DASFL: Dynamic Adaptive Split Federated Learning for Heterogeneous Clients

Aijing Li[1,3], **Yawen Li**[2*], **Guanhua Ye**[1], **Dandan Liu**[1,4], **Tong Zhao**[1] and **Zeli Guan**[1]

[1]School of Computer Science (National Pilot School of Software Engineering), Beijing University of Posts and Telecommunications, China

[2]School of Economics and Management, Beijing University of Posts and Telecommunications, China

[3]Century College, Beijing University of Posts and Telecommunications, China

[4]Zaozhuang University, China

{aijing_li, warmly0716, g.ye, luckydan6798, zhaotong, guanzeli}@bupt.edu.cn

Abstract

Split Federated Learning (SFL) has emerged as a pivotal paradigm for privacy-preserving distributed training on resource-constrained edge devices by partitioning neural networks between clients and a server. A critical design choice in SFL is the split layer, which determines the computation distribution and the semantic level of smashed data, directly impacting communication overhead, training latency, and model accuracy. Existing SFL methods largely ignore the time-varying nature of edge resources, relying on static resource profiles that lead to suboptimal efficiency and compromise model convergence. In this paper, we propose **D**ynamic **A**daptive **S**plit **F**ederated **L**earning (DASFL), a unified framework that jointly addresses dynamic resource heterogeneity and non-IID data distributions. We employ a resource-aware dynamic split layer selection strategy that enables each client to minimize per-round latency by adapting to instantaneous local conditions. We also design an adaptive masked aggregation that robustly synchronizes client updates under split-induced structural heterogeneity. Extensive experiments on multiple benchmark models and datasets, conducted under heterogeneous, time-varying resource conditions and non-IID data distributions, demonstrate that DASFL achieves a superior accuracy-efficiency trade-off and faster convergence compared to state-of-the-art SFL baselines.

1 Introduction

With the rapid deployment of deep neural networks on edge intelligence applications, collaborative model training under strict privacy and resource constraints has become a fundamental challenge [Kairouz *et al.*, 2021; Liu *et al.*, 2024]. In practice, data are generated and retained on geographically distributed edge devices with highly heterogeneous and time-varying computation capabilities and communication bandwidths, such as in mobile perception [Gecer and Garbinato, 2024], smart surveillance [Uzzaman, 2025], and Internet of Vehicles (IoV) [Jamil *et al.*, 2022]. Federated learning (FL) [McMahan *et al.*, 2017; Li *et al.*, 2020] enables privacy-preserving collaborative training by keeping raw data on device. However, FL requires all clients to repeatedly train the full model, which becomes increasingly impractical as modern deep networks grow in depth and complexity. Under heterogeneous and time-varying system conditions, computation-limited clients frequently become stragglers, severely degrading training efficiency. Split learning (SL) [Gupta and Raskar, 2018; Vepakomma *et al.*, 2018; Lin *et al.*, 2024a] alleviates this issue by partitioning a model across the client–server boundary, allowing clients to offload deep computation to the server. Despite reduced client-side computation, SL typically relies on sequential client updates, limiting scalability in multi-client settings.

Split federated learning (SFL) [Thapa *et al.*, 2022] combines the advantages of FL and SL by enabling parallel multi-client training while offloading part of the model computation to the server. A critical design choice in SFL is the selection of the split layer [Wu *et al.*, 2023; Hu *et al.*, 2025], which governs both the distribution of computation and the representation level of exchanged smashed data (i.e., intermediate activations). Shallow splits incur high communication overhead due to large low-level feature maps [Dachille *et al.*, 2025], whereas deeper splits reduce communication cost at the expense of increased client-side computation, directly affecting end-to-end training latency and system efficiency. Beyond efficiency, split layer selection also shapes learning behavior by exposing the server to different representation spaces [Joshi *et al.*, 2022], leading to distinct optimization trajectories and generalization properties. Importantly, system heterogeneity is often coupled with data heterogeneity, as devices with different resource profiles typically collect data from diverse and skewed distributions. Under such non-IID settings, split layer choices induce distinct inductive biases and convergence behaviors [Arafeh *et al.*, 2025], making learning performance highly sensitive to how computation is partitioned.

Despite these impacts, existing SFL methods predominantly rely on static split configurations. Uniform split strategies implicitly assume homogeneous client capabilities [Thapa *et al.*, 2022; Chopra *et al.*, 2021], while hetero-

*Corresponding author

geneous but fixed splits [Zhang *et al.*, 2024; Zhu *et al.*, 2024; Shen *et al.*, 2023] fail to adapt to time-varying computation and communication conditions. As a result, static split decisions lead to inefficient and suboptimal training in dynamic edge environments, motivating dynamic split layer selection as a natural solution for improving system efficiency. However, dynamic split selection introduces a largely overlooked challenge: structural heterogeneity. As clients adopt different split layers over time, they update different subsets of model parameters, causing local updates to become misaligned in the parameter space. This breaks the assumptions of conventional federated aggregation methods and can severely harm learning stability and accuracy, especially under non-IID data distributions [Zhu *et al.*, 2021].

These observations reveal two tightly coupled challenges in dynamic SFL: (i) how to adapt split decisions to heterogeneous and time-varying resources to improve training efficiency, and (ii) how to robustly aggregate structurally heterogeneous updates under non-IID data to preserve model accuracy and convergence.

To address these challenges, we propose **D**ynamic **A**daptive **S**plit **F**ederated **L**earning (DASFL), a unified framework for efficient SFL under dynamic resource heterogeneity and non-IID data distributions. For Challenge (i), we propose a resource-aware dynamic split layer selection strategy, which enables each client to adaptively select its split layer at every communication round based on instantaneous computation and communication conditions. This strategy minimizes per-round training latency and mitigates straggler effects. For Challenge (ii), We propose an adaptive masked federated aggregation mechanism that robustly aggregates client updates, mitigating aggregation interference caused by non-IID data and heterogeneous local models. Extensive experiments on multiple models and datasets demonstrate that DASFL consistently outperforms SFL baselines in both accuracy and end-to-end training latency.

Our main contributions are summarized as follows:

- We propose a resource-aware dynamic split layer selection strategy that enables clients to adapt split decisions to heterogeneous and time-varying computation and bandwidth resources, significantly improving training efficiency in SFL.
- We design an adaptive mask-based aggregation mechanism that supports robust aggregation under non-IID data distributions and split-induced structural heterogeneity, by selectively aggregating informative client-side updates while suppressing misaligned parameters.
- We develop a unified DASFL framework that tightly couples dynamic system-aware split adaptation with learning-aware aggregation, achieving stable convergence and superior accuracy-efficiency trade-offs under realistic edge heterogeneity.

