

SplitLite: Low-Rank Residual Compression for Split Learning

Tao Li, Yulin Tang, Qi Guo, and Xianhao Chen, *Member, IEEE*

Abstract—Federated fine-tuning of on-device large language models (LLMs) faces a significant computing burden. To overcome this limitation, split learning (SL) has emerged as a promising solution, which offloads the primary training workload to a powerful server. However, SL requires exchanging high-dimensional activations and gradients between clients and the server, resulting in prohibitive communication costs. To overcome this challenge, we propose SplitLite, a communication-efficient split federated LoRA fine-tuning method that exploits the low effective rank structure of consecutive-epoch activation and gradient residuals. Our key finding is that, when LoRA uses rank r updates in parameter space, the activation and gradient residuals of the same data sample between adjacent epochs also exhibit effective rank- $2r$ and rank- $4r$ structures, respectively. By revealing this property, SplitLite transmits only quantized truncated singular value decomposition (SVD) residual factors, thereby significantly reducing both activation uplink and gradient downlink traffic. Extensive experiments on the GLUE benchmark across a series of advanced on-device LLMs demonstrate that our method reduces activation uplink communication costs by up to 93.5% and total communication costs by up to 83.7%, without performance degradation.

Index Terms—Federated Learning, Split Federated Learning, Large Language Models, Parameter-Efficient Fine-Tuning.

I. INTRODUCTION

Various downstream applications of large language models (LLMs) [1], such as personal assistants, healthcare, and home robotics, rely on data distributed across organizations or individual users, which cannot be centralized due to privacy and regulatory constraints [2], [3]. Federated learning (FL) offers a privacy-preserving paradigm that keeps raw data local and shares only model updates [4]–[6]. However, conventional FL requires each participating device to train the entire model locally, which is impractical for LLMs given the limited memory and compute budgets of edge devices [7], [8]. Even with parameter-efficient fine-tuning (PEFT) methods such as low-rank adaptation (LoRA) [9], which trains lightweight low-rank adapters only while keeping the pre-trained backbone frozen, this challenge still persists. For example, fine-tuning Llama-3.2-3B with LoRA rank 24 and batch size 8 consumes over 16 GB of GPU memory, which still exceeds the capacity of many edge devices. Against this backdrop, split learning (SL) [10] has emerged as a promising alternative by partitioning the

Tao Li, Qi Guo, and Xianhao Chen are with the Department of Electrical and Computer Engineering, The University of Hong Kong, Hong Kong SAR, China (e-mail: litao@eee.hku.hk; guoqi772100@connect.hku.hk; xchen@eee.hku.hk).

Yulin Tang is with the Department of Electrical Engineering, California Institute of Technology, Pasadena, CA, USA (e-mail: ytang9@caltech.edu). (Corresponding author: Xianhao Chen.)

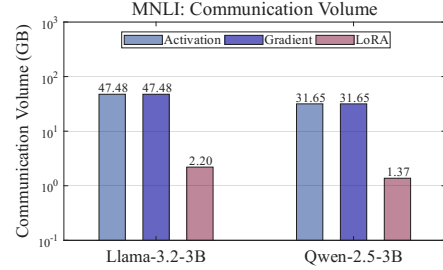


Fig. 1. Total communication overhead of LoRA aggregation, activation uploading, and gradient downloading until model convergence for Llama-3.2-3B and Qwen-2.5-3B under split federated LoRA fine-tuning on the MNLI dataset with 6 clients over 10 epochs.

model into client-side and server-side sub-models, thereby offloading the primary training workload to a powerful server. This makes the integration of LoRA and SL a natural paradigm for resource-efficient LLM fine-tuning on edge devices [11].

However, applying SL to fine-tune LLMs still faces a major bottleneck, i.e., high-dimensional activation and gradient communications at the model cut layer [12], [13]. Specifically, the frequent exchanges of high-dimensional activations and gradients at the cut layer lead to significant communication overhead (shown in Fig. 1). While forward and backward passes both involve communication costs at the cut layer between clients and the server, uplink communication puts an even heavier burden on wireless/mobile devices due to asymmetric data rates¹ and limited on-device energy. An intuitive approach to mitigate this communication overhead is activation quantization. Nevertheless, since Transformer architectures can generate activations with an extremely wide value range [16]–[18], naively quantizing activations can catastrophically degrade the fine-tuning performance for LLMs [12], [19]. As a result, prior activation-quantization methods designed for traditional SL [20] may not work effectively for LLMs. Thus, a fundamental research question remains open in split learning: *Is it possible to significantly compress activations and gradients at the cut layer of LLMs without degrading the fine-tuning performance?*

At first glance, the question seems to be paradoxical, as there is generally no free lunch. However, focusing on LoRA fine-tuning, we can leverage *two key insights* to answer the question affirmatively. First, given the same data sample, the

¹Since wireless devices are less powerful than base stations, uplink data rate is often much lower than the downlink data rate in wireless networks [14], e.g., 30.6 Mbps and 166.8 Mbps for average 5G uplink and downlink, respectively [15].

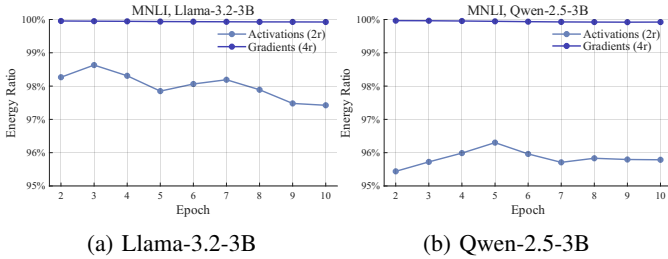


Fig. 2. Residual energy ratios of LoRA-induced low-rank temporal residuals on MNLI for (a) Llama-3.2-3B and (b) Qwen-2.5-3B. Rank- $2r$ activation and rank- $4r$ gradient residuals retain most of the adjacent-epoch residual energy, confirming their low effective ranks.

activation/gradient residuals at the cut layer across consecutive epochs are very small during fine-tuning, because they are solely induced by subtle parameter changes. Second, LoRA fine-tuning of the parameter space also leads to *low-rank* structure for such residuals at the cut layer. Given that LoRA with rank r is adopted and parameter change is sufficiently small, we show that the effective rank of the activation residual between adjacent training epochs is upper-bounded by $2r$, while that of the gradient residual is upper-bounded by $4r$. This phenomenon is also empirically validated in Fig. 2, where the majority of energy in activation/gradient residuals can be captured by low-rank approximation.

Motivated by these insights, we propose SplitLite, a communication-efficient split federated LoRA fine-tuning framework for LLMs that exploits the LoRA-induced low-rank property of activation and gradient residuals across consecutive training epochs. Specifically, we exploit the LoRA-induced low-rank structure of temporal residuals and transmit only rank- $2r$ activation residuals and rank- $4r$ gradient residuals. To achieve this, we encode the activation and gradient residuals using rank- $2r$ and rank- $4r$ truncated singular value decomposition (SVD), respectively. We further compress the resulting factors using an unbiased stochastic uniform quantizer, thereby reducing both the number of transmitted elements and the number of bits per element. Our key contributions are summarized as follows.

- We uncover the LoRA-induced low-effective-rank structure of temporal residuals at the cut layer. Specifically, for parameter updates with LoRA rank r , the effective ranks of the activation and gradient residuals between adjacent epochs are upper-bounded by $2r$ and $4r$, respectively.
- We leverage quantized rank- $2r$ activation and rank- $4r$ gradient residual factors to substantially reduce communication overhead.
- We establish a theoretical convergence analysis that explicitly captures the impact of SVD truncation and stochastic quantization on model convergence.
- Extensive experiments on the GLUE benchmark with on-device LLMs demonstrate that our method reduces activation uplink communication by up to **93.5%** and total communication cost by up to **83.7%**, without sacrificing model performance.

II. RELATED WORK

Split fine-tuning on resource-constrained devices. Fine-tuning billion-parameter LLMs through conventional FL remains impractical for resource-limited edge devices [7], [21]. SL mitigates this burden by partitioning the model between clients and a server [10], [22]. Among SL approaches, SFL is a predominant framework that integrates SL with FL. Lin et al. [23] studied the convergence of SFL and optimized cut-layer placement for heterogeneous, resource-constrained clients. Recent work further combines SFL with PEFT to support on-device LLM adaptation [11], [24], [25]. For example, SplitLoRA [11] partitions the model into client-side and server-side sub-models and updates only LoRA adapters to alleviate client-side computation overhead. However, these methods overlook SFL’s key bottleneck for LLMs: excessive communication overhead caused by repeated activation uploads and gradient downloads. This limitation has motivated communication compression techniques for intermediate activations and gradients in SL.

Communication compression in split learning. Prior SL studies reduce cut-layer traffic by compressing intermediate tensors through quantization [26] or sparsification [27]. FedLite [28] clusters and vector-quantizes cut-layer activations and introduces gradient correction to mitigate compression errors. Rand-Top- k [29] sparsifies cut-layer representations through randomized top- k selection, improving convergence and generalization over deterministic top- k [30]. However, these methods process each training round independently and do not exploit the cross-epoch redundancy arising in parameter-efficient LLM fine-tuning, where the activations and gradients tend to change gradually as only lightweight adapters are updated. AQ-SGD [19] partially exploits this redundancy by quantizing inter-epoch activation changes, but directly quantizes the backward gradients without leveraging their temporal structure. SplitCom [12] instead reduces transmission frequency by skipping sufficiently similar activations and gradients. Distinct from these approaches, SplitLite reduces the traffic of every bidirectional exchange by compressing rank- $2r$ activation and rank- $4r$ gradient residuals via truncated SVD [31] and stochastic uniform quantization [26].

