Auto-Train-Once: Controller Network Guided Automatic Network Pruning from Scratch

Xidong Wu*, Shangqian Gao*, Zeyu Zhang, Zhenzhen Li, Runxue Bao, Yanfu Zhang, Xiaoqian Wang, Heng Huang

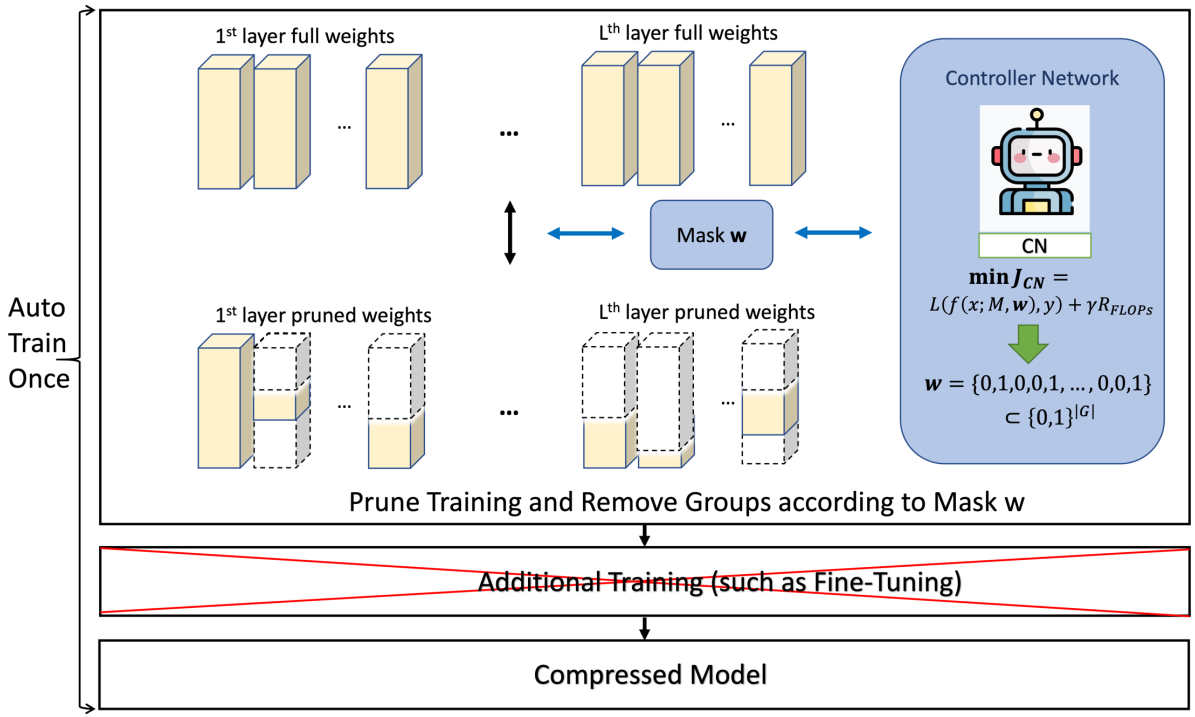
Motivation

- Current techniques for DNN pruning often involve intricate multi-step processes that require domain-specific expertise, It makes their widespread adoption challenging. To address the limitation, the Only-Train-Once (OTO) and OTOv2 are proposed to eliminate the need for additional fine-tuning steps by directly training and compressing a general DNN from scratch.
- However, OTO has poor final performance. It reformulates the objective as a constrained regularization problem. The local minima may be scattered in diverse locations. Yet, as the augmented regularization in OTO penalizes the mixed L1/L2 norm of all trainable parameters in ZIGs, it restricts the search space to converge around the origin point.

Motivation

- OTOv2 improves OTO by constructing pruning groups in ZIGs based on salience scores and only penalizes the parameters in pruning groups. However, model variables vary as training and the statically selected pruning groups in the early training stage can lead to convergence issues of local optima. Drawbacks in the algorithm design prevent them from giving a complete convergence analysis. For instance, OTO assumes the deep model as a strongly convex function and OTOv2 assumes a full gradient estimate at each iteration, which does not align with the practical settings of DNN training.

