IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received 23 January 2025, accepted 11 February 2025, date of publication 17 February 2025, date of current version 24 February 2025.

Digital Object Identifier 10.1109/ACCESS.2025.3542772

RESEARCH ARTICLE

Performance Evaluation of Decentralized Federated Learning: Impact of Fully and K-Connected Topologies, Heterogeneous Computing Resources, and Communication Bandwidth

VO VAN TRUONG[1], (Graduate Student Member, IEEE), PHAM KHANH QUAN[1], (Graduate Student Member, IEEE), DONG-HWAN PARK[2], AND TAEHONG KIM[1], (Senior Member, IEEE)

[1]School of Information and Communication Engineering, Chungbuk National University, Cheongju 28644, South Korea
[2]Electronics and Telecommunications Research Institute, Daejeon 34129, South Korea

Corresponding authors: Taehong Kim (taehongkim@cbnu.ac.kr) and Dong-Hwan Park (dhpark@etri.re.kr)

This work is supported by the Korea Agency for Infrastructure Technology Advancement (KAIA) grant funded by the Ministry of Land, Infrastructure and Transport (Grant: RS-2022-00155803).

ABSTRACT Decentralized federated learning (DFL) enables collaborative model training across distributed devices while preserving data privacy. Despite this, the performance of DFL in real-world scenarios, characterized by varying network topologies, heterogeneous client resources, and communication bandwidth, remains underexplored. This study analyzes the impact of network topologies, focusing on fully connected and k-connected structures, while examining how computing resource and communication bandwidth heterogeneity affect system efficiency using diverse datasets such as Fashion MNIST, CIFAR-10, and CIFAR-100. The results show that fully connected topologies offer robust communication but suffer from significant scalability issues. By contrast, k-connected networks provide a more scalable solution with reduced communication overhead. Furthermore, clients with limited CPU resources increase the convergence times by as much as 30%, while those with low communication bandwidth variations exacerbate the convergence times by up to 50%. These findings provide practical insights and guidelines for optimizing and designing scalable DFL systems suitable for real-world deployments.

INDEX TERMS Federated learning, decentralized learning, network topology, heterogeneity, communication bandwidth.

I. INTRODUCTION

Federated learning (FL) [1] is a machine learning [2] methodology where models are trained on local data from distributed devices while maintaining data privacy [3]. Each client trains its local model using its data and uploads parameters to a central server, aggregating them into a global model and distributing it back to the clients. In centralized FL, a single central server handles communications with all clients, aggregates model updates, and manages client coordination. Nevertheless, as the number of clients in the network grows, this centralized architecture raises the following significant challenges: scalability limitations [4], security risks [5], and single points of failure [6].

DFL [6], [7], [8] addresses these scalability limitations by eliminating the central server and enabling direct communication among clients. Instead of relying on a single server to coordinate communication and aggregation, DFL allows clients to communicate directly with their neighbors or localized groups. This decentralized communication paradigm

The associate editor coordinating the review of this manuscript and approving it for publication was Juan Wang.

reduces the overall communication overhead and distributes the computational burden across the network. For example, DFL [9] can efficiently scale to accommodate a larger number of clients without suffering from the bottlenecks inherent in FL. The security risks [10] in FL arise from the central server being a prime target for attacks, which could reveal the global model or access sensitive client data. DFL mitigates this risk by distributing model parameters among all clients, making it more difficult for attackers to compromise the system. In addition, DFL avoids the single point of failure in FL [11]; the network can continue operating if one client fails, ensuring greater robustness and resilience.

DFL is a promising field of research with applications in healthcare [12], [13], industry 4.0 [14], mobile services [15], military [16], and vehicles [17], where researchers have reviewed its applicability in specific contexts. DFL can facilitate decentralized learning across multiple clients in practical applications, such as intelligent CCTV systems in smart cities, IoT devices in smart homes, and hospital medical systems while preserving privacy. These clients often operate in environments with direct end-to-end connectivity, making a fully connected topology a natural choice for modeling. Nevertheless, real-world constraints like bandwidth limitations and varying application requirements often necessitate more selective connectivity, resulting in k-connected topology. These fully connected and k-connected topologies are distinct tradeoffs between communication efficiency and model accuracy. Thus, this study evaluates the performance of DFL in networks, focusing on these two types of topologies.

In addition to the impact of topologies, existing studies on [9], [18], [19], [20], [21], and [22] are particularly limited in terms of the effects of heterogeneous computing resources and communication bandwidth. Heterogeneous computing resources involve the diverse capabilities of devices participating in a distributed system. This encompasses variations in processing power (CPU), memory availability (RAM), storage capacity, network connectivity, and power availability (battery level) [23]. For example, some devices might have powerful multi-core CPUs, while others might have simpler, single-core processors. Similarly, memory and storage capacities can differ significantly across devices. This disparity in computing power, and variations in memory and storage capacities, can significantly affect the speed and efficiency of model training in DFL.

A heterogeneous communication bandwidth [24] refers to the variability in network connection quality among devices in a distributed system. This includes differences in bandwidth, latency, reliability, and traffic patterns [25] regarding diverse network connectivity such as 4G, 5G, and Wi-Fi. For example, some devices might be connected through high-bandwidth fiber optic connections, while others might rely on slower and less reliable satellite connections [26]. These variations in network characteristics can lead to delays, inconsistencies, and disruptions in the DFL training process.

To the best of our knowledge, this is the first study to thoroughly evaluate the impact of heterogeneous computing resources and communication bandwidth in DFL across diverse network topologies. In summary, this study offers the following contributions:

- Focus on practical implementation: This study moves beyond theoretical frameworks and delves into the practical implementation of DFL, considering real-world constraints and challenges, such as limited computing resources, varying bandwidths, and potential network topology.
- Empirical evaluation of heterogeneity: A thorough evaluation of DFL performance was conducted under heterogeneous computing resources and communication bandwidth conditions, providing concrete practical lessons of their impact on the key performance metrics, including the convergence round, processing time, convergence time, and communication costs.
- Analysis of specific topologies: This study focused on fully connected and k-connected topologies, providing a detailed analysis of their performance characteristics and tradeoffs in the context of heterogeneity. This provides valuable insights for choosing the correct topology for specific DFL applications.

The remainder of this paper is organized as follows: Section II reviews the related work on the DFL challenges, highlighting existing research and identifying gaps this study addresses. Section III provides a detailed overview of the DFL framework and algorithm, including the specific network topologies used for implementation, client resource configurations, and evaluation metrics. Section IV presents a detailed analysis of the performance evaluation results, summarizing the key lessons learned by investigating the influence of network topologies, resource heterogeneity, and communication bandwidth. Section V discusses the potential limitations of our DFL framework. Finally, section VI concludes the study and outlines potential avenues for future research.

II. RELATED WORKS

This section reviews related studies that evaluated DFL in practice. Several survey papers, including those by Martínez Beltrán, et al. [18], Yuan et al. [6] and Hallaji et al. [27], provide overviews of DFL, covering various aspects such as architectures, algorithms, communication protocols, and challenges.

Daily et al. [9] presented GossipGraD, a gossip-based DFL algorithm that leverages peer-to-peer communication to reduce communication overheads. Although GossipGraD has the potential of DFL for scalable learning, it focuses primarily on communication efficiency but does not explicitly investigate the impact of different network topologies and heterogeneous computing resources on the performance of DFL.

TABLE 1. Comparative analysis of related works in DFL.

Aspect		Daily et al. [9]	Liao et al. [19]	Palmieri et al. [20]	Takezawa et al. [22]	Hao et al. [28]	Our work
Type of Research		Algorithm	Framework	Evaluation	Framework	Framework	Evaluation
Network Topologies		✓	✗	✓	✗	✗	✓
System Heterogeneity	Computing Resources	✗	✓	✗	✗	✗	✓
	Communication Bandwidth	✓	✗	✗	✗	✓	✓
Practical Implementation		✗	✗	✗	✓	✗	✓

Liao et al. [19] proposed an adaptive configuration framework for DFL to address resource heterogeneity, aiming to improve efficiency and convergence speed. On the other hand, their work focused primarily on resource heterogeneity and lacked a detailed analysis of specific performance metrics across diverse network topologies.

Building on this, Palmieri et al. [20] examined the impact of the network topology on the performance of DFL, highlighting the significance of global centrality metrics and the challenges of knowledge transfer in heterogeneous networks. Nevertheless, their focus lay primarily on data distribution and network structure without explicitly considering the impacts of heterogeneous computing resources and communication bandwidth.

Reddi et al. [21] proposed a framework for dynamically adjusting the learning rate and communication frequency in FL based on the network conditions and client participation. Although this approach effectively improved convergence and communication efficiency and highlighted the influence of network conditions and client participation on FL performance, it focused primarily on centralized settings. In addition, it did not explicitly address the challenges of communication heterogeneity inherent in DFL, particularly the complexities of varying bandwidth limitations and network topologies.

Takezawa et al. [22] focused on designing communication-efficient network topologies for DFL. They proposed novel topologies based on expander graphs and hypercubes, showing faster convergence rates than traditional topologies. Nevertheless, their analysis assumed homogeneous client capabilities and network bandwidths, which may not hold in practical DFL deployments where resource limitations and varying communication speeds can significantly affect performance.

Hao et al. [28] focused on network communication under unreliable conditions in DFL systems. They proposed a robust protocol to address packet loss and delays, improving convergence stability in decentralized systems. Nevertheless, their work did not account for bandwidth heterogeneity, a significant challenge in practical DFL deployments, nor did it explore the interplay between the heterogeneous bandwidth and network topologies.

