3DV 2020

RANP: Resource Aware Neuron Pruning at Initialization for 3D CNNs

(Oral Presentation)

Zhiwei Xu[1,3], Thalaiyasingam Ajanthan[1], Vibhav Vineet[2], Richard Hartley[1]

[1] Australian National University (ANU) and Australian Centre for Robotic Vision (ACRV)
[2] Microsoft Research, Redmond, USA
[3] Data61, CSIRO, Canberra, Australia




Problem Setup

Generally, 3D CNNs suffer from

- Expensive computational complexity
- High requirement of GPU memory
- Infeasible for large-scale applications
- Unfriendly to resource-limited devices

=> Network pruning is a popular and high-efficient approach.

Existing Solutions

Possible adoptions from 2D CNN pruning methods

- Parameter pruning: sparse filters but no significant resource reductions
- Neuron pruning: prune-retrain manner and at test time

Existing 3D CNN pruning methods

- Sparse convolution: specified to sparse data, not generalized to dense data
- Neuron / channel pruning: slow due to the prune-retrain manner

Our Solution (RANP)

- **Large reductions** of FLOPS and memory of 3D CNNs

  50%-95% FLOPs reduction and 35%-80% memory reduction

- **Single-shot** pruning at initialization
- **Scalability** by pruning with a small spatial size and training with a large one
- **Transferability** by pruning on a dataset and training on another one
- **Lightweight** training on a single GPU
- **Fast training** with increased batch size
- **Easy adaption** to 2D CNNs

Our Solution (RANP)

Brief introduction of SNIP (aims at 2D parameter pruning)

Parameter importance by connection sensitivity with relaxed binary masks

mini-batch

data point

Objective function: $$\min_{\mathbf{c},\mathbf{w}} L(\mathbf{c}\odot\mathbf{w};\mathcal{D}) = \min_{\mathbf{c},\mathbf{w}} \frac{1}{S}\sum_{i=1}^{S} \ell\left(\mathbf{c}\odot\mathbf{w}, (\mathbf{x}_i, \mathbf{y}_i)\right) ,$$

parameter number

$$\text{s.t.} \quad \mathbf{w}\in\mathbb{R}^m, \ \mathbf{c}\in\{0,1\}^m, \ \|\mathbf{c}\|_0 \leq \kappa$$

masks on connection sensitivity

Parameter importance: $$s_j = \frac{|g_j(\mathbf{w};\mathcal{D})|}{\sum_{k=1}^{m}|g_k(\mathbf{w};\mathcal{D})|}, \text{where } g_j(\mathbf{w};\mathcal{D}) = \left.\frac{\partial L(\mathbf{c}\odot\mathbf{w};\mathcal{D})}{\partial c_j}\right|_{\mathbf{c}=\mathbf{1}}$$

Then, retain top-*k* parameters by top-*k* largest parameter importance.

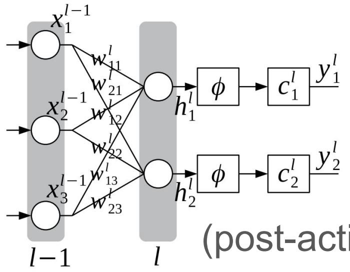
"SNIP: Single-shot network pruning based on connection sensitivity", ICLR 2019

Our Solution (RANP)

Drawbacks of SNIP for 3D CNNs

- Sparse filters by pruning parameters cannot yield significant resource reductions while huge resource consumption tackles the usage of 3D CNNs.
- A sparse filter cannot reduce the number of features of hidden layers, which cause main memory consumption in 3D CNNs, unless all parameters in a neuron are pruned which is uncertain.

