Rigging The Lottery: Making All Tickets Winners (RigL)

Efficient and accurate training for sparse networks.

Utku Evci, Trevor Gale, Jacob Menick, Pablo Samuel Castro, Erich Elsen
AI Residency

Google AI Residency

Brain Montreal

DeepMind
Motivation

Sparse networks and their advantages
Sparse Networks

- **On device training/inference:** Reduces FLOPs and network size drastically without harming performance.

- **Reduced memory footprint:** We can fit wider/deeper networks in memory and get better performance from the same hardware.

- **Architecture search:**
  Causal relationships? Interpretability?
Sparse networks perform better for the same parameter count.


**Train Large, Then Compress: Rethinking Model Size for Efficient Training and Inference of Transformers**, Zhuohan Li, Eric Wallace, Sheng Shen, Kevin Lin, Kurt Keutzer, Dan Klein, Joseph E. Gonzalez, 2020
Accelerating sparsity

Is difficult, but possible

- **Block Sparse Kernels**: Efficient sparse operations.
- **Efficient WaveRNN**: Text-to-Speech
- **Optimizing Speech Recognition for the Edge**: Speech Recognition
- **Fast Sparse Convnets**: Fast mobile inference for vision models with sparsity (1.3 - 2.4x faster).

....and more to come.
How do we find sparse networks?

→ **Pruning** method requires dense training: (1) limits the biggest sparse network we can train (2) not efficient.

→ **Training from scratch** performs much worse.

→ **Lottery** initialization doesn't help.

* The Lottery Ticket Hypothesis: Finding Sparse, Trainable Neural Networks
  Jonathan Frankle, Michael Carbin, ICLR 2019

* The Difficulty of Training Sparse Neural Networks
  Utku Evci, Fabian Pedregosa, Aidan Gomez, Erich Elsen, 2019
Can we train sparse neural networks end-to-end?

(without ever needing the dense parameterization)

(as good as the dense-to-sparse methods)
YES. RigL!

Rigging The Lottery: Making all Tickets Winners and surpassing pruning performance.
The Algorithm

➔ Start from a random sparse network.
➔ Train the sparse network.
➔ Every N steps update connectivity:
  ◆ Drop least magnitude connections
  ◆ Grow new ones using gradient information.
The Algorithm

Algorithm 1 RigL

**Input:** Network $f_\Theta$, dataset $D$

- Sparsity Distribution: $S = \{s^1, \ldots, s^L\}$
- Update Schedule: $\Delta T, T_{end}, \alpha, f_{decay}$

- $\theta \leftarrow$ Randomly sparsify $\Theta$ using $S$

for each training step $t$ do

- Sample a batch $B_t \sim D$
- $L_t = \sum_{i \sim B_t} L(f_\theta(x_i), y_i)$

if $t \ (\text{mod} \Delta T) = 0$ and $t < T_{end}$ then

for each layer $l$ do

- $k = f_{\text{decay}}(t; \alpha, T_{end})(1 - s^l)N^l$
- $\mathbb{I}_{\text{drop}} = \text{ArgTopK}(-|\theta^l|, k)$
- $\mathbb{I}_{\text{grow}} = \text{ArgTopK}_{i \notin \theta^l \setminus \mathbb{I}_{\text{drop}}}(|\nabla_{\theta^l} L_t|, k)$
- $\theta \leftarrow$ Update connections $\theta$ using $\mathbb{I}_{\text{drop}}$ and $\mathbb{I}_{\text{grow}}$

end for

else

- $\theta = \theta - \alpha \nabla_{\theta} L_t$

end if

end for
Evaluation of Connections

Before training

Evaluation of first layer connections during MNIST MLP training.
ResNet-50
RigL matches pruning performance using significantly less resources.
Exceeding Pruning Performance

- RigL
  - outperforms pruning

- ERK sparsity distribution:
  - greater performance
  - more FLOPs.

RigL (ERK)

Static

Small-Dense

Prune

14x less FLOPs and parameters.
80% ERK on Resnet-50

Stage 1-2-3

Stage 4

Stage 5
Sparse MobileNets

- Difficult to prune.
- Much better results with RigL.

same parameter/flops: **4.3% absolute improvement** in Top-1 Accuracy.
Character Level Language Modelling on WikiText-103

→ Similar to WaveRNN.
→ RigL falls short of matching the pruning performance
Static training stucks in a suboptimal basin.

RigL helps escaping from it.

More results in our workshop paper
Summary of Contributions

- Training randomly initialized sparse networks without needing the dense parametrization is possible.
- Sparse networks found by RigL can exceed the performance of pruning.
- Non-uniform sparsity distribution like ERK brings better performance.
- RigL can help us with feature selection.

Limitations

- RigL requires more iterations to converge.
- Dynamic sparse training methods seem to have suboptimal performance training RNNs.
- Sparse kernels that can utilize sparsity during training are not widely available (yet).
Thank you!

- Sparse networks are promising.
- End-to-end sparse training is possible and it has potential to replace dense->sparse training.

@evcu  @tgale  @jmenick  @psc  @eriche

https://github.com/google-research/rigl