Table 1 compares this study with key related works in DFL, focusing on their research scope, network topologies, and considerations of resource and bandwidth heterogeneity. Although existing studies [20], [21], [22], [28] investigated communication protocols and network topologies, they have largely overlooked the combined effects of heterogeneous computing resources and bandwidth limitations. In contrast, this study systematically evaluates the impacts of network topologies, heterogeneous computing resources, and communication bandwidth through practical implementation and evaluation. These factors influence the performance of DFL systems significantly influenced, with each playing a distinct role in shaping system performance. These insights offer valuable lessons for designing resource-aware strategies in DFL systems.

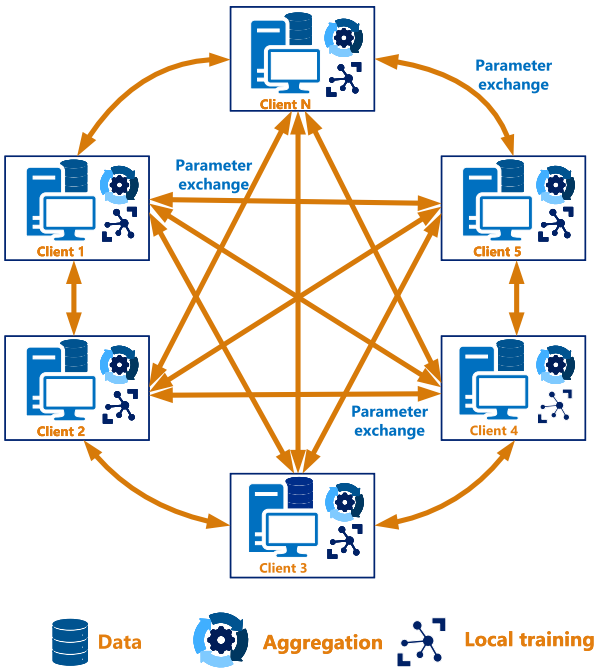


FIGURE 1. Architecture of DFL framework.

III. METHODOLOGY

This section, outlines the methodology behind this evaluation, aiming to provide a thorough understanding of the DFL algorithm. The effects of various factors on DFL performance are explored and evaluated. Accordingly, different network topologies and client resources were set up and the performance metrics were defined to measure the performance of the DFL algorithm across diverse settings.

A. ARCHITECTURE OF THE DFL FRAMEWORK AND DFL ALGORITHM

Figure 1 presents the architecture and workflow of the DFL framework. Unlike traditional FL, with a central server coordinating the process, DFL enables direct peer-to-peer communication and model exchange between clients.

No central server is involved in the DFL setup shown in the figure. Instead, there are multiple clients, each with a local dataset and model. The following are the detailed processes for each client:

- **Initialization:** Each client initializes their local model using their private local dataset.
- **Local model training:** In each communication round, clients participating in the training process utilize their local model to train on their local data for a specified number of epochs.
- **Neighbor parameter exchange:** After the local training phase, clients exchange their current local model parameters with each other. Each client sends its model parameters to its neighbors and receives model parameters from them.
- **Neighbor parameter aggregation:** When a client receives the complete set of model parameters from its neighbors, it launches several tasks, including aggregation, analysis, and evaluation, to update its model with improved parameters. The effectiveness of aggregating parameters can be explored from a smaller subset of neighbors instead of all k-connected neighbors. In addition, incorporating weighting schemes based on client capabilities or data quality or investigating alternative aggregation methods beyond simple averaging could enhance the accuracy and efficiency of the DFL process.

This cycle of local training, neighbor parameter exchange, and neighbor parameter aggregation is repeated for multiple communication rounds until the global model converges or reaches the target accuracy.

This study uses the Federated Averaging (FedAvg) [29] algorithm as the aggregation method. FedAvg is a decentralized approach where clients average their model updates with their direct neighbors, facilitating convergence across the network. It was selected because it is a foundational and widely used algorithm for evaluating the performance of DFL systems. Its proven efficiency in handling decentralized environments makes it well-suited for DFL, where direct peer-to-peer communication is critical. FedAvg involves a weighted average of the individual clients' losses to find the optimal model parameter w minimizing the global loss function, as shown below:

$$\min_{w} \mathcal{L}(w) = \sum_{n=1}^{K} \frac{d_n}{d} \mathcal{L}_n(w^n) \tag{1}$$

Equation (1) expresses the overall global loss; K is the number of participating clients (client and its neighbors), d_n is the number of training samples of client n, and $d = \Sigma_{n=1}^{K} d_n$ is the total number of training samples from the participating clients (client and its neighbors). The local loss value function $\mathcal{L}_n(w^n)$ for each client is calculated as follows:

$$\mathcal{L}_n(w^n) = \frac{1}{d_n} \sum_{i=1}^{d_n} \mathcal{L}_i(w^n) = \frac{1}{d_n} \sum_{i=1}^{d_n} l(x_i, y_i, w^n) \tag{2}$$

Equation (2) expresses the local loss value function for client n, $\mathcal{L}_i(w^n)$ is the loss value function for sample i, x_i represents the data feature for sample i, y_i stands for the data label on sample i, and w^n is the weight value for client n. Typically, $\mathcal{L}_i(w^n) = l(x_i, y_i, w^n)$, where the loss on the sample (x_i, y_i) is calculated using the weight value w on client n.

Algorithm 1 shows the operation of DFL. At the beginning of the process, each participating client i initializes its local model w_i. The algorithm then selects the nearby neighbors $\mathcal{N}$ from the pool of all participating clients to establish the specified network topology. The list of neighbors $\mathcal{N}$ could include all clients within the system or a subset of specific clients based on the selected topology.

Algorithm 1 DFL Algorithm

w_i^r: local model parameters of the client i at round r.
R: number of communication rounds.
d_i: the number of training samples of the client i.
$\mathcal{N}$: the number of neighbor clients.

1: Initialize each client i with its local model w_i^r and its own data d_i.
2: Each client i selects its own neighbors as a list of $\mathcal{N}$.
3: **for** $r = 1$ to R **do**
4: Train local model w_i^r on local data for a specified number of epochs.
5: **for** each neighbor j in $\mathcal{N}$ **do**
6: Send w_i^r to client j.
7: Receive w_j^r from client j.
8: **end for**
9: $w_i^{r+1} \leftarrow \frac{d_i w_i^r + \sum_{j=1}^{\mathcal{N}} d_j w_j^r}{d_i + \sum_{j=1}^{\mathcal{N}} d_j}$
10: **end for**

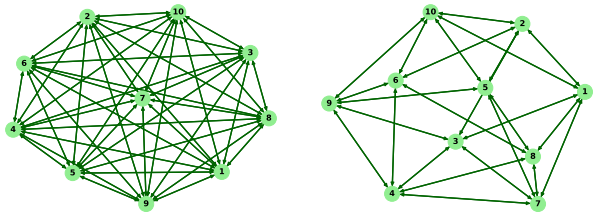
The algorithm progresses through a series of communication rounds R, ultimately achieving convergence and reaching the target accuracy across all participating clients at round R. Within each round, each participating client uses its local model w_i to train on its local data for several epochs. They then send and receive the model parameters with their list of neighbors as specified in $\mathcal{N}$. Each client then launches several tasks, such as aggregation, analysis, and evaluation, to update its model with improved parameters. For this aspect of the present study, this study utilizes FedAvg, which aggregates the models by calculating a weighted average from the received parameters. In the numerator, $d_i w_i^r + \sum_{j=1}^{\mathcal{N}} d_j w_j^r$, this part combines the knowledge from the current client and its neighbors. The model from each client is weighted by the amount of data it contains, giving more influence to clients with more data. This summation accumulates the knowledge learned from each neighbor, weighted by the amount of data contribution. The denominator $d_i + \sum_{j=1}^{\mathcal{N}} d_j$ is simply the total amount of data involved in the aggregation, ensuring the final result is a proper average of all the models. The formula calculates the weighted average of the model parameters from the current client and its neighbors, where the weights are determined by the amount of data each client possesses. Each

client then updates its local model based on these aggregated parameters before proceeding to the next round.

B. NETWORK TOPOLOGIES

Network topology is one of the important aspects affecting the performance of DFL because each client only communicates and aggregates data with its neighbors according to the underlying network structure. Nedić et al. [30] outlined 11 network topologies, such as fully connected, ring, tree, and star network topologies and their variations. Different network topologies affect the communication overhead, convergence speed, fault tolerance, and overall system robustness. Thus, careful consideration and analysis are essential when designing and deploying DFL frameworks to ensure optimal performance in various network environments.

Figure 2 presents the connectivity graphs of two network topologies used in this paper: the fully connected topology and the k-connected topology, which represent popular choices with a fixed number of connections. It is important to note that this figure provides specific examples for visualization purposes, and the DFL algorithm in our framework is not limited to these topologies. In fact, our framework is designed to work with any network topology commonly encountered in practical applications.



(a) Fully connected topology (b) 5-connected topology

FIGURE 2. The connectivity graph of (a) Fully connected topology, (b) One of the k-connected topology forms.

Figure 2a presents the fully connected topology, the most commonly used configuration in DFL. In this topology, each client maintains direct links with all other clients, enabling comprehensive parameter exchange across the network. Fully connected topologies are ideal for high-bandwidth, low-latency environments where communication overhead is less of a concern. Despite this, these topologies lead to significant processing and communication overheads because each client must send and receive model parameters from every other client [31]. In addition, the need for each client to establish links with all others poses scalability challenges, particularly in large-scale deployments.

Figure 2b shows the k-connected topology, which limits the number of connections for each client to k links with other clients. For example, each client maintains exactly 5 connections during training in the 5-connected topology depicted here. This topology reduces transmission costs compared to the fully connected topology, making it more practical for IoT devices and other systems with limited bandwidth and computational resources.

