Semi-Supervised Split Federated Learning for Image Classification on Edge Devices

Yunus Emre Alçakakan, Ömer Korçak
Marmara University
İstanbul, Türkiye
yunusalcakakan@marun.edu.tr, omer.korcak@marmara.edu.tr

Çiğdem Eroğlu Erdem
Özyeğin University
İstanbul, Türkiye
cigdem.erdem@ozyegin.edu.tr

***Abstract*—Training deep learning models on resource-constrained edge devices is challenging due to limited computational capacity and the scarcity of labeled data. Although Split Federated Learning (SplitFed) reduces client-side computation and helps keep raw data on-device, its performance degrades in label-scarce scenarios. In this paper, we propose Semi-SplitFed, a semi-supervised split federated learning framework that integrates the FixMatch algorithm into the SplitFed architecture without requiring any data on the server side. The proposed method enables clients to use abundant unlabeled data while maintaining data-locality and computational efficiency. Extensive experiments on three different image classification datasets (CIFAR-10, CIFAR-100, and Fashion-MNIST) demonstrate that Semi-SplitFed consistently outperforms the SplitFed baseline under limited labeled data settings, achieving an accuracy improvement of up to 7.56 percentage-point (pp) particularly in low-label regimes. The results highlight the effectiveness of combining semi-supervised learning with SplitFed for low-compute decentralized environments.**

I. Introduction

Today, smartphones, wearable sensors, and IoT devices generate huge amounts of data. These data are used to develop machine learning models in many different application areas such as healthcare, smart transportation systems, and smart home systems, integrating them into our daily lives [1]. In healthcare, it is used to diagnose various diseases, such as cancer detection from images, identification of problems from chest X-ray images, and detection of heart problems from electrocardiogram data [2]. In smart transportation systems, artificial intelligence is used to solve problems such as autonomous driving or traffic forecasting based on traffic car speeds. [3]. Within smart grid frameworks, AI models are leveraged to forecast energy consumption and monitor the network for irregularities [4] [5].

Since a large portion of the data generated by edge-device sensors is sensitive, sending it to central servers for model training is often restricted due to privacy and regulatory concerns. For this reason, approaches have been developed to train AI models on edge devices while keeping the data locally on the device. With Federated Learning, edge device training is performed on the devices themselves without the need to send their raw data to a central server, and global model development is achieved with all participating devices [6]. In this way, data privacy is better supported by keeping data locally on the device while also utilizing data generated by other devices.

However, in the federated learning method, it is not always possible for each device participating in the training to train the machine learning model, which requires large and computationally intensive tasks, as this may exceed the processing power of the device. At this point, a new approach, called split learning, has been developed, which enables low-power clients to participate in machine learning model training [7]. In the split learning method, the large machine learning model to be trained is divided into client and server sub-models at a determined split point. The clients train the first layers of the model, which require less computation. The server trains the layers of the model that require more computation. In split learning, each client processes its raw data through the client-side subnetwork up to a predefined cut layer and transmits the resulting intermediate activations (a.k.a. smashed data) to the server. The server then completes the forward propagation using the remaining layers of the network, computes the loss, and performs backpropagation. The gradients at the cut layer are subsequently sent back to the client to update the client-side parameters. This mechanism reduces the computational burden on the client by offloading the deeper and more computationally intensive layers to the server. In conventional split learning settings [8], clients are typically trained sequentially, which can significantly increase the overall training time.

SplitFed combines the strengths of federated learning and split learning, supporting privacy-preserving training on resource-constrained devices while offering improved training efficiency over conventional split learning [9]. In the SplitFed method, clients train their own split-models in parallel with their own data and send the intermediate results to the server. On the server, the intermediate results for each client are processed in parallel, and the estimated weight updates are sent to the clients. After training is completed on all clients in one round, both client and server global models are combined with federated averaging to obtain updated client and server models.

In many applications, the data generated by edge devices are predominantly unlabeled. Since training a deep learning model with only a small amount of labeled data is generally insufficient, the large amount of unlabeled data available should also

be used. Semi-supervised learning (SSL) techniques have been developed to allow the utilization of unlabeled data in training to increase the performance of the deep learning model [10], [11]. FixMatch [12] is a well-known SSL method, which is based on first pseudo-labeling of weakly-augmented high-confidence unlabeled data and then matching the pseudo-label with the prediction for the strongly-augmented version of the same data using consistency regularization.

Although recent advancements have explored the integration of Split Federated Learning (SFL) and Semi-Supervised Learning (SSL), the existing literature [13] focuses primarily on scenarios where labeled data reside on a central server. Such server-centric labeling assumptions limit the practical deployment of these models in environments where data are generated and maintained locally on edge devices. To the best of our knowledge, this study is among the first to develop an SSL-enabled SplitFed framework that operates without any server-side data, specifically designed for computationally limited devices where all labeled and unlabeled samples remain on the client side.

