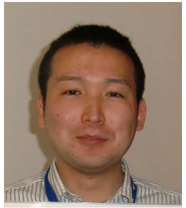
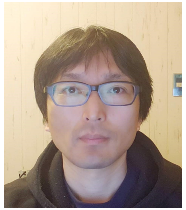
Pruning Randomly Initialized Neural Networks with Iterative Randomization


Daiki Chijiwa*


Shin'ya Yamaguchi


Yasutoshi Ida


Kenji Umakoshi


Tomohiro Inoue

Training Methods for Neural Networks

$f(x;\theta)$: a neural network parametrized by $\theta \in \mathbb{R}^N$.

$\mathcal{L}$: loss function, $\mathcal{D}_{\mathrm{data}}$: data distribution, $\mathcal{D}_{\mathrm{param}}$: parameter distribution.

Training Methods for Neural Networks

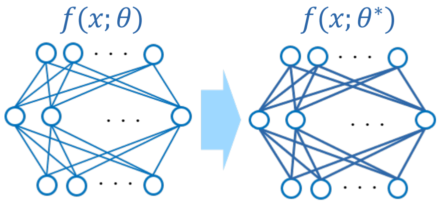
$f(x;\theta)$: a neural network parametrized by $\theta \in \mathbb{R}^N$.

$\mathcal{L}$: loss function, $\mathcal{D}_{\text{data}}$: data distribution, $\mathcal{D}_{\text{param}}$: parameter distribution.

<u>Weight-Optimization</u> (Standard Approach in Deep Learning)

Optimize the following problem with SGD:

$$\min_{\theta \in \mathbb{R}^N} \mathbb{E}_{(x,y)\sim\mathcal{D}_{\text{data}}}[\mathcal{L}(f(x;\theta), y)]$$



Training Methods for Neural Networks

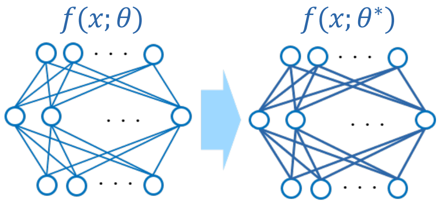
$f(x;\theta)$: a neural network parametrized by $\theta \in \mathbb{R}^N$.

$\mathcal{L}$: loss function, $\mathcal{D}_{\mathrm{data}}$: data distribution, $\mathcal{D}_{\mathrm{param}}$: parameter distribution.

<u>Weight-Optimization</u> (Standard Approach in Deep Learning)

Optimize the following problem with SGD:

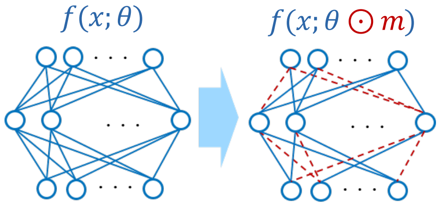
$$\min_{\theta \in \mathbb{R}^N} \mathbb{E}_{(x,y)\sim\mathcal{D}_{\mathrm{data}}}[\mathcal{L}(f(x;\theta), y)]$$



<u>Weight-Pruning Optimization</u> (Ramanujan et al., CVPR'20)

Sample and fix $\theta \sim \mathcal{D}_{\mathrm{param}}$. Then optimize:

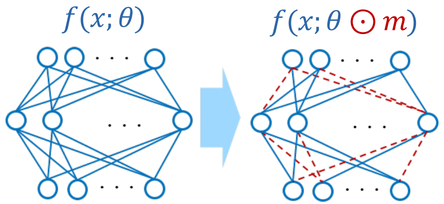
$$\min_{m \in \{0,1\}^N} \mathbb{E}_{(x,y)\sim\mathcal{D}_{\mathrm{data}}}[\mathcal{L}(f(x;\theta \odot m), y)]$$



Weight-Pruning Optimization (Ramanujan et al., CVPR'20)

Sample and fix $\theta \sim \mathcal{D}_{\mathrm{param}}$. Then optimize:

$$\min_{m \in \{0,1\}^N} \mathbb{E}_{(x,y)\sim\mathcal{D}_{\mathrm{data}}}[\mathcal{L}(f(x;\theta \odot m), y)]$$



Features:
- Discrete search space, sparse network.
- We can control the range of values in θ.