III. SYSTEM MODEL

As shown in Fig. 3, our method is based on a standard SFL framework [10]. In SFL, clients and the central server exchange intermediate activations and gradients at the cut layer during each training step. Formally, let $W = [W_{c,i}; W_{s,i}]$ denote the weights of a pre-trained LLM, partitioned into a client-side sub-model $W_{c,i}$ and a server-side sub-model $W_{s,i}$. The set of participating edge devices is denoted by $\mathcal{N} = \{1, 2, \dots, N\}$, where N is the number of edge devices. Each edge device i has its local dataset $\mathcal{D}_i = \{x_{i,k}, y_{i,k}\}_{k=1}^{n_i}$ with n_i data samples, where $x_{i,k}$ and $y_{i,k}$ are the k -th input data and its corresponding label.

We introduce trainable LoRA adapters to fine-tune both parts of W : $\mathcal{R}_{c,i}$ for the client-side sub-model and $\mathcal{R}_{s,i}$ for the server-side sub-model of each client i . During training,

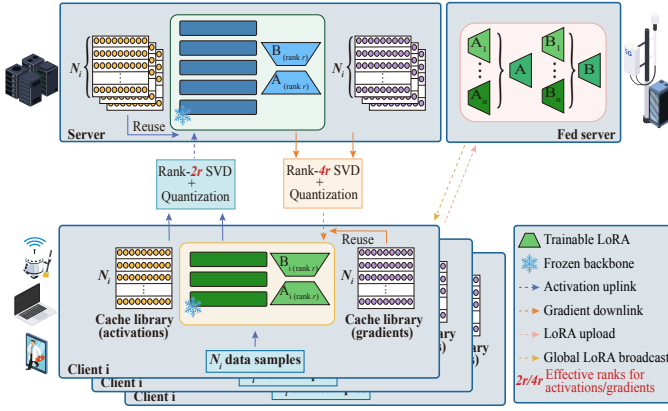


Fig. 3. System overview of SplitLite. Clients and the server maintain synchronized activation and gradient caches, and communicate only adjacent-round quantized rank- $2r$ activation residuals and rank- $4r$ gradient residuals, while periodically aggregating the trainable LoRA adapters.

only these lightweight LoRA adapters are updated while W remains frozen, which substantially reduces the number of trainable parameters. At every training step, the central server locally aggregates the server-side LoRA adapters $\{\mathcal{R}_{s,i}\}_{i=1}^N$, and every I steps, the clients upload their client-side LoRA adapters $\{\mathcal{R}_{c,i}\}_{i=1}^N$ to the federated server for aggregation², after which the averaged client-side adapter is broadcast back to the clients to continue training.

The global LoRA adapter is denoted by $\mathcal{R} = [\mathcal{R}_c; \mathcal{R}_s]$, where $\mathcal{R}_c$ and $\mathcal{R}_s$ are the aggregated client-side and server-side LoRA adapters across all clients, respectively. The objective of split federated LoRA fine-tuning is to find the adapter configuration $\mathcal{R}^*$ that minimizes the following non-convex finite-sum global loss over all participating devices:

$$\min_{\mathcal{R}} \frac{1}{N} \sum_{i=1}^N f_i(W | \mathcal{R}), \quad (1)$$

where $f_i(W | \mathcal{R})$ is the local loss function evaluated on the data of client i using the frozen global backbone W together with the global LoRA adapter $\mathcal{R}$, i.e., $f_i(W | \mathcal{R}) \triangleq \mathbb{E}_{\xi_i \sim \mathcal{D}_i} [F_i(W | \mathcal{R}; \xi_i)]$, where ξ_i captures the training randomness induced by mini-batch sampling from $\mathcal{D}_i$. Following the standard stochastic optimization setting [23], [32], the stochastic gradient is assumed to be an unbiased estimator of the true gradient, i.e., $\mathbb{E}_{\xi_i^t \sim \mathcal{D}_i} [\nabla_{\mathcal{R}} F_i(W | \mathcal{R}_i^t; \xi_i^t) | \xi^{[t-1]}] = \nabla_{\mathcal{R}} f_i(W | \mathcal{R}_i^t)$, where $\xi^{[t-1]} \triangleq [\xi_i^{\tau}]_{i \in \{1, \dots, N\}, \tau \in \{0, \dots, t-1\}}$ denotes all mini-batch sampling randomness up to training step $t-1$.

To solve problem (1), conventional SFL frameworks for LLM fine-tuning require transmitting high-dimensional cut-layer activations and gradients at every iteration, leading to substantial bidirectional communication overhead. To address this bottleneck, we propose SplitLite, which exploits the low-rank structure of activation and gradient residuals for efficient communication.

²We assume homogeneous cut layers across all clients. Our method can be extended to heterogeneous scenarios with different cut layers.

IV. DESIGN OF SPLITLITE

To address the communication bottleneck in SFL, this section presents our low-rank residual compression framework named SplitLite. We first provide an overview of our method in Sec. IV-A. We then analyze the LoRA-induced low effective rank structure of the adjacent-epoch activation and gradient residuals in Sec. IV-B, which motivates the residual compression scheme detailed in Sec. IV-C. Finally, the system implementation and optimizations are described in Sec. IV-D.

A. Overview

Fig. 3 provides an overview of our SplitLite framework, a compression scheme for activations/gradients at the cut layer in SFL. Its design is based on a structural property of LoRA fine-tuning: given the same data sample, since the adjacent-epoch change in intermediate representations is driven solely by the rank- r adapter updates, we can show that activation and gradient residuals of each sample have low effective ranks of $2r$ and $4r$, respectively. This allows the compression rank of activations/gradients to be set to $2r$ and $4r$ without compromising learning performance.

The communication procedure at the cut layer operates as follows. Each client and the server maintain synchronized caches of the cut-layer activations and gradients, initialized by exchanging the full tensors in the first epoch. In subsequent epochs, each side transmits only the residual against its latest cached versions. Prior to transmissions, the activation and gradient residuals undergo a truncated SVD to rank- $2r$ and rank- $4r$, respectively. This decomposition is then factored into two low-rank matrices by absorbing the singular values into the left factor. To further improve the compression ratio, these low-rank matrices are compressed by an unbiased stochastic quantizer. Finally, the receiver reconstructs the residuals from the quantized factors and updates its local cache, repeating this process until the model converges.

B. Insights: Low Effective Rank Residuals

We next show the rationale behind our design: if the model is fine-tuned via LoRA adapters, the residual of activations and gradients across epochs also exhibits a low-rank structure. Consider a fixed sample $x_{i,b}$ of client i , and let $Y_{i,b}^{e,\ell} \in \mathbb{R}^{S \times E}$ denote its activations at cut layer ℓ in epoch e . Similarly, let $G_{i,b}^{e,\ell} \in \mathbb{R}^{S \times E}$ denote the gradient of the loss with respect to the cut-layer activation. We define the adjacent-epoch activation and gradient residuals as $\Delta Y_{i,b}^{e,\ell} \triangleq Y_{i,b}^{e,\ell} - Y_{i,b}^{e-1,\ell}$, and $\Delta G_{i,b}^{e,\ell} \triangleq G_{i,b}^{e,\ell} - G_{i,b}^{e-1,\ell}$, respectively. Clearly, these residuals are induced entirely by updates to the client-side and server-side LoRA adapters.

Activation residual. To characterize the activation residual, let $\Phi_c^\ell(x; \mathcal{R}_c)$ denote the client-side activation mapping from input x to cut layer ℓ , implemented by the frozen client-side backbone together with its LoRA adapters. Accordingly, $Y_{i,b}^{e,\ell} = \Phi_c^\ell(x_{i,b}; \mathcal{R}_{c,i}^e)$. Assuming that the adjacent-epoch LoRA parameter variation $\Delta \mathcal{R}_{c,i}^e$ is sufficiently small and that Φ_c^ℓ is Fréchet differentiable with respect to $\mathcal{R}_c$, the

adjacent-epoch change in Φ_c^ℓ admits the following first-order approximation:

$$\Delta Y_{i,b}^{e,\ell} \approx D_{\mathcal{R}_c} \Phi_c^\ell(x_{i,b}; \mathcal{R}_{c,i}^{e-1}) [\Delta \mathcal{R}_{c,i}^e], \quad (2)$$

where $D_{\mathcal{R}_c} \Phi_c^\ell$ denotes the Fréchet derivative of Φ_c^ℓ with respect to $\mathcal{R}_c$, evaluated at $\mathcal{R}_{c,i}^{e-1}$, and $\Delta \mathcal{R}_{c,i}^e \triangleq \mathcal{R}_{c,i}^e - \mathcal{R}_{c,i}^{e-1}$ denotes the adjacent-epoch variation of the client-side LoRA parameters. Hence, the activation change is primarily characterized by the local linear response of the client-side activation mapping to the LoRA parameter update.

For a LoRA-augmented projection, the effective LoRA weight update at epoch e is given by $\mathcal{R}_{c,i}^e = B_{c,i}^e A_{c,i}^e$, where r denotes the LoRA rank. Its variation between two consecutive epochs can be expressed as

$$\mathcal{R}_{c,i}^e - \mathcal{R}_{c,i}^{e-1} = B_{c,i}^e A_{c,i}^e - B_{c,i}^{e-1} A_{c,i}^{e-1}. \quad (3)$$

Since each matrix product on the right-hand side has rank at most r , the subadditivity of matrix rank yields $\text{rank}(\mathcal{R}_{c,i}^e - \mathcal{R}_{c,i}^{e-1}) \leq 2r$. Through the first-order linearization of the client-side mapping, this low-rank parameter variation is propagated to the activation residual. Consequently, the dominant component of $\Delta Y_{i,b}^{e,\ell}$ inherits the low-rank structure of $\Delta \mathcal{R}_{c,i}^e$, yielding

$$\text{rank}(\Delta Y_{i,b}^{e,\ell}) \leq 2r. \quad (4)$$

Therefore, although nonlinear Transformer operations may introduce additional higher-order components, the dominant activation residual is driven by LoRA-constrained update directions and is thus expected to exhibit an effective rank of approximately $2r$.

