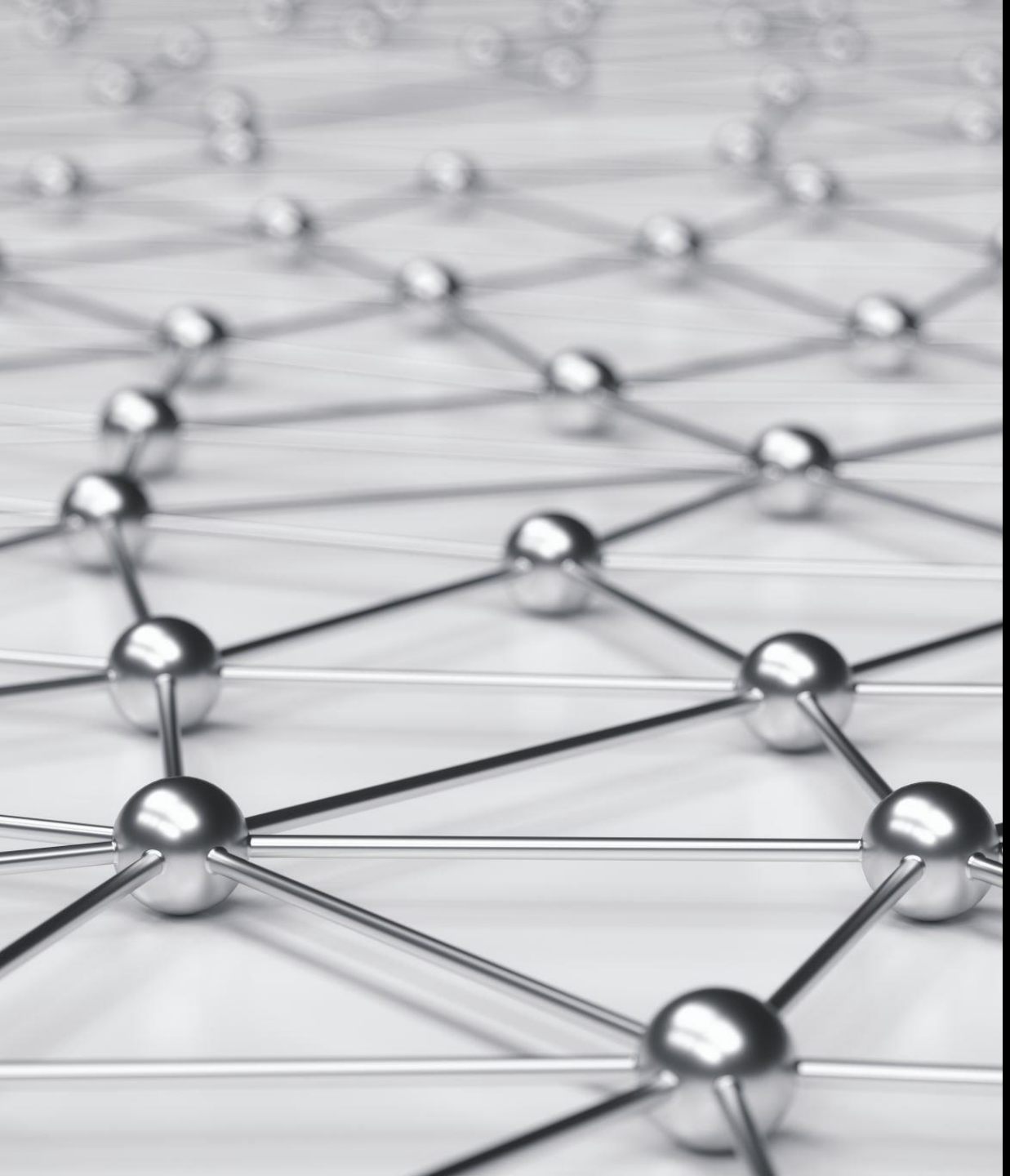


NETWORK COMPRESSION

Rohit Kumar Kaliyar

(Slide Credit: Hung-yi Lee)

Smaller Model

Less parameters

Deploying ML models in resource-constrained environments

Lower latency, Privacy, etc.

Network Pruning

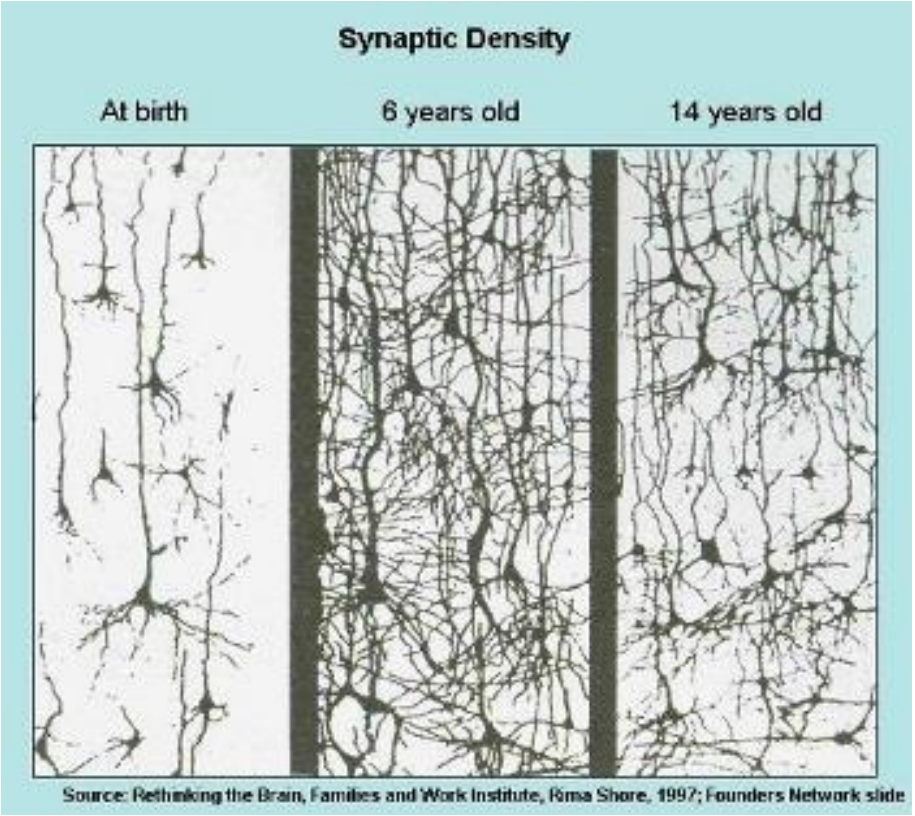
Network can be pruned

- Networks are typically over-parameterized (there is significant redundant weights or neurons)
- Prune them!