In summary, the fully connected topology facilitates direct communication among all clients, enabling parameter exchange but leading to higher communication overhead, particularly in large-scale networks [32]. In contrast, the k-connected topology reduces communication costs by limiting the number of client connections while maintaining sufficient robustness, making it suitable for resource-constrained and large-scale environments [32], [33].

This study chose the fully connected and k-connected topologies out of 11 network topologies because of their relevance in practical DFL applications. For example, intelligent CCTV systems in smart city environments, IoT devices in smart homes, and medical systems in hospitals can be modeled as DFL applications, where each entity (CCTV, IoT device, or hospital) acts as a client. In these applications, clients are assumed to have end-to-end connectivity, making the underlying network suitable for a fully connected topology. On the other hand, the communication overhead inherent in this topology means that some connections must be pruned to improve performance, leading to the k-connected topology. Accordingly, this study investigates how varying the connectivity levels from fully connected to more sparsely connected systems affects performance, providing insights into the tradeoff between communication efficiency and model accuracy and offering guidance on optimizing these practical DFL systems.

C. CLIENT RESOURCES

The performance of a DFL system can be significantly influenced by the heterogeneity of client resources participating in the training process. In practical DFL applications, such as those deployed across edge devices, IoT networks, or mobile systems, the participating clients commonly have significantly different hardware capabilities. Heterogeneous computing resources of clients refer to the variations in CPU power, memory, and energy availability. Similarly, communication bandwidth can vary widely between clients because of differences in network infrastructure, connectivity options (e.g., 4G, 5G, Wi-Fi, or satellite links), and geographic location. These variations reflect heterogeneous communication bandwidth, which may cause some clients to experience delays or reduced data transfer rates. This study aims to provide insights directly applicable to deploying DFL systems in diverse and resource-constrained environments by incorporating these realistic variations. Three distinct setups were considered to evaluate the impact of client resource heterogeneity thoroughly.

- **Homogeneous resources:** All participating clients have identical computing resources (CPU and RAM) and communication bandwidth in this baseline setup. This setup serves as a reference point for evaluating the fundamental performance of the DFL framework and

analyzing the impact of different network topologies without the influence of resource heterogeneity.

- **Heterogeneous computing resources:** Clients exhibit different computing resource capabilities in this setup. Specifically, a subset of clients is designated as low-resource with fewer CPU cores and limited memory than other participants. In addition, the number of low-resource clients in the system is increased gradually, allowing them to analyze the effects of varying degrees of computing heterogeneity in DFL performance.
- **Heterogeneous communication bandwidth:** In this setup, clients differ in their communication bandwidth capabilities. A subset of clients is designated as low-bandwidth clients with a significantly lower bandwidth than others. The effects of communication heterogeneity on the efficiency of DFL systems were examined by varying the number of low-bandwidth clients.

By considering heterogeneity in client computing resources and communication bandwidth, this study systematically evaluates its impact on the performance of DFL systems. These three setups offer a structured framework to analyze how variations in resource availability and network conditions affect the system efficiency. This approach not only highlights the challenges posed by resource and bandwidth diversity but also provides valuable insights into optimizing DFL systems for practical deployment. This evaluation focuses on the key performance metrics, including the convergence speed, processing time, and communication costs, as detailed in the following subsection.

D. EVALUATION METRICS

The following evaluation metrics are used to evaluate the performance of the DFL framework under various network topologies and client resources: convergence round, processing time per round, convergence time, communication cost per round, and total communication cost. These metrics were selected because they capture both computational and communication aspects, which are crucial in DFL environments where resource constraints and network dynamics are highly heterogeneous. Each metric provides specific insights into different characteristics of system behavior, efficiency, and resource utilization. The following provides a detailed explanation of each metric associated with the reason for its selection, supported by the relevant literature.

- **Convergence round:** This metric represents the number of communication rounds required for all participating clients to reach the target accuracy [34]. It directly measures the training speed of the algorithm. Accordingly, this metric was selected to quantify how quickly a DFL system reaches an optimal solution under limited computing resources or communication bandwidth.
- **Processing time per round:** This metric quantifies the duration required to complete a single round of the training process [34]. This includes the time taken for local model training on each client's dataset and the time required to exchange and aggregate model parameters with neighboring clients. This metric was chosen to highlight the computational cost and the impact of computing heterogeneity on the performance of the DFL system, as nodes with different computational capabilities may significantly affect the processing time.
- **Convergence time:** The convergence time measures the total elapsed time from the start of training until all participating clients in the network reach the target accuracy [35]. Convergence is achieved when every client reaches this target accuracy, regardless of its computing resources or network conditions. This metric provides an overall measure of how long it takes for a DFL system to train models to the desired accuracy, which is especially important for deploying such systems in environments with varying processing speeds and network conditions.
- **Communication cost per round:** This metric quantifies the normalized bandwidth consumption associated with exchanging model parameters among clients during a single training round [36]. In a decentralized setting, exchanging model parameters among clients can result in significant communication overhead, particularly in setups with limited network bandwidth or large model parameter sizes. This metric was selected to measure how efficiently the system uses network resources because communication delays and costs can significantly impact the overall system performance. The formula for communication cost per round is given as follows:

$$C_r = \sum_{i=1}^{N} S \times \mathcal{N}_k \tag{3}$$

where N is the number of clients participating in round r, S is the model size, and $\mathcal{N}_k$ is the number of network connections.

- **Total communication cost:** This metric is built upon the communication cost per round, representing the cumulative bandwidth consumption over the entire training process until convergence. The metric was selected because it provides insights into the total network load imposed by the DFL system, which is essential for evaluating the scalability and resource efficiency. In large-scale DFL deployments, understanding the total communication cost helps determine the feasibility of using such systems in real-world network environments with bandwidth constraints. The long-term impact of communication overheads on the system performance can be assessed using this metric, allowing the system to be optimized for better efficiency. The total communication cost is calculated as follows:

$$C = \sum_{r=1}^{R} C_r \tag{4}$$

where R is the convergence round and C_r is the communication cost per round.

IV. PERFORMANCE EVALUATION RESULTS

This section analyzes the performance of our DFL framework under three different setups: network topologies, heterogeneous computing resources, and heterogeneous communication bandwidth. The experiments were conducted using three datasets: Fashion MNIST [37], CIFAR-10, and CIFAR-100 [38], each representing varying complexity levels. For all experiments, a convolutional neural network (CNN) [39] model was used because of its effectiveness in image classification tasks. The model was trained using the stochastic gradient descent (SGD) optimizer, with a learning rate of 0.01, a batch size of 32, and six epochs per round for each client. Each experiment was repeated twice, with the means and standard deviations calculated for all measured outcomes. The target accuracy for the experiments is meticulously set to 0.9, 0.65, and 0.3 for Fashion MNIST, CIFAR-10, and CIFAR-100, respectively, carefully selected to balance the model performance and training efficiency across different network topologies. These thresholds were identified by training the DFL system using the 2-connected topology, representing the worst-case setup, because each client only connects to two neighbors. The model was trained for multiple rounds under this topology, and the highest accuracy that could be achieved for each dataset was identified. On the other hand, in real-world DFL environments, clients may not always reach the target accuracy because of various factors such as heterogeneity in hardware, network bandwidth, or local data distribution. Therefore, these thresholds represent an ideal target that may not be consistently achievable in all settings, but serve as a meaningful and practical benchmark for understanding the tradeoffs between accuracy, training time, and resource constraints across different DFL evaluations.

This section is divided into multiple subsections, each focusing on specific aspects in more detail within our DFL framework. To provide a structured approach, the impact of network topologies was first evaluated, examining the scalability and performance of fully connected and k-connected networks under a homogeneous resources setup. The impact of heterogeneous computing resources was then assessed, focusing on the effects of resource-constrained clients on DFL performance. Finally, this study analyzed the impact of heterogeneous communication bandwidth, examining how variations in communication bandwidth affect the system efficiency. These evaluations aim to provide valuable insights into the critical factors affecting the performance of our DFL system.

A. IMPACT OF NETWORK TOPOLOGIES

In this evaluation, all participating clients are homogeneous regarding client resources (computing resources and communication bandwidth) and the number of training samples. Each client has a 2-core CPU, 8GB RAM, and 1 Gbps communication bandwidth, ensuring all clients have equal computational and networking capabilities. The evaluation is divided into two parts. The first analyzes the effects of connectivity in fully connected and k-connected topologies on multiple datasets. The second focuses on scalability, examining how increasing the number of clients in selected k-connected topologies affects the local training time, convergence time, processing time, and communication costs.

1) TOPOLOGY ANALYSIS: EFFECTS OF CONNECTIVITY IN FULLY CONNECTED AND K-CONNECTED TOPOLOGIES

Table 2 provides a detailed analysis of the impact of network topologies on the DFL performance in homogeneous clients with diverse datasets: Fashion MNIST, CIFAR-10, and CIFAR-100. The metrics used to evaluate the system performance include convergence round, processing time per round, convergence time, cost per round, and total communication cost. This analysis explores the influence of different connectivity levels on the system behavior and performance, highlighting the tradeoffs between communication efficiency, convergence speed, and scalability across datasets of varying complexity.

For the Fashion MNIST dataset, the 2-connected topology takes 123 rounds to converge, with a processing time of 181.52 seconds per round, resulting in a total convergence time of 406.80 minutes. As the number of connections between clients increases, the convergence round generally decreases, reaching a minimum of 99 rounds for the 5-connected topology with a total convergence time of 376.87 minutes. Beyond the 5-connected topology, the number of convergence rounds increases, with the fully connected topology requiring 116 rounds to converge and a much longer convergence time of 501 minutes. This is attributed to too few connections restricting the ability of the model to capture underlying data patterns, resulting in less efficient learning. Conversely, while increasing connections enhance knowledge sharing, excessive connections lead to communication overhead that produces synchronization delays and additional computation costs, ultimately increasing convergence rounds.

