Learning N:M Fine-grained Structured Sparse Neural Networks From Scratch

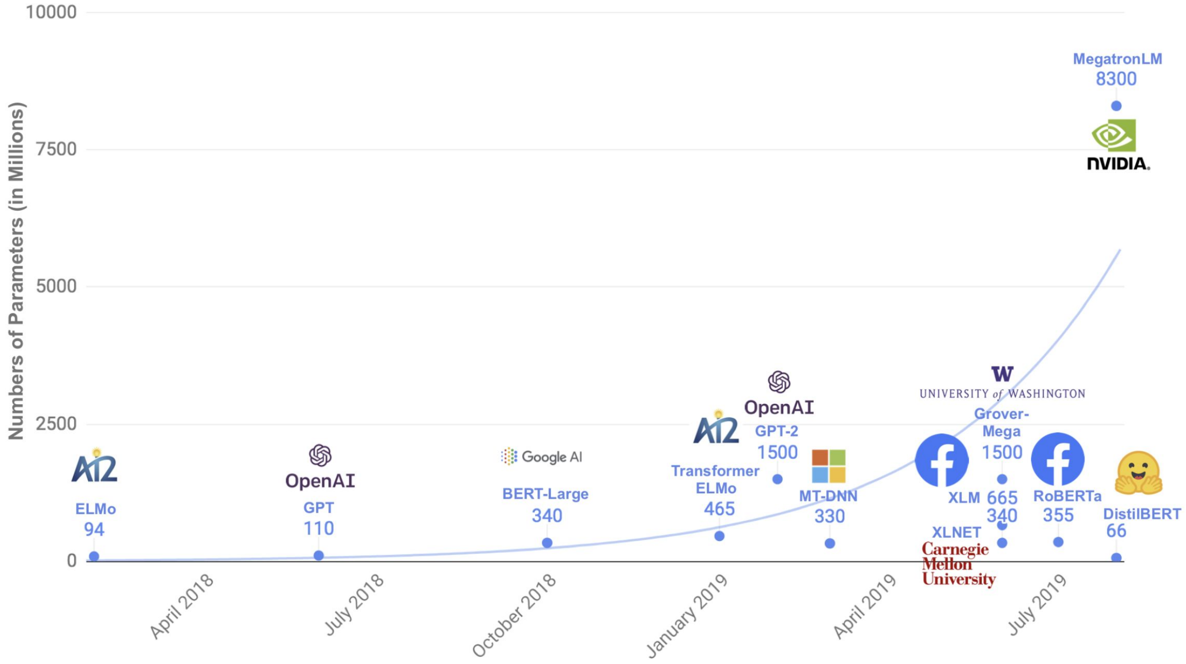
Aojun Zhou*, Yukun Ma*, Junnan Zhu, Jianbo Liu, Zhijie Zhang, Kun Yuan, Wenxiu Sun, Hongsheng Li

SenseTime Research, CUHK-Sensetime Joint Lab, CUHK, Northwestern University, NLPR of CASIA

* equal contributions

Background

Neural Networks have too many parameters!



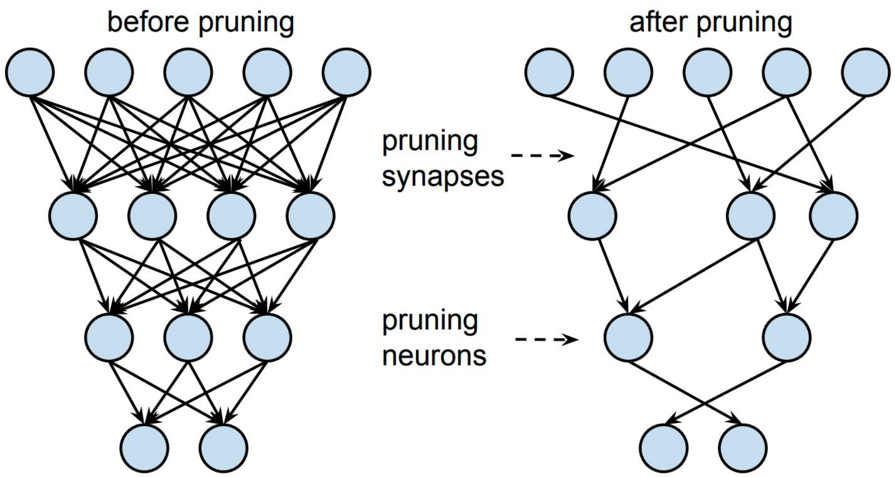
Parameter counts of several recently released pretrained language models

Sanh, Victor, et al. "DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter." *arXiv e-prints* (2019): arXiv-1910.

