is an algorithm to solve distributed convex optimization problems, combining the benefits of dual decomposition and Lagrangian methods [24]. Given its suitability for decentralized optimization, ADMM was adopted by DFL. The iterative updates for ADMM (see Appendix A.1.1) rely only on local information from the respective actor and the model updates from its neighbors. Hence, the exchange of solely the model updates is sufficient for optimizing with ADMM. In [236], the authors adopt ADMM to DFL in combination with techniques of compression (Section 4.1), local computation (Section 4.2), and partial device participation (Section 4.3).

To reduce the computational burden of ADMM, they implement its variant of the inexact alternating direction method of multipliers (iADM) [58]. With regard to communication, the per-round cost of ADMM is equivalent to wAvg. This results from the fact that the local solutions m_i^k, which are exchanged with neighboring actors, have the same dimension as the model parameters. Nevertheless, ADMM can solve non-smooth optimization problems effectively and offers a high degree of scalability.

Knowledge Transfer and Knowledge Distillation Knowledge distillation (also referred to as model distillation) is widely used as an effective technique for transferring knowledge from a large, powerful neural network (teacher model) to a smaller network (student model) by mimicking the teacher's behavior [25, 70]. Based on this idea of knowledge distillation, Zhang et al. [231] propose a strategy that enables multiple students with comparable models to learn to solve the task concurrently and collaboratively, which is called mutual learning. This strategy eliminates the teacher role from classical knowledge distillation. In [98], the authors adopt the idea of mutual learning into DFL to address the problem of client drift[4]. Their approach (named DefKT) distinguishes two roles for the participating devices: The sender, which transmits updates of its local model to its assigned receiver, and the receiver, which incorporates these updates by means of knowledge transfer. The update equations are provided in Appendix A.1.2. DefKT employs three different quantities of iterations: (1) The global number of epochs (i.e., communication rounds), (2) the number of local updating steps, and (3) the amount of training passes for the knowledge transfer process, where each training pass iterates through all data batches of the respective dataset at the actor. In terms of communication overhead, DefKT restricts the transmission of model updates to a group of senders and an equal-sized group of receivers in each communication round, both form subsets of the entire community of devices. Consequently, they drastically limit the number of activated communication links. As this also relates to the optimization technique of partial device participation, we further elaborate on the limitation of activated communication links in Section 4.3. The content of a model update in DefKT comprises the full set of model parameters. Therefore, the communicated size per model update is comparable to the general case of the other model aggregation strategies. Nevertheless, the technique of knowledge transfer and the additional configurable parameters in DefKT (e.g., the number of training passes of the knowledge transfer process) offer further possibilities to increase communication efficiency by speeding up the convergence.

With all of the model aggregation categories, the overall communication cost is indirectly influenced by the convergence rate. Depending on the exact optimization problem and the dataset, certain model aggregation strategies increase the convergence rate, thus reducing the number of communication rounds required to reach a certain accuracy. Since the number of communication rounds determines how often we perform the exchange of model updates, it constitutes a major driver for the overall communication cost. Apart from this, the model aggregation schemes also exhibit a direct effect on the communication cost, as discussed in previous paragraphs. Important to note here is that while the model aggregation class (wAvg, incAvg, ADMM, or Knowledge Transfer) defines the general framework for the aggregation strategy and, therefore, determines its baseline communication cost, specific implementations within a class may introduce additional communication overhead. For example, a wAvg scheme that optimizes the mixing weights based on statistics about the local training steps of the actors increases the communication cost by including these statistics in the content of the transmitted model update. To summarize, we have described the state-of-the-art categories of model aggregation schemes to provide insight into the general framework in which their implementations operate, and we have discussed their general implications for communication efficiency, outlining key differences in communication between the respective aggregation types.

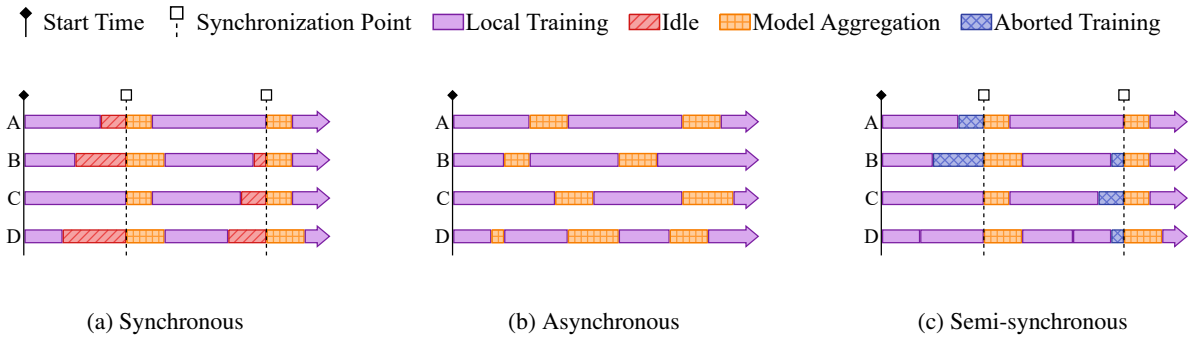
3.2. Synchronization Method

The synchronization method determines the behavior of the actors when exchanging model updates. Essentially, it defines whether the actors wait for the model updates from the other actors or continue the training process directly after sharing their model updates. Since the respective synchronization method is an instrumental aspect of the DFL system, it affects both the performance of the overall learning process and the general operability of other techniques in the DFL system. Furthermore, different synchronization methods entail different challenges, such as the challenge of stale model updates and the challenge of stragglers. In terms of communication cost, the synchronization method strongly influences the number of communicated model updates.

Following existing literature (see e.g., [19, 200, 85]), we distinguish between the categories of synchronous, asynchronous, and semi-synchronous DFL, as they provide a solid separation regarding their main differences. We elaborate on the characteristics of these three categories, put them in contrast, and discuss their impact on the communication cost. An exemplary execution timeline of four actors in the DFL system is shown in Figure 3 to visualize and compare the respective behavior of the three synchronization categories.

In **synchronous** DFL, all actors synchronize with each other directly after they perform local training. As a result, the model updates that are exchanged among the actors all have the same level of progress, which simplifies the aggregation process. It is the most popular category among the synchronization methods [86, 98, 188, 173, 184, 119, 190, 239, 57, 161]. However, heterogeneous actors cause idle times in the learning process, since all actors have to

[4] Client drift denotes the phenomenon when collaborating participants train in different directions due to data heterogeneity.



(a) Synchronous (b) Asynchronous (c) Semi-synchronous

Figure 3: Exemplary execution timeline on four actors (A/B/C/D) for the three types of synchronization methods. (3a) involves idle times, (3c) includes aborted trainings, and (3b) shows different aggregation times that vary with the number of model updates considered. Model aggregation times vary across actors due heterogeneous system characteristics (e.g., computing resources, bandwidth) and across communication rounds due to differences in model update properties (e.g., sparsity, precision).

wait for the slowest actor in the system (i.e., the problem of stragglers). As indicated in Figure 3a, faster actors wait for all other actors to complete local training (purple bar) in an idle state (red bar), before continuing with the exchange and aggregation of model updates (orange bar). The point in time when all actors have completed the local training is called the synchronization point. In synchronous DFL, the communication is limited to a single exchange of model updates per communication round, which occurs at the synchronization point. As the actors wait for each other, they perform only one local training in each communication round, and, hence, they share only a single model update with their neighbors. Note that even if we perform multiple local epochs in the local training step using the optimization technique of local computation (see Section 4.2), we obtain only one overall model update that covers these local epochs. Thus, each actor is strictly limited to sending only one model update to its neighbors in each communication round.

In contrast to synchronous DFL, actors in **asynchronous** DFL broadcast their model updates directly after finishing the local training and continue with the subsequent aggregation process. Hence, this model aggregation process does only involve neighbors with available model updates at that time. To enable this asynchronous communication, every actor permanently listens for incoming model updates and stores them in a buffer until it performs the model aggregation. The asynchronous scheme addresses the problem of stragglers encountered in the synchronous scheme, yet it introduces the challenge of supporting varying, arbitrary numbers of model updates among the neighboring actors. Additionally, it introduces the challenge of stale model updates which adversely affect the optimization. Figure 3b depicts the execution timeline for the asynchronous scheme, clearly showing the elimination of idle times that occur in the synchronous case. Note that we encode an overall increase in the duration of model aggregation in Figure 3b compared to synchronous DFL, which is due to its higher complexity and the additional check for staleness of the model updates. The actors in asynchronous DFL directly continue with the consecutive local training step and the transmission of the successive model update after performing the model aggregation. Model updates are therefore communicated at a higher frequency among the actors than it is the case with the synchronous scheme (unless the overhead in the duration for model aggregation exceeds the idle times from synchronous DFL). As a result, faster actors share more model updates than slower actors within the same time, causing asynchronous DFL to involve more communication than synchronous DFL. Note, however, that asynchronous communication has the advantage of sending model updates at different times, which prevents the actor from being flooded by a massive amount of updates at once. Among the publications discussed in the present work, two approaches employ the asynchronous synchronization method [115, 113]. Similar to the synchronous scheme, the concept of the asynchronous scheme is also strictly defined. Nevertheless, differences in the approaches exist in their handling of surrounding challenges, such as the increase in complexity of convergence and the problem of staleness. In this context, the authors of [113] provide a convergence analysis of their approach under the heterogeneous scenario, assuming strongly convex functions. Liu et al. [115], on the other hand, propose a dynamic, staleness-aware model aggregation method to address the problem of stale model updates, which is further discussed in Section 3.1.

The **semi-synchronous** method combines characteristics from the synchronous and the asynchronous scheme. It involves synchronization points on which all actors synchronize with each other, similar to synchronous DFL. In between two synchronization points, the actors start the next local iteration directly after completing a local training round, similar to the asynchronous scheme. Hence, there are no idle times with the semi-synchronous method. The key characteristic of semi-synchronous DFL is that the actors abort the local training step when reaching a synchronization point. Aborted trainings are colored in blue in Figure 3c. Although aborting the local training step wastes computational effort, the benefit behind this concept becomes apparent when multiple local training steps can be completed within

one synchronization period (i.e., the time between two synchronization points). As shown in Figure 3c, actor D successfully performs multiple local training steps in-between synchronization points, leveraging available computing resources that would be idle in synchronous DFL. Variations of the semi-synchronous method differ in the determination of synchronization points. In Figure 3c, the synchronization points are determined by the time required by the slowest actor to complete one local training. However, the synchronization points can also depend, for instance, on a pre-specified duration, on the fastest actor, or on more sophisticated evaluations among the actors (e.g., a metric of divergence). Note the risk of confusing the semi-synchronous scheme with the synchronous scheme when the technique of local computation (see Section 4.2) with multiple local updates is involved. The main difference is that the semi-synchronous scheme describes the parallel execution of all participating actors, whereas local computation determines the number of local updates performed before the model parameters are aggregated with model updates from other actors—applicable with all synchronization methods. The semi-synchronous scheme exhibits similarities to both the synchronous and the asynchronous scheme. In terms of communication, the frequency of transmitted model updates almost resembles the asynchronous method, since there are no waiting times. The difference lies in the aborted training steps, which do not yield a model update. Whether semi-synchronous synchronization exhibits a higher or lower communication frequency than asynchronous DFL depends on both the heterogeneity of computing resources between the actors and the time required for the model aggregation step. The bandwidth bottleneck that arises in synchronous DFL due to receiving multiple updates all at once also exists in the semi-synchronous scheme. This is due to the common synchronization points, which lead to actors starting the training round simultaneously. As an implication, actors are completing the round and communicating the updates concurrently, given that they have similar computing power. However, this is not the case with heterogeneous computing resources at the actors. Semi-synchronous DFL particularly addresses the challenge of stragglers, since slow actors do not prevent the other actors from continuing their computations. Additionally, the semi-synchronous method benefits from joint synchronization points, reducing the complexity of convergence. In [236], the authors propose a variant of the semi-synchronous scheme that combines key characteristics from both synchronous and asynchronous DFL, called partial synchronization. Similarly to synchronous DFL, their variant comprises waiting times. However, the actors only wait for a pre-specified minimum amount of model updates. Slower actors are considered stragglers in that particular synchronization round and, hence, are not included in the model aggregation. As a result, the actors follow an asynchronous behavior throughout the synchronization periods. Due to the presence of partial synchronization points and asynchronous behavior, the method of partial synchronization belongs to the semi-synchronous synchronization methods.

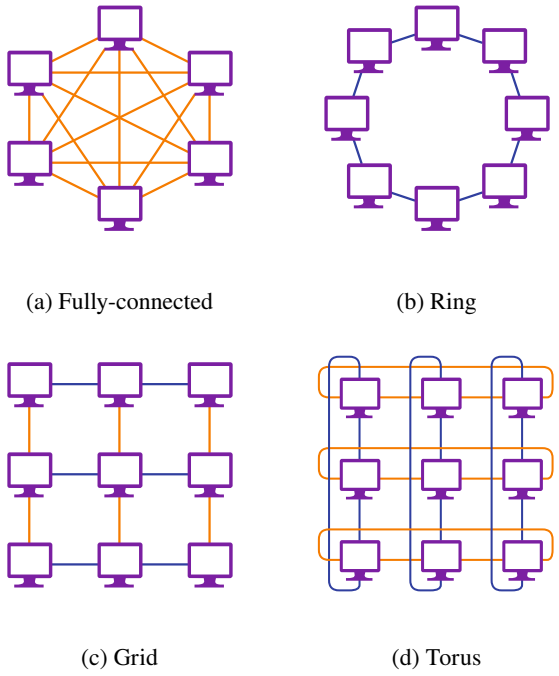
To summarize, each of the three synchronization methods has considerable advantages but also entails some drawbacks. The synchronous synchronization method creates a more predictable environment regarding convergence of the overall learning process. Synchronous communication, however, does not address the problem of stragglers in the DFL system and is more susceptible to bandwidth bottlenecks. The asynchronous synchronization methods, on the other hand, address the challenge of stragglers by eliminating waiting times for the other actors. Instead of waiting for the others' model updates, an actor incorporates only available model updates and then directly continues its execution, which drastically increases the complexity of theoretical convergence and, thus, reduces the predictability of the overall learning process. Compared to synchronous DFL, the asynchronous method increases the amount of model updates transmitted between the actors while lowering the bandwidth bottleneck that arises from receiving multiple updates at the same time. The semi-synchronous synchronization method tries to combine the advantages of both the synchronous scheme and the asynchronous scheme by allowing for asynchronous behavior within a framework that enforces synchronization points to maintain the simplicity of model aggregation and convergence. This leads to trade-offs between the benefits and drawbacks, since underlying concepts are mutually exclusive. Therefore, variants of the semi-synchronous scheme try to optimize these trade-offs.

3.3. Network Topology

As mentioned in Section 2.2, inter-node communication is a characterizing part of DFL. To facilitate communication in the first place, we require network connectivity between the actors. However, we cannot assume that all nodes are interconnected because of connectivity limitations (e.g., absence of network links between some actors) and contractual restrictions (e.g., no data contract[5] established between two parties). In this context, the network topology defines which actors are connected through a uni-directional or bi-directional network link. Similar to topologies in network science, we represent network topologies as adjacency matrices and visualize them as graphs. The most popular types of network topologies considered in DFL comprise the fully-connected mesh topology, the ring topology, the grid topology, and the torus topology, depicted in Figure 4. Due to the strict definition of these topologies and the necessity to consider arbitrary network structures in DFL, these network topologies are often combined to form hybrid variants.

In FL, we distinguish between two types of network connectivity: (1) The consistent connectivity arising from connectivity limitations and contractual limitations, which remains the same during the whole training process, and (2) the inconsistent (or variable) connectivity that changes over time due to node outages and the optimization technique of partial device participation (Section 4.3). Note that in this section, we cover the consistent connectivity, referred to

[5] A data contract defines the terms for exchanging data (in our case model parameters) between two parties.



(a) Fully-connected (b) Ring

(c) Grid (d) Torus

Figure 4: Illustration of popular network topologies considered in DFL publications. In the Fully-connected topology (Figure 4a), all actors are interconnected; in the Ring topology (Figure 4b), each actor is connected to two neighboring actors, forming a ring; in the Grid topology (Figure 4c), connections form an evenly-spaced two-dimensional grid; and in the Torus topology (Figure 4d), the grid topology is extended by connecting the actors beyond the grid boundary to the other side.

as network topology[6]. Inconsistent connectivity is regarded with partial device participation in Section 4.3, given their common characteristic that actors do not receive model updates from all of their neighboring actors and the similar change in connection links. Although the active links are not selected by the partial device participation strategy but determined by inconsistent connectivity, the reception of model updates is similar in both cases.

Fully-connected Topology The fully-connected topology—sometimes also referred to as complete graph topology—assumes that all actors are connected to each other actor in the network. Thus, with a total number of n actors, each actor maintains $n-1$ bi-directional network links (see Figure 4 (a)). Works on DFL consider a fully-connected network topology [86, 173, 184, 190] either as the basis for their implementation or in their experimental setup. An important point to note is that a fully-connected network topology does not necessarily imply that every actor communicates with all other actors, since the variable connectivity (i.e., inconsistent connectivity) may further restrict the connectivity. In this context, the authors in [173] and in [190] employ techniques from partial device participation (Section 4.3) in their experiments, thus reducing the communication to a subset of selected actors. Without further restrictions on the connectivity, the model updates are transmitted from each actor to all other actors, resulting in the maximal possible amount of $n(n-1)$ transmissions per communication round. Also worth mentioning is that in addition to considering the fully-connected mesh topology in their experiments, the authors in [86] perform experiments with the ring topology and the torus topology, the authors in [184] conduct experiments with the chain topology and the random topology, and the authors in [190] include the ring topology in their experiments. Thereby, they provide a comparison of fully-connected mesh topologies in DFL with other topologies.

Ring Topology In a ring topology, each actor is connected to exactly two neighboring actors, forming a closed chain of connected nodes with a single continuous path through the network (see Figure 4 (b)). Ring topologies emerge in DFL to reduce communication overhead, since the communication cost of DFL with a ring topology grows linearly with the number of participants. More specifically, a newly added actor places itself between two actors in the ring, resulting in one additional network link over which model parameters are exchanged. Thus, the number of model update transmissions in each communication round is given by $2n$. Koloskova et al. [86] perform experiments to compare the convergence of the fully-connected mesh topology, the ring topology, and the torus topology with $n \in \{9, 25, 64\}$ actors. Thereby, they show that such connectivity restrictions as in the ring topology cause only a mild negative impact on the convergence while reducing the order of communication growth. Furthermore, in [190], the authors consider two different methods for constructing the ring network, (1) randomly connecting the nodes to a ring topology or (2) constructing a consensus-aware ring greedily to maximize the consensus-distance between neighbors. To analyze the impact of the network topology, they additionally compare the performances of the ring topology and the fully-connected mesh topology in their experiments. In [161], the authors evaluate their proposed method using a hybrid topology called Ring of Cliques, in which 4 fully-connected cliques of 4 actors each are connected in a ring structure (16 actors in total).