The higher number of connections not only affects the convergence rounds and convergence time but also significantly impacts the processing time. For example, the 2-connected topology takes 181.52 seconds per round, while the fully connected topology takes significantly longer at 267.98 seconds per round. The cost per round also increases with connectivity, increasing from 0.13 GB for the 2-connected topology to 0.58 GB for the fully connected topology. Consequently, the total communication cost follows the same trend, with the 2-connected topology having a total cost of 16.37 GB while the fully connected topology has a substantially higher cost of 67.83 GB. This increase in processing time and communication cost is attributed to the added complexity and synchronization overhead required for each client to communicate with more neighbors. As the number of connections increases, the

TABLE 2. Impact of the network topologies on the DFL performance in homogeneous systems with diverse datasets (Mean ± Standard deviation). The target accuracy for Fashion MNIST, CIFAR-10, and CIFAR-100 are 0.9, 0.65, and 0.3, respectively.

Topology	Convergence round	Processing time (second)	Convergence time (minute)	Cost per round (GB)	Total cost (GB)	Dataset
2-connected	123 ±3	181.52 ±2.29	406.80 ±13.20	0.13	16.37 ±0.39	Fashion MNIST
3-connected	119 ±3	201.93 ±2.86	397.13 ±6.60	0.19	23.20 ±0.49	
4-connected	112 ±2	208.47 ±2.07	393.60 ±13.08	0.26	29.11 ±0.52	
5-connected	99 ±5	228.41 ±2.46	376.87 ±10.76	0.32	32.16 ±1.46	
6-connected	104 ±6	239.04 ±1.19	379.25 ±13.12	0.39	40.35 ±2.14	
7-connected	110 ±4	245.09 ±4.09	447.95 ±16.20	0.45	49.80 ±1.59	
8-connected	112 ±4	255.21 ±2.35	475.80 ±18.90	0.52	58.22 ±3.31	
Fully connected	116 ±3	267.98 ±2.01	501.00 ±17.10	0.58	67.83 ±3.08	
2-connected	128±6	172.57 ±0.86	475.90 ±18.36	0.17	21.57 ±1.01	CIFAR-10
3-connected	81 ±2	189.86 ±8.91	259.02 ±12.90	0.25	20.34 ±0.38	
4-connected	66 ±4	206.81 ±0.63	228.86 ±7.54	0.34	22.07 ±1.18	
5-connected	58 ±1	220.09 ±0.04	212.56 ±3.44	0.42	24.43 ±0.42	
6-connected	53 ±2	231.64 ±0.26	219.84 ±16.44	0.51	26.54 ±0.76	
7-connected	69 ±3	240.05 ±2.49	276.03 ±9.03	0.59	40.69 ±1.77	
8-connected	68 ±4	250.64 ±0.66	281.96 ±13.85	0.67	45.49 ±2.36	
Fully connected	76 ±6	269.43 ±9.05	343.62 ±16.38	0.76	59.52 ±4.17	
2-connected	245 ±3	178.72 ±0.15	728.58 ±7.38	0.17	42.07 ±0.43	CIFAR-100
3-connected	166 ±5	196.76 ±1.50	596.40 ±15.60	0.26	42.85 ±1.29	
4-connected	164 ±4	207.91 ±1.71	560.40 ±2.10	0.34	56.27 ±1.20	
5-connected	138 ±5	221.84 ±0.34	502.20 ±6.84	0.43	59.15 ±1.94	
6-connected	134 ±4	232.19 ±0.83	522.15 ±10.15	0.52	68.92 ±1.81	
7-connected	145 ±3	241.09 ±0.95	615.00 ±6.45	0.60	87.33 ±1.81	
8-connected	155 ±3	251.33 ±0.31	677.40 ±16.20	0.69	106.69 ±2.06	
Fully connected	162 ±5	257.75 ±1.84	722.40 ±17.10	0.77	125.06 ±3.48	

system must handle more frequent parameter exchanges and data transfers, resulting in longer processing times and higher communication bandwidth usage.

A similar trend is observed with the CIFAR-10 and CIFAR-100 datasets, as seen with the Fashion MNIST dataset. In the case of CIFAR-10, the 2-connected topology requires 128 rounds to converge with a processing time of 172.57 seconds per round, resulting in a total convergence time of 475.9 minutes. As the number of connections increases, the convergence rounds decrease and reach a minimum of 53 rounds for the 6-connected topology with a total convergence time of 219.84 minutes. Beyond the 6-connected topology, the number of convergence rounds increases again, with the fully connected topology needing 76 rounds and taking a total of 343.62 minutes to converge. Similarly, in CIFAR-100, the 2-connected topology takes 245 rounds to converge, while the 6-connected topology minimizes convergence rounds at 134. On the other hand, the fully connected topology requires 162 rounds, resulting in a longer total convergence time of 722.40 minutes.

In summary, while increasing connectivity initially enhances convergence by facilitating better communication between clients, beyond a certain point, the added communication overhead and synchronization delays outweigh these benefits. The 5-connected, and 6-connected topologies consistently provide an optimal balance between fast convergence and moderate communication cost across Fashion MNIST, CIFAR-10, and CIFAR-100. This means selecting an optimal network topology for a DFL system where both convergence speed and communication efficiency are critical. In contrast, the fully connected topology has significantly higher processing times and communication costs without yielding proportional improvements in convergence speed, highlighting the need for careful optimization of the network topology.

2) SCALABILITY ANALYSIS: IMPACT OF NUMBER OF CLIENTS IN SELECTED K-CONNECTED TOPOLOGIES

This subsection focuses on the scalability of DFL by scaling the number of clients using the CIFAR-10 dataset with a target accuracy of 0.63. The target accuracy was determined using a worst-case setup where the DFL system was trained with 30 clients, representing the maximum number of clients and the smallest data sample per client in this experiment. In addition, this study uses a 5-connected topology, which was identified as one of the most efficient in previous experiments. The number of clients is increased from 10 to 30, with the dataset distributed evenly across all clients to ensure each has the same number of non-overlapping data samples. The primary purpose of this experiment is to understand the impact of scaling the number of clients on various performance metrics, such as training time, processing time, convergence round, total convergence time, and communication cost.

Table 3 shows that as the number of clients increases, the number of data samples per client decreases, resulting in shorter training times such as 100.74 seconds for 10 clients and 45.37 seconds for 30 clients, along with reduced processing times per round from 220.09 seconds to 191.04 seconds. On the other hand, the system requires more convergence rounds as the number of clients increases, rising from 35 rounds for 10 clients to 256 rounds for 30 clients. This leads to a sharp increase in the overall convergence time, growing from 2.14 hours for 10 clients to

TABLE 3. Performance of the homogeneous system when scaling up the number of clients under the CIFAR-10 dataset with a target accuracy of 0.63.

Scalability	10 clients	20 clients	30 clients
Client data samples	5000	2500	1666
Training time (s)	100.74 ±0.22	58.89 ±0.44	45.37 ±0.32
Processing time (s)	220.09 ±0.04	210.80 ±0.62	191.04 ±0.10
Convergence round	35 ±2	161 ±2	256 ±5
Convergence time (h)	2.14 ±0.12	9.40 ±0.12	13.56 ±0.31
Cost per round (GB)	0.42	0.84	1.26
Total cost (GB)	14.74 ±0.84	135.21 ±1.26	322.86 ±6.95

13.56 hours for 30 clients. In addition, the communication costs per round increase with more clients, from 0.42 GB to 1.26 GB, while the total communication costs rise dramatically, from 14.74 GB for 10 clients to 322.86 GB for 30 clients. These results suggest that increasing the number of clients in a homogeneous system with a non-overlapping data distribution reduces the training and processing time per round. On the other hand, it significantly increases the number of rounds required for convergence. This leads to longer overall convergence times and higher communication overhead. Therefore, careful consideration must be given when scaling the number of clients to balance efficiency and resource consumption in a DFL system.

B. IMPACT OF HETEROGENEOUS COMPUTING RESOURCES

This subsection evaluates the impact of heterogeneous computing resources on the performance of a synchronous DFL system using the CIFAR-10 dataset. This study investigates how resource disparities among clients, particularly slower processing speeds in low-resource clients, affect the overall convergence behavior, synchronization process, and communication efficiency of the system. All clients have the same data distribution, with 5000 samples for each, and the target accuracy is set to 0.65.

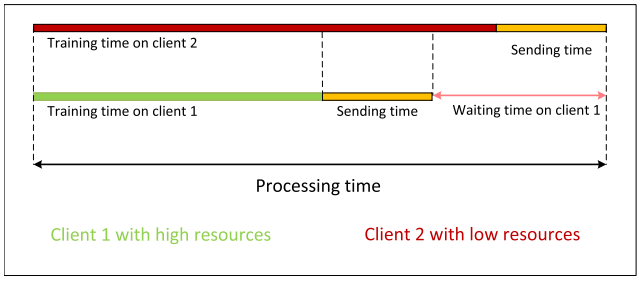


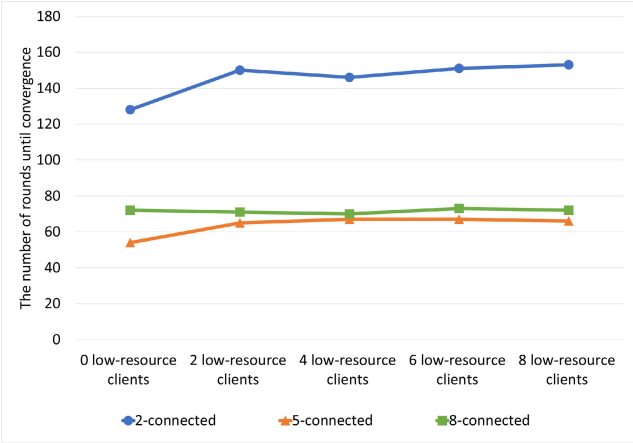
FIGURE 3. Synchronous DFL with heterogeneous computing resources clients.

