Device-Wise Federated Network Pruning

Shangqian Gao*[1], Junyi Li*[2], Zeyu Zhang*[3], Yanfu Zhang[4], Weidong Cai[5], and Heng Huang[2]

1. Electrical and Computer Engineering, University of Pittsburgh

2. Computer Science, University of Maryland College Park

3. Information, University of Arizona 4. Computer Science, College of William and Mary

5. School of Computer Science, The University of Sydney

University of Pittsburgh

CVPR JUNE 17-21, 2024 SEATTLE, WA

Task and Contributions

- Task
  - Our goal is to compress and accelerate deep neural networks with structural pruning under for federated learning.
- Contributions
  - We proposed a novel channel pruning method for federated learning. A server-side network and device-wise sub-networks are learned to achieve a better trade-off between the performance and the computational resource.
  - We proposed to use an embedding layer and a hypernetwork to generate sub-networks on each device.
  - We provided the theoretical guarantee of convergence for our method for federated learning.
  - Extensive experiments on CIFAR-10, CIFAR-100, and TinyImageNet show the effectiveness of our method.

University of Pittsburgh

CVPR JUNE 17-21, 2024 SEATTLE, WA

Federated Learning Setting

In the FL setting, we train a neural network on N local datasets $D_n,\ n \in \{1, 2, 3 \cdots, N\}$. Through this paper, the data distribution on local devices is heterogeneous. To train a neural network in this setting, we want to optimize the following optimization problem:

$$\min_{\mathcal{W}} \frac{1}{N} \sum_{n=1}^{N} \mathcal{L}(\mathcal{W}, D_n), \tag{1}$$

where $\mathcal{W}$ is the weights of the CNN, and $\mathcal{L}$ is the objective function. One common method to minimize communication costs is by using local stochastic gradient descent (SGD), where the local device performs several update steps with their local data before averaging the model weights $\mathcal{W}$.

University of Pittsburgh

CVPR JUNE 17-21, 2024 SEATTLE, WA

Method: Hypernetwork

To prune a model, we use a binary vector $\mathbf{a} \in \{0,1\}$ to represent whether to keep or prune a channel. To facilitate the learning of the sub-network architecture, we use a hypernetwork (HN) and an embedding layer to generate the architecture vector $\mathbf{a}$:

$$\begin{aligned} \mathbf{a}^0 &= \text{HN}(\text{Emb}(0;\theta_{\text{Emb}});\theta_{\text{HN}}), \\ \mathbf{a}^n &= \text{HN}(\text{Emb}(n;\theta_{\text{Emb}});\theta_{\text{HN}}),\ n = 1,\cdots,N, \end{aligned} \tag{2}$$

where θ_{Emb} is the parameters of the embedding layer Emb, and n is the index for each device, and θ_{HN} is the parameters of the HN. We use Straight-Through Gumbel-Sigmoid to enable gradient calculation for the HN. To control the pruning of each channel, we apply $\mathbf{a}$ to the feature map of each layer:

$$\hat{\mathcal{F}}_l = \mathbf{a}_l \odot \mathcal{F}_l, \tag{3}$$

where $\hat{\mathcal{F}}_l$ is the feature map after applying $\mathbf{a}_l$ (the architecture vector of lth layer). Note that we insert $\mathbf{a}_l$ after normalization and activation layers, which correspond to control the output channels of the previous convolution layer and input channels of the next convolution layer.

