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

Zhiwei Xu[1,3] Thalaiyasingam Ajanthan[1] Vibhav Vineet[2] Richard Hartley[1]
[1]Australian National University and Australian Centre for Robotic Vision
[2]Microsoft Research, Redmond, USA
[3]Data61, CSIRO, Australia
{zhiwei.xu,thalaiyasingam.ajanthan,richard.hartley}@anu.edu.au
vibhav.vineet@microsoft.com

We first provide the pseudocode of our RANP algorithm, then discuss our selection of MPMG-sum as vanilla NP, and justify our reweighting scheme against orthogonal initialization with more ablation experiments.

A. Pseudocode of RANP Procedures

In Alg. 1, we provide the pseudocode of the pruning procedures of RANP. In Alg. 2, we used a simple half-space method to automatically search for the max neuron sparsity with network feasibility. Note that this searching cannot guarantee a small accuracy loss but merely to decide the maximum pruning capability. The relation between pruning capability and accuracy was studied in the experimental section in the main paper and Table 6.

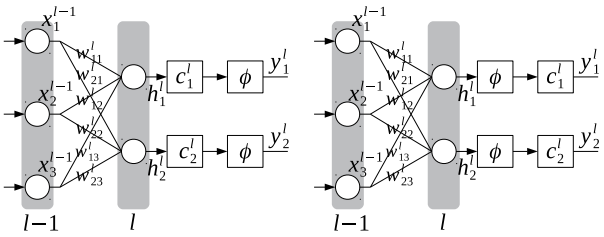
Loss Function and Metrics. Due to the page limitation, we provide loss functions and metrics used in our experiments. Standard cross-entropy function was used as the loss function for ShapeNet and UCF101. For BraTS'18, the weighted function in [4] is

$$L = L_{ce} + \alpha L_{dice} = L_{ce} + \alpha \frac{1}{C} \sum_{i=1}^{C} \frac{2|\mathbf{P}_i \cap \mathbf{G}_i|}{|\mathbf{P}| + |\mathbf{G}|} , \quad (12)$$

where $\alpha = 0.25$ is an empiric weight for dice loss, $\mathbf{P}$ is prediction, $\mathbf{G}$ is ground truth, and C is the number of classes. Meanwhile, ShapeNet accuracy was measured by mean IoU over each part of object category [7] while IoU by $|\mathbf{P} \cap \mathbf{G}|/|\mathbf{P} \cup \mathbf{G}|$ was adopted for BraTS'18. For UCF101 classification, top-1 and top-5 recall rates were used.

B. Impacts of the Activation Function

In the following, we first establish the relation between MPMG and MNMG for calculating neuron importance given a homogeneous activation function $\phi(\cdot)$ that includes but not limited to ReLU used in the 3D CNNs. Then we analyze the impact of such an activation function on the calculation of neuron importance by derivating the mask gradients on post-activations and pre-activations illustrated in Figs. 6b and 6a respectively.



(a) Pre-activations (b) Post-activations

Figure 6: Pre-activations and post-activations, where $\mathbf{x}$ are layer inputs, $\mathbf{w}$ are weights, $\mathbf{c}$ are neuron masks, $\phi(\cdot)$ is an activation function, $\mathbf{h}$ are hidden values, and $\mathbf{y}$ are outputs.

Proposition 1 *For a network activation function $\phi(w)$: $\mathbb{R} \to \mathbb{R}$ being a homogeneous function of degree 1 satisfying $\phi(cw) = c\phi(w), \forall c \geq 0$, the neuron mask gradient equals the sum of parameter mask gradients of this neuron.*

Proof: Given a neuron mask c_1 before the activation function $\phi(\cdot)$ in Fig. 6a and the output of the 1st neuron as y_1^l, we have

$$\begin{aligned} y_1^l &= \phi(c_1^l \odot h_1^l) \\ &= \phi\left(c_1^l \odot \left(x_1^{l-1} w_{11}^l + x_2^{l-1} w_{12}^l + x_3^{l-1} w_{13}^l\right)\right) \\ &= \phi\left(c_1^l x_1^{l-1} w_{11}^l + c_1^l x_2^{l-1} w_{12}^l + c_1^l x_3^{l-1} w_{13}^l\right) . \end{aligned} \quad (13)$$

The gradient of loss L over the neuron mask c_1^l is

Algorithm 1: Pruning Procedures of RANP-[f|m].

Input: Dataset $\mathcal{D} = \{(\mathbf{x}_i, \mathbf{y}_i)\}_{i=1}^{S}$ with B samples per batch, neuron sparsity κ, resource importance $\{\tau_l\}$, coefficient $\lambda > 0$, and parameter masks $\mathbf{c} = \{c_{uv}^l\}$, where layer $l \in \mathcal{K} = \{1, ..., K\}$, and neuron $u \in \mathcal{N}_l = \{1, ..., N_l\}$.

