IIETA International Information and Engineering Technology Association
Advancing the World of Information and Engineering

Journal Européen des Systèmes Automatisés
Vol. 59, No. 5, May, 2026, pp. 1373-1384
Journal homepage: http://iieta.org/journals/jesa

A Comparative Study of Centralized and Decentralized Split Federated Learning Approaches in 6G Beamforming Optimization

Samar H. Alkayat[1*], Ali H. Hamad[2]

[1] Informatics Institute for Postgraduate Studies (IIPS), University of Information Technology and Communications (UoITC), Baghdad 10066, Iraq
[2] College of Artificial Intelligence, University of Baghdad, Baghdad 10071, Iraq

Corresponding Author Email: ms202420027@uoitc.edu.iq

https://doi.org/10.18280/jesa.590516

Received: 1 March 2026
Revised: 11 May 2026
Accepted: 20 May 2026
Available online: 31 May 2026

Keywords:
decentralized split federated learning, federated learning, split learning, 6G Internet of Things, peer-to-peer ring topology

ABSTRACT

There are serious issues associated with the use of machine learning in 6G-enabled Internet of Things (IoT) systems in terms of data privacy, communication overhead, and system robustness. Current federated and split learning (SL) models are largely based on centralized aggregation servers, which cause single-point-of-failure vulnerabilities, scaling issues, and communication bottlenecks in large-scale IoT systems. To overcome these issues, this paper suggests a completely decentralized split federated learning (DSFL) model that combines SL with peer-to-peer federated coordination. The suggested design divides a deep neural network into a client-side and a server-side sub-model, allowing its computation to be offloaded and maintain the data locality. The fully decentralized way of model coordination and aggregation, with a ring-based topology, does not require a central aggregation server. The framework is tested based on a realistic 6G IoT beamforming optimization dataset in controlled independent and identically distributed (IID) data conditions. The experimental results indicate that it converges steadily and generalizes well with a test accuracy of 96% and a validation accuracy of 90.67. Class-wise analysis indicates strong performance in the presence of class imbalance, with an F1-score of 97.6% and 88.5% on the majority and minority classes, respectively and a very high recall of 92% on optimized beamforming cases. These findings validate the practicality and strength of the suggested framework of DSFL in decentralized learning in 6G IoT settings.

1. INTRODUCTION

The rapid growth of the Internet of Things (IoT) and the increasing deployment of edge computing infrastructures have led to the generation of massive volumes of distributed, heterogeneous, and privacy-sensitive data. In such environments, conventional centralized machine learning approaches become increasingly impractical due to privacy concerns, high communication overhead, and limited scalability. Transferring raw data from resource-constrained edge devices to centralized servers not only violates data locality requirements but also introduces substantial latency and bandwidth consumption, particularly in large-scale IoT deployments [1].

To address these challenges, federated learning (FL) has emerged as a distributed learning paradigm [2] that enables multiple clients to collaboratively train a shared model without exchanging raw data [3]. Instead, clients locally train models and share only model parameters or gradients with the coordinating server. This paradigm significantly improves privacy preservation and reduces data transmission requirements. However, most existing FL systems rely on centralized federated learning (CFL) architectures in which a central server coordinates the training process and aggregates client updates. Although effective, centralized FL introduces several limitations, including single-point-of-failure risks [4] communication bottlenecks, and reduced robustness in large-scale deployments [5].

To overcome these limitations, decentralized federated learning (DFL) has been proposed as an alternative paradigm that removes the dependency on a central aggregation server. In DFL, participating clients exchange and aggregate model updates directly through peer-to-peer communication over predefined network topologies. By distributing the aggregation process across participating nodes, DFL enhances system robustness and scalability while reducing centralized coordination overhead [5, 6]. Nevertheless, previous studies indicate that decentralized optimization may experience slower convergence or reduced model consistency compared to centralized approaches due to delayed information propagation among nodes [7, 8].

In parallel, split learning (SL) has been proposed to address computational and memory constraints in resource-limited edge devices [9]. In SL, a deep neural network is partitioned at a predefined cut layer into a client-side and a server-side sub-model. Clients perform the initial layers locally while the remaining layers are executed on the server using intermediate activations instead of raw data. This design reduces on-device

computational cost and preserves data locality, making SL suitable for edge environments. However, traditional SL architectures still rely on centralized server-side coordination, which limits scalability and robustness in large-scale distributed systems [10].

To combine the advantages of both paradigms, split federated learning (SFL) integrates SL with federated aggregation to enable collaborative training while reducing computation at edge devices. Although SFL improves privacy preservation and computational efficiency, most existing SFL frameworks still rely on centralized aggregation servers, thereby reintroducing communication bottlenecks and single-point-of-failure risks [11].

Recently, decentralized split federated learning (DSFL) has emerged as a promising research direction that integrates model splitting with decentralized coordination. In DSFL, the neural network is partitioned between client-side and server-side components, while model aggregation is performed through decentralized peer-to-peer communication among participating nodes. This architecture eliminates the dependency on centralized aggregation servers and enhances scalability, robustness, and fault tolerance in large-scale IoT environments [12].

These properties make DSFL particularly attractive for future 6G-enabled IoT networks, where learning systems must operate across highly distributed, heterogeneous, and resource-constrained devices under strict communication constraints. Despite recent progress, existing DSFL and decentralized learning studies are typically evaluated on standard image classification benchmarks and rarely consider practical wireless optimization tasks or realistic IoT deployment scenarios. Furthermore, limited attention has been given to evaluating decentralized SL architectures under realistic edge-computing constraints.

Therefore, this paper proposes and evaluates a fully DSFL framework for 6G-enabled IoT beamforming optimization tasks. The proposed framework integrates SL with decentralized peer-to-peer coordination through a ring topology to enable scalable and privacy-preserving collaborative learning without centralized aggregation.

The main contributions of this work are summarized as follows:

• DSFL architecture: A DSFL design that does not require centralized aggregation and keeps the computational efficiency of SL within the resource-limited IoT context.

• Decentralized coordination with rings: A P2P ring topology that supports decentralized model aggregation and minimizes communication overheads in contrast to centralized FL designs.

• Application-based testing: Experimental verification of a realistic 6G IoT beamforming optimization dataset with stable convergence and high classification.

The rest of this paper will be structured as follows: Section 2 will be a review of related work and the gaps present in existing research. Section 3 gives the proposed DSFL framework. Section 4 outlines the suggested system design and data preparation. The results and performance evaluation of the experiments are discussed in Section 5. Lastly, Section 6 provides a conclusion to the paper and the research directions in the future.

2. RELATED WORK

SFL was initially proposed to combine the advantages of SL and federated learning within distributed training environments. Early SFL frameworks primarily relied on centralized aggregation servers to coordinate model updates among participating clients. Although these architectures improved computational efficiency and preserved data privacy by avoiding direct raw-data sharing, they still suffered from scalability limitations, communication bottlenecks, and single-point-of-failure risks in large-scale distributed systems. To overcome these limitations, recent research has increasingly explored decentralized extensions of federated and SL frameworks.

Recent studies have investigated DFL algorithms to improve convergence stability and model consistency in distributed environments. For example, Shi et al. [13] proposed DFedSAM, which integrates sharpness-aware minimization with decentralized model sharing to improve generalization performance in decentralized learning systems. Despite achieving improved convergence behavior, the proposed framework focused mainly on full-model decentralized training and did not consider SL architectures or resource-constrained IoT environments.

Decentralized stochastic optimization methods have also been explored to improve communication efficiency in distributed deep learning systems. Assran et al. [14] introduced Stochastic Gradient Push (SGP), a decentralized optimization framework for distributed neural-network training without centralized parameter servers. The proposed method demonstrated effective decentralized optimization and communication-aware parameter propagation. However, the study was mainly evaluated on benchmark image-classification datasets and did not explicitly address federated SL or practical IoT optimization scenarios.