Figure 3 presents that among two neighboring clients, client 1 with high computing resources completes the training first and then sends the parameter to client 2 with low computing resources. Therefore, client 1 has to wait a certain amount of time, which is defined as the waiting time until client 2, with low computing resources, completes the neighbor parameter aggregation before moving on to the next round. This waiting time highlights is a key characteristic of synchronous DFL. Consequently, the processing time in each round for the DFL training is affected by the client having the lowest computing resources in the heterogeneous system. Hence, all participating clients must complete their local training and exchange their parameters with their neighbors before moving on to the next round.

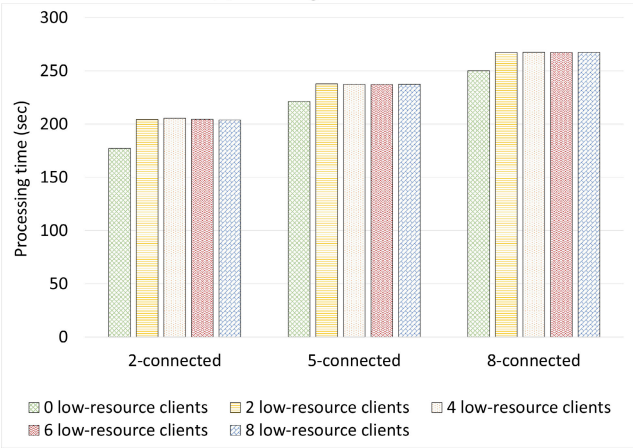
Two types of clients were defined to analyze the impact of client heterogeneous computing resources high-resource clients with 4 cores of CPU and low-resource clients with 1 core of CPU, gradually adding clients with low computing resources. Subsequently, this evaluation was segmented into 5 test cases corresponding to the growing number of low-resource clients, namely 0 low-resource clients, 2 low-resource clients, 4 low-resource clients, 6 low-resource clients, and 8 low-resource client cases. Valuable insights into the performance of the system and behavior in heterogeneous systems can be gained by systematically adjusting the number of low-resource clients.

Figure 4a shows the convergence round required for the 2-connected, 5-connected, and 8-connected topologies. In evaluations with heterogeneous computing resources, the convergence rounds remain fairly constant for highly connected topologies (5-connected and 8-connected), even as the number of low-resource clients increases. For example, the 8-connected topology consistently requires 70 rounds across all test cases. In contrast, the 2-connected topology exhibits more variability and fluctuation in convergence rounds, indicating that fewer connections lead to less stable convergence. This is because, in more connected topologies, each client can aggregate parameters from a larger and more diverse set of neighbors, which helps balance the effects of clients with varying computing resources. This leads to a more uniform and efficient convergence process. In less connected topologies, however, the limited number of connections means that the impact of low-resource clients is more noticeable, resulting in greater variability and less stable convergence.

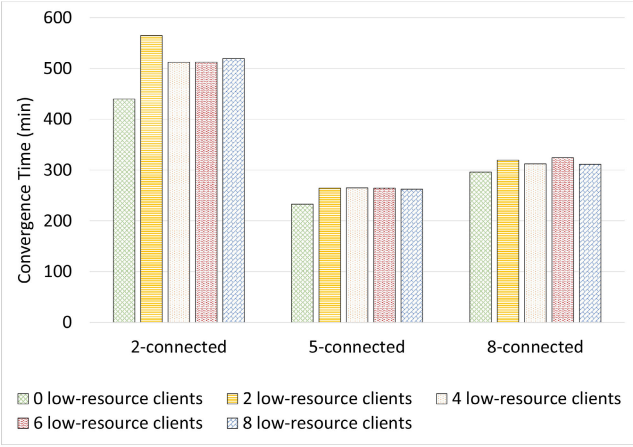
Figure 4b illustrates that the heterogeneity of computing resources among clients significantly impacts the processing time per round for all three topologies. The processing time increases as the number of connections increases. In particular, the 2-connected topology consistently has the lowest processing time per round, while the 8-connected topology has the highest. For example, the processing time for the 2-connected topology is 205 seconds, while it increases to 270 seconds for the 8-connected topology when low-resource clients are introduced. The processing time increases with the existence of low-resource clients, regardless of the number of low-resource clients in the system. This is because clients with higher computing resources must wait for clients with lower computing resources. In this experiment, The clients synchronize their data samples so that clients with the same resources can finish processing simultaneously. Therefore, the time required for all clients to synchronize becomes



(a) Convergence round



(b) Processing time per round



(c) Convergence time

FIGURE 4. Convergence round, processing time per round, and convergence time in the heterogeneous computing resources.

dominated by the lowest-resource clients, irrespective of the number of low-resource clients.

Figure 4c presents the convergence time across all test cases for the three topologies, combining the processing time per round and the convergence round. The 2-connected topology consistently has the highest convergence time, with more than 500 minutes with low-resource clients, almost twice that of the 5-connected topology, and 1.8 times that of the 8-connected topology, regardless of the presence of low-resource clients. This highlights the fact that while a larger number of connections can initially contribute to faster convergence, beyond a certain threshold, further increasing the connectivity does not significantly improve the convergence time.

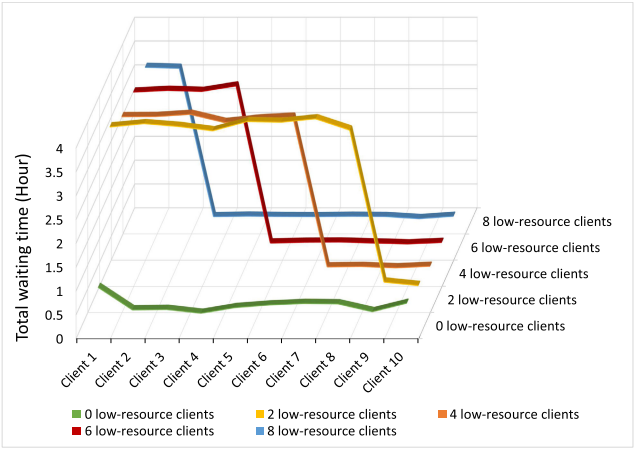
Figure 5 shows the total waiting time for each client in 2, 5, and 8-connected topologies under various test cases involving different numbers of low-resource clients, where the total waiting time refers to the cumulative time a client spends waiting during each round until convergence. In Figure 5a, the 2-connected topology shows that in the test case with no low-resource clients, the total waiting time for each client is nearly zero in the test case with no low-resource clients. On the other hand, when low-resource clients are introduced, clients with more computing resources have longer waiting times when low-resource clients are introduced. For example, in the test case with 2 low-resource clients, clients 9 and 10 (low-resource clients) have almost no waiting time, while clients 1 to 8 (the high-resource clients) experience a waiting time of approximately 3 hours. In the test case with 8 low-resource clients, clients 3 to 10 (low-resource clients) have minimal waiting time, and clients 1 and 2 (high-resource clients) must wait for the low-resource clients to finish their processes before moving on to the next round. As mentioned above, the 5 and 8 connected topologies in Figures 5b and 5c also follow the same pattern. The degree of resource heterogeneity directly influences the disparity in waiting times among clients, with higher-resource clients experiencing longer waiting periods. This disparity is a key factor in increasing the processing time. Therefore, optimizing performance necessitated addressing these waiting times be addressed.

C. IMPACT OF HETEROGENEOUS COMMUNICATION BANDWIDTH

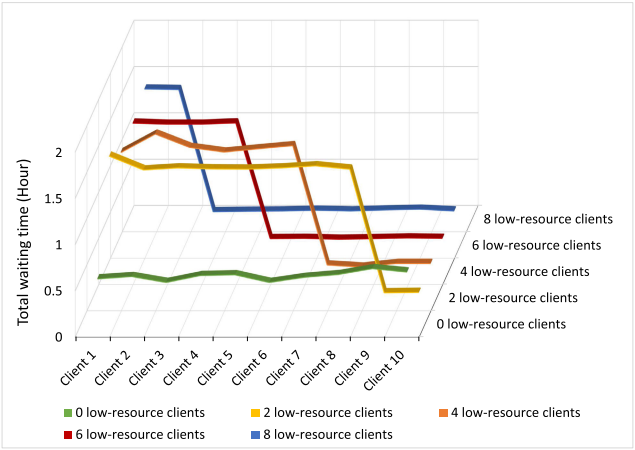
This subsection analyzes the effect of heterogeneous communication bandwidth on the performance of a synchronous DFL system, using the CIFAR-10 dataset with 10 clients and a target accuracy of 0.65. The evaluation focuses on the delays caused by low-bandwidth clients and their impact on key performance metrics, including convergence rounds, processing time, and overall system efficiency.

Figure 6 illustrates the concept of waiting time in a synchronous DFL system with heterogeneous communication bandwidth. The waiting time arises because clients with lower bandwidth face delays in exchanging parameters with other clients that have higher bandwidth. As a result, clients with higher bandwidth have to wait until they receive all the parameters from their slower neighbors.