In this context, the main objective of this study is to investigate how effective learning can be achieved under limited labeled data and resource constraints while preserving data privacy. Specifically, this work addresses the following research questions: (i) how to effectively utilize large amounts of unlabeled data when labeled data are scarce, (ii) how to enable learning without requiring direct access to raw data, and (iii) how to maintain reliable model performance under constrained computational conditions at edge devices.

In this work, we propose a novel semi-supervised split federated learning method, which combines split federated learning [9] and semi-supervised learning [12] methods to enable the training of a deep learning model on edge devices with low computational power and sparse labeled data, while keeping raw data on-device by avoiding direct data sharing. The main contributions of this study are as follows:

- We propose a novel semi-supervised split federated learning framework, namely Semi-SplitFed, that integrates semi-supervised learning (FixMatch) into split federated learning (SplitFed) without requiring any data on the server side, thereby addressing an important gap in the literature.
- The proposed framework enables the effective use of unlabeled client data while supporting data privacy by transmitting only intermediate activations (smashed data) instead of raw data or labels.
- We provide extensive experimental results on the CIFAR-10, CIFAR-100, and Fashion-MNIST datasets with varying levels of label-scarcity, demonstrating consistent and significant performance improvements up to 7.56 pp over the SplitFed baseline.

II. Related work

Many studies have been conducted in the field of distributed machine learning addressing scenarios with limited labeled data. Some of these studies are based on federated learning, while others utilize split learning. In this section, relevant studies from the literature are briefly reviewed.

A. Federated Learning with Partially Labeled Data

In a recent survey, Uludağ et al. [14] address a critical limitation in traditional federated learning (FL) frameworks, which predominantly assume the availability of fully labeled data across all client nodes. In many real-world edge environments, such as healthcare or IoT, data annotation is prohibitively expensive, time-consuming, or requires domain expertise, resulting in partially or completely unlabeled local datasets. To overcome this label scarcity, the authors present a comprehensive review of Federated Learning Utilizing Unlabeled Data (FLUD), focusing on the integration of semi-supervised, unsupervised, and self-supervised learning paradigms within decentralized training environments.

A novel taxonomy is also given in [14], which categorizes FLUD methods based on how they process unlabeled data. The authors first conceptualize FLUD environments into four distinct scenarios depending on the specific location and distribution of labeled versus unlabeled data across the server and participating clients [15], [16]. They also systematically divide existing techniques into three main groups: methods that use pseudo-labeling (which include direct and confidence-based labeling approaches) [17], methods without pseudo-labeling (which leverage techniques such as consistency regularization [15], [16], contrastive learning [3], [18], generative models [19] or autoencoders [20]), and hybrid methods that combine these strategies to mitigate issues such as catastrophic forgetting and biased label generation [21], [22].

These approaches demonstrate the effectiveness of leveraging unlabeled data in federated settings. However, since the entire model is trained on client devices, they may not be suitable for resource-constrained environments where computational capacity is limited.

B. Split Learning with Partially Labeled Data

The studies mentioned in the previous subsection leverage unlabeled data in a federated learning setup. However, since the entire deep learning model is trained on the clients, they are not suitable for devices with low computing capabilities. Hence, methods that combine split learning and semi-supervised learning, which allow low-power devices to train a common model, have been proposed.

One of the first studies to combine split learning and semi-supervised learning is [23], which uses mean teacher [24], FixMatch [12], and virtual adversarial training (VAT) [25]. In [26], the goal is to detect diseases from electrocardiogram signals, and split learning was used in combination with the mean teacher and FixMatch algorithms. In the split learning part, the size of the sent smashed data is reduced by passing it through an auto-encoder layer before it leaves the client. Communication costs are reduced by adaptively not sending data to the server if the gradients calculated on the server are below a threshold value. Performance improvements were

achieved with unlabeled data using a Temporal Convolutional Network (TCN).

While these methods enable learning on resource-constrained devices by distributing the model between client and server, they typically rely on sequential training and do not use federated learning, which can lead to increased training time and potential forgetting of previously learned information.

C. Split Federated Learning with Partially Labeled Data

Studies combining split learning and semi-supervised learning enable machine learning on resource-constrained devices with label scarcity, but they suffer from slow performance due to the lack of parallel processing which is used in federated learning. Furthermore, because processes are performed sequentially in split learning, data from previous clients can be forgotten as training progresses, leading to the catastrophic forgetting problem. Therefore, algorithms are needed that combine split-federated learning with SSL algorithms.

SemiSFL [13] is a method that combines split federated learning and semi-supervised learning, which addresses the problems of sparsely labeled and non-IID data on clients. Based on the teacher-student model of SSL methods, this approach eliminates problems arising from class imbalance during training by establishing a clustering regularization mechanism among teacher features. Dynamically adjusting how much labeled data is trained on the server results in a more stable training process.

Although SemiSFL integrates split federated learning with semi-supervised learning, it operates under the Case 2 paradigm as categorized by Uludağ et al. [14], which requires the availability of labeled data on a central server. This approach may face limitations in scenarios where the server cannot act as an annotator or where all data generation and labeling must occur locally.

