FedRTS: Federated Robust Pruning via Combinatorial Thompson Sampling

Hong Huang, Jinhai Yang, Yuan Chen, Jiaxun Ye, Dapeng Wu
Department of Computer Science
City University of Hong Kong
Hong Kong SAR, China
{hohuang-c, jinhayang7-c, ychen2752-c, jiaxunye-c}@my.cityu.edu.hk
dpwu@ieee.org

Abstract

Federated Learning (FL) enables collaborative model training across distributed clients without data sharing, but its high computational and communication demands strain resource-constrained devices. While existing methods use dynamic pruning to improve efficiency by periodically adjusting sparse model topologies while maintaining sparsity, these approaches suffer from issues such as **greedy adjustments**, **unstable topologies**, and **communication inefficiency**, resulting in less robust models and suboptimal performance under data heterogeneity and partial client availability. To address these challenges, we propose **Fed**erated **R**obust pruning via combinatorial **T**hompson **S**ampling (FedRTS), a novel framework designed to develop robust sparse models. FedRTS enhances robustness and performance through its Thompson Sampling-based Adjustment (TSAdj) mechanism, which uses probabilistic decisions informed by stable and farsighted information, instead of deterministic decisions reliant on unstable and myopic information in previous methods. Extensive experiments demonstrate that FedRTS achieves state-of-the-art performance in computer vision and natural language processing tasks while reducing communication costs, particularly excelling in scenarios with heterogeneous data distributions and partial client participation. Our codes are available at: https://github.com/Little0o0/FedRTS.

1 Introduction

Federated learning (FL) [34, 30, 25, 64, 65] enables edge devices with private local data to collaboratively train a model without sharing data. However, substantial computational and communication demands for training deep learning models often exceed the resource limitations of edge devices in cross-device FL scenarios. Although neural network pruning [12, 40, 33] offers a promising solution by removing redundant parameters, traditional pruning methods depend on resource-intensive dense model training, rendering them impractical for privacy-preserved and resource-constrained FL environments.

To address these challenges, recent federated pruning frameworks [1, 47, 57, 24, 19, 18, 42] have adopted dynamic sparse training techniques [11, 48, 22] within FL. These frameworks employ a two-loop training process: in the inner loop, model weights are updated through standard FL rounds with fixed model topology; in the outer loop, the server adjusts the model topology by pruning and reactivating parameters [11], as illustrated in Fig. 1 (left). This iterative process generates a specialized sparse model without dense model training, significantly reducing computational costs.

Despite these advancements, existing frameworks [1, 47, 57, 24, 19, 18, 42] suffer from three critical challenges in model topology adjustment, as illustrated in Fig. 1 (left). **(1) Greedy adjustment:**

39th Conference on Neural Information Processing Systems (NeurIPS 2025).

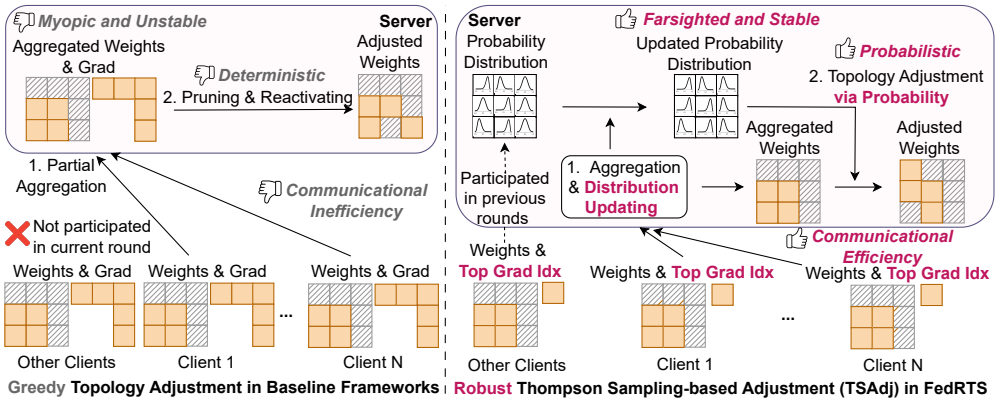

Figure 1: Illustration of topology adjustment in existing baselines [1]and FedRTS. *Left:* Existing baselines adjust model topology based on myopic and unstable aggregated weights and gradients via deterministic magnitude pruning and reactivating, resulting in greedy adjustment and high communication overhead. *Right:* FedRTS introduces Thompson Sampling-based Adjustment (TSAdj) to adjust the topology based on farsighted and stable probability distributions, achieving a robust topology and low communication overhead.

Current methods rely on myopic, aggregated information from a small subset of participating clients, ignoring data from the majority of unseen clients and prior knowledge. This leads to greedy adjustments and reduced robustness [25]. **(2) Unstable topology:** Deterministic adjustments based solely on aggregated information are prone to instability due to heterogeneous data distributions, resulting in unstable model topologies. **(3) Communication inefficiency:** Transmitting extensive auxiliary data (*e.g.,* full-size gradients) for topology updates imposes high communication costs. These limitations hinder the ability of current methods to handle client availability, data heterogeneity, and communication costs effectively, ultimately leading to suboptimal performance and inefficient resource utilization.

To address these challenges, *we reframe federated pruning as a combinatorial multi-armed bandit (CMAB) [5] problem and identify the above issues located in myopic observation and deterministic decision.* Building on this insight, we propose **Fed**erated **R**obust pruning via combinatorial **T**hompson **S**ampling (FedRTS), a novel framework designed to derive robust sparse models for cross-device FL, as illustrated in Fig. 2. FedRTS introduces the Thompson Sampling-based Adjustment (TSAdj) mechanism, depicted in Fig. 1 (right), to address the shortcomings of existing methods. First, TSAdj leverages farsighted probability distributions (including prior information from unseen clients) to mitigate the impact of partial client participation. Second, it uses probabilistic decisions based on stable, comprehensive information for topology adjustment. Third, FedRTS reduces communication overhead by requiring clients to upload only top-gradient indices instead of dense gradients.

We evaluate FedRTS on computer vision and natural language processing datasets, demonstrating its effectiveness in handling client availability and data heterogeneity. FedRTS achieves higher accuracy, better generalization, and lower communication costs compared to state-of-the-art (SOTA) methods. For instance, on the CIFAR-10 dataset with the ResNet18 model, FedRTS achieves either a 5.1% accuracy improvement or a 33.3% reduction in communication costs compared to SOTA frameworks. *To our knowledge, this paper is the first work to analyze federated pruning through a CMAB lens and apply combinatorial Thompson sampling to stabilize topology adjustments in FL.*

2 Preliminary and Challenges

2.1 Federated Dynamic Pruning

In federated pruning, $\mathcal{N}$ resource-constrained clients collaboratively train a sparse model using their local datasets $D_n, n \in \{1, 2, ..., \mathcal{N}\}$. Given the target density d', the objective is to solve the

[1]Baseline's workflow: `https://github.com/Little0o0/FedRTS/tree/main/image/workflow.mp4`

following optimization problem:

$$\min_{W,m} \sum_{n=1}^{\mathcal{N}} p_n f_n(W, m, D_n), \quad \text{s.t. } d \leq d', \tag{1}$$

where W represents the model weights, $m \in \{0,1\}^{\langle W \rangle}$ denotes the sparse model topology (also called mask), $f_n(\cdot)$ is the objective function depending on the task of the n-th for client n (*e.g.,* cross-entropy loss for the image classification task), $d = \frac{\text{sum}(m)}{\langle m \rangle}$ is the density of the topology m, and p_n is set as the proportion of the data size of client n, typically set as $p_n = \langle D_n \rangle / \sum_{i=1}^{\mathcal{N}} \langle D_i \rangle$. [2]

To solve the problem in Eq. 1 under the target density constraint d', existing federated pruning frameworks [1, 47, 57, 24, 19, 18, 42] apply dynamic sparse training techniques [11, 48, 22]. These frameworks iteratively adjust sparse on-device models during training while maintaining the density level $d \leq d'$. The process begins with a sparse model initialized via random pruning and follows a two-loop training procedure: In the inner loop, the model topology m remains fixed, and the model weights W are updated through traditional FL rounds. After ΔT inner loops, as illustrated in Fig. 1 (left), the framework enters the outer loop, where clients upload additional data (specifically, the gradients of the pruned weights) to the server. The server then adjusts the model topology by pruning and reactivating [11] based on the aggregated weights and gradients.

2.2 CMAB-Based Problem Formulation

The Combinatorial Multi-Armed Bandit (CMAB) [5, 55, 26] is a sequential decision-making framework where an agent selects combinations of arms (called super arms) with unknown reward distributions. The objective is to maximize cumulative rewards by balancing the exploration of uncertain arms and the exploitation of high-performing ones.

We formulate the topology adjustment as a CMAB problem, where each link m_i in the model topology serves as an arm. In each outer loop t, the server selects an action S_t, which includes K arms, where $K = d\langle m \rangle$. Selected arms $i \in S_t$ are activated, while others $i \notin S_t$ are deactivated, *i.e.,* $m_{t,i} = \mathbf{1}_{i \in S_t}$. After playing action, the new sparse weights $W_t = W_t^{agg} \odot m_t$ will be distributed to the clients for further training. The server then observes outcomes $X_t = (X_{t,1}, \cdots, X_{t,\langle m \rangle})$ for all arms, drawn from distributions with unknown expectation μ. After that, the server obtains the unknown rewards $R_t = R(S_t, X_t)$. At the next round $t+1$, the previous information $\{X_\tau | 1 \leq \tau \leq t\}$ is the input to the adjustment algorithm to select the action S_{t+1}. Following previous work [26], we assume that the expected reward of an action S_t only depends on S_t and and the mean vector μ, *i.e.,* there exists a function r such that $\mathbb{E}[R_t] = \mathbb{E}_{X_t}[R(S_t, X_t)] = r(S_t, \mu)$. Assuming the number of adjustments is T, the goal of CMAB is to maximize the cumulative reward, *i.e.,* $\max \sum_{t=1}^{T} \mathbb{E}[R_t] = \sum_{t=1}^{T} r(S_t, \mu)$.

2.3 Challenges in Previous Federated Dynamic Pruning Methods

From the CMAB aspect, we can rigorously analyse the challenges in the previous methods. After taking the action S_t to determine topology m_t, previous methods observe the delay outcomes X_t at the next outer loop $t+1$, defining $X_{t,i} = 1$ for i-th arm with the top importance score; otherwise, $X_{t,i} = 0$. And the next action S_{t+1} is selected **merely based on current outcomes** X_t, *i.e.,* $S_{t+1} = \{i | X_{t,i} = 1\}$. Thereby, they face three significant challenges:

Greedy Adjustment: Existing frameworks only observe the outcomes in the outer loop, ignoring the inner loop. Moreover, they discard all previous outcomes $\{X_\tau | 1 \leq \tau \leq t-1\}$, excluding information from unseen clients and previous data, leading to a myopic and greedy topology adjustment that is difficult to exploit from a global view.

Unstable Topology: Previous methods make a deterministic decision to select an action S_t based on unreliable outcomes only, without considering the high variance of outcomes and the expectation of distribution μ. Therefore, the selected action S_t is hard to maximize the reward $r(S_t, \mu)$, leading to an unstable topology.

Communication Inefficiency: To compute the outcomes X_t for all arms, existing frameworks use the magnitudes of aggregated weights and gradients as importance scores for active and inactive

[2] In this paper, $\langle \cdot \rangle$ is the cardinal number of the set.

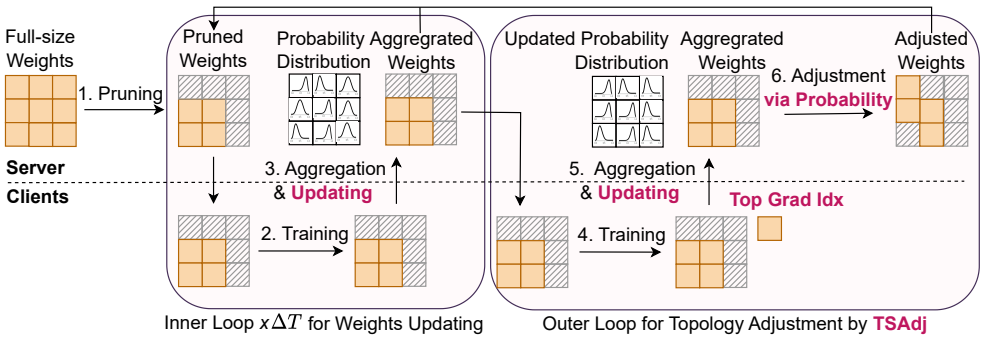

Figure 2: The overview of the FedRTS, which integrates TSAdj and utilizes the two-loop updating to develop a robust sparse model.

arms, respectively. As a result, clients must upload gradients for inactive weights, leading to a total communication cost comparable to that of a dense model, thus lacking communication efficiency. Although some approaches try to mitigate this by uploading only the top gradients [18, 19] or placing adjustments on the client side [1, 57], these greedy methods make the adjustment more unstable, leading to suboptimal performance.

3 Methodology

In this section, we present FedRTS, a novel federated pruning framework designed to develop robust model topologies, as illustrated in Fig. 2. We begin by detailing the TSAdj mechanism, which serves as the core component for searching robust sparse topologies. Next, we provide an overview of FedRTS that integrates TSAdj. Finally, we provide the theoretical regret upper bound of TSAdj.

3.1 Thompson Sampling-based Adjustment (TSAdj)

We find that the core source of the aforementioned challenges in existing methods lies in **deterministic decision strategies and myopic observations**. Therefore, we propose the Thompson Sampling-based Adjustment (TSAdj) mechanism, which includes a probabilistic decision strategy based on farsighted observations. Moreover, TSAdj effectively reduces communication overheads.

Compared to directly using myopic outcomes X_t to make deterministic decisions, our proposed TSAdj maintains individual posterior probability distributions P for each link in the model topology m during training. The higher feedback a link i has gained from previous observations, the closer the expectation of its probability $E[P_i]$ approaches 1, indicating a higher likelihood of being selected. In the outer loop for topology adjustment, the server samples random variables ξ, where $\xi_i \sim P_i$, and selects an action S_t as follows:

$$S_t = \{i \in \text{Top}(\xi, K)\}, \tag{2}$$

where $\text{Top}(\xi, K)$ are the indices of top-K elements in ξ. The action S_t is determined by the posterior distributions P, rather than deterministically based on outcomes, thereby providing a more stable topology with lower variance [51] compared to existing approaches.