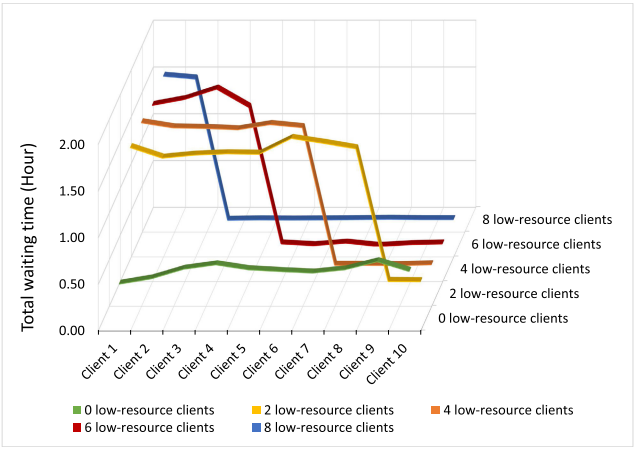
To evaluate this impact, a subset of clients was assigned a reduced bandwidth of 45 Mbps, compared to the 1000 Mbps connection speed of other clients. These clients are classified as low-bandwidth clients, and the evaluation is further divided into five test cases, each with an increasing number of



(a) Total waiting time in 2-connected topology



(b) Total waiting time in 5-connected topology



(c) Total waiting time in 8-connected topology

FIGURE 5. Total waiting time in k-connected in heterogeneous computing resources.

low-bandwidth clients: 0, 2, 4, 6, and 8 low-bandwidth clients, respectively.

Figure 7a displays the convergence rounds under varying topologies with different numbers of low-bandwidth clients. The results show that topologies with more connections tend to have more stable and lower convergence rounds, generally

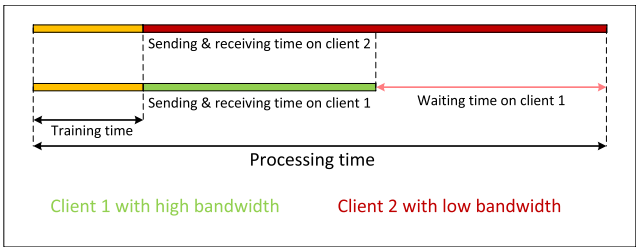


FIGURE 6. Synchronous DFL with heterogeneous communication bandwidth.

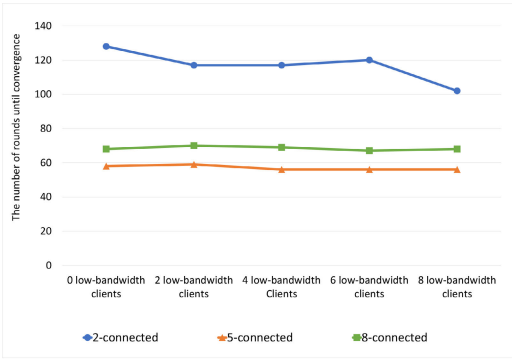
stabilizing between 60 and 70 rounds. In contrast, the 2-connected topology exhibits significant fluctuations, ranging from 100 to 140 rounds. This indicates that fewer connections result in less stable and slower convergence, while higher connectivity improves stability and reduces the number of rounds required for convergence.

Figure 7b shows the processing time per round for different topologies under heterogeneous communication bandwidth conditions. As expected, the processing time generally increases with the number of connections. The 2-connected topology consistently exhibits the lowest processing time per round with 171.24 seconds, while the 8-connected topology has the highest with 250 seconds, with 0 low-bandwidth clients. Interestingly, the presence of low-bandwidth clients increases the processing time in highly connected topologies. This is evident as the processing time for the 2-connected topology increases from approximately 239 seconds to nearly 600 seconds in the 8-connected topology when low-bandwidth clients are introduced. This observation can be attributed to the higher frequency and volume of data exchanges, which are affected more by low-bandwidth clients.

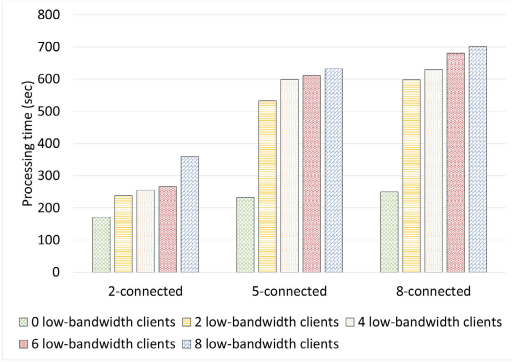
These findings suggest that the impact of low-bandwidth clients is significantly greater than that of low-resource clients in heterogeneous evaluations. This is because the congestion in the heterogeneous communication bandwidth evaluation arises from the cumulative impact of each low-bandwidth client on the network. The network becomes increasingly congested as the number of these clients increases because clients use the same communication link. This slower parameter exchange leads to longer waiting times and significantly longer processing time, resulting in more delays.

Figure 7c presents the convergence time, which is influenced by the convergence round and processing time per round. The convergence time for the 5-connected topology consistently outperforms the other topologies with a low convergence time. Hence, despite the impact of heterogeneity, carefully selecting the optimal number of connections is crucial for achieving efficient convergence in DFL. Low-bandwidth clients strongly affect the convergence time, particularly with higher connections.

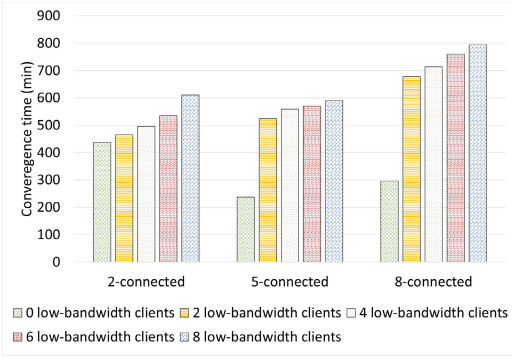
This result is also evident in Figure 7d, which shows the total waiting time across topologies as the number of



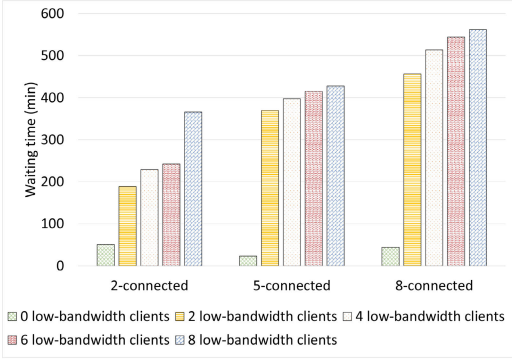
(a) Convergence round



(b) Processing time per round



(c) Convergence time



(d) Total waiting time

FIGURE 7. The convergence round, processing time per round, convergence time, and total waiting time in the heterogeneous communication bandwidth.

low-bandwidth clients increases. Initially, the total waiting time is minimal across all topologies. On the other hand, the waiting time increases dramatically with the introduction of low-bandwidth clients. For example, in the 5-connected topology, the waiting time increases from around 30 minutes to approximately 400 minutes. This increase caused by slower data transmission from low-bandwidth clients, can be interpreted as the communication time contributing to the system delay.

In summary, this result highlights the significant impact of the heterogeneous communication bandwidth on the convergence performance of different network topologies. Higher connectivity (5-connected and 8-connected topologies) generally improves stability and reduces the convergence round, but it makes the network more sensitive to increased processing and convergence times when low-bandwidth clients are present. The total waiting time also escalates dramatically with the introduction of low-bandwidth clients, worsening the overall convergence performance. Therefore, managing the effects of heterogeneous communication bandwidth is crucial for optimizing convergence in DFL environments.

D. SUMMARY OF LESSONS

This subsection summarizes the key insights from analyzing the factors influencing the DFL performance, including the network topologies, client scaling, heterogeneous computing resources, and communication bandwidth heterogeneity. It highlights how variations in connectivity and the number of clients affect the convergence speed, processing time, and communication costs, revealing the tradeoffs between connectivity and communication overhead. The analysis also underscores the bottleneck effects caused by heterogeneity in client computing resources and communication bandwidth, which significantly degrade the convergence time regardless of the connectivity level of the k-connected topologies. By examining the interplay of these factors, this summary provides practical strategies to optimize DFL performance in diverse and resource-constrained environments, focusing on enhancing scalability, communication efficiency, and convergence in real-world applications.

1) IMPACT OF NETWORK TOPOLOGIES

The performance of DFL is strongly affected by the connectivity among the clients in a homogeneous environment. Higher connectivity enhances collaboration and knowledge sharing, leading to faster and more stable convergence. On the other hand, increased connectivity also introduces higher communication overhead and longer processing times per round because of the greater frequency and volume of parameter exchanges. This analysis identifies an optimal point where the benefits of increased connectivity outweigh the associated costs, enabling faster convergence. Beyond this point, further increases in connections can hinder efficiency because of linear growth in the communication costs and processing time per round. Therefore, selecting an appropriate level of connectivity is crucial for achieving a balance between fast convergence and manageable communication costs.

2) IMPACT OF HETEROGENEOUS COMPUTING RESOURCES

The presence of disparities in client computing resources significantly affects the DFL performance, particularly in the processing and convergence time. The current evaluation focuses on high-resource and low-resource clients to show that high-resource clients must wait for slower clients, decelerating the entire process. This analysis shows that these metrics are determined primarily by the client with the lowest computing resources, producing a bottleneck effect that slows down the entire training process. This suggests that it is important to consider client computing resources during client selection. Interestingly, the number of rounds required for convergence remains relatively stable in densely connected topologies but exhibits greater instability in sparsely connected networks. This highlights the complex interplay between resource heterogeneity and network topology in DFL.

3) IMPACT OF HETEROGENEOUS COMMUNICATION BANDWIDTH

The presence of differences in the communication bandwidth among clients severely impacts DFL performance. Low-bandwidth clients act as bottlenecks during parameter exchange, leading to a substantial increase in processing time and convergence time, often several times higher than in homogeneous setups. The problem worsens as low-bandwidth clients grow, underscoring the need for bandwidth allocation strategies and communication-efficient protocols to mitigate delays. Addressing this will improve DFL scalability and performance in bandwidth-diverse networks.