Model Pruning



Auto-Train-Once (ATO) : end-to-end model pruning algorithm.

Structural pruning is a widely adopted direction to reduce the size of DNNs.

We propose a generic framework (ATO) to train and prune DNNs in a completely end-to-end and automatic manner. After model training, we can directly obtain the compressed model without additional fine-tuning steps. We design a network controller to dynamically guide the channel pruning, preventing being trapped in local optima.

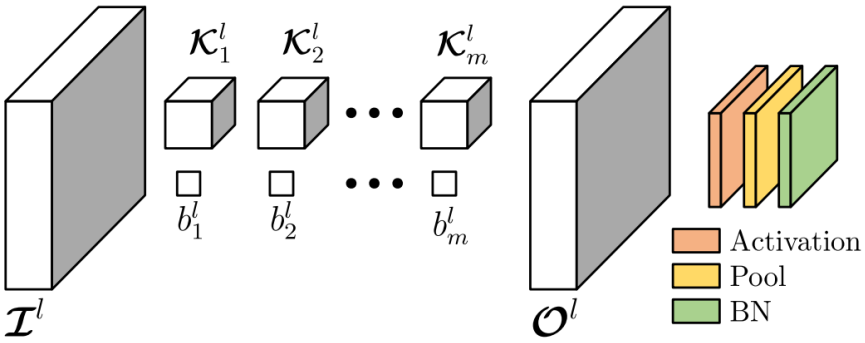
Preliminary

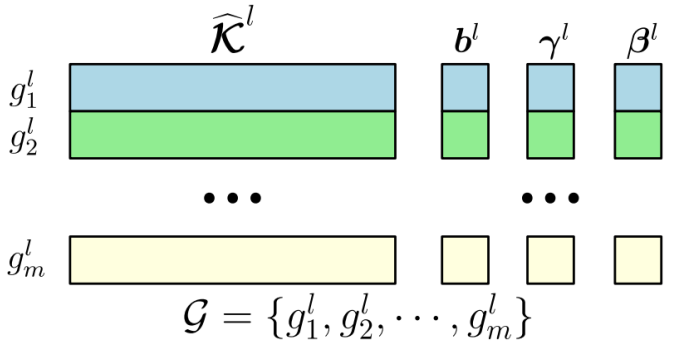
Definition 1. *(Zero-Invariant Groups (ZIGs)) [7]. In the context of a layer-wise Deep Neural Network (DNN), entire trainable parameters are divided into disjoint groups $\mathcal{G} = \{g\}$. These groups are termed zero-invariant groups (ZIGs) when each group $g \in \mathcal{G}$ exhibits zero-invariant, where zero-invariant implies that setting all parameters in g to zero leads to the output corresponding to the next layer also being zeros.*

The Convolutional layer (Conv) without bias followed by the batch-normalization layer (BN) can be shown as below:

$$\mathcal{O}^l \leftarrow \mathcal{I}^l \otimes \hat{\mathcal{K}}^l, \mathcal{I}^{l+1} \leftarrow \frac{a\left(\mathcal{O}^l\right) - \boldsymbol{\mu}^l}{\boldsymbol{\sigma}^l} \odot \boldsymbol{\gamma}^l + \boldsymbol{\beta}^l$$

where $\mathcal{I}^l$ denotes input tensor, $\otimes$ denote the convolutional operation, $\hat{\mathcal{K}}^l$ presents one output channel in l^{th} layer, $\odot$ is the element-wise multiplication, $a(\cdot)$ is the activation function, and $\boldsymbol{\mu}^l, \boldsymbol{\sigma}^l, \boldsymbol{\gamma}^l, \boldsymbol{\beta}^l$ represent running mean, standard deviation, weight and bias, respectively in BN. Each output channel of the Conv $\hat{\mathcal{K}}^l$, and corresponding channel-wise BN weight $\boldsymbol{\gamma}^l$ and bias $\boldsymbol{\beta}^l$ belong to one ZIG because they being zeros results in their corresponding channel output to be zeros as well.