Grid Topology Another characteristic network constellation is the grid topology. Here, the actors form an evenly-spaced two-dimensional grid connecting to their respective neighbors (see Figure 4 (c)). Hence, inner nodes of the grid connect to 4 adjacent nodes, while edge nodes connect to only 3 nodes and corner nodes connect to 2 neighbors. Consequently, actors in a DFL infrastructure with grid topology exchange their model parameter updates with up to 4 neighbors. In a square grid constellation, for instance, one communication round comprises $4(n-\sqrt{n})$ transmissions of model updates. However, only a few DFL approaches consider such (artificially constructed) grid topologies due

[6] In this paper, the term 'network topology' refers exclusively to consistent connectivity and, hence, does not encompass inconsistent connectivity, even though this is sometimes referred to as 'variable network topology' in the literature.

to their rarity in real-world deployments. Soltani et al. [161], for instance, perform experiments using the grid topology as well as the hybrid Ring of Cliques topology, with 16 actors each. When training with the grid topology, their results indicate slightly faster convergence in terms of communication traffic than for the Ring of Cliques. The difference, however, is so small that it does not allow for any general claims in this regard.

Torus Topology The torus topology is an extension of the grid topology in which the edge nodes additionally connect to the corresponding nodes on the opposite edge (see Figure 4 (d)). As a result, all nodes have a degree of 4 (i.e., connect to 4 nodes). The torus topology is sometimes also referred to as a 2D ring, since connecting an edge node to the opposite edge forms a continuous path (i.e., ring) through the respective row or column of the grid. Since each actor is connected to exactly 4 neighboring actors, there are $4n$ model update transmissions in each communication round. In the experiments of [86], the comparison of performance on the fully-connected mesh topology, the ring topology, and the torus topology demonstrates only mild performance differences between these topologies, with the torus topology falling between the other two in terms of performance. Beyond employing the torus topology for experiments, Zong et al. [239] introduce a 2D-attention-based device placement algorithm to minimize the overall communication cost. Here, the algorithm arranges the actors within the torus schema intending to minimize both the transmission time and the communication latency.

Other and Hybrid Topologies Besides these popular network topologies, also other topologies like the chain topology (i.e., ring topology with one link removed) [184], the exponential graph topology (i.e., every actor is connected to $\lceil \log_2 n \rceil$ actors) [115], or a random graph (i.e., connections are randomly constructed, for instance with the Erdős–Rényi random graph model) [188, 184, 236, 113] are considered in the literature. Moreover, network topologies are sometimes combined into hybrid topologies, such as the "Ring of Cliques" topology in reference [161], which arranges so-called cliques (i.e., fully-connected groups of actors) in a ring topology. In general, hybrid topologies can combine arbitrary network topologies.

We have seen that network topologies highly influence the communication cost of a DFL system, as they determine which actors can communicate with each other. Thus, network topologies restrict the number of possible model update transmissions. Among the topologies discussed, communication is most restricted by the ring topology, followed by the grid topology and then the torus topology. The fully-connected topology does not restrict communication, as it allows every actor to communicate with all other actors. The majority of DFL approaches consider fully-connected mesh topologies and random graph topologies (see Table 2). However, it is important to note that various strategies exist for constructing random graph topologies. Therefore, a direct comparison of the "popularity" of random graph topologies with that of strictly defined topologies is not meaningful.

Koloskova et al. [86] perform experiments with the fully-connected, torus, and ring topology with different numbers of devices. The fully-connected topology achieves the lowest training sub-optimality among the network topologies. Their experiments also demonstrate that, while the training sub-optimality for the torus topology is comparable to the fully-connected topology for a small number of devices, it approaches the results from the ring topology when increasing the number of devices. Tang et al. [173] compare the fully-connected topology to the ring topology, showing that the ring topology converges slower in terms of training rounds. However, note that deploying the ring topology does not require more communication, as the ring topology sends fewer model updates in each round.

4. Communication Optimization Techniques

In addition to the essential components of DFL infrastructures discussed in Section 3, various optimization techniques can be applied to further enhance the communication efficiency. While these techniques are not strictly indispensable for a DFL system, they are of great utility for reducing the communication cost. We identified three major categories of communication optimization techniques in DFL, which we discuss in the following subsections: Compression (Section 4.1), Local Computation (Section 4.2), and Partial Device Participation (Section 4.3). For each of these categories, we outline its origins, identify prominent classes, explain the general concept, and discuss the impact on communication cost with emerging challenges and trade-offs. Figure 5 illustrates the taxonomy and Table 5 links the underlying classes to common approaches from recent publications.

Many of the following techniques originated in CFL or in distributed ML, and have been adopted to DFL. However, since DFL features a different setting than CFL, these methods require adaptation to match the changes in system characteristics.

4.1. Compression

Model compression is a well-known concept for reducing the communication cost in FL. In CFL, the client applies lossy compression to the model parameter updates before sending them to the central server, and the server responds with a compressed version of the aggregated model parameters. In DFL, on the other hand, the actors compress their model parameter updates before transmitting it to their neighboring actors. While the loss of information involved in this compression creates a general trade-off between compression ratio and training performance, the scattered nature of communication links in DFL introduces additional challenges in terms of maintaining convergence. Therefore, it is highly important to take the system environment (e.g., network topology, synchronization method, etc.) into account for implementing compression in DFL.

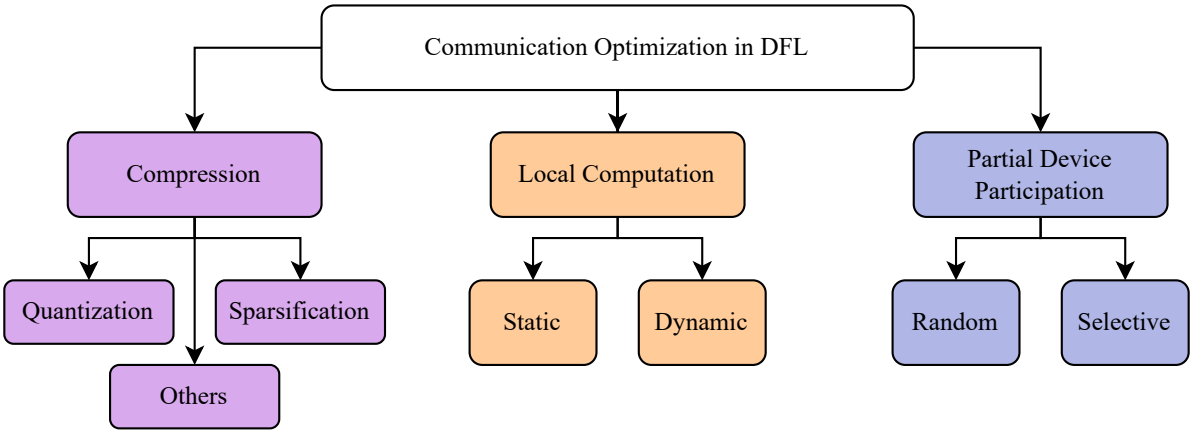


Figure 5: Taxonomy with the most prominent techniques to optimize the communication of DFL and their key categories.

Reference	Compression	Local Computation	Partial Device Participation
Koloskova et al. [86]	Quantization, Sparsification	None	Random (not in experiments)
Li et al. [98]	None	1, 10 Updates	Random 20%
Wang et al. [188]	None	Implicitly (PDP)	Selective (connectivity)
Tang et al. [173]	Sparsification	None	Selective (bandwidth), single
Wang et al. [184]	Sparsification	5 Updates	None
Liu et al. [119]	Quantization	4 Updates	None
Zhou et al. [236]	1BCS	Random in [5, 15]	Random 20%, 50%, 80%
Wang et al. [190]	Sparsification	50 Updates	Selective (consensus)
Zong et al. [239]	None	None	None
Liu et al. [115]	Sparsification	Implicitly (PDP)	Random, single
Gupta et al. [57]	None	1, 2, 5, 10 Updates	None
Liao et al. [113]	None	2 Updates	Selective (loss)
Soltani et al. [161]	None	5 Updates	Selective (similarity), single

Table 5
Publications discussed in this paper ("Reference") with the corresponding implementation categories of the communication optimization techniques: Compression ("1BCS": 1-bit compressed sensing [23]), Local Computation ("Implicitly (PDP)": the technique of Partial Device Participation causes devices to perform multiple consecutive local updates), and Partial Device Participation ("Random": participating devices are randomly selected;"Selective (XX)": participating devices are selected based on XX; "Single" / "PP%": only one / PP% neighboring devices participate in every round).

In the literature, two types of compression techniques dominate for the use with DFL: Quantization [86, 119] and sparsification [184, 173, 190, 115]. Both types belong to the class of lossy compression techniques, which use inexact approximations to compress data. Note that lossless compression techniques are generally not suitable for DFL as they require an expected structure in the data, which is not the case with model parameters or gradients. With lossy compression techniques, it is crucial to consider the impact on the model accuracy caused by the loss of information due to the entailed approximations. However, with a proper balance between model performance and system efficiency, it is possible to significantly reduce the size of the model parameters while maintaining reasonable accuracy [42].

Quantization In literature on DFL approaches, we encounter various quantization schemes. In general, quantization compresses data by reducing the number of bits required to represent the data. A simple but popular quantization scheme is randomized quantization. Implemented (among others) in reference [86], randomized quantization divides the data range into a set of evenly distributed quantization levels (i.e., bins) and then randomly rounds the data values to the adjacent lower or upper bin. Consequently, we reduce the data size—and thus reduce communication cost—by placing the bins on values that can be represented by a smaller quantity of bits. In addition to uniform quantization methods like randomized quantization, DFL also adopts non-uniform approaches. In [119], for instance, the authors apply the Lloyd-Max quantization algorithm and adjust the quantization levels adaptively to minimize the involved distortion. Thereby, they accounts for time-varying convergence rates

and variable gradient distributions while providing convergence guarantees without convex loss assumptions. Besides reducing the data size, the authors in [61] incorporate quantization to achieve differential privacy guarantees, jointly addressing the challenge of communication efficiency and privacy concerns.

Sparsification Sparsification in distributed training (i.e., the exchange of sparse updates) has shown great potential to reduce communication cost without drastic consequences for the convergence [1]. Similarly, the concept of sparsification gains tremendous popularity in DFL to decrease the amount of transmitted parameters. Basically, sparsification generates a sparse representation of a matrix by removing (i.e., zeroing) a considerable number of entries. However, various strategies exist for sparsification in DFL, trying to optimize the trade-off between communication efficiency and information loss. In this regard, the authors in [190] implement the method of random sparsification (RandomK), that preserves k randomly selected entries, and optimizes it by determining actor-specific compression ratios. The semi-decentralized FL framework GossipFL [173], on the contrary, generates a random sparsification mask at every client based on a common seed received from the lightweight coordinator. Therefore, all model parameter updates communicated among the actors exhibit the same sparsity structure in each communication round. The sparsity masks are generated by drawing from a Bernoulli distribution with a probability of $\frac{1}{c}$, where c is the compression ratio specifying the fraction of non-zero elements after sparsification. Besides randomly constructing the sparsification mask, a highly popular method is TopK sparsification, which keeps only the k elements with the highest magnitude [164, 1]. In [184], the authors propose an improvement of TopK sparsification in DFL by adding a momentum to the residual. This additional momentum restricts the impact of the current gradient, resulting in a more stable training and improved performance. Another approach to retain convergence rate with sparsification is proposed by the authors in [115]. Their adaptive sparse training method minimizes the impact of sparsification on the loss function and simultaneously considers the imminent effect of pruned parameters on the training process.

Other In addition to quantization and sparsification methods, several other compression techniques exist which qualify for reducing communication cost in DFL. However, quantization and sparsification are the most popular strategies, as they have proven their suitability for optimization tasks. Alongside these two prevalent compression types, low-rank compression is gaining traction. It exploits the observation that over-parameterized models often reside on low-intrinsic-dimensional subspaces Li et al. [97]. In DFL, this translates to approximating model updates with low-rank matrices prior to transmission, substantially reducing communication cost [29, 194]. Also worth mentioning is the DFL approach in [236], which combines sparsification with quantization by using one-bit compressive sensing (1BCS) [23]. Although 1BCS itself belongs to the group of quantization techniques, its input data must be sparse, requiring the employment of sparsification methods. Therefore, the authors in [236] perform model training subject to a sparsity constraint before applying 1BCS to allow the transmission of one-bit information among the actors. Besides DFL approaches that explicitly implement compression techniques [86, 173, 184, 119, 236, 190, 115], compression methods are typically orthogonal to other components of the system and, hence, can be integrated on top of the existing framework.

The experiments of Koloskova et al. [86] demonstrate that TopK sparsification performs slightly better than RandomK sparsification. Liu et al. [119] conduct experiments with different bit sizes for quantization with QSGD [3]. They present the training loss over communicated bits, showing that a smaller bit size outperforms a larger bit size until it fails to converge. Wang et al. [190] vary the compression ratio in their experiment. They report significant communication cost savings with smaller compression ratios for reaching a common target accuracy.

4.2. Local Computation

The concept of local computation (also known as local-updating SGD, intermittent communication, or federated averaging) is a simple yet powerful strategy to enhance communication efficiency in DFL systems. With local computation, multiple local updating steps are performed at the actor (instead of just one) before the updates to the model parameters are exchanged with the neighboring actors and the received updates are aggregated (synchronization step). Consequently, the actors only communicate after several local updates rather than after each update, reducing the communication by a constant factor. However, this intermittent synchronization introduces a trade-off between convergence rate and communication efficiency, as it gives the actors leeway to diverge. Alongside its numerous applications in CFL [101, 103, 191] and its implementation in the Federated Averaging algorithm [134], the local computation strategy shows success in general distributed optimization systems [229, 114, 177]. As a natural consequence, this concept is implemented by many DFL approaches to optimize communication efficiency [98, 184, 119, 236, 190, 57, 113, 161].

The implementations of local computation in distributed ML, CFL, and DFL follow the same concept, since all of them involve individual devices that perform computations locally and exchange updates to collaboratively approach a common goal. What changes between (and also within) these scenarios, however, is the general setting (e.g., connectivity, computing resources). Therefore, we should consider different behavior in terms of convergence, accuracy, and

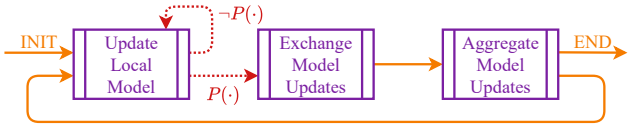


Figure 6: High-level DFL process with *local computation* through the predicate function $P(\cdot)$; Performing consecutive local updates until $P(\cdot)$ evaluates positively.

communication in the respective setting when adopting the concept of local computation.

Applications of local computation can be divided into two distinct categories, depending on whether the quantity of successive local updates is statically fixed or dynamically adapted. With static specification, each actor in the DFL system performs a constant number of τ local iterations per global synchronization. With dynamic adaptation, on the other hand, the point of aggregation is determined dynamically based on criteria such as divergence to a global reference model [81] and computing resources of the individual devices [225]. Figure 6 depicts the high-level DFL process including general local computation through a predicate function $P(\cdot)$ which is represented by the dotted red lines. Here, the result of $P(\cdot)$ indicates whether to perform another round of local updating or to proceed with the exchange and aggregation of model parameter updates (i.e.,, the synchronization step). The construction of $P(\cdot)$ allows to statically specify the amount of local updates τ (e.g., $P(t) : t \mod \tau = 0$, with current epoch t) as well as to define more sophisticated, dynamic synchronization rules. Parameter synchronization in a statically specified interval is often referred to as periodic synchronization.

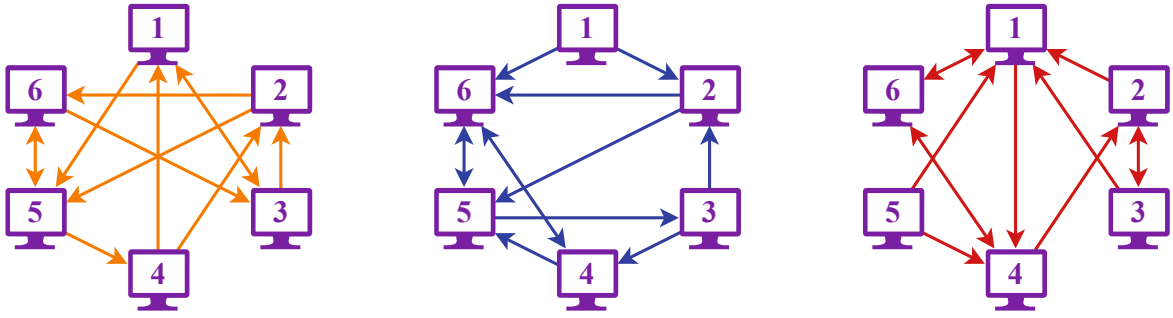
Static In the context of DFL, the majority of approaches that implement local computation define static quantities of local updates. It is important to note, that the proper choice of the amount of local updates highly depends on the respective data properties and the system configuration. To excel the involved trade-off between convergence rate and communication efficiency, we thus have to consider (1) the heterogeneity of data partitions among actors to avoid concept drifts, (2) the computing resources of the individual actors to allow for fairly-balanced contributions, and (3) the general communication scenario (i.e., network topology (Section 3.3), synchronization method (Section 3.2), and partial device participation (Section 4.3)) to reduce the communication overhead while ensuring global convergence. By restricting the synchronization of model parameters to every τ rounds instead of every round, we reduce the communication cost by a factor of $\frac{1}{\tau}$. Note that the efficiency of local computation depends on the inter-play of the amount of local updates τ, the applied batch size, and the respective learning rate, as we can decrease the batch size and the learning rate to allow for an increase in τ. Within the scope of common approaches, the authors in [98, 184, 119, 113, 161] conduct experiments with a static quantity of up to 10 local updates. Furthermore, in [190], the authors perform 50 local training epochs with a batch size of only 32 data samples. In [57], on the other hand, they compare the accuracy with 1, 2, 5, and 10 local updates in their experiments, showing the importance of performing at least 2 local updates to accelerate convergence in their experimental setup. In the experiments of [236], the respective authors allow each actor to perform an individual amount of local training epochs, randomly drawn from $[5, 15]$. However, they do not elaborate on the impact of heterogeneous quantities of local updates.