To the best of our knowledge, prior studies have not explicitly addressed a Case 1 scenario in which both labeled and unlabeled data remain entirely on the client side for collaborative training on resource-constrained devices. By combining semi-supervised learning with the SplitFed architecture in such decentralized and resource-constrained environments, our proposed method addresses a significant gap in the literature.

III. Proposed Semi-supervised Split Federated Learning Method (Semi-SplitFed)

In this section, we first give an overview of the proposed method and the federated split learning method used. Then, we give details of how the semi-supervised learning FixMatch is integrated into split federated so that no labeled data is needed at the server.

A. Overview of the Method

The proposed system (see Fig. 1) consists of N clients with low computational capabilities and a high-performance central server. Each client possesses a limited amount of labeled data along with a substantially larger volume of unlabeled data, whereas the server does not maintain any dataset.

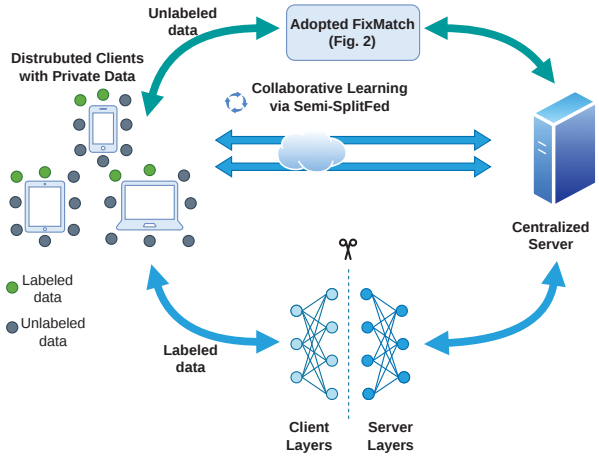

Fig. 1. System overview of Semi-SplitFed. Each clients keeps labeled and unlabeled data locally, generate smashed activations at the cut layer, and transmit them to the server. The server computes logits and the FixMatch losses (supervised + consistency-based unsupervised) without any server-side labels, then returns cut-layer gradients; client and server submodels are aggregated across clients each round.

The method is built upon the SplitFed Learning architecture, where computationally intensive tasks from concurrently operating clients are delegated to the server. This design reduces client-side workload while enabling inter-client model sharing to support collaborative training.

To address limited labeled data, the FixMatch semi-supervised algorithm is incorporated into the SplitFed framework. The abundant unlabeled data at each client is exploited to enhance learning performance and improve the overall accuracy of the model. The overall system architecture is illustrated in Fig. 1.

B. Split Federated Learning (SplitFed)

The SplitFed [9] method partitions a deep neural network in a designated cut layer, assigning the initial few computationally light layers to the client side, while assigning the computationally intensive deeper layers to the server. During forward propagation, intermediate activations (smashed data) generated by the client-side subnetwork are transmitted to the server, where they are processed to produce the corresponding logits. The loss is computed on the server and the resulting gradients are propagated back to the clients through the split boundary.

Let $X_{L_i,k}$ denote an input sample with a ground-truth (one-hot) label $Y_{i,k}$ at the client k, where $k \in \{1, \ldots, N\}$. In each communication round of split federated learning, on the client side, each input $X_{i,k}$ is processed through the client subnetwork f with parameters W_k^C to produce intermediate activations as follows:

$$A_{i,k} = f(X_{i,k}; W_k^C). \quad (1)$$

These activations are sent to the server, which applies its server-side subnetwork with parameters W_G^S to generate logits as follows:

$$Z_{i,k} = g(A_{i,k}; W_G^S). \tag{2}$$

The predicted class probability distribution for the input $X_{L_i,k}$ is then obtained by applying the softmax function to the logits as follows:

$$p(Y|X_{L_i,k}) = \text{Softmax}(Z_{i,k}). \tag{3}$$

For a mini-batch of size B at client k, the cross-entropy loss $\mathcal{L}_k$ between the true labels and the model predictions is computed at the server as:

$$\mathcal{L}_k = \frac{1}{B}\sum_{i=1}^{B} H(Y_{i,k}, p(Y|X_{L_i,k})), \tag{4}$$

where $H(.,.)$ is the cross-entropy function.

After computing the loss $\mathcal{L}_k$, the server updates its parameters as $W_k^S \leftarrow W_G^S$ by gradient-descent and backpropagation. In accordance with the split learning mechanism, the gradients at the cut layer are transmitted back to the clients, enabling the client-side subnetworks to update their parameters accordingly. At the end of each training round, the client and server models are aggregated using the FedAvg [6] method to obtain globally synchronized parameters:

$$W_G^C \leftarrow \frac{1}{N}\sum_{k=1}^{N} W_k^C \tag{5}$$

$$W_G^S \leftarrow \frac{1}{N}\sum_{k=1}^{N} W_k^S \tag{6}$$

The updated global client model W_G^C is then distributed to all participating clients, while the global server model W_G^S is retained by the server to initiate the next training round.

C. Adaptation of FixMatch to Split Federated Learning

FixMatch [12] is a fundamental semi-supervised learning approach that enables effective utilization of unlabeled data during training. The method relies on the assumption that unlabeled samples with highly confident label predictions should preserve their predicted labels under strong data augmentation.