2 Related Work

Existing SFL approaches primarily aim to improve scalability, communication efficiency, and privacy preservation, while typically adopting a fixed split layer across all clients and communication rounds. SplitFed [Thapa *et al.*, 2022] extends classical SL by incorporating federated averaging for both client-side and server-side sub-models, enabling parallel training while maintaining global model consistency. Subsequent efforts improve SFL from several directions, including efficient parallelization and gradient reduction [Lin *et al.*, 2024b], privacy-preserving activation sharing [Ma *et al.*, 2023], communication compression [Oh *et al.*, 2025], and hierarchical edge-assisted training [Khan *et al.*, 2023; Fan *et al.*, 2025]. communication–privacy trade-off in [Zhang *et al.*, 2023]. Despite these advances, most existing SFL methods rely on static split configurations and implicitly assume homogeneous system resources, limiting their applicability in dynamic and resource-constrained environments.

Heterogeneity is a fundamental challenge in SFL, encompassing both system heterogeneity due to diverse device capabilities and statistical heterogeneity arising from non-IID data distributions [Kairouz *et al.*, 2021]. From a theoretical perspective, the convergence behavior of SFL under heterogeneous data distributions has been formally analyzed in [Han *et al.*, 2024; Lin *et al.*, 2025b], providing important insights into the stability of SFL optimization. To address system heterogeneity in SFL, AdaptSFL [Lin *et al.*, 2025a] allows clients with different capabilities to adopt different split layers, while EdgeSplit [Zhang *et al.*, 2024] and ESFL [Zhu *et al.*, 2024] optimize model partitioning and resource allocation through joint optimization formulations. From the aggregation perspective, sparse or selective update mechanisms have been explored to reduce communication and suppress noisy updates, such as Deep Gradient Compression [Lin *et al.*, 2018] and FedMask [Li *et al.*, 2021].While most existing approaches consider client heterogeneity, they typically assume static resource conditions and aligned model structures, overlooking the non-stationary nature of computation and communication resources in practical edge environments.

3 Methodology

We propose Dynamic Adaptive Split Federated Learning (DASFL), a unified framework that enables efficient and robust SFL under dynamic system heterogeneity and non-IID data distributions. DASFL consists of three tightly coupled components: (1) Resource-aware dynamic split layer selection, where each client adaptively chooses its split layer at every round based on instantaneous computation and communication resources; (2) Split learning under structurally heterogeneous local models, where clients and the main server collaboratively train different subsets of model parameters according to the selected split layers; and (3) Adaptive masked federated aggregation, tightly integrated with the split learning process, where informative client-side updates are selectively aggregated on the federated server. The overall training procedure is summarized in Figure 1 and Algorithm 1.

3.1 Problem Definition

We consider split federated learning (SFL) with heterogeneous clients $\mathcal{K} = \{1, \ldots, K\}$ under non-IID data. Client k owns a private dataset $\mathcal{D}_k$ and never shares raw data. We adopt a dual-server architecture with a Main Server (MS) executing SL on the server-side sub-model, and a Federated

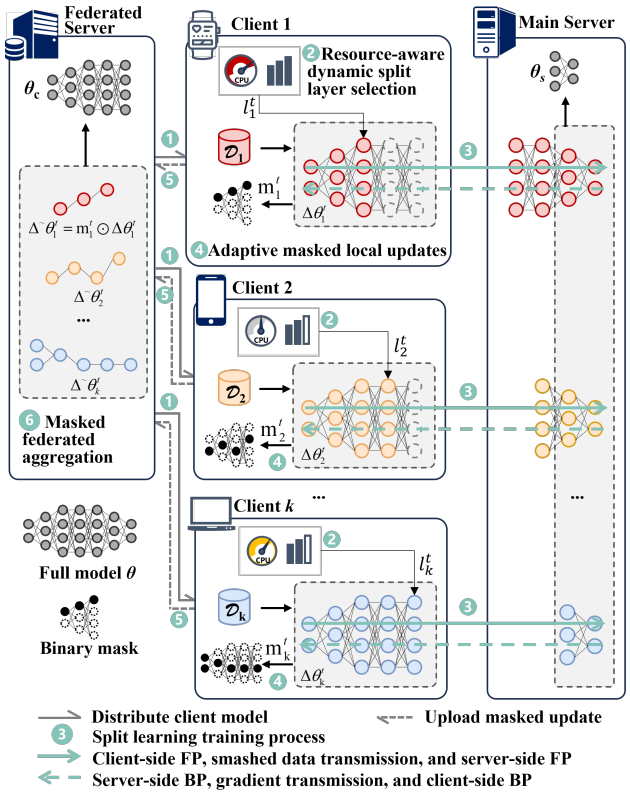


Figure 1: The overview of DASFL

Server (FS) aggregating masked client-side updates. Given a candidate split layer $l \in \mathcal{L}$, the model is partitioned into $f_c^{(l)}(\cdot;\boldsymbol{\theta}_c)$ and $f_s^{(l)}(\cdot;\boldsymbol{\theta}_s)$. At round t, client k selects $l_k^t \in \mathcal{L}$, inducing structural heterogeneity where clients update different parameter subsets. We aim to optimize $\boldsymbol{\theta} = \{\boldsymbol{\theta}_c, \boldsymbol{\theta}_s\}$ by minimizing the aggregated empirical risk:

$$\min_{\boldsymbol{\theta}_c,\boldsymbol{\theta}_s} \frac{1}{K}\sum_{k=1}^{K} \mathbb{E}_{(\mathbf{x},y)\sim\mathcal{D}_k}\left[\ell\Big(f_s^{(l)}\big(f_c^{(l)}(\mathbf{x};\boldsymbol{\theta}_c),\boldsymbol{\theta}_s\big),y\Big)\right] \quad (1)$$

Dynamic split l_k^t affects which layers participate in SFL and aggregation at each round, but not the objective itself.

