DPAI: Differentiable Pruning At Initialization With Node-Path Balance Principle

.ichuan Xiang[1*] Quan Nguyen–Tri[2,3*] ,Lan–Cuong Nguyen[2,3], Hoang Pham[1], Khoat Than[2], Long Tran–Thanh[1], Hongkai Wen[1]

[1]University of Warwick [2] Hanoi University of Science and Technology [3]FPT Software AI Center

Introduction

Lottery Ticket Hypothesis (LTH) suggests the existence of sparse networks at initialization that can be trained to full accuracy.

Pruning at Initialization (PaI) identifies sparse networks before training based on:

- SNIP $|\frac{\partial \mathcal{L}}{\partial \theta} \odot \theta|$
- GraSP $(H\frac{\partial \mathcal{L}}{\partial \theta}) \odot \theta$
- SynFlow $|\frac{\partial R}{\partial \theta} \odot \theta|$
- Magnitude $|\theta|$

Node-Path Balancing (NPB) principle optimizing subnetwork's topologies, shows better efficient sparse networks.

However, current NPB implementations require solving large-scale discrete optimization problems, limiting practical efficiency. We introduce Differentiable Pruning at Initialization (DPaI):

1. **Converts discrete NPB optimization into a differentiable formulation.**
2. **Dynamically optimizes pruning masks to enhance network topology.**
3. **Utilizes efficient gradient-based methods for fast, superior pruning.**

Method Overview

DPaI starts by estimating layer-wise sparsity using the ERK method, then iteratively optimizes a differentiable pruning mask by updating scores based on the Node-Path Balancing (NPB) metric. The algorithm adjusts scores using gradient-based updates to align pruning decisions with network topology, enabling efficient discovery of sparse yet trainable subnetworks.

The figure below illustrates the details of algorithm:

```
Algorithm 1 Differentiable Pruning at Initialization (DPaI)
1:  Input: network $f(x, \mathbf{W})$, final sparsity $\rho$, iteration steps $T$, hyperparameter $\alpha, \beta, \eta$
2:  Initialize the score parameters: $s_{i,j}^{(l)} \sim \mathcal{N}(0,1)$
3:  Obtain layer-wise sparsity $k^{(l)}$ from $\rho$ by using the ERK method
4:  for $t \in 1, \ldots, T$ do
5:      Binarize the mask on each layer: $m_{i,j}^{(l)} \leftarrow \text{Top}_{k^{(l)}}(|s_{i,j}^{(l)}|)$
6:      Compute the number of effective paths: $\mathcal{R}_P \leftarrow f(\mathbb{1}, \mathbf{M})$
7:      Compute the derivatives with respecting to each objective: $\frac{\delta \mathcal{R}_P}{\delta s_{i,j}^{(l)}}, \frac{\delta \mathcal{R}_N}{\delta s_{i,j}^{(l)}}, \frac{\delta \mathcal{R}_C}{\delta s_{i,j}^{(l)}}$
8:      Update the score parameters: $s_{i,j}^{(l)} \leftarrow s_{i,j}^{(l)} + \eta\left((1-\alpha)\frac{\delta \mathcal{R}_P}{\delta s_{i,j}^{(l)}} + \alpha\left((1-\beta)\frac{\delta \mathcal{R}_N}{\delta s_{i,j}^{(l)}} + \beta\frac{\delta \mathcal{R}_C}{\delta s_{i,j}^{(l)}}\right)\right)$
9:  end for
10: Output: pruned network $f(x, \mathbf{M} \odot \mathbf{W})$
```

Figure 1. DPaI Algorithm Details

Differentiable Calculation of the Effective Path: Denotes $P(v_j^{(i)})$ is the number of incoming paths to a node $v_i^{(l)}$. The number of effective paths is the number of incoming paths to the nodes in the last layer L:

$$\mathcal{R}_P = \sum_{j=1}^{h^{(L)}} P(v_j^{(L)}), \quad P(v_j^{(l)}) = \sum_{i=1}^{h^{(l-1)}} m_{i,j}^{(l)} P(v_i^{(l-1)}), \quad P(v_j^{(0)}) = 1$$

Differentiable Calculation of the Effective Node/Channel: A node $v_i^{(l)}$ is activated if there is a path connecting the input nodes to it $P(v_j^{(i)}) > 0$, and there is a path connecting it to the output nodes $\frac{\delta \mathcal{R}_p}{\delta P(v_j^{(i)})}$ Therefore, a node $v_i^{(l)}$ is considered effective when

$$N(v_j^{(l)}) = P(v_j^{(l)})\frac{\delta \mathcal{R}_P}{\delta P(v_j^{(l)})} = \frac{\delta \mathcal{R}_P}{\delta P(v_j^{(l)})}\sum_{i=1}^{h^{(l-1)}} m_{i,j}^{(l)} P(v_i^{(l-1)}) \propto \sum_{i=1}^{h^{(l-1)}} \left|\frac{\delta \log \mathcal{R}_P}{\delta s_{i,j}^{(l)}}\right| m_{i,j}^{(l)} > 0$$

Results

DPaI consistently outperforms prior PaI methods across multiple datasets and architectures, especially at extreme sparsity levels. It achieves notable accuracy gains (up to 4.6%) on ResNet models and remains competitive on VGG-19.

On ImageNet-1K with EfficientNet-B0, DPaI surpasses Synflow with both higher average and best accuracy. In ViT-B/16 experiments, DPaI also delivers substantial improvements, showcasing its adaptability to transformer architectures. Additionally, DPaI offers lower and more stable pruning time compared to existing methods, confirming its practical efficiency.

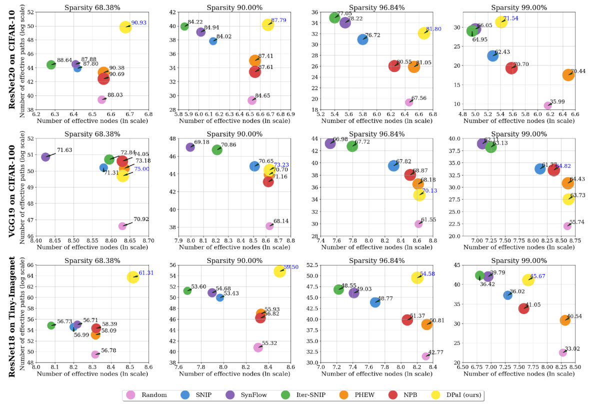

Figure 2. Accuracy of PaI methods across datasets and sparsity levels(highlighted in blue).