The proposed integration of FixMatch into the SplitFed framework is as follows. In each round of federated split learning at client k, let X_L denote labeled data with (one-hot) ground-truth labels Y and let X_U denote unlabeled data. For unlabeled samples, X_U^{weak} denotes inputs obtained through basic data augmentation operations such as random flipping and small translations, whereas X_U^{strong} denotes inputs generated through more aggressive and diverse image transformations that substantially alter the visual appearance of the sample. We will omit the sample index i and the client index k to simplify the notation.

The client-side model with parameters W_k^C computes intermediate activations (smashed data) as follows:

$$A_L = f(X_L; W_k^C), \tag{7}$$

$$A_U^{weak} = f(X_U^{weak}; W_k^C), \tag{8}$$

$$A_U^{strong} = f(X_U^{strong}; W_k^C), \tag{9}$$

which are then sent to the server. The server-side model with parameters W_G^S then computes the corresponding logits:

$$Z_L = g(A_L; W_G^S), \tag{10}$$

$$Z_U^{weak} = g(A_U^{weak}; W_G^S), \tag{11}$$

$$Z_U^{strong} = g(A_U^{strong}; W_G^S). \tag{12}$$

For weakly augmented unlabeled samples, the predicted class probability distribution is computed as follows:

$$Q_U^{weak} = \text{Softmax}(Z_U^{weak}). \tag{13}$$

The pseudo-label (one-hot) $\hat{Y}$ and the confidence mask M are defined as follows:

$$\hat{Y} = \arg\max(Q_U^{weak}), \tag{14}$$

$$M = \mathbb{1}(\max(Q_U^{weak}) \geq \tau). \tag{15}$$

If the confidence threshold τ is not satisfied, $M = 0$ and the unsupervised loss becomes zero. The supervised loss for labeled data in each batch is computed as follows.

$$\mathcal{L}_s = \frac{1}{B}\sum_{B} H(Y, \text{Softmax}(Z_L)) \tag{16}$$

The unsupervised loss is defined via consistency regularization between the pseudo-label derived from weak augmentation and the prediction of the strongly augmented counterpart on each batch:

$$\mathcal{L}_u = \frac{1}{B}\sum_{B} M \cdot H(\hat{Y}, \text{Softmax}(Z_U^{strong})). \tag{17}$$

The total loss is formulated as:

$$\mathcal{L}_{total} = \mathcal{L}_s + \lambda\mathcal{L}_u, \tag{18}$$

where λ denotes the unsupervised loss weight that controls the contribution of the unlabeled data term.

Based on $\mathcal{L}_{total}$, both client and server models are updated according to the SplitFed training procedure, completing a communication round. The adaptation of FixMatch to the SplitFed framework, named the proposed Semi-SplitFed method, is illustrated in Fig. 2 and is described in Algorithm 1.

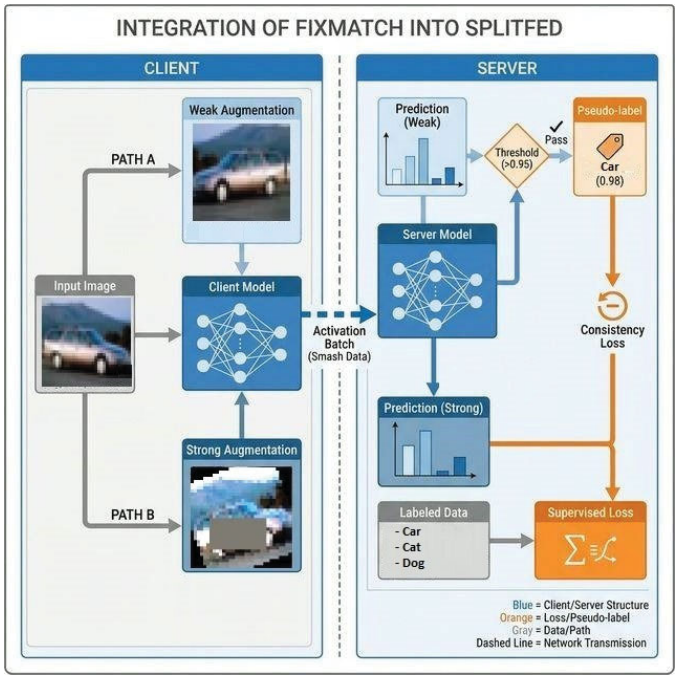

Fig. 2. Proposed Semi-SplitFed framework, Path A and path B denote the weak and strong data augmentations for unlabeled data at the clients.

IV. Experimental Study

This section presents the experimental setup and results for evaluating the proposed Semi-SplitFed framework. We first describe the experimental setup, followed by the datasets used in our experiments, and the implementation details. We then report the image recognition results on three datasets. Finally, we compare our approach with the original SplitFed method under varying labeled data ratios.

A. Experimental Setup

For Semi-SplitFed, we adapt the split federated learning structure using the public codebase of [9] to semi-supervised learning by modifying the data pipeline and server-side loss computations.