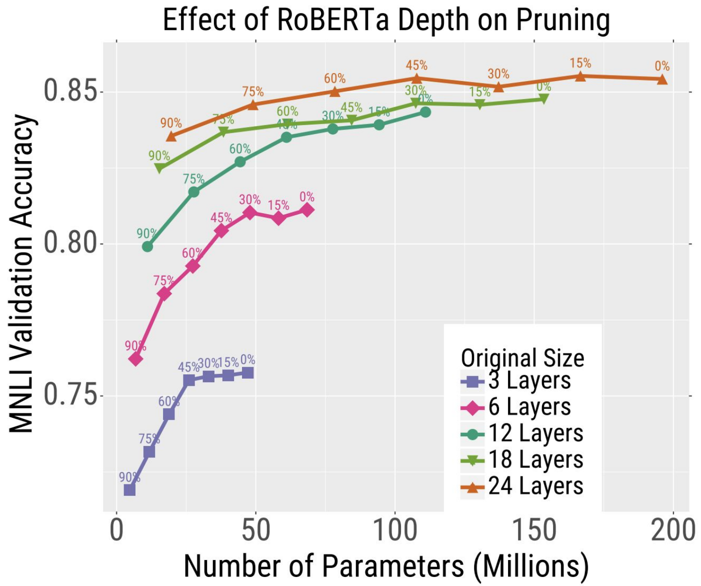
Sparse Networks

Better efficiency with comparable performance

Better performance with the same # of parameters

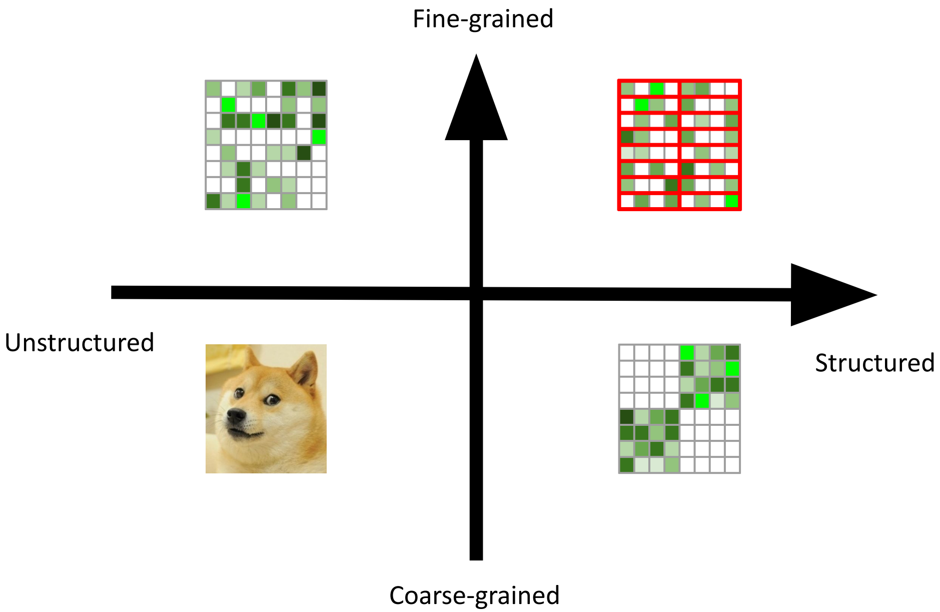


Synapses and neurons before and after pruning. Han, Song, et al. "Learning both Weights and Connections for Efficient Neural Network." *NIPS*. 2015.



RoBERTa's performance vs. number of parameters. Li, Zhuohan, et al. "Train Big, Then Compress: Rethinking Model Size for Efficient Training and Inference of Transformers." *International Conference on Machine Learning*. PMLR, 2020.

