[Supplementary Materials] Exploring Lottery Ticket Hypothesis in Spiking Neural Networks

Youngeun Kim, Yuhang Li, Hyoungseob Park, Yeshwanth Venkatesha, Ruokai Yin, and Priyadarshini Panda

Department of Electrical Engineering
Yale University
New Haven, CT, USA
{youngeun.kim, yuhang.li, hyoungseob.park, yeshwanth.venkatesha, ruokai.yin, priya.panda}@yale.edu

A. Backpropagation Training with Surrogate Gradients

Following the previous surrogate gradients works [8,7,9,10,14,12,6,13], we train the weight parameters with surrogate gradients function in order to circumvent non-differentiable problem of LIF neurons. Specifically, the output spikes o_i^t are generated if the membrane potential u_i^t exceeds a firing threshold. Here, we use $arctan(\cdot)$ function following the previous work [3]:

$$\frac{\partial o_i^t}{\partial u_i^t} = \frac{1}{\pi} arctan(\pi x) + \frac{1}{2}. \tag{1}$$

We follow the loss function L used in [1] which accumulates cross-entropy loss across all timesteps. The gradients for each weight are computed in both spatial and time axis [14,9]. According to the chain rule, we can compute the gradients of the weight parameters W_l as:

$$\frac{\partial L}{\partial W_l} = \sum_t \left(\frac{\partial L}{\partial O_l^t}\frac{\partial O_l^t}{\partial U_l^t} + \frac{\partial L}{\partial U_l^{t+1}}\frac{\partial U_l^{t+1}}{\partial U_l^t}\right)\frac{\partial U_l^t}{\partial W_l}. \tag{2}$$

Here, O_l^t and U_l^t are the matrix form of output spikes and membrane potential at time-step t for layer l, respectively.

B. Late Rewinding for SNNs

Frankle *et al.* [4] present *Late Rewinding*, which rewinds the network to the weights at epoch i rather than initialization. This enables IMP to discover the winning ticket with less performance drop in high sparsity regime by providing a more stable starting point. As shown in Fig. 1, we found that *Late Rewinding* shows better performance than the original IMP in deep SNNs. Throughout our main paper, we apply *Late Rewinding* to IMP for experiments where we rewind the network to epoch 20.

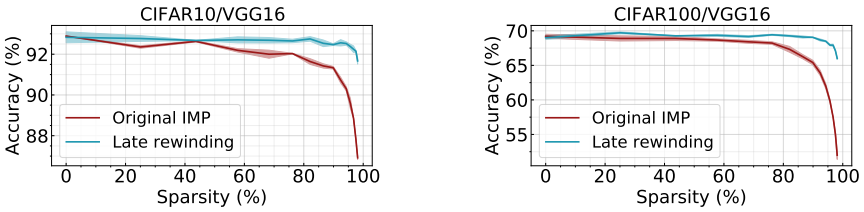

Fig. 1. Original IMP vs. Late rewinding LTH.

C. Variation of KL Divergence at the Early Training Stage

We found that the normalized KL divergence metric (for each T_{early}) shows *high variation in the first epoch*, as shown in Fig. 2. After epoch 2, the standard deviation of the normalized KL divergence metric goes smaller, and the mean normalized KL divergence maintains a similar value across training. Therefore, we train SNN for 2 epochs in order to stabilize the KL divergence metric while minimizing training cost. Note, we can also select more higher number of epochs, however, they will provide the same T_{early}.

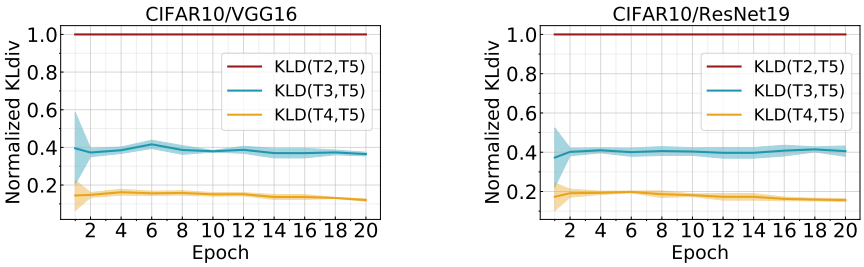

Fig. 2. Normalized KL divergence between the predicted class probabilities from different timesteps. We show mean/standard deviation from 5 random runs.