Unlike existing approaches that develop the outcomes using immediate and myopic data in the outer loop, TSAdj derives much more comprehensive outcomes X_t from both the inner and outer loops. After performing action S_t, the outcome X_t is observed in the end of loop t, and is formulated as a fusion of individual outcomes X_t^n and aggregated outcomes X_t^{agg}:

$$X_t = \gamma X_t^{agg} + (1-\gamma) \sum_{n \in C_t} p_n X_t^n, \tag{3}$$

where γ is the trade-off ratio, C_t is the set of clients that participate FL in loop t. This fusion mechanism mitigates biases that arise from unstable aggregated data.

We consider the semi-outcomes for X_t^{agg} and X_t^n in the inner loop, *i.e.,* only the arms $i \in S_t$ have the outcomes. We define a constant $\kappa < K$, which denotes the number of core links (arms) in the

model topology, and during the adjustment, we expect that $K - \kappa$ less important links are pruned, while $K - \kappa$ candidate links are reactivated. Following previous methods, we use the magnitude of weights to get the outcomes for active arms $i \in S_t$. Therefore, at the end of inner loop, we have

$$X_{t,i}^{agg} = h(i, |W_{t+1}^{agg}|, \kappa), \quad X_{t,i}^{n} = h(i, |W_{t+1}^{n}|, \kappa), \quad i \in S_t, \tag{4}$$

where W_{t+1}^{agg} and W_{t+1}^{n} represent the aggregated weights and client n's weights, respectively. Here we use $t+1$ instead of t because the local training is finished. $h(\cdot)$ is a discriminative function defined as $h(i, x, \kappa) = \mathbf{1}_{i \in \text{Top}(x,\kappa)}$. In the inner loop, for arms $i \notin S_t$, both the aggregated outcomes $X_{t,i}^{agg}$ and the individual outcomes $X_{t,i}^{n}$ are set to $None$.

We consider the full outcomes of X_t^{agg} and X_t^n for outer loop. Beyond the outcomes of active arms in Eq. 4, we estimate the outcomes for inactive arms $i \notin S_t$ using the magnitudes of their gradients,

$$X_{t,i}^{agg} = 0.5, \quad X_{t,i}^{n} = h(i, |G_{t+1}^{n}|, K - \kappa), \quad i \notin S_t, \tag{5}$$

where the G_{t+1}^n denotes the gradients of the weights W_{t+1}^n. Here, $K - \kappa$ is used to treat some inactivated arms as candidates for reactivation. We do not use the aggregated gradients G_{t+1}^{agg} to compute $X_{t,i}^{agg}$ due to the high variance. Notably, to achieve Eq. 5, the clients only need to upload the Top indices of gradients magnitude, $\mathcal{I}_{t+1}^n = \text{Top}(|G_{t+1}^n|, K - \kappa)$, rather than the full gradients, significantly reducing the communication overhead.

To construct a posterior probability distribution, TSAdj updates the distribution P based on outcomes X_t. Following the Thompson Sampling framework, each P_i is modeled as a $Beta$ distribution with factors α_i, β_i, representing the likelihood of selecting an arm into S_t: a higher α_i shifts $\mathbb{E}[P_i]$ towards 1, whereas a higher β_i shifts it towards 0. The (α_i, β_i) are updated via Bayes' rule, leveraging its conjugacy properties and using the outcomes $X_{t,i}$ as follows:

$$(\alpha_i, \beta_i) \leftarrow (\alpha_i + \lambda X_{t,i}, \beta_i + \lambda(1 - X_{t,i})), \quad X_{t,i} \neq None, \tag{6}$$

where λ is a scaling factor that determines the influence of outcomes. Over time, the distribution gradually converges towards the true distribution, aligning with the observed outcomes. The overflow of TSAdj is shown in Algorithm 1.

Algorithm 1 TSAdj

Input: Participated clients C_t, uploaded sparse weights $\{W_{t+1}^n\}_{n \in C_t}$ and top gradients' indices $\{\mathcal{I}_{t+1}^n\}_{n \in C_t}$, trade-off ratio γ, scaling factor λ, global distribution factors (α, β), number of selected arms K, sparse model topology m_t.
Output: Updated sparse weights W_{t+1} and topology m_{t+1}
$W_{t+1}^{agg} \leftarrow \sum_{n \in C_t} p_n W_{t+1}^n$
$X_t^{agg}, \{X_t^n\}_{n \in C_t} \leftarrow$ calculate with W_{t+1}^{agg}, $\{W_{t+1}^n\}_{n \in C_t}$ and $\{\mathcal{I}_{t+1}^n\}_{n \in C_t}$ by Eq. 4 and Eq. 5
$X_t \leftarrow \gamma X_t^{agg} + (1 - \gamma) \sum_{n \in C_t} p_n X_t^n$
$(\alpha_i, \beta_i) \leftarrow (\alpha_i + \lambda X_{t,i}, \beta_i + \lambda(1 - X_{t,i}))$, for $i = 1, 2, \ldots, \langle X_t \rangle$ and $X_{t,i} \neq None$
$S_{t+1} \leftarrow \{i | \xi_i \in \text{Top}(\xi, K)\}, \quad \xi = \{\xi_i | \xi_i \sim Beta(\alpha_i, \beta_i)\}$ // Select new action
$m_{t+1} \leftarrow \mathbf{1}_{\{i \in S_{t+1}\}}$ // Adjust topology based on selected action
$W_{t+1} \leftarrow W_{t+1}^{agg} \odot m_{t+1}$

In summary, TSAdj leverages the posterior probability distributions P to probabilistically guide topology adjustments and uses comprehensive outcomes X_t to update the distributions, which mitigates instability caused by myopic adjustments, enabling smoother convergence and developing a more reliable topology. Moreover, TSAdj requires the clients to upload only the most critical gradient information, thereby reducing the communication overhead.

3.2 Proposed FedRTS framework

FedRTS is a novel federated pruning framework designed to address the limitations of existing dynamic sparse training methods by integrating the Thompson Sampling-based Adjustment (TSAdj) mechanism, as illustrated in Fig. 2. The framework begins with an initialized model weight and sparse topology sampled from Beta distributions P. FedRTS employs a two-loop training process to iteratively update model weights and refine the sparse topology.

In the inner loop, the model topology remains fixed while the server updates the model weights and distribution P based on semi-outcomes. In the outer loop, the server collects the full outcomes by requiring clients to upload the top gradient indices of inactivated weights. Using this information, the server applies TSAdj to make robust topology adjustments, and the updated topology is then distributed to clients. FedRTS continues this iterative process until model convergence. The details of FedRTS are shown in Algorithm 2 and Sec. B.2.

By integrating TSAdj, FedRTS introduces a novel approach to sparse topology adjustment. Leveraging Thompson Sampling, it mitigates the instability caused by unstable aggressive topology, ensuring smoother convergence through probabilistically guided adjustments based on historical information. This adaptive adjustment strategy enhances the robustness of the sparse topology and improves overall model performance in FL tasks.

3.3 Overhead Analysis

The overhead of TSAdj arises from maintaining and sampling the Beta distribution on the server. In each round, the server performs the following operations with $\langle C_t \rangle$ sampled clients: weight averaging with a time complexity of $\mathcal{O}(\langle C_t \rangle \langle W \rangle)$, computing the outcome using Quicksort with an expected time complexity of $\mathcal{O}(\langle C_t \rangle \langle W \rangle)$, element-wise Beta distribution updates in $\mathcal{O}(\langle W \rangle)$, and arms selection in $\mathcal{O}(\langle W \rangle)$. The overall complexity thus simplifies to $\mathcal{O}(\langle C_t \rangle \langle W \rangle)$. Compared to the baseline complexities of FedMef, FedTiny, and PruneFL, which are also $\mathcal{O}(\langle C_t \rangle \langle W \rangle)$, TSAdj achieves the same computational order. This demonstrates that TSAdj introduces no significant additional computational burden on the server, even with large models.

Critically, TSAdj introduces no additional computations on client devices compared to the federated pruning baseline, as its overhead is entirely server-side. In cross-device federated learning, the server is typically assumed to possess substantial computational resources; therefore, TSAdj only introduces a slight overhead on the system.

3.4 Theoretical Analysis

For simplifications, we omit the inner loop of FedRTS to isolate the TSAdj mechanism and consider only semi-outcomes, *i.e.,* ignore the outcomes $X_{t,i}$ that $i \notin S_t$. We adopt the following assumptions:

Assumption 3.1. *(L-continuity) $\exists L \in \mathbb{R}, \forall S, \mu, \mu'$ satisfies $|r(S, \mu) - r(S, \mu')| \leq L||\mu - \mu'||_1$.*

Assumption 3.2. *(Independent importance score) The importance scores used for pruning are treated as mutually independent.*

Assumption 3.3. *(Mean-field Approximation) when $\{x_i\}$ are mutually independent, the discriminative function $h(\cdot)$ admits a mean-field approximation: $h(i, x, \kappa) = \mathbf{1}_{i \in \{x, \kappa\}} \approx \mathbf{1}_{x_i \geq \sigma}$, where the threshold σ is determined self-consistently via mean-field decoupling (independent of individual x_i) and satisfies $\langle x \rangle - \sum_i^{\langle x \rangle} \mathrm{CDF}_i(\sigma) = \kappa$, where CDF_i is the cumulative distribution function of x_i.*

Assumption 3.1 is standard in bandit optimization work [26]. Assumption 3.2 is implicitly or explicitly adopted in pruning methods [46, 12, 52]. Assumption 3.3 holds for large scale $\langle x \rangle$ (>100k) [53, 54, 35].

While weight magnitudes are not strictly theoretically independent when used as importance scores, both empirical and theoretical evidence supports simplification in Assumption 3.2: (1) Recent work demonstrates that trained weights in large networks exhibit asymptotic independence [54, 53], and (2) this assumption is well-established in the pruning work [12, 46], where weight dependencies are typically disregarded during selection. While other importance scores (*e.g.,* Fisher information [52]) might better satisfy independence, they are computationally prohibitive in FL. Thus, we follow prior works [24, 19] to apply weight magnitude pruning, acknowledging a minor discrepancy between theory and practice. Under assumptions 3.2 and 3.3, outcomes $X_{t,i}$ simplify to:

$$X_{t,i} = \gamma \mathbf{1}_{|W_{t,i}| \geq \sigma} + (1 - \gamma) \sum_n^{\mathcal{N}} p_n \mathbf{1}_{|W_{t,i}^n| \geq \sigma}, \quad i \in S_t, \tag{7}$$

which is mutually independent.

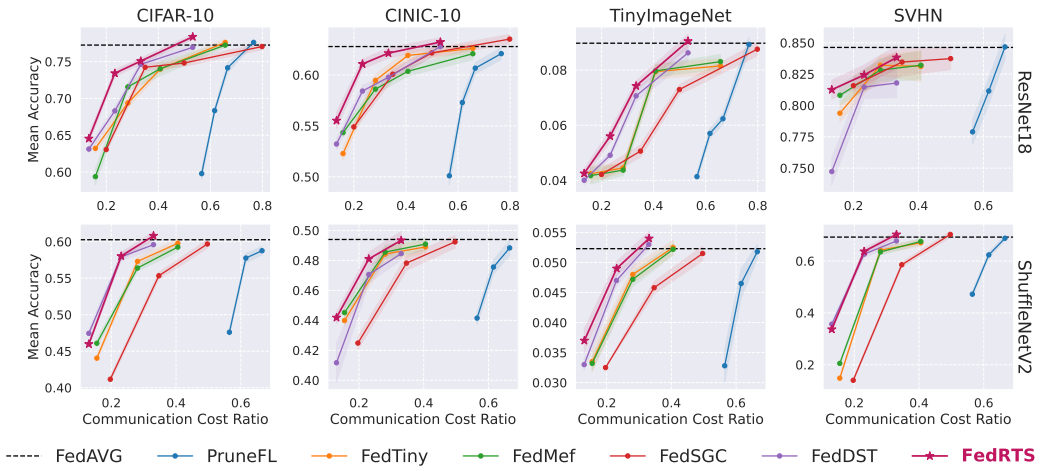

Figure 3: Testing accuracy of FedRTS and different federated pruning baselines on the four CV datasets with different densities, where the ratio is communication cost relative to dense FedAVG.

To analyze the worst-case performance of TSAdj, we define the cumulative regret $Reg(T)$ as:

$$Reg(T) = \mathbb{E}\left[\sum_{t=1}^{T} \Delta_{S_t}\right], \tag{8}$$

where $\Delta_{S_t} = \max\{r(S^*, \mu) - r(S_t, \mu), 0\}$ and S^* is the greedy optimal action based on μ, which is obtained by Algorithm 3. The maximum reward gap is set as $\Delta_{\max} = \max_S \Delta_S$. The maximum reward gap of actions containing arm s is $\Delta_s^{\max} = \max_{S:s\in S} \Delta_S$. We define S^* and S_t are sequences, *i.e.*, $S^* = (s_1^*, \ldots, s_K^*)$, $S_t = (s_{t,1}, \ldots, s_{t,K})$, and $s_k^* \in S^*$ is the k-th element added to S^*. We denote $S_k^* = (s_1^*, \ldots, s_k^*)$ and $S_{t,k} = (s_{t,1}, \ldots, s_{t,k})$. For any arm s, define the marginal reward gap $\Delta_{s,k} = r(S_k^*, \mu) - r(S_{k-1}^* \cup \{s\}, \mu)$. Inspired by previous work [5, 26], we provide a regret upper bound for TSAdj.