Sparsity Types

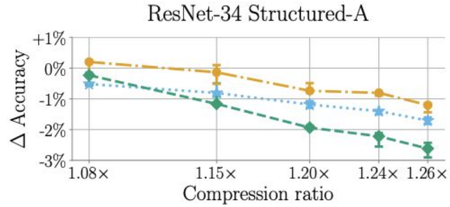


3 types of sparse networks with compression ratio 2 (half the parameters are zero)

Structured and Coarse-grained

low accuracy (the picture below)

high speedup



The accuracy drops significantly as the compression ratio increases. Renda, Alex, Jonathan, Frankle, and Michael, Carbin. "Comparing Rewinding and Fine-tuning in Neural Network Pruning." . In International Conference on Learning Representations.2020.

Unstructured and Fine-grained

high accuracy

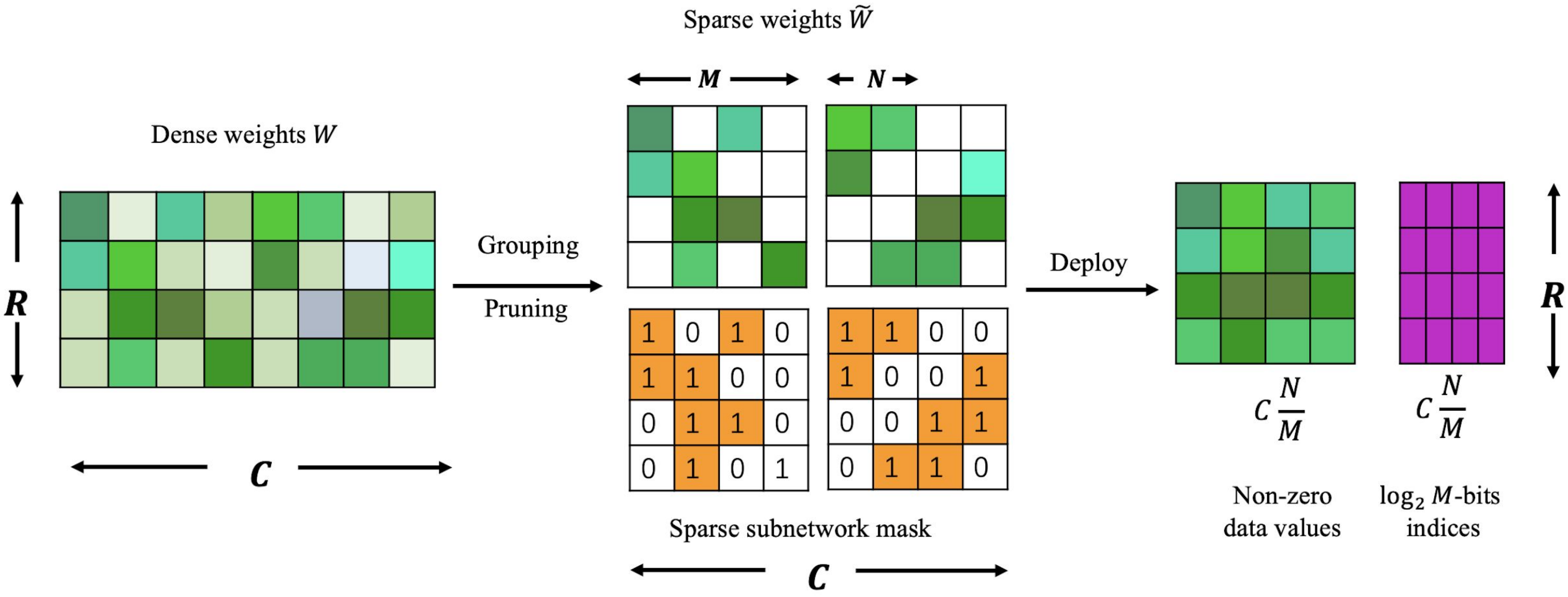
low speedup (Ma, Xiaolong, et al, "Non-Structured DNN Weight Pruning – Is It Beneficial in Any Platform?,"" IEEE Transactions on Neural Networks and Learning Systems (TNNLS), 2020.)