Optimal Brain Damage

Yann Le Cun, John S. Denker and Sara A. Solla
AT&T Bell Laboratories, Holmdel, N. J. 07733

(NIPS, 1989)



Network Pruning

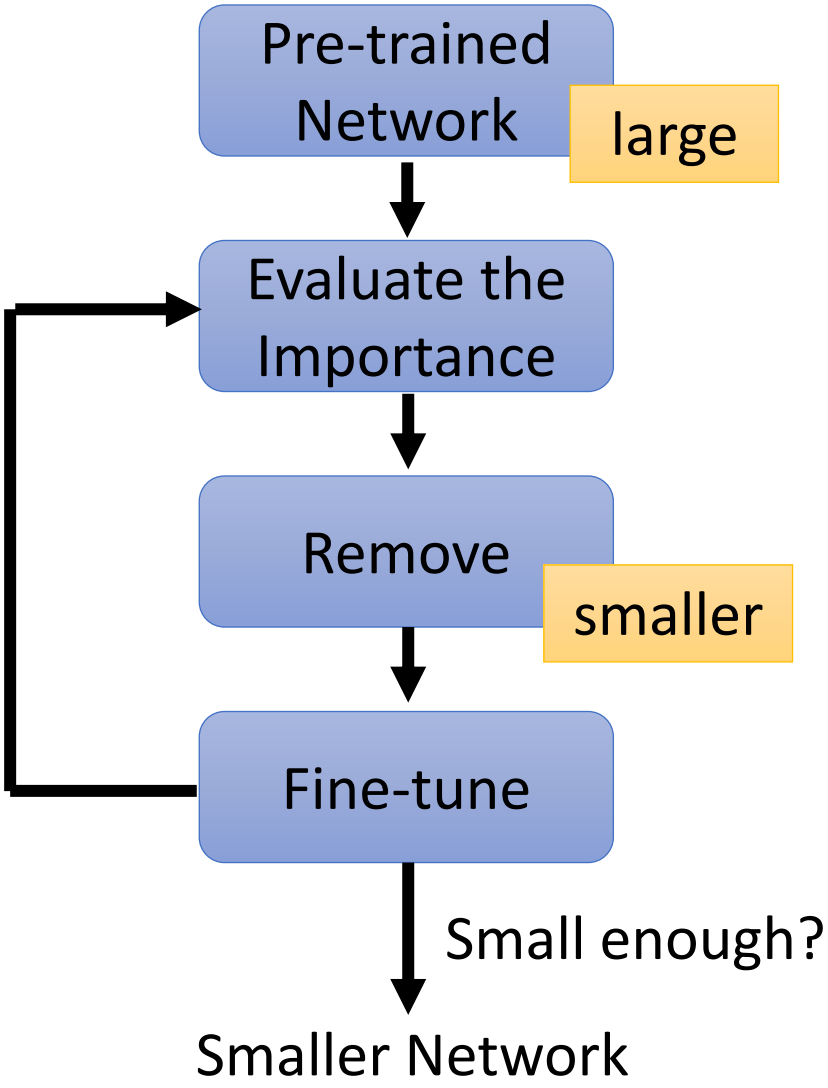
- Importance of a weight:

  absolute values, life long …

- Importance of a neuron:

  the number of times it wasn't zero on a given data set ……

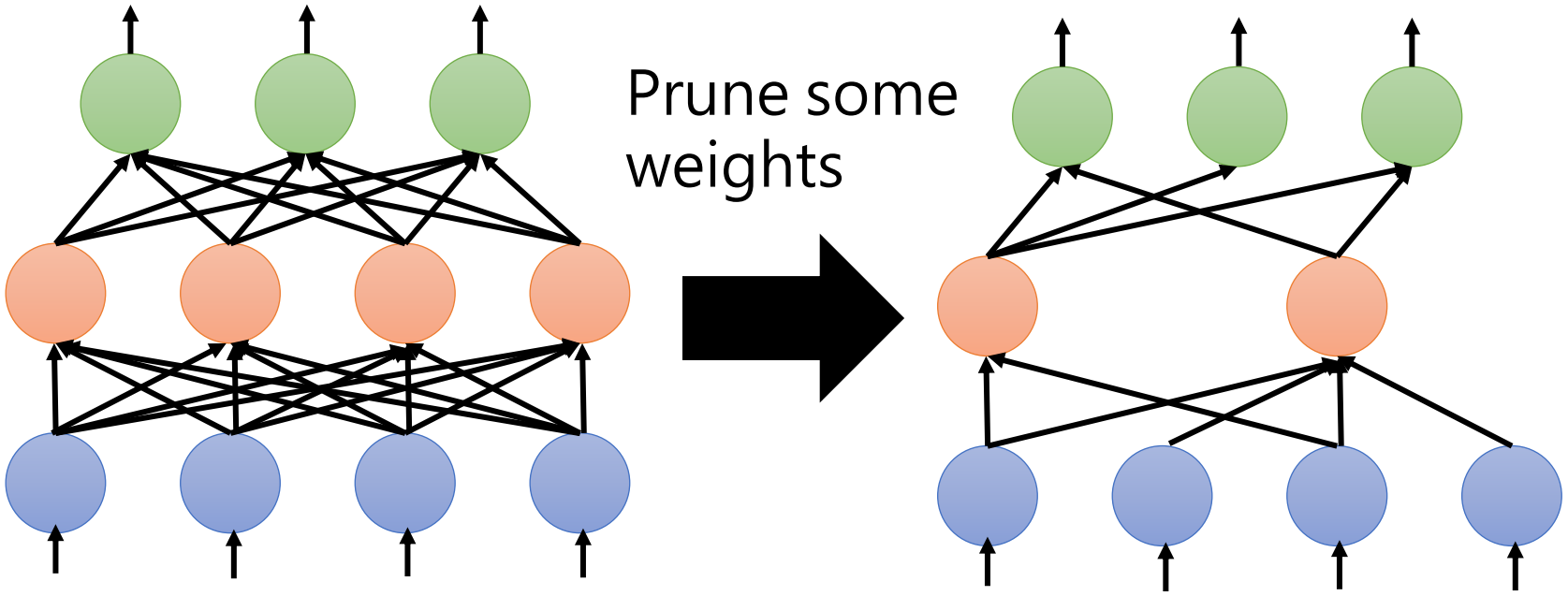
- After pruning, the accuracy will drop (hopefully not too much)
- Fine-tuning on training data for recover
- Don't prune too much at once, or the network won't recover.