Gradient residual. To characterize the gradient residual, define the server-side backward mapping as $\mathcal{B}^\ell(Y, \mathcal{R}_s) \triangleq \nabla_Y F_s(Y; W_s, \mathcal{R}_s)$. Accordingly, the cut-layer gradient can be expressed as $G_{i,b}^{e,\ell} = \mathcal{B}^\ell(Y_{i,b}^{e,\ell}, \mathcal{R}_{s,i}^e)$. A first-order Fréchet expansion around $(Y_{i,b}^{e-1,\ell}, \mathcal{R}_{s,i}^{e-1})$ gives

$$\Delta G_{i,b}^{e,\ell} \approx D_Y \mathcal{B}_{e-1}^\ell [\Delta Y_{i,b}^{e,\ell}] + D_{\mathcal{R}_s} \mathcal{B}_{e-1}^\ell [\Delta \mathcal{R}_{s,i}^e], \quad (5)$$

where $D_Y \mathcal{B}_{e-1}^\ell$ and $D_{\mathcal{R}_s} \mathcal{B}_{e-1}^\ell$ denote the Fréchet derivatives of $\mathcal{B}^\ell$ with respect to Y and $\mathcal{R}_s$, respectively, evaluated at $(Y_{i,b}^{e-1,\ell}, \mathcal{R}_{s,i}^{e-1})$, and $\Delta \mathcal{R}_{s,i}^e \triangleq \mathcal{R}_{s,i}^e - \mathcal{R}_{s,i}^{e-1}$ denotes the adjacent-epoch variation of the server-side LoRA parameters.

Thus, the gradient residual is jointly induced by the client-side activation shift and the server-side LoRA update. Since both sources are driven by LoRA weight variations with effective ranks of approximately $2r$, the dominant gradient variation is expected to concentrate in a subspace of dimension approximately $4r$, yielding

$$\text{rank}(\Delta G_{i,b}^{e,\ell}) \leq 4r. \quad (6)$$

We emphasize that the above low-rank characterization relies on first-order linearization and should be interpreted as an effective, rather than exact algebraic, rank property for the nonlinear Transformer. Specifically, for a matrix Z

Algorithm 1 SplitLite Algorithm

Require: Client set $\mathcal{N}$, local datasets $\{\mathcal{D}_i\}_{i=1}^N$, total iterations T , LoRA rank r , client-side aggregation interval I , bitwidths q_y, q_g , and learning rate η

- 1: Initialize the frozen backbone $W = [W_{c,i}; W_{s,i}]$, client-side adapters $\{\mathcal{R}_{c,i}^0\}_{i=1}^N$, server-side adapters $\{\mathcal{R}_{s,i}^0\}_{i=1}^N$, and empty activation and gradient caches
- 2: **for** $t = 0, 1, \dots, T - 1$ **do**
- 3: **for** $i \in \mathcal{N}$ **in parallel do**
- 4: Sample a mini-batch $\mathcal{B}_i^t \subseteq \mathcal{D}_i$
- 5: **for** $b \in \mathcal{B}_i^t$ **do**
- 6: Compute $Y_{i,b}^{t,\ell} = \Phi_c^\ell(x_{i,b}; \mathcal{R}_{c,i}^t)$
- 7: $\tilde{Y}_{i,b}^{t,\ell} \leftarrow$ **Algorithm 2**($Y_{i,b}^{t,\ell}, y, r, q_y$)
- 8: Perform server-side forward and backward propagation using $\tilde{Y}_{i,b}^{t,\ell}$ to obtain $G_{i,b}^{t,\ell}$
- 9: $\tilde{G}_{i,b}^{t,\ell} \leftarrow$ **Algorithm 2**($G_{i,b}^{t,\ell}, g, r, q_g$)
- 10: Perform client-side backpropagation using $\tilde{G}_{i,b}^{t,\ell}$
- 11: **end for**
- 12: Form the mini-batch gradients $\tilde{g}_{c,i}^t$ and $\tilde{g}_{s,i}^t$
- 13: Update both LoRA adapters:

$$\mathcal{R}_{d,i}^{t+1} \leftarrow \mathcal{R}_{d,i}^t - \eta \tilde{g}_{d,i}^t, \quad d \in \{c, s\}$$

- 14: **end for**
- 15: **Server-side local aggregation:**

$$\mathcal{R}_s^{t+1} \leftarrow \frac{1}{N} \sum_{i=1}^N \mathcal{R}_{s,i}^{t+1}, \quad \mathcal{R}_{s,i}^{t+1} \leftarrow \mathcal{R}_s^{t+1}, \quad \forall i$$

- 16: **if** $(t + 1) \bmod I = 0$ **then**
- 17: Clients upload $\{\mathcal{R}_{c,i}^{t+1}\}_{i=1}^N$ to the federated server
- 18: **Federated aggregation and broadcast:**

$$\mathcal{R}_c^{t+1} \leftarrow \frac{1}{N} \sum_{i=1}^N \mathcal{R}_{c,i}^{t+1}, \quad \mathcal{R}_{c,i}^{t+1} \leftarrow \mathcal{R}_c^{t+1}, \quad \forall i$$

- 19: **end if**
 - 20: **end for**
 - 21: **return** $\mathcal{R}^T = [\mathcal{R}_c^T; \mathcal{R}_s^T]$
-

with singular values $s_j(Z)$, define its top- k energy ratio as $\Gamma_k(Z) \triangleq \sum_{j=1}^k s_j^2(Z) / \sum_j s_j^2(Z)$ and its effective rank under an energy threshold θ as $r_{\text{eff}}(Z; \theta) \triangleq \min k : \Gamma_k(Z) \geq \theta$. As shown in Fig. 2, both $\Gamma_{2r}(\Delta Y_{i,b}^{e,\ell})$ and $\Gamma_{4r}(\Delta G_{i,b}^{e,\ell})$ remain close to 1, indicating that the activation and gradient residuals between adjacent epochs have effective ranks of $2r$ and $4r$, respectively. This motivates their rank- $2r$ and rank- $4r$ truncated SVD compression.

C. Algorithm: Low-Rank Bidirectional Residual Compression

We now present the proposed low-rank bidirectional residual compression mechanism for reducing both activation uplink and gradient downlink communication at the cut layer. The key idea is to maintain synchronized sample-wise caches at the sender and receiver and transmit only the low-rank temporal

Algorithm 2 Low-Rank Residual Compression ($Z_{i,b}^{t,\ell}$, d , r , q_d)

Require: Tensor $Z_{i,b}^{t,\ell}$, direction $d \in \{y, g\}$, LoRA rank r , bitwidth q_d

Ensure: Reconstructed tensor $\tilde{Z}_{i,b}^{t,\ell}$

- 1: Set $\kappa_y = 2r$ and $\kappa_g = 4r$
- 2: **if** the synchronized sender–receiver cache is empty **then**
- 3: Transmit $Z_{i,b}^{t,\ell}$ in full and set $\tilde{Z}_{i,b}^{t,\ell} \leftarrow Z_{i,b}^{t,\ell}$
- 4: **else**
- 5: $\Delta\tilde{Z}_{i,b}^{t,\ell} \leftarrow Z_{i,b}^{t,\ell} - \tilde{Z}_{i,b}^{\pi_{i,b}(t),\ell}$
- 6: $U\Sigma V^\top \leftarrow \text{SVD}_{\kappa_d}(\Delta\tilde{Z}_{i,b}^{t,\ell})$
- 7: Independently quantize and transmit

$$\hat{P} \leftarrow \mathcal{Q}_{q_d}(U\Sigma), \quad \hat{Q} \leftarrow \mathcal{Q}_{q_d}(V^\top)$$

- 8: Identically update the sender and receiver caches:

$$\tilde{Z}_{i,b}^{t,\ell} \leftarrow \tilde{Z}_{i,b}^{\pi_{i,b}(t),\ell} + \hat{P}\hat{Q}$$

9: **end if**

10: **return** $\tilde{Z}_{i,b}^{t,\ell}$

residual relative to the most recently reconstructed tensor of the same sample.

Low-rank temporal residual compression. For each sample b , the client and the server maintain synchronized reconstructed activation and gradient caches. When sample b is processed again, the activation and gradient residuals relative to their most recently reconstructed caches are computed as $\Delta\tilde{Y}_{i,b}^{t,\ell} = Y_{i,b}^{t,\ell} - \tilde{Y}_{i,b}^{\pi_{i,b}(t),\ell}$, $\Delta\tilde{G}_{i,b}^{t,\ell} = G_{i,b}^{t,\ell} - \tilde{G}_{i,b}^{\pi_{i,b}(t),\ell}$, where $\pi_{i,b}(t)$ denotes the most recent iteration before t at which sample b was processed.

Motivated by the low effective rank of the activation and gradient residuals revealed in Sec. IV-B, we approximate $\Delta\tilde{Y}_{i,b}^{t,\ell}$ and $\Delta\tilde{G}_{i,b}^{t,\ell}$ using rank- $2r$ and rank- $4r$ truncated SVD, respectively:

$$\Delta\tilde{Y}_{i,b}^{t,\ell} \approx U_{y,i,b}^{t,\ell} \Sigma_{y,i,b}^{t,\ell} \left(V_{y,i,b}^{t,\ell}\right)^\top, \quad (7)$$

$$\Delta\tilde{G}_{i,b}^{t,\ell} \approx U_{g,i,b}^{t,\ell} \Sigma_{g,i,b}^{t,\ell} \left(V_{g,i,b}^{t,\ell}\right)^\top. \quad (8)$$

Here, $U_{y,i,b}^{t,\ell} \in \mathbb{R}^{S \times 2r}$, $\Sigma_{y,i,b}^{t,\ell} \in \mathbb{R}^{2r \times 2r}$, and $\left(V_{y,i,b}^{t,\ell}\right)^\top \in \mathbb{R}^{2r \times E}$, whereas $U_{g,i,b}^{t,\ell} \in \mathbb{R}^{S \times 4r}$, $\Sigma_{g,i,b}^{t,\ell} \in \mathbb{R}^{4r \times 4r}$, and $\left(V_{g,i,b}^{t,\ell}\right)^\top \in \mathbb{R}^{4r \times E}$.