Method: DWNP

The channel pruning objective function can be formulated as follows:

$$\min_{\theta} \frac{1}{N} \sum_{n=1}^{N} \mathcal{L}(\mathcal{W}, D_n^{\mathbf{a}}; \mathbf{a}^0 \odot \mathbf{a}^n) + \lambda[(\mathcal{R}(T(\mathbf{a}^0), p_s T_{\text{total}}) + \frac{1}{N} \sum_{n=1}^{N} \mathcal{R}(T(\mathbf{a}^0 \odot \mathbf{a}^n), p_d^n T_{\text{total}})], \tag{4}$$

where θ contains both θ_{HN} and θ_{Emb}, $\mathbf{a}^0$ and $\mathbf{a}^n$ are generated by using Eq. 2, $D_n^{\mathbf{a}}$ is a subset of the local datasets D_n, $\mathcal{R}$ is the regularization loss to control the FLOPs of the sub-network, $p_s \in (0, 1]$ is a predefined hyperparameter to control the preserved FLOPs of the server-side sub-network, $p_d^n \in (0, 1]$, $n = 1, \cdots, N$ are also predefined hyperparameters to control the FLOPs of sub-networks on each device, $T(\mathbf{a}^0)$ or $T(\mathbf{a}^0 \odot \mathbf{a}^n)$ is the current FLOPs decided by the sub-network architecture $\mathbf{a}^0$ or $\mathbf{a}^0 \odot \mathbf{a}^n$, and T_{total} is the total FLOPs of the CNN.

University of Pittsburgh

CVPR JUNE 17-21, 2024 SEATTLE, WA

Convergence Guarantee

Theorem 3.1. *Suppose we choose the upper level learning rate η and the lower level learning rate γ as:*

$$\eta = \min\left\{\frac{1}{4Lr_\theta}, \left(\frac{2bN\Delta_\theta}{K_{HN}L\sigma^2}\right)^{1/2}\right\},$$

and

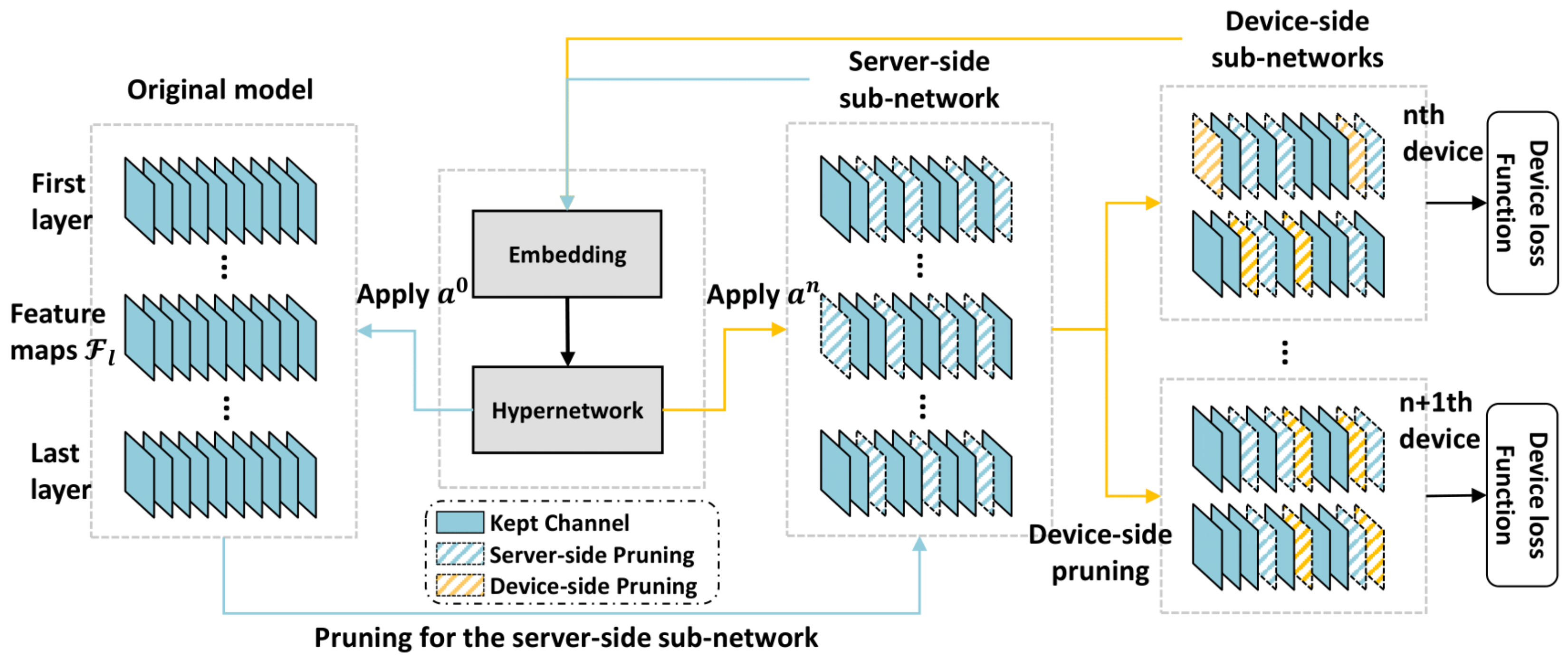
$$\gamma = \min\left\{\frac{1}{4L}, \left(\frac{\lambda bN}{K_{HN}L^2\sigma^2}\right)^{1/2}\right\},$$