Network Pruning - Practical Issue

- Weight pruning

The network architecture becomes irregular.

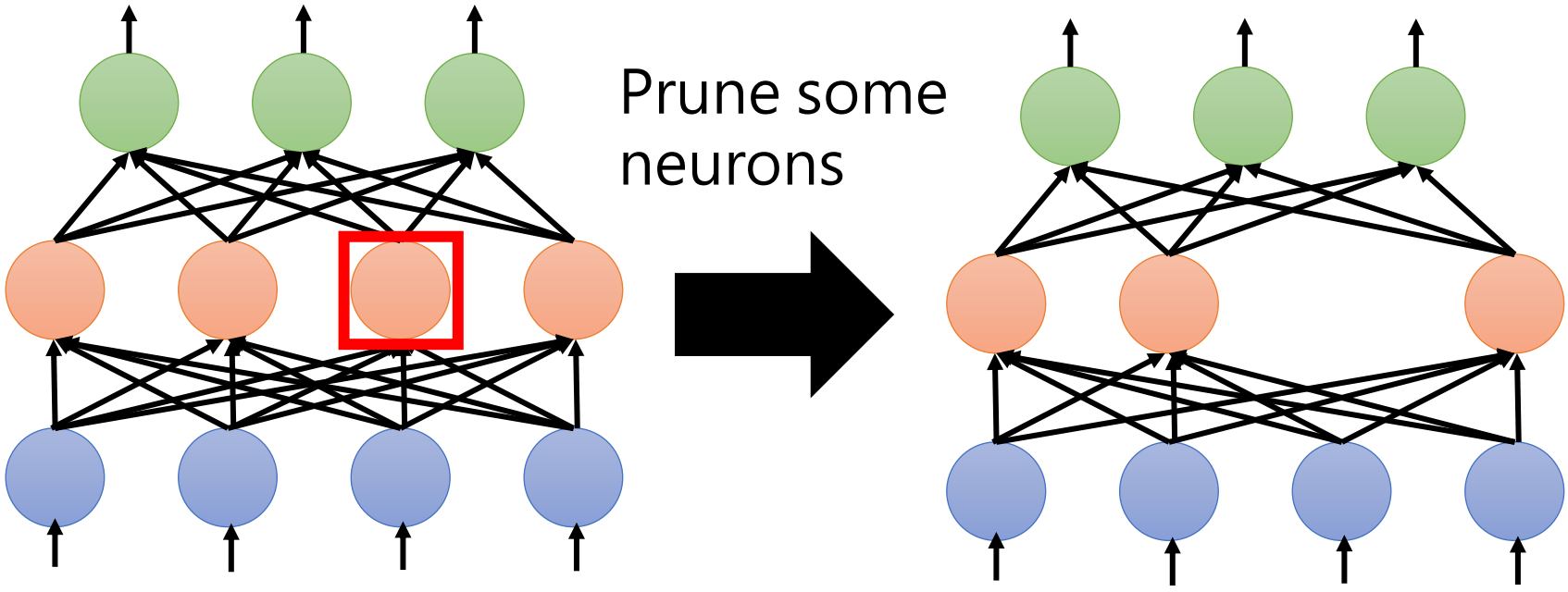


Hard to implement, hard to speedup ……

Network Pruning - Practical Issue

- Neuron pruning

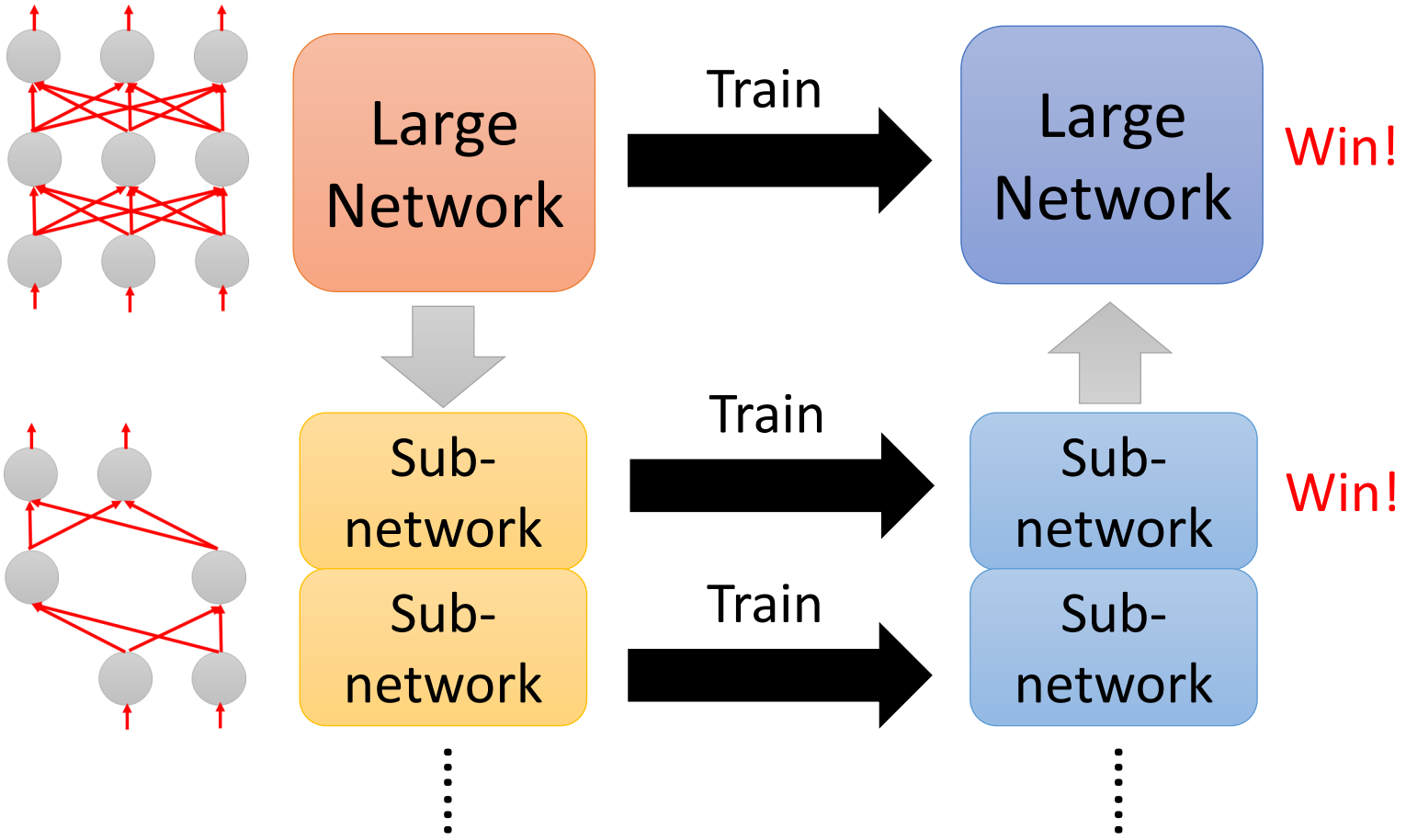
The network architecture is regular.