Theorem 3.4. *(Upper Bound) Under assumptions 3.2, 3.3 and 3.1 and with outcomes X_t defined in Eq. 7, the regret $Reg(T)$ of TSAdj can be upper bounded by:*

$$\begin{aligned} Reg(T) &\leq \sum_{s\neq s_1^*} \max_{k:s\notin S_k^*} \frac{6\Delta_s^{\max}\log T}{(\Delta_{s,k} - 2LK\epsilon)^2} + \left(2K + 4\langle W\rangle + \frac{KU + 8K\epsilon^4}{\epsilon^6}\right)\Delta_{max} \\ &= O\left(\sum_{s\neq s_1^*} \max_{k:s\notin S_k^*} \frac{\Delta_{max} L^2 \log T}{\Delta_{s,k}^2} + \Delta_{max}\right). \end{aligned} \tag{9}$$

for any ϵ such that $\forall s \neq s_1^*$ and $s \notin S_k^*, \Delta_{s,k} > 2LK\epsilon$, where U is a universal constant.

The proof is provided in Sec. C.2. **Crucially, our theorem hold for any pruning method satisfying Assumption 3.2 and 3.3**, including future advancements in efficient independent-scoring techniques.

4 Evaluation

To validate the efficacy of FedRTS, we conduct comprehensive experiments in this section. Notably, we evaluate each experiment five times, reporting the mean results along with the standard errors. In the figures, the shaded area represents the standard errors. In the table, the best results are highlighted in **purple** color, while the second-best results are marked in **blue** color.

4.1 Experiment Setup

We provide an overview of the experimental setup in this section, with additional details in Sec. F.

Target Density	Method	PPL ↓	Avg. Acc ↑	Comm. Cost ↓
1	FedAVG	20.56	0.4387	260.41MB
50%	FedDST	**20.10**	0.4261	**138.30 MB**
	FedSGC	26.13	0.4110	207.45 MB
	FedMef	20.61	**0.4352**	165.96 MB
	FedRTS	**18.54**	**0.4422**	**138.84 MB**
30%	FedDST	24.46	0.4263	**86.26 MB**
	FedSGC	32.39	0.3939	129.39 MB
	FedMef	**21.00**	**0.4304**	103.51 MB
	FedRTS	**18.56**	**0.4405**	**86.44 MB**
20%	FedDST	26.49	0.4219	**60.22 MB**
	FedSGC	43.00	0.3550	90.33 MB
	FedMef	**21.53**	**0.4293**	72.26 MB
	FedRTS	**19.93**	**0.4333**	**60.34 MB**

Table 1: Performance comparison on TinyStories with GPT-2-32M.

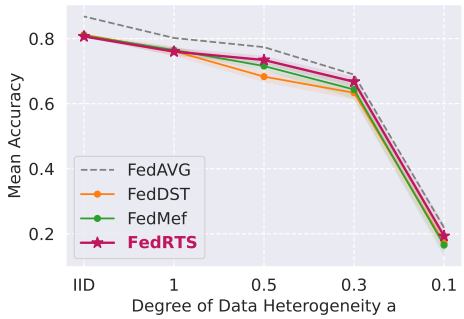

Figure 4: Accuracy on CIFAR-10 under various degrees of data heterogeneity.

Datasets and Models. Following previous work [18, 19, 1, 24] on federated dynamic pruning, we conduct experiments on CV tasks using two lightweight models, ResNet18 [13] and ShuffleNetV2 [66], across four well-known image classification datasets: CIFAR-10 [27], CINIC-10 [7], TinyImageNet [8] and SVHN [43]. For NLP tasks, we use the GPT-2-32M model on the TinyStories dataset [10], a language understanding benchmark designed for small language models.

Federated Learning Settings. Following the settings used in previous frameworks [18, 19, 1], we conduct $T_{max} = 500$ communication rounds with 5 local training epochs. Batch sizes are set to 64 for CV tasks and 16 for NLP tasks. In each round, 10 clients are randomly selected from a total of $\mathcal{N} = 100$ clients. To generate data heterogeneity on the client, we utilize the Dirichlet distribution with factor a. The types of heterogeneity for CV and NLP tasks are different. CV tasks apply label imbalance with $a = 0.5$ by default. Unlabeled NLP tasks set data quantity imbalance, with $a = 5$ by default. We perform an outer loop for topology adjustment every $\Delta T = 10$ inner loops until the total number of loops t reaches $T_{end} = 300$. The optimizer used is SGD, with a learning rate of $\eta = 0.01$.

Baseline Settings. We select the following SOTA federated pruning frameworks as baselines: PruneFL [24], FedTiny [18], FedDST [1], FedSGC [57], and FedMef [19]. These methods incorporate dynamic pruning in FL and achieve SOTA performance across various settings, as detailed in Sec. F.2.

Implementation Details. Following previous work [18, 19], we exclude bias terms, normalization layers, and the output layer from pruning while ensuring the overall density $d \leq d'$. Layer-wise sparsity follows the Erdos-Renyi-Kernel distribution [11]. We set the scaling factor $\lambda = 10$ to match the number of selected clients per loop, ensuring that the averaged information maintains individual-level influence. The trade-off ratio is set as $\gamma = 0.5$ to assign equal importance to the individual and aggregated model. We set the layer-wise $\kappa^l = K^l - \frac{\alpha_{adj}}{2}(1 + \cos(\frac{t\pi}{T_{end}}))K^l$, where K^l is the number of active weights at l-th layer and α_{adj} is 0.4 by default, is adopted directly from prior works [19, 11]. Training FLOPs and communication costs are calculated as described in Sec. F.4.

4.2 Performance Evaluation

To show the effectiveness of FedRTS, we conduct experiments on CV and NLP tasks under various target density levels. Broader experiments including efficiency analysis are in Sec. E.2 and E.3.

Computer Vision Tasks. In the CV tasks, we conduct experiments using the ResNet18 and ShuffleNetV2 models, testing four target density levels: 50%, 30%, 20%, and 10% on CIFAR-10, CINIC-10, TinyImageNet, and SVHN datasets, reporting the accuracy and communication costs ratio per outer loop. In some cases, FedRTS achieves similar accuracy to FedAVG at the 30% density ratio; therefore, results for the 50% density ratio are omitted. As shown in Fig. 3, FedRTS consistently outperforms all baseline methods in most scenarios in terms of both accuracy and communication efficiency. For instance, in the experiment with ResNet18 in CIFAR-10, FedRTS achieves a 5.1% improvement in accuracy over the best baseline methods while maintaining a similar communication cost. These gains are attributed to the robust adjustment, stable topology, and communication efficiency provided by TSAdj in FedRTS.

An evident trend is the superior accuracy achieved by ResNet18 compared to ShuffleNetV2, due to ResNet18's larger parameter space and higher representational capacity. Furthermore, FedDST and

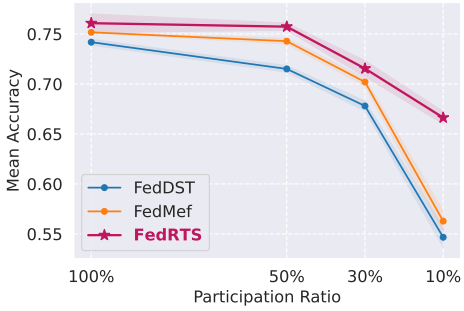

Figure 5: Accuracy on CIFAR-10 with various participation ratios of clients.

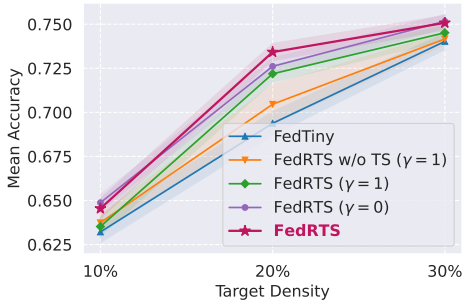

Figure 6: Accuracy on CIFAR-10 for FedRTS and various variants.

FedMef consistently outperform other baseline methods. Based on these empirical observations, we select ResNet18 as the default CV model and identify FedDST and FedMef as the primary reference for subsequent experiments.

Natural Language Processing Tasks. To evaluate the effectiveness of FedRTS across different domains, we conduct experiments on NLP tasks using the TinyStories dataset with the GPT-2-32M model. We test three target density levels: 50%, 30%, and 20%. Perplexity (PPL), which assesses model performance by considering the entire probability distribution in generated text, serves as the primary evaluation metric. As shown in Tab. 1, FedRTS outperforms all baselines, demonstrating its superiority in the NLP domain. Notably, FedRTS achieves better performance in perplexity than in accuracy-based evaluations, even surpassing full-size FedAVG at low-density levels. This improvement may stem from its probabilistic adjustment strategy, which enhances flexibility and adaptability during optimization. Unlike deterministic approaches, probabilistic adjustment enables FedRTS to prioritize critical parameters effectively, improving its ability to adapt to complex probability distributions. This is particularly beneficial for NLP tasks, where capturing intricate text patterns is essential, leading to lower perplexity and stronger performance in low-resource settings.

4.3 Robustness Analysis

To demonstrate the robustness of FedRTS in the face of heterogeneous data distributions and partial client participation, we conduct experiments with different degrees of data heterogeneity and client availability in this section. Further experiments on the impact of the hyperparameters γ, λ, κ, and ΔT are provided in Sec. E.1.

The Impact of Data Heterogeneity. Data heterogeneity, or non-independent and identically distributed (non-IID) data, is a critical factor affecting performance in FL. To demonstrate the robustness of FedRTS, we conduct experiments with varying degrees of data heterogeneity on the CIFAR-10 and TinyStories datasets. By adjusting the Dirichlet distribution factor a, we control the degree of data heterogeneity, where a lower a indicates higher heterogeneity. As shown in Fig. 4 and Fig. 14, FedRTS outperforms all baselines, particularly in highly heterogeneous settings, demonstrating its effectiveness in handling non-IID data. This improvement is primarily attributed to the stable topology produced by TSAdj in FedRTS.

The Impact of Client Availability. Client availability is another significant challenge in FL. To evaluate the robustness of FedRTS under different levels of client availability, we conduct experiments with participation ratios of 100%, 50%, 30%, 10% using a total of 10 clients. To mitigate slow convergence at lower participation levels, we adopt a high learning rate of 1 with SGD, as Adam exhibited instability with NaN values in our experiments. As shown in Fig. 5, FedRTS consistently outperforms all baselines, particularly at low participation ratios, demonstrating its adaptability and robustness in limited-client scenarios.

4.4 Ablation Study

To analyze the individual contributions of FedRTS in addressing greedy and unstable topology adjustment problems, we conduct an ablation study on the CIFAR-10 dataset with various target

density levels. We consider the benefits of FedRTS to come from three aspects: probabilistic decision, farsighted information, and stability from the fusion mechanism in Eq. 3. To evaluate these aspects independently, we consider the following variants of FedRTS: (1) *FedRTS ($\gamma = 0$)*: This variant does not use the aggregated information to update the probability distribution. (2) *FedRTS ($\gamma = 1$)*: This variant only uses the aggregated information to update the probability distribution, which disables the fusion mechanism to reduce stability. (3) *FedRTS without TS ($\gamma = 1$)*: This variant fuses historical information via a momentum mechanism similar to FedAdam [49] and uses deterministic adjustment to obtain the new topology, disabling both probabilistic decision and fusion mechanism. (4) *FedTiny*: This serves as a baseline to evaluate the overall effectiveness of FedRTS, disabling all three aspects.

The results in Fig. 6 demonstrate the effectiveness of FedRTS. First, the performance of *FedRTS* and *FedRTS ($\gamma = 0$)* is similar but better than *FedRTS ($\gamma = 1$)*, demonstrating that aggregated information suffers from instability. A fusion mechanism including individual information can make the adjusted topology more stable and achieve better performance. Second, without deterministic adjustment, *FedRTS w/o TS ($\gamma = 1$)* performs much worse than *FedRTS ($\gamma = 1$)*, indicating the benefits of probabilistic adjustment. Third, *FedRTS w/o TS ($\gamma = 1$)* is better than *FedTiny*, showing the effectiveness of farsighted information. Thus, the ablation study demonstrates FedRTS's effectiveness in terms of probabilistic decision, farsighted information, and stability from the fusion mechanism.

5 Limitation

While FedRTS with TSAdj effectively addresses key challenges: greedy adjustments, unstable topologies, and communication inefficiency in federated dynamic pruning, our approach shares a common limitation with prior magnitude-based pruning methods [12, 46]. The theoretical analysis relies on Assumption 3.2, which presumes independence among importance scores. In practice, weight magnitudes exhibit only asymptotic independence [53], creating a minor but acknowledged gap between theory and empirical results.

We considered alternative importance metrics like Fisher information [52] that might better satisfy the independence requirement. However, their computational complexity makes them impractical for federated settings with resource-constrained clients. Following established practice in the pruning literature, we therefore adopt magnitude-based pruning while transparently acknowledging this theoretical-empirical discrepancy, a compromise that reflects the inherent trade-offs in practical federated learning systems.

While the mean-field approximation in Assumption 3.3 provides theoretical grounding for TSAdj, its validity weakens for extremely small models (<100K parameters) where finite-size effects become significant [53]. However, at such small scales, the benefits of pruning become marginal, as these models can typically be trained directly on-device without compression. Thus, this limitation has minimal practical impact on FedRTS.

The FedRTS also presents the following two limitations. Fairness and bias: The method dynamically adjusts model structure per global feedback. In highly heterogeneous settings, this could unintentionally bias the model toward data-rich or majority-class clients. Security/privacy considerations: While the method reduces communication cost, sharing top gradient indices might still leak sensitive information.

6 Conclusion

In this paper, we analyze federated pruning from the perspective of combinatorial multi-armed bandits and identify key limitations in existing methods, including greedy adjustments, unstable topologies, and communication inefficiencies. To address these challenges, we propose Federated Robust Pruning via Combinatorial Thompson Sampling (FedRTS), a novel framework for developing robust sparse models. The core component of FedRTS, the Thompson Sampling-based Adjustment (TSAdj) module, enhances robustness and performance by making probabilistic decisions based on stable, farsighted information rather than deterministic decisions driven by unstable, myopic observations. We provide a theoretical regret upper bound for TSAdj and conduct extensive experiments, demonstrating that FedRTS achieves state-of-the-art performance in both computer vision and natural language processing tasks.

Acknowledgments and Disclosure of Funding

This paper is partially supported by Hong Kong Research Grants Council (RGC) grant C1042-23GF and grant #11203523 and Hong Kong Innovation and Technology Fund (ITF) grant MHP/061/23 and grant MHP/034/22.