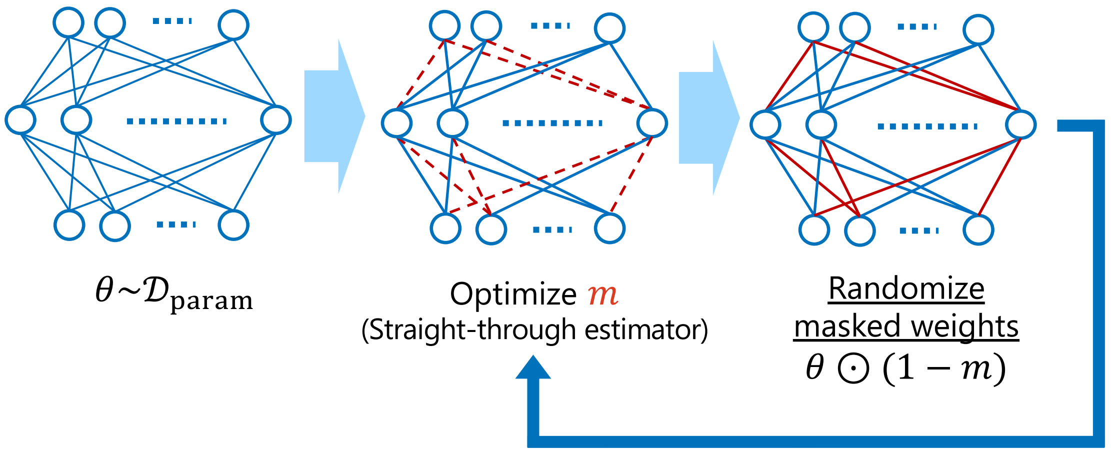
  E.g. If we take a uniform distribution over $\{1, -1\}$ for $\mathcal{D}_{\mathrm{param}}$,

  θ is a vector of binary parameters during/after training.

Drawback:
- (Pensia et al., NeurIPS'20) The weight-pruning requires logarimithmic over-parametrization for $f(x;\theta)$ to achieve the same approximation capacity as the weight-optimization.

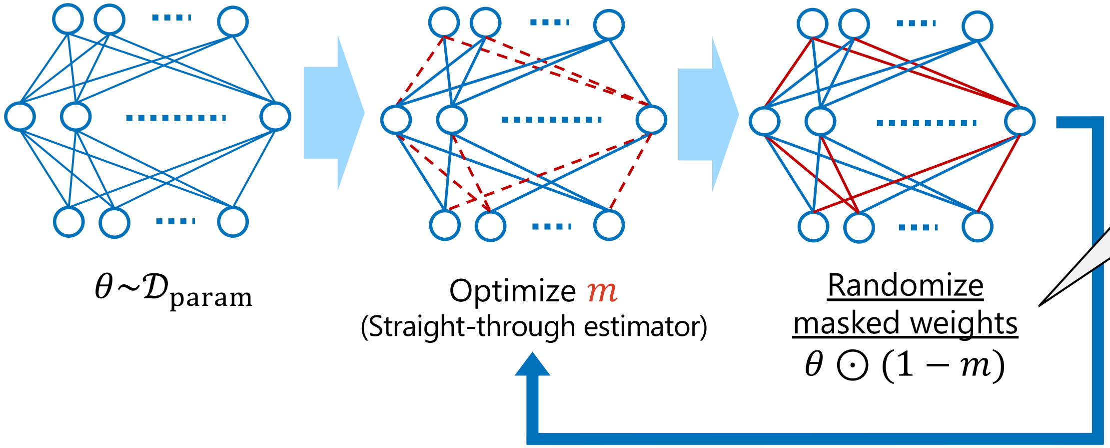
Our Approach: Iterative Randomization

Randomizing pruned weights during the weight-pruning optimization
leads to <u>increase the number of network parameters virtually</u>.



Our Approach: Iterative Randomization

Randomizing pruned weights during the weight-pruning optimization leads to <u>increase the number of network parameters virtually</u>.



Naïve Randomization:

$$\theta \leftarrow \theta \odot m + \tilde{\theta} \odot (1 - m),$$

where $\tilde{\theta} \sim \mathcal{D}_{\mathrm{param}}$.

Also, we proposed a stabilized version of the randomizing operation in our paper. (*Partial Randomization*)

Results (Theoretical Justification)

$f(x)$: a target neural network, d_i: width size for i-th layer of $f(x)$.

$g(x)$: a neural network to be pruned, $\widetilde{d}_j$: width size for j-th layer of $g(x)$.

<u>Assume:</u> the weights of $g(x)$ are sampled from the uniform distribution over $[-1,1]$.

Theorem 4.1 (Main Theorem) *Fix $\epsilon, \delta > 0$, and we assume that $\|F_i\|_{\text{Frob}} \leq 1$. Let $R \in \mathbb{N}$, and assume that $g(x)$ satisfies the re-sampling assumption for R.*

If $\widetilde{d}_{2i-1} \geq 2d_{i-1}\lceil \frac{64l^2 d_{i-1}^2 d_i}{\epsilon^2 R^2} \log(\frac{2ld_{i-1}d_i}{\delta})\rceil$ holds for all $i = 1, \cdots, l$, then with probability at least $1-\delta$, there exist binary matrices $M = \{M_j\}_{1\leq j\leq 2l}$ such that

$$\|f(x) - g_M(x)\|_2 \leq \epsilon, \text{ for } \|x\|_\infty \leq 1. \tag{7}$$

In particular, if R is larger than $\frac{8ld_{i-1}}{\epsilon}\sqrt{d_i \log(\frac{2ld_{i-1}d_i}{\delta})}$, then $\widetilde{d}_{2i-1} = 2d_{i-1}$ is enough.