ResNet-8 was used as the deep learning model, which consists of an input layer, 3 residual blocks, and a fully connected classification layer. The client model consists of the input and the first residual layer, and the server model consists of the rest of the network.

Training is conducted with $N = 5$ clients using global rounds $T = 400$. In each round, clients perform $E = 1$ local epoch with a batch size of $B = 256$. We used the SGD optimizer with the Nesterov momentum value of 0.9 and the weight decay as 5×10^{-4}, similar to the FixMatch study [12]. Initially, the learning rate was set as $\eta = 0.03$, and the learning rate was changed throughout the training using cosine annealing.

For semi-supervised learning, a pseudo-labeling threshold of $\tau = 0.95$ was used. The unsupervised loss parameter was defined as $\lambda = 1.0$, balancing its contribution with the supervised loss.

Algorithm 1 Proposed Semi-SplitFed Framework

Parameters: Number of clients N, global rounds T, local epochs E, learning rate η, FixMatch pseudo-label threshold τ, unsupervised loss weight λ.
Client Data: Each client $k \in \{1, \ldots, N\}$ holds a local dataset $\mathcal{D}_k = L_k \cup U_k$, where L_k is the labeled subset and U_k is the unlabeled subset.
Initialize: Global client model $W_G^C \sim \mathcal{N}(0, \sigma^2)$, global server model $W_G^S \sim \mathcal{N}(0, \sigma^2)$
Outputs: Client-side W_G^C and server-side models W_G^S
for round $t = 1$ to T **do**
  for each client $k = 1$ to N in parallel **do**
    Synchronize local models: $W_k^C \leftarrow W_G^C$
    for local epoch $e = 1$ to E **do**
      for batch $\{X_L, X_U\} \subset \mathcal{D}_k$ **do**
        Compute X_U^{weak}, X_U^{strong}
        Forward Pass (Client → Server):
        Compute smashed data using (7) – (9)
        Send smashed data to server
        Compute server logits using (10) – (12)
        Loss Computation at server:
        Compute pseudo-labels using (13) – (15)
        Compute $\mathcal{L}_s$ and $\mathcal{L}_u$ using (16) – (17)
        Compute total loss $\mathcal{L}_{total}$ as in (18)
        Backward Pass (Split Learning):
        Update server model W_k^S using $\mathcal{L}_{total}$
        Send cut-layer gradients to client
        Update client model W_k^C
      end for
    end for
    Send updated client weights W_k^C to server
  end for
  Federated Aggregation:
  Aggregate client models W_G^C using (5)
  Aggregate server models W_G^S using (6)
  Broadcast W_G^C to all clients
end for

All experiments were performed on Apple M4 Pro chip with 24 GB memory. Clients were simulated on a single machine.

B. Datasets

We evaluate the proposed framework on three widely used benchmark image classification datasets:

Fashion-MNIST (FMNIST) [27]: This data set consists of 28×28 grayscale images belonging to 10 different fashion items (e.g. T-shirt, shoes, bag), with 60,000 training samples and 10,000 test samples. Although it has a structure similar to MNIST, it is slightly more challenging.

CIFAR-10 [28]: This dataset consists of 32×32 color images belonging to 10 different classes (e.g., airplane, automobile, bird), with 50,000 training samples and 10,000 test samples.

CIFAR-100 [28]: Similar to CIFAR-10, this dataset consists of 32×32 color images belonging to 100 different classes, with

TABLE I. CLASSIFICATION ACCURACIES OF THE PROPOSED METHOD (SEMI-SPLITFED) ON DIFFERENT DATASETS. COMPARISON WITH THE BASELINE SPLITFED AND FULLY-LABELED UPPERBOUND SPLITFED RESULTS ARE ALSO GIVEN FOR DIFFERENT AMOUNTS OF LABELED DATA. THE BEST RESULTS ARE SHOWN IN BOLD.

Dataset	Method	Labeled Data Ratio	Samples/Class	Accuracy (%)	Improvement (pp)
Fashion-MNIST	Supervised SplitFed (Baseline)	1%	60	81.27±0.8	+3.92
	Semi-SplitFed (Ours)	**1%**	**60**	$\mathbf{85.19 \pm 0.86}$	
	Fully Supervised SplitFed (Upperbound)	100%	6,000	94.2±0.07	
CIFAR-10	Supervised SplitFed (Baseline)	10%	500	77.74±0.27	+7.56
	Semi-SplitFed (Ours)	**10%**	**500**	$\mathbf{85.3 \pm 0.1}$	
	Fully Supervised SplitFed (Upperbound)	100%	5,000	89.75±0.14	
CIFAR-100	Supervised SplitFed (Baseline)	30%	150	52.58±0.21	+5.88
	Semi-SplitFed (Ours)	**30%**	**150**	$\mathbf{58.46 \pm 0.31}$	
	Fully Supervised SplitFed (Upperbound)	100%	500	65.62±0.28	
CIFAR-100	Supervised SplitFed (Baseline)	10%	50	37.23±0.18	+5.55
	Semi-SplitFed (Ours)	**10%**	**50**	$\mathbf{42.78 \pm 1.28}$	
	Fully Supervised SplitFed (Upperbound)	100%	500	65.62±0.28	

50,000 training samples and 10,000 test samples.