In healthcare and privacy-sensitive applications, DFL based on ring topologies has been proposed to improve communication efficiency and privacy preservation. Wang et al. [15] developed a ring-topology DFL framework integrated with deep generative models for medical-data applications. Although the proposed ring-based coordination demonstrated improved communication efficiency in decentralized environments, the study focused primarily on healthcare image-classification tasks and did not investigate SL architectures or application-oriented 6G-enabled IoT optimization scenarios.

Hybrid approaches combining federated SL with blockchain technologies have also been proposed to decentralize coordination and improve trustworthiness in distributed systems. Beis-Penedo et al. [16] introduced HLF-FSL, which integrates SFL with the Hyperledger Fabric blockchain platform to reduce centralized trust assumptions. Nevertheless, the framework still relied on centralized execution of server-side computations, which may introduce scalability limitations and communication overhead in large-scale IoT deployments.

Momentum-based decentralized optimization techniques have also been explored to improve convergence stability and communication efficiency. Sun et al. [17] proposed Decentralized Federated Averaging (DFedAvg), which enhances decentralized parameter aggregation through momentum-based optimization. Although the framework improved convergence behavior in decentralized environments, the study primarily focused on full-model federated learning and did not consider split-learning architectures or decentralized split coordination mechanisms.

Privacy-preserving decentralized learning has gained

increasing attention in biomedical and sensitive-data environments. Tajabadi et al. [18] provided a comprehensive overview of privacy-preserving decentralized learning methods for biomedical applications. However, the study mainly focused on privacy-preserving strategies and biomedical data protection rather than decentralized split-learning architectures for IoT-oriented optimization environments.

In addition, Chen et al. [19] proposed communication-efficient decentralized learning approaches to reduce bandwidth requirements and communication overhead in distributed learning systems. However, most existing studies focus mainly on communication-efficient federated learning or benchmark-oriented distributed optimization, while limited attention has been given to fully DSFL architectures evaluated under application-oriented IoT optimization scenarios.

Recent studies have also highlighted the importance of topology-aware decentralized learning strategies for improving communication efficiency and scalability in distributed edge environments. In particular, ring-based decentralized coordination mechanisms have demonstrated the potential to reduce synchronization overhead and communication congestion by restricting parameter exchange to neighboring nodes only. Nevertheless, existing studies remain largely limited to benchmark datasets, healthcare applications, or theoretical optimization analysis rather than practical distributed optimization problems in resource-constrained 6G-enabled IoT environments.

In general, the available literature mainly focuses on centralized SFL, full-model DFL, benchmark-based evaluations, or theoretical optimization frameworks. Fully DSFL remains a relatively recent and insufficiently explored research direction, particularly in application-oriented IoT optimization environments. Moreover, most existing studies do not experimentally investigate decentralized split learning under communication-efficient peer-to-peer coordination mechanisms within realistic 6G-enabled IoT optimization scenarios. Therefore, this paper addresses an important research gap by introducing and experimentally evaluating a fully DSFL architecture for 6G-enabled IoT beamforming optimization using ring-based peer-to-peer coordination. Table 1 provides a comparative summary of representative centralized, decentralized, and SFL studies, including their application domains, datasets, coordination strategies, and learning paradigms.

Table 1. Comparative summary of representative centralized, decentralized, and split federated learning studies

Ref.	Application	Dataset	Coordination	FL	DFL	SL	SFL	DSFL	Reported Accuracy	Main Limitation
[13]	N/A	Benchmark datasets	Decentralized	✓	✓	✗	✗	✗	~88–91%	No split learning support
[14]	N/A	MNIST, CIFAR-10	Decentralized	✗	✓	✗	✗	✗	~90%	Benchmark-oriented evaluation
[15]	HealthCare Systems	Medical datasets	Ring-based	✓	✓	✗	✗	✗	~92–95%	Limited to healthcare applications
[16]	Blockchain-based SFL	Benchmark datasets	Blockchain-assisted	✓	Partial	✓	✓	✗	~91–94%	Centralized server-side execution
[17]	N/A	Benchmark datasets	Decentralized	✓	✓	✗	✗	✗	~89–92%	No split coordination
[18]	Biomedical Learning	Biomedical datasets	Decentralized	✓	✓	✗	✗	✗	N/A	Focused on privacy preservation
[19]	Communication-efficient	Benchmark / Theoretical	Decentralized	✓	✓	✗	✗	✗	N/A	Limited IoT-oriented evaluation
Proposed Work	6G IoT Beamforming Optimization	Beamforming Dataset	Ring-based	✗	✗	✓	✗	✓	96–97%	Fully decentralized split learning

Note: federated learning (FL); decentralized federated learning (DFL); split learning (SL); split federated learning (SFL); decentralized split federated learning (DSFL)

2.1 Research gap

The current body of work on DFL is mainly based on full-model training and assessed on benchmark datasets, thereby restricting its use in resource-constrained IoT systems. However, the majority of SFL methods continue to use centralized aggregation servers, which again introduce the risks of single-point-of-failure and bottlenecks in communication. Even though few works explore the concept of decentralized or hybrid split learning schemes, much of it does not consider the concept of convergence stability, topology-conscious coordination, or communication efficiency in purely decentralized environments. Moreover, the realistic 6G-enabled IoT optimization case scenarios are almost never taken into account by current evaluations. Unlike these previous works, this paper explicitly discusses these shortcomings by introducing a fully DSFL architecture that unites split model training with peer-to-peer coordination across a ring topology and is tested on a 6G IoT beamforming optimization problem. Consequently, the current research addresses a vital research gap since it bridges the gap between decentralized learning, split model architecture, and practical 6G IoT requirements.

3. METHODOLOGY

This section will introduce the suggested DSFL model. The suggested system combines split learning and completely decentralized coordination that allows collaborative model learning between resource-constrained IoT devices without the aid of a centralized aggregation server. The DSFL model has the IoT clients jointly training a common deep neural network in a decentralized peer-to-peer setting. The clients are connected in a fixed topology of a ring, and the model updates can be directly exchanged without any centralized control. With the split learning paradigm, the neural network is partitioned vertically to form two sub-models, a client-side model (head) and a server-side model (tail). Raw data are processed locally by each client on the head model and are only exchanged with intermediate activations during training. Such design ensures data privacy since it does not transmit raw

data. The ring topology is used to perform model synchronization and aggregation in a decentralized way among the clients involved. Exchange of model parameters between each client is limited to its neighbors, resulting in better scalability, robustness, and removing the single-point-of-failure problems inherent in centralized architectures. In general, the offered DSFL system is an efficient, scalable, and non-invasive system of collaborative training in 6G-enabled IoT contexts that combines the benefits of split learning and DFL. Mathematically, the overall model can be represented as:

Mathematically, the overall model can be represented as:

$$f(x) = f_s(f_c(x)) \tag{1}$$

where, $f_c(x)$ represents the client-side model and $f_s(.)$ represents the server-side model. During forward propagation, each client computes:

$$\alpha = f_c(x) \tag{2}$$

where, α is the intermediate activation (smashed data), which is then forwarded within the decentralized network to complete the prediction:

$$\hat{y} = f_s(\alpha) \tag{3}$$

After computing the loss function $(y, \hat{y})$ the gradients are propagated backward, and model parameters are updated as follows:

$$\omega_c \leftarrow \omega_c - \eta \nabla_{w_c} L(y, \hat{y}) \tag{4}$$

$$\omega_s \leftarrow \omega_s - \eta \nabla_{w_s} L(y, \hat{y}) \tag{5}$$

In contrast to centralized federated learning, DSFL performs model aggregation in a decentralized manner using a ring-based topology. In this structure, each client communicates only with its neighboring nodes and updates its parameters based on locally received models. The decentralized aggregation at each client *i* can be expressed as:

$$w_i^{(t+1)} = \sum_{j \in N(i)} \alpha_{ij} \; w_i^{(t)} \tag{6}$$

where:

- $w_i^{(t+1)}$ is the updated model of client *i* at round *t*+1.
- $N(i)$ represents the set of neighboring clients (e.g., previous and next nodes in the ring).
- α_{ij} are the weighting coefficients $\omega_i^{(t+1)} = \sum_{j \in N(i)} \alpha_{ij} = 1$.
- $\omega_i^{(t)}$ are the model parameters received from neighboring clients.

