LG전자 AI Scientist 고급과정

Pruning (Part I)

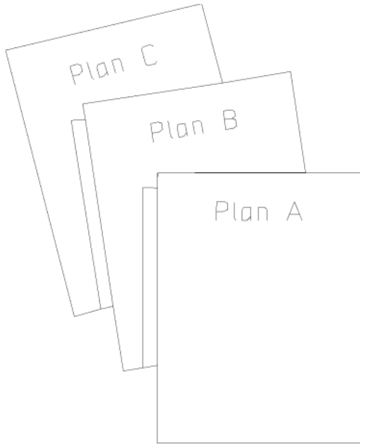
서울대학교 컴퓨터공학부 이 영 기

Human-Centered Computer Systems Lab

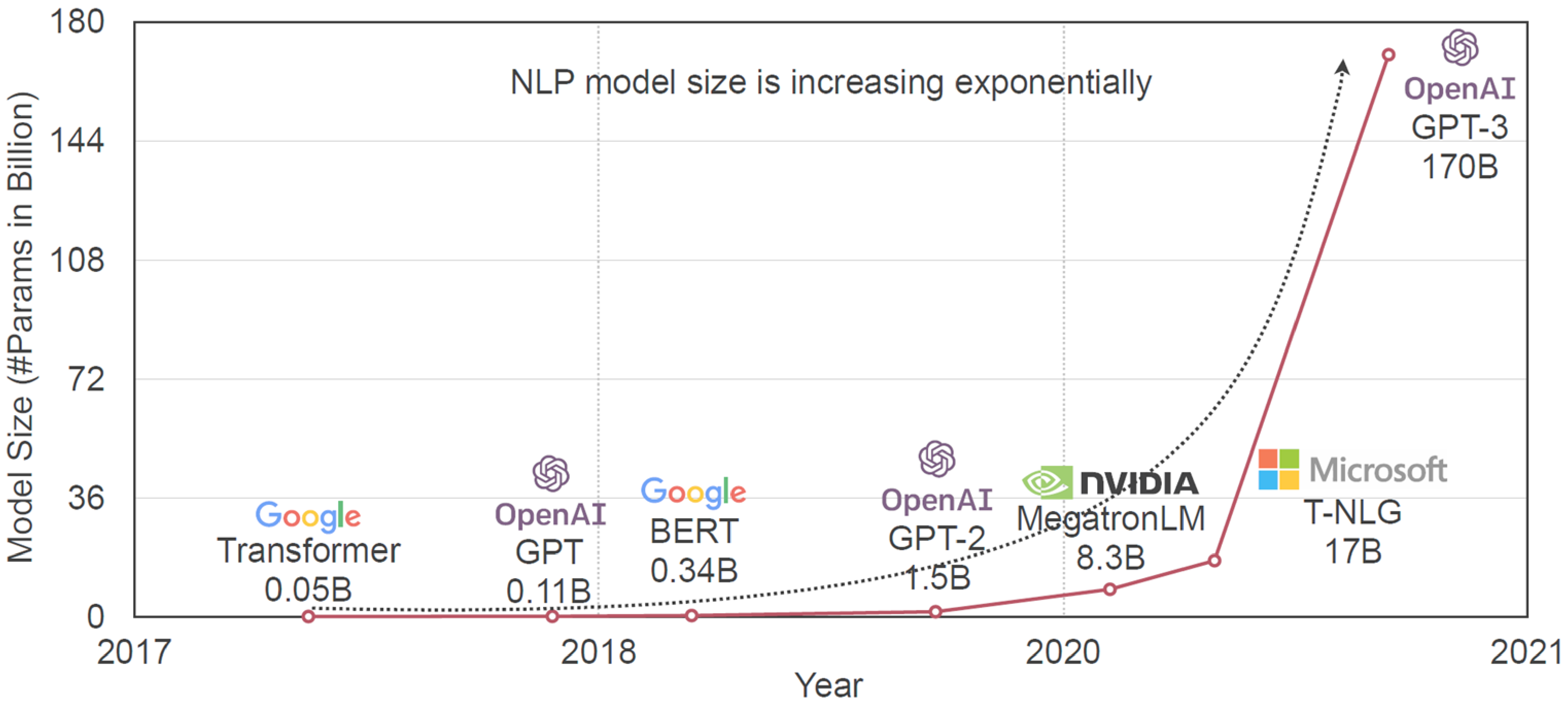
SEOUL NATIONAL UNIVERSITY

Overview

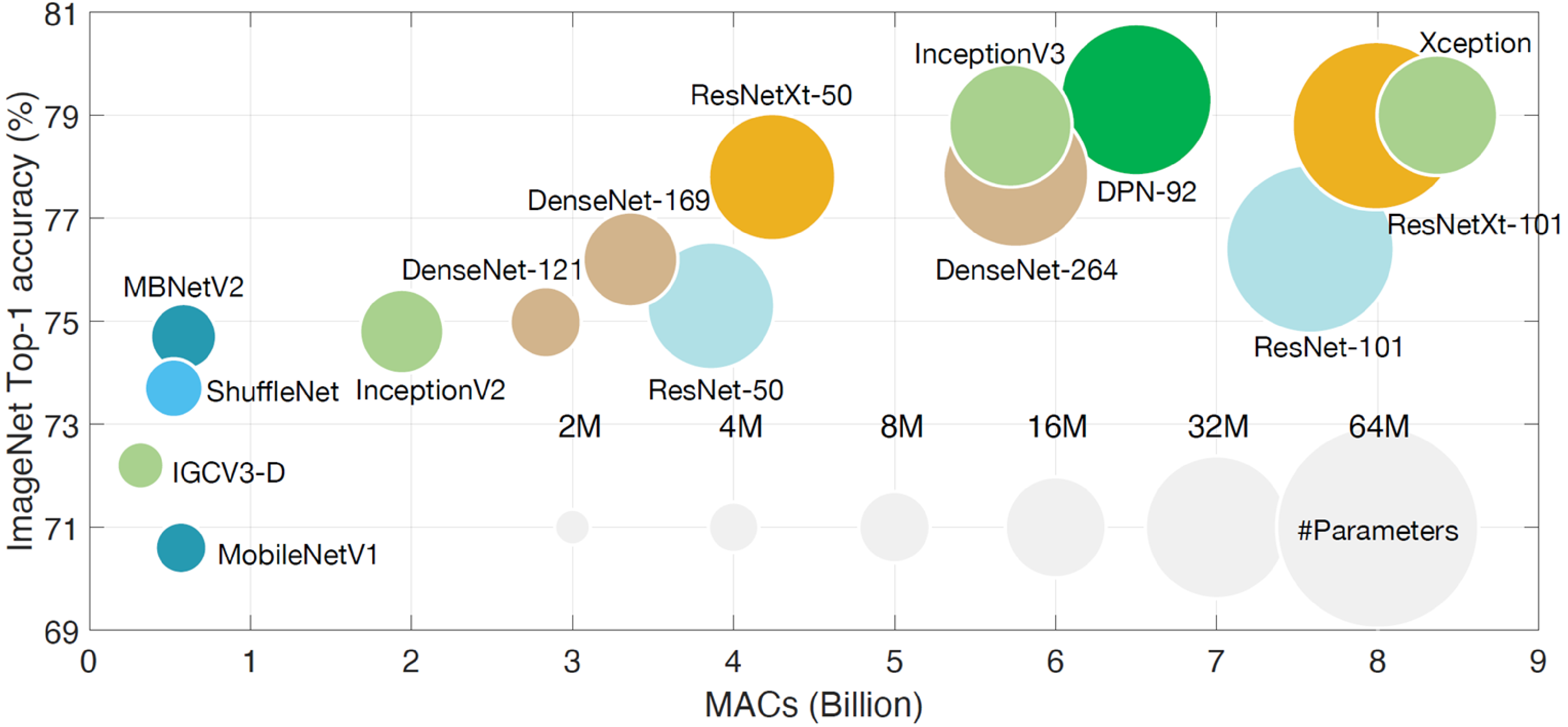
- Objective
  - Understand neural network pruning to reduce the storage and computation requirements
  - Learn different granularities and criteria of neural network pruning
- Content
  - Introduction to pruning
  - Pruning granularity: Fine-grained / Pattern-based / Channel-level pruning