Output: Binary neuron masks $\hat{\mathbf{c}} = \{\hat{c}_u^l\}$.

for batch $t \in \{1, ..., \lfloor S/B \rfloor\}$ **do**
- $\mathcal{D}^t \leftarrow \{(\mathbf{x}_i, \mathbf{y}_i)\}_{i=(t-1)B+1}^{tB}$ ▷mini-batch
- $g_{uv}^l \leftarrow \partial L(\mathbf{c} \odot \mathbf{w}; \mathcal{D}^t)/\partial c_{uv}^l$ ▷parameter mask gradient, Eq. 2
- $g_{uv}^l \leftarrow |g_{uv}^l|$, for MPMG ▷parameter mask importance, Eq. 6
- $\nabla c_{uv}^l \overset{+}{\leftarrow} g_{uv}^l, \forall u \in \mathcal{N}_l, \forall v \in \mathcal{N}_{l-1}$ ▷gradient accumulation

$\nabla c_{uv}^l \leftarrow \nabla c_{uv}^l / \lfloor S/B \rfloor, \forall u \in \mathcal{N}_l, \forall v \in \mathcal{N}_{l-1}, \forall l \in \mathcal{K}$ ▷average on mini-batch

$s_u^l \leftarrow |\sum_{v=1}^{N_{l-1}} \nabla c_{uv}^l|, \forall u \in \mathcal{N}_l, \forall l \in \mathcal{K}$ ▷vanilla neuron importance, Eq. 8

$\bar{s}^l \leftarrow \sum_{u \in \mathcal{N}_l} s_u^l / N_l, \forall l \in \mathcal{K}$ ▷mean neuron importance, Eq. 9

$\tilde{s}_u^l \leftarrow (\max_{j \in \mathcal{K}} \bar{s}^j / \bar{s}^l) s_u^l, \forall u \in \mathcal{N}_l, \forall l \in \mathcal{K}$ ▷weighting, Eq. 9

$\hat{s}_u^l \leftarrow (1 + \lambda e^{-\tau_l} / \sum_{j \in \mathcal{K}} e^{-\tau_j}) \tilde{s}_u^l, \forall u \in \mathcal{N}_l, \forall l \in \mathcal{K}$ ▷reweighting, Eq. 10

$\{\ddot{s}_u\} \leftarrow \text{SortDescending}\left(\{\hat{s}_u^l\}\right), \forall u \in \mathcal{N}_l, \forall l \in \mathcal{K}$ ▷sorting in descending

$\hat{c}_u^l \leftarrow 1[\hat{s}_u^l - \ddot{s}_\kappa \geq 0], \forall u \in \mathcal{N}_l, \forall l \in \mathcal{K}$ ▷binary neuron mask, Eq. 11

Algorithm 2: Auto-Search for Max Neuron Sparsity .

Input: Dataset $\mathcal{D}$, layerwise resource usage $\boldsymbol{\tau}$ w.r.t. FLOPs or memory, coefficient $\lambda > 0$, lower and upper sparsity κ_{min} and κ_{max}, threshold $\delta = 1e-4$. "feasible network" means not all neurons are removed in each layer.

Output: Max neuron sparsity κ^*.

Initilize $\kappa_{min} \leftarrow 0, \kappa_{max} \leftarrow 1$

while $(\kappa_{max} - \kappa_{min} > \delta)$ **do**
- $\kappa = 0.5\,(\kappa_{min} + \kappa_{max})$
- $y = \text{NeuronPruning}(\mathcal{D}, \boldsymbol{\tau}, \lambda, \kappa)$ ▷Alg. 1
- **if** $y == 0$(*feasible network*) **then**
  - $\kappa_{min} \leftarrow \kappa$
- **else**
  - $\kappa_{max} \leftarrow \kappa$

$\kappa^* = \kappa$

$$\begin{aligned}\frac{\partial L}{\partial c_1^l} &= \frac{\partial L}{\partial y_1^l}\frac{\partial y_1^l}{\partial c_1^l} \\ &= \frac{\partial L}{\partial y_1^l}\left(x_1^{l-1}w_{11}^l + x_2^{l-1}w_{12}^l + x_3^{l-1}w_{13}^l\right) .\end{aligned} \tag{14}$$

Meanwhile, if setting masks on weights of this neuron directly, we can obtain

$$y_1^l = \phi(c_{11}^l x_1^{l-1} w_{11}^l + c_{12}^l x_2^{l-1} w_{12}^l + c_{13}^l x_3^{l-1} w_{13}^l) , \tag{15}$$

then the gradient of weight mask, *e.g.*, c_{11}^l, from loss is

$$\frac{\partial L}{\partial c_{11}^l} = \frac{\partial L}{\partial y_1^l}\frac{\partial y_1^l}{\partial c_{11}^l} = \frac{\partial L}{\partial y_1^l} x_1^{l-1} w_{11}^l . \tag{16}$$

Similarly,

$$\begin{aligned}&\frac{\partial L}{\partial c_{11}^l} + \frac{\partial L}{\partial c_{12}^l} + \frac{\partial L}{\partial c_{13}^l} \\ =&\frac{\partial L}{\partial y_1^l}\left(x_1^{l-1}w_{11}^l + x_2^{l-1}w_{12}^l + x_3^{l-1}w_{13}^l\right) .\end{aligned} \tag{17}$$

Clearly, Eq. 14 equals Eq. 17. Hence, the neuron mask gradients can be calculated by parameter mask gradients. To this end, the proof is done.