then we have:

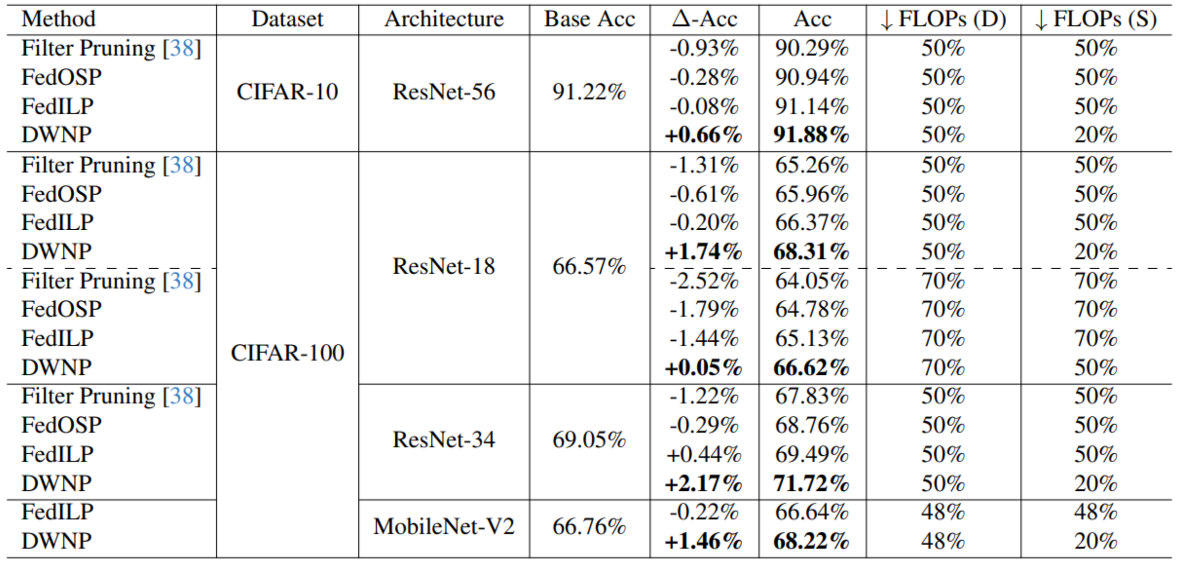
$$\frac{1}{K_{HN}}\sum_{k_{HN}=0}^{K_{HN}-1}\mathbb{E}\|\nabla h(\bar{\theta}_{k_{HN}})\|^2 = O\left(\frac{1}{(bNK_{HN})^{1/2}}\right)$$

where b is the mini-batch size, N is the number of devices, and K_{HN} is the number of update steps to the upper level variable θ.