3.2 Resource-aware Dynamic Split Layer Selection

Dynamic system heterogeneity makes a fixed split point suboptimal: a shallow split reduces client computation but increases smashed-data traffic, while a deep split does the opposite. To adapt to time-varying resources, DASFL performs *per-round, per-client* split-point selection. Specifically, each client estimates the round latency of every candidate split layer by combining (i) client-side training compute time determined by the partitioned sub-model and its instantaneous compute capability, and (ii) communication time including smashed-data exchange with the MS and masked-update with the FS. The client then selects the split layer that minimizes the estimated latency in Eq. (6), enabling decentralized adaptation without extra coordination.

Resource State Model. Each client k has maximum compute capacity $C_k^{\max}$ and uplink budget $B_k^{\max}$. At round t, its effective compute is

$$C_{k,t} = C_k^{\max}\left(1-\rho_{k,t}^{\text{cpu}}\right) \quad (2)$$

where $\rho_{k,t}^{\text{cpu}} \in [0,1]$ is CPU utilization, and available uplink bandwidth is $b_{k,t} \le B_k^{\max}$.

Computation Time Estimation. Let $\text{MAC}_c^{\text{FP}}(l)$ be the MACs for one forward pass of $f_c^{(l)}$. We approximate per-sample training cost as

$$\text{MAC}_c(l) \approx \gamma\,\text{MAC}_c^{\text{FP}}(l) \quad (3)$$

with $\gamma \approx 2$. Let n_k be the number of samples processed per round by client k (e.g., $n_k = E\cdot$batch size). Then

$$T_{k,t}^{\text{comp}}(l) = \frac{n_k\,\text{MAC}_c(l)}{C_{k,t}} \quad (4)$$

We omit MS computation time, assuming it is stable and dominated by edge-side latency.

Communication Time Estimation. In SFL, communication includes SL smashed-data exchange between client and MS and masked-update upload and client-model broadcast between client and FS.

For split layer l, SL requires bidirectional transfer of smashed data and gradients, modeled as $2\,\text{Size}_{\text{act}}(l)$ per round. FS-side aggregation traffic is denoted by $\text{Size}_{\text{param}}(l)$, representing the *actual* per-round upload+download volume (dense zero-masked tensor or sparse index–value pairs with index overhead). Using the uplink-dominant approximation with effective bandwidth $b_{k,t}$, we estimate

$$T_{k,t}^{\text{comm}}(l) = \frac{2\,\text{Size}_{\text{act}}(l) + \text{Size}_{\text{param}}(l)}{b_{k,t}} \quad (5)$$

Dynamic Split Layer Selection. By jointly considering computation and communication costs, client k selects its split layer at round t by minimizing the estimated latency:

$$l_k^t = \arg\min_{l\in\mathcal{L}}\left(\beta_1 T_{k,t}^{\text{comp}}(l) + \beta_2 T_{k,t}^{\text{comm}}(l)\right) \quad (6)$$

where β_1 and β_2 are weighting coefficients that balance computation and communication overheads. This lightweight decision rule enables each client to adapt its split layer to heterogeneous and time-varying resource conditions without introducing additional learning or coordination overhead. We set $\beta_1 = \beta_2 = 1$ by default.

3.3 Split Learning Training and Adaptive Masked Federated Aggregation

We describe the training and aggregation procedure of DASFL under dynamically selected split layers, which naturally induce structurally heterogeneous local models across clients.

Algorithm 1 DASFL Under One Federated Round

Input: Client set $\mathcal{K}$; candidate split layer set $\mathcal{L}$; initial client-side and server-side parameters $\boldsymbol{\theta}_c^{t-1}$, $\boldsymbol{\theta}_s^{t-1}$
Output: Updated $\boldsymbol{\theta}_c^t$, $\boldsymbol{\theta}_s^t$

for all clients $k \in \mathcal{K}$ **in parallel do**
  Sense instantaneous resources and select split layer l_k^t using Eq. (6)
  Client→MS (SL FP): perform client-side FP up to layer l_k^t and compute smashed data $\mathbf{a}_k^t$
  Transmit $\mathbf{a}_k^t$ to the MS
end for
for all received $\mathbf{a}_k^t$ from clients $k \in \mathcal{K}$ **do**
  MS-side SL: perform server-side FP and BP
  MS→Client (SL BP): return $\nabla \mathbf{a}_k^t$ to client k
end for
for all clients $k \in \mathcal{K}$ **in parallel do**
  Complete BP on the client-side sub-model and update local client-side parameters
  Compute local **client-side** update $\Delta\boldsymbol{\theta}_k^t$ (w.r.t. $\boldsymbol{\theta}_c^{t-1}$)
  Generate mask $\mathbf{m}_k^t$ using Eqs. (7)–(8)
  Obtain masked update $\tilde{\Delta\boldsymbol{\theta}}_k^t$ using Eq. (9)
  Client→FS: upload $\tilde{\Delta\boldsymbol{\theta}}_k^t$
end for
FS aggregation: update client-side global parameters $\boldsymbol{\theta}_c^t$ using Eq. (10)
MS aggregation: synchronize server-side parameters $\boldsymbol{\theta}_s^t$ on the MS (FedAvg over shared server-side layers)
FS→Clients: broadcast updated client-side parameters $\boldsymbol{\theta}_c^t$ to all clients

Split Learning under Structurally Heterogeneous Local Models. At communication round t, client k selects a split layer l_k^t and performs SL collaboratively with the MS, which hosts the server-side sub-model. Specifically, client k executes FP on its local sub-model up to layer l_k^t to compute the smashed data $\mathbf{a}_k^t = f_c^{(l_k^t)}(\mathbf{x}; \boldsymbol{\theta}_c)$, and transmits them to the MS. Upon receiving $\mathbf{a}_k^t$, the MS completes the remaining forward propagation using the server-side sub-model, computes the training loss $\hat{y} = f_s^{(l_k^t)}(\mathbf{a}_k^t; \boldsymbol{\theta}_s)$, and performs backward propagation. The gradient $\nabla \mathbf{a}_k^t$ with respect to the smashed data is then returned to client k. Client k completes back-propagation through its client-side sub-model and updates the corresponding local parameters. Since different clients may select different split layers at each round, only a subset of the global model parameters is updated by each client, resulting in structurally heterogeneous local models.