  - Pruning criterion: Magnitude-based / Scaling-based pruning
- After this module, you should be able to
  - Grasp the concept of neural network pruning and its effects on deep learning models
  - Understand various types of pruning and their advantages and disadvantages



Today’s AI is too BIG!



Today's AI is too BIG!



MLPerf (the Olympic Game for AI Computing)

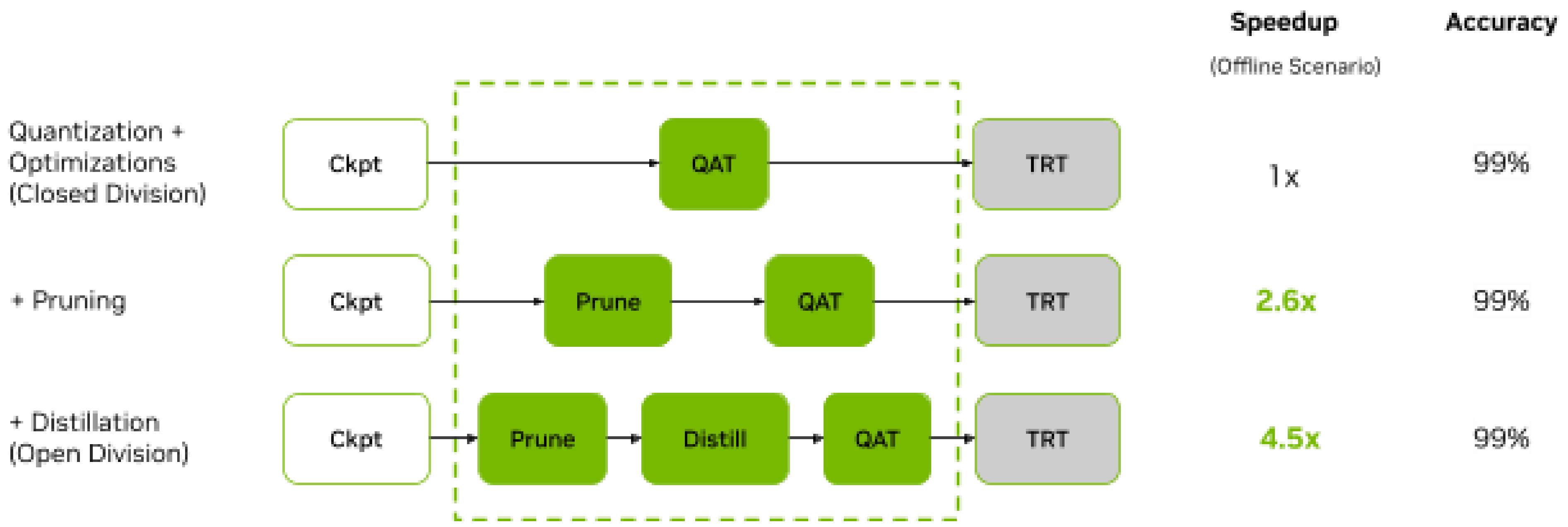
- The open division submission on BERT
  - More than 4x while maintaining 99% accuracy