Furthermore, given such a homogeneous activation function in Prop. 1, the importance of a post-activation equals the importance of its pre-activation. In more detail, for post-activations in Fig. 6b, output y_1^l is

$$\begin{aligned}y_1^l &= c_1^l \odot \phi(h_1^l) \\ &= c_1^l \odot \phi\left(x_1^{l-1}w_{11}^l + x_2^{l-1}w_{12}^l + x_3^{l-1}w_{13}^l\right) .\end{aligned} \tag{18}$$

Since the activation function satisfies $c\phi(w) = \phi(cw)$,

$$y_1^l = \phi(c_1^l x_1^{l-1} w_{11}^l + c_1^l x_2^{l-1} w_{12}^l + c_1^l x_3^{l-1} w_{13}^l) . \tag{19}$$

The neuron importance determined by neuron mask c_1^l is

$$\begin{aligned}\frac{\partial L}{\partial c_1^l} &= \frac{\partial L}{\partial y_1^l}\frac{\partial y_1^l}{\partial c_1^l} \\ &= \frac{\partial L}{\partial y_1^l}\left(x_1^{l-1}w_{11}^l + x_2^{l-1}w_{12}^l + x_3^{l-1}w_{13}^l\right) .\end{aligned} \tag{20}$$

Clearly, Eq. 20 equals Eq. 14. Now, the importance of pre-activations and post-activations is the same given such a homogeneous activation function.

C. Resource Aware Reweighting Scheme

As described in Sec. 4.2 in the main paper, the reweighting of RANP is conducted by first balancing the layer-wise distribution of neuron importance and then adopting resource importance τ_l for layer $l \in \mathcal{K}$ to further reduce resources. Since FLOPs and memory are the main resources of 3D CNNs, τ_l is defined by FLOPs or memory as follows.

Generally, given input dimension of the lth layer $(x_{\text{in}}, x_h, x_w, x_d)$[1], neuron dimension $(f_{\text{out}}, f_{\text{in}}, f_h, f_w, f_d)$, and output dimension $(y_{\text{in}}, y_h, y_w, y_d)$ with $x_{\text{in}} = f_{\text{in}}$ and $f_{\text{out}} = y_{\text{in}}$, the resource importance in terms of FLOPs or memory is defined by

$$\begin{aligned} \text{FLOPs: } \tau_l &= [(f_h f_w f_d + f_h f_w f_d - 1) f_{\text{in}} \\ &\quad + f_{\text{in}} - 1 + 1|_{\text{bias}}] \, y_{\text{in}} y_h y_w y_d \\ &= (2 f_h f_w f_d f_{\text{in}} - 1 + 1|_{\text{bias}}) \, y_{\text{in}} y_h y_w y_d, \quad (21a) \\ \text{Memory: } \tau_l &= y_{\text{in}} y_h y_w y_d, \quad (21b) \end{aligned}$$

where $(f_h f_w f_d)$ is the number of operations of multiplications of filter[2] and layer input, $(f_h f_w f_d - 1)$ is for additions of values from the multiplications, (f_{in}) is for multiplications over all f_{in} filters, $(f_{\text{in}} - 1)$ is for additions of values from all these multiplications, $(1|_{\text{bias}})$ is for an addition when the neuron has a bias, and $(y_{\text{in}} y_h y_w y_d)$ is for all elements of the layer output.

D. More Ablation Study

In this section, we add more experimental results for the analysis of selecting MPMG-sum as vanilla NP, Glorot initialization for network initialization compared with orthogonal initialization [5] to handle the imbalanced layer-wise distribution of neuron importance, and visualization of neuron distribution by RANP for BraTS'18 in addition to that for ShapeNet in the main paper.

Figures in this sections are for 3D-UNets on ShapeNet and BraTS'18 because 3D-UNets used in our experiments typically clarify the neuron imbalance and memory issues and are clear for illustration with a limited number of layers, *i.e.*, 15 layers, while MobileNetV2 and I3D have more than 55 layers but many are not typical 3D convolutional layers with 3^3 kernel size filters.