Adaptive Masked Local Updates. To address non-IID data distributions and structural heterogeneity induced by dynamic split layers, each client generates adaptive masked local updates before federated aggregation. At round t, client k computes its local update $\Delta\boldsymbol{\theta}_{c,k}^t = \boldsymbol{\theta}_{c,k}^t - \boldsymbol{\theta}_c^{t-1}$. Let $\mathcal{L}_k^t$ denote the set of effective layers updated by client k and N_ℓ the number of parameters in layer $\ell \in \mathcal{L}_k^t$. Given a base keep ratio α, we allocate the update budget evenly across effective layers to avoid large layers dominating the retained updates and to ensure that each effective layer contributes to aggregation. The per-layer budget is

$$K_\ell = \frac{\alpha}{|\mathcal{L}_k^t|} \sum_{\ell' \in \mathcal{L}_k^t} N_{\ell'} \qquad \ell \in \mathcal{L}_k^t \tag{7}$$

Accordingly, smaller layers have higher effective keep ratios, while larger layers are more aggressively subsampled. This design prioritizes balanced cross-layer participation rather than proportional per-layer retention. To avoid extreme imbalance, the layer-specific keep ratio $r_\ell = K_\ell / N_\ell$ is clipped to $[r_{\min}, r_{\max}]$. For each layer, a quantile-based threshold ε_ℓ^t is determined such that approximately a fraction r_ℓ of parameters is retained. The resulting binary mask is constructed as

$$\mathbf{m}_{k,\ell,i}^t = \mathbb{I}\big(\big|\Delta\theta_{k,\ell,i}^t\big| > \varepsilon_\ell^t\big) \tag{8}$$

and the masked local update is given by

$$\tilde{\Delta\boldsymbol{\theta}}_{c,k}^t = \mathbf{m}_k^t \odot \Delta\boldsymbol{\theta}_{c,k}^t \tag{9}$$

Eq. (9) is the *only* operation that applies masking to the update values, i.e., unselected entries are set to zero (or omitted under sparse transmission).

Masked Federated Aggregation. After local training and adaptive masking, clients upload their masked client-side updates $\tilde{\Delta\boldsymbol{\theta}}_{c,k}^t$ to the federated server (FS) for global aggregation. Since different clients may update different subsets of parameters due to dynamic split layers, the FS performs coordinate-wise normalized aggregation: each parameter is averaged only over clients that actually updated it in the current round. Recall that $\tilde{\Delta\theta}_{c,k,i}^t = m_{k,i}^t \Delta\theta_{c,k,i}^t$; thus $m_{k,i}^t$ in the denominator only counts effective contributors for coordinate i. Concretely, the FS updates each client-side parameter $\theta_{c,i}^t$ by

$$\theta_{c,i}^t = \theta_{c,i}^{t-1} + \frac{\sum_{k=1}^K w_k\, m_{k,i}^t \Delta\theta_{c,k,i}^t}{\sum_{k=1}^K w_k\, m_{k,i}^t + \varepsilon} \tag{10}$$

where w_k is proportional to the local dataset size of client k, and ε is a small constant for numerical stability. Note that $m_{k,i}^t$ in the denominator does *not* apply a second masking to the update. Instead, it identifies the set of effective contributors for coordinate i in round t so that the update is normalized by the total weight of clients that participated in updating that coordinate. Client-side parameters not updated by any client (i.e., $\sum_{k=1}^K m_{k,i}^t = 0$) remain unchanged.

When sparse upload is adopted, the binary mask $\mathbf{m}_k^t$ need not be transmitted explicitly. Client k uploads index–value pairs for the retained coordinates $\mathcal{S}_k^t = \{i : m_{k,i}^t = 1\}$. The FS reconstructs $\sum_k m_{k,i}^t w_k$ in Eq. (10) by accumulating weights over received indices, making the aggregation exactly consistent with the sparsified communication protocol.

3.4 Convergence Analysis

We further provide a convergence analysis (Appendix A) showing that DASFL admits a standard stationarity guarantee under smoothness, with additional error terms that depend on the masking ratio and the update coverage under dynamic split.

4 Experiments

4.1 Experiment Settings

Models and Datasets. We evaluate DASFL on three representative models and datasets: LeNet [LeCun *et al.*, 1998] on Fashion-MNIST [Xiao *et al.*, 2017], ResNet-18 on CIFAR-10, and ResNet-50 on CIFAR-100 [He *et al.*, 2016; Krizhevsky *et al.*, 2009]. These configurations cover lightweight to deep architectures and varying task complexity.

Baselines. We compare DASFL with representative methods from federated learning, split learning, and heterogeneity-aware split federated learning. (1) FL [McMahan *et al.*, 2017]: standard federated learning with full-model local training and FedAvg aggregation. (2) SL [Gupta and Raskar, 2018]: split learning with a fixed split layer and sequential client updates. (3) SFL [Thapa *et al.*, 2022]: parallel split learning with federated aggregation. For SL and SFL, we evaluate a shallow split ($l = s$) that minimizes client-side computation, and a middle split ($l = m$) that balances computation and communication costs. We further compare with heterogeneous SFL methods. (4) EdgeSplit [Zhang *et al.*, 2024], (5) ESFL [Zhu *et al.*, 2024], and (6) AdaptSFL [Lin *et al.*, 2025a].

Metrics. We report average test accuracy to evaluate learning performance and round latency to measure system efficiency. The per-round completion time is defined as the makespan $T^{\text{round}} = \max_{k\in\mathcal{K}_t}\left(T^{\text{comp}}_{k,t} + T^{\text{comm}}_{k,t}\right)$. In addition, we report the *average per-client* computation and communication times $\overline{T}_{\text{comp}} = \frac{1}{T}\sum_{t=1}^{T}\frac{1}{|\mathcal{K}_t|}\sum_{k\in\mathcal{K}_t} T^{\text{comp}}_{k,t}$ and $\overline{T}_{\text{comm}} = \frac{1}{T}\sum_{t=1}^{T}\frac{1}{|\mathcal{K}_t|}\sum_{k\in\mathcal{K}_t} T^{\text{comm}}_{k,t}$. These averages characterize typical client costs and are not intended to sum to T^{round}.

Data Heterogeneity. We evaluate both IID and non-IID data distributions across clients. Non-IID partitions are generated using a Dirichlet distribution [Hsu *et al.*, 2019], with $\alpha = 0.5$ and $\alpha = 0.1$ representing moderate and severe data heterogeneity, respectively.