In a ring topology (where the network contains a single ring), clients normally combine their model with their immediate neighbors, which guarantees the slow spread of the model changes throughout the network. This mechanism of decentralized aggregation eliminates the use of a central server, communication bottlenecks and enhances the robustness of the system. Meanwhile, it also maintains privacy, swapping only intermediate activations and model parameters and thus rendering DSFL very appropriate to scalable and reliable 6G IoT devices.

3.1 Decentralized topology and coordination

The distributed peer-to-peer coordination scheme is one of the main differences when compared to conventional SFL architectures, which are based on a centralized aggregation server. The proposed design consists of the participating clients connected in a ring topology, each node only connecting to its adjacent nodes in the ring. This decentralized communication approach allows direct communication for the exchange of model parameters among the clients, without relying on a centralized coordination. There are different network topologies that can be used to implement decentralized learning such as ring, mesh, star, and fully connected graphs [20] with each topology having its own advantages and disadvantages in terms of communication overhead, convergence behavior, scalability and robustness.

Table 2 summarizes the advantages and limitations of common decentralized communication topologies used in distributed learning systems.

Table 2. Simplified comparison of common decentralized communication topologies used in distributed learning systems

Topology	Advantages	Limitations
Ring	Low communication overhead	Slower convergence
Fully Connected	Fast convergence	High communication cost
Star	Simple coordination	Central bottleneck
Mesh	High robustness	Complex communication

In this work, the ring topology was chosen for its simplicity, minimal communication overhead, less synchronization complexity, and applicability to the resource-constrained 6G-enabled IoT environments. In such topology, the communication is only with its immediate neighbors, thus significantly reducing the number of communication links and bandwidth requirement compared to fully connected decentralized topologies. This is especially crucial in a large-scale IoT system, where communication efficiency, energy usage, and scalability serve as vital constraints. Unlike star architectures, where the communication is centralized in a specific entity, the ring reduces single points of failure and centralized bottlenecks and allows full decentralized peer-to-peer collaboration of the clients involved. The fully connected topologies have the advantage of transmitting information and reaching consensus more quickly, but the number of involved clients would lead to higher communication and synchronization overhead. Thus, the ring topology offers a practical compromise between communication efficiency, robustness against decentralization, and ease of implementation in scalable IoT environments. The DFL architecture can be implemented in various ways, such as a fully connected network, a partially connected network (e.g., star, ring, and random graphs), or a clustered node layout as shown in Figure 1. The ring-based design in the proposed DSFL framework ensures a progressive and efficient dissemination of model updates across the network, potentially enhancing the robustness of the system without the centralized bottlenecks typical of conventional learning architectures. In training, local updates of model parameters are exchanged locally among neighboring peers, thereby being decentralized. This coordination plan eliminates

communication congestion, increases fault-tolerance, and increases robustness of the system without global synchronization. Thus, the proposed framework is very appropriate for 6G empowered IoT deployment for scalability, reliability, and communication efficiency. While the current study explores a ring topology as a proof-of-concept decentralized coordination mechanism, other decentralized topologies, such as star, mesh, and fully connected topologies still need to be compared in experimental evaluations to understand further the scalability, convergence behavior, robustness and communication efficiency under various 6G IoT deployments.

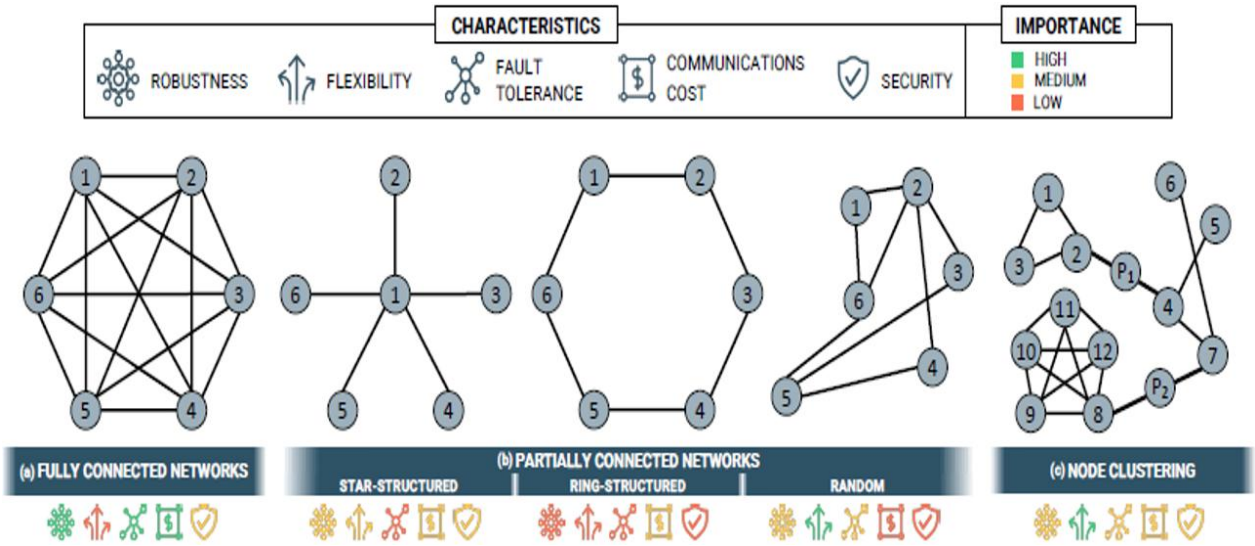

Figure 1. Network topologies in decentralized federated learning (DFL), including fully connected, partially connected (star, ring, and random), and node clustering structures [5, 6]

4. PROPOSED SYSTEM DESIGN

This section introduces the architecture of the proposed split federated learning (DSFL) framework that is decentralized. It is a statement of the dataset attributes and preprocessing, the elected model structure and division policy, and a description of the entire process of the system to run distributed learning in 6G IoT networks, which offers a practical implementation of the suggested methodology.

4.1 Dataset analysis

To assess the proposed distributed learning frameworks, the IoT Beamforming Optimization Dataset for 6G-enabled IoT communication environments was used. The data is available on Kaggle: https://www.kaggle.com/datasets/ziya07/6g-iot-intelligent-management-dataset. Table 3 shows a summary of the main characteristics of the dataset. The dataset comprises 1000 samples and 21 input features, each corresponding to one of the different features related to communication, networking and beamforming in intelligent 6G IoT optimization scenarios. These features include signal quality, interference conditions, beamforming gain, resource utilization, latency, throughput, mobility conditions, and antenna-related parameters, which can be used to assess the distributed learning frameworks in realistic 6G IoT environments. Two output classes (optimized and non-optimized beamforming) exist in this dataset. The original data set has a very high-class imbalance (83.2%:16.8% non-optimized: optimized). In real-world communication optimization situations, the fully optimized beamforming conditions are less common, which is why such imbalance is often observed. To tackle this problem, several preprocessing methods have been implemented prior to training, such as the removal of duplicated and missing samples, Z-score normalization of numerical features and encoding of categorical features. To prevent the bias and leakage of evaluation, the data set was split into training set, validation set, and testing set, maintaining the same class distribution as the original set. The data set was then divided into training set, validation set and testing set without altering the class distribution. Random oversampling was subsequently only performed in the training data to balance the minority class, without changing the validation and testing data sets, to provide an unbiased and fair evaluation of the performance. After the balancing process, the training set was made equal for each class, 50% for each class as shown in Figure 2. Relatively high accuracy values were found before class balancing but the high-class imbalance affected the results and they were not necessarily indicative of the models' ability to detect the minority optimized class. Lower recall was noted particularly in some of the models, which showed biased learning behavior towards the majority class. To address the issue of under-representation of minority classes and to ensure a more balanced and reliable assessment, random oversampling was used in the final experimental design. Random oversampling might lead to an overfitting problem by introducing minority samples multiple times, but several regularization techniques were taken into account such as Dropout regularization with dropout rate 0.4, L2 weight decay ($\lambda = 0.005$) and early stopping of the training process. Besides, the training and validation curves exhibited stable convergence behavior and comparatively small generalization gap, with no severe overfitting was observed after implementing the balancing strategy. Finally, the balanced training data set was randomly split across three IoT clients based on stratified client-wise splitting, which ensured that the class proportion remained balanced across the participating clients in the data set and was fair to the participating clients during distributed training.