D.1. MPMG-sum as Vanilla Neuron Pruning

In Sec. 5.2 in the main paper, we select MPMG-sum as vanilla neuron pruning for the trade-off between computational resources and accuracy. To give a comprehensive study of this selection, we demonstrate detailed results of mean, max, and sum operations of MPMG and MNMG in Table 6. Note that we relax the sum operation in Eq. 8 in the main paper to mean, max, and sum.

In Table 6, we aim at obtaining the maximum neuron sparsity due to the target of reducing the computational resources at an extreme sparsity level with minimal accuracy loss. Vividly, for *ShapeNet*, MPMG-sum achieves the largest maximum neuron sparsity 78.24% among all with only ∼0.53% accuracy loss. Differently, for *BraTS'18*, MNMG-sum has the largest maximum neuron sparsity 81.32%; however, the accuracy loss can reach up to ∼8.48%. In contrast, while MPMG-sum has the second-largest maximum neuron sparsity 78.17%, the accuracy loss is much smaller than MNMG-sum. For *UCF101*, it is surprising that many manners have low accuracy. As we analyse the reason in the footnote in Table 6, with the extreme neuron sparsity, some layers of the pruned networks have only 1 neuron retained, losing sufficient features for learning, and thus, leading to low accuracy.

Hence, considering the comprehensive performance of reducing resources and maintaining the accuracy, MPMG-sum is selected as vanilla NP. Note that any neuron sparsity greater than the maximum neuron sparsity will make the pruned network infeasible by pruning the whole layer(s).

D.2. Initialization for Neuron Imbalance

The imbalanced layer-wise distribution of neuron importance hinders pruning at a high sparsity level due to the pruning of the whole layer(s). For 2D classification tasks in [5], orthogonal initialization is used to effectively solve this problem for balancing the importance of parameters; but it does not improve our neuron pruning results in 3D tasks and even leads to a poor pruning capability with a lower maximum neuron sparsity than Glorot initialization [3]. This is briefly mentioned in Sec. 4.1 in the main paper. Here, we compare the resource reducing capability using Glorot initialization and orthogonal initialization.

Resource reductions. In Table 7, vanilla neuron pruning (*i.e.*, MPMG-sum) with Glorot initialization, *i.e.*, vanilla-xn, achieves smaller FLOPs and memory consumption than those with orthogonal initialization, *i.e.*, vanilla-ort, except FLOPs with 3D-UNet on ShapeNet and I3D on UCF101. This exception of I3D on UCF101 is possibly caused by the high ratio of 1^3 kernel size filters in I3D, *i.e.*, 37 out of 57 convolutional layers, because those 1^3 kernel size filters can be regarded as 2D filters on which orthogonal initialization

[1] The dimension order follows that of PyTorch.

[2] Here, we refer a 3D filter with dimension (f_h, f_w, f_d).

[3] For MobileNetV2 pruned by MPMG-mean, MPMG-max, MNMG-max, and MNMG-sum, the accuracy is very low because 1) the neuron sparsity here is the extreme (largest) value, a larger one will make network infeasible by removing whole layer(s) and 2) the distribution of neuron importance is rather imbalanced possibly caused by the high mixture of 1^3 kernels and 3^3 in MobileNetV2.

In the pruned networks, we observe that, for MPMG-mean, MPMG-max, and MNMG-max, the last convolutional layer has only 1 neuron retained; for MNMG-sum, 2 convolutional layers have only 1 neuron retained. Note that, this imbalance issue can be greatly alleviated by the reweighting of our RANP, while we select MPMG-sum as vanilla NP merely according to the results in Table 6.

Table 6: More results of vanilla NP in addition to Table 1 in the main paper. **Main resource consumption** (GFLOPs and memory) are considered but not parameters whose resource consumption is much smaller than memory. Among the neuron pruning methods, we marked bold **the best** and underlined the second best. Overall, we selected MPMG-sum as vanilla NP and the corresponding neuron sparsity for large resource reductions with small accuracy loss.