Dynamic While, to the best of our knowledge, literature does not yet explicitly address dynamic strategies for local computation specifically in DFL (except with the help of a central node [112]), they are successful in distributed ML [81, 225]. Given the strong similarity of DFL to distributed ML in this regard, we can adopt many dynamic approaches for local computation from distributed ML directly into DFL. Nevertheless, two common approaches from recent literature implicitly employ a variable number of local training epochs [188, 115], thus belonging to the category of dynamic local computation. It is important to note that these approaches do not explicitly facilitate multiple local training epochs. Instead, other system components cause the repeated local training of the actors. More specifically, the authors in [188] incorporate a random selection of network links from partial device participation (Section 4.3), which could result in actors not exchanging model updates, eventually leading to consecutive local updates. In reference [115], on the other hand, each actor sends its model parameter update to one random neighbor in each round, which may result in some actors not receiving a model update and, hence, continuing the training on stale model parameters (i.e., performing a consecutive local training epoch).

Gupta et al. [57] perform experiments with varying numbers of local updates. They show that more local training rounds do not add communication cost directly. It speeds up the convergence in the beginning of the training process. As a result, it saves some communication rounds to reach a certain accuracy and, thus, reduces communication cost indirectly. However, Gupta et al. [57] also show that more local updates do not necessarily translate to a better final model.

4.3. Partial Device Participation

Partial Device Participation in FL has already been implemented in 2016 when the Federated Averaging algorithm was introduced [134]. The goal of partial device participation, also known as client selection and neighbor selection, is to reduce the communication overhead by limiting communication of each actor to only a subset of neighboring actors in our DFL system. Thereby, this technique further restricts the network connectivity in the DFL system, since it only allows the transmission of model parameter updates over a fraction of network links. However, in contrast to the time-consistent network topology (Section 3.3), the connectivity between actors in partial device participation changes throughout the learning procedure (i.e., in every epoch).



(a) Epoch 1 (b) Epoch 2 (c) Epoch 3

Figure 7: Exemplary Partial Device Participation with random selection of 40% (i.e., 2/5) neighboring actors over three epochs (Figure 7a, Figure 7b, and Figure 7c); Drawing a new subset of activated links in each epoch.

Relying on the concept of information spread in gossip protocols, consensus between the actors can be achieved under basic assumptions [72, 203].

With partial device participation, we can reduce the communication in DFL by a constant factor. An exemplary scenario of three epochs of learning in a fully-connected network of 6 actors with random selection of 40% of neighbors is depicted in Figure 7. Without partial device participation, every actor would transmit 5 model parameter updates in each epoch, resulting in (6 actors $\cdot$ 5 transmissions) $=$ 30 transmissions per epoch. In contrast, with a participation ratio of 40%, this number reduces to (6 actors $\cdot$ (40% $\cdot$ 5 transmissions)) $=$ 12 transmissions.

Random partial device participation emerges as a simple but powerful strategy to reduce communication in DFL while maintaining reasonable convergence. Therefore, many DFL approaches adopt this random selection strategy in their system, employing a variety of participation rates [86, 236, 115]. Given the changed setting in approaches that leverage Mutual Knowledge Transfer for model aggregation, the authors in [98] apply random partial device participation to select both the senders and the receivers of model parameter updates in the system. Besides random selection of neighbors, more sophisticated strategies exist which try to further reduce communication overhead. To achieve this, these strategies focus on minimizing the participation ratio to communicate with fewer neighbors in each round while preserving convergence and selecting proper neighbors to increase convergence rate and thus reduce the number of epochs required to converge.

The authors in [173] and in [161] limit the communication to only a single neighbor per actor in each epoch. Moreover, in [173], they consider the respective inter-node bandwidth when selecting the neighbor but still allow some randomness in the selection process in order to facilitate convergence. Without this level of randomness, their strategy would always select the connection with the highest bandwidth, resulting in a permanent change of network topology in the case of consistent bandwidths. In [161], on the other hand, the authors select the most informative neighbor for the parameter transmission. More specifically, they pick the neighbor with the currently most divergent model parameters, measured by the cosine distance between the last layers of the model. By doing so, however, we require additional communication to retrieve the last layer of the model parameters from all neighbors. Therefore, the advantage of this approach regarding communication reduction highly depends on the size of the model and on the number of potential neighbors.

The loss-based strategy from Liao et al. [113] also requires additional communication, as the actors transmit their training loss (single value) to all connected actors directly after the local updating step. Based on these loss values and the communication frequencies to the individual connected actors, each actor then calculates priorities to select the set of activated neighbors in each epoch. Note that they embed a limit for the communication frequency in the priority calculations such that the priority of selecting a specific actor reaches its maximal value after a certain period without communication.

Unlike the selection of neighboring actors from the perspective of each individual actor (actor-centric selection), the authors in [188] and in [190] activate a subset of bi-directional links (network-centric selection). To achieve this, Wang et al. [188] decompose the base network topology into matchings (i.e., sets of disjoint links). They then assign an activation probability to each matching and optimize these probabilities to maximize the algebraic connectivity. Thereby, they prioritize connectivity-critical links, which eventually enhances the trade-off between convergence and communication. Similarly, in [190], the authors construct an active network topology on top of the consistent network topology in each epoch, trying to minimize the round time. To achieve this, they calculate the time required by each existing network link to remove slow network links iteratively from the base network topology. Simultaneously, they apply

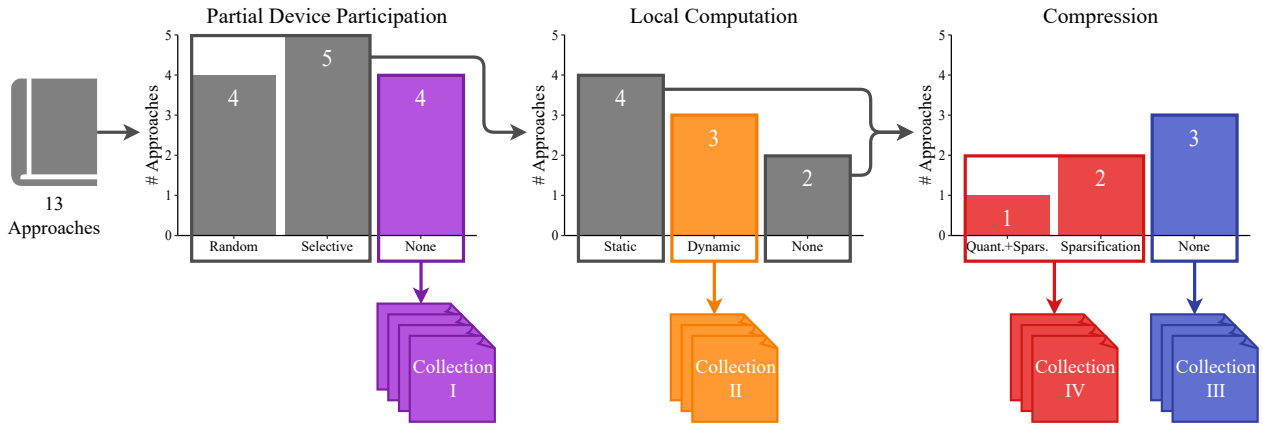


Figure 8: Illustration of our extraction process for assigning the publications from the DFL literature to four coherent collections. The first collection extracts approaches that do not implement partial device participation; the second collection takes out the approaches that employ a dynamic strategy for local computation; and the third and fourth collections differentiate between whether or not they implement compression.

Collection	Commonality	Publications
I	Full device participation (Section 5.1)	Wang et al. [184]; Liu et al. [119]; Zong et al. [239]; Gupta et al. [57]
II	Dynamic number of local updates (Section 5.2)	Wang et al. [188]; Zhou et al. [236]; Liu et al. [115]
III	Perfect Parameter Transmission (Section 5.3)	Li et al. [98]; Liao et al. [113]; Soltani et al. [161]
IV	Sparsification for compression (Section 5.4)	Koloskova et al. [86]; Tang et al. [173]; Wang et al. [190]

Table 6
Grouping of publications into four distinct collections according to commonalities in implemented optimization techniques.

constraints based on the consensus distance to guarantee convergence, and adjust the individual compression ratios at the actors to comply with these constraints.

Zhou et al. [236] experiment with varying fractions of participating devices. They report that a higher participation rate induces higher data transmission volume but fewer iterations to reach a fixed target accuracy. Zhou et al. [236] present the data transmission volume side by side to the number of iterations and the target accuracy in tables over different participation rates, offering a well-organized overview of the impact of partial device participation.

5. Communication Efficiency of DFL Approaches

In this section, we apply the taxonomies from Section 3 and Section 4 to our selected publications. Thereby, we showcase the usability of the taxonomies and take a holistic look at the implementations of the underlying techniques in common DFL approaches. Additionally, we highlight the main objectives of the approaches, briefly summarize their experiments, and review their findings.

Unfortunately, a direct comparison of the experimental results is precluded by substantial diversity across datasets, models, implementations, evaluation metrics, and experimental configurations. A valid comparison would require the development of a unified DFL framework that re-implements all of the methods considered and evaluates them using the consistent metrics within the identical experimental environment. This effort would advance the field significantly. The design and provision of a reusable framework for experiments on DFL methods deserves their own publication and is therefore beyond the scope of the current work.

To avoid addressing each publication individually, we divide the 13 DFL publications discussed in this paper into four collections based on our taxonomy of communication optimization techniques. More specifically, we look at the distribution of the respective strategies to extract coherent sets from the 13 publications. The extraction process is

Key Novelty	MomTopK Sparsification	LM-DFL Adaptive Lloyd-Max Quantization	Fedcs Parameter Synchronization	TravellingFL Aggregation
Reference	Wang et al. [184]	Liu et al. [119]	Zong et al. [239]	Gupta et al. [57]
Framework	Custom using PyTorch	Custom using PyTorch	N/A	Custom simulator w/ Docker and Sockets
Dataset	N-BaIoT [135]	MNIST [93], CIFAR-10 [88]	Randomly generated	CIFAR-10 [88]
Data Distribution	Non-IID	Non-IID	IID	IID
Model	Custom feed-forward NN	Custom CNN	N/A	VGG-16, VGG-19 [160], ResNet-50 [63]
Baselines	FedAvg [134], TopK [1] and RandomK [172] sparsification	No quantization, ALQ [48], QSGD [3]	Gossip [72] w/ fully-connected and torus topology; several device placement algorithms	FedAvg [134], FedProx [151], FedCAMS [195]
# Actors	5	10	16, 25, 64, 100	7, 10, 20
Evaluation Aspect	Communication cost, robustness to adversaries	Training loss, communication cost, quantization distortion	Optimization rate, communication overhead	Training and test accuracy, communication cost
Communication Metric	Communication cost w.r.t. convergence rate	Communicated bits over time	Communication time overhead	Test accuracy over communication rounds
Reported Improvement	Communication reduction of 60% compared to FedAvg	9.7%/28.6% (MNIST/CIFAR-10) improved training loss w/ same communication cost than 8-bit QSGD	Communication improvement increases with dimension of topology	Average reduction of communication in time by 23.3%
Main Comparability Limitation	Model and dataset	Model	Framework, model, and communication metric	Framework

Table 7
Overview showing the key novelty, the experimental setting, the primary communication improvement, and the main limitation for comparing with others of the DFL publications in Collection I.

illustrated in Figure 8. First, we consider the strategies from partial device participation, extracting the approaches that do not implement partial device participation (i.e., each actor involves all neighboring actors in the model exchange phase) to form Collection I. Next, we consider the strategies for local computation in approaches that implement partial device participation, and construct Collection II of approaches with dynamic numbers of local updates. Finally, the distribution of compression strategies naturally divides the remaining approaches into Collection III without compression and Collection IV in which the approaches apply sparsification. Table 6 lists the four collections including the publications they contain and their uniting commonality. By discussing the publications in these collections throughout the following subsections, we reveal typical combinations of the collection feature with other communication-influencing methods. By contrasting the experimental conditions from the individual publications in a collection side-by-side (i.e., Table 7, Table 8, Table 9, Table 10), we reveal the main limitation preventing their direct comparability. This further underlines the necessity of a unified framework for DFL experiments.

5.1. Collection I: Full Device Participation

Collection I comprises references [184, 119, 239, 57], which, in addition to implementing full device participation, all employ the synchronous *synchronization method* (Section 3.2). The combination of full device participation and a synchronous DFL system offers the advantage of relying on exactly one update from each neighboring actor in every communication round. In contrast, actors in asynchronous DFL must support varying numbers of received updates anyway, thus promoting the implementation of PDP. In [184, 119, 239], they follow the wAvg method for *model aggregation* (Section 3.1), whereas in [57], the authors follow the incAvg method. These four publications differ in their *network topology* (see Section 3.3), *compression* method (see Section 4.1), and the number of local updates for *local computation* (see Section 4.2), covering a large portion of possible combinations with different strategies from these techniques. Wang et al. [184] perform 5 local updates, apply their novel sparsification method (TopK sparsification with momentum), and conduct experiments under several network topologies (fully-connected, random graph, and chain topology). In [119], the authors also implement a

novel compression method, namely the Lloyd-Max quantizer with adaptive quantization levels. In their experiments, they perform 4 local updates, but we were unable to discover from the publication whether they use a random graph or a fully-connected network topology. In contrast to these two approaches, Gupta et al. [57] do not implement a compression method to reduce the size of the communicated model updates. Their primary innovation lies in selecting the best path through the network for the incAvg aggregation method, optimizing both accuracy and communication cost. They also implement *local computation*, considering the set of 1, 2, 5, and 10 local updates under the directed graph network topology. In [239], the authors do not facilitate any of the communication optimization techniques discussed in Section 4. However, they are optimizing the communication overhead by arranging the network topology into a torus topology. Within this framework, they employ a two-layer circular parameter synchronization strategy to reduce the overall frequency of parameter synchronization.

Table 7 provides an overview of the experimental settings and the main results on communication efficiency in the publications of Collection I. Wang et al. [184] report a 60% reduction in communication cost and state that their approach converges the fastest among the baselines that utilize sparsification. In their experiments, MomTopK sparsification converges twice as fast as the approach with RandomK sparsification [172], while TopK sparsification [1] struggles to converge. In the results of Wang et al. [184] and Liu et al. [119], DFL without compression shows the best convergence among experiments with the same number of iterations. Nevertheless, LM-DFL [119] outperforms all other baselines in both the number of iterations and time progression. Zong et al. [239] additionally evaluate the reduction in communication overhead of their device placement algorithm in an ablation study under various network sizes, obtaining optimization rates between 40% and 83% compared to a random placement. Gupta et al. [57] further examine the effect of different local training epochs (i.e., *local computation* in Section 4.2), the scalability with different amounts of actors, their selection of the optimal path through the network, and the performance of different model architectures, and report the results in their publication [57].

5.2. Collection II: Dynamic Number of Local Updates

The publications in Collection II employ a dynamic number of local updates for *local computation* (see Section 4.2), either by randomly selecting this quantity from a prespecified range [236] or by indirectly allowing for consecutive local updates due to other techniques [188, 115]. In [188] and in [115], the authors employ a wAvg method for *model aggregation* (see Section 3.1, whereas in [236], they utilize the inexact alternating direction method of multipliers (iADM). These three publications cover all three types of *synchronization methods* (see Section 3.2), underlining the compatibility of dynamic *local computation* in synchronous [188], asynchronous [115], and semi-synchronous [236] DFL systems. The authors in both [236] and [115] implement a random selection of devices for *partial device participation* (see Section 4.3), alongside the *compression* technique (see Section 4.1) of compressed sensing or sparsification, respectively. In contrast, Wang et al. [188] do not involve *compression*. Their novelty lies in their matching-based selection strategy (MATCHA) for *partial device participation*, in which they decompose the network topology into a set of matchings[7], select a set of matchings that are activated, and optimize the selection to ensure fast convergence and minimize the communication delay. Activating only a subset of matchings opens up the possibility that some actors do not connect to a neighboring actor. In this case, the respective actors do not share their model update but continue to train for a consecutive local epoch, thus facilitating the technique of *local computation* implicitly. Similarly, Liu et al. [115] employ a random selection of one neighboring actor for transmitting the model update. Hence, actors that do not receive a model update from their neighbors perform another updating round without additional input, i.e., *local computation*. Note that even though the authors in [115] call the iterations of this updating round "local updates", we do not consider this as an implementation of *local computation* as it involves the aggregation of model updates from neighboring actors, thus not being strictly local. In [236], they draw a random number between 10 and 15 in each round to determine the number of local updates in one of their experiments, thus directly implementing a dynamic number of local updates.

Table 8 presents the experimental settings and the key improvements in the publications of Collection II. A distinctive characteristic of MATCHA [188] is that the communication budget is specified as a fraction of the iteration-wise communication cost in vanilla decentralized SGD. Wang et al. [188] also demonstrate that MATCHA can be used as a tool to improve other decentralized computation tasks, reporting a 2× speedup in wall-clock time when implementing MATCHA on top of ChocoSGD [86]. Zhou et al. [236] conduct an ablation study of their DFL approach (CEPS) to assess its behavior under different number of nodes, values of the differential privacy parameter ε, and participation rates for *partial device participation*, showing that a higher participation rate leads to increased computational time, which is compensated by a decreased number of iterations needed to converge. Moreover, they demonstrate that increasing the participation rate leads to fewer iterations but a higher data transmission volume. The experiments of Liu et al. [115] partition the dataset into heterogeneous partitions using the Dirichlet distribution [100], as opposed to the homogeneous data partitions used in the experiments of Wang et al. [188] and Zhou et al. [236].

[7] A matching is a subgraph of the original network with a degree of one (i.e., each actor is connected to at most one neighboring actor).

Key Novelty	MATCHA Participant Selection (PDP)	CEPS 1-Bit Compressed Sensing	AEDFL Dynamic Aggregation and Selection
Reference	Wang et al. [188]	Zhou et al. [236]	Liu et al. [115]
Framework	Custom using PyTorch	MATLAB	Custom in Python
Dataset	CIFAR-10, CIFAR-100 [88], Penn Treebank corpus [132]	Randomly generated, Qsar oral toxicity [9], real-sim [27]	EMNIST [38], CIFAR-10 [88], Tiny-ImageNet [92], IMDb
Data Distribution	IID	IID	Non-IID
Model	ResNet-50 [63]	Linear and logistic regression	LeNet-5 [93], ResNet-8 [63], VGG-9 [160], TextCNN [237]
Baselines	Vanilla DecenSGD [107, 186], P-DecenSGD [177, 186], ChocoSGD [86]	D-PSGD [107], DFedAvgM [166], and DFedSAM [158]	FedAvg [134], FedProx [104], FedNova [187], AD-PSGD [108], and 10 more
# Actors	8	32, 64, 128	100
Evaluation Aspect	Training loss, communication cost	Training loss, iterations, communication cost	Training accuracy, time, computation cost
Communication Metric	Communication cost specified as hyperparameter	Communication rounds, data transmission volume	Time under bandwidth restrictions
Reported Improvement	50% communication cost compared to vanilla DecenSGD	Saving more than 90% compared to all baselines	No communication gain reported
Main Comparability Limitation	Communication metric	Framework, model, and dataset	Framework, reported improvement

Table 8
Overview showing the key novelty, the experimental setting, the primary communication improvement, and the main limitation for comparing with others of the DFL publications in Collection II.