To further reduce the communication payload, we absorb the singular-value matrix into the corresponding left singular-vector matrix, thereby avoiding its separate transmission:

$$\begin{aligned} P_{y,i,b}^{t,\ell} &\triangleq U_{y,i,b}^{t,\ell} \Sigma_{y,i,b}^{t,\ell}, & Q_{y,i,b}^{t,\ell} &\triangleq \left(V_{y,i,b}^{t,\ell}\right)^\top, \\ P_{g,i,b}^{t,\ell} &\triangleq U_{g,i,b}^{t,\ell} \Sigma_{g,i,b}^{t,\ell}, & Q_{g,i,b}^{t,\ell} &\triangleq \left(V_{g,i,b}^{t,\ell}\right)^\top. \end{aligned} \quad (9)$$

Thus, $\Delta\tilde{Y}_{i,b}^{t,\ell} \approx P_{y,i,b}^{t,\ell} Q_{y,i,b}^{t,\ell}$ and $\Delta\tilde{G}_{i,b}^{t,\ell} \approx P_{g,i,b}^{t,\ell} Q_{g,i,b}^{t,\ell}$.

Overall, the proposed low-rank factorization reduces the per-sample transmission from $2SE$ to $6r(S+E)$ elements, corresponding to a relative communication cost of $3r\left(\frac{1}{S} + \frac{1}{E}\right)$.

With $S = 128$, $E = 3072$, and $r = 8$, it requires only approximately 19.5% of the uncompressed bidirectional communication. The gain becomes greater for longer sequences and larger hidden dimensions. We further quantize the low-rank residual factors using unbiased stochastic uniform quantization to reduce the per-element bit width.

Stochastic uniform quantization. Low-rank factorization compresses the transmitted tensor along the element-count dimension but leaves the precision of each retained element untouched. We therefore apply stochastic uniform quantization to low-rank factors, an orthogonal mechanism that lowers the per-element bit width without degrading model performance. Unlike deterministic rounding, which pushes small factor entries consistently in the same direction and lets systematic errors accumulate over iterations, the stochastic quantizer [26] is unbiased, i.e., $\mathbb{E}_{\mathcal{Q}}[\mathcal{Q}(x)] = x$ for any factor matrix x .

Accordingly, the sender quantizes the low-rank factors as

$$\hat{P}_{y,i,b}^{t,\ell} = \mathcal{Q}\left(P_{y,i,b}^{t,\ell}\right), \quad \hat{Q}_{y,i,b}^{t,\ell} = \mathcal{Q}\left(Q_{y,i,b}^{t,\ell}\right), \quad (10)$$

$$\hat{P}_{g,i,b}^{t,\ell} = \mathcal{Q}\left(P_{g,i,b}^{t,\ell}\right), \quad \hat{Q}_{g,i,b}^{t,\ell} = \mathcal{Q}\left(Q_{g,i,b}^{t,\ell}\right), \quad (11)$$

and transmits only the quantized factors together with the corresponding quantization parameters. Upon reception, the activation and gradient residuals are reconstructed as

$$\Delta\hat{Y}_{i,b}^{t,\ell} = \hat{P}_{y,i,b}^{t,\ell} \hat{Q}_{y,i,b}^{t,\ell}, \quad (12)$$

$$\Delta\hat{G}_{i,b}^{t,\ell} = \hat{P}_{g,i,b}^{t,\ell} \hat{Q}_{g,i,b}^{t,\ell}. \quad (13)$$

These reconstructed residuals are then added to the corresponding cached values to recover the activations and gradients used for forward and backward propagation:

$$\tilde{Y}_{i,b}^{t,\ell} = \tilde{Y}_{i,b}^{\pi_{i,b}(t),\ell} + \Delta\hat{Y}_{i,b}^{t,\ell}, \quad (14)$$

$$\tilde{G}_{i,b}^{t,\ell} = \tilde{G}_{i,b}^{\pi_{i,b}(t),\ell} + \Delta\hat{G}_{i,b}^{t,\ell}. \quad (15)$$

The recovered activations and gradients then update the cache entries for sample b at both the sender and the receiver, keeping the two caches synchronized for the next residual computation. In this manner, low-rank factorization reduces the number of communicated elements and stochastic quantization reduces the bits per element, yielding a multiplicative reduction in communication costs while preserving model performance.

Communication analysis. Instead of transmitting the full $S \times E$ tensor, our method communicates only $\kappa_d(S + E)$ elements, where the bottleneck dimension κ_d is determined by the LoRA rank r . For each communication direction, this low-rank factorization reduces the communication complexity from $O(SE)$ to $O(\kappa_d(S + E))$. Moreover, since the relative communication cost scales as $\kappa_d\left(\frac{1}{S} + \frac{1}{E}\right)$, the resulting communication gain becomes more pronounced as the sequence length or hidden dimension increases, making our method particularly attractive for long-context fine-tuning [33] and large models. Furthermore, the unbiased stochastic quantizer decreases the bitwidth of each transmitted element. Consequently, our framework jointly reduces both the number of communicated elements and the bits per element, substantially

lowering bidirectional communication overhead without compromising fine-tuning performance.

The SplitLite training procedure is outlined in Algorithms 1 and 2

D. Implementation Considerations

Computation costs. A potential concern is the SVD computation cost. While a full SVD of an $S \times E$ residual costs $O(SE \min(S, E))$, our scheme requires only the leading $\kappa_d \in \{2r, 4r\}$ singular triplets and thus costs $O(SE\kappa_d)$ using truncated randomized SVD [31]. Moreover, SVD and quantization are independent across samples, enabling batched and pipelined execution that overlaps with communication. As shown in Table II, despite the additional compression computation, SplitLite consistently achieves the lowest end-to-end fine-tuning latency among all evaluated methods across three GLUE tasks, reducing the latency of Llama-3.2-3B by 65.5%–73.6% relative to SplitLoRA.

Storage costs. The second implementation consideration is storage cost. SplitLite maintains per-sample caches of the reconstructed activation $\tilde{Y}_{i,b}^{t,\ell}$ and gradient $\tilde{G}_{i,b}^{t,\ell}$ on both sides of the cut layer. Since the caches are overwritten in place, they require at most $2n_i SE$ entries on client i . For Llama-3.2-3B fine-tuning, this amounts to about 1 GB per client on MRPC and 100 GB on MNLI due to its larger local dataset. As only the current mini-batch $\mathcal{B}_i^t$ is accessed per iteration, the caches can be offloaded to CPU memory or SSD [34], stored in low precision with synchronized rounding [12]. Following activation-caching systems [19], cache loading and updates can be overlapped with forward and backward computation or scheduled during training idle periods.

V. CONVERGENCE ANALYSIS

We establish the convergence guarantee under residual compression, rank- $2r/4r$ truncated SVD, and stochastic uniform quantization. Recall $\mathcal{R}_i^t = [\mathcal{R}_{c,i}^t; \mathcal{R}_{s,i}^t]$, $\mathcal{R}^t = \frac{1}{N} \sum_i \mathcal{R}_i^t$, and the applied gradient $\tilde{g}_i^t = g_i^t + \delta_i^t$ with perturbation δ_i^t , so the averaged update is $\mathcal{R}^{t+1} = \mathcal{R}^t - \eta(\bar{g}^t + \bar{\delta}^t)$, where $\bar{g}^t = \frac{1}{N} \sum_i g_i^t$ and $\bar{\delta}^t = \frac{1}{N} \sum_i \delta_i^t$.

Assumption 1 (Lipschitz regularity). ∇f_i , $\nabla_{\mathcal{R}_s} F_i(\mathcal{R}_i; \cdot)$, and $\nabla_{\mathcal{R}_c} F_i(\mathcal{R}_i; \cdot)$ are β -, $L_Y^{(\ell)}$ -, and $L_G^{(\ell)}$ -Lipschitz: for any $\mathcal{R}, \mathcal{R}'$ and $Y_1, Y_2, G_1, G_2 \in \mathbb{R}^{S \times E}$,

$$\|\nabla f_i(\mathcal{R}) - \nabla f_i(\mathcal{R}')\| \leq \beta \|\mathcal{R} - \mathcal{R}'\|, \quad (16)$$

$$\|\nabla_{\mathcal{R}_s} F_i(\mathcal{R}_i; Y_1) - \nabla_{\mathcal{R}_s} F_i(\mathcal{R}_i; Y_2)\| \leq L_Y^{(\ell)} \|Y_1 - Y_2\|_F, \quad (17)$$

$$\|\nabla_{\mathcal{R}_c} F_i(\mathcal{R}_i; G_1) - \nabla_{\mathcal{R}_c} F_i(\mathcal{R}_i; G_2)\| \leq L_G^{(\ell)} \|G_1 - G_2\|_F, \quad (18)$$

with ℓ the cut layer.

Assumption 2 (Bounded variance and second moment). The stochastic gradient is unbiased, and its per-layer variance and second moment are bounded:

$$\mathbb{E}_{\xi_i \sim \mathcal{D}_i} \|\nabla F_i(\mathcal{R}; \xi_i) - \nabla f_i(\mathcal{R})\|^2 \leq \sum_{j=1}^L \sigma_j^2, \forall \mathcal{R}, \forall i, \quad (19)$$

$$\mathbb{E}_{\xi_i \sim \mathcal{D}_i} \|\nabla F_i(\mathcal{R}; \xi_i)\|^2 \leq \sum_{j=1}^L G_j^2, \forall \mathcal{R}, \forall i, \quad (20)$$

where L is the number of layers, and σ_j^2, G_j^2 bound the variance and second moment for the j -th layer of model W .

Assumption 3 (Unbiased bounded quantization). The quantizer $\mathcal{Q}(\cdot)$ is unbiased and variance-bounded, and its quantization error variance is bounded by $c_q < \frac{1}{\sqrt{2(1+8r)}}$,

$$\mathbb{E}_{\mathcal{Q}}[\mathcal{Q}(x)] = x, \quad \mathbb{E}_{\mathcal{Q}}\|\mathcal{Q}(x) - x\|^2 \leq c_q^2 \|x\|^2, \quad (21)$$

Lemma 1 (Recursive stability of activation cache error). Under Assumption 3, Algorithms 1 and 2 ensure

$$\mathbb{E}\|\varepsilon_{y,i,b}^{t,\ell}\|_F^2 \leq \sum_{\substack{\tau \in \mathcal{T}_{i,b} \\ \min \tau_{i,b} < \tau \leq t}} \rho_y^{m_{i,b}(\tau) - m_{i,b}(\tau)} A_{y,i,b}^{\tau,\ell}, \quad (22)$$

where $A_{y,i,b}^{\tau,\ell} = \sum_{k>2r} (\sigma_{y,i,b,k}^{\tau,\ell})^2 + 2\bar{c}_y^2 \mathbb{E}\|\Delta Y_{i,b}^{\tau,\ell}\|_F^2$.