Easy to implement, easy to speedup ……

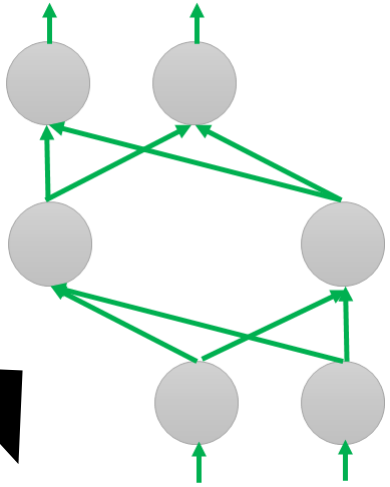
Why Pruning?

Lottery Ticket Hypothesis



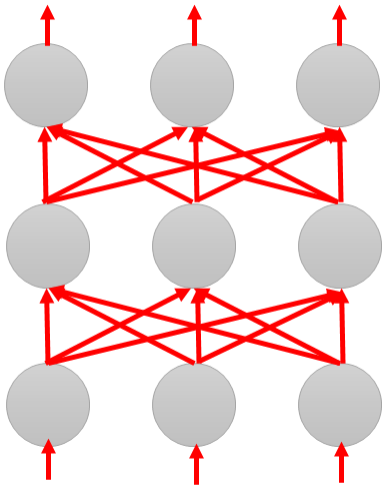
Why Pruning?