References

[1] Sameer Bibikar, Haris Vikalo, Zhangyang Wang, and Xiaohan Chen. Federated dynamic sparse training: Computing less, communicating less, yet learning better. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 6080–6088, 2022.

[2] Ning Chen, Tie Qiu, Weisheng Si, and Dapeng Oliver Wu. Daimo: motif density enhances topology robustness for highly dynamic scale-free iot. *IEEE Transactions on Mobile Computing*, 2024.

[3] Ning Chen, Tie Qiu, Xiaobo Zhou, Songwei Zhang, Weisheng Si, and Dapeng Oliver Wu. A distributed co-evolutionary optimization method with motif for large-scale iot robustness. *IEEE/ACM Transactions on Networking*, 32(5):4085–4098, 2024.

[4] Ning Chen, Songwei Zhang, Xiaobo Zhou, Song Cao, and Tie Qiu. Fast robustness enhancement for dynamic iiot topology with adaptive bayesian learning. *IEEE Transactions on Mobile Computing*, 2025.

[5] Wei Chen, Yajun Wang, and Yang Yuan. Combinatorial multi-armed bandit: General framework and applications. In *International conference on machine learning*, pages 151–159. PMLR, 2013.

[6] Wei Chen, Yajun Wang, Yang Yuan, and Qinshi Wang. Combinatorial multi-armed bandit and its extension to probabilistically triggered arms. *Journal of Machine Learning Research*, 17 (50):1–33, 2016.

[7] L. N. Darlow, E. J. Crowley, A. Antoniou, and A. J. Storkey. Cinic-10 is not imagenet or cifar-10. *arXiv preprint arXiv:1810.03505*, 2018.

[8] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009.

[9] Tim Dettmers and Luke Zettlemoyer. Sparse networks from scratch: Faster training without losing performance. *arXiv preprint arXiv:1907.04840*, 2019.

[10] Ronen Eldan and Yuanzhi Li. Tinystories: How small can language models be and still speak coherent english? *arXiv preprint arXiv:2305.07759*, 2023.

[11] Utku Evci, Trevor Gale, Jacob Menick, Pablo Samuel Castro, and Erich Elsen. Rigging the lottery: Making all tickets winners. In *International Conference on Machine Learning*, pages 2943–2952. PMLR, 2020.

[12] Song Han, Huizi Mao, and William J Dally. Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding. *arXiv preprint arXiv:1510.00149*, 2015.

[13] 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. IEEE, 2016.

[14] Yihui He, Xiangyu Zhang, and Jian Sun. Channel pruning for accelerating very deep neural networks. In *Proceedings of the IEEE international conference on computer vision*, pages 1389–1397, 2017.

[15] Robert Hönig, Yiren Zhao, and Robert Mullins. Dadaquant: Doubly-adaptive quantization for communication-efficient federated learning. In *International Conference on Machine Learning*, pages 8852–8866. PMLR, 2022.

[16] Juntao Hu, Zhengjie Yang, Peng Wang, Guanyi Zhao, Hong Huang, Zhimin Zong, and Dapeng Oliver Wu. Federated learning for medical image analysis: Privacy-preserving paradigms and clinical challenges. *Transactions on Artificial Intelligence*, 1(1):153–169, 2025.

[17] Hong Huang and Dapeng Wu. Quaff: Quantized parameter-efficient fine-tuning under outlier spatial stability hypothesis. *arXiv preprint arXiv:2505.14742*, 2025.

[18] Hong Huang, Lan Zhang, Chaoyue Sun, Ruogu Fang, Xiaoyong Yuan, and Dapeng Wu. Distributed pruning towards tiny neural networks in federated learning. In *2023 IEEE 43rd International Conference on Distributed Computing Systems (ICDCS)*, pages 190–201. IEEE, 2023.

[19] Hong Huang, Weiming Zhuang, Chen Chen, and Lingjuan Lyu. Fedmef: Towards memory-efficient federated dynamic pruning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 27548–27557, 2024.

[20] Hong Huang, Decheng Wu, Rui Cen, Guanghua Yu, Zonghang Li, Kai Liu, Jianchen Zhu, Peng Chen, Xue Liu, and Dapeng Wu. Tequila: Trapping-free ternary quantization for large language models. *arXiv preprint arXiv:2509.23809*, 2025.

[21] Steven A Janowsky. Pruning versus clipping in neural networks. *Physical Review A*, 39(12): 6600, 1989.

[22] Siddhant Jayakumar, Razvan Pascanu, Jack Rae, Simon Osindero, and Erich Elsen. Top-kast: Top-k always sparse training. *Advances in Neural Information Processing Systems*, 33: 20744–20754, 2020.

[23] Divyansh Jhunjhunwala, Advait Gadhikar, Gauri Joshi, and Yonina C Eldar. Adaptive quantization of model updates for communication-efficient federated learning. In *ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 3110–3114. IEEE, 2021.

[24] Yuang Jiang, Shiqiang Wang, Victor Valls, Bong Jun Ko, Wei-Han Lee, Kin K Leung, and Leandros Tassiulas. Model pruning enables efficient federated learning on edge devices. *IEEE Transactions on Neural Networks and Learning Systems*, 34(12):10374–10386, 2022.

[25] Peter Kairouz, H Brendan McMahan, Brendan Avent, Aurélien Bellet, Mehdi Bennis, Arjun Nitin Bhagoji, Kallista Bonawitz, Zachary Charles, Graham Cormode, Rachel Cummings, et al. Advances and open problems in federated learning. *Foundations and trends® in machine learning*, 14(1–2):1–210, 2021.

[26] Fang Kong, Yueran Yang, Wei Chen, and Shuai Li. The hardness analysis of thompson sampling for combinatorial semi-bandits with greedy oracle. *Advances in Neural Information Processing Systems*, 34:26701–26713, 2021.

[27] A. Krizhevsky, G. Hinton, and et al. Learning multiple layers of features from tiny images. In *Proceedings of the International Conference on Machine Learning*. PMLR, 2009.

[28] Yann LeCun, John Denker, and Sara Solla. Optimal brain damage. *Advances in neural information processing systems*, 2, 1989.

[29] Namhoon Lee, Thalaiyasingam Ajanthan, and Philip Torr. Snip: Single-shot network pruning based on connection sensitivity. In *International Conference on Learning Representations*, 2018.

[30] Xiang Li, Kaixuan Huang, Wenhao Yang, Shusen Wang, and Zhihua Zhang. On the convergence of fedavg on non-iid data. *arXiv preprint arXiv:1907.02189*, 2019.

[31] Zonghang Li, Tao Li, Wenjiao Feng, Rongxing Xiao, Jianshu She, Hong Huang, Mohsen Guizani, Hongfang Yu, Qirong Ho, Wei Xiang, and Steve Liu. Prima.cpp: Fast 30-70b llm inference on heterogeneous and low-resource home clusters, 2025. URL `https://arxiv.org/abs/2504.08791`.

[32] Zhuang Liu, Jianguo Li, Zhiqiang Shen, Gao Huang, Shoumeng Yan, and Changshui Zhang. Learning efficient convolutional networks through network slimming. In *Proceedings of the IEEE international conference on computer vision*, pages 2736–2744, 2017.

[33] Xiaolong Ma, Minghai Qin, Fei Sun, Zejiang Hou, Kun Yuan, Yi Xu, Yanzhi Wang, Yen-Kuang Chen, Rong Jin, and Yuan Xie. Effective model sparsification by scheduled grow-and-prune methods. *arXiv preprint arXiv:2106.09857*, 2021.

[34] 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.

[35] Song Mei, Theodor Misiakiewicz, and Andrea Montanari. Mean-field theory of two-layers neural networks: dimension-free bounds and kernel limit. In *Conference on learning theory*, pages 2388–2464. PMLR, 2019.

[36] Fanxu Meng, Hao Cheng, Ke Li, Huixiang Luo, Xiaowei Guo, Guangming Lu, and Xing Sun. Pruning filter in filter. *Advances in Neural Information Processing Systems*, 33:17629–17640, 2020.

[37] Decebal Constantin Mocanu, Elena Mocanu, Peter Stone, Phuong H Nguyen, Madeleine Gibescu, and Antonio Liotta. Scalable training of artificial neural networks with adaptive sparse connectivity inspired by network science. *Nature communications*, 9(1):1–12, 2018.

[38] P Molchanov, S Tyree, T Karras, T Aila, and J Kautz. Pruning convolutional neural networks for resource efficient inference. In *5th International Conference on Learning Representations, ICLR 2017-Conference Track Proceedings*, 2019.

[39] Pavlo Molchanov, Stephen Tyree, Tero Karras, Timo Aila, and Jan Kautz. Pruning convolutional neural networks for resource efficient inference. *arXiv preprint arXiv:1611.06440*, 2016.

[40] Pavlo Molchanov, Arun Mallya, Stephen Tyree, Iuri Frosio, and Jan Kautz. Importance estimation for neural network pruning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 11264–11272, 2019.

[41] Michael C Mozer and Paul Smolensky. Skeletonization: A technique for trimming the fat from a network via relevance assessment. *Advances in neural information processing systems*, 1, 1988.

[42] Muhammad Tahir Munir, Muhammad Mustansar Saeed, Mahad Ali, Zafar Ayyub Qazi, and Ihsan Ayyub Qazi. Fedprune: Towards inclusive federated learning. *arXiv preprint arXiv:2110.14205*, 2021.

[43] Y. Netzer, T. Wang, A. Coates, A. Bissacco, B. Wu, and A. Y. Ng. Reading digits in natural images with unsupervised feature learning. In *Proceedings of the 2011 Neural Information Processing Systems (NIPS)*, 2011.

[44] Jeff M Phillips. Chernoff-hoeffding inequality and applications. *arXiv preprint arXiv:1209.6396*, 2012.

[45] Hongming Piao, Yichen Wu, Dapeng Wu, and Ying Wei. Federated continual learning via prompt-based dual knowledge transfer. In *Forty-first International Conference on Machine Learning*, 2024.

[46] Xin Qian and Diego Klabjan. A probabilistic approach to neural network pruning. In *International Conference on Machine Learning*, pages 8640–8649. PMLR, 2021.

[47] Xinchi Qiu, Javier Fernandez-Marques, Pedro PB Gusmao, Yan Gao, Titouan Parcollet, and Nicholas Donald Lane. Zerofl: Efficient on-device training for federated learning with local sparsity. *arXiv preprint arXiv:2208.02507*, 2022.

[48] Md Aamir Raihan and Tor Aamodt. Sparse weight activation training. *Advances in Neural Information Processing Systems*, 33:15625–15638, 2020.

[49] Sashank Reddi, Zachary Charles, Manzil Zaheer, Zachary Garrett, Keith Rush, Jakub Konečnỳ, Sanjiv Kumar, and H Brendan McMahan. Adaptive federated optimization. *arXiv preprint arXiv:2003.00295*, 2020.

[50] Amirhossein Reisizadeh, Aryan Mokhtari, Hamed Hassani, Ali Jadbabaie, and Ramtin Pedarsani. Fedpaq: A communication-efficient federated learning method with periodic averaging and quantization. In *International conference on artificial intelligence and statistics*, pages 2021–2031. PMLR, 2020.

[51] Aadirupa Saha and Branislav Kveton. Only pay for what is uncertain: Variance-adaptive thompson sampling. *arXiv preprint arXiv:2303.09033*, 2023.

[52] Sidak Pal Singh and Dan Alistarh. Woodfisher: Efficient second-order approximation for neural network compression. *Advances in Neural Information Processing Systems*, 33:18098–18109, 2020.

[53] Justin Sirignano and Konstantinos Spiliopoulos. Mean field analysis of neural networks: A central limit theorem. *Stochastic Processes and their Applications*, 130(3):1820–1852, 2020.

[54] Justin Sirignano and Konstantinos Spiliopoulos. Mean field analysis of neural networks: A law of large numbers. *SIAM Journal on Applied Mathematics*, 80(2):725–752, 2020.

[55] Aleksandrs Slivkins et al. Introduction to multi-armed bandits. *Foundations and Trends® in Machine Learning*, 12(1-2):1–286, 2019.

[56] Hidenori Tanaka, Daniel Kunin, Daniel L Yamins, and Surya Ganguli. Pruning neural networks without any data by iteratively conserving synaptic flow. *Advances in Neural Information Processing Systems*, 33:6377–6389, 2020.

[57] Chris Xing Tian, Yibing Liu, Haoliang Li, Ray CC Cheung, and Shiqi Wang. Gradient-congruity guided federated sparse training. *arXiv preprint arXiv:2405.01189*, 2024.

[58] Shuguang Wang, Qian Zhou, Kui Wu, Jinghuai Deng, Dapeng Wu, Wei-Bin Lee, and Jianping Wang. Interventional root cause analysis of failures in multi-sensor fusion perception systems. *Proceedings 2025 Network and Distributed System Security Symposium*, 2025. URL `https://api.semanticscholar.org/CorpusID:276862477`.

[59] Siwei Wang and Wei Chen. Thompson sampling for combinatorial semi-bandits. In *International Conference on Machine Learning*, pages 5114–5122. PMLR, 2018.

[60] Yun Wang, Junjie Hu, Junhui Hou, Chenghao Zhang, Renwei Yang, and Dapeng Oliver Wu. Rose: Robust self-supervised stereo matching under adverse weather conditions. *arXiv preprint arXiv:2509.19165*, 2025.

[61] Yun Wang, Kunhong Li, Longguang Wang, Junjie Hu, Dapeng Oliver Wu, and Yulan Guo. Adstereo: Efficient stereo matching with adaptive downsampling and disparity alignment. *IEEE Transactions on Image Processing*, 2025.

[62] Yun Wang, Jiahao Zheng, Chenghao Zhang, Zhanjie Zhang, Kunhong Li, Yongjian Zhang, and Junjie Hu. Dualnet: Robust self-supervised stereo matching with pseudo-label supervision. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pages 8178–8186, 2025.

[63] Yichen Wu, Hongming Piao, Long-Kai Huang, Renzhen Wang, Wanhua Li, Hanspeter Pfister, Deyu Meng, Kede Ma, and Ying Wei. Sd-lora: Scalable decoupled low-rank adaptation for class incremental learning. *arXiv preprint arXiv:2501.13198*, 2025.