D. Time Cost for VGG and ResNet Architectures

To provide the reference on search time measured in the main paper, we show time cost for one training iteration (feedforward + backward) on VGG16 and ResNet19 in Table 1. We use SpikingJelly which provides optimized LIF neuron implementation on NVIDIA RTX 2080ti GPU and Intel(R) Xeon(R) Gold 6240 CPU @ 2.60GHz processor. The execution time is proportional to the size of the backward computational graph [2] which is affected by timesteps. The results show the execution time exponentially increases if the computational graph

Table 1. Time cost for one iteration (feedforward + backward) with respect to the number of timesteps. We use batch size 128 for the experiments.

Method (unit:seconds)	Number of Timesteps				
	T=1	T=2	T=3	T=4	T=5
VGG16 [11]	26.45	29.15	32.44	37.26	44.49
ResNet19 [5]	27.57	31.14	39.76	49.85	61.50

Table 2. Ablation studies on a metric for T_{early} selection. We use JSD and KLD for experiments. Note, the value at T=2 is fixed to 1 as a reference. We report mean and stdandard deviation from 5 runs.

Method	Jensen–Shannon Divergence			Kullback-Leibler Divergence		
	T=2	T=3	T=4	T=2	T=3	T=4
VGG16 [11]	1± 0.00	0.281± 0.028	0.167± 0.027	1± 0.00	0.274± 0.052	0.158± 0.063
ResNet19 [5]	1± 0.00	0.375± 0.010	0.233± 0.021	1± 0.00	0.367±0.031	0.243± 0.038

goes larger. Also, we observe that ResNet19 requires more execution time than VGG16, and the difference increase with higher timesteps.

E. Experiments on Early Time (ET) tickets

In Table 3, we report the change in accuracy and search speed gain from applying ET to IMP and EB, on SVHN, Fashion-MNIST, CIFAR10, and CIFAR100 datasets. The results support our hypothesis that important weight connectivity of the SNN can be discovered from shorter timesteps.

F. Applying Various Distance Metrics

To discover T_{early}, we use Kullback-Leibler Divergence (KLD) between the class probabilites across different timesteps (refer Algorithm 1 in the main paper). In this section, we measure the normalized Jensen–Shannon Divergence (JSD), a symmetric version of KLD, in Table 2. We can observe that both KLD and JSD show a similar distance for VGG16 and ResNet19 architecture. This implies that our T_{early} algorithm is robust to the distance metric.

G. Experiments on deep/shallow networks

To further validate the existence of LTH in SNNs, we ran the experiments with ResNet34 on TinyImageNet. In Fig.3 **Left**, we found that *Winning Ticket* exists even in deep networks on larger dataset with various pruning methods including *IMP, EB, IMP+ET*, and *EB+ET*. We also report a random pruning as a baseline, which implies *Winning Ticket* is difficult to be discovered with a naïve approach. We also evaluate AlexNet on CIFAR10 in Fig.3 **Right**. The results corroborate our statement in which LTH can be applied to SNNs.

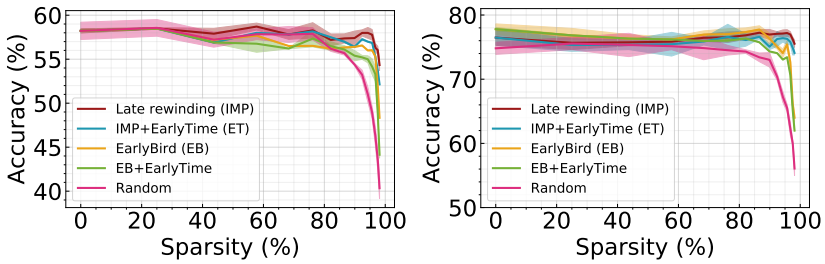

Fig. 3. Accuracy comparison of pruning methods on two settings. **Left:** ResNet34 on TinyImageNet. **Right:** AlexNet on CIFAR10. We show mean/standard deviation from 5 random runs.