Proof. Decompose $\varepsilon_{y,i,b}^{t,\ell} = \varepsilon_{\text{svd}} + \varepsilon_q$ with $\varepsilon_{\text{svd}} = PQ - R_{y,i,b}^{t,\ell}$, where $P = P_{y,i,b}^{t,\ell}$, $Q = Q_{y,i,b}^{t,\ell}$, $\hat{P} = P + \Delta P$, $\hat{Q} = Q + \Delta Q$, and $\varepsilon_q = \hat{P}\hat{Q} - PQ = \Delta P Q + P \Delta Q + \Delta P \Delta Q$. By Assumption 3, $\mathbb{E}[\Delta P | P] = \mathbb{E}[\Delta Q | Q] = 0$ with $\Delta P, \Delta Q$ conditionally independent, so $\mathbb{E}[\varepsilon_q | P, Q] = 0$ and the cross term vanishes; with Eckart–Young–Mirsky [35], [36],

$$\mathbb{E}\|\varepsilon_{y,i,b}^{t,\ell}\|_F^2 = \sum_{k>2r} (\sigma_{y,i,b,k}^{t,\ell})^2 + \mathbb{E}\|\varepsilon_q\|_F^2. \quad (23)$$

Submultiplicativity, conditional independence, and Assumption 3, with $\|Q\|_2 = 1$, $\|Q\|_F^2 = 2r$, $\|P\|_2^2 \leq \|P\|_F^2 = \|PQ\|_F^2$, bound the quantization term,

$$\begin{aligned} \mathbb{E}\|\varepsilon_q\|_F^2 &\leq c_q^2 \|Q\|_2^2 \|P\|_F^2 + c_q^2 \|P\|_2^2 \|Q\|_F^2 + c_q^4 \|P\|_F^2 \|Q\|_F^2 \\ &\leq (c_q^2(1+2r) + 2rc_q^4) \|PQ\|_F^2 = \bar{c}_y^2 \|PQ\|_F^2, \end{aligned} \quad (24)$$

which yields the one-step bound

$$\mathbb{E}\|\varepsilon_{y,i,b}^{t,\ell}\|_F^2 \leq \sum_{k>2r} (\sigma_{y,i,b,k}^{t,\ell})^2 + \bar{c}_y^2 \|P_{y,i,b}^{t,\ell} Q_{y,i,b}^{t,\ell}\|_F^2. \quad (25)$$

Since $R_{y,i,b}^{t,\ell} = \Delta Y_{i,b}^{t,\ell} - \varepsilon_{y,i,b}^{\pi_{i,b}(t),\ell}$ and $\|P_{y,i,b}^{t,\ell} Q_{y,i,b}^{t,\ell}\|_F^2 \leq \|R_{y,i,b}^{t,\ell}\|_F^2$, applying $\|x - y\|^2 \leq 2\|x\|^2 + 2\|y\|^2$ to (25) gives

$$\mathbb{E}\|\varepsilon_{y,i,b}^{t,\ell}\|_F^2 \leq A_{y,i,b}^{t,\ell} + \rho_y \mathbb{E}\|\varepsilon_{y,i,b}^{\pi_{i,b}(t),\ell}\|_F^2. \quad (26)$$

Indexing the appearances of sample b as $\tau_1 < \dots < \tau_M = t$ ($M = m_{i,b}(t)$, $\pi_{i,b}(\tau_m) = \tau_{m-1}$) and unrolling from $\varepsilon_{y,i,b}^{\tau_1,\ell} = 0$ with $\rho_y = 2\bar{c}_y^2 < 1$ yields the claim. $\square$

Lemma 2 (Recursive stability of gradient cache error). Under Assumption 3, Algorithms 1 and 2 ensure

$$\mathbb{E}\|\varepsilon_{g,i,b}^{t,\ell}\|_F^2 \leq \sum_{\substack{\tau \in \mathcal{T}_{i,b} \\ \min \tau_{i,b} < \tau \leq t}} \rho_g^{m_{i,b}(\tau) - m_{i,b}(\tau)} A_{g,i,b}^{\tau,\ell}, \quad (27)$$

where $A_{g,i,b}^{\tau,\ell} = \sum_{k>4r} (\sigma_{g,i,b,k}^{\tau,\ell})^2 + 2\bar{c}_g^2 \mathbb{E}\|\Delta G_{i,b}^{\tau,\ell}\|_F^2$.

Proof. The argument mirrors Lemma 1 with rank $4r$. With $P = P_{g,i,b}^{t,\ell}$, $Q = Q_{g,i,b}^{t,\ell}$, the factor noises are unbiased and

conditionally independent, so the cross term vanishes and, by Eckart–Young–Mirsky,

$$\mathbb{E}\|\varepsilon_{g,i,b}^{t,\ell}\|_F^2 = \sum_{k>4r} (\sigma_{g,i,b,k}^{t,\ell})^2 + \mathbb{E}\|\varepsilon_q\|_F^2. \quad (28)$$

Using $\|Q\|_2 = 1$, $\|Q\|_F^2 = 4r$, $\|P\|_2^2 \leq \|P\|_F^2 = \|PQ\|_F^2$, and Assumption 3,

$$\mathbb{E}\|\varepsilon_q\|_F^2 \leq (\bar{c}_q^2(1+4r) + 4rc_q^4)\|PQ\|_F^2 = \bar{c}_g^2\|PQ\|_F^2, \quad (29)$$

giving the one-step bound $\mathbb{E}\|\varepsilon_{g,i,b}^{t,\ell}\|_F^2 \leq \sum_{k>4r} (\sigma_{g,i,b,k}^{t,\ell})^2 + \bar{c}_g^2\|P_{g,i,b}^{t,\ell} Q_{g,i,b}^{t,\ell}\|_F^2$. With $R_{g,i,b}^{t,\ell} = \Delta G_{g,i,b}^{t,\ell} - \varepsilon_{g,i,b}^{\pi_{i,b}(t),\ell}$, $\|PQ\|_F^2 \leq \|R_{g,i,b}^{t,\ell}\|_F^2$, and $\|x - y\|^2 \leq 2\|x\|^2 + 2\|y\|^2$,

$$\mathbb{E}\|\varepsilon_{g,i,b}^{t,\ell}\|_F^2 \leq A_{g,i,b}^{t,\ell} + \rho_g \mathbb{E}\|\varepsilon_{g,i,b}^{\pi_{i,b}(t),\ell}\|_F^2. \quad (30)$$

Unrolling from $\varepsilon_{g,i,b}^{\tau_1,\ell} = 0$ with $\rho_g = 2\bar{c}_g^2 < 1$ yields the claim. $\square$

By Assumption 1, for each sample b , $\mathbb{E}\|\delta_{i,b}^t\|^2 \leq (L_Y^{(\ell)})^2 \mathbb{E}\|\varepsilon_{y,i,b}^{t,\ell}\|_F^2 + (L_G^{(\ell)})^2 \mathbb{E}\|\varepsilon_{g,i,b}^{t,\ell}\|_F^2$. Averaging over the mini-batch and clients, the average perturbation obeys $\mathbb{E}\|\bar{\delta}^t\|^2 \leq \epsilon_{\text{comp}}^t$, where

$$\epsilon_{\text{comp},i}^t = \frac{1}{B} \sum_{b \in \mathcal{B}_i^t} \left[(L_Y^{(\ell)})^2 \mathbb{E}\|\varepsilon_{y,i,b}^{t,\ell}\|_F^2 + (L_G^{(\ell)})^2 \mathbb{E}\|\varepsilon_{g,i,b}^{t,\ell}\|_F^2 \right], \quad (31)$$

where $\epsilon_{\text{comp}}^t = \frac{1}{N} \sum_{i=1}^N \epsilon_{\text{comp},i}^t$.

Lemma 3 (Client-side LoRA model drift). *Under Assumption 2, Algorithms 1 and 2 ensure*

$$\begin{aligned} \mathbb{E}\|\mathcal{R}_{c,i}^t - \mathcal{R}_c^t\|^2 &\leq \mathbf{1}_{\{I>1\}} 8\eta^2 I^2 \sum_{j=1}^{L_c} G_j^2 \\ &\quad + \mathbf{1}_{\{I>1\}} 4\eta^2 I \sum_{\tau=t_0}^{t-1} (\epsilon_{\text{comp},i}^\tau + \epsilon_{\text{comp}}^\tau), \end{aligned} \quad (32)$$

with I the client-side aggregation interval.