[64] Zhengjie Yang, Sen Fu, Wei Bao, Dong Yuan, and Albert Y Zomaya. Hierarchical federated learning with momentum acceleration in multi-tier networks. *IEEE Transactions on Parallel and Distributed Systems*, 34(10):2629–2641, 2023.

[65] Zhengwu Yang and Han Zhang. Comparative analysis of structured pruning and unstructured pruning. In *International Conference on Frontier Computing*, pages 882–889. Springer, 2021.

[66] Xiangyu Zhang, Xinyu Zhou, Mengxiao Lin, and Jian Sun. Shufflenet: An extremely efficient convolutional neural network for mobile devices. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 6848–6856, 2018.

NeurIPS Paper Checklist

The checklist is designed to encourage best practices for responsible machine learning research, addressing issues of reproducibility, transparency, research ethics, and societal impact. Do not remove the checklist: **The papers not including the checklist will be desk rejected.** The checklist should follow the references and follow the (optional) supplemental material. The checklist does NOT count towards the page limit.

Please read the checklist guidelines carefully for information on how to answer these questions. For each question in the checklist:

- You should answer [Yes] , [No] , or [NA] .
- [NA] means either that the question is Not Applicable for that particular paper or the relevant information is Not Available.
- Please provide a short (1–2 sentence) justification right after your answer (even for NA).

The checklist answers are an integral part of your paper submission. They are visible to the reviewers, area chairs, senior area chairs, and ethics reviewers. You will be asked to also include it (after eventual revisions) with the final version of your paper, and its final version will be published with the paper.

The reviewers of your paper will be asked to use the checklist as one of the factors in their evaluation. While "[Yes] " is generally preferable to "[No] ", it is perfectly acceptable to answer "[No] " provided a proper justification is given (e.g., "error bars are not reported because it would be too computationally expensive" or "we were unable to find the license for the dataset we used"). In general, answering "[No] " or "[NA] " is not grounds for rejection. While the questions are phrased in a binary way, we acknowledge that the true answer is often more nuanced, so please just use your best judgment and write a justification to elaborate. All supporting evidence can appear either in the main paper or the supplemental material, provided in appendix. If you answer [Yes] to a question, in the justification please point to the section(s) where related material for the question can be found.

IMPORTANT, please:

- **Delete this instruction block, but keep the section heading "NeurIPS Paper Checklist"**,
- **Keep the checklist subsection headings, questions/answers and guidelines below.**
- **Do not modify the questions and only use the provided macros for your answers**.

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The identified challenges and key claims presented in both the abstract and introduction precisely align with the paper's core contributions and methodological scope.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

Justification: In Section 5

Guidelines:

- The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
- The authors are encouraged to create a separate "Limitations" section in their paper.
- The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
- The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
- The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
- The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
- If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
- While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [Yes]

   Justification: In the Section 3.4 and C.2

   Guidelines:

   - The answer NA means that the paper does not include theoretical results.
   - All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
   - All assumptions should be clearly stated or referenced in the statement of any theorems.
   - The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
   - Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
   - Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: All information, including hyperparameters, pseudo-code, is disclosed, and the source code is given.

   Guidelines:

- The answer NA means that the paper does not include experiments.
- If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
  (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes]

   Justification: All datasets are public, and the anonymized version of codes is given, with a detailed readme file.

   Guidelines:

   - The answer NA means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.

- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

   Answer: answerYes

   Justification: In the Section 4.1

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: All experiments were performed five times with different random seeds, and standard errors are reported.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
   - The assumptions made should be given (e.g., Normally distributed errors).
   - It should be clear whether the error bar is the standard deviation or the standard error of the mean.
   - It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
   - For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
   - If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: In the Section 4.1

   Guidelines:

   - The answer NA means that the paper does not include experiments.

- The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
- The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
- The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: Yes, all respects.

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: In the Section D

    Guidelines:

    - The answer NA means that there is no societal impact of the work performed.
    - If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
    - The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
    - The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
    - If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

    Answer: [NA]

    Justification: the paper poses no such risks.

Guidelines:

- The answer NA means that the paper poses no such risks.
- Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
- Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [NA]

    Justification: This paper does not use existing assets

    Guidelines:

    - The answer NA means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
    - For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
    - If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [NA]

    Justification: The paper does not release new assets.

    Guidelines:

    - The answer NA means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

Answer: [NA]

Justification: The paper does not involve crowdsourcing nor research with human subjects.

Guidelines:

- The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
- Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
- According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

Answer: [NA]

Justification: The paper does not involve crowdsourcing nor research with human subjects.

Guidelines:

- The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does not impact the core methodology, scientific rigorousness, or originality of the research, declaration is not required.

Answer: [NA]

Justification: The core method development in this research does not involve LLMs.

Guidelines:

- The answer NA means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy (`https://neurips.cc/Conferences/2025/LLM`) for what should or should not be described.

Appendices Contents

A Related Work

A.1 Neural Network Pruning

Neural network pruning [41, 28, 21] is a widely used technique for reducing the computational and memory footprint of neural networks by eliminating redundant parameters. Pruning methods can be categorized into structured pruning [36, 14, 32, 39], which removes entire structures like filters or layers, and unstructured pruning [12, 21], which targets individual weights. While structured pruning simplifies model deployment by maintaining regular architectures, it often suffers from significant performance degradation at higher sparsity levels (*e.g.,* beyond 10% [39]), limiting its compression potential. In contrast, unstructured pruning can achieve sparsity levels exceeding 50% with minimal accuracy loss, making it a more flexible approach. For this reason, our discussion focuses primarily on unstructured pruning.

Traditional pruning pipelines follow a train-prune-retrain paradigm: a dense model is first trained, less important weights are pruned based on some criterion (*e.g.*, magnitude [21, 12], gradients [41, 38], or Fisher information [52]), and the model is then fine-tuned to recover performance. However, this process is computationally expensive, as it requires training a dense model before pruning.

To avoid this overhead, recent work explores dynamic sparse training (DST) [37, 9, 11], where sparse topologies are optimized during training without ever densifying the model. While promising, existing DST methods rely on deterministic pruning-and-regrowth strategies, which can lead to high variance in model performance and inefficient exploration of sparse topologies, especially in federated learning scenarios with heterogeneous data and partial client participation. To mitigate these challenges, we propose TSAdj, a probabilistic adjustment mechanism that enables more stable and robust sparse topology evolution.

A.2 Federated Pruning

Federated Pruning is a technique that applies neural network pruning within the Federated Learning (FL) process. To avoid the requirements of dense training in traditional pruning methods, some previous methods determine the sparse model topology before federated training and do not adjust it during the FL process, such as SNIP [29] and Synflow [56]. This approach of pruning at initialization may overlook crucial data information, potentially leading to suboptimal model structures.

Dynamic sparse training methods, such as those employed by PruneFL [24], adjust the model during FL rounds. However, these methods require the entire model gradient to be sent to the server to ensure final performance, resulting in high communication costs. FedDST [1] and FedSGC [57] alleviate this issue by applying dynamic sparse training on clients. FedTiny [18] and FedMef [19] require clients only to upload the Top gradients. Despite these advancements, these methods adjust the model topology based on limited information from the currently selected clients, leading to issues of greedy adjustment and unstable topology. FedRTS addresses these challenges by applying TSAdj, significantly mitigating the problems associated with limited client information and enhancing the robustness of model adjustments by leveraging probabilistic distributions.

A.3 Combinatorial Multi-Arm Bandit

The Multi-Armed Bandit (MAB) problem [55] is a fundamental framework for decision-making under uncertainty, where an agent repeatedly selects among multiple actions ("arms") with initially unknown rewards to maximize cumulative gains. The Combinatorial Multi-Armed Bandit (CMAB) [5] extends this classic problem to scenarios where rewards depend on combinations of selected arms rather than individual choices.

Two primary approaches have emerged for solving CMAB problems: The first is the Combinatorial Upper Confidence Bound (CUCB) [5], which uses optimistic reward estimates to balance exploration and exploitation. The second is Combinatorial Thompson Sampling (CTS) [6, 59], which employs probabilistic sampling from posterior distributions to dynamically adjust its strategy.

While CUCB's deterministic nature makes it theoretically appealing for analysis, its inherent optimism [5] can lead to persistently selecting clearly suboptimal arm combinations. This behavior prevents stable convergence, making CUCB less suitable in dynamic pruning where consistent per-

formance is crucial, as will be discussed in detail in Sec. E.3.1. For this reason, we built our TSAdj method on the more adaptive CTS framework.

B Algorithm

B.1 TSAdj

In each loop t, the aggregated weights W_{t+1}^{agg} will be calculated in each loop when the server receives the weights W_{t+1}^{n} from each client $n \in C_t$, and TSAdj is invoked to adjust the sparse topology m_t only in the outer loop. The outcomes $X_{t,i}$ for link m_i are computed as described in Equation 3, which balances global and client-specific contributions. Specifically, outcomes $X_{t,i}^{agg}$ and $X_{t,i}^{n}$ are calculated based on Equations 4 on the server, using the aggregated weights W_{t+1} and the uploaded weights W_{t+1}^{n}, while $X_{t,i}^{n}$ will be updated in the outer loop using the indices of the top $K - \kappa$ gradients $\mathcal{I}_{t+1}^{n}$ from each client n based on Equation 5. Outcomes $X_{t,i}^{agg}$ and $X_{t,i}^{n}$ are combined to compute the final outcome $X_{t,i}$ for link m_i in Equation 3, where $\gamma \in [0, 1]$ controls the trade-off between global and client-specific contributions, and p_n reflects the relative weight of client n based on the size of its local dataset D_n. A larger γ emphasizes client-specific observations, while a smaller γ prioritizes global contributions.

The posterior probability distribution P for each link m_i in the topology m is modeled as a beta distribution parameterized by (α_i, β_i). At initialization, (α_i, β_i) are set to small positive values (*e.g.,* $\alpha_i = \beta_i = 1$) to represent a uniform prior, ensuring that all links have an equal probability of activation at the outset. The beta parameters (α_i, β_i) for each link m_i are then updated incorporating with the computed outcomes $X_{t,i}$, via Equation 6, where λ controls the trade-off between exploration and exploitation by adjusting the impact of the current outcomes on the beta distribution's update. A larger λ increases the influence of the current loop's outcomes $X_{t,i}$, accelerating the concentration of the Beta distribution towards a certain probability. Conversely, a smaller λ preserves uncertainty to encourage exploration. Over time, as α_i and β_i grow under the influence of λ and $X_{t,i}$, the Beta distribution becomes sharper and more concentrated, reflecting an increasingly certain probability of selecting m_i as an active link in the topology m.

In each loop t, the topology adjustment dynamically activates or deactivates links based on a probabilistic action S_t in Equation 2, where ξ is sampled from the updated posterior distribution P. For each link m_i, the probability ξ_i is independently sampled from its corresponding Beta distribution P_i. Links are activated if their probabilities ξ_i rank among the top K in magnitude. The TSAdj mechanism outputs the updated sparse topology m_{t+1} and the weights $W_{t+1} \odot m_{t+1}$, where $\odot$ denotes element-wise multiplication. The updated mask m_{t+1} represents the optimized topology for the next training loop $t + 1$.

B.2 FedRTS

The framework of FedRTS begins by initializing the global model weights W_0 and a sparse mask m_0 with a predefined sparsity level d'. The sparse mask m_0 determines the active parameters in the initial global model. Additionally, hyperparameters such as the total number of training loops T_{end} and the adjustment interval ΔT are set. These parameters control the frequency of topology adjustments and the overall training duration.

The training process is conducted over T_{max} loops. At the beginning of each training loop t, the server employs a random sampling mechanism to select a subset of clients C_t from the total $\mathcal{N}$ available clients for participation in the training process. This sampling process is conducted uniformly at random, ensuring that each client, indexed by n from $1, \ldots, \mathcal{N}$, has an equal probability of being chosen. This randomized partial participation mechanism not only avoids bias in client selection but also ensures scalability and reduces communication overhead, making it well-suited for large-scale distributed systems.

The server then broadcasts the global sparse weights W_t and the corresponding mask m_t to all participating clients C_t. Each selected client $n \in C_t$ trains the received model (W_t, m_t) on its local dataset D_n for a fixed number of loops, producing intermediate outputs. Specifically, gradients G_{t+1}^{n} will be computed only during outer loops for topology adjustment. The client identifies the gradients with the top $K - \kappa$ magnitudes based on the gradients G_{t+1}^{n} and upload its corresponding

Algorithm 2 FedRTS

Input: Initial model W_0, target sparsity d', learning rate η, maximum of training loops T_{end}, maximum of all loops T_{max}, adjustment interval ΔT, training datasets $\{D_n\}_{n\in\mathcal{N}}$, the number of core links κ,
Output: Final sparse model $W_{T_{max}}, m_{T_{max}}$

Randomly initialize m_0 with sparsity d',
for each loop $t = 0, 1, 2, \ldots, T_{max} - 1$ **do**
  // The server does
  Sample a random subset of clients C_t
  Broadcast server sparse model (W_t, m_t) to all clients $n \in C_t$

  for each client $n \in C_t$ **do**
    // The client n does
    $W_t^n, m_t^n \leftarrow W_t, m_t$ // Fetch global model
    $W_{t+1}^n \leftarrow W_t^n - \eta \nabla f_n(W_t^n, m_t, D_n) \odot m_t$
    Transmit W_{t+1}^n to server
    if t mod ΔT = 0 and $t < T_{end}$ **then**
      Sample batch data $\mathcal{B}_n \sim D_n$
      $G_{t+1}^n \leftarrow \nabla f_n(W_{t+1}^n, m_t, \mathcal{B}_n) \odot m_t$
      $\mathcal{I}_{t+1}^n \leftarrow \{i | i \in \text{Top}(|G_{t+1}^n|, K - \kappa)\}$ // Top $K - \kappa$ magnitudes' indices
      Transmit $\mathcal{I}_{t+1}^n$ to server
    end if
  end for

  // The server does
  if t mod ΔT = 0 and $t < T_{end}$ **then**
    $W_{t+1}^{agg}, m_{t+1} \leftarrow$ TSAdj($\{W_{t+1}^n\}_{n\in C_t}, \{\mathcal{I}_{t+1}^n\}_{n\in C_t}$) // TSAdj in Algorithm. 1.
  else
    // Update beta distribution based on weights' outcomes
    $W_{t+1}^{agg} \leftarrow \sum_{n\in C_t} p_n W_{t+1}^n$
    $m_{t+1} \leftarrow m_t$ // inner loop do not update topology
    $X_t^{agg}, \{X_t^n\}_{n\in C_t} \leftarrow$ calculate with W_{t+1}^{agg} and $\{W_{t+1}^n\}_{n\in C_t}$ by Eq. 4
    $X_t \leftarrow \gamma X_t^{agg} + (1-\gamma) \sum_{n\in C_{t+1}} p_n X_t^n$
    for each links $i = 1, 2, \ldots, \langle X_t \rangle$ **do**
      $(\alpha_i, \beta_i) \leftarrow (\alpha_i + \lambda X_{t,i}, \beta_i + \lambda(1 - X_{t,i}))$, $X_{t,i} \neq None$
    end for
  end if
end for

indices $\mathcal{I}_{t+1}^n = \{i | i \in \text{Top}(|G_{t+1}^n|, K - \kappa)\}$, representing the most promising links for the potential reactivation, rather than the complete gradients to the server, further reducing communication overhead. In contrast, weights W_{t+1}^n are computed and uploaded to the server during both inner and outer loops to facilitate global aggregation. Unlike gradients, which are selectively uploaded, the complete set of weights is transmitted to ensure comprehensive integration into the global model.