Key Novelty	Def-KT Aggregation	AsyDFL Synchronization	DFLStar Participant Selection (PDP)
Reference	Li et al. [98]	Liao et al. [113]	Soltani et al. [161]
Framework	N/A	Custom using PyTorch	Custom using PyTorch
Dataset	MNIST [93], Fashion-MNIST [202], CIFAR-10 [88], CIFAR-100 [88]	EMNIST [38] and ImageNet-100 [148]	CIFAR-10 [88], CIFAR-100 [88]
Data Distribution	IID and Non-IID	Non-IID	Non-IID
Model	Custom feed-forward NN and CNN	AlexNet [89], VGG16 [160]	Custom CNN
Baselines	FullAvg [65, 147, 154], Combo [72]	AsyNG [33], AGP [8], OSGP [7], NetMax [235]	D-PSGD [107], PASGD [186]
# Actors	10, 50, 100	30	16
Evaluation Aspect	Training accuracy, communication cost	Training loss, Test accuracy, time, communication cost	Test accuracy, time, computation cost
Communication Metric	Accuracy over communication rounds	Data transmission volume to achieve target accuracy	Data transmission volume
Reported Improvement	No communication improvement reported explicitly	54%/71% less data transmission volume than AGP/OSGP	Communication cost reduction of 95.9%/81.6% compared to D-PSGD/PASGD
Main Comparability Limitation	Framework, reported improvement	/	Model

Table 9
Overview showing the key novelty, the experimental setting, the primary communication improvement, and the main limitation for comparing with others of the DFL publications in Collection III.

5.3. Collection III: Perfect Parameter Transmission

Collection III comprises the subset of three approaches from the unassigned publications that do not compress the parameters in the model update before sending them over the network, thus employing perfect parameter transmission [98, 113, 161]. Instead of optimizing communication cost through *compression*, all three approaches implement *partial device participation* (see Section 4.3) and static *local computation* (see Section 4.2) to enhance communication efficiency. Li et al. [98] select 20% of the actors to participate

in training and conduct experiments with 1 and 10 local updates at the actors. In [113], the authors introduce a loss-based strategy for *partial device participation* and perform two consecutive local updates in their experiments. Soltani et al. [161] also propose a novel technique for *partial device participation*. In contrast to [113], each actor selects only a single neighboring actor to participate in the parameter exchange based on the last layer model similarity. Their experiments involve a fixed quantity of five local updates per communication round. All three approaches of Collection III differ in their network topology. In [98], the authors consider the popular fully-connected topology. In [113], on the other hand, they employ a random, directed network topology with a minimum amount of incoming neighbors for each device, and in [161], the authors perform their experiments using both the grid topology and the hybrid topology called Ring of Cliques. When it comes to the *model aggregation* method (see Section 3.1), the authors in [113] and in [161] implement the popular type of wAvg. Conversely, Li et al. [98] propose model fusion by transferring learned knowledge between devices to address the problem of data heterogeneity. Similarly outstanding is the *synchronization method* (see Section 3.2) by Liao et al. [113], as they propose an asynchronous DFL system while the authors in [98] and in [161] implement the synchronous protocol.

The experiments of the publications in Collection III are summarized in Table 9. Li et al. [98] do not report improvements in communication cost explicitly. Instead, they configure all approaches to have the same communication overhead per round and evaluate the impact of different numbers of actors in the network, the effect of various amounts of local updates, and the differences in training caused by heterogeneous partitioning. The experiments show that Def-KT [98] outperforms all baselines. The experiments by Liao et al. [113] reveal the superiority of asynchronous approaches in general, demonstrating that the synchronous baseline OSGP requires almost twice as much time to reach the target accuracy than AsyDFL and other baselines. Soltani et al. [161] report decreases in training time of 78% and 55.8% in addition to the communication cost reductions of 95.9% and 81.6% for D-PSGD and PASGD, respectively.

5.4. Collection IV: Sparsification for Model Compression

The remaining three common approaches constitute Collection IV, finding their commonality in employing sparsification for *compression* (see Section 4.1) before sending the model update over the network [86, 173, 190]. Since these three approaches focus on advancing communication optimization techniques, they maintain simplicity with regard to other DFL components by implementing the popular wAvg strategy for *model aggregation* (see Section 3.1) with the synchronous *synchronization method* (see Section 3.2). In addition to experimenting with a fully-connected *network topology* (see Section 3.3) in all three publications, Koloskova et al. [86] consider the ring topology and the torus topology, and Wang et al. [190] consider the ring topology. Only one of the publications in Collection IV, namely the approach in [190], utilizes *local computation* (see Section 4.2) with 50 local updates per round. The key contributions by Koloskova et al. [86] are a novel gossip-based stochastic gradient descent algorithm and a novel gossip algorithm with support for arbitrary compression under linear convergence. They consider the technique of *partial device participation* (see Section 4.3) in the form of an exemplary compression operator that randomly removes model updates with a given probability. However, they do not include this operator in their experiments. In [173], the authors propose a novel gossip matrix generation algorithm to optimize bandwidth utilization, thus introducing a new technique for *partial device participation*. Here, each actor communicates with only one neighboring actor, transmitting a model update that is compressed using a global sparsification mask. Employing the same communication optimization categories, Wang et al. [190] tackle the challenge of slow convergence rate due to fixed communication topologies and static compression ratios. More specifically, they propose an algorithm to optimize the selection for *partial device participation* and to determine the compression ratios dynamically for each actor.

We provide an overview of the experiments of the publications in Collection IV in Table 10. The experiments of Tang et al. [173] and Wang et al. [190] both include CHOCO-SGD [86] as a baseline. The experiments of Tang et al. [173] show that both GossipFL [173] and CHOCO-SGD preserve the convergence performance on homogeneously partitioned data at high compression ratios. They conduct experiments in two settings: real-world bandwidths with 14 actors and random bandwidths with 32 or 100 actors. In contrast, Wang et al. [190] deploy 20 actors that are connected with different, fluctuating bandwidths to simulate LAN and WAN connections. CHOCO-SGD emerges as the most communication-efficient approach among all evaluated methods, yet converging about 3× slower to a lower accuracy than CoCo [190].

6. Discussion

Based on the established taxonomy of key DFL components critical to communication efficiency, we analyzed the impact on communication efficiency within the general framework of primary strategies (see Section 3). It becomes apparent that the respective network topology exerts the most direct effect on communication cost by determining the connectivity between actors. In contrast, individual model aggregation strategies and synchronization methods show a strong indirect influence on communication cost through the convergence rate. We further categorized popular techniques to optimize communication in DFL and examined their impact on communication efficiency in Section 4. Our findings highlight the versatility of compression and local computation techniques, given their orthogonality to other DFL constituents and their excellent tuning capability. To

Key Novelty	CHOCO Aggregation	GossipFL Sparsification	CoCo Participant Selection and Quantization
Reference	Koloskova et al. [86]	Tang et al. [173]	Wang et al. [190]
Framework	N/A	FedML	Custom using PyTorch
Dataset	epsilon [163], rcv1 [95]	MNIST [93], CIFAR-10 [88]	CIFAR-10 [88], CIFAR-100 [88]
Data Distribution	IID and Non-IID	IID and Non-IID	IID and Non-IID
Model	Logistic regression	Custom CNN, ResNet-20 [63]	VGG-9 [160] and ResNet-9 [63]
Baselines	Decentralized SGD, DCD-SGD [171], ECD-SGD [171]	FedAvg [87], S-FedAvg [134], D-PSGD [107], DCD-PSGD [171], and CHOCO-SGD [86]	DCD-PSGD [171], CHOCO-SGD [86], SASP [172]
# Actors	9, 25, 64	14, 32, 100	20
Evaluation Aspect	Training error, communication cost	Validation and test accuracy, communication cost, bandwidth utilization	Test accuracy, time, communication cost
Communication Metric	Communication rounds, data transmission volume	Communication time, data transmission volume	Data transmission volume
Reported Improvement	100×/15× less communication than exact communication/4-bit QSGD	14×/16×/154× communication cost reduction to reach target accuracy compared to FedAvg/CHOCO-SGD/DCD-PSGD	Communication cost reduction of 50% on average
Main Comparability Limitation	Framework, dataset	Framework	/

Table 10
Overview showing the key novelty, the experimental setting, the primary communication improvement, and the main limitation for comparing with others of the DFL publications in Collection IV.

identify methodological combinations addressed in existing research, Section 5 explored the interplay of strategies from our taxonomies in concrete realizations. In addition to providing a detailed review, this analysis highlights underexplored research directions. For instance, integrating compression methods with knowledge transfer for model aggregation and full device participation presents a promising combination for future work. Furthermore, methods for dynamically determining the number of local updates for local computation are currently under-researched, suggesting an avenue for developing sophisticated strategies to enhance communication efficiency in DFL.

Our summary of the experimental findings of DFL approaches elucidates the overarching impact of their implemented strategies on both communication efficiency and performance. However, there is no universal approach to reducing communication cost in a DFL system, given the dependence on the overall system configuration, such as data properties, model properties, and system components, as well as the numerous trade-offs between the challenge of communication efficiency and other challenges. To clarify these trade-offs, we discuss the effect of strategies to address the core challenges in DFL (as listed in Section 2.2) on communication efficiency in the following paragraphs.

Security and privacy × communication efficiency In DFL, additional measures are often integrated to address security and privacy risks that the distributed and decentralized design alone cannot mitigate (see Section 2.2). Those methods can influence communication efficiency in various ways. Fault-tolerant DFL systems that are robust to node failures or dropouts often require more connections to provide redundancy. Cryptographic methods for node verification or blockchain-based systems require the exchange of additional information such as cryptographic proofs or transaction metadata. Depending on the method, detecting poisoning attacks may also necessitate additional communication, although some approaches eliminate this need [49]. Integrating privacy-enhancing technologies into DFL systems can also significantly influence communication efficiency, as the pursuit of stronger privacy guarantees usually involves increased communication overhead or a reduction in overall system performance. Under multi-party computation (MPC), each node splits its update into secret shares and distributes them among a set of peers, which substantially increases communication overhead as multiple messages have to be transmitted. However, more communication-efficient variants have been proposed [82]. Differential privacy (DP) is most commonly applied locally: Each node adds carefully calibrated noise to its model updates before sharing them with peers, thus ensuring that individual data points cannot be inferred from the updates. While this does not change the size of the model updates, it can lead to slower convergence which, in turn, results in a higher number of training rounds to achieve acceptable accuracy. This effect can be reduced by using notions of DP that leverage the properties of DFL to amplify privacy guarantees [39, 40]. When using homomorphic encryption (HE), each node typically encrypts

its local model updates and aggregation is performed on the encrypted data. Encryption increases the size of model updates, thereby increasing the communication cost. Furthermore, decentralized key management, e.g., multi-key HE [67], requires the exchange of additional information, further adding to communication overhead.

Data heterogeneity × communication efficiency Heterogeneous data distributions between actors can cause local model parameters to converge at different stationary points, thereby hindering global convergence [19]. A straight-forward approach to address this concern is to communicate in a dense network (e.g., construct a fully-connected network topology with full device participation) and to increase the frequency of synchronization among the actors (i.e., lower the number of local updates). This, however, conflicts with the challenge of communication efficiency, causing communication cost to spiral up drastically. Therefore, there exists a general trade-off between communication efficiency and the challenge of data heterogeneity. Work to optimize this trade-off includes the development of specific aggregation methods [113, 179, 74], the design of heterogeneity-aware network topologies [17], and the consideration of data heterogeneity for partial device participation [113, 33, 190]. However, some of these approaches impose tight assumptions on the DFL scenario (e.g., assuming knowledge about the distributions of labels at other actors [17]), or even contradict the DFL paradigm by exchanging data (e.g., sharing synthetic data [74]). Others exchange supplementary information among the actors to prevent their gradients from drifting apart (e.g., sharing the training loss [113, 33] or estimated consensus distances [190] to aid the partial device participation strategy). Such supplementary information is typically in an aggregated state and significantly smaller than a gradient or a full set of model parameters. Hence, exposing aggregated statistics about the current state of the actor to detect gradient drifts emerges as a promising direction to avoid frequent synchronizations when addressing data heterogeneity. The main challenge here is to correctly detect gradient drifts based on the aggregated statistics. This detection method could then be applied to methods such as local computation (see Section 4.2), partial device participation (see Section 4.3), or other DFL constituents to dynamically prevent actors from diverging.

System heterogeneity × communication efficiency The most common approach to overcome the problem of stragglers, arising from differences in computing and communication resources among actors, is to enable asynchronous [113, 115] or semi-synchronous [236] training. However, this leads to an increase in communication overhead compared to synchronous DFL, as discussed in Section 3.2. Moreover, asynchronous and semi-synchronous DFL systems need to account for stale model updates. A standard approach to mitigating the effects of staleness is to either exclude outdated model updates from the aggregation or to incorporate the updates with reduced weights, for example by assigning an advancement-dependent weighting factor in the aggregation rule [115]. This, in turn, leads to considerable communication overhead, as these model updates have been transmitted but are then not used to their full extent. Consequently, a fundamental trade-off arises between the challenges of communication efficiency and the challenge of system heterogeneity. The more we optimize a DFL application for communication efficiency, the more leeway we give to the problem of stragglers—and thus potentially undermine a slow actor. Conversely, if the DFL system focuses on overcoming the problems of system heterogeneity, the transmitted information is not fully exploited, thereby introducing redundant communication overhead.

DFL emerges as a useful tool for various applications, thanks to its great scalability and privacy-preserving characteristic. Among the most popular application scenarios are healthcare [11, 174, 110, 197], Internet of Things (IoT) [96, 109, 129, 139, 144, 145], satellite networks [47, 199, 211, 213, 222], and vehicles [12, 20, 62, 141, 146, 170, 204, 205]. DFL allows healthcare institutions to collaborate on a common optimization problem. Here, the geo-spatial distribution between institutions promotes optimizing communication efficiency to overcome the burden of long communication times. The sensitive nature of data in healthcare applications underlines the necessity to address the privacy challenge [174, 110, 197], in order to provide privacy guarantees and ensure that data remains undisclosed to other parties. Therefore, the trade-off between privacy and communication efficiency emerges as a key aspect to consider in healthcare DFL applications. In addition to DFL among institutions, the growing field of the Internet of Medical Things in healthcare is drawing increasing attention toward DFL on mobile and wearable devices. Due to varying computing and communication resources of these devices, the need arises to optimize the trade-off between the challenge of system heterogeneity and communication efficiency—alongside the privacy trade-off. Similarly, DFL applications in IoT are confronted with a wide variety of devices which show different system characteristics, accentuating the challenge of system heterogeneity [96, 144]. Since these devices also differ in their data acquisition, the challenge of data heterogeneity becomes evident [96, 139, 145]. Given the typically dense connectivity of IoT networks, the DFL system comprises a multitude of communication links. Since extensive communication over a vast number of links would slow down the entire network, communication needs to be optimized—either by communicating only over a fraction of available links or by reducing the size of communicated information. Therefore, the challenge of communication efficiency emerges in IoT applications alongside the two aforementioned challenges, requiring the consideration of the compromise between system heterogeneity and communication efficiency, as well as between data heterogeneity and communication efficiency. Optimizing these trade-offs helps to overcome the problems arising from different devices and

data, while minimizing the communication cost to reduce the delay caused by frequent communication among actors. In contrast to frequent communication, DFL applications in satellite networks face the situation of highly sporadic, irregular connectivity [47] and energy restrictions [199]. Here, reducing communication is crucial to complete the exchange of model updates among actors within a given time frame, as well as to minimize the energy consumption of wireless communication. Due to differences in cameras and sensors, satellite DFL applications also face the challenge of data heterogeneity [199, 213, 222]. The similar case applies to DFL applications in vehicular networks [62], revealing the importance of the trade-off between data heterogeneity and communication efficiency in both application scenarios. Alongside this trade-off, DFL in vehicles faces the challenge of privacy and security, particularly in military applications involving unmanned aerial vehicles [20], highlighting the trade-off between the challenge of privacy and security with the challenge of communication efficiency. In all of these application scenarios, optimizing the DFL application entails addressing multiple challenges simultaneously, as the challenge of communication efficiency affects all of them. In this work, we have elaborated on DFL techniques to address the challenge of communication efficiency. Future research would benefit from continuing this work by analyzing the trade-offs between communication-efficiency methodologies and techniques from other challenges, as in-depth understanding about these trade-offs is crucial for the development of respective DFL methodologies.

7. Open Challenges

Despite significant progress in the field of communication-efficient DFL, several critical challenges remain open. These can be broadly organized into two groups: challenges related to *how communication is optimized*, and challenges related to how such *optimizations are evaluated and deployed*.

On the communication optimization side, three directions stand out. First, theoretical foundations for combined optimization are largely missing. While individual techniques such as compression, partial device participation, and topology optimization are well-studied in isolation, their joint behavior remains poorly understood. In particular, convergence guarantees that account for the compounding effects of simultaneously applied optimizations are needed to predict communication–accuracy trade-offs when combining multiple strategies. Second, adaptive and context-aware communication deserves greater attention. Current approaches predominantly employ static strategies, yet real-world deployments face dynamic conditions including fluctuating bandwidth, varying node availability, and shifting data distributions. Developing principled methods for runtime adaptation of communication parameters—compression rates, aggregation frequencies, and neighborhood structures—remains a largely open problem. Third, heterogeneity-aware communication budgeting is underexplored. The DFL literature increasingly acknowledges system heterogeneity, but few works explicitly optimize communication allocation across nodes with disparate capabilities. Asymmetric schemes in which resource-constrained nodes transmit less frequently or at higher compression rates, while still maintaining convergence guarantees, offer a natural yet insufficiently studied avenue.

On the evaluation and deployment side, three complementary challenges emerge. Scalability analysis beyond current benchmarks is needed: most experimental evaluations consider networks of tens to hundreds of nodes, yet DFL increasingly targets large-scale deployments. Whether current optimization techniques remain effective at scale, and how communication cost grow with realistic network topologies, requires systematic investigation. Communication–privacy co-optimization presents another open frontier. Privacy-preserving mechanisms such as differential privacy, secure aggregation, or encrypted evaluations often introduce substantial communication overhead, and the interplay between privacy guarantees and communication reduction remains insufficiently explored——particularly whether gains in one dimension come at the expense of the other. Finally, standardized evaluation protocols would substantially improve reproducibility and cross-study comparison. Our review reveals considerable heterogeneity in experimental setups; benchmarks that report communication cost in consistent units, account for both upstream and downstream traffic, and reflect realistic network conditions would place the field on firmer empirical footing.