Proof. Since server-side adapters aggregate every iteration, $\mathcal{R}_{s,i}^t = \mathcal{R}_s^t$, so $\|\mathcal{R}_i^t - \mathcal{R}^t\|^2 = \|\mathcal{R}_{c,i}^t - \mathcal{R}_c^t\|^2$. Let $t_0 \leq t$ be the latest client-side aggregation ($t_0 \bmod I = 0$, $t - t_0 \leq I$), at which $\mathcal{R}_{c,i}^{t_0} = \mathcal{R}_c^{t_0}$. Accumulating the applied gradients over $[t_0, t)$,

$$\mathcal{R}_c^t - \mathcal{R}_{c,i}^t = \eta \sum_{\tau=t_0}^{t-1} \tilde{g}_{c,i}^\tau - \eta \sum_{\tau=t_0}^{t-1} \frac{1}{N} \sum_{k=1}^N \tilde{g}_{c,k}^\tau. \quad (33)$$

By $\|x + y\|^2 \leq 2\|x\|^2 + 2\|y\|^2$ (with the indicator since $t = t_0$ when $I = 1$), $\|\sum_{\tau} z^\tau\|^2 \leq (t - t_0) \sum_{\tau} \|z^\tau\|^2$, and Jensen's inequality,

$$\begin{aligned} \mathbb{E}\|\mathcal{R}_c^t - \mathcal{R}_{c,i}^t\|^2 &\leq \mathbf{1}_{\{I>1\}} 2\eta^2 (t - t_0) \sum_{\tau=t_0}^{t-1} (\mathbb{E}\|\tilde{g}_{c,i}^\tau\|^2 \\ &\quad + \frac{1}{N} \sum_{k=1}^N \mathbb{E}\|\tilde{g}_{c,k}^\tau\|^2). \end{aligned} \quad (34)$$

The applied second moment satisfies $\mathbb{E}\|\tilde{g}_{c,i}^\tau\|^2 \leq 2 \sum_{j=1}^{L_c} G_j^2 + 2\epsilon_{\text{comp},i}^\tau$, by the bounded second moment and $\mathbb{E}\|\delta_{c,i}^\tau\|^2 \leq \mathbb{E}\|\delta_i^\tau\|^2 \leq \epsilon_{\text{comp},i}^\tau$. Substituting this with its client average, using $t - t_0 \leq I$, and averaging over clients with $\frac{1}{N} \sum_i \epsilon_{\text{comp},i}^\tau = \epsilon_{\text{comp}}^\tau$ yields the claim. $\square$

Theorem 1 (Main convergence result). *Let Assumptions 1–3 hold. If $0 < \eta \leq \frac{1}{2\beta}$ and $c_q < \frac{1}{\sqrt{2(1+8r)}}$, then*

$$\begin{aligned} \frac{1}{T} \sum_{t=0}^{T-1} \mathbb{E}\|\nabla f(\mathcal{R}^t)\|^2 &\leq \frac{4(f(\mathcal{R}^0) - f^*)}{\eta T} \\ &\quad + \mathbf{1}_{\{I>1\}} 16\beta^2 \eta^2 I^2 \sum_{j=1}^{L_c} G_j^2 + \frac{4\beta\eta}{N} \sum_{j=1}^L \sigma_j^2 \\ &\quad + 4 \left(1 + \beta\eta + \mathbf{1}_{\{I>1\}} 4\beta^2 \eta^2 I^2 \right) \frac{1}{T} \sum_{t=0}^{T-1} \epsilon_{\text{comp}}^t. \end{aligned} \quad (35)$$

where L and f^* denote the total number of global model layers and the minimum value of problem (1).

Proof. By β -smoothness of f and $\mathcal{R}^{t+1} = \mathcal{R}^t - \eta(\bar{g}^t + \bar{\delta}^t)$,

$$\begin{aligned} f(\mathcal{R}^{t+1}) &\leq f(\mathcal{R}^t) - \eta \langle \nabla f(\mathcal{R}^t), \bar{g}^t \rangle \\ &\quad - \eta \langle \nabla f(\mathcal{R}^t), \bar{\delta}^t \rangle + \frac{\beta\eta^2}{2} \|\bar{g}^t + \bar{\delta}^t\|^2. \end{aligned} \quad (36)$$

We bound the three terms in expectation with $h^t = \frac{1}{N} \sum_i \nabla f_i(\mathcal{R}_i^t)$ and $\mathbb{E}[\bar{g}^t] = h^t$. The polarization identity, Jensen's inequality, and Assumption 1 (so $\mathbb{E}\|\nabla f(\mathcal{R}^t) - h^t\|^2 \leq \beta^2 D^t$) give

$$\begin{aligned} \mathbb{E}[-\eta \langle \nabla f(\mathcal{R}^t), \bar{g}^t \rangle] &\leq -\frac{\eta}{2} \mathbb{E}\|\nabla f(\mathcal{R}^t)\|^2 - \frac{\eta}{2} \mathbb{E}\|h^t\|^2 \\ &\quad + \frac{\eta\beta^2}{2} D^t, \end{aligned} \quad (37)$$

where $D^t \triangleq \frac{1}{N} \sum_{i=1}^N \mathbb{E}\|\mathcal{R}_i^t - \mathcal{R}^t\|^2$.

As truncated SVD biases $\bar{\delta}^t$, Young's inequality and the perturbation bound $\mathbb{E}\|\bar{\delta}^t\|^2 \leq \epsilon_{\text{comp}}^t$ give

$$\mathbb{E}[-\eta \langle \nabla f(\mathcal{R}^t), \bar{\delta}^t \rangle] \leq \frac{\eta}{4} \mathbb{E}\|\nabla f(\mathcal{R}^t)\|^2 + \eta \epsilon_{\text{comp}}^t, \quad (38)$$

while $\|x + y\|^2 \leq 2\|x\|^2 + 2\|y\|^2$ with Assumption 2 bounds the quadratic term,

$$\begin{aligned} \mathbb{E}\left[\frac{\beta\eta^2}{2} \|\bar{g}^t + \bar{\delta}^t\|^2\right] &\leq \beta\eta^2 \mathbb{E}\|h^t\|^2 + \frac{\beta\eta^2}{N} \sum_{j=1}^L \sigma_j^2 \\ &\quad + \beta\eta^2 \epsilon_{\text{comp}}^t. \end{aligned} \quad (39)$$

Summing the three, the $\mathbb{E}\|h^t\|^2$ coefficient $-(\frac{\eta}{2} - \beta\eta^2)$ is non-positive for $\eta \leq 1/(2\beta)$ and is dropped; substituting Lemma 3 for D^t then yields

$$\begin{aligned} \frac{\eta}{4} \mathbb{E}\|\nabla f(\mathcal{R}^t)\|^2 &\leq \mathbb{E}[f(\mathcal{R}^t)] - \mathbb{E}[f(\mathcal{R}^{t+1})] \\ &\quad + \frac{\beta\eta^2}{N} \sum_{j=1}^L \sigma_j^2 + (\eta + \beta\eta^2) \epsilon_{\text{comp}}^t \\ &\quad + \mathbf{1}_{\{I>1\}} 4\beta^2 \eta^3 \left(I^2 \sum_{j=1}^{L_c} G_j^2 + I \sum_{\tau=t_0(t)}^{t-1} \epsilon_{\text{comp}}^\tau \right). \end{aligned} \quad (40)$$

Summing over $t = 0, \dots, T-1$, the function values telescope to $f(\mathcal{R}^0) - \mathbb{E}[f(\mathcal{R}^T)] \leq f(\mathcal{R}^0) - f^*$, and the drift double sum counts each $\epsilon_{\text{comp}}^\tau$ at most I times, $\sum_{t=0}^{T-1} \sum_{\tau=t_0(t)}^{t-1} \epsilon_{\text{comp}}^\tau \leq I \sum_{\tau=0}^{T-1} \epsilon_{\text{comp}}^\tau$. Dividing by $\eta T/4$ yields the stated bound. $\square$

Insights. The four terms in Theorem 1 respectively capture the $\mathcal{O}(1/T)$ optimization gap, client-side model drift, stochastic gradient noise, and compression error. In the fourth term, the factor $1 + \beta\eta$ captures the direct effect of compression-induced gradient perturbation, whereas the I^2 -dependent factor

TABLE I. LLM fine-tuning accuracy (%) on GLUE tasks and communication cost relative to SplitLoRA (%). Each cell is reported as “accuracy/uplink-only communication cost/total communication cost”, where SplitLoRA is normalized to 100%. Boldface denotes the highest accuracy and the lowest traffic for each task.

Model	Task	SplitLoRA	SplitLite	AQ-SGD	Rand-Top- k	AdaptSFL
Qwen-2.5 (1.5B)	MNLI	83.8/100/100	85.1/6.8/16.9	84.5/25.0/37.5	84.9/50.5/40.2	83.7/100/100
	MRPC	78.7/100/100	81.4/6.8/16.9	78.2/25.0/37.5	80.8/50.5/40.2	77.7/100/100
	RTE	81.2/100/100	82.0/6.8/16.9	82.0/25.0/37.5	82.0/50.5/40.2	73.3/100/100
Qwen-2.5 (3B)	MNLI	86.9/100/100	87.4/6.6/16.6	86.6/25.0/37.5	87.4/50.6/40.3	86.9/100/100
	MRPC	77.2/100/100	77.5/6.6/16.6	77.7/25.0/37.5	79.2/50.6/40.3	75.3/100/100
	RTE	83.8/100/100	85.9/6.6/16.6	87.4/25.0/37.5	86.3/50.6/40.3	80.1/100/100
Gemma-3 (1B)	MNLI	82.7/100/100	83.8/6.9/17.4	82.7/25.0/37.5	83.7/50.5/40.2	81.9/100/100
	MRPC	77.5/100/100	78.2/6.9/17.4	77.9/25.0/37.5	75.7/50.5/40.2	77.9/100/100
	RTE	76.9/100/100	80.1/6.9/17.4	79.4/25.0/37.5	78.7/50.5/40.2	73.7/100/100
Llama-3.2 (3B)	MNLI	87.2/100/100	87.1/6.5/16.3	87.4/25.0/37.5	87.6/52.5/41.2	87.1/100/100
	MRPC	82.4/100/100	82.4/6.5/16.3	82.4/25.0/37.5	82.1/52.5/41.2	74.5/100/100
	RTE	88.8/100/100	88.8/6.5/16.3	88.5/25.0/37.5	87.7/52.5/41.2	66.8/100/100

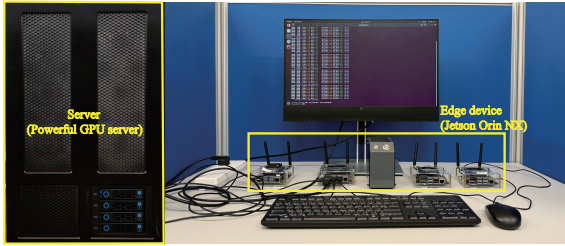


Fig. 4. The testbed composed of an RTX 4090 server and multiple Jetson Orin NX clients.

captures its indirect effect through client drift. By Lemmas 1 and 2, increasing r reduces the SVD tails $\sum_{k>2r} \sigma_{y,k}^2$ and $\sum_{k>4r} \sigma_{g,k}^2$, but increases quantization error and communication. A higher bitwidth lowers c_q and tightens the stable cache recursion with $\rho_y, \rho_g < 1$. Thus, the bitwidth should be selected to balance model performance and communication overhead. At the same time, a smaller I reduces both the second term and the drift-induced part of the fourth term at the cost of more frequent LoRA synchronization.