Table 3. Dataset details

Attribute	Description
Dataset	IoT Beamforming Optimization Dataset
No. of Records	1,000
No. of Features	21
No. of Classes	2 (Binary: Optimized / Not Optimized)
Class Distribution	0 (Not Optimized): 83.2% 1 (Optimized): 16.8%

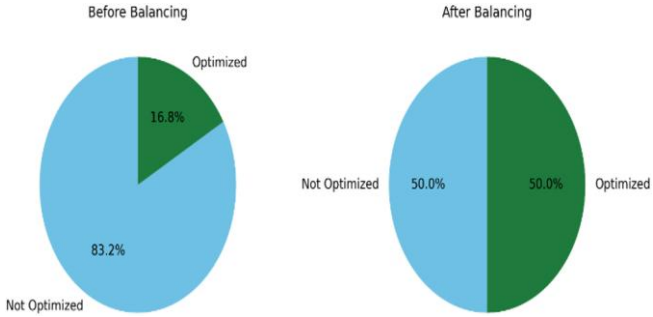

Figure 2. Class distribution before and after applying random oversampling

Table 4. Hyperparameter values and descriptions

Parameter	Value	Description
Number of Clients	3	Simulates distributed learning using three clients.
Global Rounds	100	Total number of communication rounds for model aggregation.
Local Epochs	20	Each client trains locally for 20 epochs before aggregation.
Batch Size	16	Number of samples per batch to balance stability and efficiency.
Client Learning Rate	0.001	Step size for updating local client models.
Server Learning Rate	0.001	Step size for updating the global server model.
Use L2 Regularization	True	Applied to prevent overfitting and improve generalization.
L2 Weight	0.005	Regularization coefficient controlling weight decay.
Dropout Rate	0.4	Dropout fraction to enhance model robustness.
Early Stopping Patience	10	Stops training if validation loss does not improve after 20 epochs.
Gradient Clip Norm	1.0	Limits gradient magnitude to stabilize training.
Random Seed	42	Ensures experiment reproducibility.
Number of Layers	8	Total number of dense layers in the deep neural network.
Client Head Layers	2	Number of layers assigned to the client-side model.
Server Tail Layers	6	Number of layers assigned to the server-side model.

4.2 Model architecture and split strategy

The base model of all learning paradigms is a fully connected deep neural network (DNN). The network consists of eight dense layers, five of them with 50 neurons and ReLU activation followed by Dropout regularization (rate = 0.4). A single-neuron Sigmoid output layer is used for binary classification. The model in the DSFL and SFL configurations is vertically divided following the second dense layer, where the client-side sub-model performs local feature extraction while the server-side sub-model handles higher-level representation learning and classification. The split point is selected to achieve a balance between local computational requirements and communication overhead. All learning paradigms employ the same training settings, including the Adam optimizer (learning rate = 0.001), Binary Cross-Entropy loss, L2 regularization ($\lambda = 0.005$), batch size of 16, and early stopping conditions. The complete network architecture and hyperparameter settings used throughout the experiments are summarized in Table 4. This ensures that any observed performance differences can be attributed to the learning paradigm rather than variations in model architecture or training configuration.

The proposed DSFL framework is summarized in Algorithm 1. Initially, the neural network is partitioned into client-side and server-side sub-models according to the configuration presented in Table 4. During each communication round, clients perform local forward propagation on their private data and transmit only intermediate activations to complete the remaining computations. After backward propagation and parameter updates, decentralized aggregation is conducted through a ring-topology communication mechanism, where each client exchanges model parameters only with its neighboring clients. This process is repeated iteratively until model convergence is achieved.

Algorithm 1. Split federated learning (SFL) training procedure

Inputs: K clients, total rounds T, local epochs E, learning rate η. **Initialize:** Client weights W_c^0, Server weights W_s^0.

1. Global Round (Outer Loop)

For each round $t = 1, 2, \ldots,$ R:

Broadcast: There is no Fed Server; clients keep their current local client-side models and proceed with decentralized coordination.

- **Broadcast:** The Fed Server sends the current global client-side model W_c^t to all selected clients $k \in S_t$.
- **Client/Server Interaction (Inner Loop):** Each client k performs local training for E epochs:
- **Forward Pass:**

1. Client k computes smashed data: $A_k = f(X_k; W_{c,k})$.
2. Client sends A_k and labels Y_k to the **Main Server**.
3. Main Server computes the output: $\hat{Y}_k = f(A_k; W_s)$.
4. Main Server calculates the loss: $L = \ell(\hat{Y}_k, Y_k)$.

- **Backward Pass:**

5. Main Server computes gradients for the server-side: ∇W_s and gradients with respect to the smashed data: $g_{A,k} = \frac{\partial L}{\partial A_k}$.
6. Main Server updates server weights: $W_s \leftarrow W_s - \eta \nabla W_s$.
7. Main Server sends $g_{A,k}$ back to Client k.
8. Client k computes client-side gradients using the chain rule:

$$\nabla W_{c,k} = g_{A,k} \cdot \frac{\partial A_k}{\partial W_{c,k}}$$

9. Client k updates local weights: $W_{c,k} \leftarrow W_{c,k} - \eta \nabla W_{c,k}$.

2. Aggregation Step

$$W_{C_k}^{t+1} = \sum_{j \in \mathcal{N}_k \cup \{k\}} \alpha_{kj} W_{C_j}$$

(where, n_k is the number of samples at client k and n is the total samples).

4.3 Workflow of the proposed decentralized split federated learning framework for 6G Internet of Things systems

Figure 3 illustrates the overall workflow of the proposed DSFL framework, including dataset preprocessing, split learning operations, client-side training, and ring-based peer-to-peer model aggregation among participating clients.

The proposed DSFL framework, during training, combines split learning with full decentralization of federated coordination. A deep neural network is divided at a specific cut layer into two parts: client-side sub-model (the first two dense layers) and the server-side sub-model (the rest of the six layers). The client-side sub-model has each client doing a forward propagation using its own private data. The clients do not send raw data, but send the intermediary activations, also known as smashed data, to the split main server. The server-side sub-model may finish the rest of the forward propagation, calculate the loss function, and do backpropagation. The respective gradients are subsequently returned to the respective client to enable the client-side sub-model update its local parameters without revealing any private information.

The training is performed with 100 rounds of communication, and each participating client executes one local epoch in each round and is then coordinated with the model. It is necessary to mention that, the split server in the DSFL framework is only concerned with performing the server-side sub-model computations and is not involved in aggregating the models centrally. Rather, model aggregation is done in a completely decentralized peer-to-peer fashion among the clients involved.

Once the local split updates are done, the framework does the decentralized model coordination using a ring-based topology. In this topology, every client only transfers its locally updated model parameters to its immediate neighbors in the ring. These values are then received and an average of local parameters is taken and the local updates are propagated in a sequence over the ring network. This distributed coordination scheme removes the point of failure risk of centralized federated learning architectures as well as the risk of communication bottlenecks. This procedure is repeated until the model converges, with several communication rounds. Lastly, the trained model is tested on the unaltered test data in applying the conventional measures of classification performance to determine the effectiveness of the proposed DSFL framework in 6G-enabled IoT settings.

Meanwhile, SL process (bottom segment) separates the neural network between a client and the central server. The initial few layers are dealt with by the client whereas the rest of the layers are executed on the server side. In the training process, the client transmits in-between outputs- known as smashed data to the central server which does the forward calculation, loss calculation and back-propagation. The server is now used to send smashed gradients back to assist the client in updating the local layers. The process has the advantage of lowering the computational burden on client computers and ensuring that the raw data remain hidden. These measures combined, combine the distributed collaboration of FL with the split computation of SL to provide a privacy-preserving, resource-efficient learning model, and can be used in large-scale 6G-enabled IoT systems.