In addition to these open challenges, emerging communication paradigms as 6G and semantic communication are expected to shift the focus from raw model parameter updates to semantic-aware information exchange. Future 6G networks are anticipated to deliver ultra-low latency, high data rates, and native support for edge intelligence. These capabilities enable semantic-aware communication and thereby reduce communication cost drastically while preserving learning utility by exchanging smaller-sized semantic features instead of full model parameter updates [124, 234].

8. Conclusion

This work provides a systematic review on communication efficiency of DFL methods. We introduced taxonomies for both the components that influence communication and the optimization techniques commonly employed in DFL literature. Guided by these taxonomies, we analyzed each class of strategies, examining their direct effects on communication efficiency as well as indirect effects mediated, for example, through convergence behavior. The resulting insights span from the baseline communication impact of general strategy classes to the nuanced trade-offs of specific implementations found in recent work. By revisiting DFL approaches in thematic collections, we highlight prevalent methodological combinations and, conversely, identify underexplored areas ripe for future investigation.

In summary, this work offers a structured analysis of the individual components and optimization techniques that shape communication in DFL, culminating in a review of how these elements combine in contemporary publications. In doing so, we aim to address the absence of a dedicated review of communication efficiency in DFL literature. We hope the taxonomies introduced here provide a shared vocabulary for systematically comparing and combining future approaches in the field.

Acknowledgments

Funding This work was supported by the "PRO'k'RESS" project (grant No. 60155996) and the "QUICHE" project (grant No. 54741739), both funded by the Austrian Research Promotion Agency (FFG). A visit to the ELSA Workshop on Privacy-Preserving Machine Learning was reimbursed by the project European Lighthouse on Safe and Secure AI (ELSA; funded by the European Union; project grant No. 101070617) through the ELSA PhD & Postdoc Mobility Programme.

Employment This work was conducted by all authors as part of their employment at the Know Center Research GmbH. The Know Center Research GmbH is a COMET competence center that is financed by the Austrian Federal Ministry of Innovation, Mobility, and Infrastructure (BMIMI), the Austrian Federal Ministry of Economy, Energy, and Tourism (BMWET), the Province of Styria, the Steirische Wirtschaftsförderungsgesellschaft m.b.H. (SFG), the Vienna business agency, and the Standortagentur Tirol. The COMET programme is managed by the Austrian Research Promotion Agency FFG.

References

[1] Aji, A.F., Heafield, K., 2017. Sparse communication for distributed gradient descent, in: Palmer, M., Hwa, R., Riedel, S. (Eds.), Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, Association for Computational Linguistics, Copenhagen, Denmark. pp. 440–445. doi:10.18653/v1/D17-1045.

[2] Alalyan, F., Bousalem, B., Jaafar, W., Langar, R., 2024. Secure peer-to-peer federated learning for efficient cyberattacks detection in 5g and beyond networks. ICC 2024 - IEEE International Conference on Communications , 1752–1757.

[3] Alistarh, D., Grubic, D., Li, J., Tomioka, R., Vojnovic, M., 2016. Qsgd: Communication-efficient sgd via gradient quantization and encoding, in: Neural Information Processing Systems.

[4] Almanifi, O.R.A., Chow, C.O., Tham, M.L., Chuah, J.H., Kanesan, J., 2023. Communication and computation efficiency in federated learning: A survey. Internet Things 22, 100742.

[5] Asadi, N., İbrahim Bengu, H., Wulfert, L., Wöhrle, H., Kellerer, W., 2025. Gist - optimizing segmentation for decentralized federated learning on tiny devices. Proceedings of the Federated Learning and Edge AI for Privacy and Mobility .

[6] Asheralieva, A., Niyato, D.T., Wei, X., 2025. Dynamic distributed model compression for efficient decentralized federated learning and incentive provisioning in edge computing networks. IEEE Transactions on Mobile Computing 24, 6293–6314.

[7] Assran, M., Loizou, N., Ballas, N., Rabbat, M., 2019. Stochastic gradient push for distributed deep learning, in: Chaudhuri, K., Salakhutdinov, R. (Eds.), Proceedings of the 36th International Conference on Machine Learning, PMLR. pp. 344–353.

[8] Assran, M., Rabbat, M.G., 2018. Asynchronous gradient push. IEEE Transactions on Automatic Control 66, 168–183.

[9] Asuncion, A.U., 2007. Uci machine learning repository, university of california, irvine, school of information and computer sciences.

[10] Attanayaka, D., Porambage, P., Liyanage, M., Ylianttila, M., 2023. Peer-to-peer federated learning based anomaly detection for open radio access networks. ICC 2023 - IEEE International Conference on Communications , 5464–5470.

[11] Baghersalimi, S., Teijeiro, T., Aminifar, A., Alonso, D.A., 2024. Decentralized federated learning for epileptic seizures detection in low-power wearable systems. IEEE Transactions on Mobile Computing 23, 6392–6407.

[12] Barbieri, L., Brambilla, M., Nicoli, M., 2024. A compressed decentralized federated learning frame work for enhanced environmental awareness in v2v networks. 2024 IEEE International Conference on Acoustics, Speech, and Signal Processing Workshops (ICASSPW) , 374–378.

[13] Barbieri, L., Savazzi, S., Nicoli, M., 2023. A layer selection optimizer for communication-efficient decentralized federated deep learning. IEEE Access 11, 22155–22173.

[14] Behera, M.R., Chakraborty, S., 2024. pfedgame - decentralized federated learning using game theory in dynamic topology. 2024 16th International Conference on COMmunication Systems & NETworkS (COMSNETS) , 651–655.

[15] Behjati, M., Alobaidy, H.A.H., Nordin, R., Abdullah, N.F., 2025. Uav-assisted federated learning with hybrid lora p2p/lorawan for sustainable biosphere. Frontiers Commun. Networks 6.

[16] Beilharz, J., Pfitzner, B., Schmid, R., Geppert, P., Arnrich, B., Polze, A., 2021. Implicit model specialization through dag-based decentralized federated learning. Proceedings of the 22nd International Middleware Conference .

[17] Bellet, A., Kermarrec, A.M., Lavoie, E., 2021. D-cliques: Compensating for data heterogeneity with topology in decentralized federated learning. 2022 41st International Symposium on Reliable Distributed Systems (SRDS) , 1–11.

[18] Beltrán, E.T.M., Bovet, G., Pérez, G.M., Celdrán, A.H., 2025a. Nebula - decentralized federated learning for heterogeneous networks. Proceedings of the ACM SIGCOMM 2025 Posters and Demos .

[19] Beltrán, E.T.M., Pérez, M.Q., S'anchez, P.M.S., Bernal, S.L., Bovet, G., Pérez, M.G., P'erez, G.M., Celdr'an, A.H., 2022. Decentralized federated learning: Fundamentals, state of the art, frameworks, trends, and challenges. IEEE Communications Surveys & Tutorials 25, 2983–3013.

[20] Beltrán, E.T.M., Sánchez, P.M.S., Bovet, G., Stiller, B., Pérez, G.M., Celdrán, A.H., 2025b. Flighter: Decentralized federated learning and situational awareness for secure military aerial reconnaissance. IEEE Communications Magazine 63, 136–142.

[21] Benhelal, M.S., Jouaber, B., Afifi, H., Moungla, H., 2025. Communication-efficient multi-level decentralized federated learning for trajectory prediction. GLOBECOM 2025 - 2025 IEEE Global Communications Conference , 3909–3914.

[22] Bonawitz, K., Eichner, H., Grieskamp, W., Huba, D., Ingerman, A., Ivanov, V., Kiddon, C., Konečnỳ, J., Mazzocchi, S., McMahan, B., Van Overveldt, T., Petrou, D., Ramage, D., Roselander, J., 2019. Towards federated learning at scale: System design. Proceedings of Machine Learning and Systems 1, 374–388.

[23] Boufounos, P.T., Baraniuk, R., 2008. 1-bit compressive sensing. 2008 42nd Annual Conference on Information Sciences and Systems , 16–21.

[24] Boyd, S.P., Parikh, N., Chu, E., Peleato, B., Eckstein, J., 2011. Distributed optimization and statistical learning via the alternating direction method of multipliers. Found. Trends Mach. Learn. 3, 1–122.

[25] Bucila, C., Caruana, R., Niculescu-Mizil, A., 2006. Model compression, in: Knowledge Discovery and Data Mining.

[26] Cao, J., Lian, Z., Liu, W., Zhu, Z., Ji, C., 2021. Hadfl: Heterogeneity-aware decentralized federated learning framework. 2021 58th ACM/IEEE Design Automation Conference (DAC) , 1–6.

[27] Chang, C.C., Lin, C.J., 2011. Libsvm: A library for support vector machines. ACM Trans. Intell. Syst. Technol. 2, 27:1–27:27.

[28] Chellapandi, V.P., Upadhyay, A., Hashemi, A., Żak, S.H., 2023. On the convergence of decentralized federated learning under imperfect information sharing. IEEE Control Systems Letters 7, 2982–2987.

[29] Chen, C., He, Y., Li, P., Jia, W., Dong, Y., Yuan, K., 2025. Greedy low-rank gradient compression provably converges for distributed learning, in: 2025 IEEE International Conference on Cloud Computing Technology and Science (CloudCom), pp. 1–8. doi:10.1109/CloudCom67567.2025.11331531.

[30] Chen, C., Liu, J., Tan, H., Li, X., Wang, K.I.K., Li, P., Sakurai, K., Dou, D., 2024. Trustworthy federated learning: privacy, security, and beyond. Knowledge and Information Systems 67, 2321 – 2356.

[31] Chen, H., Wang, H., Long, Q., Jin, D., Li, Y., 2023a. Advancements in federated learning: Models, methods, and privacy. ACM Computing Surveys .

[32] Chen, J., Yuan, Y., 2025. Decentralized personalization for federated medical image segmentation via gossip contrastive mutual learning. IEEE transactions on medical imaging 44, 2768 – 2783.

[33] Chen, M., Xu, Y., Xu, H., Huang, L., 2023b. Enhancing decentralized federated learning for non-iid data on heterogeneous devices. 2023 IEEE 39th International Conference on Data Engineering (ICDE) , 2289–2302.

[34] Chen, Y., Blum, R.S., Sadler, B.M., 2022. Communication efficient federated learning via ordered admm in a fully decentralized setting. 2022 56th Annual Conference on Information Sciences and Systems (CISS) , 96–100.

[35] Cheng, J., Cai, Y., Zhao, J., Yang, Y., Zhang, S., Cao, Y., 2024. A decentralized federated learning model based on global channel selection. 2024 6th International Conference on Frontier Technologies of Information and Computer (ICFTIC) , 1645–1648.

[36] Chetoui, M., Akhloufi, M.A., 2023. Peer-to-peer federated learning for covid-19 detection using transformers. Comput. 12, 106.

[37] Chung, W.C., Lo, C.A., Lin, Y.H., Chen, Z.H., Hung, C.L., 2025. Decentralized federated learning with non-iid data: Challenges, trends, and future opportunities. ACM Computing Surveys 58, 1 – 41.

[38] Cohen, G., Afshar, S., Tapson, J.C., van Schaik, A., 2017. Emnist: Extending mnist to handwritten letters. 2017 International Joint Conference on Neural Networks (IJCNN) , 2921–2926.

[39] Cyffers, E., Bellet, A., 2022. Privacy amplification by decentralization, in: Camps-Valls, G., Ruiz, F.J.R., Valera, I. (Eds.), Proceedings of The 25th International Conference on Artificial Intelligence and Statistics, PMLR. pp. 5334–5353.

[40] Cyffers, E., Even, M., Bellet, A., Massoulié, L., 2022. Muffliato: Peer-to-peer privacy amplification for decentralized optimization and averaging, in: Koyejo, S., Mohamed, S., Agarwal, A., Belgrave, D., Cho, K., Oh, A. (Eds.), Advances in Neural Information Processing Systems, Curran Associates, Inc.. pp. 15889–15902.

[41] Dai, R., Shen, L., He, F., Tian, X., Tao, D., 2022. Dispfl: Towards communication-efficient personalized federated learning via decentralized sparse training, in: International Conference on Machine Learning.

[42] Dantas, P.V., da Silva, W.S., Cordeiro, L.C., Carvalho, C.B., 2024. A comprehensive review of model compression techniques in machine learning. Appl. Intell. 54, 11804–11844.

[43] Du, M., Zheng, H., Gao, M., Feng, X., 2024. Adaptive decentralized federated learning in resource-constrained iot networks. IEEE Internet of Things Journal 11, 10739–10753.

[44] Duchesne, M., Zhang, K., Chamseddine, T., 2024. Multi-confederated learning: Inclusive non-iid data handling with decentralized federated learning. Proceedings of the 39th ACM/SIGAPP Symposium on Applied Computing .

[45] Dwork, C., McSherry, F., Nissim, K., Smith, A.D., 2006. Calibrating noise to sensitivity in private data analysis. J. Priv. Confidentiality 7, 17–51.

[46] Dwork, C., Roth, A., 2014. The algorithmic foundations of differential privacy. Found. Trends Theor. Comput. Sci. 9, 211–407.

[47] Elmahallawy, M., Luo, T., Ibrahem, M.I., 2023. Secure and efficient federated learning in leo constellations using decentralized key generation and on-orbit model aggregation. GLOBECOM 2023 - 2023 IEEE Global Communications Conference , 5727–5732.

[48] Faghri, F., Tabrizian, I., Markov, I., Alistarh, D., Roy, D.M., Ramezani-Kebrya, A., 2020. Adaptive gradient quantization for data-parallel sgd, in: Larochelle, H., Ranzato, M., Hadsell, R., Balcan, M., Lin, H. (Eds.), Advances in Neural Information Processing Systems, Curran Associates, Inc.. pp. 3174–3185.

[49] Fang, M., Zhang, Z., Hairi, Khanduri, P., Liu, J., Lu, S., Liu, Y., Gong, N.Z., 2024. Byzantine-robust decentralized federated learning.

[50] Fang, X., Chen, L., Yin, H., 2025. Qat-dfl: A communication efficient decentralized federated learning with quantization-aware training design. 2025 Seventeenth International Conference on Wireless Communications and Signal Processing (WCSP) , 1–6.

[51] Gabay, D., Mercier, B., 1976. A dual algorithm for the solution of nonlinear variational problems via finite element approximation. Computers & Mathematics With Applications 2, 17–40.

[52] Gabrielli, E., Pica, G., Tolomei, G., 2023. A survey on decentralized federated learning. ArXiv abs/2308.04604.

[53] Geiping, J., Bauermeister, H., Dröge, H., Moeller, M., 2020. Inverting gradients - how easy is it to break privacy in federated learning?, in: Larochelle, H., Ranzato, M., Hadsell, R., Balcan, M., Lin, H. (Eds.), Advances in Neural Information Processing Systems, Curran Associates, Inc.. pp. 16937–16947.

[54] Ghimire, A., Asiri, A., Hildebrand, B., Amsaad, F.H., 2023. Implementation of secure and privacy-aware ai hardware using distributed federated learning. 2023 IEEE 16th Dallas Circuits and Systems Conference (DCAS) , 1–6.

[55] Gong, X., Gong, X., Sun, Y., Mao, S., 2025. Lightweight decentralized federated learning with arbitrary client participation. Proceedings of the Twenty-sixth International Symposium on Theory, Algorithmic Foundations, and Protocol Design for Mobile Networks and Mobile Computing .

[56] Guo, N., Pan, Y., Yang, Y., 2024. Decentralized federated learning with local differential privacy and comprehensive neighbor selection. Proceedings of the 2024 14th International Conference on Communication and Network Security .

[57] Gupta, V., Luqman, A., Chattopadhyay, N., Chattopadhyay, A., Niyato, D.T., 2024. Travellingfl: Communication efficient peer-to-peer federated learning. IEEE Transactions on Vehicular Technology 73, 5005–5019.

[58] Hager, W.W., Zhang, H., 2019. Inexact alternating direction methods of multipliers for separable convex optimization. Computational Optimization and Applications 73, 201 – 235.

[59] Hallaji, E., Razavi-Far, R., Saif, M., Wang, B., Yang, Q., 2024. Decentralized federated learning: A survey on security and privacy. IEEE Transactions on Big Data 10, 194–213.

[60] Hashemi, A., Acharya, A., Das, R., Vikalo, H., Sanghavi, S., Dhillon, I.S., 2021. On the benefits of multiple gossip steps in communication-constrained decentralized federated learning. IEEE Transactions on Parallel and Distributed Systems PP, 1–1.

[61] Hasircioglu, B., Gunduz, D., 2023. Communication efficient private federated learning using dithering. ICASSP 2024 - 2024 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP) , 7575–7579.

[62] He, H., Du, J., Jiang, C., Wang, J., Song, J., 2024. Mobility-aware decentralized federated learning for autonomous underwater vehicles. GLOBECOM 2024 - 2024 IEEE Global Communications Conference , 2347–2352.

[63] He, K., Zhang, X., Ren, S., Sun, J., 2015. Deep residual learning for image recognition. 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR) , 770–778.

[64] He, L., Bian, A., Jaggi, M., 2018. Cola: Decentralized linear learning, in: Neural Information Processing Systems.

[65] Hegedüs, I., Danner, G., Jelasity, M., 2019. Gossip learning as a decentralized alternative to federated learning, in: IFIP International Conference on Distributed Applications and Interoperable Systems.

[66] Herath, K., Mahangade, S., Bagchi, S., 2025. A lightweight reputation-based mechanism for incentivizing cooperation in decentralized federated learning. 2025 55th Annual IEEE/IFIP International Conference on Dependable Systems and Networks Workshops (DSN-W) , 298–304.

[67] Hijazi, N.M., Aloqaily, M., Guizani, M., 2024a. Collaborative iot learning with secure peer-to-peer federated approach. Computer Communications .

[68] Hijazi, N.M., Aloqaily, M., Guizani, M., 2024b. Collaborative iot learning with secure peer-to-peer federated approach. Comput. Commun. 228, 107948.

[69] Hijazi, N.M., Aloqaily, M., Guizani, M., Ouni, B., Karray, F., 2024c. Secure federated learning with fully homomorphic encryption for iot communications. IEEE Internet of Things Journal 11, 4289–4300.

[70] Hinton, G.E., Vinyals, O., Dean, J., 2015. Distilling the knowledge in a neural network. ArXiv abs/1503.02531.

[71] Hitaj, B., Ateniese, G., Pérez-Cruz, F., 2017. Deep models under the gan: Information leakage from collaborative deep learning. Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security .

[72] Hu, C., Jiang, J., Wang, Z., 2019. Decentralized federated learning: A segmented gossip approach. ArXiv abs/1908.07782.