Upon receiving the locally updated gradient indices $\mathcal{I}_{t+1}^n$ in the outer loop and weights W_{t+1}^n from the participating clients, the server aggregates the weights to update global model weights W_{t+1}^n using a weighted average formula. Mathematically, the aggregated weights W_{t+1}^{agg} are computed as: $W_{t+1}^{agg} = \sum_{n\in C_t} p_n W_{t+1}^n$. If the current loop t is an outer loop (i.e., $t \mod \Delta T = 0$), the server invokes the TSAdj mechanism to adjust the sparse mask m_t. The updated mask m_{t+1} is then applied to the global model.

After completing T_{end} training loops, the server outputs the final globally aggregated weights $W_{T_{max}}$ and an optimized mask $m_{T_{max}}$, both derived from the iterative training involving all participating clients. The weights $W_{T_{max}}$ represent the learned parameters that effectively accommodate heterogeneous data distributions, while the topology $m_{T_{max}}$ preserves the robust and reliable structure of the model.

C Theoretical Details

C.1 Greedy Optimal Action

Algorithm 3 Greedy Optimal Action

Input: The mean vector μ and action size K
Return: The greedy optimal action S^*
Initialize: $S^* = \emptyset$
for $k \in 1, \ldots, K$ **do**
  $S^* = S^* \cup \{\text{argmax}_{i \neq S^*} r(S^* \cup \{i\}, \mu)\}$
end for

C.2 Proof of Theorem 3.4

The cumulative regret $Reg(T)$ is defined as:

$$Reg(T) = \mathbb{E}\left[\sum_{t=1}^{T} \Delta_{S_t}\right], \tag{10}$$

where $\Delta_{S_t} = \max\{r(S^*, \mu) - r(S_t, \mu), 0\}$ and S^* is the greedy optimal action based on μ, which is obtained by Alg. 3. The maximum reward gap is set as $\Delta_{\max} = \max_S \Delta_S$. The maximum reward gap of actions containing arm s is $\Delta_s^{\max} = \max_{S: s \in S} \Delta_S$. We define S^* and S_t are sequences, *i.e.*, $S^* = (s_1^*, \ldots, s_K^*)$, $S_t = (s_{t,1}, \ldots, s_{t,K})$, and $s_k^* \in S^*$ is the k-th element added to S^*. We denote $S_k^* = (s_1^*, \ldots, s_k^*)$ and $S_{t,k} = (s_{t,1}, \ldots, s_{t,k})$. For any arm s, define the marginal reward gap $\Delta_{s,k} = r(S_k^*, \mu) - r(S_{k-1}^* \cup \{s\}, \mu)$.

Theorem. *(Upper Bound) Under assumptions 3.2, 3.3 and 3.1 and with outcomes defined in Eq. 7, the regret $Reg(T)$ of TSAdj can be upper bounded by:*

$$\begin{aligned} Reg(T) &\leq \sum_{s \neq s_1^*} \max_{k: s \notin S_k^*} \frac{6\Delta_s^{\max} L^2 \log T}{(\Delta_{s,k} - 2LK\epsilon)^2} + \left(2K + 4\langle W \rangle + \frac{KU + 8K\epsilon^4}{\epsilon^6}\right)\Delta_{max} \\ &= O\left(\sum_{s \neq s_1^*} \max_{k: s \notin S_k^*} \frac{\Delta_{max} L^2 \log T}{\Delta_{s,k}^2}\right). \end{aligned} \tag{11}$$

for any ϵ such that $\forall s \neq S_1^*$ and $i \notin S_k^*, \Delta_{s,k} > 2LK\epsilon$, where U is a universal constant.

Proof. For any arm $s \neq s_1^*$, we apply the exploration price $F(i)$ as

$$F(s) = \max_{k: s \notin S_k^*} \frac{6L^2 logT}{(\Delta_{s,k} - 2LK\epsilon)^2} \tag{12}$$

Therefore, at the t-th iteration, there are two events

$$A_t \coloneqq \left\{\exists\, i \in \{1, \ldots, \langle W \rangle\} \Rightarrow \left|\xi_{t,i} - \hat{\mu}_{t,i}\right|^2 > \frac{3\log T}{2N_{t,i}}\right\} \tag{13}$$

$$B_t \coloneqq \left\{\exists\, i \in \{1, \ldots, \langle W \rangle\} \Rightarrow \left|\mu_i - \hat{\mu}_{t,i}\right|^2 > \frac{3\log T}{2N_{t,i}}\right\}, \tag{14}$$

where $N_{t,s} = \sum_{\tau=1}^{t} \mathbf{1}_{s \in S_\tau}$ is the number of oberservations of arm s and $\hat{\mu}_{t,s} = \frac{1}{N_{t,s}} \sum_{\tau=1}^{t} \sum_{s \in S_\tau} X_{\tau,s}$ is the empirical mean outcome of s at the start of round t.

Based on Eq. 13 and 14 , the regret $Reg(T)$ can be devided into three terms as

$$Reg(T) \leq \mathbb{E}\left[\sum_{t=1}^{T} \mathbf{1}_{\neg A_t \wedge \neg B_t} \Delta_{S_t}\right] + \mathbb{E}\left[\sum_{t=1}^{T} \mathbf{1}_{A_t} \Delta_{S_t}\right] + \mathbb{E}\left[\sum_{t=1}^{T} \mathbf{1}_{B_t} \Delta_{S_t}\right]. \tag{15}$$

We provide the Lemma C.1, C.2 and C.3 to bound these three terms one by one. Therefore, we can obtain

$$Reg(T) \leq \sum_{s \neq s_1^*} \max_{k: s \notin S_k^*} \frac{6\Delta_s^{\max} L^2 \log T}{(\Delta_{s,k} - 2LK\epsilon)^2} + \left(2K + 4\langle W \rangle + \frac{KU + 8K\epsilon^4}{\epsilon^6}\right)\Delta_{max} \tag{16}$$

C.2.1 Lemmas

Lemma C.1. *Under all assumptions and settings in Theorem 3.4, we have*

$$\mathbb{E}\left[\sum_{t=1}^{T}\mathbf{1}_{\neg A_t\wedge\neg B_t}\Delta_{S_t}\right]\leq\sum_{s\neq s_1^*}\max_{k:s\notin S_k^*}\frac{6\Delta_s^{\max}L^2\log T}{(\Delta_{s,k}-2LK\epsilon)^2}+\left(2K+\frac{KU+8K\epsilon^4}{\epsilon^6}\right)\Delta_{max}\quad(17)$$

Proof. To bound this term, we should study the difference between $s_{t,k}$ and s_k^*, that is, this term can be bounded by

$$\mathbb{E}\left[\sum_{t=1}^{T}\mathbf{1}_{\neg A_t\wedge\neg B_t}\Delta_{S_t}\right]\leq\sum_{k=1}^{K}\mathbb{E}\left[\sum_{t=1}^{T}\Delta_{S_t}\cdot\mathbf{1}_{\neg A_t\wedge\neg B_t\wedge C_{t,k}}\right]+\sum_{k=1}^{K}\mathbb{E}\left[\sum_{t=1}^{T}\mathbf{1}_{s_k^*=s_{t,k}\wedge||\xi_{t,s_k^*}-\mu_{s_k^*}||_\infty\leq\epsilon}\right]\cdot\Delta_{\max},\quad(18)$$

where $C_{t,k}$ denotes as an event which is defined as

$$C_{t,k}:=\{S_{k-1}^*=S_{t,k-1}\wedge s_k^*\neq s_{t,k}\wedge||\xi_{t,S_{k-1}^*}-\mu_{S_{k-1}^*}||_\infty\leq\epsilon\}.\quad(19)$$

We define $\xi_{t,S}=\{\xi_{t,i}|i\in S\}$ and $\mu_S=\{\mu_i|i\in S\}$. According to Lemma C.4, the first term in Eq. 18 can be bounded by:

$$\begin{aligned}\sum_{k=1}^{K}\mathbb{E}\left[\sum_{t=1}^{T}\Delta_{S_t}\cdot\mathbf{1}_{\neg A_t\wedge\neg B_t\wedge C_{t,k}}\right]&\leq\sum_{k=1}^{K}\mathbb{E}\left[\sum_{s\notin S_k^*}\sum_{t=1}^{T}\Delta_{S_t}\cdot\mathbf{1}_{s=s_{t,k}\wedge N_{t,s}\leq F(s)}\right]+\frac{UK}{\epsilon^6}\Delta_{max}\\&\leq\mathbb{E}\left[\sum_{s\neq s_1^*}\sum_{t=1}^{T}\sum_{k=1}^{K}\Delta_{S_t}\cdot\mathbf{1}_{s=s_{t,k}\wedge N_{t,s}\leq F(s)}\right]+\frac{UK}{\epsilon^6}\Delta_{max}\\&\leq\mathbb{E}\left[\sum_{s\neq s_1^*}\sum_{t=1}^{T}\Delta_{S_t}\cdot\mathbf{1}_{s\in S_t\wedge N_{t,s}\leq F(s)}\right]+\frac{UK}{\epsilon^6}\Delta_{max}\\&\leq\sum_{s\neq s_1^*}F(s)\Delta_s^{\max}+\frac{UK}{\epsilon^6}\Delta_{max}\\&\leq\sum_{s\neq s_1^*}\max_{k:s\notin S_k^*}\frac{6\Delta_s^{\max}L^2\log T}{(\Delta_{s,k}-2LK\epsilon)^2}+\frac{UK}{\epsilon^6}\Delta_{max},\end{aligned}\quad(20)$$

where U is a universal constant. Based on Lemma C.5, the second term can be bounded by

$$\sum_{k=1}^{K}\mathbb{E}\left[\sum_{t=1}^{T}\mathbf{1}_{s_k^*=s_{t,k}\wedge||\xi_{t,s_k^*}-\mu_{s_k^*}||_\infty\leq\epsilon}\right]\Delta_{\max}\leq 2K+\frac{8K}{\epsilon^2}\Delta_{\max}.\quad(21)$$

Combining Eq. 20 and Eq. 21, we have

$$\mathbb{E}\left[\sum_{t=1}^{T}\mathbf{1}_{\neg A_t\wedge\neg B_t}\Delta_{S_t}\right]\leq\sum_{s\neq s_1^*}\max_{k:s\notin S_k^*}\frac{6\Delta_s^{\max}L^2\log T}{(\Delta_{s,k}-2LK\epsilon)^2}+\left(2K+\frac{KU+8K\epsilon^4}{\epsilon^6}\right)\Delta_{max}.\quad(22)$$

Lemma C.2. *Under all assumptions and setting in Theorem 3.4, we have*

$$\mathbb{E}\left[\sum_{t=1}^{T}\mathbf{1}_{A_t}\Delta_{S_t}\right]\leq 2\Delta_{\max}\langle W\rangle.\quad(23)$$

Proof.

$$\begin{aligned}
\mathbb{E}\left[\sum_{t=1}^{T}\mathbf{1}_{A_t}\Delta_{S_t}\right] &\le \mathbb{E}\left[\sum_{t=1}^{T}\mathbf{1}_{A_t}\right]\Delta_{\max}\\
&\le \sum_{i=1}^{\langle W\rangle}\mathbb{E}\left[\sum_{t=1}^{T}\mathbf{1}_{|\xi_{t,i}-\hat{\mu}_{t,i}|^2>\frac{3\log T}{2N_{t,i}}}\right]\Delta_{\max}\\
&\le \sum_{i=1}^{\langle W\rangle}\sum_{t=1}^{T}\sum_{\tau=1}^{T-1}\mathbb{P}\left(N_{t,i}=\tau\wedge|\xi_{t,i}-\hat{\mu}_{t,i}|^2>\frac{3\log T}{2N_{t,i}}\right)\Delta_{\max}\\
&= \sum_{i=1}^{\langle W\rangle}\sum_{t=1}^{T}\sum_{\tau=1}^{T-1}\mathbb{P}\left(N_{t,i}=\tau\right)\cdot\mathbb{P}\left(|\xi_{t,i}-\hat{\mu}_{t,i}|^2>\frac{3\log T}{2N_{t,i}}\middle|N_{t,i}=\tau\right)\Delta_{\max}.
\end{aligned}\tag{24}$$