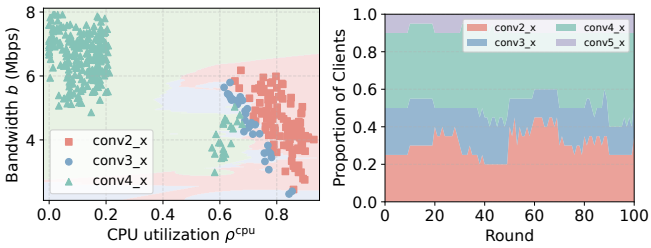
System Heterogeneity. We emulate heterogeneous and time-varying edge environments by modeling client-side computation and uplink bandwidth variability. Clients exhibit diverse compute and network capabilities, and their resource availability fluctuates over training rounds to reflect realistic background workloads and wireless dynamics. We consider heterogeneous clients with diverse computation and communication capabilities. Specifically, clients are grouped into three computation tiers (low, medium, and high) and two network conditions (poor and good), capturing practical device- and network-level heterogeneity. Detailed resource models are provided in Appendix B.

Split Layer Configuration. We fix the candidate split layer set $\mathcal{L}$ across all methods to ensure fair comparison. For ResNet-18, candidate split layers are defined at the outputs of the four main residual stages: $\mathcal{L} = \{\texttt{conv2\_x}, \texttt{conv3\_x}, \texttt{conv4\_x}, \texttt{conv5\_x}\}$, which correspond to increasing client-side model depths and progressively higher-level feature representations. This design provides a representative spectrum of computation–communication trade-offs. For static SL and SFL baselines, we consider two commonly used configurations: a shallow split ($l{=}s$) at `conv2_x`, which minimizes client-side computation, and a middle split ($l{=}m$) at `conv4_x`, which balances computation and communication costs. The candidate split layer configurations for other models (LeNet and ResNet-50), together with their corresponding computation cost (MACs) and activation size profiling results, are provided in Appendix B. All split point profiles are obtained offline and used as lookup tables during training.

Parameter Settings. We consider a federated setting with 20 clients and a participation rate of 0.5 per round. Each selected client performs one local epoch before aggregation. Standard optimizers and learning rates are used for different models. Full hyperparameter settings and profiling details are provided in Appendix B.

4.2 Performance Analysis

Accuracy under Data Heterogeneity. Table 1 reports the final test accuracy under IID and non-IID data distributions. Overall, DASFL consistently outperforms all baselines across all model–dataset pairs. The performance gap becomes more pronounced as data heterogeneity increases. Under highly non-IID settings (Dirichlet $\alpha = 0.1$), DASFL achieves up to 5.2–16.4 percentage points higher accuracy than static-split SFL and AdaptSFL, demonstrating strong robustness to severe data skew. In contrast, methods with fixed split configurations suffer from degraded convergence, as a single split layer cannot simultaneously accommodate heterogeneous clients and evolving system conditions.



(a) (b)

Figure 2: Resource-aware Split Layer Dynamic Selection under Heterogeneous and Time-varying Resources. (a) Dynamic Split Layer Decisions of a Medium-tier Client during Training in the CPU—Bandwidth Space. (b) Evolution of Split Layer Proportions over Communication Rounds.

Training Efficiency and Latency. Table 2 summarizes the training latency in terms of computation, communication, and per-round latency. Conventional FL incurs prohibitively high computation cost for deep models, while SL and static-split SFL exhibit strong sensitivity to the chosen split layer. Shallow splits amplify communication overhead, whereas deeper splits increase client-side computation and straggler effects. By dynamically adapting split layers to instantaneous resource conditions, DASFL achieves the lowest or near-lowest

Model-Dataset	LeNet-FMNIST			ResNet18-CIFAR10			ResNet50-CIFAR100		
Dirichlet α	10	0.5	0.1	10	0.5	0.1	10	0.5	0.1
FL	89.88±0.21	87.59±0.57	85.26±0.84	83.34±0.49	81.17±0.40	61.24±1.01	76.49±0.34	73.39±1.09	66.62±0.91
SL(l=s)	90.63±1.60	86.01±0.81	85.52±0.78	83.05±0.44	70.32±1.07	54.06±0.39	65.63±0.71	63.82±0.68	61.08±1.34
SL(l=m)	90.66±1.72	87.12±0.66	85.59±0.90	83.83±0.30	80.87±0.88	53.01±0.71	74.77±0.80	72.90±0.92	65.10±1.15
SFL(l=s)	88.89±0.83	80.74±1.55	75.70±1.04	77.90±0.62	77.72±0.94	58.02±1.20	76.14±0.54	73.03±1.69	56.09±1.41
SFL(l=m)	89.86±0.78	84.97±1.35	80.86±1.10	84.81±0.76	80.65±0.21	61.73±0.82	76.97±0.72	73.45±1.36	57.27±1.18
EdgeSplit	89.30±1.04	85.56±1.20	76.86±1.17	84.69±0.54	78.58±1.22	61.56±1.53	77.43±1.29	72.48±1.97	59.21±1.36
ESFL	87.67±1.34	84.88±1.12	77.98±1.45	83.47±0.94	80.62±1.63	56.78±1.77	75.81±1.71	71.63±1.92	58.30±1.69
AdaptSFL	89.44±1.32	87.97±1.41	70.28±1.32	84.18±0.56	81.28±1.76	50.48±1.72	77.98±1.79	74.04±1.39	53.25±1.87
DASFL	**91.27±0.77**	**89.80±1.29**	**87.68±1.53**	**85.75±0.58**	**82.56±1.56**	**66.96±1.64**	**78.61±1.90**	**76.18±2.03**	**69.98±1.86**

Table 1: Test accuracy comparison across different datasets. For each baseline, the average of final local test accuracy over all clients is reported. We run each baseline 3 times. The top results are emphasized in bold.