Our Solution (RANP)



(post-activation)

Neuron importance

Neuron function with activation: $\mathbf{x}^l = \phi\left(\mathbf{h}^l\right)$, where $\mathbf{h}^l = \mathbf{W}^l\mathbf{x}^{l-1} + \mathbf{b}^l$

neuron number at layer *l*

With a neuron mask (post-activation): $\mathbf{x}^l = \mathbf{c}^l \odot \phi\left(\mathbf{h}^l\right)$, where $\mathbf{c}^l \in \{0,1\}^{N_l}$, $\forall l \in \mathcal{K}$

---

vanilla NI: $s_u^l = \left|\left.\frac{\partial L\left(\mathbf{c},\mathbf{w};\mathcal{D}\right)}{\partial c_u^l}\right|_{\mathbf{c}=\mathbf{1}}\right|$, where $\left.\frac{\partial L\left(\mathbf{c},\mathbf{w};\mathcal{D}\right)}{\partial c_u^l}\right|_{\mathbf{c}=\mathbf{1}} = \sum_{v=1}^{N_{l-1}} \left.\frac{\partial L\left(\mathbf{c}\odot\mathbf{w};\mathcal{D}\right)}{\partial c_{uv}^l}\right|_{\mathbf{c}=\mathbf{1}}$

weighted NI: $\tilde{s}_u^l = \frac{\max_{k=1}^K \bar{s}^k}{\bar{s}^l} s_u^l$, where $\bar{s}^k = \frac{1}{N_k}\sum_{u=1}^{N_k} s_u^k$, $\forall k \in \mathcal{K}$

resource aware NI: $\hat{s}_u^l = \left(1 + \lambda\,\mathrm{softmax}(-\tau_l)\right)\tilde{s}_u^l = \left(1 + \lambda\frac{e^{-\tau_l}}{\sum_{k=1}^K e^{-\tau_k}}\right)\tilde{s}_u^l$

layerwise resource constraint (FLOPs or memory)

Experiments

We evaluated RANP on two 3D tasks

- *3D semantic segmentation*

  Datasets: sparse data: ShapeNet (sparse point clouds)

  dense data: BraTS'18 (medical images)

  Models: 3D-UNets (15-layer and 23-layer)

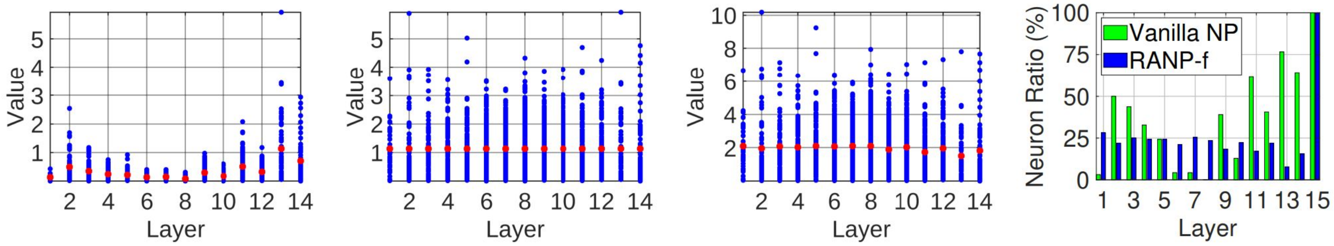
- *Video action classification*

  Dataset: UCF101

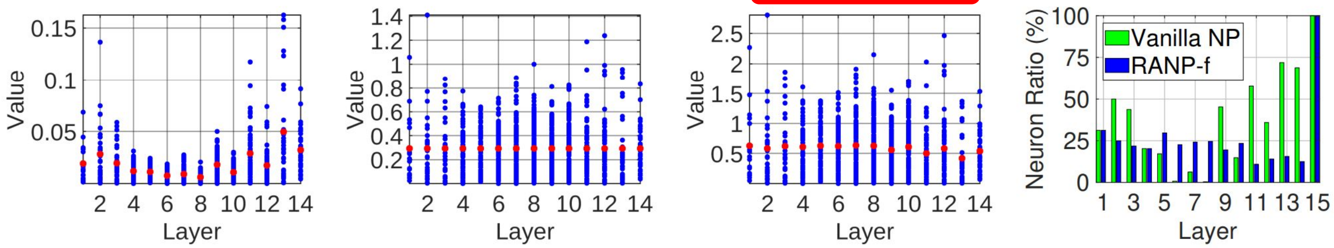
  Models: MobileNetV2 and I3D

Experiments

- Effects of resource constraint on balancing layerwise neuron importance



(a) ShapeNet, Vanilla NP Eq. 6 (b) ShapeNet, Weighted NP Eq. 9 (c) ShapeNet, RANP-f Eq. 10 (d) ShapeNet, Retain ratio