[73] Hu, C., Tei, K., 2025. Adaptive defense mechanisms against dynamic poisoning attacks in decentralized federated learning. 2025 IEEE International Conference on Autonomic Computing and Self-Organizing Systems Companion (ACSOS-C) , 179–181.

[74] Huang, C.Y., Srinivas, K., Zhang, X., Li, X., 2024. Overcoming data and model heterogeneities in decentralized federated learning via synthetic anchors, in: Salakhutdinov, R., Kolter, Z., Heller, K., Weller, A., Oliver, N., Scarlett, J., Berkenkamp, F. (Eds.), Proceedings of the 41st International Conference on Machine Learning, PMLR. pp. 20111–20133.

[75] Hui, Y., Chen, X., Chen, L., 2024. A novel decentralized federated split learning framework in 6g edge networks. 2024 10th International Conference on Computer and Communications (ICCC) , 2123–2127.

[76] Im, C., Kim, J., 2025. Fully-distributed fairness-aware federated learning. IEEE INFOCOM 2025 - IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS) , 1–2.

[77] Ito, Y., Ochiai, H., Esaki, H., 2023. Self-organizing hierarchical topology in peer-to-peer federated learning: Strategies for scalability, robustness, and non-lid data. 2023 IEEE Future Networks World Forum (FNWF) , 1–7.

[78] Jang, S., Lim, A., Lee, Y., Lee, S., Lee, J., 2025. P3p-fed: Peer-to-peer personalized federated learning with dht-based local clustering. Proceedings of the 54th International Conference on Parallel Processing .

[79] Jeong, E., Kountouris, M., 2023. Personalized decentralized federated learning with knowledge distillation. ICC 2023 - IEEE International Conference on Communications , 1982–1987.

[80] Kairouz, P., McMahan, H.B., Avent, B., Bellet, A., Bennis, M., Bhagoji, A.N., Bonawitz, K., Charles, Z.B., Cormode, G., Cummings, R., D'Oliveira, R.G.L., Rouayheb, S.Y.E., Evans, D., Gardner, J., Garrett, Z., Gascón, A., Ghazi, B., Gibbons, P.B., Gruteser, M., Harchaoui, Z., He, C., He, L., Huo, Z., Hutchinson, B., Hsu, J., Jaggi, M., Javidi, T., Joshi, G., Khodak, M., Konecný, J., Korolova, A., Koushanfar, F., Koyejo, O., Lepoint, T., Liu, Y., Mittal, P., Mohri, M., Nock, R., Özgür, A., Pagh, R., Raykova, M., Qi, H., Ramage, D., Raskar, R., Song, D.X., Song, W., Stich, S.U., Sun, Z., Suresh, A.T., Tramèr, F., Vepakomma, P., Wang, J., Xiong, L., Xu, Z., Yang, Q., Yu, F.X., Yu, H., Zhao, S., 2019. Advances and open problems in federated learning. Found. Trends Mach. Learn. 14, 1–210.

[81] Kamp, M., Adilova, L., Sicking, J., Hüger, F., Schlicht, P., Wirtz, T., Wrobel, S., 2019. Efficient decentralized deep learning by dynamic model averaging, in: Berlingerio, M., Bonchi, F., Gärtner, T., Hurley, N., Ifrim, G. (Eds.), Machine Learning and Knowledge Discovery in Databases, Springer International Publishing, Cham. pp. 393–409.

[82] Kanagavelu, R., Wei, Q., Li, Z., Zhang, H., Samsudin, J., Yang, Y., Goh, R., Wang, S., 2022. Ce-fed: Communication efficient multi-party computation enabled federated learning. Array 15, 100207.

[83] Kazemi, K., 2025. Identification of solar panel conditions using fully decentralized federated learning. 2025 11th International Conference on Control, Instrumentation and Automation (ICCIA) , 01–06.

[84] Khan, M.K., 2025. Decentralized federated learning with adaptive aggregator selection. Proceedings of the 26th International Middleware Conference .

[85] Khan, Q.W., Khan, A., Rizwan, A., Ahmad, R., Khan, S., Kim, D.H., 2023. Decentralized machine learning training: A survey on synchronization, consolidation, and topologies. IEEE Access 11, 68031–68050.

[86] Koloskova, A., Stich, S.U., Jaggi, M., 2019. Decentralized stochastic optimization and gossip algorithms with compressed communication, in: International Conference on Machine Learning.

[87] Konecný, J., McMahan, H.B., Yu, F.X., Richtárik, P., Suresh, A.T., Bacon, D., 2016. Federated learning: Strategies for improving communication efficiency. ArXiv abs/1610.05492.

[88] Krizhevsky, A., 2009. Learning Multiple Layers of Features from Tiny Images. Master's thesis. University of Toronto.

[89] Krizhevsky, A., Sutskever, I., Hinton, G.E., 2012. Imagenet classification with deep convolutional neural networks. Communications of the ACM 60, 84 – 90.

[90] Lacour, D.D., Lacoste, M., Südholt, M., Traoré, J., 2023. Towards scalable resilient federated learning: A fully decentralised approach. 2023 IEEE International Conference on Pervasive Computing and Communications Workshops and other Affiliated Events (PerCom Workshops) , 621–627.

[91] Lanza, C., Tuytelaars, T., Miozzo, M., Angelats, E., Dini, P., 2025. Joint class and domain continual learning for decentralized federated processes. IEEE Access 13, 56982–56993.

[92] Le, Y., Yang, X.S., 2015. Tiny imagenet visual recognition challenge.

[93] LeCun, Y., Bottou, L., Bengio, Y., Haffner, P., 1998. Gradient-based learning applied to document recognition. Proc. IEEE 86, 2278–2324.

[94] Lee, W.T., 2025. Dpefed: A decentralized personalization and ensemble-based federated learning framework for healthcare. 2025 10th International Conference on Intelligent Computing and Signal Processing (ICSP) , 1198–1204.

[95] Lewis, D.D., Yang, Y., Rose, T.G., Li, F., 2004. Rcv1: A new benchmark collection for text categorization research. J. Mach. Learn. Res. 5, 361–397.

[96] Li, B., Gao, W., Xie, J., Gong, M., Wang, L., Li, H., 2024a. Prototype-based decentralized federated learning for the heterogeneous time-varying iot systems. IEEE Internet of Things Journal 11, 6916–6927.

[97] Li, C., Farkhoor, H., Liu, R., Yosinski, J., 2018. Measuring the intrinsic dimension of objective landscapes. ArXiv abs/1804.08838.

[98] Li, C., Li, G., Varshney, P.K., 2020a. Decentralized federated learning via mutual knowledge transfer. IEEE Internet of Things Journal 9, 1136–1147.

[99] Li, C., Xiao, M., Skoglund, M., 2024b. A communication-efficient semi-decentralized approach for federated learning with stragglers. 2024 IEEE Information Theory Workshop (ITW) , 229–234.

[100] Li, Q., Diao, Y., Chen, Q., He, B., 2021. Federated learning on non-iid data silos: An experimental study. 2022 IEEE 38th International Conference on Data Engineering (ICDE) , 965–978.

[101] Li, Q., Wen, Z., Wu, Z., He, B., 2019a. A survey on federated learning systems: Vision, hype and reality for data privacy and protection. IEEE Transactions on Knowledge and Data Engineering

35, 3347–3366.

[102] Li, S.Y., Zhou, T., Tian, X., Tao, D., 2022a. Learning to collaborate in decentralized learning of personalized models. 2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) , 9756–9765.

[103] Li, T., Sahu, A.K., Talwalkar, A., Smith, V., 2019b. Federated learning: Challenges, methods, and future directions. IEEE Signal Processing Magazine 37, 50–60.

[104] Li, T., Sahu, A.K., Zaheer, M., Sanjabi, M., Talwalkar, A., Smith, V., 2020b. Federated optimization in heterogeneous networks, in: Dhillon, I., Papailiopoulos, D., Sze, V. (Eds.), Proceedings of Machine Learning and Systems, pp. 429–450.

[105] Li, W., Lv, T., Ni, W., Zhao, J., Hossain, E., Poor, H.V., 2025. Route-and-aggregate decentralized federated learning under communication errors. IEEE Transactions on Neural Networks and Learning Systems 36, 16675–16691.

[106] Li, Z., Lu, J., Luo, S., Zhu, D., Shao, Y., Li, Y., Zhang, Z., Wang, Y., Wu, C., 2022b. Towards effective clustered federated learning: A peer-to-peer framework with adaptive neighbor matching. IEEE Transactions on Big Data 10, 812–826.

[107] Lian, X., Zhang, C., Zhang, H., Hsieh, C.J., Zhang, W., Liu, J., 2017. Can decentralized algorithms outperform centralized algorithms? a case study for decentralized parallel stochastic gradient descent, in: Neural Information Processing Systems.

[108] Lian, X., Zhang, W., Zhang, C., Liu, J., 2018. Asynchronous decentralized parallel stochastic gradient descent, in: Dy, J., Krause, A. (Eds.), Proceedings of the 35th International Conference on Machine Learning, PMLR. pp. 3043–3052.

[109] Lian, Z., Su, C., 2022. Decentralized federated learning for internet of things anomaly detection. Proceedings of the 2022 ACM on Asia Conference on Computer and Communications Security .

[110] Lian, Z., Yang, Q., Wang, W., Zeng, Q., Alazab, M., Zhao, H., Su, C., 2022. Deep-fel: Decentralized, efficient and privacy-enhanced federated edge learning for healthcare cyber physical systems. IEEE Transactions on Network Science and Engineering 9, 3558–3569.

[111] Liang, X., Tang, J., Siew, M., 2025. Decentralized federated learning framework for social iot with dynamic network topology. IEEE Internet of Things Journal 12, 35001–35013.

[112] Liao, Y., Xu, Y., Xu, H., Wang, L., Qian, C., 2022. Adaptive configuration for heterogeneous participants in decentralized federated learning. IEEE INFOCOM 2023 - IEEE Conference on Computer Communications , 1–10.

[113] Liao, Y., Xu, Y., Xu, H.Z., Chen, M., Wang, L., Qiao, C., 2024. Asynchronous decentralized federated learning for heterogeneous devices. IEEE/ACM Transactions on Networking 32, 4535–4550.

[114] Lin, T., Stich, S.U., Patel, K.K., Jaggi, M., 2020. Don't use large mini-batches, use local sgd, in: International Conference on Learning Representations.

[115] Liu, J., Che, T., Zhou, Y., Jin, R., Dai, H., Dou, D., Valduriez, P., 2023a. Aedfl: Efficient asynchronous decentralized federated learning with heterogeneous devices, in: SDM.

[116] Liu, J., Huo, Y., Qu, P., Xu, S., Liu, Z., Ma, Q., Huang, J., 2024a. Fedcd: A hybrid federated learning framework for efficient training with iot devices. IEEE Internet of Things Journal 11, 20040–20050.

[117] Liu, J., Liu, J., Xu, H.Z., Liao, Y., Wang, Z., Ma, Q., 2024b. Yoga: Adaptive layer-wise model aggregation for decentralized federated learning. IEEE/ACM Transactions on Networking 32, 1768–1780.

[118] Liu, S., Li, Q., Zhou, X., Ge, X., 2025. Energy-efficient d2d caching with decentralized federated deep reinforcement learning. 2025 30th Asia-Pacific Conference on Communications (APCC) , 1–6.

[119] Liu, W., Chen, L., Chen, Y., Wang, W., 2023b. Communication-efficient design for quantized decentralized federated learning. IEEE Transactions on Signal Processing 72, 1175–1188.

[120] Lodhi, A.H., Akgün, B., Özkasap, Ö., 2023. Implications of node selection in decentralized federated learning. 2023 31st Signal Processing and Communications Applications Conference (SIU) , 1–4.

[121] Lu, S., Zhang, Y., Wang, Y., 2020. Decentralized federated learning for electronic health records. 2020 54th Annual Conference on Information Sciences and Systems (CISS) , 1–5.

[122] Lu, Y., Yu, Z., Suri, N., 2022. Privacy-preserving decentralized federated learning over time-varying communication graph. ACM Transactions on Privacy and Security 26, 1 – 39.

[123] Luo, J.A., Chung, W.C., 2025. Personalized decentralized federated learning with hierarchical clustering in non-iid environments. 2025 IEEE International Conference on Consumer Electronics - Taiwan (ICCE-Taiwan) , 509–510.

[124] Luo, M., Mao, J., Xiong, G., Anjum, M.R., 2026. A semantic communication-assisted federated learning framework for internet of vehicles. Journal of Intelligent Computing and Networking .

[125] Luqman, A., Chattopadhyay, A., Lam, K.Y., 2023. Membership inference vulnerabilities in peer-to-peer federated learning. Proceedings of the 2023 Secure and Trustworthy Deep Learning Systems Workshop .

[126] Luqman, A., Chattopadhyay, A., Zhang, R., Niyato, D.T., 2025a. Banditmatch: A game-theoretic approach for stable preference matching in peer-to-peer federated learning. 2025 3rd International Conference on Federated Learning Technologies and Applications (FLTA) , 211–217.

[127] Luqman, A., Mahesh, R., Chattopadhyay, A., 2025b. Efficient model propagation for peer-to-peer federated learning using minimum spanning tree and gossip networks. Proceedings of the International Workshop on Secure and Efficient Federated Learning .

[128] Ma, Q., Liu, J., Xu, H.Z., Jia, Q., Xie, R., 2025. Fractal: Data-aware clustering and communication optimization for decentralized federated learning. IEEE Transactions on Big Data 11, 2102–2118.

[129] Ma, T., Wang, H., Li, C., 2023. Quantized distributed federated learning for industrial internet of things. IEEE Internet of Things Journal 10, 3027–3036.

[130] Ma, Z., Xu, Y., Xu, H., Liu, J., Xue, Y., 2024. Like attracts like: Personalized federated learning in decentralized edge computing. IEEE Transactions on Mobile Computing 23, 1080–1096.

[131] Maejima, K., Nishio, T., Yamazaki, A., Hara-Azumi, Y., 2023. Tram-fl: Routing-based model training for decentralized federated learning. 2024 IEEE 21st Consumer Communications & Networking Conference (CCNC) , 1038–1039.

[132] Marcus, M.P., Santorini, B., Marcinkiewicz, M.A., 1993. Building a large annotated corpus of english: The penn treebank. Comput. Linguistics 19, 313–330.

[133] Masmoudi, N., Jaafar, W., 2024. Ocd-fl: A novel communication-efficient peer selection-based decentralized federated learning. IEEE Transactions on Vehicular Technology 74, 6856–6861.

[134] McMahan, H.B., Moore, E., Ramage, D., Hampson, S., y Arcas, B.A., 2016. Communication-efficient learning of deep networks from decentralized data, in: International Conference on Artificial Intelligence and Statistics.

[135] Meidan, Y., Bohadana, M., Mathov, Y., Mirsky, Y., Breitenbacher, D., Shabtai, A., Elovici, Y., 2018. N-baiot—network-based detection of iot botnet attacks using deep autoencoders. IEEE Pervasive Computing 17, 12–22.

[136] Mulitze, F., Woisetschläger, H., Jacobsen, H.A., 2025. Mar-fl: A communication efficient peer-to-peer federated learning system. ArXiv abs/2512.05234.

[137] Nedić, A., Olshevsky, A., 2013. Distributed optimization over time-varying directed graphs. 52nd IEEE Conference on Decision and Control , 6855–6860.

[138] Nguyen, M., Tong, V., Souihi, S., Mellouk, A., 2023. Fully-decentralized federated learning for qoe estimation. GLOBECOM 2023 - 2023 IEEE Global Communications Conference , 5859–5864.

[139] Ochiai, H., Sun, Y., Jin, Q., Wongwiwatchai, N., Esaki, H., 2022. Wireless ad hoc federated learning: A fully distributed cooperative machine learning. ArXiv abs/2205.11779.

[140] Pakpahan, A.F., Hwang, I.S., 2024. Peer-to-peer federated learning on software-defined optical access network. IEEE Access 12, 84435–84451.

[141] Pan, D., Khoshkholghi, M.A., Mahmoodi, T., 2023. Decentralized federated learning methods for reducing communication cost and energy consumption in uav networks, in: Mobile Computing, Applications, and Services.

[142] Parasnis, R., Hosseinalipour, S., Chu, Y.W., Brinton, C.G., Chiang, M., 2023. Connectivity-aware semi-decentralized federated learning over time-varying d2d networks. Proceedings of the Twenty-fourth International Symposium on Theory, Algorithmic Foundations, and Protocol Design for Mobile Networks and Mobile Computing .

[143] Pei, H., Chen, X., Jiang, Y., Wang, F., Huang, L., 2025. Full speed ahead: A novel selective approach to speed up feature extraction in decentralized federated learning. Proceedings of the 2025 ACM Southeast Conference .

[144] Pinyoanuntapong, P., Huff, W.H., Lee, M., Chen, C., Wang, P., 2022. Toward scalable and robust aiot via decentralized federated learning. IEEE Internet of Things Magazine 5, 30–35.

[145] Qiu, W., Ai, W., Chen, H., Feng, Q., Tang, G., 2023. Decentralized federated learning for industrial iot with deep echo state networks. IEEE Transactions on Industrial Informatics 19, 5849–5857.

[146] Qu, Y., Dai, H., Yan, Z., Chen, J., Dong, C., Wu, F., Guo, S., 2021. Decentralized federated learning for uav networks: Architecture, challenges, and opportunities. IEEE Network 35, 156–162.

[147] Roy, A.G., Siddiqui, S., Pölsterl, S., Navab, N., Wachinger, C., 2019. Braintorrent: A peer-to-peer environment for decentralized federated learning. ArXiv abs/1905.06731.

[148] Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M.S., Berg, A.C., Fei-Fei, L., 2014. Imagenet large scale visual recognition challenge. International Journal of Computer Vision 115, 211 – 252.

[149] Sad, C., Retsinas, G., Soudris, D., Siozios, K., Masouros, D., 2025. Towards asynchronous peer-to-peer federated learning for heterogeneous systems. Proceedings of the 5th Workshop on Machine Learning and Systems .

[150] Sadiev, A., Borodich, E., Beznosikov, A., Dvinskikh, D., Chezhegov, S., Tappenden, R., Takác, M., Gasnikov, A.V., 2021. Decentralized personalized federated learning: Lower bounds and optimal algorithm for all personalization modes. EURO J. Comput. Optim. 10, 100041.

[151] Sahu, A.K., Li, T., Sanjabi, M., Zaheer, M., Talwalkar, A., Smith, V., 2018. On the convergence of federated optimization in heterogeneous networks. ArXiv abs/1812.06127.

[152] Saif, M., Hassan, M.Z., Hossain, M.J., 2024. Decentralized aggregation for energy-efficient federated learning in mmwave aerial-terrestrial integrated networks. IEEE Transactions on Machine Learning in Communications and Networking 2, 1283–1304.