Model-Dataset	LeNet-FMNIST			ResNet18-CIFAR10			ResNet50-CIFAR100		
Latency(s)	$\overline{T}_{\text{comp}}$	$\overline{T}_{\text{comm}}$	T^{round}	$\overline{T}_{\text{comp}}$	$\overline{T}_{\text{comm}}$	T^{round}	$\overline{T}_{\text{comp}}$	$\overline{T}_{\text{comm}}$	T^{round}
FL	4.58	0.47	**17.84**	419.95	123.28	1744.74	1336.22	269.74	5376.28
SL(l=s)	0.53	53.43	275.60	162.86	219.50	1921.74	187.33	880.06	5428.39
SL(l=m)	1.11	15.86	86.71	232.29	120.63	1796.75	262.26	502.21	3889.30
SFL(l=s)	0.55	24.57	68.32	167.98	223.17	1040.79	190.89	894.79	3026.13
SFL(l=m)	1.14	16.19	51.43	236.70	122.65	1067.93	267.24	510.62	2062.73
EdgeSplit	1.20	10.91	37.08	233.33	107.84	1003.74	286.22	428.55	1913.91
ESFL	1.22	9.80	30.50	220.96	103.40	968.38	295.12	390.79	1838.96
AdaptSFL	1.19	10.38	26.21	219.12	106.27	<u>913.05</u>	281.17	421.36	<u>1809.24</u>
DASFL	0.70	12.06	<u>21.74</u>	210.13	89.83	**738.24**	330.42	365.91	**1585.69**

Table 2: Training latency under different models and datasets. T_{round} is the per-round makespan (slowest-client completion time), while $\overline{T}_{\text{comp}}$ and $\overline{T}_{\text{comm}}$ are average per-client costs; therefore they do not add up to T_{round}.

per-round latency in most settings, and is particularly advantageous for deeper models where FL becomes compute-prohibitive, reducing round latency by up to 19.1-30.9% compared with static SFL baselines.

Dynamic Split Layer Behavior. Figure 2 shows that DASFL converges not to a fixed split layer, but to a resource-conditioned adaptation pattern. Figure 2(a) shows that split choices form structured regions in the CPU–bandwidth space: deeper splits are selected under favorable resource conditions, while shallower splits are preferred when computation or communication becomes constrained. Thus, similar resource conditions tend to induce similar split decisions. Figure 2(b) shows that no single split persistently dominates over rounds. The observed variation therefore reflects structured adaptation to heterogeneous and time-varying resources, rather than unstructured oscillation.

4.3 Ablation Study

We conduct ablation studies on ResNet-18 to analyze the individual contributions of adaptive masked aggregation and resource-aware dynamic split layer selection. The results are summarized in Table 3.

Impact of adaptive masked aggregation. We first fix the dynamic split layer selection mechanism and replace the proposed adaptive masked aggregation with three alternative aggregation strategies, denoted as DASFL-a. *Full* uploads both client-side and server-side sub-model parameters to the federated server, where a complete model is reconstructed and aggregated. *Com* aggregates only the common client-side parameters shared across all clients. *Layer-wise* performs layer-level aggregation on client-side sub-models.

As shown in Table 3, all three variants result in inferior accuracy–latency trade-offs compared to DASFL, particularly under non-IID data distributions. The *Full* variant suffers from excessive communication overhead, leading to the largest per-round latency. In contrast, *Com* and *Layer-wise* reduce communication cost but fail to properly aggregate structurally heterogeneous updates, which significantly degrades accuracy under $\alpha = 0.5$ and $\alpha = 0.1$. These results show that conventional aggregation schemes are inadequate when dynamic split selection induces structural heterogeneity, highlighting the necessity of adaptive masked aggregation.

Impact of dynamic split layer selection. We next retain the adaptive masked aggregation mechanism and ablate the split layer selection strategy, denoted as DASFL-d. *Uni* enforces a unified split layer across all clients and rounds, while *Random* randomly selects a split layer at each round without considering system resources. Both variants underperform the full DASFL model. Specifically, *Uni* lacks the flexibility to accommodate heterogeneous and time-varying computation and communication resources, leading to suboptimal accuracy and latency. *Random* introduces uncoordinated split decisions, which destabilize training and yield inferior accuracy–efficiency trade-offs. In contrast, DASFL consistently achieves higher accuracy with lower per-round latency, demonstrating that resource-aware dynamic split layer selection is essential for robust and efficient SFL.

Method	Variant	Test Accuracy			T^{round}
		α=10	α=0.5	α=0.1	
DASFL-a	Full	**85.86±0.65**	82.79±0.83	65.76±1.46	2154.50
	Com	84.59±0.60	76.08±0.87	55.17±1.26	**713.10**
	Layer-wise	84.78±0.92	79.23±1.09	59.44±1.27	816.88
DASFL-d	Uni	84.38±0.97	81.20±1.18	61.98±2.13	1074.48
	Random	85.21±0.93	81.67±1.37	62.11±2.29	1090.29
DASFL	–	85.75±0.58	**82.56±1.56**	**66.96±1.64**	738.24

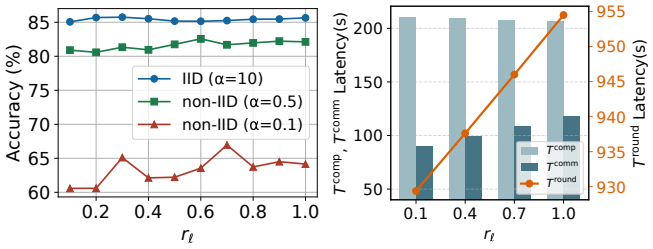
Table 3: Ablation study on ResNet18.

4.4 Hyperparameter Analysis

We analyze the sensitivity of DASFL to two key hyperparameters: the keep ratio r_ℓ used in adaptive masked aggregation and the weighting coefficients (β_1, β_2) controlling dynamic split layer selection.

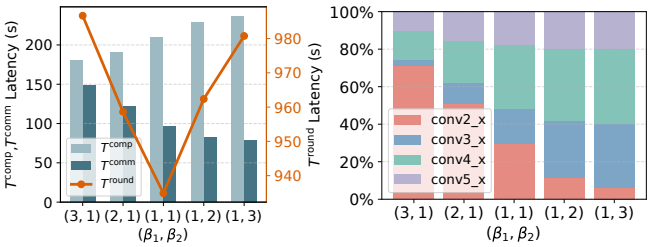
Effect of keep ratio r_ℓ. Figure 3(a) shows the final test accuracy under different r_ℓ values for IID and non-IID data distributions. Under the IID setting ($\alpha = 10$), model performance remains stable across a wide range of r_ℓ, indicating that parameter masking does not degrade accuracy when data are homogeneous. In contrast, under non-IID settings, accuracy becomes more sensitive to r_ℓ. Very small keep ratios lead to noticeable performance degradation, while moderate values consistently achieve higher accuracy. This trend is especially pronounced under extreme non-IID data ($\alpha = 0.1$), where intermediate r_ℓ yields the best performance, highlighting the importance of balancing sparsity and information retention. Figure 3(b) reports the corresponding system latency. As r_ℓ increases, communication latency grows monotonically due to the larger volume of uploaded parameters, while computation latency slightly decreases. As a result, the overall round latency increases with larger r_ℓ, revealing a clear accuracy–efficiency trade-off. These results indicate that moderate keep ratios provide a favorable balance between robustness to data heterogeneity and system efficiency, and are adopted in subsequent experiments.