Auto-Train-Once

Algorithm 5 ATO Algorithm

1: **Input:** Target model with model weights $\mathcal{M}$ (no need to be pre-tained). Datasets $\mathcal{D}$, $\mathcal{D}_{\text{CN}}$, learning rate η, λ, γ, total steps T, warm-up steps T_{w}, controller network training steps T_{start} and T_{end}
2: **Initialization:** Construct ZIGs $\mathcal{G}$ of $\mathcal{M}$. Build controller network with weight $\mathcal{W}$ based on the size of $\mathcal{G}$. $\mathbf{w}$ is initialized as $\{0,1\}^{|\mathcal{G}|}$
3: **for** $t = 1, 2 \ldots, T$ **do**
4: **for** a mini-batch (x, y) in D **do**
5: Compute the stochastic gradient estimator $\nabla_{\mathcal{M}} \mathcal{L}(\mathcal{M})$ in Eq.
6: Update model weights $\mathcal{M}$ with any stochastic optimizer.
7: **if** $T \geq T_{\text{w}}$ **then**
8: Perform projection operator and update following Eq. or Eq. on ZIGs with $\mathbf{w}$.
9: **end if**
10: **end for**
11: **if** $T_{start} \leq T \leq T_{end}$ **then**
12: $\mathcal{W}, \mathbf{w} \leftarrow \text{CN-Update}(\mathcal{M}, \mathcal{W}, \mathbf{w}, \mathcal{D}_{\text{CN}})$
13: **end if**
14: **end for**
15: **Output:** Directly remove pruned structures with mask $\mathbf{w}$ and construct a compressed model.

Similar to AGN in SNAP.

$$\min_{\mathcal{M}} \mathcal{J}(\mathcal{M}) := \mathcal{L}(\mathcal{M}) + g(\mathcal{M})$$
$$= \mathcal{L}\big(f(x; \mathcal{M}), y\big) + \sum_{g \in \mathcal{G}} \lambda_g \left\| [\mathcal{M}]_g \right\|$$

Upper: Task loss with selected regularizations on ZIGs.

Lower: Standard Proximal Gradients for Group Lasso

$$\text{prox}_{\eta\lambda_g}([\boldsymbol{z}]_g) = \begin{cases} [\boldsymbol{z}]_g - \eta\lambda_g \frac{[\boldsymbol{z}]_g}{\|[\boldsymbol{z}]_g\|_2}, \\ \text{if } \| [\boldsymbol{z}]_g \| \geq \alpha\lambda_g, \\ 0, \text{ otherwise .} \end{cases}$$

$$\left[\text{Proj}_{\mathcal{S}_k}^{HS}(\boldsymbol{z})\right]_g := \begin{cases} 0 & \text{if } [\boldsymbol{z}]_g^\top [\mathcal{M}]_g < \epsilon \left\| [\mathcal{M}]_g \right\|^2 \\ [\boldsymbol{z}]_g & \text{otherwise.} \end{cases}$$

Half Space Projection [Chen, Tianyi, et al NeurIPS 2021]: Set parameters to zero, if the updated parameters and the original ones are close.

Controller Network

Layer Type	Shape
Input	$\lvert\mathcal{B}\rvert \times 64$
Bi-GRU	$64 \times 128 \times 2$
LayerNorm + ReLU	256
Linear Layer	$256 \times \lvert B_i\rvert, B_i \subset \mathcal{B}$
Concatenate	$\lvert\mathcal{G}\rvert$
Gumbel-Sigmoid	-
Round $\rightarrow \mathbf{w}$	-

We make efforts to reduce extra computational overhead due to the use of controller network. We only use 5% data to train it. Controller Network uses bi-directional gated recurrent units (GRU). It is followed by linear layers. To reduce the training costs, we divide ZIGs into multiple disjoint blocks and the sum of $|B_i| = |G|$. The block is one layer in ResNet models or one InvertedResidual block in MobileNetv2. We stop the training of CN in the half of total step.

$$\mathbf{w} = \text{round}(\text{sigmoid}((o + s + b)/\tau))$$

where sigmoid(·) is the sigmoid function, round(·) is the rounding function, s is sampled from Gumbel distribution ($s \sim$ Gumbel(0, 1)), b and τ are constants with values as 3.0 and 0.4, respectively.