VI. EVALUATION

A. Experimental Setup

Datasets and models. We evaluate SplitLite on a subset of the GLUE benchmark [37], covering three representative tasks: MNLI (three-way natural-language inference), MRPC (paraphrase detection), and RTE (binary natural-language inference). We partition each dataset across clients under a Non-IID setting, where Non-IID partitions are generated using a Dirichlet distribution with $\alpha = 0.5$ [38]. We adopt four pre-trained causal language models of different scales: Qwen-2.5-1.5B, Qwen-2.5-3B [39], Gemma-3-1B [40], and Llama-3.2-3B [41]. All SFL experiments are conducted with $N = 6$ clients.

Baselines. We benchmark SplitLite against four representative methods: SplitLoRA [11], which partitions the model into client-side and server-side sub-models and updates only LoRA adapters without compressing cut-layer communication; AdaptSFL [23], which adaptively selects the cut layer and

local update frequency to balance computation and communication under resource constraints; AQ-SGD [19], which quantizes cut-layer activations based on their temporal variation across training epochs; and Rand-Top- k [29], which sparsifies activations and gradients by randomly retaining the top- k coordinates.

System settings. For all LoRA-based methods, the rank is set to 16. The model is split at layer 3 for homogeneous methods, while AdaptSFL [23] adopts heterogeneous local depths of $[3, 3, 2, 2, 3, 3]$ across six clients. For SplitLite, activation and gradient residuals are compressed to ranks of 32 and 64, respectively, followed by 4-bit and 8-bit quantization. For AQ-SGD [19], 4-bit and 8-bit fixed-point quantization are applied to activation residuals and gradients, respectively. For Rand-Top- k [29], the sparsification ratio is set to 0.3. Extensive simulations and realistic laboratory experiments were conducted on the testbed system shown in Fig. 4, where the server is equipped with four NVIDIA RTX 4090 GPUs and the client nodes consist of multiple NVIDIA Jetson Orin NX devices [42]. To emulate realistic wireless communication between clients and the server, gRPC [43] is adopted as the communication framework.

B. Experimental Results

1) *Fine-Tuning Model Performance:* Table I reports the fine-tuning accuracy and communication overhead on GLUE tasks. SplitLite achieves accuracy comparable to or higher than SplitLoRA across nearly all model-task combinations; for example, on MNLI, it improves Gemma-3-1B from 82.7% to 83.8% and Qwen-2.5-3B from 86.9% to 87.4%. Meanwhile, SplitLite reduces the activation uplink and total communication to 6.5%–6.9% and 16.3%–17.4% of those of SplitLoRA, corresponding to maximum reductions of 93.5% and 83.7%, respectively. These results verify that the LoRA-induced temporal residuals can be represented by extremely compact low-rank factors without sacrificing model performance. The MRPC training curves in Fig. 5 further show stable convergence, while comparisons with AQ-SGD and Rand-Top-

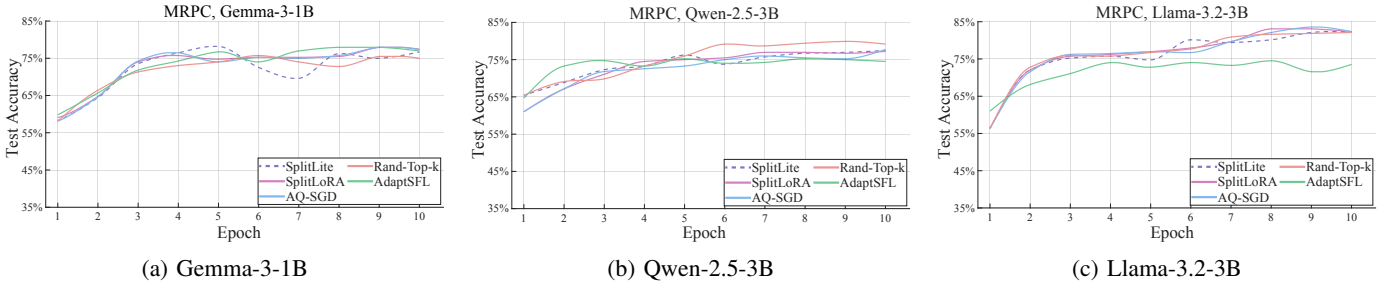


Fig. 5. Training curves of compared methods on MRPC for (a) Gemma-3-1B, (b) Qwen-2.5-3B, and (c) Llama-3.2-3B.

TABLE II. LLM fine-tuning latency (in hours) on GLUE tasks. Boldface denotes the lowest latency for each task.

Model	Task	SplitLoRA	SplitLite	AQ-SGD	Rand-Top- k
Qwen-2.5 (1.5B)	MNLI	6.93	2.02	2.74	3.82
	MRPC	1.30	0.38	0.51	0.71
	RTE	0.88	0.25	0.34	0.48
Qwen-2.5 (3B)	MNLI	9.40	2.73	3.83	5.26
	MRPC	1.76	0.51	0.71	0.98
	RTE	1.19	0.34	0.48	0.66
Gemma-3 (1B)	MNLI	5.27	1.71	2.13	2.95
	MRPC	0.98	0.31	0.39	0.54
	RTE	0.67	0.22	0.27	0.37
Llama-3.2 (3B)	MNLI	6.90	2.38	3.19	4.22
	MRPC	2.58	0.68	1.01	1.45
	RTE	1.22	0.37	0.52	0.71

TABLE III. Impact of heterogeneous cut layers on SplitLite.

Model	Task	Homogeneous	Heterogeneous
Qwen-2.5 (1.5B)	MNLI	85.1 /6.8/16.9	84.7/6.8/16.9
	MRPC	81.4 /6.8/16.9	80.6/6.8/16.9
	RTE	82.0 /6.8/16.9	76.2/6.8/16.9
Qwen-2.5 (3B)	MNLI	87.4 /6.6/16.6	87.1/6.6/16.6
	MRPC	77.5 /6.6/16.6	75.5/6.6/16.6
	RTE	85.9 /6.6/16.6	80.1/6.6/16.6
Gemma-3 (1B)	MNLI	83.8 /6.9/17.4	83.0/6.9/17.4
	MRPC	78.2 /6.9/17.4	77.9/6.9/17.4
	RTE	80.1 /6.9/17.4	73.7/6.9/17.4
Llama-3.2 (3B)	MNLI	87.1 /6.5/16.3	86.6/6.5/16.3
	MRPC	82.4 /6.5/16.3	75.5/6.5/16.3
	RTE	88.8 /6.5/16.3	66.8/6.5/16.3

k confirm the favorable communication–accuracy trade-off of SplitLite.

As reported in Table II, SplitLite also achieves the lowest end-to-end fine-tuning latency across all evaluated settings, reducing training time by up to 74% over SplitLoRA. On MNLI, for example, the latency decreases from 9.40 to 2.73 hours for Qwen-2.5-3B and from 6.90 to 2.38 hours for Llama-3.2-3B.

2) *Communication Compression*: Table I reports the communication cost of all methods relative to SplitLoRA. Following the steady-state communication accounting adopted by AQ-SGD [19], we report the communication costs of all methods from the second epoch onward. Our method consistently achieves the lowest communication overhead across all evaluated models and tasks.

Specifically, SplitLite reduces the activation uplink and total communication to 6.5%–6.9% and 16.3%–17.4% of those of SplitLoRA, corresponding to maximum reductions of 93.5% and 83.7%, respectively. The larger uplink savings are particularly beneficial for wireless edge devices, whose uplink rates are typically lower than downlink rates. For example, on MNLI with Gemma-3-1B, SplitLite reduces the uplink and total communication to 6.9% and 17.4%, respectively, while improving accuracy from 82.7% to 83.8%. It also substantially outperforms AQ-SGD and Rand-Top- k , which retain 25.0% and 50.5%–52.5% of the uplink cost, and 37.5% and 40.2%–41.2% of the total communication cost, respectively.

These results demonstrate that exploiting the LoRA-induced low effective rank of temporal residuals effectively eliminates redundant cut-layer traffic on both uplink and downlink.

3) *Ablation Study*: To assess the impact of cut-layer heterogeneity, we compare SplitLite under homogeneous and heterogeneous cut-layer configurations, as reported in Table III. The heterogeneous configuration follows the cut-layer assignment adopted in AdaptSFL [23]. Unlike SplitLoRA [11], where all clients share the same cut layer, AdaptSFL adopts client-specific split depths according to different device capabilities. As shown in Tables I and III, while cut-layer heterogeneity introduces additional aggregation inconsistency and leads to accuracy degradation relative to the homogeneous configuration, SplitLite under heterogeneous cut layers consistently matches or surpasses AdaptSFL in accuracy at only 6.5%–6.9% uplink and 16.3%–17.4% total communication cost. For instance, on MNLI with Qwen-2.5-3B, SplitLite achieves 87.1% accuracy under the heterogeneous configuration against AdaptSFL’s 86.9%, while reducing total communication cost by 83.4%. Notably, the communication overhead under heterogeneous cut layers stays identical to that of the homogeneous case, confirming that the uplink and total communication savings are preserved regardless of cut-layer configuration. These observations indicate that our low-rank residual compression framework is robust to cut-layer heterogeneity.