	Closed Division	Open Division	Speedup
Offline samples/sec	1029	4609	4.5x

https://developer.nvidia.com/blog/leading-mlperf-inference-v3-1-results-gh200-grace-hopper-superchip-debut/

MLPerf (the Olympic Game for AI Computing)

- The key techniques are pruning, distillation and quantization



	Speedup (Offline Scenario)	Accuracy
Quantization + Optimizations (Closed Division)	1x	99%
+ Pruning	2.6x	99%
+ Distillation (Open Division)	4.5x	99%

https://developer.nvidia.com/blog/leading-mlperf-inference-v3-1-results-gh200-grace-hopper-superchip-debut/

Neural Network Pruning

- A widely-used technique that can reduce the parameter counts of neural networks by more than 90%.
- It decreases the storage(memory) requirements and improves computation efficiency of neural networks.

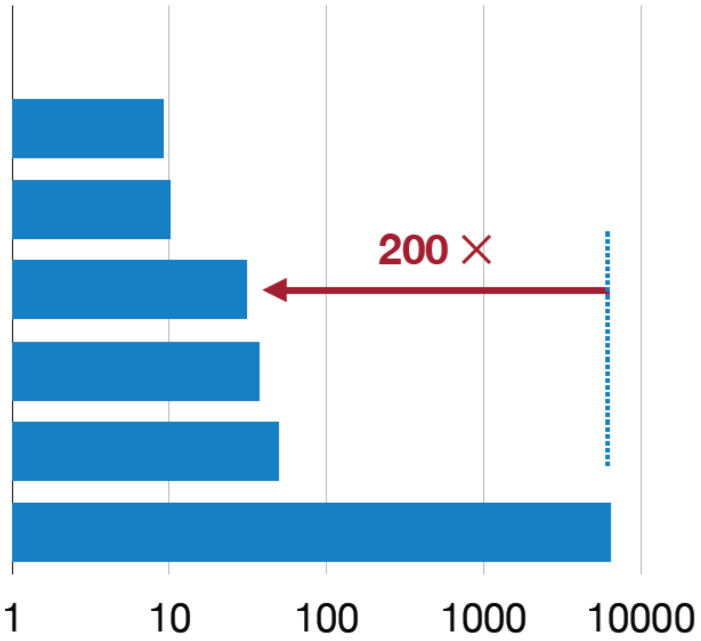
Memory is Expensive

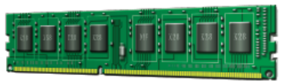
- Data movement □ More memory reference □ more energy

Operation	Energy [pJ]
32 bit int ADD	0.1
32 bit float ADD	0.9
32 bit Register File	1
32 bit int MULT	3.1
32 bit float MULT	3.7
32 bit SRAM Cache	5
32 bit DRAM Memory	640