It is also worthwhile to provide the traditional centralized SFL architecture that is currently used in distributed learning systems before presenting the proposed decentralized system. The training process in centralized SFL is based on two central components: a main server (MS) that implements the server-side sub-model of the split neural network and a federated server (FS) that implements global model aggregation.

Learning process starts with dataset preprocessing, in which the raw dataset is first subjected to various preparation steps such as data cleaning, feature normalization and categorical encoding. The processed data is then divided into training, validation and test data. To overcome the imbalance in the classes, the training dataset is balanced and then allocated to many participating clients. The clients are provided with a part of the training data and local computations are carried out by the client in the process of training.

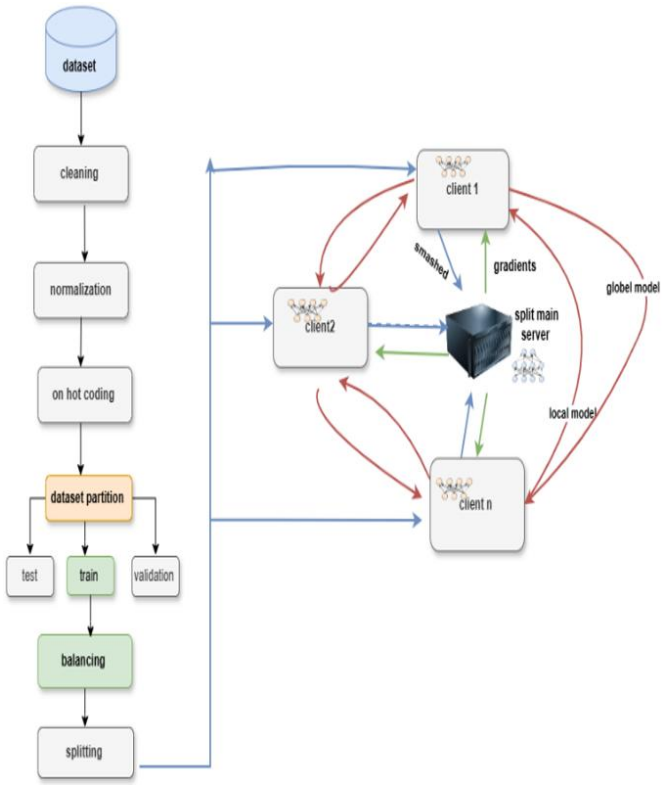

Figure 3. Workflow of the proposed decentralized split federated learning (DSFL) framework, from dataset preprocessing to ring-based peer-to-peer model aggregation among clients

Figure 4 illustrates the architecture of the centralized SFL framework used in this study. In training, the deep neural network is split vertically based on the split learning paradigm. Local forward propagation is performed by each client on its local data using the client-side sub-model.The clients do not send the raw data to the server, but rather, intermediate activations, otherwise known as smashed data to the main server. The rest of the forward propagation is done by the main server, the loss function is calculated and backpropagation done. These gradients are then sent back to the clients to update the client-side model parameters whilst maintaining data privacy.

Besides the split learning mechanism, a centralized stage of aggregation is also added by the federated learning component. Once local training has been accomplished the local model parameters of each client are transmitted to the federated server. The federated server combines these parameters to generate a global model that is again disseminated to all the involved clients to proceed with the subsequent training round. This centralized coordination

makes it easier to synchronize, but may cause some scalability limitations, communication overhead, and single-point-of-failure risk.

Although centralized SFL architecture facilitates learning by collaborating without exchanging raw data, it is still based on centralized coordination using the federated aggregation server. This reliance may result in scalability constraints and communication bottlenecks in large-scale IoT systems. The suggested DSFL architecture, in its turn, eliminates the centralized aggregation server and introduces a decentralized coordination scheme among the participating clients in a peer-to-peer fashion. A ring-based topology allows exchanging model updates directly between adjacent nodes, which makes it possible to perform distributed model aggregation without a centralized control. This decentralized architecture increases the resiliency of the system, removes the risks of single-point-of-failure, and increases the scalability of large-scale 6G IoT systems.

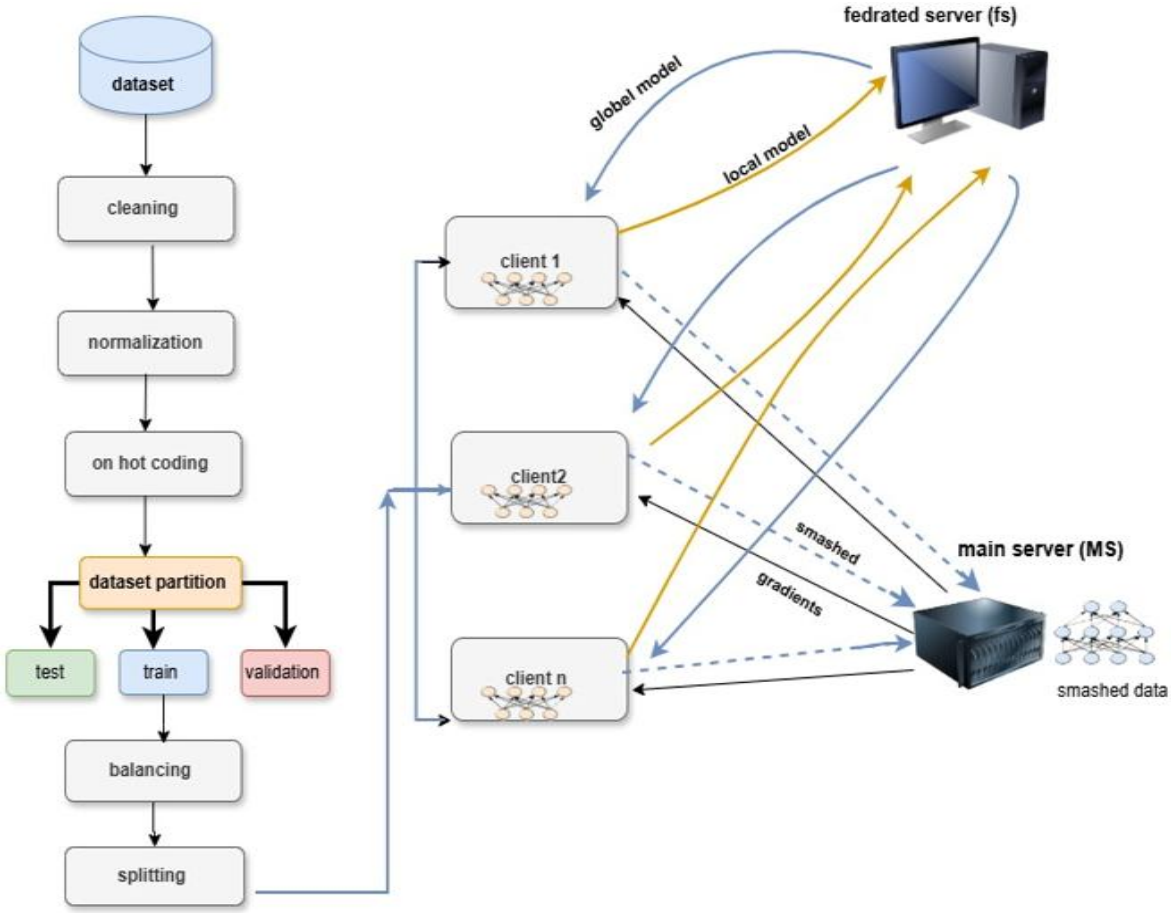

Figure 4. Architecture of centralized split federated learning (SFL)

5. RESULTS AND DISCUSSION

This part contains a detailed comparison between the suggested DSFL system and the centralized split federated learning (CSFL) system. Both models were trained using the same experimental conditions, such as same data partitioning, model architecture and hyperparameter settings, which provided fair and consistent evaluation.