Method Overview

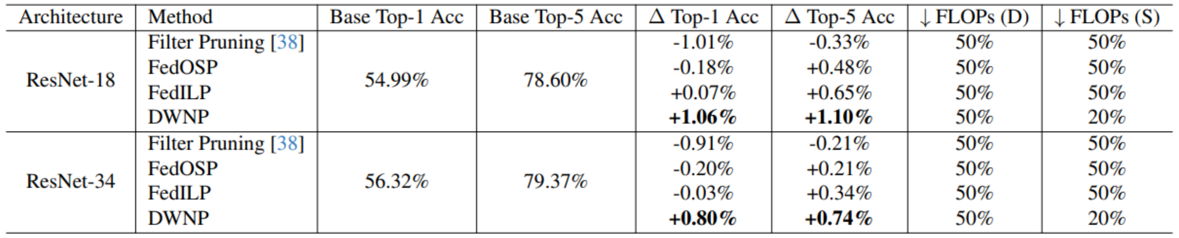


Experiments



Method	Dataset	Architecture	Base Acc	Δ-Acc	Acc	↓ FLOPs (D)	↓ FLOPs (S)
Filter Pruning [38]	CIFAR-10	ResNet-56	91.22%	-0.93%	90.29%	50%	50%
FedOSP				-0.28%	90.94%	50%	50%
FedILP				-0.08%	91.14%	50%	50%
DWNP				**+0.66%**	**91.88%**	50%	20%
Filter Pruning [38]	CIFAR-100	ResNet-18	66.57%	-1.31%	65.26%	50%	50%
FedOSP				-0.61%	65.96%	50%	50%
FedILP				-0.20%	66.37%	50%	50%
DWNP				**+1.74%**	**68.31%**	50%	20%
Filter Pruning [38]				-2.52%	64.05%	70%	70%
FedOSP				-1.79%	64.78%	70%	70%
FedILP				-1.44%	65.13%	70%	70%
DWNP				**+0.05%**	**66.62%**	70%	50%
Filter Pruning [38]		ResNet-34	69.05%	-1.22%	67.83%	50%	50%
FedOSP				-0.29%	68.76%	50%	50%
FedILP				+0.44%	69.49%	50%	50%
DWNP				**+2.17%**	**71.72%**	50%	20%
FedILP		MobileNet-V2	66.76%	-0.22%	66.64%	48%	48%
DWNP				**+1.46%**	**68.22%**	48%	20%

Table 1. Results of CIFAR-10 and CIFAR-100. 'Base Acc' represents the baseline training accuracy. 'Δ-Acc' represents the accuracy changes before and after pruning. 'Acc' represents the accuracy after pruning. '↓ FLOPs (D)' and '↓ FLOPs (S)' represent the pruned FLOPs of device-side and server-side sub-networks.



Architecture	Method	Base Top-1 Acc	Base Top-5 Acc	Δ Top-1 Acc	Δ Top-5 Acc	↓ FLOPs (D)	↓ FLOPs (S)
ResNet-18	Filter Pruning [38]	54.99%	78.60%	-1.01%	-0.33%	50%	50%
	FedOSP			-0.18%	+0.48%	50%	50%
	FedILP			+0.07%	+0.65%	50%	50%
	DWNP			**+1.06%**	**+1.10%**	50%	20%
ResNet-34	Filter Pruning [38]	56.32%	79.37%	-0.91%	-0.21%	50%	50%
	FedOSP			-0.20%	+0.21%	50%	50%
	FedILP			-0.03%	+0.34%	50%	50%
	DWNP			**+0.80%**	**+0.74%**	50%	20%

Table 2. Comparison results on TinyImageNet with ResNet-18/34. 'Base Top-1/5' represents the baseline training Top-1/5 accuracy. 'Δ Top-1/5 Acc' represents the Top-1/5 accuracy changes before and after pruning.

University of Pittsburgh

CVPR JUNE 17-21, 2024 SEATTLE, WA

Thanks for Watching!

University of Pittsburgh

CVPR
JUNE 17-21, 2024
SEATTLE, WA