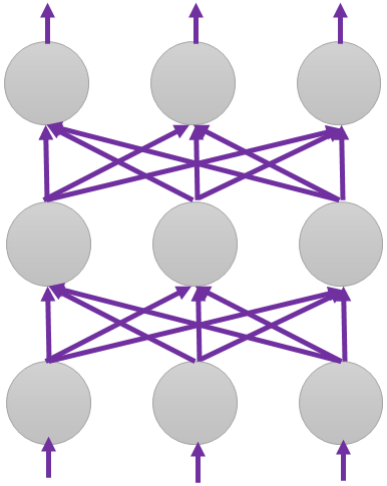
Lottery Ticket Hypothesis

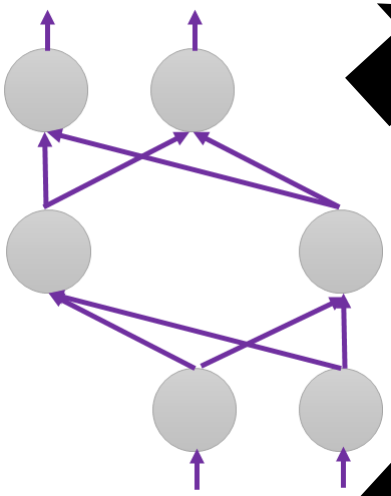


Random init

Again







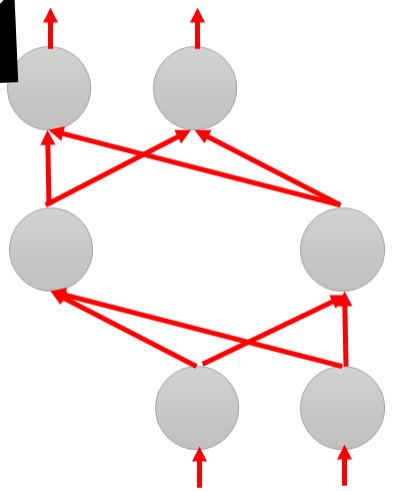
Random init

Trained

Pruned

Original

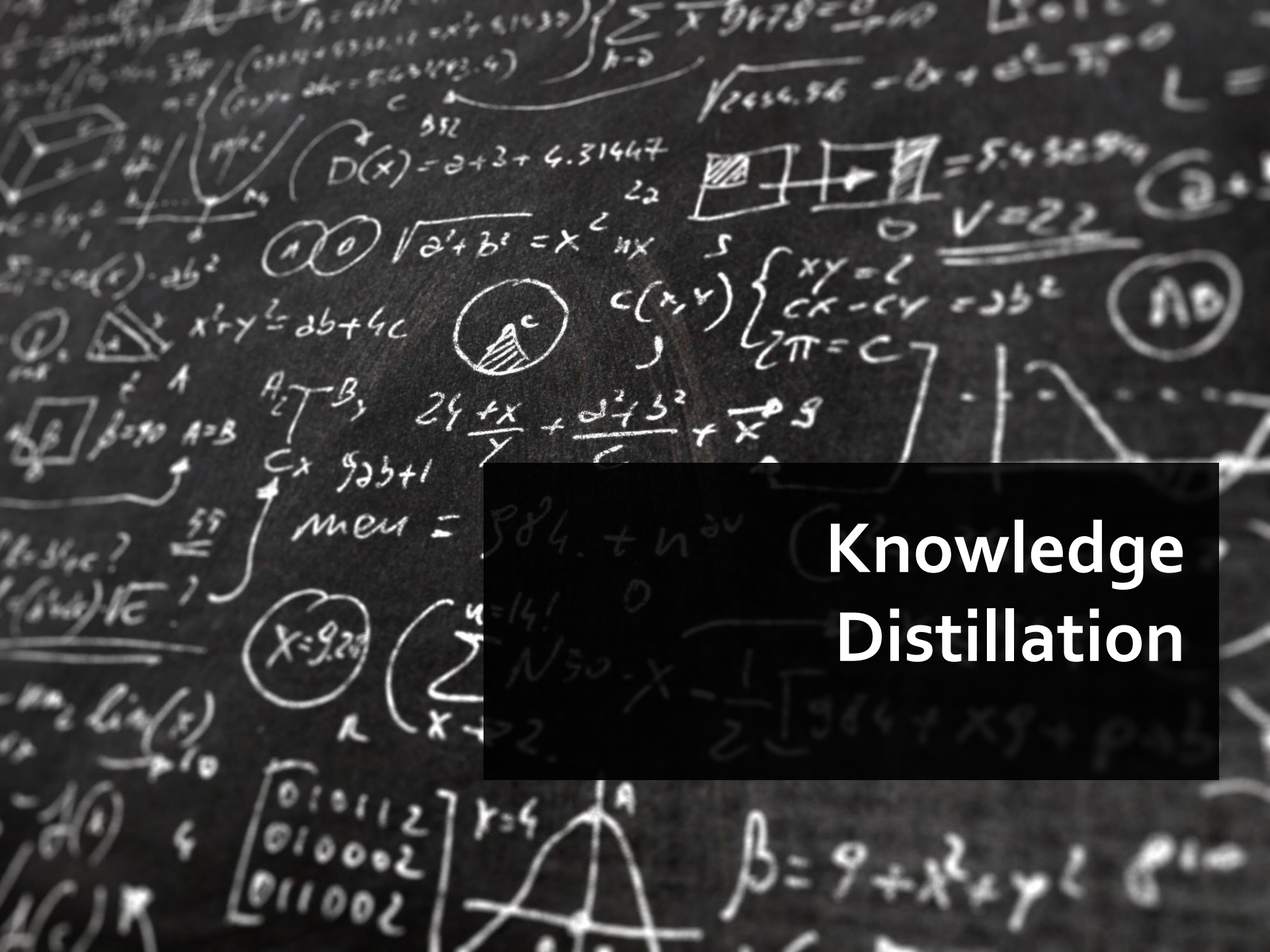
Random init



Random Init weights

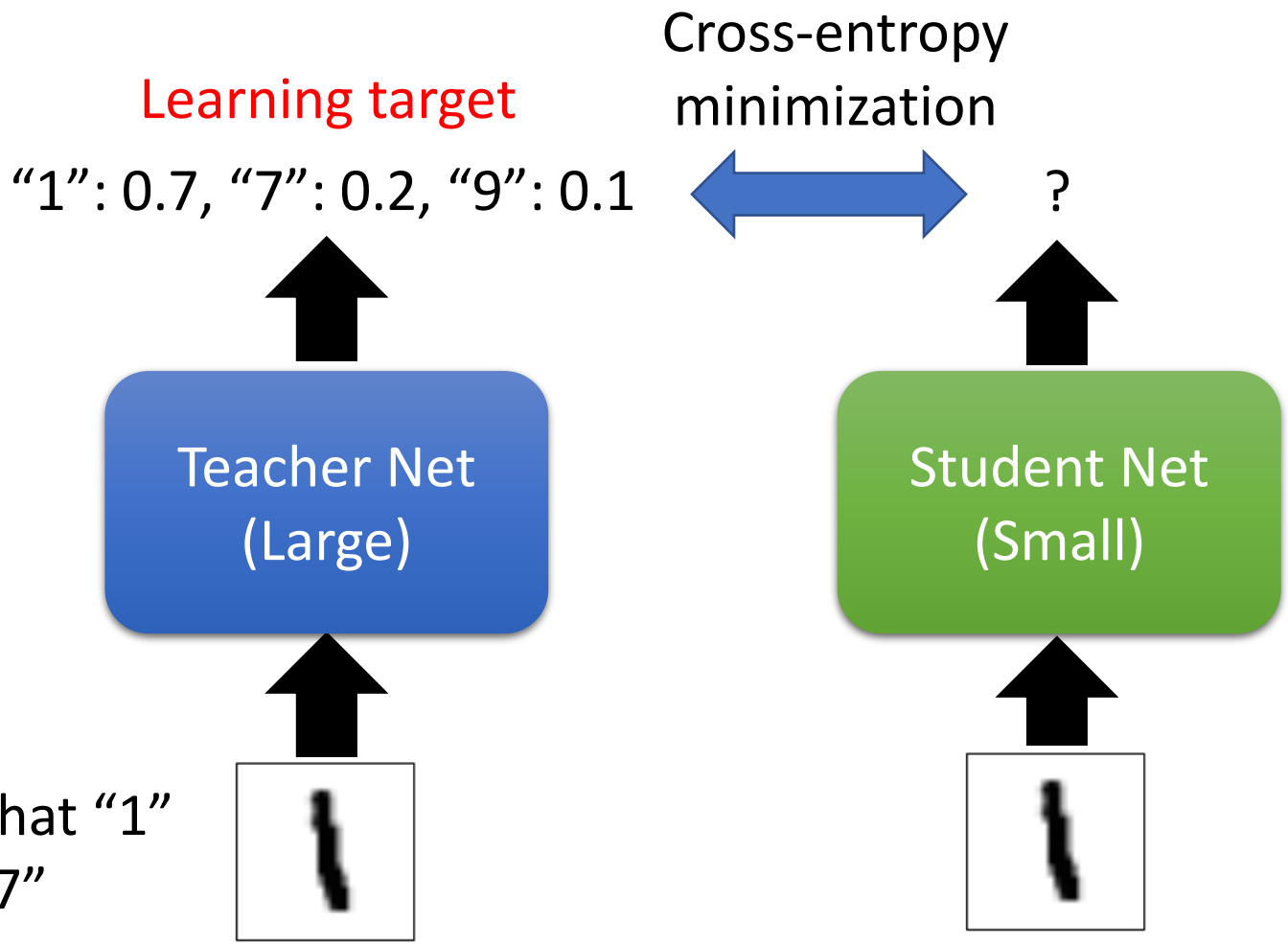
Trained weight

Another random Init weights


Knowledge
Distillation

Knowledge Distillation

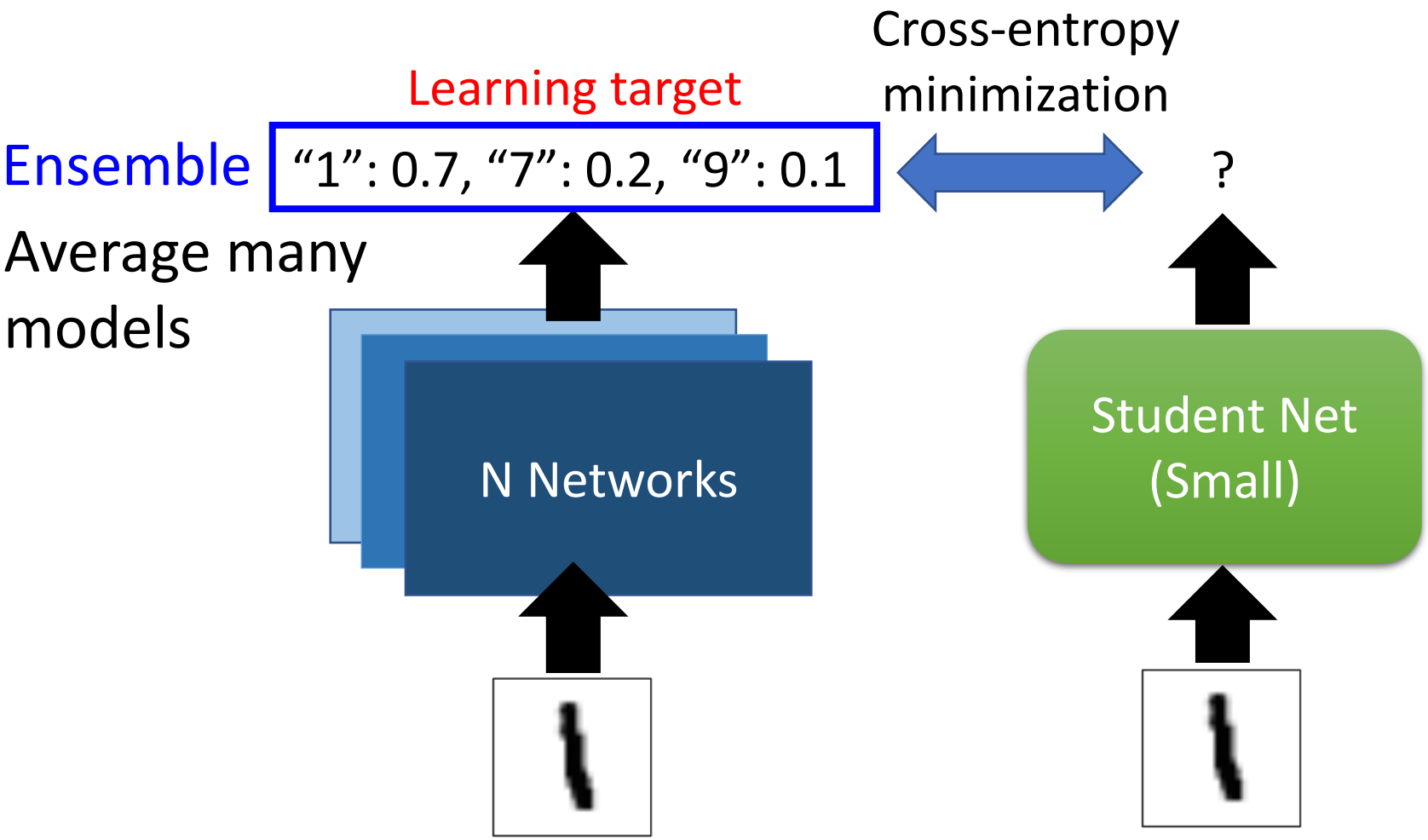
Knowledge Distillation
https://arxiv.org/pdf/1503.02531.pdf
Do Deep Nets Really Need to be Deep?
https://arxiv.org/pdf/1312.6184.pdf



Providing the information that "1" is similar to "7"

Knowledge Distillation

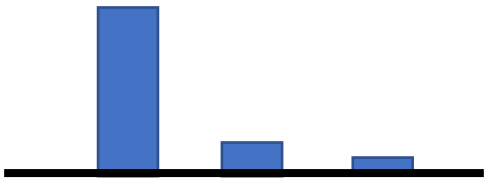
Knowledge Distillation
https://arxiv.org/pdf/1503.02531.pdf
Do Deep Nets Really Need to be Deep?
https://arxiv.org/pdf/1312.6184.pdf

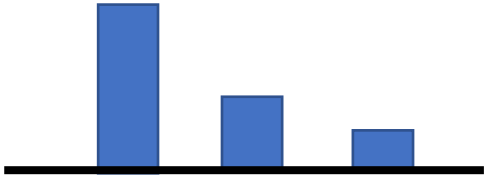


Knowledge Distillation

- Temperature for softmax

$$y_i' = \frac{exp(y_i)}{\sum_j exp(y_j)} \quad \xrightarrow{T = 100} \quad y_i' = \frac{exp(y_i/T)}{\sum_j exp(y_j/T)}$$