Dataset	Model	Manner	Sparsity(%)	Param(MB)	GFLOPs	Memory(MB)	Metrics(%)		
								mIoU	
ShapeNet	3D-UNet	Full[2]	0	62.26	237.85	997.00		83.79±0.21	
		MPMG-mean	68.10	5.08	110.14	819.97		83.33±0.18	
		MPMG-max	70.24	4.54	107.38	809.88		**83.79±0.10**	
		MPMG-sum	78.24	2.54	**55.69**	**557.32**		83.26±0.14	
		MNMG-mean	63.03	4.23	112.95	834.98		83.46±0.13	
		MNMG-max	73.93	3.67	103.57	796.44		83.51±0.08	
		MNMG-sum	66.93	4.29	100.34	783.14		83.65±0.02	
							ET	TC	WT
BraTS'18	3D-UNet	Full[2]	0	15.57	478.13	3628.00	72.96±0.60	73.51±1.54	86.79±0.35
		MPMG-mean	65.64	1.48	226.86	3038.27	73.51±0.82	73.28±1.14	87.15±0.43
		MPMG-max	75.78	0.83	189.43	2812.53	**73.67±0.98**	72.73±1.70	86.44±0.71
		MPMG-sum	78.17	0.55	104.50	1936.44	71.94±1.68	69.39±2.29	84.68±0.78
		MNMG-mean	63.85	1.08	176.76	2790.64	73.35±0.70	**73.38±0.94**	**87.21±0.38**
		MNMG-max	80.05	0.59	169.99	2676.05	72.52±1.91	72.40±1.74	84.63±0.60
		MNMG-sum	81.32	0.35	**73.50**	**1933.20**	64.48±1.10	68.47±1.59	80.71±1.07
							Top-1	Top-5	
UCF101	MobileNetV2	Full[6]	0	9.47	0.58	157.47	47.08±0.72	76.68±0.50	
		MPMG-mean	26.31	4.39	0.55	156.00	2.98±0.14 [3]	14.04±0.14	
		MPMG-max	29.48	3.96	0.54	155.38	3.49±0.12	13.64±0.10	
		MPMG-sum	33.15	6.35	0.55	155.17	**46.32±0.79**	**75.42±0.60**	
		MNMG-mean	38.91	2.79	0.50	147.69	29.13±0.92	62.93±1.37	
		MNMG-max	50.33	2.59	0.53	153.45	2.84±0.06	13.40±0.23	
		MNMG-sum	39.89	4.66	**0.43**	**120.01**	1.03±0.00	5.76±0.00	
	I3D	Full[1]	0	47.27	27.88	201.28	51.58±1.86	77.35±0.63	
		MPMG-mean	16.47	31.57	26.50	196.51	51.88±2.00	77.98±1.46	
		MPMG-max	19.83	30.06	26.31	195.62	**52.44±1.25**	**78.08±1.27**	
		MPMG-sum	25.32	29.93	25.76	192.42	51.57±1.46	78.07±1.34	
		MNMG-mean	35.36	16.69	**15.37**	**124.85**	49.26±0.96	75.70±1.49	
		MNMG-max	40.27	17.86	23.73	184.77	44.90±1.19	74.43±1.26	
		MNMG-sum	32.87	20.00	16.03	125.17	46.90±1.26	74.02±1.25	

can effectively deal with [5]. While this ratio is also high in MobileNetV2, *i.e.*, 34 out of 52 convolutional layers, it is unnecessary to have the same problem as I3D since it is also affected by the number of neurons in each layer. Note that since 3D-UNets used are all with 3^3 kernel size filters, the orthogonal initialization for 3D-UNet in most cases is inferior to Glorot initialization according to our experiments. Meanwhile, in Table 7, this gap between vanilla-ort and vanilla-xn is very small on MobileNetV2 and I3D.

Nevertheless, with RANP-f and Glorot initialization, *i.e.*, RANP-f-xn, more FLOPs and memory can be reduced than using orthogonal initialization, *i.e.*, RANP-f-ort.

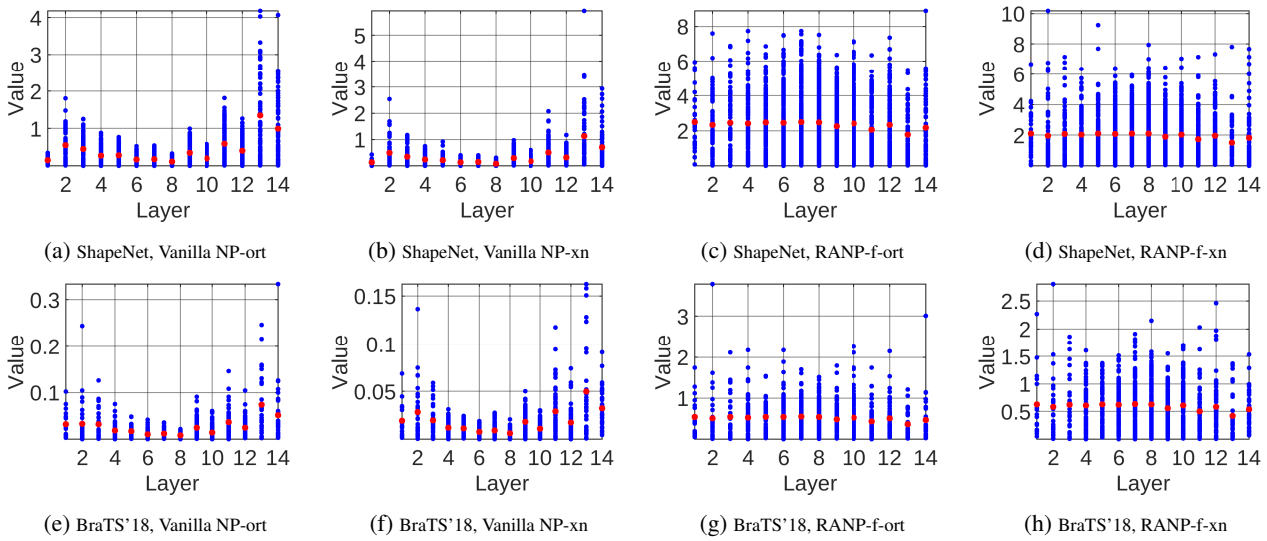
Balance of Neuron Importance Distribution. More importantly, with reweighting by RANP in Fig. 7, the values of neuron importance are more balanced and stable than those of vanilla neuron importance. This can largely avoid network infeasibility without pruning the whole layer(s).