Figure 5 illustrates the training and validation accuracy of the DSFL model over decentralized communication rounds. The training accuracy steadily increases to approximately 95%, while the validation accuracy reaches about 91%, indicating stable learning performance and good generalization capability. Figure 6 presents the training and validation loss curves of the DSFL model. Both curves show a gradual decrease throughout the training process, with a relatively small gap between training and validation loss, indicating low overfitting and stable convergence behavior. Likewise, the centralized method, illustrated in Figure 7, has a stable convergence with a somewhat more smooth behavior because the aggregation is synchronized. The validation accuracy is about 93-94, which shows the impact of centralized coordination in diminishing update variance among clients. Conversely, Figure 8 demonstrates that the CSFL model aso attains a smoother and more stable loss curve. The aggregation mechanism is centralized which leads to a much lesser loss in training of around 0.08, which can also be seen in Table 5 and it signifies closer convergence to optimisation due to the global updates. Figures 9 and 10 present the training and validation precision of the CSFL and DSFL models over communication rounds. Both models exhibit a consistent improvement in precision throughout the training process, indicating effective learning and stable convergence behavior. The training and validation precision curves increase gradually and reach high values by the end of training, demonstrating the ability of both frameworks to correctly identify positive samples. In addition, the relatively small gap between the training and validation precision curves suggests good generalization capability and limited overfitting. The results indicate that both centralized and decentralized approaches achieve comparable precision performance while maintaining stable learning behavior.

Figures 11 and 12 illustrate the training and validation recall of the CSFL and DSFL models over communication rounds. Both models exhibit a steady increase in recall, reaching approximately 0.95 for training and 0.88 for validation. The small gap between the training and validation curves indicates good generalization capability and stable learning performance. Figures 13 and 14 present the confusion matrices of the DSFL and CSFL models. Both models demonstrate strong classification performance, correctly classifying most samples in both classes. DSFL correctly classifies 121 majority-class samples, while CSFL correctly classifies 120 samples. For the minority class, both models correctly identify 23 samples with only two misclassified instances. These results indicate effective classification performance and a low error rate for both frameworks. Figures 15 and 16 present the classification reports of the CSFL and DSFL models, respectively. As shown in Figure 15, the CSFL model achieved an accuracy of 95%, a precision of 0.90, a recall of 0.94, and an F1-score of 0.92. Similarly, Figure 16 shows that the DSFL model achieved an accuracy of 96%, a precision of 0.92, a recall of 0.94, and an F1-score of 0.93. These results indicate that both models provide strong classification performance, while DSFL achieves slightly better overall results. A quantitative comparison of the two methods is provided in Table 5. Even though CSFL has lower training loss because of centralized aggregation, the DSFL framework has a better performance in accuracy (96 percent vs. 95 percent), precision (0.92 vs. 0.90), and F1-score (0.93 vs. 0.91), but the same recall (0.94). This establishes the fact that predictive performance is maintained in decentralized coordination without any central server.

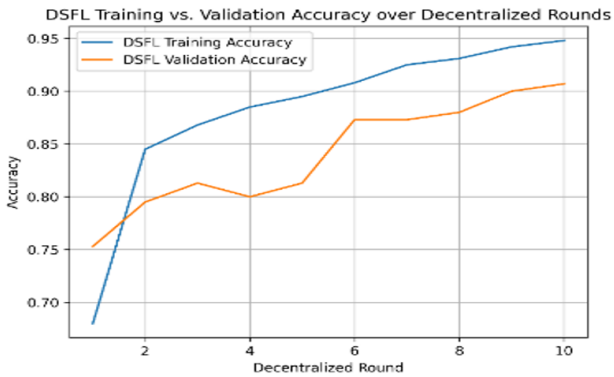

Figure 5. Decentralized split federated learning (DSFL) training and validation accuracy over rounds

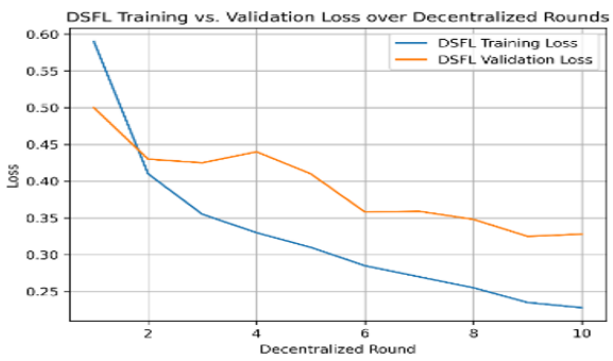

Figure 6. Decentralized split federated learning (DSFL) training and validation loss over rounds

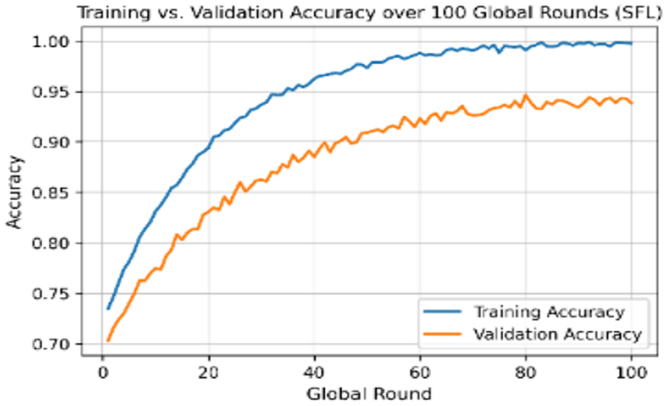

Figure 7. Centralized split federated learning (CSFL) training and validation accuracy over rounds

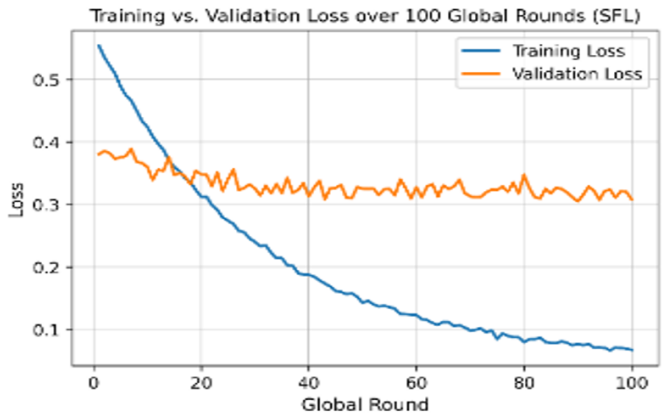

Figure 8. Centralized split federated learning (CSFL) training and validation loss over rounds

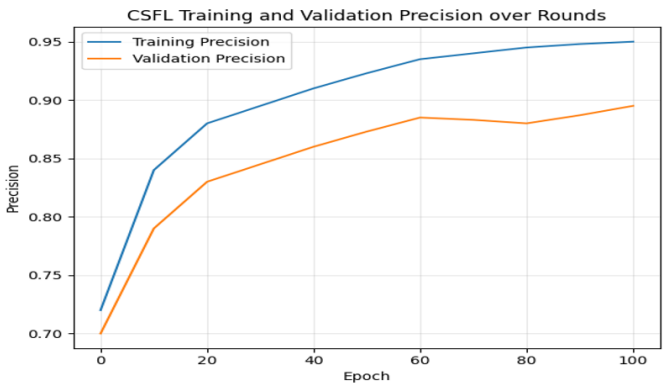

Figure 9. Centralized split federated learning (CSFL) training and validation precision over rounds

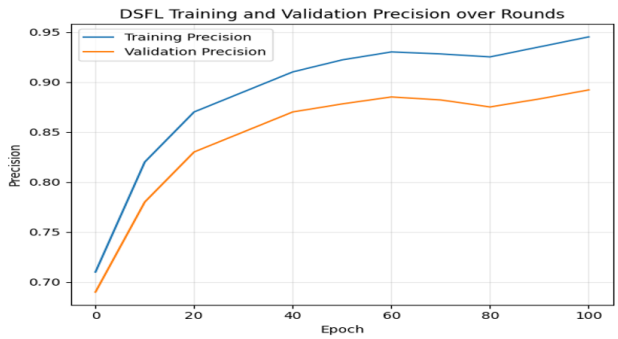

Figure 10. Decentralized split federated learning (DSFL) training and validation precision over rounds

Moreover, DSFL has a number of architectural benefits such as better scalability, increased robustness and the removal of a single point of failure. These features render it especially appropriate to large-scale distributed systems like 6G-capable IoT systems.