V. DISCUSSION

This section discusses the potential limitations of our DFL framework, focusing on key challenges such as network topology selection, resource heterogeneity, communication bandwidth constraints, and non-independent and identically distributed (non-IID) data. Addressing these limitations can guide future work to enhance the practicality and robustness of the framework for real-world DFL deployments.

A. NETWORK TOPOLOGY SELECTION

Although this study suggests that topologies like the 5-connected network strike an effective balance between the communication cost and convergence speed, the selection of network topology remains a key limitation. In practice, the network topology can vary dynamically according to the changes in network infrastructure, client availability, and bandwidth conditions. Future DFL systems can use dynamic topology adjustment strategies, allowing the network to reconfigure itself in response to real-time conditions. This could involve clients forming new connections dynamically based on their current network conditions, resource availability, or data similarity. This would help maintain optimal performance across various network conditions without requiring manual intervention or static network design.

B. RESOURCE HETEROGENEITY

This evaluation has shown that clients with fewer computing resources can decelerate the overall convergence process, acting as bottlenecks in the training workflow. In real-world applications, clients often have varying computational power, memory, and network bandwidth. This variability can exacerbate the issue in setups where clients possess significantly different hardware capabilities. Future iterations of this framework could mitigate the impact of low-resource clients by incorporating asynchronous training methods [40], [41] that allow clients to update their models without waiting for slower clients. Additionally, resource-aware scheduling [42] could allocate tasks based on each client's capabilities, enabling those with higher computational power to handle a larger share of the training workload, improving overall system efficiency.

C. COMMUNICATION BANDWIDTH CONSTRAINTS

Another challenge is communication bandwidth heterogeneity. Clients with limited bandwidth can experience delays in model updates, leading to increased communication costs. These results show that low-bandwidth clients adversely affect the convergence time and overall system performance. The impact could be even more severe in large-scale and bandwidth-constrained networks, potentially resulting in unacceptable delays. One approach to address this issue is dynamic bandwidth allocation [43], which adjusts the bandwidth according to the network load and client needs. This would help prevent overwhelming low-bandwidth clients while enabling high-bandwidth clients to transmit data more efficiently. Another potential solution is communication-efficient protocols, such as compression techniques [44] or quantization [45], [46], which reduce the size of the model updates and minimize the communication overhead.

D. NON-IID DATA

Our DFL framework was evaluated with non-overlapping and IID data across clients. In real-world applications, however, data is often non-IID, meaning that the data available on each client may exhibit significant variation. This can lead to performance degradation because certain clients contribute less to the global model than others. Future studies will extend this evaluation to include non-IID data environments. This extension will allow for a deeper understanding of how non-IID data affects learning efficiency and model accuracy across different topologies, providing more insights into the robustness and applicability of DFL systems under various real-world conditions.

VI. CONCLUSION

This study designed a practical environment to implement and evaluate the performance of DFL. Increasing

connections does not always lead to faster convergence, showing that an optimal number of connections exists to balance the convergence speed and communication costs. In addition, the significant impact of resource and bandwidth heterogeneity on performance highlights the need for resource-aware scheduling, dynamic bandwidth allocation, and communication-efficient protocols in practical DFL deployments. These insights can be applied directly to the design of real-world DFL networks, guiding system architects in optimizing network topologies.

REFERENCES

[1] K. Q. Pham and T. Kim, "Elastic federated learning with Kubernetes vertical pod autoscaler for edge computing," *Future Gener. Comput. Syst.*, vol. 158, pp. 501–515, Sep. 2024, doi: 10.1016/j.future.2024.04.047.

[2] T. Singh, M. Kundroo, and T. Kim, "WSN-driven advances in soil moisture estimation: A machine learning approach," *Electronics*, vol. 13, no. 8, p. 1590, Apr. 2024. [Online]. Available: https://www.mdpi.com/2079-9292/13/8/1590

[3] H. B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. Y. Arcas, "Communication-efficient learning of deep networks from decentralized data," in *Proc. 20th Int. Conf. Artif. Intell. Statist. (AISTATS)*, 2017, pp. 1273–1282.

[4] K. Bonawitz, H. Eichner, W. Grieskamp, D. Huba, A. Ingerman, V. Ivanov, C. Kiddon, J. Konečný, S. Mazzocchi, H. Brendan McMahan, T. Van Overveldt, D. Petrou, D. Ramage, and J. Roselander, "Towards federated learning at scale: System design," 2019, *arXiv:1902.01046*.

[5] P. Kairouz et al., "Advances and open problems in federated learning," *Found. Trends Mach. Learn.*, vol. 14, nos. 1–2, pp. 1–210, 2021.

[6] L. Yuan, Z. Wang, L. Sun, P. S. Yu, and C. G. Brinton, "Decentralized federated learning: A survey and perspective," *IEEE Internet Things J.*, vol. 11, no. 21, pp. 34617–34638, Nov. 2024, doi: 10.1109/JIOT.2024.3407584.

[7] S. S. Lalitha, T. Javidi, and F. Koushanfar, "Fully decentralized federated learning," in *Proc. 3rd workshop Bayesian Deep Learn. (NeurIPS)*, 2018, pp. 1–34.

[8] K. Ouyang, J. Yu, X. Cao, and Z. Liao, "Towards reliable federated learning using blockchain-based reverse auctions and reputation incentives," *Symmetry*, vol. 15, no. 12, p. 2179, Dec. 2023. [Online]. Available: https://www.mdpi.com/2073-8994/15/12/2179

[9] J. Daily, A. Vishnu, C. Siegel, T. Warfel, and V. Amatya, "GossipGraD: Scalable deep learning using gossip communication based asynchronous gradient descent," 2018, *arXiv:1803.05880*.

[10] V. Mothukuri, R. M. Parizi, S. Pouriyeh, Y. Huang, A. Dehghantanha, and G. Srivastava, "A survey on security and privacy of federated learning," *Future Gener. Comput. Syst.*, vol. 115, pp. 619–640, Oct. 2020. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0167739X20329848

[11] T. Vogels, H. Hendrikx, and M. Jaggi, "Beyond spectral gap: The role of the topology in decentralized learning," *J. Mach. Learn. Res.*, vol. 24, pp. 1–31, Jan. 2023.

[12] Z. Lian, Q. Yang, W. Wang, Q. Zeng, M. Alazab, H. Zhao, and C. Su, "DEEP-FEL: Decentralized, efficient and privacy-enhanced federated edge learning for healthcare cyber physical systems," *IEEE Trans. Netw. Sci. Eng.*, vol. 9, no. 5, pp. 3558–3569, Sep. 2022, doi: 10.1109/TNSE.2022.3175945.

[13] Z. Wang, Y. Hu, S. Yan, Z. Wang, R. Hou, and C. Wu, "Efficient ring-topology decentralized federated learning with deep generative models for medical data in eHealthcare systems," *Electronics*, vol. 11, no. 10, p. 1548, May 2022, doi: 10.3390/electronics11101548.

[14] J. Kang, D. Ye, J. Nie, J. Xiao, X. Deng, S. Wang, Z. Xiong, R. Yu, and D. Niyato, "Blockchain-based federated learning for industrial metaverses: Incentive scheme with optimal AoI," in *Proc. IEEE Int. Conf. Blockchain (Blockchain)*, Aug. 2022, pp. 71–78. [Online]. Available: https://api.semanticscholar.org/CorpusID

[15] L. Wang, Y. Xu, H. Xu, M. Chen, and L. Huang, "Accelerating decentralized federated learning in heterogeneous edge computing," *IEEE Trans. Mobile Comput.*, vol. 22, no. 9, pp. 5001–5016, Sep. 2023, doi: 10.1109/TMC.2022.3178378.

[16] Y. Wang, Z. Su, N. Zhang, and A. Benslimane, "Learning in the air: Secure federated learning for UAV-assisted crowdsensing," *IEEE Trans. Netw. Sci. Eng.*, vol. 8, no. 2, pp. 1055–1069, Apr. 2021, doi: 10.1109/TNSE.2020.3014385.

[17] J.-H. Chen, M.-R. Chen, G.-Q. Zeng, and J.-S. Weng, "BDFL: A Byzantine-fault-tolerance decentralized federated learning method for autonomous vehicle," *IEEE Trans. Veh. Technol.*, vol. 70, no. 9, pp. 8639–8652, Sep. 2021, doi: 10.1109/TVT.2021.3102121.

[18] E. T. Martínez Beltrán, M. Q. Pérez, P. M. S. Sánchez, S. L. Bernal, G. Bovet, M. G. Pérez, G. M. Pérez, and A. H. Celdrán, "Decentralized federated learning: Fundamentals, state of the art, frameworks, trends, and challenges," *IEEE Commun. Surveys Tuts.*, vol. 25, no. 4, pp. 2983–3013, 4th Quart., 2023, doi: 10.1109/COMST.2023.3315746.

[19] Y. Liao, Y. Xu, H. Xu, L. Wang, and C. Qian, "Adaptive configuration for heterogeneous participants in decentralized federated learning," in *Proc. IEEE Conf. Comput. Commun.*, May 2023, pp. 1–10. [Online]. Available: https://api.semanticscholar.org/CorpusID

[20] L. Palmieri, C. Boldrini, L. Valerio, A. Passarella, and M. Conti, "Impact of network topology on the performance of decentralized federated learning," 2024, *arXiv:2402.18606*.

[21] S. Reddi, Z. Charles, M. Zaheer, Z. Garrett, K. Rush, J. Konecný, S. Kumar, and H. Brendan McMahan, "Adaptive federated optimization," 2020, *arXiv:2003.00295*.