$y_1 = 100$ $y_1' = 1$

$y_2 = 10$ $y_2' \approx 0$

$y_3 = 1$ $y_3' \approx 0$

$y_1/T = 1$ $y_1' = 0.56$

$y_2/T = 0.1$ $y_2' = 0.23$

$y_3/T = 0.01$ $y_3' = 0.21$

Parameter Quantization

Parameter Quantization

- 1. Using less bits to represent a value
- 2. Weight clustering

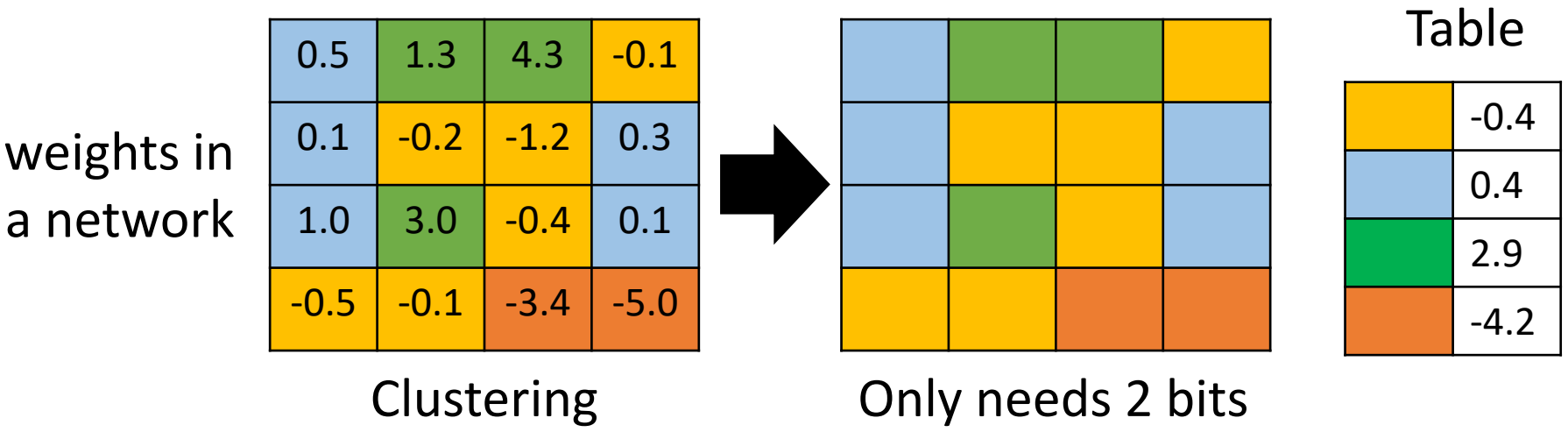
weights in a network

0.5	1.3	4.3	-0.1
0.1	-0.2	-1.2	0.3
1.0	3.0	-0.4	0.1
-0.5	-0.1	-3.4	-5.0

Clustering

Parameter Quantization

- 1. Using less bits to represent a value
- 2. Weight clustering



- 3. Represent frequent clusters by less bits, represent rare clusters by more bits
  - e.g. Huffman encoding

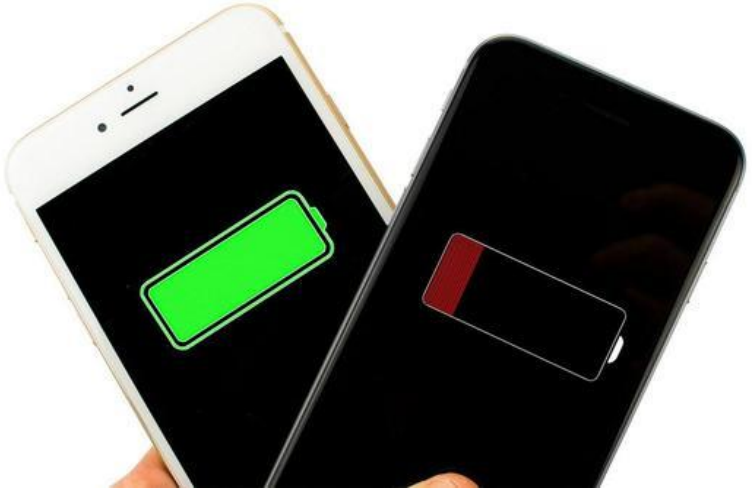
Dynamic Computation

Dynamic Computation

- The network adjusts the computation it need.

Different devices

high/low battery



- Why don’t we prepare a set of models?

Dynamic Depth

$$L = e_1 + e_2 + \cdots + e_L$$

high battery

Layer 1

Layer 2

…

Layer L

$\boldsymbol{y}$ ⇠ e_L ⇢ $\hat{\boldsymbol{y}}$

Extra Layer 1

Extra Layer 2

Does it work well?