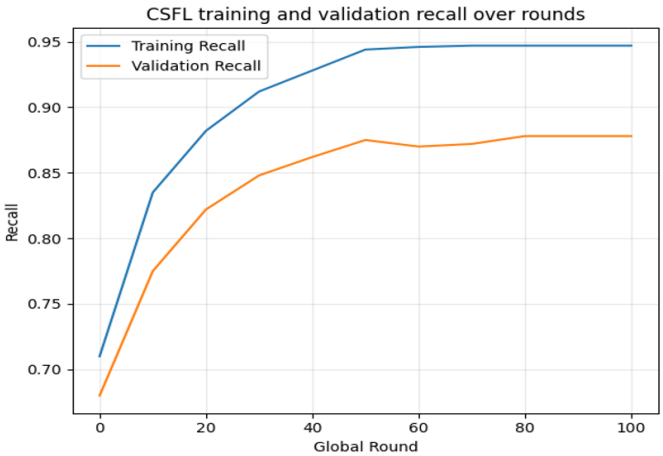

Figure 11. Centralized split federated learning (CSFL) training and validation recall over rounds

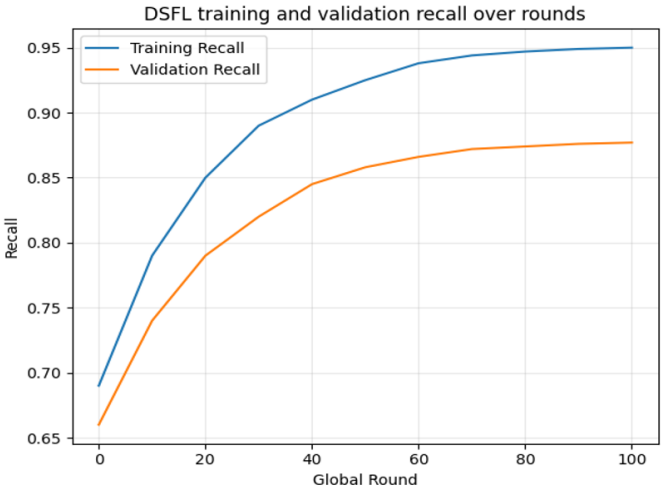

Figure 12. Decentralized split federated learning (DSFL) training and validation recall over rounds

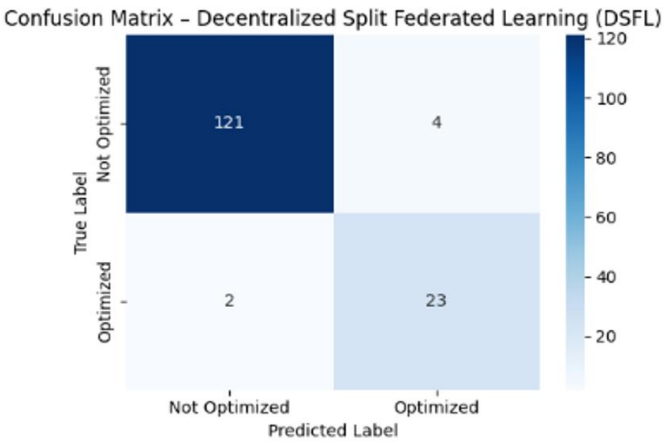

Figure 13. Confusion matrices of decentralized split federated learning (DSFL)

Table 5 indicates that distributed models outperform the baseline approach. Among them, SFL achieved strong performance with an accuracy of 0.95 and an F1-score of 0.91, along with relatively low loss. This demonstrates that combining federated learning and split learning is effective in improving both accuracy and privacy preservation.

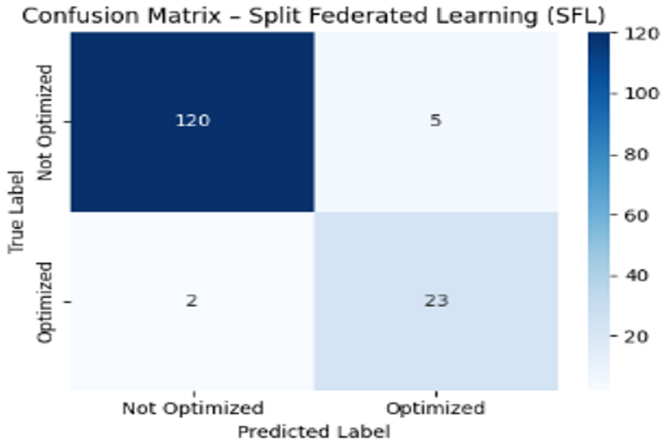

Figure 14. Confusion matrices of centralized split federated learning (CSFL)

Classification Report:

	precision	recall	f1-score	support
Not Optimized	0.98	0.96	0.97	125
Optimized	0.82	0.92	0.87	25
accuracy			0.95	150
macro avg	0.90	0.94	0.92	150
weighted avg	0.96	0.95	0.95	150

Figure 15. Centralized split federated learning (CSFL) classification reports

DSFL Classification Report:

	precision	recall	f1-score	support
Not Optimized	0.98	0.97	0.98	125
Optimized	0.85	0.92	0.88	25
accuracy			0.96	150
macro avg	0.92	0.94	0.93	150
weighted avg	0.96	0.96	0.96	150

Figure 16. Decentralized split federated learning (DSFL) classification reports

Table 5. Comparison of centralized split federated learning (CSFL) and decentralized split federated learning (DSFL) performance

Model	Accuracy	Precision	Recall	F1-Score	Loss
Baseline Learning	0.87	0.80	0.84	0.77	0.41
Split Federated Learning (SFL)	0.95	0.90	0.94	0.91	0.08
DSFL	0.96	0.92	0.94	0.93	0.25

6. CONCLUSION AND FUTURE WORK

This article compared a DSFL system to optimizing beamforming in 6G-enabled IoT that has the drawback of overcoming the limits of a centralized federated and split learning system, especially in scalability, robustness, and exposure to single-point-of-failure. Combining split learning with fully decentralized, peer-to-peer coordination, the proposed structure is able to maintain data locality and to allow collaborative model training among distributed IoT clients without the need to rely on an aggregation server. Experimental outcomes achieved using a real dataset of beamforming optimization prove high classification rates, consistency in convergence, and the ability to deal with the issue of class imbalance, which supports the relevance of the proposed solution to real-life and scalable 6G IoT optimization settings.

In spite of such positive outcomes, the current research is restricted to controlled distributions of IID data, a single number of involved clients, and a prescribed ring-shaped topology of communication. The methodological study of non-IID data distributions is also a major future work direction, which are inherent in physical IoT networks conditions because of the heterogeneity of devices, spatial heterogeneity, and changing network conditions. Future studies will aim to generalize the proposed DSFL framework to non-IID environments through the analysis of their effects on convergence stability, model consistency, and efficiency in communication. Moreover, other forms of decentralized communication topology such as star, mesh, and general graph-based topologies will be investigated to further improve the robustness to data heterogeneity. Future directions involve larger-scale applications with more clients involved, and incorporating more privacy-saving and robustness-enabling techniques, where the goal is to support reliable and efficient DSFL in the context of realistic and dynamic 6G intelligent networks.

6.1 Limitations and future directions

The current experimental evaluation was performed in a controlled environment with a small-scale distributed environment comprising of three distributed clients. The configuration was carefully designed to enable a fair comparison between the two paradigms of distributed learning studied and in the same computation and communication setup. The results shown in the examples demonstrate the feasibility and effectiveness of the proposed DSFL framework, however a practical deployment of the 6G-enabled IoT environments will involve a large number of clients, non-IID data distributions, changing network conditions, and diverse devices. The statistical heterogeneity due to the diversity of device classes, user activities, sensing and communication conditions can affect the convergence stability, efficiency of the communication, and generalization of the model in the real-world decentralized IoT systems. Moreover, the more clients that join, the more complex the interplay of various issues such as synchronization latency, decentralized coordination stability, communication overhead and the ability of different clients. The proposed decentralized ring-based coordination, however, should boost scalability by removing the centralized aggregation bottlenecks with the help of concurrent communication between neighbouring clients and offloading the deeper computations via the split-learning architecture, which will redirect the computational and memory burden from the client.