Effect of weight configurations (β_1, β_2). We further study the impact of the weighting coefficients (β_1, β_2) in Eq. (6), which balance computation and communication costs during dynamic split layer selection. Five representative configurations are evaluated: $(3, 1)$, $(2, 1)$, $(1, 1)$, $(1, 2)$, and $(1, 3)$. As shown in Figure 4(a), increasing β_1 relative to β_2 biases split layer selection toward shallower client-side models, reducing computation latency but increasing communication latency due to larger smashed data transmission. Conversely, larger β_2 favors deeper split layers, lowering communication latency at the cost of higher client-side computation. Among all configurations, $(\beta_1, \beta_2) = (1, 1)$ achieves the lowest overall round latency, indicating a well-balanced trade-off. Figure 4(b) further illustrates the distribution of selected split layers under different (β_1, β_2) settings. Larger β_1 leads to more frequent selection of early layers (e.g., `conv2_x`), whereas larger β_2 shifts selections toward deeper layers (e.g., `conv4_x` and `conv5_x`). This behavior confirms that the proposed split selection mechanism effectively adapts to different resource-weighting preferences and explains the observed latency trends. Accordingly, we adopt $\beta_1 = \beta_2 = 1$ as the default configuration.



(a) (b)

Figure 3: Hyperparameter analysis of the keep ratio r_ℓ. (a) Impact of r_ℓ on final test accuracy. (b) Impact of r_ℓ on training latency.



(a) (b)

Figure 4: Effect of weight configurations (β_1, β_2). (a) Latency trade-off under different settings. (b) Normalized distribution of split layer selections under different settings.

Limitations. DASFL is currently evaluated on image classification benchmarks with moderate-scale clients under a controlled resource simulator. Its advantage may diminish when client resources are nearly homogeneous or static, where a fixed split may already suffice. Moreover, the current resource model does not fully capture continuous fluctuations in real deployments, which we leave to future work.

5 Conclusion

This paper presented Dynamic Adaptive Split Federated Learning (DASFL), a unified framework for efficient and robust split federated learning under heterogeneous, time-varying resources and non-IID data. DASFL integrates resource-aware dynamic split layer selection with adaptive masked federated aggregation, enabling clients to adapt partitioning to instantaneous conditions while maintaining stable optimization under split-induced structural heterogeneity. Extensive experiments across multiple model–dataset pairs show that DASFL consistently achieves higher accuracy and lower training latency than FL, SL, static-split SFL, and recent heterogeneity-aware baselines, especially under severe data and resource heterogeneity. Overall, effective SFL in realistic edge environments requires not only dynamic split selection but also aggregation mechanisms tailored to structural heterogeneity. DASFL provides a practical foundation for adaptive, resource-aware federated learning in edge intelligence systems.

Acknowledgements

This work was supported by the National Natural Science Foundation of China (U22B2038, 62532002, 62192784, U23A20319).

References

[Arafeh *et al.*, 2025] Mohamad Arafeh, Mohamad Wazzeh, Hani Sami, Hakima Ould-Slimane, Chamseddine Talhi, Azzam Mourad, and Hadi Otrok. Efficient privacy-preserving ml for iot: Cluster-based split federated learning scheme for non-iid data. *Journal of Network and Computer Applications*, 236:104105, 2025.

[Chopra *et al.*, 2021] Ayush Chopra, Surya Kant Sahu, Abhishek Singh, Abhinav Java, Praneeth Vepakomma, Vivek Sharma, and Ramesh Raskar. Adasplit: Adaptive trade-offs for resource-constrained distributed deep learning. *arXiv preprint arXiv:2112.01637*, 2021.

[Dachille *et al.*, 2025] Justin Dachille, Chao Huang, and Xin Liu. The impact of cut layer selection in split federated learning. *arXiv preprint arXiv:2412.15536*, 2025.

[Fan *et al.*, 2025] Wenhao Fan, Penghui Chen, Xiongfei Chun, and Yuan'an Liu. Madrl-based model partitioning, aggregation control, and resource allocation for cloud-edge-device collaborative split federated learning. *IEEE Transactions on Mobile Computing*, 24(6):5324–5341, 2025.

[Gecer and Garbinato, 2024] Melike Gecer and Benoit Garbinato. Federated learning for mobility applications. *ACM Computing Surveys*, 56(5):1–28, 2024.

[Gupta and Raskar, 2018] Otkrist Gupta and Ramesh Raskar. Distributed learning of deep neural network over multiple agents. *Journal of Network and Computer Applications*, 116:1–8, 2018.

[Han *et al.*, 2024] Pengchao Han, Chao Huang, Geng Tian, Ming Tang, and Xin Liu. Convergence analysis of split federated learning on heterogeneous data. *Advances in Neural Information Processing Systems*, 37:103476–103544, 2024.

[He *et al.*, 2016] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[Hsu *et al.*, 2019] Tzu-Ming Harry Hsu, Hang Qi, and Matthew Brown. Measuring the effects of non-identical data distribution for federated visual classification. *arXiv preprint arXiv:1909.06335*, 2019.

[Hu *et al.*, 2025] Zhanyi Hu, Tianchen Zhou, Bingzhe Wu, Cen Chen, and Yanhao Wang. A review and experimental evaluation on split learning. *Future Internet*, 17(2):87, 2025.

[Jamil *et al.*, 2022] Sonain Jamil, MuhibUr Rahman, and Fawad. A comprehensive survey of digital twins and federated learning for industrial internet of things (iiot), internet of vehicles (iov) and internet of drones (iod). *Applied System Innovation*, 5(3):56, 2022.

[Joshi *et al.*, 2022] Praveen Joshi, Chandra Thapa, Seyit Camtepe, Mohammed Hasanuzzaman, Ted Scully, and Haithem Afli. Performance and information leakage in splitfed learning and multi-head split learning in healthcare data and beyond. *Methods and Protocols*, 5(4):60, 2022.

[Kairouz *et al.*, 2021] Peter Kairouz, H. Brendan McMahan, Brendan Avent, et al. Advances and open problems in federated learning. *Foundations and Trends® in Machine Learning*, 14(1–2):1–210, 2021.