References

1. Deng, S., Li, Y., Zhang, S., Gu, S.: Temporal efficient training of spiking neural network via gradient re-weighting. In: International Conference on Learning Representations (2022), `https://openreview.net/forum?id=_XNtisL32jv`
2. Fang, W., Chen, Y., Ding, J., Chen, D., Yu, Z., Zhou, H., Tian, Y., other contributors: Spikingjelly. `https://github.com/fangwei123456/spikingjelly` (2020)
3. Fang, W., Yu, Z., Chen, Y., Masquelier, T., Huang, T., Tian, Y.: Incorporating learnable membrane time constant to enhance learning of spiking neural networks. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 2661–2671 (2021)
4. Frankle, J., Dziugaite, G.K., Roy, D.M., Carbin, M.: Stabilizing the lottery ticket hypothesis. arXiv preprint arXiv:1903.01611 (2019)
5. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: CVPR. pp. 770–778 (2016)
6. Kim, Y., Li, Y., Park, H., Venkatesha, Y., Panda, P.: Neural architecture search for spiking neural networks. arXiv preprint arXiv:2201.10355 (2022)
7. Lee, C., Sarwar, S.S., Panda, P., Srinivasan, G., Roy, K.: Enabling spike-based backpropagation for training deep neural network architectures. Frontiers in Neuroscience **14** (2020)
8. Lee, J.H., Delbruck, T., Pfeiffer, M.: Training deep spiking neural networks using backpropagation. Frontiers in neuroscience **10**, 508 (2016)
9. Neftci, E.O., Mostafa, H., Zenke, F.: Surrogate gradient learning in spiking neural networks. IEEE Signal Processing Magazine **36**, 61–63 (2019)
10. Shrestha, S.B., Orchard, G.: Slayer: Spike layer error reassignment in time. arXiv preprint arXiv:1810.08646 (2018)
11. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. ICLR (2015)
12. Wu, H., Zhang, Y., Weng, W., Zhang, Y., Xiong, Z., Zha, Z.J., Sun, X., Wu, F.: Training spiking neural networks with accumulated spiking flow. ijo **1**(1) (2021)
13. Wu, J., Xu, C., Zhou, D., Li, H., Tan, K.C.: Progressive tandem learning for pattern recognition with deep spiking neural networks. arXiv preprint arXiv:2007.01204 (2020)

Table 3. Effect of the proposed Early-Time ticket. We compare the accuracy and search time of Iterative Magnitude Pruning (IMP), Early-Bird (EB) ticket, Early-Time (ET) ticket on four sparsity levels. We show search speed gain and accuracy change from applying ET. Note, ResNet19 requires 1.06 $\sim$ 1.38$\times$ forward+backward time compared to VGG16 because of a large computational graph from multiple timesteps.