Rough Energy Cost For Various Operations in 45nm 0.9V

Relative Energy Cost

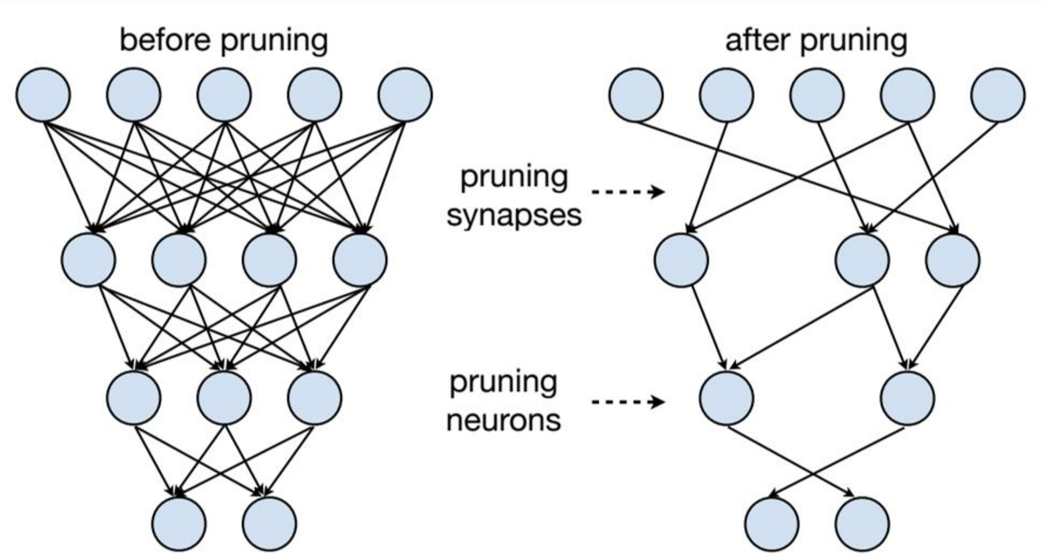


1  = 200 ×+

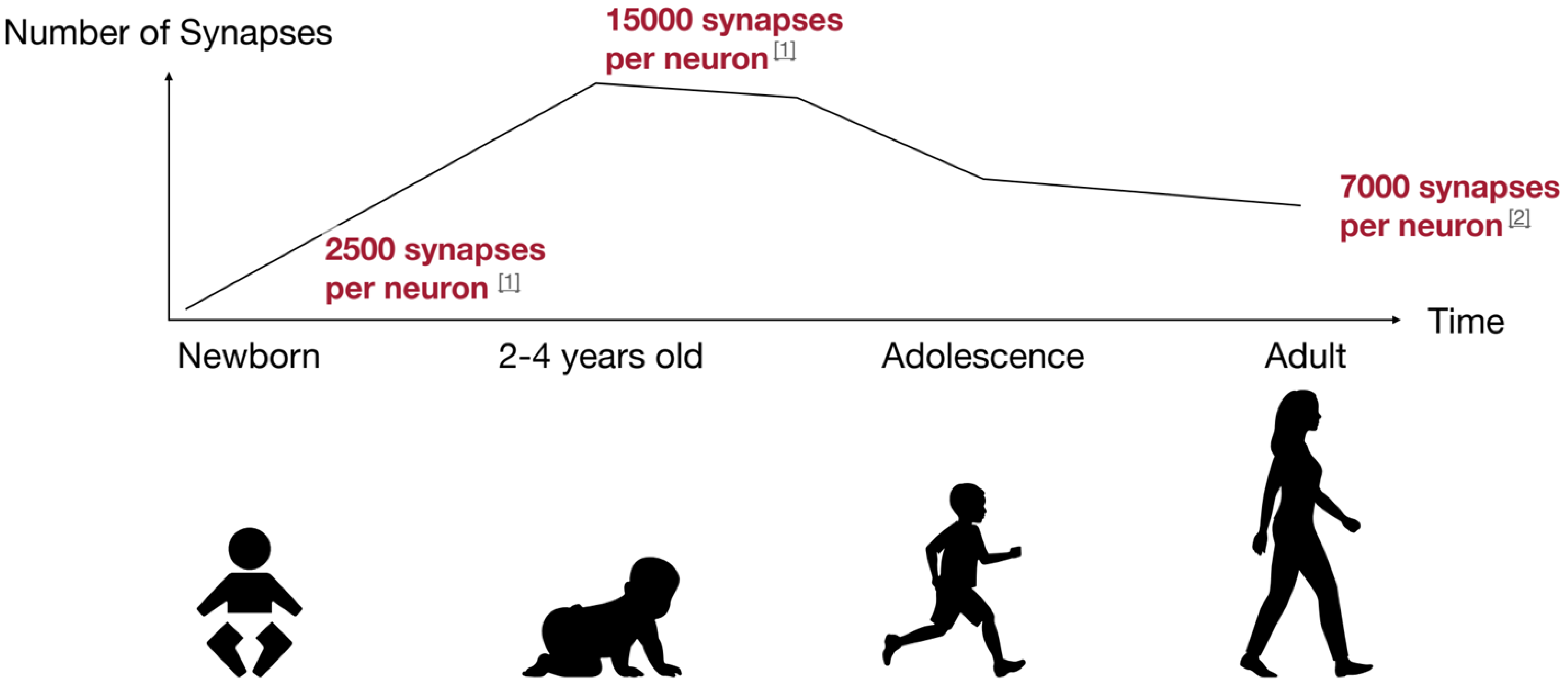
How should we make deep learning more efficient?

Neural Network Pruning

- Introduction to Pruning
  - What is pruning?
  - How should we formulate pruning?
- Determine the Pruning Granularity
  - In what pattern should we prune the neural network?
- Determine the Pruning Criterion
  - What synapses/neurons should we prune?
- Determine the Pruning Ratio
  - What should the target sparsity be for each layer?
- Fine-tune/Train Pruned Neural Network
  - How should we improve the performance of pruned models?