[153] S'anchez, P.M.S., Beltrán, E.T.M., Llamas, M.F., Bovet, G., P'erez, G.M., Celdrán, A.H., 2024. Profe: Communication-efficient decentralized federated learning via distillation and prototypes. ICC 2025 - IEEE International Conference on Communications , 1596–1601.

[154] Savazzi, S., Nicoli, M., Rampa, V., Kianoush, S., 2020. Federated learning with mutually cooperating devices: A consensus approach towards server-less model optimization. ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP) , 3937–3941.

[155] Shahid, O., Pouriyeh, S., Parizi, R.M., Sheng, Q.Z., Srivastava, G., Zhao, L., 2021. Communication efficiency in federated learning: Achievements and challenges. ArXiv abs/2107.10996.

[156] Shang, M., Zhang, D., Li, F., 2023. Decentralized distributed federated learning based on multi-key homomorphic encryption. 2023 International Conference on Data Security and Privacy Protection (DSPP) , 260–265.

[157] Shi, W., Ling, Q., Yuan, K., Wu, G., Yin, W., 2013. On the linear convergence of the admm in decentralized consensus optimization. IEEE Transactions on Signal Processing 62, 1750–1761.

[158] Shi, Y., Shen, L., Wei, K., Sun, Y., Yuan, B., Wang, X., Tao, D., 2023. Improving the model consistency of decentralized federated learning, in: Krause, A., Brunskill, E., Cho, K., Engelhardt, B., Sabato, S., Scarlett, J. (Eds.), Proceedings of the 40th International Conference on Machine Learning, PMLR. pp. 31269–31291.

[159] Shinde, S.S., Tarchi, D., 2023. Joint air-ground distributed federated learning for intelligent transportation systems. IEEE Transactions on Intelligent Transportation Systems 24, 9996–10011.

[160] Simonyan, K., Zisserman, A., 2014. Very deep convolutional networks for large-scale image recognition. CoRR abs/1409.1556.

[161] Soltani, B., Haghighi, V., Zhou, Y., Sheng, Q.Z., Yao, L., 2024. Dflstar: A decentralized federated learning framework with self-knowledge distillation and participant selection, in: International Conference on Information and Knowledge Management.

[162] Song, X., Wang, Z., Baek, K.D., Ko, I.Y., 2025. Personalized user models in a real-world edge computing environment: A peer-to-peer federated learning framework. J. Web Eng. 23, 1057–1084.

[163] Sonnenburg, S., Franc, V., Yom-Tov, E., Sebag, M., 2008. Pascal large scale learning challenge. 25th International Conference on Machine Learning (ICML2008) Workshop. J. Mach. Learn. Res 10, 1937–1953.

[164] Stich, S.U., Cordonnier, J.B., Jaggi, M., 2018. Sparsified sgd with memory, in: Bengio, S., Wallach, H., Larochelle, H., Grauman, K., Cesa-Bianchi, N., Garnett, R. (Eds.), Advances in Neural Information Processing Systems, Curran Associates, Inc.

[165] Sun, C., Liu, X., Huang, X., Fischione, C., 2024. Layer-wise efficient federated learning with distributed clustering and d2d communications. 2024 IEEE 25th International Workshop on Signal Processing Advances in Wireless Communications (SPAWC) , 831–835.

[166] Sun, T., Li, D., Wang, B., 2021a. Decentralized federated averaging. IEEE Transactions on Pattern Analysis and Machine Intelligence 45, 4289–4301.

[167] Sun, Y., Shao, J., Mao, Y., Zhang, J., 2021b. Semi-decentralized federated edge learning for fast convergence on non-iid data. 2022 IEEE Wireless Communications and Networking Conference (WCNC) , 1898–1903.

[168] Syros, G., Yar, G., Boboila, S., Nita-Rotaru, C., Oprea, A., 2023. Backdoor attacks in peer-to-peer federated learning. ACM Transactions on Privacy and Security 28, 1 – 28.

[169] Tan, A.Z., Yu, H., zhen Cui, L., Yang, Q., 2021. Towards personalized federated learning. IEEE Transactions on Neural Networks and Learning Systems 34, 9587–9603.

[170] Tan, G., Yuan, H., Hu, H., Zhou, S., Zhang, Z., 2024. A framework of decentralized federated learning with soft clustering and 1-bit compressed sensing for vehicular networks. IEEE Internet of Things Journal 11, 23617–23629.

[171] Tang, H., Gan, S., Zhang, C., Zhang, T., Liu, J., 2018. Communication compression for decentralized training, in: Neural Information Processing Systems.

[172] Tang, Z., Shi, S., Chu, X., 2020. Communication-efficient decentralized learning with sparsification and adaptive peer selection. 2020 IEEE 40th International Conference on Distributed Computing Systems (ICDCS) , 1207–1208.

[173] Tang, Z., Shi, S., Li, B., Chu, X., 2023. Gossipfl: A decentralized federated learning framework with sparsified and adaptive communication. IEEE Transactions on Parallel and Distributed Systems 34, 909–922.

[174] Tedeschini, B.C., Savazzi, S., Stoklasa, R., Barbieri, L., Stathopoulos, I., Nicoli, M., Serio, L., 2022. Decentralized federated learning for healthcare networks: A case study on tumor segmentation. IEEE Access 10, 8693–8708.

[175] Thabet, S.K.S., Soltani, B., Zhou, Y., Sheng, Q.Z., Wen, S., 2024. Towards efficient decentralized federated learning: A survey, in: International Conference on Advanced Data Mining and Applications.

[176] Torkzaban, N., Gholami, A., Baras, J.S., 2025. Defel-fun: Decentralized federated learning framework for uav-enabled networks. ICC 2025 - IEEE International Conference on Communications , 6554–6559.

[177] Tsianos, K.I., Lawlor, S.F., Rabbat, M.G., 2012. Communication/computation tradeoffs in consensus-based distributed optimization, in: Neural Information Processing Systems.

[178] Tummala, V.M.R., Hazra, A., Kalita, A., Mohan, G., 2024. Cluster based pseudo hierarchical decentralized federated learning in uav networks. 2024 IEEE 100th Vehicular Technology Conference (VTC2024-Fall) , 1–5.

[179] Tyou, I., Murata, T., Fukami, T., Takezawa, Y., Niwa, K., 2024. A localized primal-dual method for centralized/decentralized federated learning robust to data heterogeneity. IEEE Transactions on Signal and Information Processing over Networks 10, 94–107.

[180] Uddin, M.P., Xiang, Y., Hasan, M.M., Bai, J., Zhao, Y., Gao, L., 2025. A systematic literature review of robust federated learning: Issues, solutions, and future research directions. ACM Computing Surveys 57, 1 – 62.

[181] Vanhaesebrouck, P., Bellet, A., Tommasi, M., 2017. Decentralized Collaborative Learning of Personalized Models over Networks, in: Singh, A., Zhu, J. (Eds.), Proceedings of the 20th International Conference on Artificial Intelligence and Statistics, PMLR. pp. 509–517.

[182] Verbraeken, J., Wolting, M., Katzy, J., Kloppenburg, J., Verbelen, T., Rellermeyer, J.S., 2019. A survey on distributed machine learning. ACM Computing Surveys (CSUR) 53, 1 – 33.

[183] Wang, H., Muñoz-González, L., Eklund, D., Raza, S., 2021a. Non-iid data re-balancing at iot edge with peer-to-peer federated learning for anomaly detection. Proceedings of the 14th ACM Conference on Security and Privacy in Wireless and Mobile Networks .

[184] Wang, H., Muñoz-González, L., Hameed, M.Z., Eklund, D., Raza, S., 2023a. Sparsfa: Towards robust and communication-efficient peer-to-peer federated learning. Comput. Secur. 129, 103182.

[185] Wang, H., Wu, J., Kong, D., Wang, Y., Li, Y., Bian, Z., Ma, J., Zeng, D., 2025a. A peer-to-peer federated incremental learning framework for multi-institutional low-dose ct denoising. Proceedings of the International Workshop on Personalized Incremental Learning in Medicine .

[186] Wang, J., Joshi, G., 2021. Cooperative sgd: A unified framework for the design and analysis of local-update sgd algorithms. Journal of Machine Learning Research 22, 1–50.

[187] Wang, J., Liu, Q., Liang, H., Joshi, G., Poor, H.V., 2020. Tackling the objective inconsistency problem in heterogeneous federated optimization, in: Larochelle, H., Ranzato, M., Hadsell, R., Balcan, M., Lin, H. (Eds.), Advances in Neural Information Processing Systems, Curran Associates, Inc.. pp. 7611–7623.

[188] Wang, J., Sahu, A.K., Joshi, G., Kar, S., 2022a. Matcha: A matching-based link scheduling strategy to speed up distributed optimization. IEEE Transactions on Signal Processing 70, 5208–5221.

[189] Wang, J., Xiao, H., Ma, F., 2024. Asymmetric mutual learning for decentralized federated medical imaging. Proceedings of the 15th ACM International Conference on Bioinformatics, Computational Biology and Health Informatics .

[190] Wang, L., Xu, Y., Xu, H., Chen, M., Huang, L., 2023b. Accelerating decentralized federated learning in heterogeneous edge computing. IEEE Transactions on Mobile Computing 22, 5001–5016.

[191] Wang, S., Tuor, T., Salonidis, T., Leung, K.K., Makaya, C., He, T., Chan, K.S., 2018. Adaptive federated learning in resource constrained edge computing systems. IEEE Journal on Selected Areas in Communications 37, 1205–1221.

[192] Wang, T., Liu, Y., Zheng, X., Dai, H., Jia, W., Xie, M., 2021b. Edge-based communication optimization for distributed federated learning. IEEE Transactions on Network Science and Engineering 9, 2015–2024.

[193] Wang, X., Lalitha, A., Javidi, T., Koushanfar, F., 2022b. Peer-to-peer variational federated learning over arbitrary graphs. IEEE Journal on Selected Areas in Information Theory 3, 172–182. doi:10.1109/JSAIT.2022.3189051.

[194] Wang, X., Li, X., Zhou, Z., Li, C., Liu, Y., 2025b. Adf-lora: Alternating low-rank aggregation for decentralized federated fine-tuning. ArXiv abs/2511.18291.

[195] Wang, Y., Lin, L., Chen, J., 2022c. Communication-efficient adaptive federated learning, in: Chaudhuri, K., Jegelka, S., Song, L., Szepesvari, C., Niu, G., Sabato, S. (Eds.), Proceedings of the 39th International Conference on Machine Learning, PMLR. pp. 22802–22838.

[196] Wei, K., Li, J., Ding, M., Ma, C., Yang, H.H., Farhad, F., Jin, S., Quek, T.Q.S., Poor, H.V., 2019. Federated learning with differential privacy: Algorithms and performance analysis. IEEE Transactions on Information Forensics and Security 15, 3454–3469.

[197] Wei, M., Yu, W., Chen, D., 2025. Accdfl: Accelerated decentralized federated learning for healthcare iot networks. IEEE Internet of Things Journal 12, 5329–5345.

[198] Wink, T., Nochta, Z., 2021. An approach for peer-to-peer federated learning. 2021 51st Annual IEEE/IFIP International Conference on Dependable Systems and Networks Workshops (DSN-W) , 150–157.

[199] Wu, C., Zhu, Y., Wang, F., 2022. Dsfl: Decentralized satellite federated learning for energy-aware leo constellation computing. 2022 IEEE International Conference on Satellite Computing (Satellite) , 25–30.

[200] Wu, J., Dong, F., Leung, H., Zhu, Z., Zhou, J., Drew, S., 2023. Topology-aware federated learning in edge computing: A comprehensive survey. ACM Computing Surveys 56, 1 – 41.

[201] Wulfert, L., Asadi, N., Chung, W.Y., Wiede, C., Grabmaier, A., 2023. Adaptive decentralized federated gossip learning for resource-constrained iot devices. Proceedings of the 4th International Workshop on Distributed Machine Learning .

[202] Xiao, H., Rasul, K., Vollgraf, R., 2017. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. ArXiv abs/1708.07747.

[203] Xiao, L., Boyd, S.P., 2003. Fast linear iterations for distributed averaging. 42nd IEEE International Conference on Decision and Control (IEEE Cat. No.03CH37475) 5, 4997–5002 Vol.5.

[204] Xiao, Y., Ye, Y., Huang, S., Hao, L., Ma, Z., Xiao, M., Mumtaz, S., Dobre, O.A., 2021. Fully decentralized federated learning-based onboard mission for uav swarm system. IEEE Communications Letters 25, 3296–3300.

[205] Xiong, K., Wang, R., Leng, S., Huang, C., Yuen, C., 2024. Ris-empowered topology control for decentralized federated learning in urban air mobility. IEEE Internet of Things Journal 11, 40757–40770.

[206] Xu, C., Qu, Y., Xiang, Y., Gao, L., 2021. Asynchronous federated learning on heterogeneous devices: A survey. Comput. Sci. Rev. 50, 100595.

[207] Xu, D., Liu, Y., Wen, G., Jin, Y., Chai, T., Yang, T., 2025a. Defedtl: A decentralized federated transfer learning method for fault diagnosis. IEEE Transactions on Industrial Informatics 21, 1704–1713.

[208] Xu, Y., Xiao, M., Wu, J., Gao, G., Li, D., Xu, H., Zhang, T., 2025b. Enhancing decentralized federated learning with model pruning and adaptive communication. IEEE Transactions on Industrial Informatics 21, 70–84.

[209] Xu, Z., Jiang, F., Niu, L., Jia, J., Poovendran, R., 2024. Poster: Brave: Byzantine-resilient and privacy-preserving peer-to-peer federated learning. Proceedings of the 19th ACM Asia Conference on Computer and Communications Security .

[210] Yan, D., Yang, Y., Hu, M., Fu, X., Chen, M., 2025. Mmdfl: Multi-model-based decentralized federated learning for resource-constrained aiot systems. 2025 62nd ACM/IEEE Design Automation Conference (DAC) , 1–7.

[211] Yan, Z.H., Li, D., 2024a. Convergence time optimization for decentralized federated learning with leo satellites via number control. IEEE Transactions on Vehicular Technology 73, 4517–4522.

[212] Yan, Z.H., Li, D., 2024b. Performance analysis for resource constrained decentralized federated learning over wireless networks. IEEE Transactions on Communications 72, 4084–4100.

[213] Yang, M., Zhang, J., Wang, L., Vo, N.S., Wang, X., 2026. Dfedsat: Communication-efficient and robust decentralized federated learning for leo satellite constellations. IEEE Transactions on Cognitive Communications and Networking 12, 4972–4985. doi:10.1109/TCCN.2025.3638773.

[214] Yang, Q., Liu, Y., Chen, T., Tong, Y., 2019. Federated machine learning. ACM Transactions on Intelligent Systems and Technology

(TIST) 10, 1 – 19.
[215] Yao, A.C.C., 1982. Protocols for secure computations. 23rd Annual Symposium on Foundations of Computer Science (sfcs 1982) , 160–164.
[216] Ye, H., Liang, L., Li, G.Y., 2021. Decentralized federated learning with unreliable communications. IEEE Journal of Selected Topics in Signal Processing 16, 487–500.
[217] Yu, G., Wang, X., Sun, C., Wang, Q., Yu, P., Ni, W., Liu, R.P., 2023. Ironforge: An open, secure, fair, decentralized federated learning. IEEE Transactions on Neural Networks and Learning Systems 36, 354–368.
[218] Yuan, L., Sun, L., Yu, P.S., Wang, Z., 2023a. Decentralized federated learning: A survey and perspective. IEEE Internet of Things Journal 11, 34617–34638.
[219] Yuan, Y., Liu, J., Jin, D., Yue, Z., Yang, T., Chen, R., Wang, M., Sun, C., Xu, L., Hua, F., Guo, Y., Tang, X., He, X., Yi, X., Li, D., Wen, G., Yu, W., Zhang, H.T., Chai, T., Sui, S., Ding, H., 2023b. Decefl: a principled fully decentralized federated learning framework.
[220] Zantedeschi, V., Bellet, A., Tommasi, M., 2019. Fully decentralized joint learning of personalized models and collaboration graphs, in: International Conference on Artificial Intelligence and Statistics.
[221] Zehtabi, S., Hosseinalipour, S., Brinton, C.G., 2022. Resource-constrained decentralized federated learning via personalized event-triggering. IEEE Transactions on Networking 34, 1912–1926.
[222] Zhai, Z., Wu, Q., Yu, S., Li, R., Zhang, F., Chen, X., 2024. Fedleo: An offloading-assisted decentralized federated learning framework for low earth orbit satellite networks. IEEE Transactions on Mobile Computing 23, 5260–5279.
[223] Zhang, A., Zhao, P., Lu, W., Zhang, G., 2025a. Decentralized federated learning towards communication efficiency, robustness, and personalization. ACM Transactions on Sensor Networks 21, 1 – 20.
[224] Zhang, A., Zhao, P., Lu, W., Zhang, G., 2025b. Personalized decentralized federated learning: A privacy-enhanced and byzantine-resilient approach. IEEE Transactions on Computational Social Systems 12, 3206–3217.
[225] Zhang, J., Tu, H., Ren, Y., Wan, J., Zhou, L., Li, M., Wang, J., 2018. An adaptive synchronous parallel strategy for distributed machine learning. IEEE Access 6, 19222–19230.
[226] Zhang, L., Xu, J., Vijayakumar, P., Sharma, P.K., Ghosh, U., 2023. Homomorphic encryption-based privacy-preserving federated learning in iot-enabled healthcare system. IEEE Transactions on Network Science and Engineering 10, 2864–2880.
[227] Zhang, L., Xu, Z.L., 2023. k-pfed: Communication-efficient personalized federated clustering. 2023 IEEE 3rd International Conference on Electronic Technology, Communication and Information (ICETCI) , 1026–1029.
[228] Zhang, X., Fang, M., Liu, Z., Yang, H., Liu, J., Zhu, Z., 2022. Net-fleet: achieving linear convergence speedup for fully decentralized federated learning with heterogeneous data. Proceedings of the Twenty-Third International Symposium on Theory, Algorithmic Foundations, and Protocol Design for Mobile Networks and Mobile Computing .
[229] Zhang, X., Trmal, J., Povey, D., Khudanpur, S., 2014. Improving deep neural network acoustic models using generalized maxout networks. 2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP) , 215–219.
[230] Zhang, Y., Gao, F., Zhou, K., 2024. Efficient communication for decentralized federated learning: An energy disaggregation case study. 2024 IEEE 15th International Symposium on Power Electronics for Distributed Generation Systems (PEDG) , 1–5.
[231] Zhang, Y., Xiang, T., Hospedales, T.M., Lu, H., 2017. Deep mutual learning. 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition , 4320–4328.
[232] Zhao, J., Zhu, H., Wang, F., Lu, R., Liu, Z., Li, H., 2022. Pvd-fl: A privacy-preserving and verifiable decentralized federated learning framework. IEEE Transactions on Information Forensics and Security 17, 2059–2073.
[233] Zhao, Z., Mao, Y., Liu, Y., Song, L., Ouyang, Y., Chen, X., Ding, W., 2023. Towards efficient communications in federated learning: A contemporary survey. Journal of the Franklin Institute 360, 8669–8703. doi:https://doi.org/10.1016/j.jfranklin.2022.12.053.
[234] Zheng, G., Ni, Q., Navaie, K., Pervaiz, H.B., Zarakovitis, C.C., 2023. A distributed learning architecture for semantic communication in autonomous driving networks for task offloading. IEEE Communications Magazine 61, 64–68.
[235] Zhou, P., Lin, Q., Loghin, D., Ooi, B.C., Wu, Y., Yu, H., 2020a. Communication-efficient decentralized machine learning over heterogeneous networks. 2021 IEEE 37th International Conference on Data Engineering (ICDE) , 384–395.
[236] Zhou, S., Xu, K., Li, G.Y., 2023. Communication-efficient decentralized federated learning via one-bit compressive sensing. 2024 IEEE 99th Vehicular Technology Conference (VTC2024-Spring) , 1–5.
[237] Zhou, Y., Pu, G., Ma, X., Li, X., Wu, D.O., 2020b. Distilled one-shot federated learning. ArXiv abs/2009.07999.
[238] Zhu, L., Liu, Z., Han, S., 2019. Deep leakage from gradients, in: Neural Information Processing Systems.
[239] Zong, R., Qin, Y., Wu, F., Tang, Z., Li, K., 2023. Fedcs: Efficient communication scheduling in decentralized federated learning. Inf. Fusion 102, 102028.