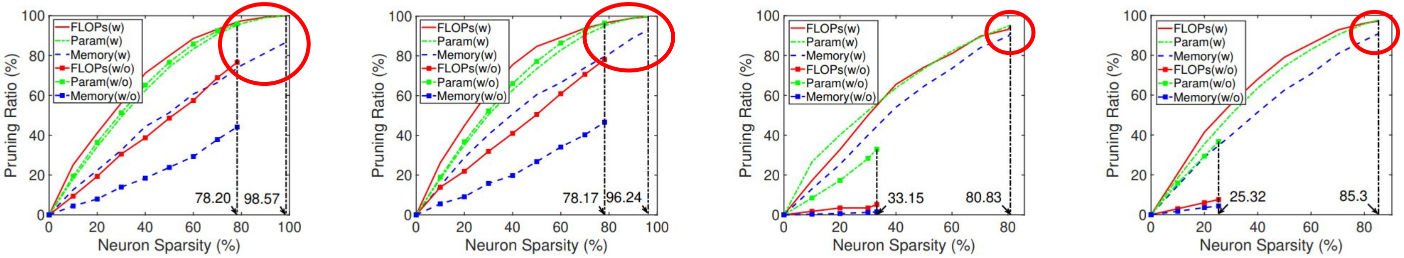


(e) BraTS’18, Vanilla NP Eq. 6 (f) BraTS’18, Weighted NP Eq. 9 (g) BraTS’18, RANP-f Eq. 10 (h) BraTS’18, Retain ratio

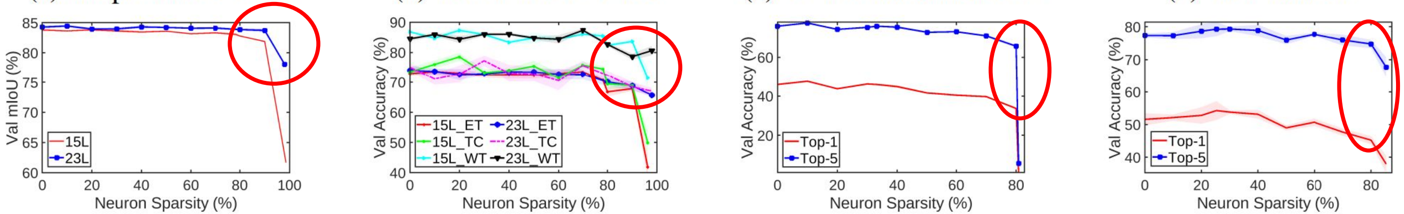
From vanilla NP to weighted NP to RANP NP (weighted NP with resource constraint of FLOPs).

Experiments

- Pruning ability & neuron sparsity



(a) ShapeNet/3D-UNet (b) BraTS'18/3D-UNet (c) UCF101/MobileNetV2 (d) UCF101/I3D



(e) ShapeNet/3D-UNet (f) BraTS'18/3D-UNet (g) UCF101/MobileNetV2 (h) UCF101/I3D

With minimal accuracy loss, much more resources can be reduced with (w) reweighting by RANP-f than without (w/o) it by vanilla NP. (a)-(d) are resources reductions (w) and (w/o) reweighting; (e)-(h) are accuracy by pruning sparsity.