Structured and Fine-grained

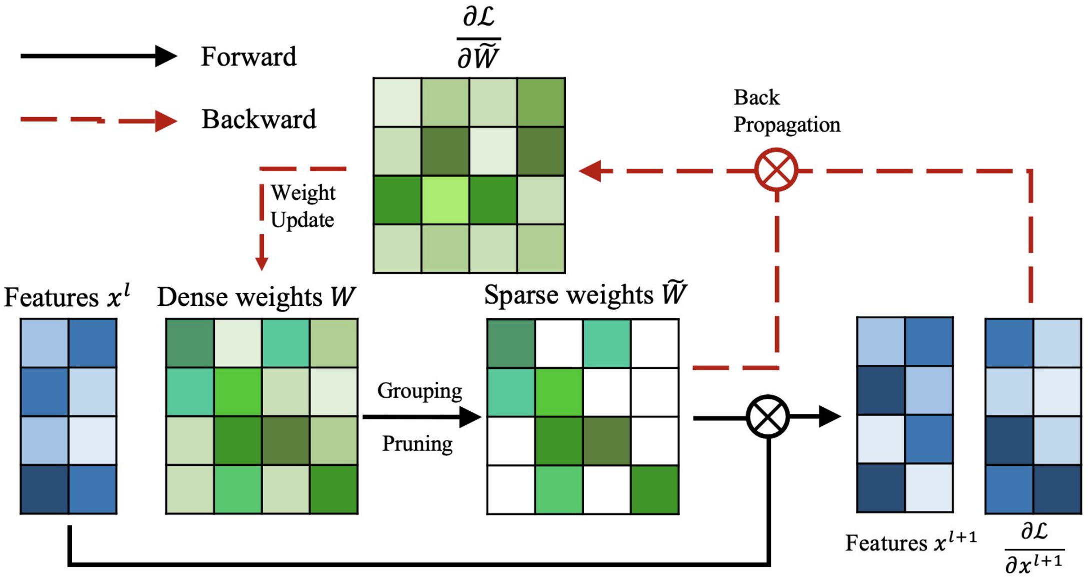
high accuracy and high speedup!

N:M Fine-grained Structured Sparse Network

Supported by NVIDIA Ampere GPU



Training an *N:M* Sparse Network From Scratch



Straight-Through Estimator

forward as sparse network, backward as dense network

Simple idea, but with poor performance

$\underset{\text{STE}}{76.2}$ vs $\underset{\text{dense}}{77.3}$

For pruned weights:

zero in forward, non-zero in backward

more roughly approximated gradients

For unpruned weights:

non-zero both in forward and backward

more accurate gradients

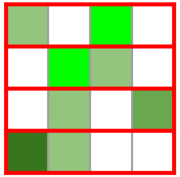
Q1: How about lowering the impact of the inaccurate gradients when updating the network?

Proposed Methods

Sparse Architecture Divergence (SAD)