A. Appendix

A.1. Model Aggregation

A.1.1. ADMM

Equation 3 presents the iterative updates for ADMM in the case of global variable consensus optimization[8] [24] from the perspective of actor i, as provided in Shi et al. [157]. Here, $f_i(\cdot,\cdot)$ denotes the local objective function, c is the penalty parameter from the augmented Lagrangian, $\mathcal{N}_i$ is the set of neighbors, m_i^k is the local solution at iteration k, and λ_i^{k+1} are the local Lagrange multipliers. Similarly to Equation 2 for incAvg, x_i represents the private data of actor i and ∇ is the differential operator. Note that the neighboring actors exchange their model updates $\{m_j^k, \forall j \in \mathcal{N}_i\}$ (i.e., the local solutions) beforehand, as they are required for the update step.

$$\begin{aligned} \lambda_i^{k+1} &= \lambda_i^k + c\left(|\mathcal{N}_i| m_i^k - \sum_{j\in\mathcal{N}_i} m_j^k\right) \\ m_i^{k+1} &= \left(\nabla f_i(m_i^k, x_i) + 2c|\mathcal{N}_i| I\right)^{-1} \\ &\quad \cdot \left(c|\mathcal{N}_i| m_i^k + c\sum_{j\in\mathcal{N}_i} m_j^k - \lambda_i^{k+1}\right) \end{aligned} \tag{3}$$

A.1.2. Knowledge Transfer and Knowledge Distillation

Equation 4 presents the updates of both the local model of the sender m_s and the received model m_r in the l-th iteration of knowledge transfer. Here, η_s and η_r denote the learning rates of the knowledge transfer update, B_l denotes the l-th data batch, and $P_{r,l}$ and $P_{s,l}$ are the soft predictions (i.e., the outputs of the model) obtained on the data batch

[8] In global variable consensus optimization, one of the three update steps from general ADMM is omitted.

B_l under the respective model m_s or m_r. The loss functions $\text{Loss}_s(\cdot)$ and $\text{Loss}_r(\cdot)$ consist of two additive terms: (1) The general, task-specific loss function evaluated on the soft predictions $P_{r,l}$ and $P_{s,l}$, and (2) the Kullback-Leibler divergence that quantifies the match of the two sets of soft predictions. More detailed information on these loss functions can be found in reference [98].

$$m_s = m_s - \eta_s \frac{\partial \text{Loss}_s(m_s, B_l, P_{r,l})}{\partial m_s}$$
$$m_r = m_r - \eta_r \frac{\partial \text{Loss}_r(m_r, B_l, P_{s,l})}{\partial m_r} \quad (4)$$

A.2. Excluded Publications

We demonstrate the criteria which led to the selection of the main publications that are discussed in this work by listing excluded publications with the corresponding reason for exclusion. Reasons for exclusion are the degree of centralization in Table 11, the application focus in Tables 12, 13, 14, 15, the focus on a different challenge in Tables 17, 16, 18, the primary focus not on communication in Table 19, or focus on evaluation and analysis of DFL in Table 20.

For the remaining publications, we show in the following list that their most distinctive techniques are already covered by our selected papers:

- Asadi et al. [5]: "Gist - Optimizing Segmentation for Decentralized Federated Learning on Tiny Devices"
  Implements wAvg *model aggregation* with dynamic mixing weights, similar to Wang et al. [184] and Liu et al. [115]. Implements *partial device participation* with selection based on model update properties, similar to Wang et al. [190], Liao et al. [113], and Soltani et al. [161].

- Barbieri et al. [13]: "A Layer Selection Optimizer for Communication-Efficient Decentralized Federated Deep Learning"
  Implements layer-wise TopK sparsification, similar to general TopK sparsification from Wang et al. [184] but on the level of entire NN layers.

- Chen et al. [34]: "Communication Efficient Federated Learning via Ordered ADMM in a Fully Decentralized Setting"
  Implements a variant of ADMM for *model aggregation*, similar to [236]. Implements selective *partial device participation* based on model update properties, similar to Wang et al. [190], Liao et al. [113], and Soltani et al. [161].

- Cheng et al. [35]: "A decentralized federated learning model based on global channel selection"
  Implements model update-aware sparsification, similar to Wang et al. [184] and Liu et al. [115].

- Dai et al. [41]: "DisPFL: Towards Communication-Efficient Personalized Federated Learning via Decentralized Sparse Training"
  Implements training-aware sparsification as *compression* method, similar to Wang et al. [184] and Liu et al. [115].

- Fang et al. [50]: "QAT-DFL: A Communication Efficient Decentralized Federated Learning with Quantization-Aware Training Design"
  Implements quantization as *compression* method, similar to Koloskova et al. [86] and Liu et al. [119].

- Gong et al. [55]: "Lightweight Decentralized Federated Learning with Arbitrary Client Participation"
  Implements dynamic *local computation*, similar to Zhou et al. [236]. Implements selective *partial device participation* based on connectivity properties, similar to Wang et al. [188].

- Li et al. [105]: "Route-and-Aggregate Decentralized Federated Learning Under Communication Errors"
  Implements wAvg *model aggregation* with dynamic mixing weights, similar to Wang et al. [184] and Liu et al. [115]. Implements selective *partial device participation* based on network properties, similar to Tang et al. [173].

- Liu et al. [117]: "YOGA: Adaptive Layer-Wise Model Aggregation for Decentralized Federated Learning"
  Addresses communication cost reduction on system level instead of individual DFL methods.

- Lodhi et al. [120]: "Implications of Node Selection in Decentralized Federated Learning"
  Implements selective *partial device participation* based on model update properties, similar to Wang et al. [190], Liao et al. [113], and Soltani et al. [161].

- Ma et al. [130]: "Like Attracts Like: Personalized Federated Learning in Decentralized Edge Computing"
  Implements sparsification as *compression* method, similar to Koloskova et al. [86], Tang et al. [173], Wang et al. [184], Wang et al. [190], and Liu et al. [115]. Implements selective *partial device participation* based on model update properties, similar to Wang et al. [190], Liao et al. [113], and Soltani et al. [161].

- Maejima et al. [131]: "Tram-FL: Routing-based Model Training for Decentralized Federated Learning"
  Implements the incAvg scheme for *model aggregation*, similar to Gupta et al. [57].

- Masmoudi and Jaafar [133]: "OCD-FL: A Novel Communication-Efficient Peer Selection-Based Decentralized Federated Learning"
  Implements selective *partial device participation* based on model update properties, similar to Wang et al. [190], Liao et al. [113], and Soltani et al. [161].

Reference	Title
Behjati et al. [15]	UAV-assisted federated learning with hybrid LoRa P2P/LoRaWAN for sustainable biosphere
Hui et al. [75]	A Novel Decentralized Federated Split Learning Framework in 6G Edge Networks
Im and Kim [76]	Fully-Distributed Fairness-Aware Federated Learning
Khan [84]	Decentralized Federated Learning with Adaptive Aggregator Selection
Lacour et al. [90]	Towards Scalable Resilient Federated Learning: A Fully Decentralised Approach
Li et al. [106]	Towards Effective Clustered Federated Learning: A Peer-to-Peer Framework With Adaptive Neighbor Matching
Li et al. [99]	A Communication-Efficient Semi-Decentralized Approach for Federated Learning with Stragglers
Liu et al. [116]	FedCD: A Hybrid Federated Learning Framework for Efficient Training With IoT Devices
Ma et al. [128]	FRACTAL: Data-Aware Clustering and Communication Optimization for Decentralized Federated Learning
Parasnis et al. [142]	Connectivity-Aware Semi-Decentralized Federated Learning over Time-Varying D2D Networks
Sun et al. [167]	Semi-Decentralized Federated Edge Learning for Fast Convergence on Non-IID Data
Sun et al. [165]	Layer-wise Efficient Federated Learning with Distributed Clustering and D2D Communications
Wang et al. [192]	Edge-Based Communication Optimization for Distributed Federated Learning

Table 11
Publications not included in the primary selected papers due to involved centralization.

Reference	Title
Chen and Yuan [32]	Decentralized Personalization for Federated Medical Image Segmentation via Gossip Contrastive Mutual Learning
Chetoui and Akhloufi [36]	Peer-to-Peer Federated Learning for COVID-19 Detection Using Transformers
Lee [94]	DPEFed: A Decentralized Personalization and Ensemble-based Federated Learning Framework for Healthcare
Lu et al. [121]	Decentralized Federated Learning for Electronic Health Records
Roy et al. [147]	BrainTorrent: A Peer-to-Peer Environment for Decentralized Federated Learning
Wang et al. [189]	Asymmetric Mutual Learning for Decentralized Federated Medical Imaging
Wang et al. [185]	A Peer-to-Peer Federated Incremental Learning Framework for Multi-Institutional Low-Dose CT Denoising
Wei et al. [197]	AccDFL: Accelerated Decentralized Federated Learning for Healthcare IoT Networks

Table 12
Publications not included in the primary selected papers due to their focus on healthcare applications.

- S'anchez et al. [153]: "ProFe: Communication-Efficient Decentralized Federated Learning via Distillation and Prototypes"
  Implements knowledge distillation for *model aggregation*, similar to Li et al. [98]. Implements quantization as *compression* method, similar to Koloskova et al. [86] and Liu et al. [119].
- Zhang et al. [230]: "Efficient Communication for Decentralized Federated Learning: An Energy Disaggregation Case Study"
  Implements a *partial device participation* strategy where only a single device participates in the round, similar to Tang et al. [173], Liu et al. [115], and Soltani et al. [161].

Reference	Title
Asheralieva et al. [6]	Dynamic Distributed Model Compression for Efficient Decentralized Federated Learning and Incentive Provisioning in Edge Computing Networks
Du et al. [43]	Adaptive Decentralized Federated Learning in Resource-Constrained IoT Networks
Liang et al. [111]	Decentralized Federated Learning Framework for Social IoT With Dynamic Network Topology
Ma et al. [129]	Quantized Distributed Federated Learning for Industrial Internet of Things
Pei et al. [143]	Full Speed Ahead: A Novel Selective Approach to Speed up Feature Extraction in Decentralized Federated Learning
Song et al. [162]	Personalized User Models in a Real-world Edge Computing Environment: A Peer-to-peer Federated Learning Framework
Wulfert et al. [201]	Adaptive Decentralized Federated Gossip Learning for Resource-Constrained IoT Devices
Xu et al. [208]	Enhancing Decentralized Federated Learning With Model Pruning and Adaptive Communication
Yan et al. [210]	MMDFL: Multi-Model-based Decentralized Federated Learning for Resource-Constrained AIoT Systems

Table 13
Publications not included in the primary selected papers due to their focus on IoT applications.

Reference	Title
Barbieri et al. [12]	A Compressed Decentralized Federated Learning Framework for Enhanced Environmental Awareness in V2V Networks
Benhelal et al. [21]	Communication-Efficient Multi-Level Decentralized Federated Learning for Trajectory Prediction
Saif et al. [152]	Decentralized Aggregation for Energy-Efficient Federated Learning in mmWave Aerial-Terrestrial Integrated Networks
Shinde and Tarchi [159]	Joint Air-Ground Distributed Federated Learning for Intelligent Transportation Systems
Torkzaban et al. [176]	DeFeL-FUN: Decentralized Federated Learning Framework for UAV-enabled Networks
Tummala et al. [178]	Cluster Based Pseudo Hierarchical Decentralized Federated Learning in UAV Networks
Xiao et al. [204]	Fully Decentralized Federated Learning-Based On-Board Mission for UAV Swarm System

Table 14
Publications not included in the primary selected papers due to their focus on vehicle applications.

Reference	Title
Attanayaka et al. [10]	Peer-to-Peer Federated Learning based Anomaly Detection for Open Radio Access Networks
Kazemi [83]	Identification of Solar Panel Conditions Using Fully Decentralized Federated Learning
Nguyen et al. [138]	Fully-Decentralized Federated Learning for QoE Estimation
Pakpahan and Hwang [140]	Peer-to-Peer Federated Learning on Software-Defined Optical Access Network
Xu et al. [207]	DeFedTL: A Decentralized Federated Transfer Learning Method for Fault Diagnosis
Yang et al. [213]	DFedSat: Communication-Efficient and Robust Decentralized Federated Learning for LEO Satellite Constellations

Table 15
Publications not included in the primary selected papers due to their focus on other applications.

Reference	Title
Beilharz et al. [16]	Implicit Model Specialization through DAG-based Decentralized Federated Learning
Duchesne et al. [44]	MultiConfederated Learning: Inclusive Non-IID Data handling with Decentralized Federated Learning
Guo et al. [56]	Decentralized Federated Learning with Local Differential Privacy and Comprehensive Neighbor Selection
Jang et al. [78]	P3P-Fed: Peer-to-Peer Personalized Federated Learning with DHT-based Local Clustering
Jeong and Kountouris [79]	Personalized Decentralized Federated Learning with Knowledge Distillation
Luo and Chung [123]	Personalized Decentralized Federated Learning with Hierarchical Clustering in Non-IID Environments
Zehtabi et al. [221]	Resource-Constrained Decentralized Federated Learning via Personalized Event-Triggering
Zhang et al. [228]	NET-FLEET: Achieving Linear Convergence Speedup for Fully Decentralized Federated Learning with Heterogeneous Data
Zhang and Xu [227]	k-PFed: Communication-efficient Personalized Federated Clustering

Table 16
Publications not included in the primary selected papers due to their focus on the data heterogeneity challenge.

Reference	Title
Alalyan et al. [2]	Secure Peer-to-Peer Federated Learning for Efficient Cyberattacks Detection in 5G and Beyond Networks
Fang et al. [49]	Byzantine-Robust Decentralized Federated Learning
Ghimire et al. [54]	Implementation of Secure and Privacy-aware AI Hardware using Distributed Federated Learning
Herath et al. [66]	A Lightweight Reputation-Based Mechanism for Incentivizing Cooperation in Decentralized Federated Learning
Hijazi et al. [68]	Collaborative IoT learning with secure peer-to-peer federated approach
Hu and Tei [73]	Adaptive Defense Mechanisms against Dynamic Poisoning Attacks in Decentralized Federated Learning
Lu et al. [122]	Privacy-preserving Decentralized Federated Learning over Time-varying Communication Graph
Luqman et al. [125]	Membership Inference Vulnerabilities in Peer-to-Peer Federated Learning
Shang et al. [156]	Decentralized Distributed Federated Learning Based on Multi-Key Homomorphic Encryption
Syros et al. [168]	Backdoor Attacks in Peer-to-Peer Federated Learning
Wink and Nochta [198]	An Approach for Peer-to-Peer Federated Learning
Xu et al. [209]	POSTER: Brave: Byzantine-Resilient and Privacy-Preserving Peer-to-Peer Federated Learning
Zhang et al. [223]	Decentralized Federated Learning towards Communication Efficiency, Robustness, and Personalization
Zhao et al. [232]	PVD-FL: A Privacy-Preserving and Verifiable Decentralized Federated Learning Framework

Table 17
Publications not included in the primary selected papers due to their focus on the privacy and security challenge.

Reference	Title
Ito et al. [77]	Self-Organizing Hierarchical Topology in Peer-to-Peer Federated Learning: Strategies for Scalability, Robustness, and Non-IID Data
Luqman et al. [126]	BanditMatch: A Game-theoretic Approach for Stable Preference Matching in Peer-to-Peer Federated Learning
Luqman et al. [127]	Efficient Model Propagation for Peer-to-Peer Federated Learning using Minimum Spanning Tree and Gossip Networks
Mulitze et al. [136]	MAR-FL: A Communication Efficient Peer-to-Peer Federated Learning System
Sad et al. [149]	Towards Asynchronous Peer-to-Peer Federated Learning for Heterogeneous Systems

Table 18
Publications not included in the primary selected papers due to their focus on the system heterogeneity challenge.

Reference	Title
Behera and Chakraborty [14]	pFedGame — Decentralized Federated Learning using Game Theory in Dynamic Topology
Beltrán et al. [18]	DEMO: NEBULA — Decentralized Federated Learning for Heterogeneous Networks
Wang et al. [193]	Peer-to-Peer Federated Learning on Graphs
Liu et al. [118]	Energy-Efficient D2D Caching with Decentralized Federated Deep Reinforcement Learning
Yu et al. [217]	IRONFORGE: An Open, Secure, Fair, Decentralized Federated Learning
Yuan et al. [219]	DeceFL: a principled fully decentralized federated learning framework

Table 19
Publications not included in the primary selected papers due to insufficient focus on communication efficiency.

Reference	Title
Chellapandi et al. [28]	On the Convergence of Decentralized Federated Learning Under Imperfect Information Sharing
Hashemi et al. [60]	On the Benefits of Multiple Gossip Steps in Communication-Constrained Decentralized Federated Learning
Yan and Li [212]	Performance Analysis for Resource Constrained Decentralized Federated Learning Over Wireless Networks

Table 20
Publications not included in the primary selected papers since their main contribution is an analysis or evaluation of DFL methods.