Experiments

- Strong pruning ability of RANP compared with others

Dataset	Model		Manner	Sparsity(%)	Param(MB)	GFLOPs	Memory(MB)	Metrics(%)		
								mIoU		
ShapeNet [10]	3D-UNet		Full[5]	0	62.26	237.85	997.00	83.79±0.21		
			SNIP[18] NP	98.98	5.31 (91.5↓)	126.22 (46.9↓)	833.20 (16.4↓)	**83.70±0.20**		
			Random NP		3.05 (95.1↓)	10.36 (95.6↓)	267.95 (73.1↓)	82.90±0.19		
			Layer-wise NP		2.99 (95.2↓)	11.63 (95.1↓)	296.22 (70.3↓)	83.25±0.14		
		ours	Vanilla NP	78.24	2.54 (95.9↓)	55.69 (76.6↓)	557.32 (44.1↓)	83.26±0.14		
		ours	Weighted NP		2.97 (95.2↓)	12.06 (94.9↓)	301.56 (69.8↓)	83.12±0.09		
		ours	RANP-m		3.39 (94.6↓)	**6.68 (97.2↓)**	**214.95 (78.4↓)**	82.35±0.24		
		ours	RANP-f		2.94 (95.3↓)	7.54 (96.8↓)	262.66 (73.7↓)	83.07±0.22		
								ET	TC	WT
BraTS'18 [11], [20]	3D-UNet		Full[5]	0	15.57	478.13	3628.00	72.96±0.60	73.51±1.54	86.79±0.35
			SNIP[18] NP	98.88	1.09 (93.0↓)	233.11 (51.2↓)	2999.64 (17.3↓)	**73.33±1.89**	71.98±2.15	**86.44±0.39**
			Random NP		0.75 (95.2↓)	22.59 (95.3↓)	817.59 (77.5↓)	67.27±0.99	71.62±1.20	74.16±1.33
			Layer-wise NP		0.75 (95.2↓)	24.09 (95.0↓)	836.88 (77.0↓)	69.74±1.33	71.49±1.62	86.38±0.39
		ours	Vanilla NP	78.17	0.55 (96.5↓)	104.50 (78.1↓)	1936.44 (46.6↓)	71.94±1.68	69.39±2.29	84.68±0.78
		ours	Weighted NP		0.79 (95.0↓)	22.40 (95.3↓)	860.64 (76.3↓)	71.50±0.63	**75.05±1.19**	84.05±0.65
		ours	RANP-m		0.87 (94.4↓)	**13.47 (97.2↓)**	**506.97 (86.0↓)**	66.70±2.94	62.99±2.38	82.90±0.41
		ours	RANP-f		0.76 (95.1↓)	16.97 (96.5↓)	729.11 (80.0↓)	70.73±0.66	74.50±1.05	85.45±1.06
								Top-1	Top-5	
UCF101 [40]	MobileNetV2		Full[21]	0	9.47	0.58	157.47	47.08±0.72	76.68±0.50	
			SNIP[18] NP	86.26	3.67 (61.3↓)	0.54 (6.9↓)	155.35 (1.3↓)	45.78±0.04	75.08±0.17	
			Random NP		4.58 (51.6↓)	0.34 (41.4↓)	106.68 (32.3↓)	44.74±0.36	74.69±0.58	
			Layer-wise NP		4.56 (51.8↓)	0.33 (43.1↓)	106.92 (32.1↓)	44.90±0.36	75.54±0.34	
		ours	Vanilla NP	33.15	6.35 (32.9↓)	0.55 (5.2↓)	155.17 (1.5↓)	**46.32±0.79**	75.42±0.60	
		ours	Weighted NP		4.82 (49.1↓)	0.30 (48.3↓)	100.33 (36.3↓)	46.19±0.51	75.72±0.30	
		ours	RANP-m		4.87 (48.6↓)	0.27 (53.4↓)	**84.51 (46.3↓)**	45.11±0.41	75.53±0.37	
		ours	RANP-f		4.83 (49.0↓)	**0.26 (55.2↓)**	88.01 (44.1↓)	45.87±0.41	**75.75±0.30**	
	I3D		Full[22]	0	47.27	27.88	201.28	51.58±1.86	77.35±0.63	
			SNIP[18] NP	81.09	30.06 (36.4↓)	26.31 (5.6↓)	195.62 (2.8↓)	52.38±3.55	78.32±3.24	
			Random NP		26.36 (44.2↓)	16.45 (41.0↓)	145.07 (27.9↓)	52.42±2.52	79.05±2.06	
			Layer-wise NP		26.67 (43.6↓)	16.93 (39.3↓)	150.95 (25.0↓)	52.77±1.99	78.41±1.07	
		ours	Vanilla NP	25.32	29.93 (36.7↓)	25.76 (7.6↓)	192.42 (4.4↓)	51.57±1.46	78.07±1.34	
		ours	Weighted NP		26.57 (43.8↓)	15.56 (44.2↓)	142.57 (29.2↓)	54.09±0.82	79.26±0.61	
		ours	RANP-m		26.75 (43.4↓)	14.08 (49.5↓)	130.44 (35.2↓)	52.11±3.05	77.54±2.64	
		ours	RANP-f		26.69 (43.5↓)	**13.98 (49.9↓)**	**130.22 (35.3↓)**	**54.27±2.88**	**79.27±2.13**	