Experiment – CIFAR

Dataset	Architecture	Method	Baseline Acc	Pruned Acc	Δ-Acc	Pruned FLOPs
CIFAR-10	ResNet-18	OTOv2 [17]	93.02 %	92.86%	-0.16%	79.7%
		ATO (ours)	94.41%	**94.51%**	**+ 0.10%**	**79.8%**
	ResNet-56	DCP-Adapt [119]	93.80%	93.81%	+0.01%	47.0%
		SCP [50]	93.69%	93.23%	−0.46%	51.5%
		FPGM [41]	93.59%	92.93%	−0.66%	52.6%
		SFP [40]	93.59%	92.26%	−1.33%	52.6%
		FPC [39]	93.59%	93.24%	−0.25%	52.9%
		HRank [73]	93.26%	92.17%	−0.09%	50.0%
		DMC [31]	93.62%	92.69%	+0.07%	50.0%
		GNN-RL [110]	93.49%	93.59%	+0.10%	54.0%
		ATO (ours)	93.50%	**93.74%**	**+ 0.24%**	**55.0%**
		ATO(ours)	93.50%	93.48 %	−0.02%	**65.3%**
	MobileNetV2	Uniform [119]	94.47%	94.17%	−0.30%	26.0%
		DCP [119]	94.47%	94.69%	+0.22%	26.0%
		DMC [31]	94.23%	94.49%	+0.26%	40.0%
		SCOP [96]	94.48%	94.24%	-0.24%	40.3%
		ATO (ours)	94.45%	**94.78%**	**+0.33%**	**45.8%**
CIFAR-100	ResNet-18	OTOv2 [17]	-	74.96%	-	39.8%
		ATO (ours)	77.95%	**76.79%**	**−0.07%**	**40.1%**
	ResNet-34	OTOv2 [17]	-	76.31%	-	49.5%
		ATO (ours)	78.43 %	**78.54 %**	**+0.11%**	**49.5%**