According to Lemma C.6, we have

$$\begin{aligned}
&\sum_{i=1}^{\langle W\rangle}\sum_{t=1}^{T}\sum_{\tau=1}^{T-1}\mathbb{P}\left(N_{t,i}=\tau\right)\cdot\mathbb{P}\left(|\xi_{t,i}-\hat{\mu}_{t,i}|^2>\frac{3\log T}{2N_{t,i}}\middle|N_{t,i}=\tau\right)\Delta_{\max}\\
&\qquad\le \sum_{i=1}^{\langle W\rangle}\sum_{t=1}^{T}\sum_{\tau=1}^{T-1}\mathbb{P}\left(N_{t,i}=\tau\right)\cdot 2\exp(-3\log T)\Delta_{\max}\\
&\qquad\le \sum_{i=1}^{\langle W\rangle}\sum_{t=1}^{T}\frac{2}{T}\Delta_{\max}\\
&\qquad= 2\Delta_{\max}\langle W\rangle.
\end{aligned}\tag{25}$$

Lemma C.3. *With all assumptions and setting in Theorem 3.4, we have*

$$\mathbb{E}\left[\sum_{t=1}^{T}\mathbf{1}_{B_t}\Delta_{S_t}\right]\le 2\Delta_{\max}\langle W\rangle.\tag{26}$$

Proof.

$$\begin{aligned}
\mathbb{E}\left[\sum_{t=1}^{T}\mathbf{1}_{B_t}\Delta_{S_t}\right] &\le \mathbb{E}\left[\sum_{t=1}^{T}\mathbf{1}_{B_t}\right]\Delta_{\max}\\
&\le \sum_{i=1}^{\langle W\rangle}\mathbb{E}\left[\sum_{t=1}^{T}\mathbf{1}_{|\mu_i-\hat{\mu}_{t,i}|^2>\frac{3\log T}{2N_{t,i}}}\right]\Delta_{\max}\\
&\le \sum_{i=1}^{\langle W\rangle}\sum_{t=1}^{T}\sum_{\tau=1}^{T-1}\mathbb{P}\left(N_{t,i}=\tau\wedge|\mu_i-\hat{\mu}_{t,i}|^2>\frac{3\log T}{2N_{t,i}}\right)\Delta_{\max}\\
&= \sum_{i=1}^{\langle W\rangle}\sum_{t=1}^{T}\sum_{\tau=1}^{T-1}\mathbb{P}\left(N_{t,i}=\tau\right)\cdot\mathbb{P}\left(|\mu_i-\hat{\mu}_{t,i}|^2>\frac{3\log T}{2N_{t,i}}\middle|N_{t,i}=\tau\right)\Delta_{\max}.
\end{aligned}\tag{27}$$

According to Lemma C.7, we have

$$\begin{aligned}
&\sum_{i=1}^{\langle W\rangle}\sum_{t=1}^{T}\sum_{\tau=1}^{T-1}\mathbb{P}\left(N_{t,i}=\tau\right)\cdot\mathbb{P}\left(|\mu_i-\hat{\mu}_{t,i}|^2>\frac{3\log T}{2N_{t,i}}\middle|N_{t,i}=\tau\right)\Delta_{\max}\\
&\qquad\le \sum_{i=1}^{\langle W\rangle}\sum_{t=1}^{T}\sum_{\tau=1}^{T-1}\mathbb{P}\left(N_{t,i}=\tau\right)\cdot 2\exp(-3\log T)\Delta_{\max}\\
&\qquad\le \sum_{i=1}^{\langle W\rangle}\sum_{t=1}^{T}\frac{2}{T}\Delta_{\max}\\
&\qquad= 2\Delta_{\max}\langle W\rangle.
\end{aligned}\tag{28}$$

Lemma C.4. *Under all assumptions and settings in Theorem 3.4, with $C_{t,k}$ defined in Eq. 19, we have*

$$\mathbb{E}\left[\sum_{t=1}^{T}\Delta_{S_t}\cdot\mathbf{1}_{\neg A_t\wedge\neg B_t\wedge C_{t,k}}\right]\leq\mathbb{E}\left[\sum_{s\notin S_k^*}\sum_{t=1}^{T}\Delta_{S_t}\cdot\mathbf{1}_{s=s_{t,k}\wedge N_{t,s}\leq F(s)}\right]+\frac{U\Delta_{max}}{\epsilon^6}. \tag{29}$$

Proof. This Lemma is a special case in [26] (Lemma 1), setting the size of the unit as 1 and $U = CC'$.

Lemma C.5. *With all assumptions and setting in Theorem 3.4, we have*

$$\mathbb{E}\left[\sum_{t=1}^{T}\mathbf{1}_{s_k^*=s_{t,k}\wedge||\xi_{t,s_k^*}-\mu_{s_k^*}||_\infty\leq\epsilon}\right]\Delta_{\max}\leq 2+\frac{8}{\epsilon^2}\Delta_{\max} \tag{30}$$

Proof. This Lemma is a special case in [26] (Lemma 3), setting the size of each unit as 1.

Lemma C.6. *With all assumptions and setting in Theorem 3.4, for any arm $i\in\{1,\cdots,\langle W\rangle\}$ and round t, we have*

$$\mathbb{P}\left(|\xi_{t,i}-\hat{\mu}_{t,i}|^2>\epsilon\middle|\alpha_{t,i},\beta_{t,i}\right)\leq 2\exp(-2\epsilon N_{t,i}), \tag{31}$$

where $\alpha_{t,i}$ and $\beta_{t,i}$ are the values of α_t and β_t in the TSAdj before the start of round t

Proof. This Lemma has been proved in [59] (Lemma 3).

Lemma C.7. *With all assumptions and setting in Theorem 3.4, let $X_1,\cdots,X_K$ be indentical independent random variable such that $X_i\in[0,1]$ and $\mathbb{E}[X_i]=\mu$ for any $i\in\{1,\ldots,K\}$. Then for any $\epsilon\geq 0$, we have*

$$\mathbb{P}\left(\left|\sum_{i=1}^{K}X_i-\mu\right|>K\epsilon\right)\leq 2\exp(-2K\epsilon^2), \tag{32}$$

Proof. This Lemma can be proved by Chernorff-Hoeffding Bound in [44](Theorem 1.1).

D Broader Impact

Beyond its technical contributions, FedRTS holds significant practical potential for democratizing efficient federated learning. By substantially reducing communication and computation costs via dynamic sparse training, our approach could broaden the adoption of FL in resource-constrained environments, such as the Internet of Things [31, 4, 2, 3, 58], edge perception [61, 62, 60], and privacy-sensitive medical applications [16]. However, like all pruning-based methods, FedRTS carries a risk of inadvertently amplifying biases if sparsity patterns disproportionately discard weights that encode features of minority classes. To mitigate this, we recommend integrating our method with fairness-aware regularization and continual learning techniques [45, 63] in sensitive domains. Future work will explicitly investigate the societal implications of deploying FedRTS at scale, with a focus on applications where model sparsity may impact predictive reliability.

E Additional Experiments Results

To further validate our findings, we perform additional experiments to investigate the impact of various hyperparameters and the convergence behavior of our proposed methods.

E.1 The Impact of Hyperparameters

E.1.1 The Impact of trade-off ratio γ

The trade-off ratio γ modulates the weight of individual models' information. Traditional methods only incorporate the information from the aggregated model, which is insufficient. To verify the impact of individual models' information, we conduct experiments on the trade-off ratio γ. As illustrated in Fig. 7, the test accuracy reaches its peak when the γ value is between 0.3 and 0.5. This indicates that the information from both individual models and the aggregated model is significant in model topology adjustment.

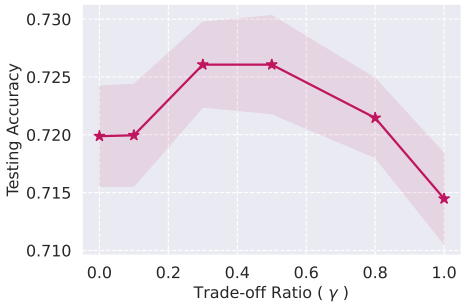

Figure 7: Impact of trade-off ratio γ

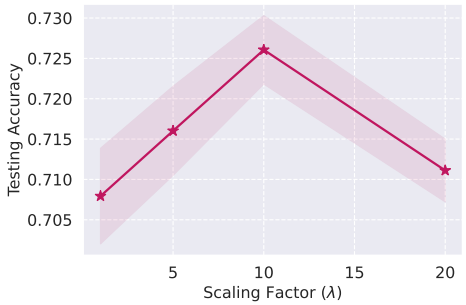

Figure 8: Impact of the scaling factor λ

ΔT	3	5	10	20
FedDST	0.664	0.683	**0.723**	**0.716**
FedMef	**0.745**	**0.737**	0.713	0.705
FedRTS	**0.735**	**0.722**	**0.734**	**0.731**

Table 2: Performance under Different Adjustment Interval (ΔT)

	CIFAR-10		CINIC-10	
	FLOPs	Comm. Cost	FLOPs	Comm. Cost
FedDST	3.4E16	221.9 GB	8.2E16	290.7 GB
FedMef	3.5E16	227.6 GB	6.1E16	222.3 GB
FedRTS	3.1E16	204.2 GB	2.9E16	105.6 GB

Table 3: The overall training FLOPs and communicational cost (Comm. Cost) for FedRTS and other frameworks to reach the accuracy of dense FedAVG.

E.1.2 The Impact of the scaling factor λ

The scaling factor λ regulates the impact of the current reward $X(t)$ at step t. A λ value that is too small will amplify the uncertainty in Thompson Sampling, whereas a λ value that is too large will disregard the uncertainty arising from insufficient samples. Thus, selecting an appropriate value for λ is crucial for enhancing FedRTS's performance. In a separate set of experiments, the influence of the reward scaling λ on the results is investigated. Figure 8 shows that the test accuracy attains its maximum at approximately 0.73 when λ is 10.

E.1.3 The Impact of the number of core links κ

The number of core links κ critically controls the adjustment intensity (the number of weights pruned and reactivated) in each outer loop, creating an essential trade-off: large κ values lead to sluggish topology evolution, while small κ risks substantial information loss. We formalize this through our decay schedule: $\kappa^l = K^l - \frac{\alpha_{adj}}{2}(1 + \cos(\frac{t\pi}{T_{end}}))K^l$, where K^l is the number of active weights at l-th layer. Our empirical investigation on CIFAR-10 in Fig. 13 demonstrates FedRTS's robustness across varying α_{adj}, consistently outperforming baseline methods.

E.1.4 The Impact of Adjustment Interval ΔT

In federated dynamic sparse training, the adjustment interval (ΔT) between two outer loops for adjustment critically influences model performance through a fundamental trade-off. Excessively long intervals may delay the discovery of optimal sparse patterns, while overly frequent adjustments can prevent adequate model adaptation between updates and require higher resource consumption.

Our experiments with FedRTS on CIFAR-10 explored this trade-off by evaluation with various adjustment intervals ($\Delta T \in \{3, 5, 10, 20\}$). As depicted in Table 2, FedRTS consistently outperformed baseline methods across most cases. Notably, FedRTS's performance improved with shorter intervals, demonstrating FedRTS's unique ability to quickly recover from topology changes, which is a crucial advantage in dynamic sparse training scenarios.

E.2 Efficiency Analysis

E.2.1 Training Cost

We evaluate FedRTS's efficiency by analyzing training costs on CIFAR-10 and CINIC-10. To compare expenses, we measure total training FLOPs and communication costs required to match the

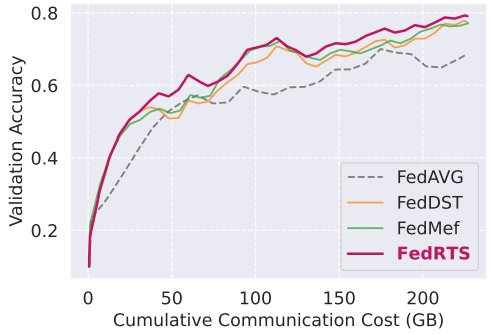

Figure 9: Validation accuracy of FedRTS and SOTA frameworks on CIFAR-10 dataset for cumulative communicational cost.

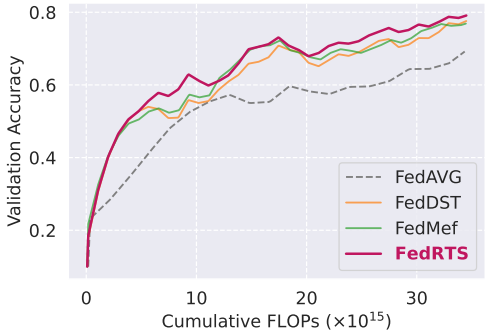

Figure 10: Validation accuracy of FedRTS and SOTA frameworks on CIFAR-10 dataset for cumulative FLOPs during training.

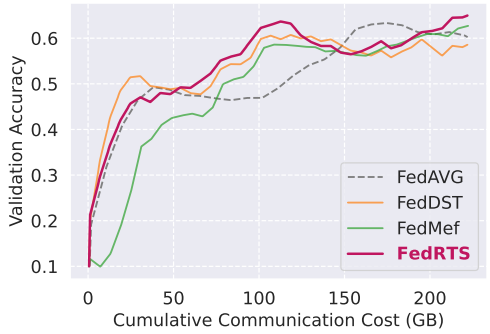

Figure 11: Validation accuracy of FedRTS and SOTA frameworks on CINIC-10 dataset for cumulative communicational cost during training.

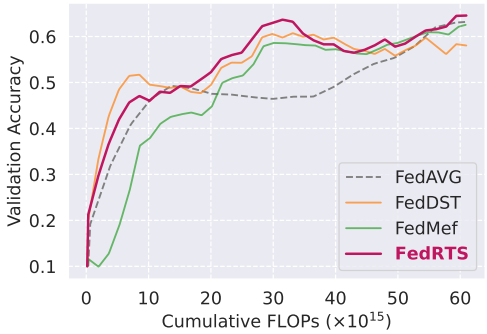

Figure 12: Validation accuracy of FedRTS and SOTA frameworks on CINIC-10 dataset for cumulative FLOPs during training.

testing accuracy of dense FedAVG. The accuracy is averaged over 10 rounds to mitigate variance from random factors, ensuring a more stable evaluation.

As shown in Tab. 3, FedRTS achieves the testing accuracy of dense FedAVG with significantly lower costs. On CINIC-10, it achieves 48% training FLOPs and the communication cost compared to the best baseline FedMef, which highlights the efficiency of FedRTS.