[22] Y. Takezawa, R. Sato, H. Bao, K. Niwa, and M. Yamada, "Beyond exponential graph: Communication-efficient topologies for decentralized learning via finite-time convergence," 2023, *arXiv:2305.11420*.

[23] P. K. Quan, M. Kundroo, and T. Kim, "Experimental evaluation and analysis of federated learning in edge computing environments," *IEEE Access*, vol. 11, pp. 33628–33639, 2023, doi: 10.1109/ACCESS.2023.3262945.

[24] W. Y. B. Lim, N. C. Luong, D. T. Hoang, Y. Jiao, Y.-C. Liang, Q. Yang, D. Niyato, and C. Miao, "Federated learning in mobile edge networks: A comprehensive survey," *IEEE Commun. Surveys Tuts.*, vol. 22, no. 3, pp. 2031–2063, 3rd Quart., 2020, doi: 10.1109/COMST.2020.2986024.

[25] L. H. Phuc, L.-A. Phan, and T. Kim, "Traffic-aware horizontal pod autoscaler in Kubernetes-based edge computing infrastructure," *IEEE Access*, vol. 10, pp. 18966–18977, 2022, doi: 10.1109/ACCESS.2022.3150867.

[26] H. T. Nguyen, V. Sehwag, S. Hosseinalipour, C. G. Brinton, M. Chiang, and H. V. Poor, "Fast-convergent federated learning," *IEEE J. Sel. Areas Commun.*, vol. 39, no. 1, pp. 201–218, Jan. 2021. [Online]. Available: https://api.semanticscholar.org/CorpusID

[27] E. Hallaji, R. Razavi-Far, M. Saif, B. Wang, and Q. Yang, "Decentralized federated learning: A survey on security and privacy," *IEEE Trans. Big Data*, vol. 10, no. 2, pp. 194–213, Apr. 2024. [Online]. Available: https://api.semanticscholar.org/CorpusID

[28] H. Ye, L. Liang, and G. Y. Li, "Decentralized federated learning with unreliable communications," *IEEE J. Sel. Topics Signal Process.*, vol. 16, no. 3, pp. 487–500, Apr. 2022, doi: 10.1109/JSTSP.2022.3152445.

[29] T. Sun, D. Li, and B. Wang, "Decentralized federated averaging," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 45, no. 4, pp. 4289–4301, Apr. 2023.

[30] A. Nedic, A. Olshevsky, and M. G. Rabbat, "Network topology and communication-computation tradeoffs in decentralized optimization," *Proc. IEEE*, vol. 106, no. 5, pp. 953–976, May 2018, doi: 10.1109/JPROC.2018.2817461.

[31] A. Badie-Modiri, C. Boldrini, L. Valerio, J. Kertész, and M. Karsai, "Initialisation and network effects in decentralised federated learning," 2024, *arXiv:2403.15855*.

[32] Q. Chen, Z. Wang, Y. Zhou, J. Chen, D. Xiao, and X. Lin, "CFL: Cluster federated learning in large-scale peer-to-peer networks," 2022, *arXiv:2204.03843*.

[33] S. Bhattacharya, D. Helo, and J. Siegel, "Impact of network topology on Byzantine resilience in decentralized federated learning," 2024, *arXiv:2407.05141*.

[34] P. Liu, J. Jiang, G. Zhu, L. Cheng, W. Jiang, W. Luo, Y. Du, and Z. Wang, "Training time minimization for federated edge learning with optimized gradient quantization and bandwidth allocation," *Frontiers Inf. Technol. Electron. Eng.*, vol. 23, no. 9, pp. 1247–1263, 2022. [Online]. Available: https://link.springer.com/article/10.1631/FITEE.2100538

[35] Z. Yan and D. Li, "Convergence time optimization for decentralized federated learning with LEO satellites via number control," *IEEE Trans. Veh. Technol.*, vol. 73, no. 3, pp. 4517–4522, Mar. 2024, doi: 10.1109/TVT.2023.3322461.

[36] W. Liu, L. Chen, and W. Zhang, "Decentralized federated learning: Balancing communication and computing costs," *IEEE Trans. Signal Inf. Process. Over Netw.*, vol. 8, pp. 131–143, 2022. [Online]. Available: https://api.semanticscholar.org/CorpusID
[37] H. Xiao, K. Rasul, and R. Vollgraf, "Fashion-MNIST: A novel image dataset for benchmarking machine learning algorithms," 2017, *arXiv:1708.07747*.
[38] A. Krizhevsky and G. Hinton, "Learning multiple layers of features from tiny images," Dept. Comput. Sci., Univ. Toronto, Toronto, ON, Canada, Tech. Rep. TR-2009, 2009. [Online]. Available: https://www.cs.toronto.edu/~kriz/learning-features-2009-TR.pdf
[39] K. O'Shea and R. Nash, "An introduction to convolutional neural networks," 2015, *arXiv:1511.08458*.
[40] D. Shenaj, M. Toldo, A. Rigon, and P. Zanuttigh, "Asynchronous federated continual learning," in *Proc. IEEE CVPRW*, Jun. 2023, pp. 5055–5063. [Online]. Available: https://api.semanticscholar.org/CorpusID
[41] T. Zhang, L. Gao, S. Lee, M. Zhang, and S. Avestimehr, "TimelyFL: Heterogeneity-aware asynchronous federated learning with adaptive partial training," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. Workshops (CVPRW)*, Jun. 2023, pp. 5064–5073. [Online]. Available: https://doi.ieeecomputersociety.org/10.1109/CVPRW59228.2023.00535
[42] M. Skocaj, P. E. I. Rivera, R. Verdone, and M. Erol-Kantarci, "Uplink scheduling in federated learning: An importance-aware approach via graph representation learning," in *Proc. IEEE Int. Conf. Commun. Workshops (ICC Workshops)*, May 2023, pp. 1014–1019, doi: 10.1109/iccworkshops57953.2023.10283576.
[43] L. Yu, X. Sun, R. Albelaihi, C. Park, and S. Shao, "Dynamic client clustering, bandwidth allocation, and workload optimization for semi-synchronous federated learning," *Electronics*, vol. 13, no. 23, p. 4585, Nov. 2024, doi: 10.3390/electronics13234585.
[44] L. G. Ribeiro, M. Leonardon, G. Müller, V. Fresse, and M. Arzel, "Federated learning compression designed for lightweight communications," 2023, *arXiv:2310.14693*.
[45] T. Tian, H. Shi, R. Ma, and Y. Liu, "FedACQ: Adaptive clustering quantization of model parameters in federated learning," *Int. J. Web Inf. Syst.*, vol. 20, no. 1, pp. 88–110, Feb. 2024, doi: 10.1108/ijwis-08-2023-0128.
[46] Z. Zhao, Y. Mao, Z. Shi, Y. Liu, T. Lan, W. Ding, and X.-P. Zhang, "AQUILA: Communication efficient federated learning with adaptive quantization in device selection strategy," *IEEE Trans. Mobile Comput.*, vol. 23, no. 6, pp. 7363–7376, 2024. [Online]. Available: https://doi.ieeecomputersociety.org/10.1109/TMC.2023.3332901

VO VAN TRUONG (Graduate Student Member, IEEE) received the B.S. degree in control and automation engineering from Ton Duc Thang University, Ho Chi Minh City, Vietnam, in 2023. He is currently pursuing the Ph.D. degree with the School of Information and Communication Engineering, Chungbuk National University, South Korea. He worked as a Student Research Assistant with the Automation and IoT Applications Research Group, Ton Duc Thang University, under the supervision of M.S. Thieu Quang Tri. His primary research interests include edge computing, edge AI, federated learning, decentralized federated learning, the IoT applications, and container orchestration.

PHAM KHANH QUAN (Graduate Student Member, IEEE) received the B.S. degree in electronics and telecommunications engineering from Ton Duc Thang University, Ho Chi Minh City, Vietnam, in 2020, and the M.S. degree from the School of Information and Communication Engineering, Chungbuk National University (CBNU), South Korea, in 2023. He is currently pursuing the Ph.D. degree with the School of Information and Communication Engineering, Chungbuk National University. He was working as a Student Research Assistant of Wireless Communications Research Group, Ton Duc Thang University, under the supervision of Dr. Phuong T. Tran. His major interests include edge computing, edge AI, federated learning, reinforcement learning, the IoT applications, and container orchestration.

DONG-HWAN PARK received the B.S. and M.S. degrees in electronics engineering from Kyungpook National University, South Korea, in 1999 and 2001, respectively. He joined the Electronics and Telecommunications Research Institute (ETRI), Daejeon, South Korea, in 2001. His research work focused on intelligent IoT frameworks and the IoT platforms. He developed sematic-based IoT platforms and smart home network middleware, such as Jini, HAVi, and UPnP. His research interests include cloud-based IoT platforms and intelligent IoT device platforms.

TAEHONG KIM (Senior Member, IEEE) received the B.S. degree in computer science from Ajou University, Republic of Korea, in 2005, and the M.S. degree in information and communication engineering and the Ph.D. degree in computer science from Korea Advanced Institute of Science and Technology (KAIST), in 2007 and 2012, respectively. He was a Research Staff Member with the Samsung Advanced Institute of Technology (SAIT) and the Samsung DMC Research and Development Center, from 2012 to 2014. He was a Senior Researcher with the Electronics and Telecommunications Research Institute (ETRI), Republic of Korea, from 2014 to 2016. Since 2016, he has been an Associate Professor with the School of Information and Communication Engineering, Chungbuk National University, Republic of Korea. His research interests include edge computing, container orchestration, the Internet of Things, and federated learning. Since 2020, he has been an Associate Editor of IEEE ACCESS.