Experiment - ImageNet

Architecture	Method	Base Top-1	Base Top-5	Pruned Top-1 (Δ Top-1)	Pruned Top-5 (Δ Top-5)	Pruned FLOPs
ResNet-34	FPGM [41]	73.92%	91.62%	72.63% (−1.29%)	91.08% (−0.54%)	41.1%
	Taylor [82]	73.31%	-	72.83% (−0.48%)	-	24.2%
	DMC [31]	73.30%	91.42%	72.57% (−0.73%)	91.11% (−0.31%)	43.4%
	SCOP [96]	73.31%	91.42%	72.62% (−0.69%)	90.98% (−0.44%)	**44.8%**
	ATO (ours)	73.31%	91.42%	**72.92%** (**−0.39%**)	**91.15%** (**−0.27%**)	44.1%
ResNet-50	DCP [119]	76.01%	92.93%	74.95% (−1.06%)	92.32% (−0.61%)	55.6%
	CCP [86]	76.15%	92.87%	75.21% (−0.94%)	92.42% (−0.45%)	54.1%
	FPGM [41]	76.15%	92.87%	74.83% (−1.32%)	92.32% (−0.55%)	53.5%
	ABCP [74]	76.01%	92.96%	73.86% (−2.15%)	91.69% (−1.27%)	54.3%
	DMC [31]	76.15%	92.87%	75.35% (−0.80%)	92.49% (−0.38%)	55.0%
	Random-Pruning [66]	76.15%	92.87%	75.13% (−1.02%)	92.52% (−0.35%)	51.0%
	DepGraph [28]	76.15%	-	75.83% (−0.32%)	-	51.7%
	DTP [71]	76.13%	-	75.55% (−0.58%)	-	**56.7%**
	ATO (ours)	76.13%	92.86%	**76.59%** (**+0.46%**)	**93.24%** (**+0.38%**)	55.2%
	DTP [71]	76.13%	-	75.24 % (−0.89%)	-	60.9%
	OTOv2 [17]	76.13%	92.86%	75.20% (−0.93%)	92.22% (−0.66%)	**62.6%**
	ATO (ours)	76.13%	92.86%	**76.07%** (**−0.06%**)	**92.92%** (**+0.06%**)	61.7%
	DTP [71]	76.13%	-	74.26% (−1.87%)	-	67.3%
	OTOv1 [16]	76.13%	92.86%	74.70% (−1.43%)	92.10% (−0.76%)	64.5%
	OTOv2 [17]	76.13%	92.86%	74.30% (−1.83%)	92.10% (−0.76%)	**71.5%**
	ATO (ours)	76.13%	92.86%	**74.77%** (**−1.36%**)	**92.25%** (**−0.61%**)	71.0%
MobileNet-V2	Uniform [89]	71.80%	91.00%	69.80% (−2.00%)	89.60% (−1.40%)	30.0%
	AMC [42]	71.80%	-	70.80% (−1.00%)	-	30.0%
	CC [70]	71.88%	-	70.91% (−0.97%)	-	28.3%
	MetaPruning [76]	72.00%	-	71.20% (−0.80%)	-	**30.7%**
	Random-Pruning [66]	71.88%	-	70.87% (−1.01%)	-	29.1%
	ATO (ours)	71.88%	90.29%	**72.02%** (**+0.14%**)	**90.19%** (**−0.10%**)	30.1%

Experiment

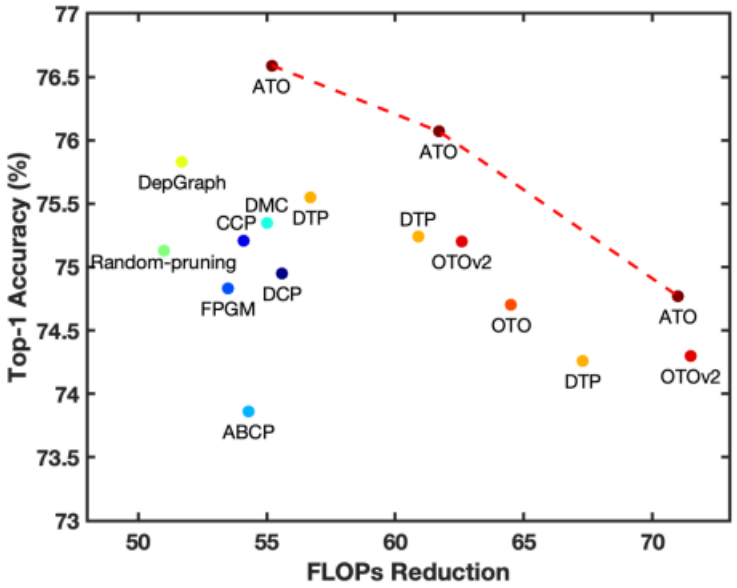


Figure ResNet50 on ImageNet

Table 1. summary of ATO and existing methods

Method	ATO	OTOs	Others
Training cost	Low	Low	High
Addition fine-tuning	No	No	Yes
Optimizer design	Dynamic	Static	Static
Convergence guarantee	√	╳	-
Gradient projection	General	(D)HSPG	-

ATO has a dynamic optimizer design (dynamic regularization for ZIGs). ATO is not restricted to gradient projections. ATO provides a comprehensive convergence guarantee compared to OTOs.

ATO outperforms OTO and OTOv2 with different pruning rates.