To simulate various labeled data ratios, a parameter is defined, and the labeled data is distributed to all clients in an IID manner.

Data augmentation methods were applied to the datasets at clients. Weak augmentation is used for both supervised and unsupervised learning, while strong augmentation is used only for unsupervised learning. The weak and strong augmentations applied to the datasets are as follows:

Weak Augmentation: For FMNIST, random horizontal flipping and random cropping with a padding of 2 pixels are applied. For CIFAR-10 and CIFAR-100, the augmentation is similar but uses a padding size of 4 pixels.

Strong Augmentation: For all datasets, two augmentations (e.g., contrast, rotation, brightness changes) on top of the weak augmentation are applied. Then, the Cutout (Random Erasing) augmentation technique is applied, which randomly removes a rectangular region from the image.

C. Image Classification Results

We evaluate the performance of the proposed Semi-SplitFed framework under different labeled data ratios. The comparative results are summarized in Table I. We benchmark our method against two reference methods, both based on the SplitFed [9] framework:

- Supervised SplitFed (Baseline): The standard Split Federated Learning approach trained on the available labeled data subset, representing the lower bound performance in label-scarcity cases.
- Supervised SplitFed (Upperbound): The ideal scenario in which 100% of training data is labeled, delivering the best possible performance.

We experimented with various degrees of scarcity of labels, ranging from 1% to 30% labeled data ratios, corresponding to different numbers of labeled samples per class.

Table I presents the classification accuracies under different label-scarcity settings. Across all datasets and labeled data ratios, the proposed Semi-SplitFed framework consistently outperforms the supervised SplitFed baseline using the same amount of labeled data. Each experiment is repeated three times, and the reported accuracies correspond to the average performance over the last 50 communication rounds.

In particular, on the CIFAR-10 dataset using only 10% labeled data, Semi-SplitFed improves the baseline accuracy by 7.56 pp and achieves an accuracy close to the fully supervised upper bound. Even in the most challenging setting for the CIFAR-100 dataset with 10% labeled data, the proposed method yields a substantial improvement of 5.55 pp, demonstrating its robustness and scalability in highly label-scarce scenarios.

To analyze the training dynamics under limited labeled data, Fig. 3 presents the evolution of test accuracy over communication rounds on CIFAR-10 with 10% labeled data. During the early training phase (approximately the first 100 rounds), the supervised SplitFed baseline achieves slightly faster accuracy gains. This behavior is expected, as the baseline is optimized solely using ground-truth labels, whereas Semi-SplitFed additionally incorporates pseudo-labeled samples whose reliability improves progressively during training. As model confidence increases, Semi-SplitFed gradually exceeds the baseline and maintains a consistent performance advantage in subsequent rounds.

In later communication rounds, the supervised SplitFed baseline exhibits intermittent sharp accuracy drops, indicating instability under label-scarce supervision. These fluctuations are substantially reduced in Semi-SplitFed, which converges to a smoother and more stable performance plateau.

The training time analysis in Fig. 4 indicates that Semi-SplitFed incurs higher wall clock time than the SplitFed baseline. This overhead is due to additional computation introduced

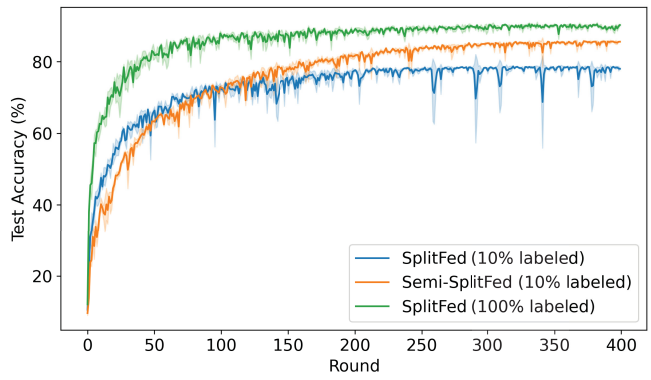

Fig. 3. Test accuracy versus communication rounds on CIFAR-10. Curves represent the mean performance over three independent runs, and the shaded regions denote ± one standard deviation.