0-th iteration

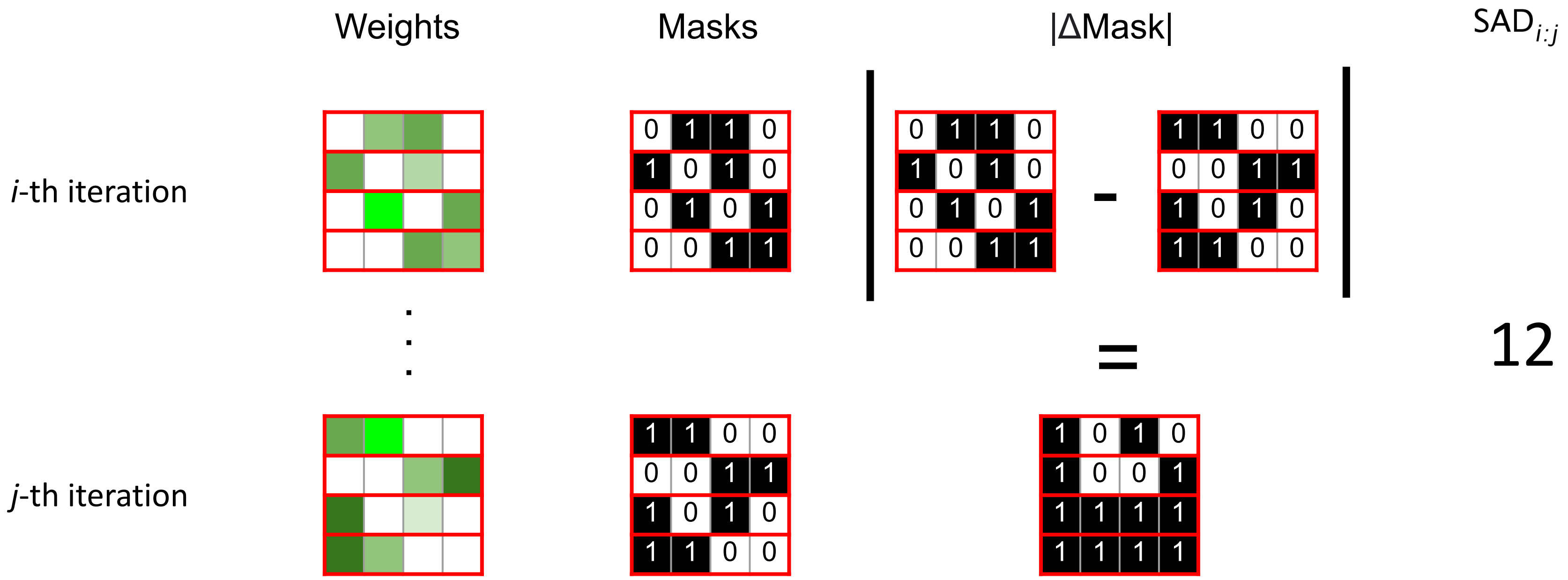
Weights



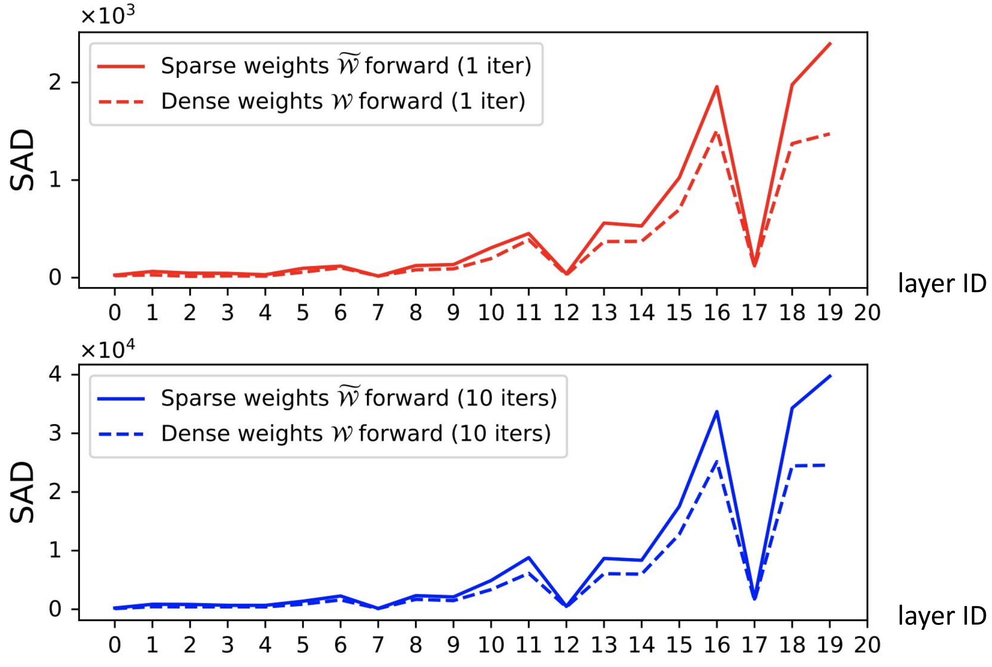
Masks

1	0	1	0
0	1	1	0
0	1	0	1
1	1	0	0

Sparse Architecture Divergence (SAD)



Further Investigations into STE



$SAD_{0:i}$ for densely trained and STE-trained networks

Q2: Will preventing high SAD help alleviate the performance drop?