[Khan *et al.*, 2023] Latif U Khan, Mohsen Guizani, Ala Al-Fuqaha, Choong Seon Hong, Dusit Niyato, and Zhu Han. A joint communication and learning framework for hierarchical split federated learning. *IEEE Internet of Things Journal*, 11(1):268–282, 2023.

[Krizhevsky *et al.*, 2009] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009.

[LeCun *et al.*, 1998] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[Li *et al.*, 2020] Tian Li, Anit Kumar Sahu, Ameet Talwalkar, and Virginia Smith. Federated learning: Challenges, methods, and future directions. *IEEE signal processing magazine*, 37(3):50–60, 2020.

[Li *et al.*, 2021] Ang Li, Jingwei Sun, Xiao Zeng, Mi Zhang, Hai Li, and Yiran Chen. Fedmask: Joint computation and communication-efficient personalized federated learning via heterogeneous masking. In *Proceedings of the 19th ACM conference on embedded networked sensor systems*, pages 42–55, 2021.

[Lin *et al.*, 2018] Yujun Lin, Song Han, Huizi Mao, Yu Wang, and William J. Dally. Deep gradient compression: Reducing the communication bandwidth for distributed training. In *International Conference on Learning Representations (ICLR)*, 2018.

[Lin *et al.*, 2024a] Zheng Lin, Guanqiao Qu, Xianhao Chen, and Kaibin Huang. Split learning in 6g edge networks. *IEEE Wireless Communications*, 31(4):170–176, 2024.

[Lin *et al.*, 2024b] Zheng Lin, Guangyu Zhu, Yiqin Deng, Xianhao Chen, Yue Gao, Kaibin Huang, and Yuguang Fang. Efficient parallel split learning over resource-constrained wireless edge networks. *IEEE Transactions on Mobile Computing*, 23(10):9224–9239, 2024.

[Lin *et al.*, 2025a] Zheng Lin, Guanqiao Qu, Wei Wei, Xianhao Chen, and Kin K Leung. Adaptsfl: Adaptive split federated learning in resource-constrained edge networks. *IEEE Transactions on Networking*, 2025.

[Lin *et al.*, 2025b] Zheng Lin, Wei Wei, Zhe Chen, Chan-Tong Lam, Xianhao Chen, Yue Gao, and Jun Luo. Hierarchical split federated learning: Convergence analysis and system optimization. *IEEE Transactions on Mobile Computing*, 2025.

[Liu *et al.*, 2024] Hou-I Liu, Marco Galindo, Hongxia Xie, Lai-Kuan Wong, Hong-Han Shuai, Yung-Hui Li, and Wen-Huang Cheng. Lightweight deep learning for resource-constrained environments: A survey. *ACM Computing Surveys*, 56(10):1–42, 2024.

[Ma *et al.*, 2023] Jing Ma, Lv Xixiang, Yong Yu, and Stephan Sigg. Ppsfl: Privacy-preserving split federated learning via functional encryption. *Authorea Preprints*, 2023.

[McMahan *et al.*, 2017] Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Aguera y Arcas. Communication-efficient learning of deep networks from decentralized data. In *Artificial intelligence and statistics*, pages 1273–1282. PMLR, 2017.

[Oh *et al.*, 2025] Yongjeong Oh, Jaeho Lee, Christopher G Brinton, and Yo-Seb Jeon. Communication-efficient split learning via adaptive feature-wise compression. *IEEE Transactions on Neural Networks and Learning Systems*, 2025.

[Shen *et al.*, 2023] Jinglong Shen, Nan Cheng, Xiucheng Wang, Feng Lyu, Wenchao Xu, Zhi Liu, Khalid Aldubaikhy, and Xuemin Shen. Ringsfl: An adaptive split federated learning towards taming client heterogeneity. *IEEE Transactions on Mobile Computing*, 23(5):5462–5478, 2023.

[Thapa *et al.*, 2022] Chandra Thapa, Pathum Chamikara Mahawaga Arachchige, Seyit Camtepe, and Lichao Sun. Splitfed: When federated learning meets split learning. In *Proceedings of the AAAI conference on artificial intelligence*, volume 36, pages 8485–8493, 2022.

[Uzzaman, 2025] Arfan Uzzaman. Federated learning–driven real-time disease surveillance for smart hospitals using multi-source heterogeneous healthcare data. *ASRC Procedia: Global Perspectives in Science and Scholarship*, 1(01):1390–1423, 2025.

[Vepakomma *et al.*, 2018] Praneeth Vepakomma, Otkrist Gupta, Tristan Swedish, and Ramesh Raskar. Split learning for health: Distributed deep learning without sharing raw patient data. *arXiv preprint arXiv:1812.00564*, 2018.

[Wu *et al.*, 2023] Wen Wu, Mushu Li, Kaige Qu, Conghao Zhou, Xuemin Shen, Weihua Zhuang, Xu Li, and Weisen Shi. Split learning over wireless networks: Parallel design and resource management. *IEEE Journal on Selected Areas in Communications*, 41(4):1051–1066, 2023.

[Xiao *et al.*, 2017] Han Xiao, Kashif Rasul, and Roland Vollgraf. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. *arXiv preprint arXiv:1708.07747*, 2017.

[Zhang *et al.*, 2023] Zongshun Zhang, Andrea Pinto, Valeria Turina, Flavio Esposito, and Ibrahim Matta. Privacy and efficiency of communications in federated split learning. *IEEE Transactions on Big Data*, 9(5):1380–1391, 2023.

[Zhang *et al.*, 2024] Mingjin Zhang, Jiannong Cao, Yuvraj Sahni, Xiangchun Chen, and Shan Jiang. Resource-efficient parallel split learning in heterogeneous edge computing. In *2024 International Conference on Computing, Networking and Communications (ICNC)*, pages 794–798. IEEE Computer Society, 2024.

[Zhu *et al.*, 2021] Hangyu Zhu, Jinjin Xu, Shiqing Liu, and Yaochu Jin. Federated learning on non-iid data: A survey. *Neurocomputing*, 465:371–390, 2021.

[Zhu *et al.*, 2024] Guangyu Zhu, Yiqin Deng, Xianhao Chen, Haixia Zhang, Yuguang Fang, and Tan F Wong. Esfl: Efficient split federated learning over resource-constrained heterogeneous wireless devices. *IEEE Internet of Things Journal*, 11(16):27153–27166, 2024.