Setting	Method	Accuracy				Winning Ticket Search Time (hours)			
		p=68.30%	p=89.91%	p=95.69%	p=98.13%	p=68.30%	p=89.91%	p=95.69%	p=98.13%
SVHN VGG16	IMP	95.53	95.56	95.41	95.18	7.42	14.99	20.56	26.09
	IMP + ET	95.25	95.15	94.36	94.28	5.48	11.03	15.09	19.14
	Δ Acc. / Speed Gain	-0.28	-0.41	-1.05	-0.90	$\times$1.35	$\times$1.35	$\times$1.36	$\times$1.36
	EB	95.15	95.29	94.81	94.09	0.76	0.23	0.11	0.08
	EB + ET	94.85	94.86	94.26	94.08	0.56	0.18	0.09	0.07
	Δ Acc. / Speed Gain	-0.30	-0.43	-0.55	-0.01	$\times$1.34	$\times$1.29	$\times$1.22	$\times$1.18
SVHN Res19	IMP	96.08	96.05	95.90	95.51	10.25	20.59	28.74	35.87
	IMP + ET	95.62	95.70	95.46	94.84	6.62	13.25	18.22	22.68
	Δ Acc. / Speed Gain	-0.46	-0.35	-0.44	-0.67	$\times$1.54	$\times$1.55	$\times$1.57	$\times$1.58
	EB	95.75	95.38	95.39	94.56	1.44	0.80	0.35	0.12
	EB + ET	95.53	95.68	95.20	94.34	0.56	0.18	0.09	0.07
	Δ Acc. / Speed Gain	-0.22	+0.30	-0.19	-0.22	$\times$1.51	$\times$1.49	$\times$1.43	$\times$1.25
FMNIST VGG16	IMP	94.81	94.74	94.63	94.52	7.52	14.77	20.40	26.10
	IMP + ET	94.87	94.97	94.57	93.61	5.59	11.00	15.05	19.13
	Δ Acc. / Speed Gain	+0.06	+0.23	-0.06	-0.91	$\times$1.34	$\times$1.34	$\times$1.35	$\times$1.36
	EB	94.04	93.92	93.69	93.16	0.81	0.32	0.08	0.06
	EB + ET	93.68	93.73	93.21	92.25	0.60	0.24	0.07	0.05
	Δ Acc. / Speed Gain	-0.36	-0.19	-0.48	-0.91	$\times$1.34	$\times$1.31	$\times$1.18	$\times$1.12
FMNIST Res19	IMP	95.00	94.96	94.96	94.68	10.35	20.63	28.92	35.90
	IMP + ET	94.67	94.87	94.5	94.63	6.63	13.10	18.34	22.83
	Δ Acc. / Speed Gain	-0.33	-0.09	-0.46	-0.05	$\times$1.56	$\times$1.57	$\times$1.57	$\times$1.57
	EB	94.21	94.14	93.98	93.60	1.53	0.76	0.20	0.10
	EB + ET	93.94	93.89	93.48	92.51	1.01	0.51	0.15	0.08
	Δ Acc. / Speed Gain	-0.27	-0.25	-0.50	-1.09	$\times$1.51	$\times$1.49	$\times$1.38	$\times$1.16
CIFAR10 VGG16	IMP	92.66	92.54	92.38	91.81	14.97	29.86	40.84	51.99
	IMP + ET	92.49	92.09	91.54	91.10	11.19	22.00	30.11	38.26
	Δ Acc. / Speed Gain	-0.17	-0.45	-0.84	-0.71	$\times$1.34	$\times$1.35	$\times$1.35	$\times$1.35
	EB	91.74	91.05	89.55	84.64	1.96	0.74	0.11	0.09
	EB + ET	91.27	90.66	88.95	84.86	1.44	0.55	0.07	0.06
	Δ Acc. / Speed Gain	-0.47	-0.39	-0.60	+0.22	$\times$1.36	$\times$1.34	$\times$1.18	$\times$1.12
CIFAR10 Res19	IMP	93.47	93.49	93.22	92.43	21.01	42.20	58.91	73.54
	IMP + ET	93.10	92.72	92.68	91.36	13.35	26.62	37.27	46.40
	Δ Acc. / Speed Gain	-0.37	-0.77	-0.54	-1.07	$\times$1.57	$\times$1.59	$\times$1.58	$\times$1.58
	EB	91.00	90.84	89.90	85.22	2.49	0.87	0.24	0.08
	EB + ET	90.83	91.21	89.65	85.45	1.63	0.58	1.70	0.07
	Δ Acc. / Speed Gain	-0.17	+0.37	-0.50	-1.09	$\times$1.52	$\times$1.49	$\times$1.38	$\times$1.16
CIFAR100 VGG16	IMP	69.08	68.90	68.00	66.02	15.02	29.99	41.03	52.05
	IMP + ET	68.27	67.99	66.51	64.41	11.24	22.42	30.53	38.32
	Δ Acc. / Speed Gain	-0.81	-0.91	-1.49	-1.61	$\times$1.33	$\times$1.34	$\times$1.34	$\times$1.36
	EB	67.35	65.82	61.90	52.11	2.27	0.99	0.32	0.06
	EB + ET	67.26	64.18	61.81	52.77	1.66	0.73	0.24	0.05
	Δ Acc. / Speed Gain	-0.09	-1.64	-0.09	+0.66	$\times$1.36	$\times$1.35	$\times$1.31	$\times$1.12
CIFAR100 ResNet19	IMP	71.64	71.38	70.45	67.35	21.21	42.29	59.17	73.52
	IMP + ET	71.06	70.45	69.23	65.49	13.56	27.05	37.88	46.65
	Δ Acc. / Speed Gain	-0.58	-0.93	-1.22	-1.86	$\times$1.56	$\times$1.56	$\times$1.56	$\times$1.57
	EB	69.41	65.87	62.18	52.92	3.08	1.71	0.43	0.09
	EB + ET	68.98	65.76	62.20	51.50	2.00	1.12	0.29	0.07
	Δ Acc. / Speed Gain	-0.43	-0.12	+0.02	-1.42	$\times$1.53	$\times$1.52	$\times$1.45	$\times$1.16

14. Wu, Y., Deng, L., Li, G., Zhu, J., Shi, L.: Spatio-temporal backpropagation for training high-performance spiking neural networks. Frontiers in neuroscience **12**, 331 (2018)