VII. CONCLUSION

This paper proposed SplitLite, a communication-efficient split federated LoRA fine-tuning framework for on-device LLMs that exploits the LoRA-induced low effective rank of cut-layer activation and gradient residuals across adjacent epochs. SplitLite transmits only quantized truncated-SVD factors of rank- $2r$ activation and rank- $4r$ gradient residuals, reducing bidirectional cut-layer traffic without auxiliary learning. We theoretically characterize this low-rank structure

and incorporate both SVD truncation and quantization errors into the convergence analysis. Experiments on GLUE with advanced on-device LLMs demonstrate reductions of up to 93.5% in activation uplink communication and 83.7% in total communication cost, without degrading the fine-tuning performance. Beyond edge learning, SplitLite holds strong potential for model-parallel cloud training, which we leave as future work.

REFERENCES

- [1] O. Friha, M. A. Ferrag, B. Kantarci, B. Cakmak, A. Ozgun, and N. Ghoualmi-Zine, "Llm-based edge intelligence: A comprehensive survey on architectures, applications, security and trustworthiness," *IEEE Open Journal of the Communications Society*, vol. 5, pp. 5799–5856, 2024.
- [2] Q. Li, Y. Diao, Q. Chen, and B. He, "Federated learning on non-iid data silos: An experimental study," in *2022 IEEE 38th international conference on data engineering (ICDE)*. IEEE, 2022, pp. 965–978.
- [3] P. Voigt and A. Von dem Bussche, "The EU general data protection regulation (GDPR)," *A practical guide*, 1st ed., Cham: Springer International Publishing, vol. 10, no. 3152676, pp. 10–5555, 2017.
- [4] 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.
- [5] Z. Tang, J. Huang, R. Yan, Y. Wang, Z. Tang, S. Shi, A. C. Zhou, and X. Chu, "Bandwidth-aware and overlap-weighted compression for communication-efficient federated learning," in *Proceedings of the 53rd International Conference on Parallel Processing*, 2024, pp. 866–875.
- [6] G. Qu, Q. Chen, W. Wei, Z. Lin, X. Chen, and K. Huang, "Mobile edge intelligence for large language models: A contemporary survey," *IEEE Communications Surveys & Tutorials*, 2025.
- [7] M. Xu, D. Cai, Y. Wu, X. Li, and S. Wang, "{FwdLLM}: Efficient federated finetuning of large language models with perturbed inferences," in *2024 USENIX Annual Technical Conference (USENIX ATC 24)*, 2024, pp. 579–596.
- [8] L. Cao, Y. Zhu, and W. Gong, "Sfprompt: Communication-efficient split federated fine-tuning for large pre-trained models over resource-limited devices," *arXiv preprint arXiv:2407.17533*, 2024.
- [9] E. J. Hu, Y. Shen, P. Wallis, Z. Allen-Zhu, Y. Li, S. Wang, L. Wang, W. Chen *et al.*, "Lora: Low-rank adaptation of large language models," *ICLR*, vol. 1, no. 2, p. 3, 2022.
- [10] 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.
- [11] Z. Lin, X. Hu, Y. Zhang, Z. Chen, Z. Fang, X. Chen, A. Li, P. Vepakomma, and Y. Gao, "Splitlora: A split parameter-efficient fine-tuning framework for large language models," *arXiv preprint arXiv:2407.00952*, 2024.
- [12] T. Li, Y. Tang, Y. Song, C. Wu, X. Liu, P. Li, and X. Chen, "Splitcom: Communication-efficient split federated fine-tuning of llms via temporal compression," *arXiv preprint arXiv:2602.10564*, 2026.
- [13] S. Zhang, G. Cheng, W. Wu, X. Huang, L. Song, and X. Shen, "Split fine-tuning for large language models in wireless networks," *IEEE Journal of Selected Topics in Signal Processing*, 2025.
- [14] H. Lim, J. Lee, J. Lee, S. D. Sathyanarayana, J. Kim, A. Nguyen, K. T. Kim, Y. Im, M. Chiang, D. Grunwald *et al.*, "An empirical study of 5g: Effect of edge on transport protocol and application performance," *IEEE Transactions on Mobile Computing*, vol. 23, no. 4, pp. 3172–3186, 2023.
- [15] R. Wyrzykowski, "Mobile network experience report: Hong kong," May 2024. [Online]. Available: <https://www.opensignal.com/reports/2024/05/hongkong/mobile-network-experience>
- [16] T. Dettmers, M. Lewis, Y. Belkada, and L. Zettlemoyer, "Gpt3. int8 (): 8-bit matrix multiplication for transformers at scale," *Advances in neural information processing systems*, vol. 35, pp. 30318–30332, 2022.
- [17] M. Sun, X. Chen, J. Z. Kolter, and Z. Liu, "Massive activations in large language models," *arXiv preprint arXiv:2402.17762*, 2024.
- [18] Y. Bondarenko, M. Nagel, and T. Blankevoort, "Understanding and overcoming the challenges of efficient transformer quantization," in *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, 2021, pp. 7947–7969.
- [19] J. Wang, B. Yuan, L. Rimanic, Y. He, T. Dao, B. Chen, C. Ré, and C. Zhang, "Fine-tuning language models over slow networks using activation quantization with guarantees," *Advances in Neural Information Processing Systems*, vol. 35, pp. 19215–19230, 2022.
- [20] 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*, Dec. 2018.
- [21] D. Cai, Y. Wu, S. Wang, F. X. Lin, and M. Xu, "Efficient federated learning for modern nlp," in *Proceedings of the 29th annual international conference on mobile computing and networking*, 2023, pp. 1–16.
- [22] O. Gupta and R. Raskar, "Distributed learning of deep neural network over multiple agents," *Journal of Network and Computer Applications*, vol. 116, pp. 1–8, 2018.
- [23] Z. Lin, G. Qu, W. Wei, X. Chen, and K. K. Leung, "AdaptSFL: Adaptive Split Federated Learning in Resource-constrained Edge Networks," *arXiv preprint arXiv:2403.13101*, Mar. 2024.
- [24] K. Zhao, C. Zhu, M. Chen, C. Huang, Z. Yang, and Z. Zhang, "Sfllm: Efficient split federated learning for large language model over wireless networks," in *GLOBECOM 2025-2025 IEEE Global Communications Conference*. IEEE, 2025, pp. 1835–1840.
- [25] Z. Wang, Y. Zhou, Y. Shi, and K. B. Letaief, "Federated fine-tuning for pre-trained foundation models over wireless networks," *IEEE Transactions on Wireless Communications*, vol. 24, no. 4, pp. 3450–3464, 2025.
- [26] D. Alistarh, D. Grubic, J. Li, R. Tomioka, and M. Vojnovic, "Qsgd: Communication-efficient sgd via gradient quantization and encoding," *Advances in neural information processing systems*, vol. 30, 2017.
- [27] Y. Lin, S. Han, H. Mao, Y. Wang, and B. Dally, "Deep gradient compression: Reducing the communication bandwidth for distributed training," in *International conference on learning representations*, 2018.
- [28] J. Wang, H. Qi, A. S. Rawat, S. Reddi, S. Waghmare, F. X. Yu, and G. Joshi, "Fedlite: A scalable approach for federated learning on resource-constrained clients," *arXiv preprint arXiv:2201.11865*, 2022.
- [29] F. Zheng, C. Chen, L. Lyu, and B. Yao, "Reducing communication for split learning by randomized top-k sparsification," *arXiv preprint arXiv:2305.18469*, 2023.
- [30] S. U. Stich, J.-B. Cordonnier, and M. Jaggi, "Sparsified sgd with memory," *Advances in neural information processing systems*, vol. 31, 2018.
- [31] N. Halko, P.-G. Martinsson, and J. A. Tropp, "Finding structure with randomness: Probabilistic algorithms for constructing approximate matrix decompositions," *arXiv preprint arXiv:0909.4061*, 2009.
- [32] H. Yu, R. Jin, and S. Yang, "On the linear speedup analysis of communication efficient momentum sgd for distributed non-convex optimization," in *International Conference on Machine Learning*. PMLR, 2019, pp. 7184–7193.
- [33] Y. Chen, S. Qian, H. Tang, X. Lai, Z. Liu, S. Han, and J. Jia, "Longlora: Efficient fine-tuning of long-context large language models," *arXiv preprint arXiv:2309.12307*, 2023.
- [34] S. Rajbhandari, O. Ruwase, J. Rasley, S. Smith, and Y. He, "Zero-infinity: Breaking the gpu memory wall for extreme scale deep learning," in *Proceedings of the international conference for high performance computing, networking, storage and analysis*, 2021, pp. 1–14.
- [35] C. Eckart and G. Young, "The approximation of one matrix by another of lower rank," *Psychometrika*, vol. 1, no. 3, pp. 211–218, 1936.
- [36] L. Mirsky, "Symmetric gauge functions and unitarily invariant norms," *The quarterly journal of mathematics*, vol. 11, no. 1, pp. 50–59, 1960.
- [37] A. Wang, A. Singh, J. Michael, F. Hill, O. Levy, and S. Bowman, "Glue: A multi-task benchmark and analysis platform for natural language understanding," in *Proceedings of the 2018 EMNLP workshop Black-boxNLP: Analyzing and interpreting neural networks for NLP*, 2018, pp. 353–355.
- [38] T.-M. H. Hsu, H. Qi, and M. Brown, "Measuring the effects of non-identical data distribution for federated visual classification," *arXiv preprint arXiv:1909.06335*, 2019.
- [39] A. Yang, B. Yang, B. Zhang *et al.*, "Qwen2.5 technical report," *arXiv preprint arXiv:2412.15115*, 2024.
- [40] Gemma Team, "Gemma 3 technical report," *arXiv preprint arXiv:2503.19786*, 2025.

- [41] A. Grattafiori, A. Dubey, A. Jauhri, A. Pandey, A. Kadian, A. Al-Dahle, A. Letman, A. Mathur, A. Schelten, A. Vaughan *et al.*, “The llama 3 herd of models,” *arXiv preprint arXiv:2407.21783*, 2024.
- [42] NVIDIA Corporation, “NVIDIA Jetson Orin for next-generation robotics and edge ai,” <https://www.nvidia.com/en-us/autonomous-machines/embedded-systems/jetson-orin/>.
- [43] Google and CNCF, “gRPC: A high performance, open-source universal RPC framework,” <https://grpc.io/>, 2015.