Now, we analyse the neuron distribution from the observation of neuron importance values and network structures. *Fig. 8 illustrates a detailed comparison between orthogonal and Glorot initialization by each two subfigures in column of Fig. 7.* In Figs. 8a-8c, vanilla neuron importance by Glorot initialization is more stable and compact than that by orthogonal initialization. After applying the reweighting scheme of RANP-f, the importance tends to be in a similar tendency, shown in Figs. 8b-8d. Consequently, in Figs. 8e-8f, neuron ratios are more balanced after the reweighting than without reweighting, especially the 8th layer. Thus, we choose Glorot initialization as network initialization. Note that we adopt the same neuron sparsity for these two initialization experiments in Table 7 and Fig. 8.

D.3. Visualization of Balanced Neuron Distribution by RANP

In Fig. 9, neuron importance by MPMG-sum is more balanced than by MNMG-sum, which avoids pruned networks by MPMG-sum to be infeasible, that is at least 1 neuron will be retained in each layer.

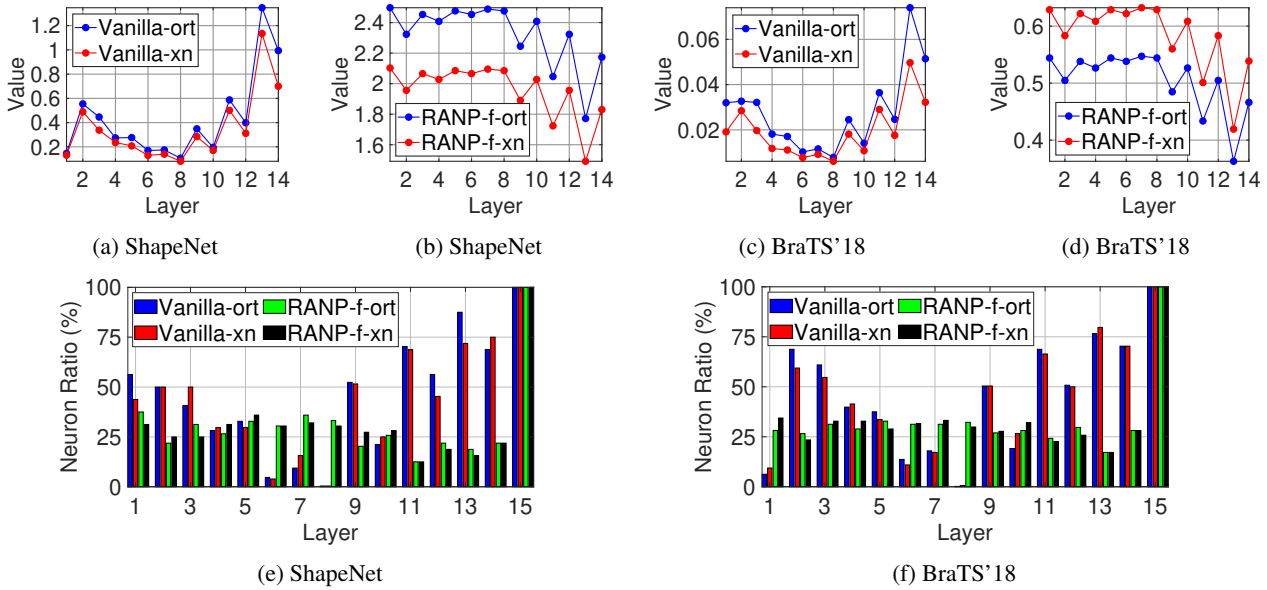
In addition to the distribution of retained neuron ratios in Fig. 2 in the main paper for ShapeNet, which is also shown in the first row of Fig. 10, the last row of Fig. 10 is for BraTS'18. Moreover, Fig. 11 illustrates the distribution of



(a) ShapeNet, Vanilla NP-ort (b) ShapeNet, Vanilla NP-xn (c) ShapeNet, RANP-f-ort (d) ShapeNet, RANP-f-xn

(e) BraTS'18, Vanilla NP-ort (f) BraTS'18, Vanilla NP-xn (g) BraTS'18, RANP-f-ort (h) BraTS'18, RANP-f-xn

Figure 7: Neuron importance of 15-layer 3D-UNet by MPMG-sum with orthogonal and Glorot initialization. Blue: neuron values; red: mean values. By vanilla NP, orthogonal initialization does not result in a balanced neuron importance distribution compared to Glorot initialization whereas by our RANP-f, the values are more balanced and resource aware on FLOPs, enabling pruning at the extreme sparsity.