Sparse-Refined Straight-Through Estimator (SR-STE)

SR-STE updating rule:

$$W_{t+1} = W_t - \gamma_t \left(g\left(\widetilde{W}_t\right) + \boxed{\lambda_W \varepsilon_t \odot W_t} \right)$$

sparse-refined term

STE updating rule:

$$W_{t+1} = W_t - \gamma_t g\left(\widetilde{W}_t\right)$$

W_t: weights after t – th iteration

$\widetilde{W}_t$: weights of the pruned version

λ_w: weight of the sparse – refined term

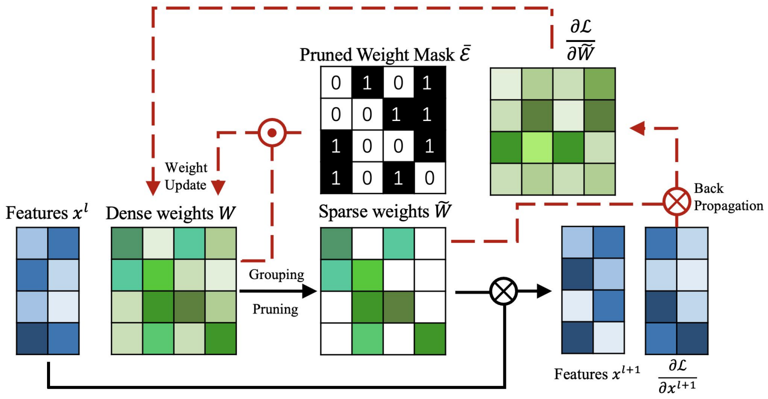
γ_t: step size

ε_t: 0 – 1 pruning mask

g: gradients

1. To reduce SGD step size for pruned parameters since their gradients are more roughly approximated

2. To prevent ineffective sparse architecture change



Experimental Results

Model	Method	Sparse Pattern	Top-1 Acc(%)	Params(M)	Flops(G)
ResNet50	-	Dense	77.3	25.6	4.09
ResNet50	SR-STE	2:4	77.0	12.8	2.05
ResNet50	**SR-STE**	**4:8**	**77.4**	**12.8**	**2.05**
ResNet50	SR-STE	1:4	75.9	6.4	1.02
ResNet50	**SR-STE**	**2:8**	**76.4**	**6.4**	**1.02**
ResNet50 x1.25	**SR-STE**	**2:8**	**77.5**	9.9	1.6

Table 1. ILSVRC validation accuracy with different sparse patterns

Experimental Results (Cont'd)

Model	Method	Sparse Pattern	Top-1 Acc	Epochs
ResNet18	ASP (Nvidia, 2020)	2:4	70.7	200
ResNet18	STE	2:4	69.9	120
ResNet18	**SR-STE**	2:4	**71.2**	120
ResNet50	ASP(Nvidia, 2020)	2:4	76.8	200
ResNet50	STE	2:4	76.4	120
ResNet50	**SR-STE**	**2:4**	**77.0**	**120**

Table 2. ILSVRC validation accuracy of 2:4 sparse models trainde with different methods.

Experimental Results (Cont’d)

Method	Top-1 Acc(%)	Sparsity(%)	Params(M)	Flops(G)	Structured	Uniform
ResNet50	77.3	0.0	25.6	4.09	-	-
DSR*	71.6	80	5.12	0.82	✗	✗
RigL	74.6	80	5.12	0.92	✗	✓
GMP	75.6	80	5.12	0.82	✗	✓
STR	76.1	81	5.22	0.71	✗	✗
STE	76.2	80	5.12	0.82	✗	✓
SR-STE	**77.0**	**80**	**5.12**	**0.82**	✗	✓
SR-STE	**76.4**	**75(2:8)**	**6.40**	**1.02**	✓	✓
RigL	67.5	95	1.28	0.32	✗	✓
GMP	70.6	95	1.28	0.20	✗	✓
STR	70.2	95	1.24	0.16	✗	✗
STE	68.4	95	1.28	0.20	✗	✓
SR-STE	**72.4**	**95**	**1.28**	**0.20**	✗	✓
SR-STE	**72.2**	**94(1:16)**	**1.60**	**0.25**	✓	✓

Table 3. ILSVRC validation accuracy of state-of-the-art sparse model training methods.

Thank you!

Please follow our work @

code: https://github.com/NM-sparsity/NM-sparsity

paper: https://openreview.net/pdf?id=K9bw7vqp_s