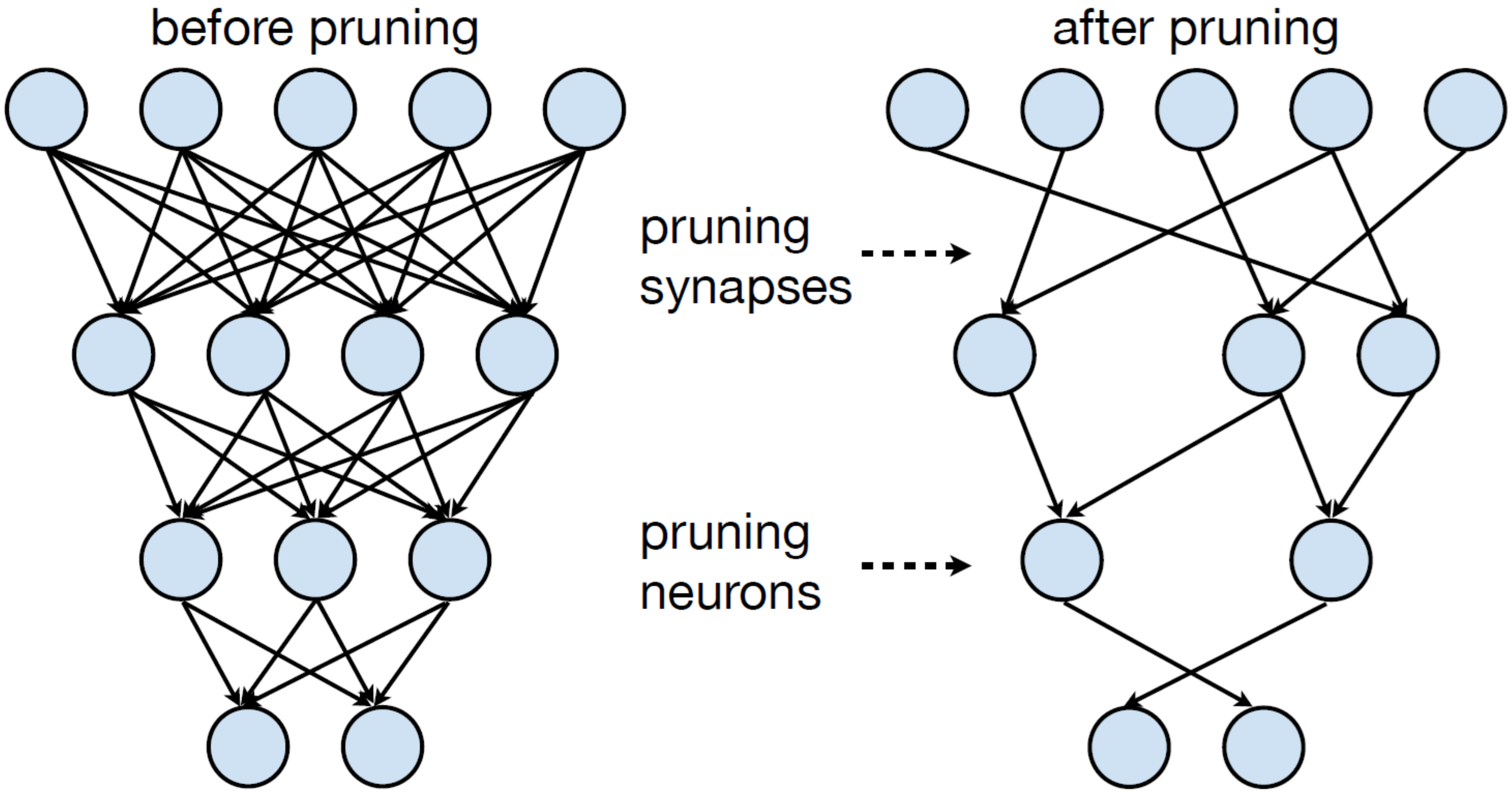
Pruning Happens in Human Brain



Drachman D.A., “Do We Have Brain to Spare“, Neurology 2004
Walsh, C.A., “Peter Huttenlocher (1931-2013)”, Nature 2013