Results (Theoretical Justification)

$f(x)$: a target neural network, d_i: width size for i-th layer of $f(x)$.

$g(x)$: a neural network to be pruned, $\tilde{d}_j$: width size for j-th layer of $g(x)$.

<u>Assume:</u> the weights of $g(x)$ are sampled from the uniform distribution over $[-1,1]$.

R : # of randomizing iters.

$R = 1$: previous works

The over-parameterization factor

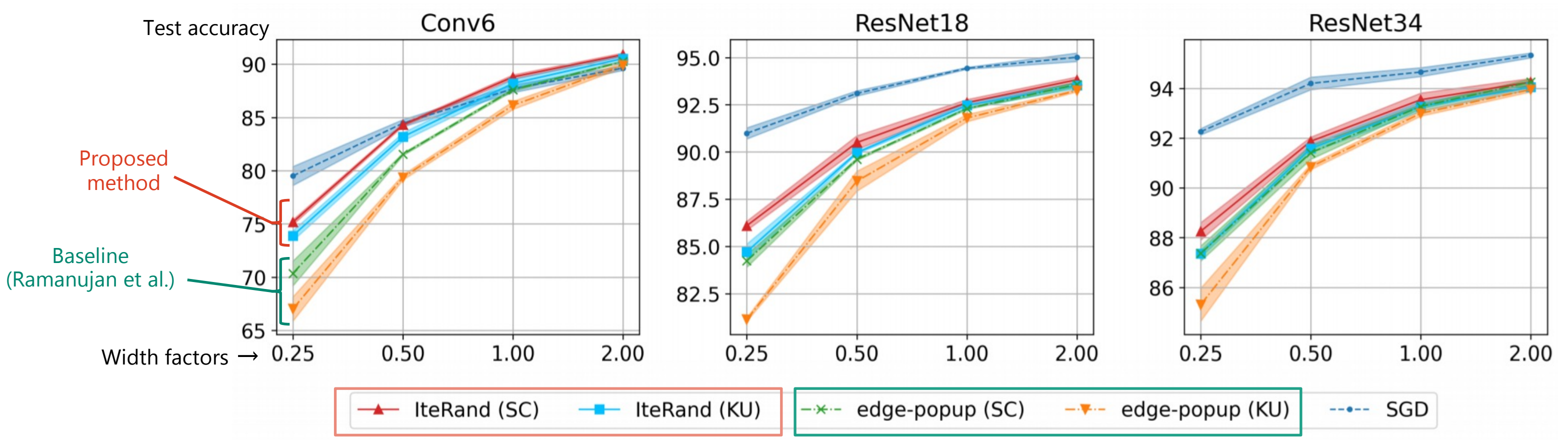
Theorem 4.1 (Main Theorem) *Fix $\epsilon, \delta > 0$, and we assume that $\|F_i\|_{\mathrm{Frob}} \leq 1$. Let $R \in \mathbb{N}$, and assume that $g(x)$ satisfies the re-sampling assumption for R.*
If $\boxed{\tilde{d}_{2i-1} \geq 2d_{i-1}\lceil \frac{64l^2 d_{i-1}^2 d_i}{\epsilon^2 R^2} \log(\frac{2ld_{i-1}d_i}{\delta})\rceil}$ holds for all $i = 1, \cdots, l$, then with probability at least $1-\delta$, there exist binary matrices $M = \{M_j\}_{1 \leq j \leq 2l}$ such that

$$\|f(x) - g_M(x)\|_2 \leq \epsilon, \text{ for } \|x\|_\infty \leq 1. \tag{7}$$

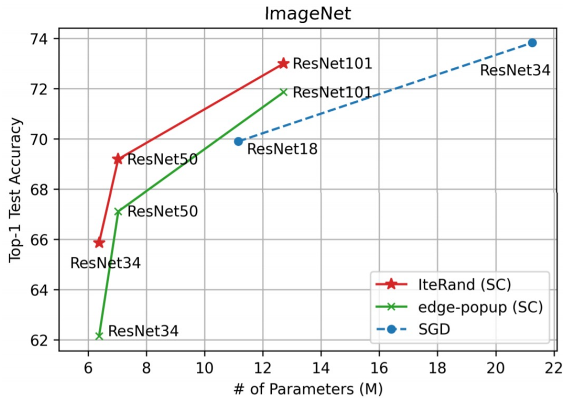
In particular, if R is larger than $\frac{8ld_{i-1}}{\epsilon}\sqrt{d_i \log(\frac{2ld_{i-1}d_i}{\delta})}$, then $\tilde{d}_{2i-1} = 2d_{i-1}$ is enough.

$O(\log(d))$ (Pensia et al., $R = 1$)
$\rightarrow O(1)$ (when $R \gg 1$)

Results (CIFAR-10 & ImageNet experiments)



CIFAR-10



ImageNet