E.2.2 Convergence Behavior

Convergence is crucial for effective learning, precise predictions, and optimized resource use. To evaluate this, we track the validation accuracy of FedRTS and SOTA frameworks on CIFAR-10 and CINIC-10 against cumulative communication and computational costs, as shown in Fig. 9, Fig. 10, Fig. 11, and Fig. 12. Accuracy is averaged over 10 rounds to reduce variance and provide a stable representation. Our experimental results demonstrate distinct convergence patterns across datasets. On CIFAR-10, FedRTS exhibits superior convergence characteristics throughout the entire training process. The CINIC-10 dataset reveals a more complex trajectory: while FedDST initially demonstrates higher performance during early training phases, FedRTS consistently surpasses competing methods in later stages when considering both communication efficiency and computational costs. These empirical results provide strong evidence for the effectiveness of the TSAdj mechanism in FedRTS, particularly in optimizing the trade-off between model performance and resource consumption during federated training.

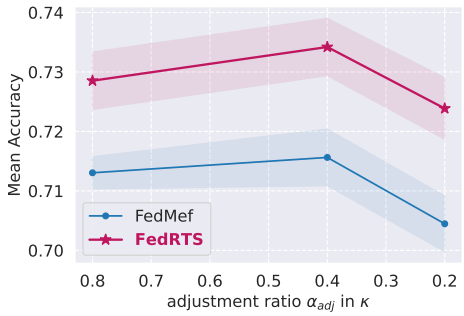

Figure 13: Accuracy on CIFAR-10 with ResNet under various degrees of adjustment ratio α_{adj}.

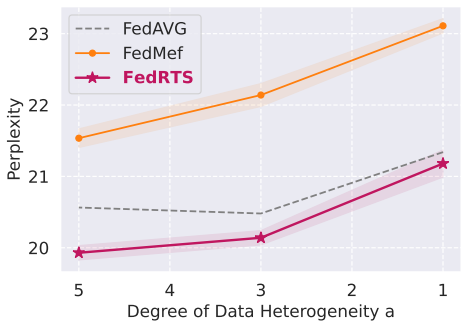

Figure 14: Perplexity on TinyStories under various degrees of data heterogeneity. The lower a represents a higher non-IID degree.

E.3 Broader Experiments

E.3.1 FedRTS vs. FedCUCB

The Combinatorial Upper Confidence Bound (CUCB) is another optimization method for the Combinatorial Multi-Armed Bandit (CMAB) problem. Unlike Combinatorial Thompson Sampling (CTS), which relies on probabilistic sampling, CUCB employs deterministic optimistic reward estimates to balance exploration and exploitation. However, its inherent optimism can lead to persistently selecting suboptimal arm combinations, particularly in dynamic pruning scenarios where adaptability is crucial.

CUCB maintains upper confidence bounds (UCBs) rather than probability distributions for each arm i, which formulated as

$$\bar{\mu}_{t,i} = \hat{\mu}_{t,i} + \sqrt{\frac{3\log t}{2\Psi_i}}, \tag{33}$$

where Ψ_i denotes the number of times arm i has been selected, $\hat{\mu}_{t,i}$ denotes the empirical mean reward (initialized as $\hat{\mu}_{0,i} = 0$). The selected arms for action S_t are determined by

$$S_t = \{i \in \mathrm{Top}(\bar{\mu}_t, K)\}$$

For the selected arms $i \in S_t$ with outcomes X_t,the empirical mean is updated as:

$$\Psi_i \leftarrow \begin{cases} \Psi_i, & i \notin S_t \\ \Psi_i + 1, & i \in S_t \end{cases}, \quad \hat{\mu}_{t+1,i} = \begin{cases} \hat{\mu}_{t,i}, & i \notin S_t \\ \frac{\hat{\mu}_{t,i}(\Psi_i - 1) + X_{t,i}}{\Psi_i}, & i \in S_t \end{cases}, \tag{34}$$

The next round's action selection then uses the updated UCB $\bar{\mu}_{t+1,i}$ computed by Eq. 33.

A notable drawback of FedCUCB is its handling of under-explored ("bad") arms. If an arm i is rarely selected (low Ψ_i), its confidence bound $\bar{\mu}_{t,i}$ grows indefinitely as $t \to \infty$, eventually forcing its selection. This behavior disrupts convergence, as the algorithm repeatedly revisits suboptimal arms instead of refining the topology.

To evaluate this limitation, we propose FedCUCB, a federated pruning framework based on CUCB. In contrast to FedRTS, FedCUCB maintains UCB $\bar{\mu}_t$ and define the outcomes X_t as same in Eq. 3. The results in Fig. 15 demonstrates that FedCUCB performs slightly worse than FedRTS, but still outperform other baselines.

E.3.2 Pruning vs. Weights Quantization

Parallel to pruning-based approaches, quantization techniques [50, 23, 15, 17, 20] offer an alternative for reducing communication overhead in FL. While both strategies aim to alleviate transmission costs,

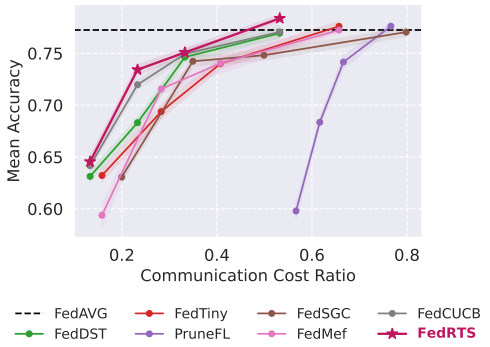

Figure 15: Accuracy on CIFAR-10 with ResNet18 for FedRTS, FedCUCB and other federated dynamic pruning methods.

Method	Comm. Cost	FLOPs	Memory	Acc.
FedAVG	89.52 MB	17.8E+12	134.29MB	77.24
FedPAQ	22.48 MB	17.8E+12	134.29 MB	**73.34**
FedDST	**20.85 MB**	**3.8E+12**	**31.28 MB**	68.31
FedRTS	**20.85 MB**	**3.8E+12**	**31.28 MB**	**73.41**

Table 4: Performance and training cost comparison between federated pruning (FedDST, FedRTS) and federated quantization (FedPAQ) frameworks on CIFAR-10 using ResNet18. FedPAQ apply 8-bit quantization, while FedDST and FedRTS employ a target density of $d' = 0.2$.

weights quantization operates after dense local training, whereas pruning enables sparse training by design. This fundamental difference grants pruning distinct advantages in resource-constrained settings: it reduces computation through selective updates, lowers memory usage by maintaining dynamic sparsity, and ensures consistent communication efficiency across rounds.

Empirical evidence supports these trade-offs. As shown in Table 4, our FedRTS outperforms 8-bit FedPAQ [50], a weights quantization baseline, in computational and memory efficiency, while retaining competitive model accuracy. This makes pruning a more holistic solution for FL deployments on edge devices.

F Additional Setup Details

F.1 Datasets

In the CV tasks, we apply our approach using ResNet18 on CIFAR-10, CINIC-10, TinyImageNet, and SVHN datasets.

- CIFAR-10: It contains 50,000 training images and 10,000 testing images. Each image in CIFAR-10 is a 3×32×32 RGB image, implying 10 classes of objects.
- CINIC-10: It contains three equal subsets - train, validation, and test–each comprising 90,000 images. CINIC-10 is an extended version of CIFAR-10 that includes additional images from ImageNet.
- TinyImageNet: It contains 200 classes with 500 training images, 50 validation images, and 50 test images per class, each resized to 64×64 pixels.
- SVHN: There are 73,257 digit images in the training set and 26,032 images in the testing set. The digit images are obtained from house numbers in Google Street View images.

In the NLP tasks, we choose GPT-2-32M as the backbone model with the TinyStories dataset. TinyStories contains synthetically generated short stories that only use a limited vocabulary. We truncate each story to 256 tokens, with 2,120,000 stories used for training and 22,000 for testing.

F.2 Baselines

The selected federated pruning baselines differ in their approaches to sparse model initialization and subsequent model topology adjustment:

- PruneFL: Initializes the sparse model by training on a powerful device and applies adaptive pruning by importance measure. In our implementation, we perform random pruning instead of pruning with powerful devices due to resource constraints. Moreover, we set the time t_j of each parameter j with 1.

- FedDST: Applies dynamic sparse training on clients and performs greedy aggregation to produce a new global topology.
- FedTiny: Initializes the model by adaptive batch normalization selection and adjusts the model mask based on the aggregated top-K gradients of parameters.
- FedSGC: Adjusts model topology on devices guided by the gradient congruity, which is similar to FedDST but requires extra communication cost.
- FedMef: Utilizes budget-aware extrusion to transfer essential information of pruned parameters to other parameters and reduces activation memory by scaled activation pruning.

F.3 Extra Information

The experiments are simulated via multi-process on an Nvidia RTX 5880 GPU, an Intel Core i9 CPU with 48 GB of memory.

F.4 Communication and Computational Cost

In our experiments, we analyzed the proposed FedRTS against other methods by examining the computational FLOPs and communication costs. We began by introducing sparse compression strategies and then detailed how we calculated these metrics.

F.4.1 Compression Strategy

When storing a matrix, two key components are values and positions. Compression techniques focus on reducing the storage required for the positions of non-zero values within the matrix. Consider a scenario where we aim to store the positions of m non-zero values with a b-bit width in a sparse matrix M, which comprises n elements and has a shape of nr × nc. The density of the matrix, denoted by d = m/n, determines the compression scheme applied to represent M. The storage involves using o bits to represent the positions of the m non-zero values, resulting in an overall storage size of s.

- For density ($d \in [0.9, 1]$), a dense scheme is utilized, leading to $s = nb$.
- For density $d \in [0.3, 0.9)$, the bitmap (BM) method is employed, storing a map with n bits, where $o = n$ and $s = o + mb$.
- For density $d \in [0.1, 0.3)$, the coordinate offset (COO) scheme is applied, storing elements with their absolute offsets and requiring $o = m\lceil log2n \rceil$ extra bits for position storage. Thus, the overall storage becomes $s = o + mb$.
- For density $d \in [0., 0.1)$, the compressed sparse row (CSR) and compressed sparse column (CSC) methods are used based on size considerations. These methods use column and row indexes to store element positions, with CSR needing $o = m\lceil log_2 n_c \rceil + n_r \lceil log_2 m \rceil$ bits. The overall storage size is $s = o + mb$.

Reshaping is performed on tensors before compression, enabling the determination of the memory required for training the network's parameters.

F.4.2 Communication Cost

In terms of communication costs, FedRTS shares a communication cost strategy similar to that of FedTiny. In contrast to other baseline methods like FedDST, where clients need to upload TopK gradients to the server every ΔR rounds to facilitate parameter expansion, the quantity of TopK gradients, denoted as ξ_t, aligns with the number of marked parameters θ low. Here, $\xi_t = \zeta_t(1-s_m)n_\theta$, with $\zeta_t = 0.2(1 + cos\frac{t\pi}{R_{stop}E})$representing the adjustment rate for the t-th iteration. This results in minimal upload overhead. Moreover, there is no communication overhead for the model mask m during download since sparse storage formats like bitmap and coordinate offset contain the same element position information. Auxiliary data such as the learning rate schedule are excluded.

We define the storage for dense and sparse parameters as O_d and O_s, respectively. Notably, in the inner loop, all federated pruning methods except FedAVG share the same communication cost, which amounts to $2O_s$. However, in the outer loop, different federated learning frameworks exhibit varying communication costs. The communication costs of different federated learning frameworks in the outer loop are detailed below:

- FedAVG: The data exchange amounts to $2O_d$, encompassing the uploading and downloading of dense parameters.
- FedDST: In this scenario, the model mask does not necessitate additional space for storage as the compressed sparse parameters already embody the mask information. Hence, the data exchange per round stands at $2O_s$, covering the upload and download of sparse parameters.
- PruneFL: PruneFL requires the clients to upload the full-size gradients squared for adjustment, therefore, the data exchange in the outer loop is $O_d + O_s$.
- FedSGC: FedSGC requires the server to send the gradient congruity to the server for each round, which has the same size as sparse weight. Therefore, the data exchange in the outer loop is $3O_s$.
- FedTiny and FedMef: In comparison to FedDST, FedTiny and FedMef entail the upload of TopK gradients every ΔR rounds. Therefore, the maximum data exchange per round sums up to $2O_s + O_\xi$, where O_ξ signifies the storage required for the TopK gradients.
- FedRTS: When compared to FedTiny and FedMef, FedRTS only requires uploading the index of the TopK gradients every ΔR rounds, while FedTiny and FedMef necessitate uploading both the index and the values of TopK gradients at the same intervals. Consequently, the maximum data exchange per round amounts to $2O_s + 0.5O_\xi$, where O_ξ represents the storage needed for the TopK gradients.

F.4.3 Computational FLOPs

Training FLOPs encompass both forward-pass FLOPs and backward-pass FLOPs, with operations tallied on a per-layer basis. During the forward pass, layer activations are sequentially computed using prior activations and layer parameters. In the backward pass, each layer computes activation gradients and parameter gradients, with twice as many FLOPs expended in the backward pass as in the forward pass. FLOPs related to batch normalization and loss calculation are excluded.

In a detailed breakdown, assuming inference FLOPs for dense and static sparse models are denoted as F_d and F_s respectively, and the local iteration count is E, the maximum training FLOPs for each framework are as follows:

- FedAVG: Requires training a dense model, resulting in training FLOPs per round amounting to $3F_dE$.
- PruneFL: it requires calculating the dense gradients for each backward, therefore the training FLOPs is $(2F_d + F_s)E$.
- FedMef: Introduces a minor calculation overhead for BaE and SAP. Therefore, the maximum training FLOPs are estimated as $3(F_s + F_o)(E - 1) + (F_s + F_o) + 2F_d$, where Fo represents the computational overhead of BaE and SAP. Fo is approximated as $F_o = 4(1 - s_m)n_\theta + n_a log_{n_a}$, with $4(1 - s_m)n_\theta$ representing the FLOPs for regularization and WSConv, and$n_a log n_a$ indicating the FLOPs for activation pruning.
- FedRTS, FedTiny, FedSGC, and FedDST: Implement RigL-based methods to adjust model architectures, necessitating clients to compute dense gradients in the final iteration. The maximum training FLOPs sum up to $3F_s(E - 1) + F_s + 2F_d$.