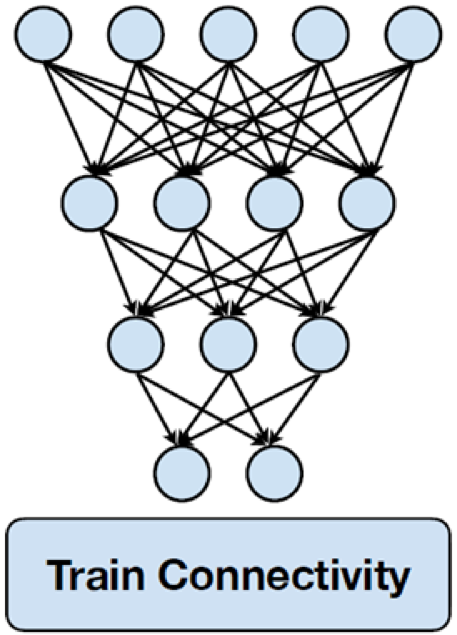
Neural Network Pruning

- Make neural network smaller by removing synapses and neurons

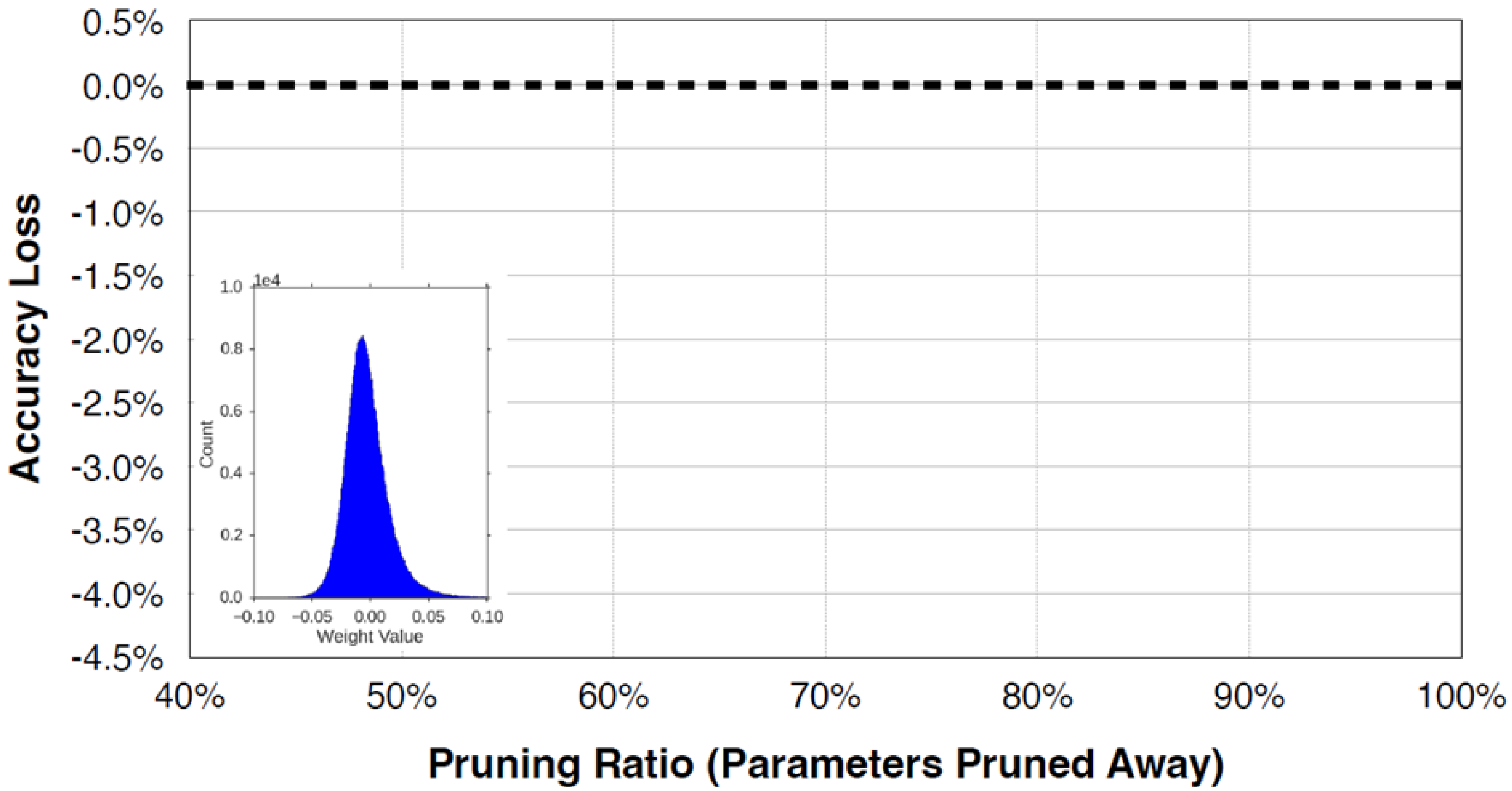


Han et al. "Learning both weights and connections for efficient neural network.“, NeurIPS 2015