The proposed framework of DSFL does not provide formal assurances regarding adversarial attacks, inference attacks, or leaks of client information from intermediate outputs of the model and model parameters that are transmitted. Hence, the privacy analysis conducted in this work is restricted to an architectural one, which aims at decentralized coordination and split learning. Furthermore, the deployment involved in the current implementation did not incorporate some of the more sophisticated techniques for privacy and security such as: differential privacy, homomorphic encryption, secure aggregation, blockchain-assisted verification, and adversarially robust decentralized coordination.

Further, there were no quantitative evaluations performed of the metrics of the operational system level, such as communication overhead, bandwidth usage, latency, amount of training communication, throughput, and robustness to node failures at this moment. The primary objective of the present study was to show the feasibility and convergence properties of the proposed DSFL framework in controlled conditions as a proof-of-concept study, while keeping experimental factors constant.

In future, the proposed framework will be further extended to large-scale and more diverse IoT supported by 6G with non-IID distribution of clients, dynamic topology, varying communication environment and increased number of devices involved. The proposed DSFL framework is further examined for its robustness, trustworthiness and applicability in realistic decentralized IoT deployments through further study of communication efficiency, fault tolerance, scalability, and hardware compatibility and advanced communication mechanisms to maintain privacy.

ACKNOWLEDGEMENT

The authors would also like to acknowledge that they have been given a chance to undertake this research through the assistance of the Informatics Institute of Postgraduate Studies (IIPS), University of Information Technology and Communications (UoITC) and the College of Artificial Intelligence, University of Baghdad which has provided the academic environment and support that enabled the realization of this research.

REFERENCES

[1] McMahan, B., Moore, E., Ramage, D., Hampson, S., y Arcas, B.A. (2017). Communication-efficient learning of deep networks from decentralized data. Artificial Intelligence and Statistics, 54: 1273-1282.

[2] Shi, W., Cao, J., Zhang, Q., Li, Y., Xu, L. (2016). Edge computing: Vision and challenges. IEEE Internet of Things Journal, 3(5): 637-646. https://doi.org/10.1109/JIOT.2016.2579198

[3] Liu, B., Lv, N., Guo, Y., Li, Y. (2024). Recent advances on federated learning: A systematic survey. Neurocomputing, 597: 128019. https://doi.org/10.1016/j.neucom.2024.128019

[4] Yang, Q., Liu, Y., Chen, T., Tong, Y. (2019). Federated machine learning: Concept and applications. ACM Transactions on Intelligent Systems and Technology

(TIST), 10(2): 1-19. https://doi.org/10.1145/3298981
[5] Beltrán, E.T.M., Pérez, M.Q., Sánchez, P.M.S., Bernal, S.L., et al. (2023). Decentralized federated learning: Fundamentals, state of the art, frameworks, trends, and challenges. IEEE Communications Surveys & Tutorials, 25(4): 2983-3013. https://doi.org/10.1109/COMST.2023.3315746
[6] Yuan, L., Wang, Z., Sun, L., Yu, P.S., Brinton, C.G. (2024). Decentralized federated learning: A survey and perspective. IEEE Internet of Things Journal, 11(21): 34617-34638. https://doi.org/10.1109/JIOT.2024.3407584
[7] Wu, J., Dong, F., Leung, H., Zhu, Z., Zhou, J., Drew, S. (2024). Topology-aware federated learning in edge computing: A comprehensive survey. ACM Computing Surveys, 56(10): 1-41. https://doi.org/10.1145/3659205
[8] Wang, J., Liang, B., Zhu, Z., Fapi, E.T., Dalal, H. (2024). Communication-efficient network topology in decentralized learning: A joint design of consensus matrix and resource allocation. IEEE Transactions on Networking, 33(2): 761-776. https://doi.org/10.1109/TNET.2024.3511333
[9] Hu, Z., Zhou, T., Wu, B., Chen, C., Wang, Y. (2025). A review and experimental evaluation on split learning. Future Internet, 17(2): 87. https://doi.org/10.3390/fi17020087
[10] Gupta, O., Raskar, R. (2018). Distributed learning of deep neural network over multiple agents. Journal of Network and Computer Applications, 116: 1-8. https://doi.org/10.1016/j.jnca.2018.05.003
[11] Thapa, C., Arachchige, P.C.M., Camtepe, S., Sun, L. (2022). Splitfed: When federated learning meets split learning. Proceedings of the AAAI Conference on Artificial Intelligence, 36(8): 8485-8493. https://doi.org/10.1609/aaai.v36i8.20825
[12] Zhang, W., Zhan, D., Yu, H., Zhang, L., Zhao, B., Du, X., Tian, Z. (2025). Distributed machine learning for next-generation communication networks: A survey on privacy, fairness, efficiency, and trade-offs. Information Fusion, 126: 103657. https://doi.org/10.1016/j.inffus.2025.103657
[13] Shi, Y., Shen, L., Wei, K., Sun, Y., Yuan, B., Wang, X., Tao, D. (2023). Improving the model consistency of decentralized federated learning. International Conference on Machine Learning, 202: 31269-31291. https://doi.org/10.48550/arXiv.2302.04083
[14] Assran, M., Loizou, N., Ballas, N., Rabbat, M. (2019). Stochastic gradient push for distributed deep learning. International Conference on Machine Learning, 97: 344-353. https://doi.org/10.48550/arXiv.1811.10792
[15] Wang, Z., Hu, Y., Yan, S., Wang, Z., Hou, R., Wu, C. (2022). Efficient ring-topology decentralized federated learning with deep generative models for medical data in ehealthcare systems. Electronics, 11(10): 1548. https://doi.org/10.3390/electronics11101548
[16] Beis-Penedo, C., Díaz-Redondo, R.P., Fernández-Vilas, A., Veiga, M.F., Troncoso-Pastoriza, F. (2026). HLF-FSL: A decentralized federated split learning solution for IoT on hyperledger fabric. Array, 29: 100685. https://doi.org/10.1016/j.array.2026.100685
[17] Sun, T., Li, D., Wang, B. (2022). Decentralized federated averaging. IEEE Transactions on Pattern Analysis and Machine Intelligence, 45(4): 4289-4301. https://doi.org/10.48550/arXiv.2104.11375
[18] Tajabadi, M., Martin, R., Heider, D. (2024). Privacy-preserving decentralized learning methods for biomedical applications. Computational and Structural Biotechnology Journal, 23: 3281-3287. https://doi.org/10.1016/j.csbj.2024.08.024
[19] Chen, L., Liu, W., Chen, Y., Wang, W. (2024). Communication-efficient design for quantized decentralized federated learning. IEEE Transactions on Signal Processing, 72: 1175-1188. https://doi.org/10.1109/TSP.2024.3363887
[20] Nedic, A., Ozdaglar, A. (2009). Distributed subgradient methods for multi-agent optimization. IEEE Transactions on Automatic Control, 54(1): 48-61. https://doi.org/10.1109/TAC.2008.2009515

NOMENCLATURE

Symbol	Description
ω	Model parameters
$\omega_i^{(t)}$	Model parameters of client i at round t
$f_c(x)$	Client-side sub-model
$f_s(.)$	Server-side sub-model
n_k	Number of samples at client k
x	Input data
y	Ground truth label
$\hat{Y}$	Predicted output
η	Learning rate
L	Loss function
∇L	Gradient of loss
α=$f_c(x)$	Smashed data
$\hat{y}$=$f_s(\alpha)$	Prediction function
$N(i)$	Neighbors of client i
α_{ij}	Aggregation weight
T	Communication rounds
E	Local epochs
K	Number of clients