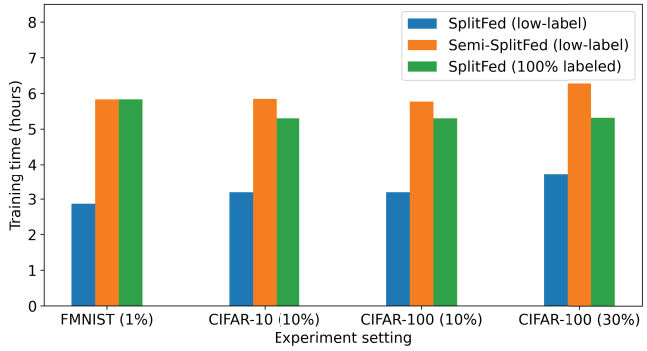

Fig. 4. Wall-clock training time (hours) after 400 communication rounds across different datasets and labeled data ratios. The comparison includes SplitFed with limited labels, Semi-SplitFed (Ours), and the fully supervised SplitFed (100% labeled) setting.

by unlabeled data processing and dual data augmentation in the FixMatch framework. However, the total training duration remains close to that of the fully supervised SplitFed (100% labeled) setting. This suggests that Semi-SplitFed operates at a computational scale comparable to that of the ideal fully labeled scenario while effectively exploiting unlabeled data under limited label availability.

V. Limitations

The proposed framework is designed in accordance with the Case 1 scenario defined in [14], where both labeled and unlabeled data reside entirely on client devices without requiring any server-side data. Accordingly, the method assumes settings in which such a data distribution is available.

The proposed Semi-SplitFed framework also assumes that unlabeled data across clients share a similar label space. In practical federated learning scenarios, data distributions of different classes may vary significantly across clients, which could influence the behavior of pseudo-labeling-based methods.

In addition, the current framework does not explicitly consider more challenging semi-supervised settings such as open-set scenarios, where unlabeled data may include out-of-distribution (OOD) samples outside the labeled data space. Handling such cases remains an interesting direction for further investigation.

Despite these limitations, the proposed method provides an effective and practical solution for semi-supervised split federated learning under commonly adopted settings, and serves as a strong basis for future extensions to more complex scenarios.

VI. Conclusion and Future Work

In this study, we addressed the challenge of training machine learning models on devices with limited computational capacity while handling sparsely labeled data and keeping raw data on-device by avoiding direct data sharing. To tackle this problem, we proposed a semi-supervised SplitFed framework that combines the SplitFed architecture with the FixMatch semi-supervised learning algorithm, enabling efficient training while supporting data privacy by avoiding direct raw-data sharing. Moreover, the proposed Semi-SplitFed method does not require the existence of any data on the server. The method was evaluated on widely used benchmark datasets, including CIFAR-10, CIFAR-100, and Fashion-MNIST, demonstrating its applicability to both simple and complex image classification tasks.

The experimental results show that the proposed framework consistently outperforms the supervised SplitFed baseline across all datasets and labeled data ratios. In particular, Semi-SplitFed achieves performance close to the fully supervised upper bound on CIFAR-10 with only 10% labeled data and provides substantial improvements in the most challenging CIFAR-100 settings. These findings highlight the robustness and scalability of the method and its potential for deployment in real-world applications on low-compute edge devices.

For future work, we plan to extend the evaluation to real-world datasets, which often present non-IID distributions across clients. Investigating the performance of the framework under such heterogeneous and realistic conditions will provide deeper insight into its practical applicability and potential limitations. Future investigations may also explore refinements in model design and adjustments to semi-supervised learning settings to further improve the framework's flexibility and effectiveness. In addition, future work will include a detailed analysis of communication cost by analyzing the overhead introduced by the transmission of intermediate activations between clients and the server, and evaluating its impact on overall system efficiency.

Acknowledgment

The authors acknowledge the use of AI-assisted tools based on large language models (LLMs) for figure creation, grammar and writing refinement in certain parts of this manuscript. The sentences rephrased by the tools were reviewed and edited by the authors to ensure their accuracy and relevance.

REFERENCES

[1] D. C. Nguyen, M. Ding, P. N. Pathirana, A. Seneviratne, J. Li, and H. V. Poor, "Federated learning for internet of things: A comprehensive survey," *IEEE communications surveys & tutorials*, vol. 23, no. 3, pp. 1622–1658, 2021.

[2] D. C. Nguyen, Q.-V. Pham, P. N. Pathirana, M. Ding, A. Seneviratne, Z. Lin, O. Dobre, and W.-J. Hwang, "Federated learning for smart healthcare: A survey," *ACM Computing Surveys (Csur)*, vol. 55, no. 3, pp. 1–37, 2022.

[3] F. Zhang, K. Kuang, L. Chen, Z. You, T. Shen, J. Xiao, Y. Zhang, C. Wu, F. Wu, Y. Zhuang *et al.*, "Federated unsupervised representation learning," *Frontiers of Information Technology & Electronic Engineering*, vol. 24, no. 8, pp. 1181–1193, 2023.

[4] N. Mendes, J. Mendes, J. Mohammadi, and P. Moura, "Federated learning framework for prediction of net energy demand in transactive energy communities," *Sustainable Energy, Grids and Networks*, vol. 40, p. 101522, 2024.

[5] J. Jithish, B. Alangot, N. Mahalingam, and K. S. Yeo, "Distributed anomaly detection in smart grids: a federated learning-based approach," *IEEE Access*, vol. 11, pp. 7157–7179, 2023.

[6] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas, "Communication-efficient learning of deep networks from decentralized data," in *Artificial intelligence and statistics*. PMLR, 2017, pp. 1273–1282.