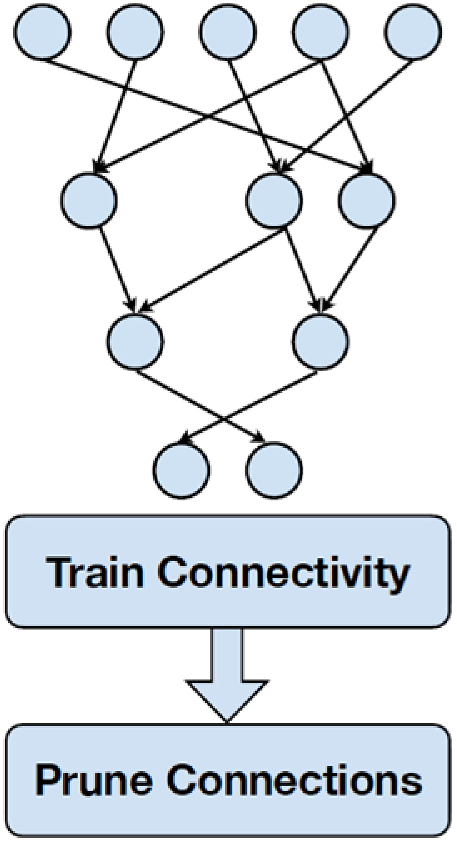
Neural Network Pruning



Train Connectivity

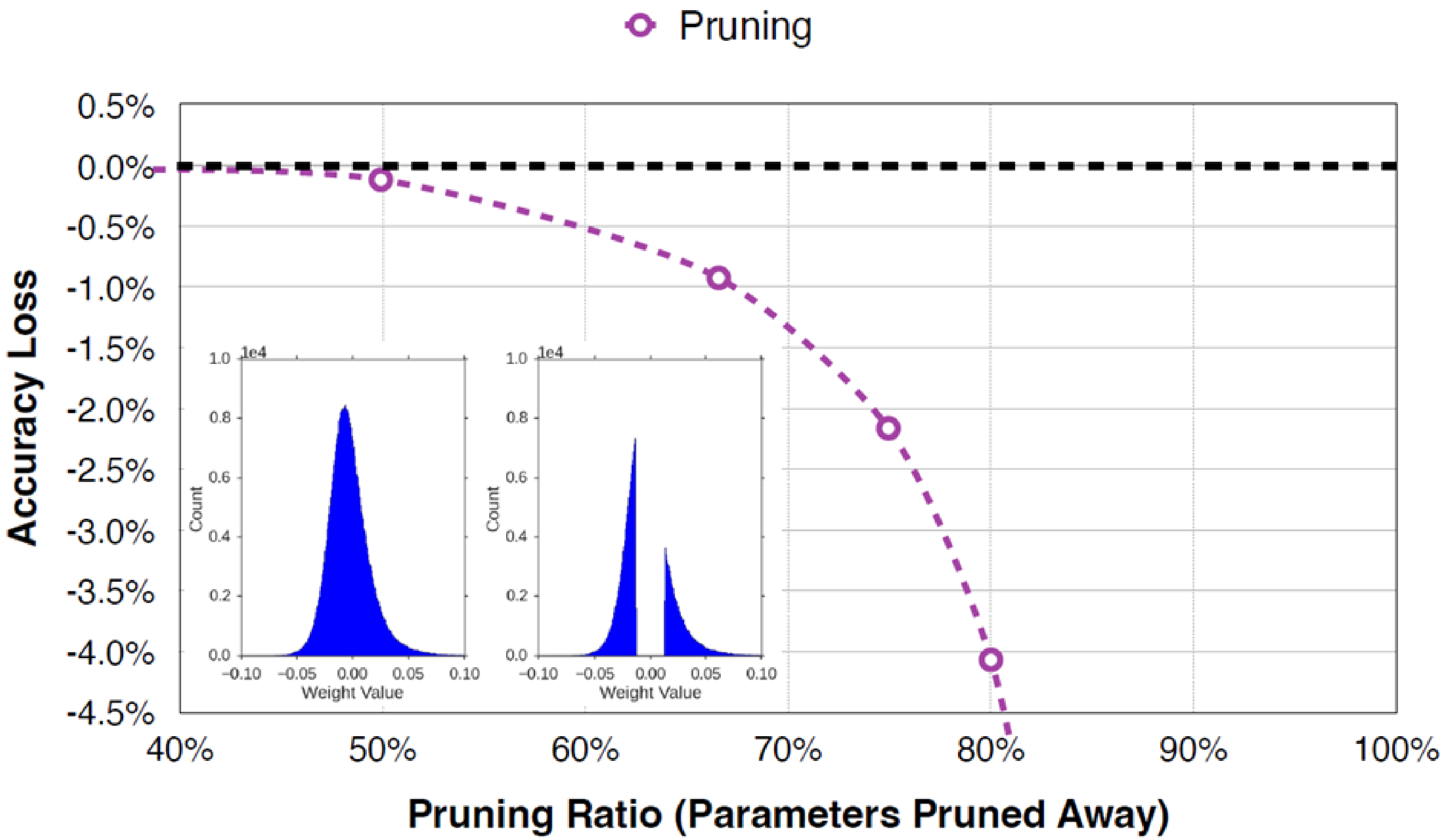


Neural Network Pruning