All models are trained from scratch for 100 epochs on ShapeNet and UCF101, 200 on BraTS'18. Metrics are calculated by the last 5 epochs. "sparsity" is max parameter sparsity for SNIP NP and max neuron sparsity for others. Overall, our RANP-f performs best with large reductions of main resource consumption (GFLOPs / memory) with negligible accuracy loss.

Experiments

- **Transferability** with Interactive Models

  Pruning a 23-layer 3D-UNet on ShapeNet -> then applied to BraTS'18 training and vice versa.

Manner	ShapeNet mIoU(%)	BraTS'18 ET(%)	BraTS'18 TC(%)	BraTS'18 WT(%)
Full[5]	**84.27±0.21**	**74.04±1.45**	**75.11±2.43**	84.49±0.74
RANP-f(ours)	83.86±0.15	71.13±1.43	72.40±1.48	83.32±0.62
T-RANP-f(ours)	83.25±0.17	72.74±0.69	73.25±1.69	**85.22±0.57**

Transfer learning by 23-layer 3D-UNets interactively pruned and trained between ShapeNet and BraTS'18. Accuracy loss from RANP-f to T-RANP-f is negligible. "T": transferred.

Experiments

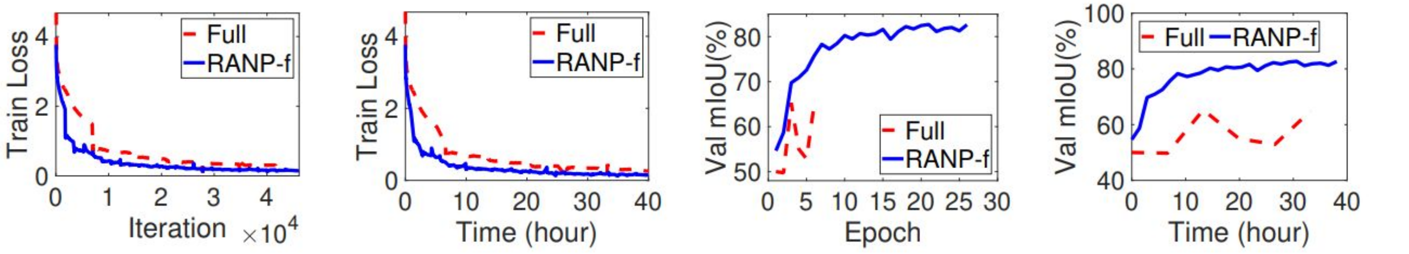
- **Lightweight** Training on a Single GPU

Manner	Layer	Batch	GPU(s)	Sparsity(%)	mIoU(%)
Full	15	12	2	0	83.79±0.21
Full	23	12	2	0	84.27±0.21
RANP-f(ours)	23	12	**1**	80	**84.34±0.21**

ShapeNet: a deeper 23-layer 3D-UNet is achievable on a single GPU with 80% neuron pruning.