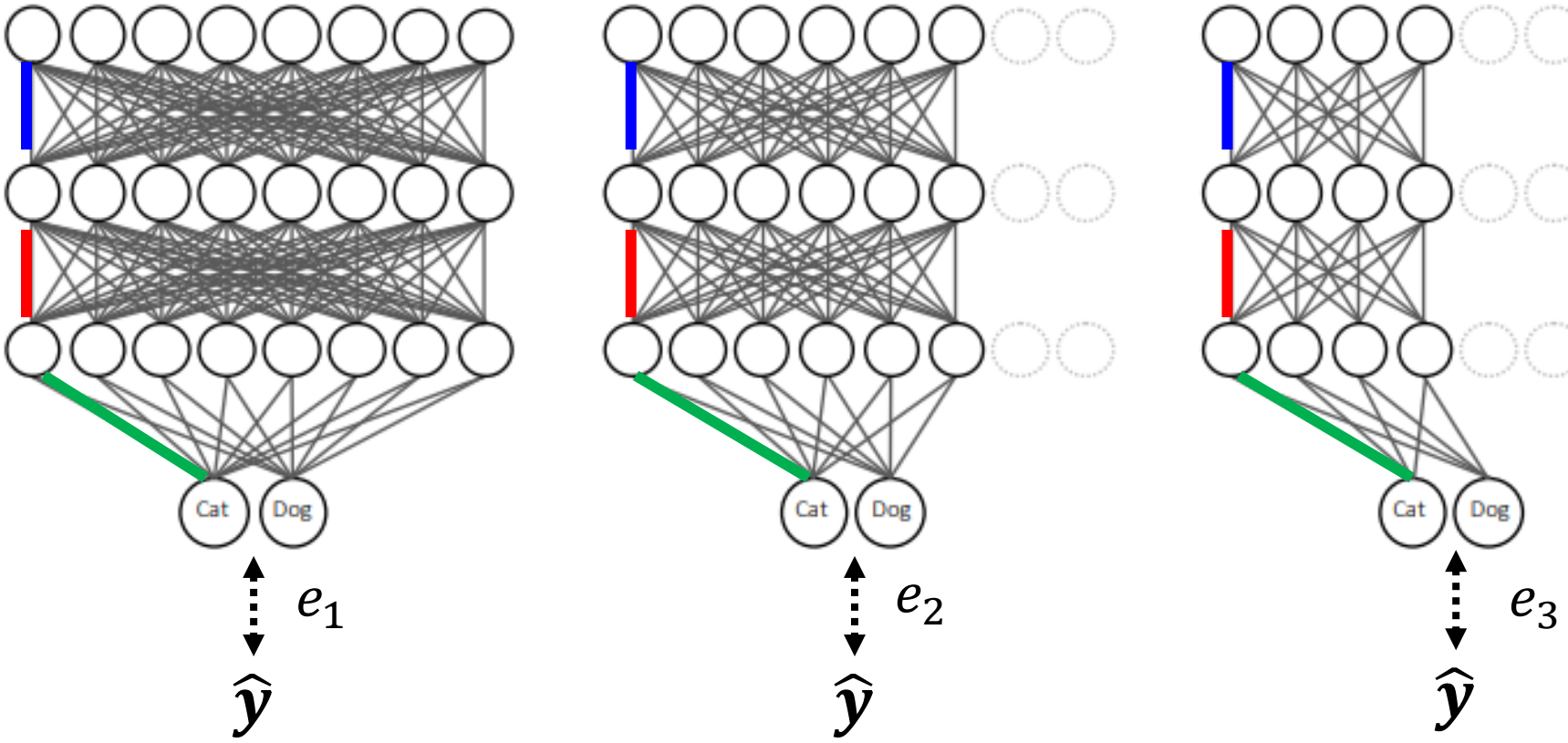
low battery

$\boldsymbol{y}'$ ⇕ e_1 $\hat{\boldsymbol{y}}$

$\boldsymbol{y}''$ ⇕ e_2 $\hat{\boldsymbol{y}}$

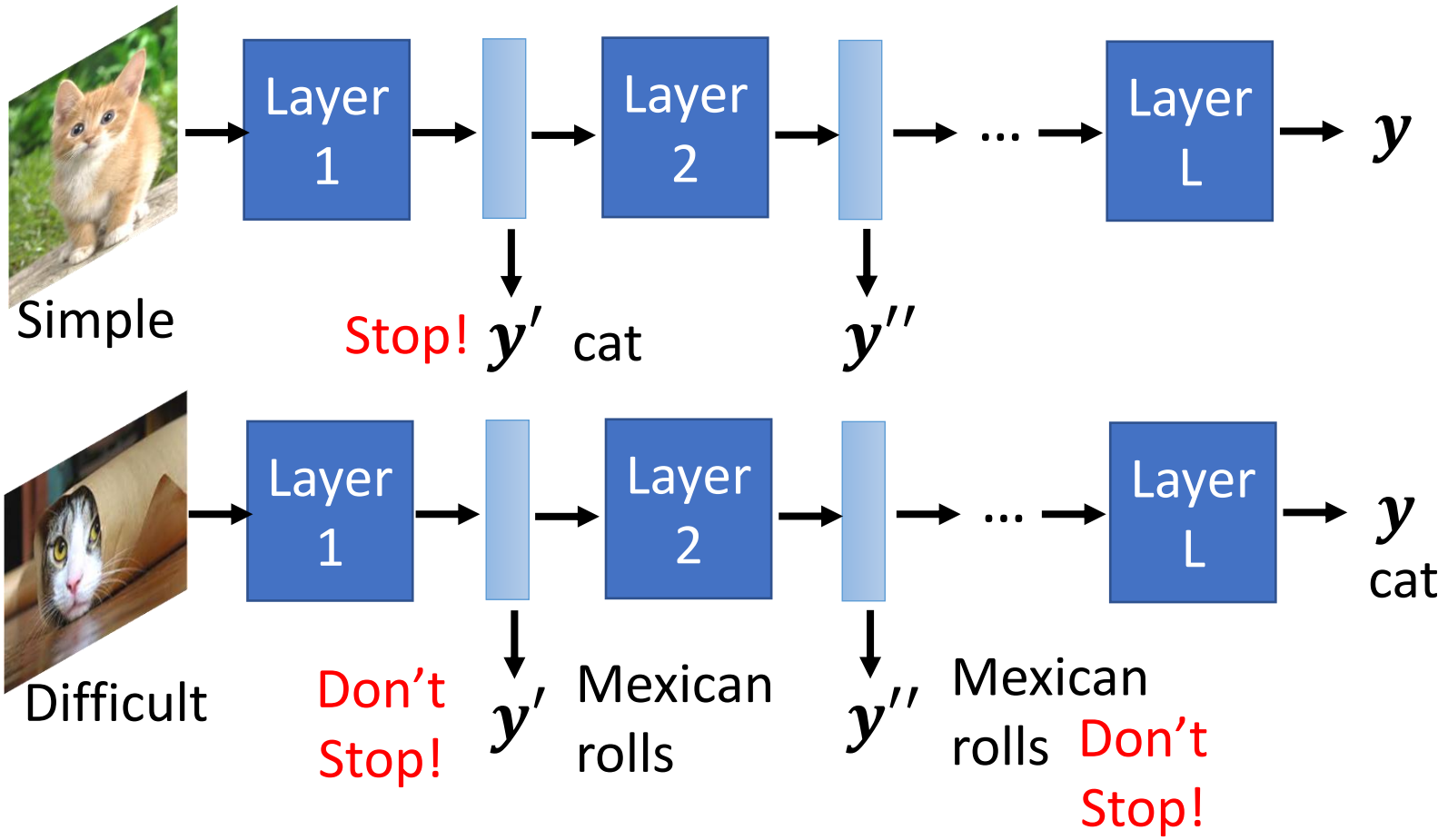
Dynamic Width

$$L = e_1 + e_2 + e_3$$



Slimmable Neural Networks
https://arxiv.org/abs/1812.08928

Computation based on Sample Difficulty



- SkipNet: Learning Dynamic Routing in Convolutional Networks
- Runtime Neural Pruning
- BlockDrop: Dynamic Inference Paths in Residual Networks