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

[8] H. Madaan, M. G. V. Kulkarni, and A. Pant, "Vulnerability due to training order in split learning," in *ICT Systems and Sustainability: Proceedings of ICT4SD 2021, Volume 1*. Springer, 2022, pp. 103–112.

[9] C. Thapa, P. C. M. Arachchige, S. Camtepe, and L. Sun, "SplitFed: When federated learning meets split learning," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 36, no. 8, 2022, pp. 8485–8493.

[10] X. Yang, Z. Song, I. King, and Z. Xu, "A survey on deep semi-supervised learning," *IEEE transactions on knowledge and data engineering*, vol. 35, no. 9, pp. 8934–8954, 2022.

[11] Y. Chen, X. Tan, B. Zhao, Z. Chen, R. Song, J. Liang, and X. Lu, "Boosting semi-supervised learning by exploiting all unlabeled data," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 7548–7557.

[12] K. Sohn, D. Berthelot, N. Carlini, Z. Zhang, H. Zhang, C. A. Raffel, E. D. Cubuk, A. Kurakin, and C.-L. Li, "Fixmatch: Simplifying semi-supervised learning with consistency and confidence," *Advances in neural information processing systems*, vol. 33, pp. 596–608, 2020.

[13] Y. Xu, Y. Liao, H. Xu, Z. Sun, L. Huang, and C. Qiao, "Semisfl: Split federated learning on unlabeled and non-iid data," 2023, unpublished. [Online]. Available: https://arxiv.org/abs/2307.15870

[19] A. Wijesinghe, S. Zhang, S. Qi, and Z. Ding, "Ufed-gan: Secure federated learning over wireless sensor networks with unlabeled data," in *2024 IEEE International Conference on Communications Workshops (ICC Workshops)*. IEEE, 2024, pp. 1048–1053.

[14] K. Uludag, Ç. E. Erdem, and Ö. Korçak, "Leveraging unlabeled data in federated learning: A review," *IEEE Access*, 2025.

[15] D. Yang, Z. Xu, W. Li, A. Myronenko, H. R. Roth, S. Harmon, S. Xu, B. Turkbey, E. Turkbey, X. Wang *et al.*, "Federated semi-supervised learning for covid region segmentation in chest ct using multi-national data from china, italy, japan," *Medical image analysis*, vol. 70, p. 101992, 2021.

[16] Q. Liu, H. Yang, Q. Dou, and P.-A. Heng, "Federated semi-supervised medical image classification via inter-client relation matching," in *International Conference on Medical Image Computing and Computer-Assisted Intervention*. Springer, 2021, pp. 325–335.

[17] Y. Liu, X. Shang, Y. Zhang, Y. Lu, C. Gong, J.-H. Xue, and H. Wang, "Mind the gap: Confidence discrepancy can guide federated semi-supervised learning across pseudo-mismatch," in *Proceedings of the Computer Vision and Pattern Recognition Conference*, 2025, pp. 10 173–10 182.

[18] Y. Wu, D. Zeng, Z. Wang, Y. Sheng, L. Yang, A. J. James, Y. Shi, and J. Hu, "Federated contrastive learning for dermatological disease diagnosis via on-device learning," in *2021 IEEE/ACM International Conference On Computer Aided Design (ICCAD)*. IEEE, 2021, pp. 1–7.

[20] R. Beuran *et al.*, "Fedmse: Semi-supervised federated learning approach for iot network intrusion detection," *Computers & Security*, vol. 151, p. 104337, 2025.

[21] M. Li, Q. Li, and Y. Wang, "Class balanced adaptive pseudo labeling for federated semi-supervised learning," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2023, pp. 16 292–16 301.

[22] S. Chen and J. Shen, "Exploitation maximization of unlabeled data for federated semi-supervised learning," *IEEE Transactions on Emerging Topics in Computational Intelligence*, 2024.

[23] M. Barhoush, A. Ayad, and A. Schmeink, "Semi-supervised learning in distributed split learning architecture and IoT applications," in *2023 IEEE 15th International Symposium on Autonomous Decentralized System (ISADS)*. IEEE, 2023, pp. 1–6.

[24] A. Tarvainen and H. Valpola, "Mean teachers are better role models: Weight-averaged consistency targets improve semi-supervised deep learning results," *Advances in neural information processing systems*, vol. 30, 2017.

[25] T. Miyato, S.-i. Maeda, M. Koyama, and S. Ishii, "Virtual adversarial training: a regularization method for supervised and semi-supervised learning," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 41, no. 8, pp. 1979–1993, 2018.

[26] A. Ayad, M. Barhoush, M. Frei, B. Völker, and A. Schmeink, "An efficient and private ecg classification system using split and semi-supervised learning," *IEEE Journal of Biomedical and Health Informatics*, vol. 27, no. 9, pp. 4261–4272, 2023.

[27] H. Xiao, K. Rasul, and R. Vollgraf, "Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms," *arXiv preprint arXiv:1708.07747*, 2017.

[28] A. Krizhevsky, G. Hinton *et al.*, "Learning multiple layers of features from tiny images," 2009.