Experiments

- **Fast Training** with Increased Batch Size

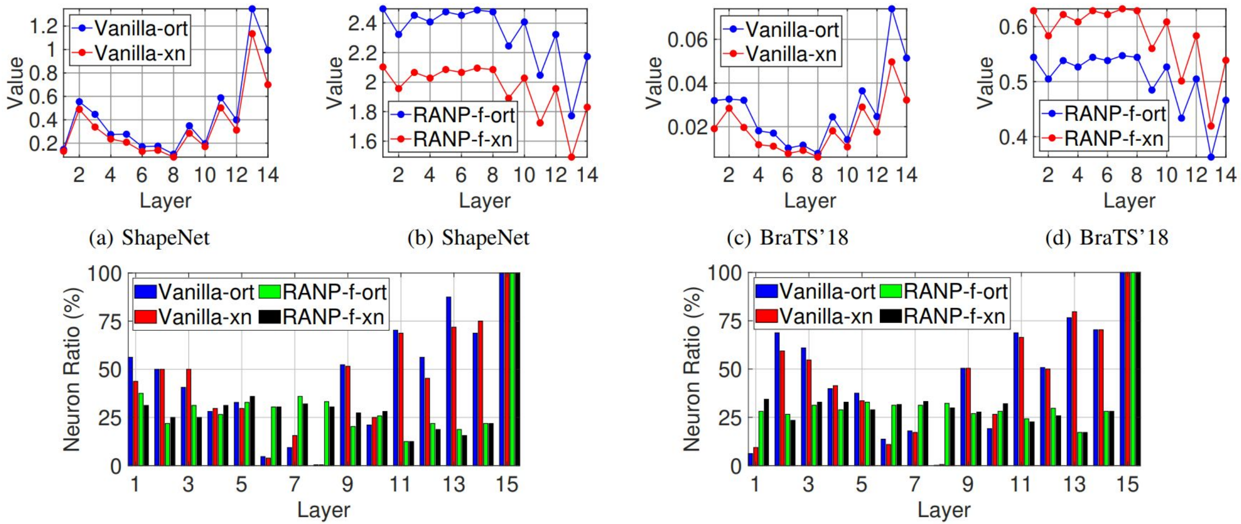


(a) (b) (c) (d)

ShapeNet: a faster convergence on a single GPU with 23-layer 3D-UNet is achievable with increased batch size due to the largely reduced resources by our RANP-f. Batch size is 1 for “Full” and 4 for “RANP-f”. Experiments run for 40 hours.

Experiments (appendix)

3D CNNs: failure of orthogonal initialization in signal propagation for neuron balance



(a) ShapeNet (b) ShapeNet (c) BraTS'18 (d) BraTS'18

(e) ShapeNet (f) BraTS'18

(a)-(d) are neuron importance values. (e)-(f) are neuron retained ratios. Vanilla versions (both orthogonal and Glorot initializations) prune all the neuron in layer 8, leading to network infeasibility while our RANP-f versions have a balanced distribution of retained neurons.

"A signal propagation perspective for pruning neural networks at initialization" ICLR 2020

Conclusion

We proposed effective and efficient Resource Aware Neuron Pruning (RANP)

- High effectiveness on resource reductions *(50%-95% FLOPs and 35%-80% memory)*
- High efficiency (single-shot by pruning at initialization)
- Scalability by pruning with a small spatial size and training with a large one
- Transferability by pruning on a dataset and training on another one
- Lightweight training on a single GPU
- Fast training with increased batch size
- Useful for (not limited to) 3D CNNs

Code of RANP is available at

https://github.com/zwxu064/RANP.git

- Q&A -