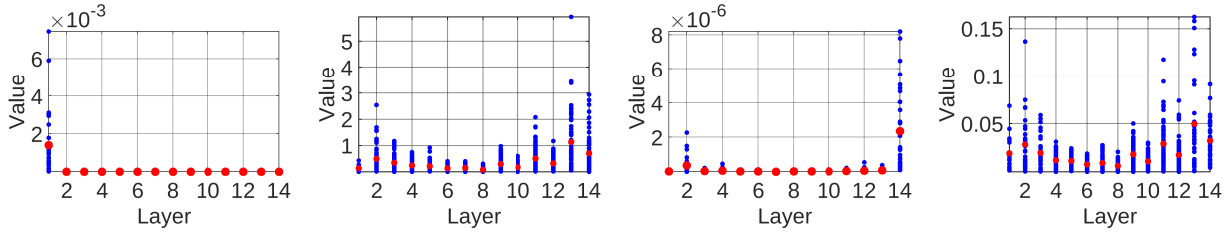
(a) ShapeNet (b) ShapeNet (c) BraTS'18 (d) BraTS'18

(e) ShapeNet (f) BraTS'18

Figure 8: Comparison of neuron distribution with orthogonal and Glorot initialization before and after reweighting. (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.

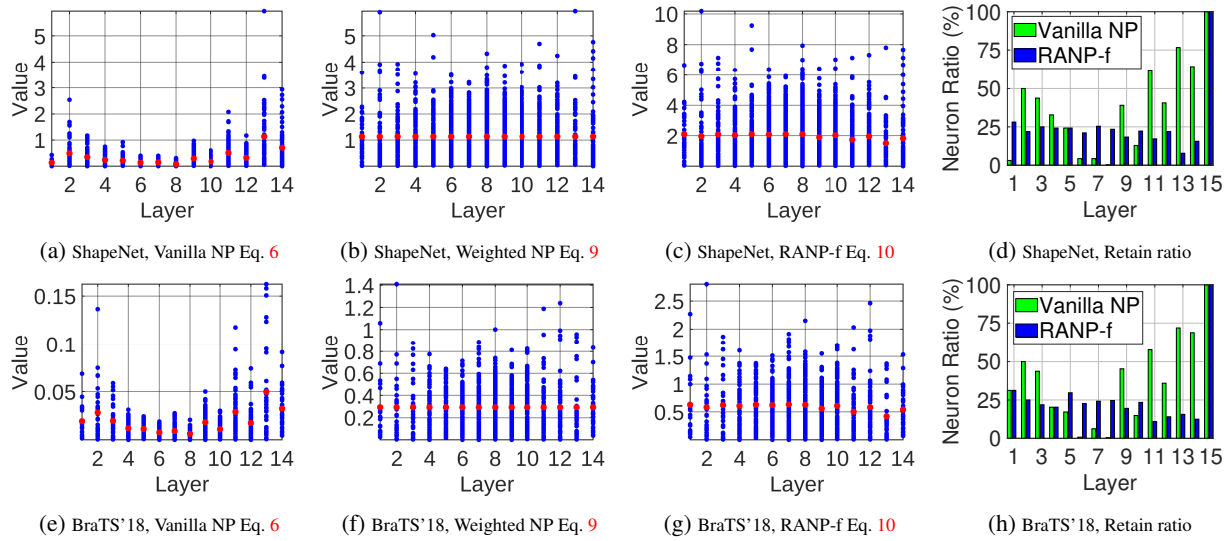
neurons retained in each layer by vanilla neuron pruning (*i.e.*, vanilla NP) and RANP-f compare to the full network.

Clearly, upon pruning, neurons in each layer are largely reduced except the last layer where all neurons are retained for the number of segmentation classes. In Fig. 11, vanilla NP has very few neurons in, *e.g.*, the 8th layer, resulting in low accuracy or network infeasibility. By contrast, the neuron distribution by RANP-f is more balanced to improve the pruning capability.



(a) ShapeNet, MNMG-sum (b) ShapeNet, MPMG-sum (c) BraTS'18, MNMG-sum (d) BraTS'18, MPMG-sum

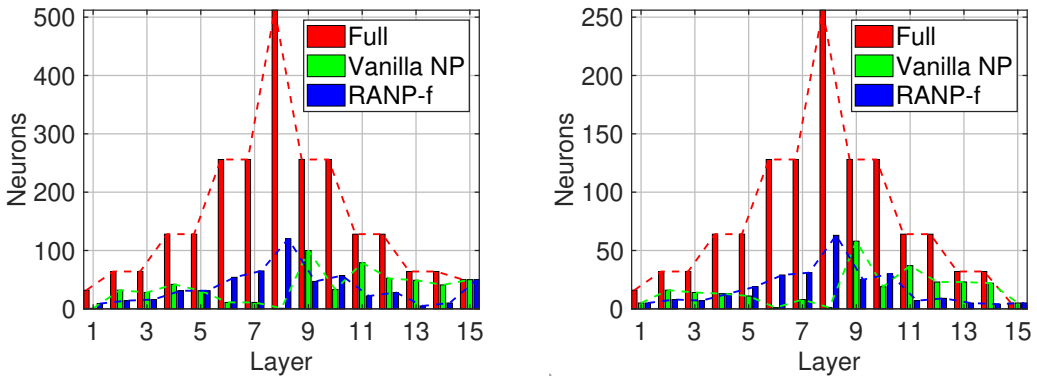
Figure 9: MNMG-sum and MPMG-sum on ShapeNet and BraTS'18 with max neuron sparsity in Table 6. Blue: neuron values; red: mean values. Clearly, neuron importance distribution by MPMG-sum is more balanced than by MNMG-sum.



(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

Figure 10: Balanced neuron importance distribution by MPMG-sum on ShapeNet and BraTS'18. Neuron sparsity is 78.24% on ShapeNet and 78.17% on BraTS'18. Blue: neuron values; red: mean values.



(a) ShapeNet (b) BraTS'18

Figure 11: Layer-wise neuron distribution of 3D-UNets.

References

[1] J. Carreira and A. Zisserman. Quo vadis, action recognition? a new model and the Kinetics dataset. *CVPR*, 2017.

[2] O. Cicek, A. Abdulkadir, S. Lienkamp, T. Brox, and O. Ronneberger. 3D U-Net: Learning dense volumetric segmentation from sparse annotation. *MICCAI*, 2016.

[3] X. Glorot and Y. Bengio. Understanding the difficulty of training deep feedforward neural networks. *International Confer-*

Table 7: Impact of parameter initialization on neuron pruning. "ort": orthogonal initialization; "xn": Glorot initialization; "f": FLOPs. "Sparsity" is the least max neuron sparsity among all manners to ensure the network feasibility. RANP-f with Glorot initialization achieves the least FLOPs and memory consumption.

Dataset(Model)	Manner	Sparsity(%)	Param(MB)	GFLOPs	Mem(MB)
ShapeNet (3D-UNet)	Full[2]	0	62.26	237.85	997.00
	Vanilla-ort	70.53	4.40	72.65	630.00
	Vanilla-xn	70.53	4.56	73.22	618.35
	RANP-f-ort	70.53	5.40	21.73	366.29
	RANP-f-xn	70.53	5.52	**15.06**	**328.66**
BraTS'18 (3D-UNet)	Full[2]	0	15.57	478.13	3628.00
	Vanilla-ort[5]	72.20	0.95	159.91	2240.33
	Vanilla-xn	72.20	0.92	130.28	2109.19
	RANP-f-ort	72.20	1.24	33.28	967.56
	RANP-f-xn	72.20	1.29	**23.31**	**850.56**
UCF101 (MobileNetV2)	Full[6]	0	9.47	0.58	157.47
	Vanilla-ort[5]	30.21	6.80	0.56	155.71
	Vanilla-xn	30.21	6.77	0.55	155.48
	RANP-f-ort	30.21	5.12	0.32	105.88
	RANP-f-xn	30.21	5.19	**0.28**	**94.50**
UCF101 (I3D)	Full[1]	0	47.27	27.88	201.28
	Vanilla-ort[5]	24.24	30.56	25.83	192.70
	Vanilla-xn	24.24	30.64	25.85	192.88
	RANP-f-ort	24.24	27.39	15.94	144.10
	RANP-f-xn	24.24	27.38	**14.63**	**133.80**

ence on Artificial Intelligence and Statistics (AISTATS), 2010.

[4] P. Kao, T. Ngo, A. Zhang, J. Chen, and B. Manjunath. Brain tumor segmentation and tractographic feature extraction from structural MR images for overall survival prediction. *Workshop on MICCAI*, 2018.

[5] N. Lee, T. Ajanthan, S. Gould, and P. Torr. A signal propagation perspective for pruning neural networks at initialization. *ICLR*, 2020.

[6] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L. Chen. Mobilenetv2: Inverted residuals and linear bottlenecks. *CVPR*, 2018.

[7] L. Yi, L. Shao, and M. Savva. Large-scale 3D shape reconstruction and segmentation from shapenet core55. *arXiv preprint arXiv:1710.06104*, 2017.