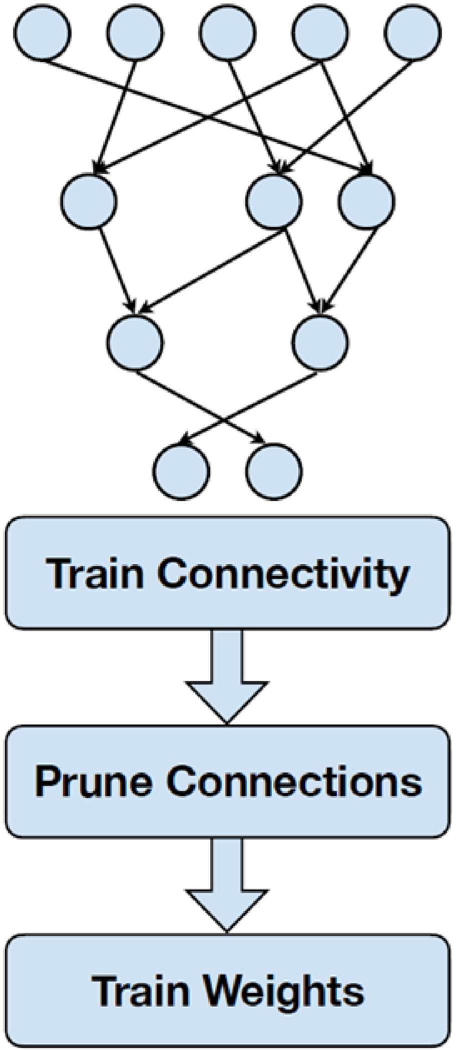


Train Connectivity

Prune Connections



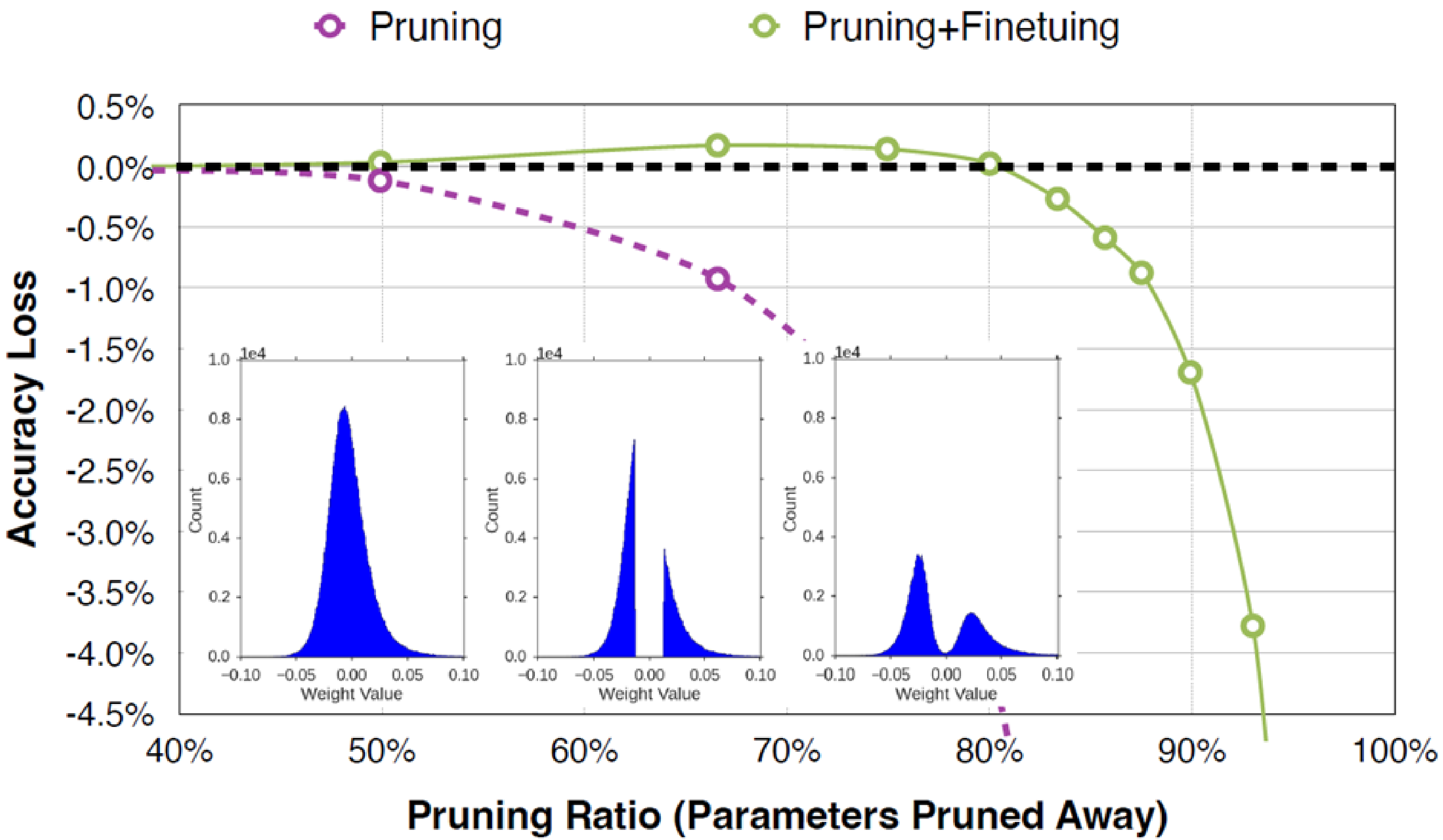
Neural Network Pruning



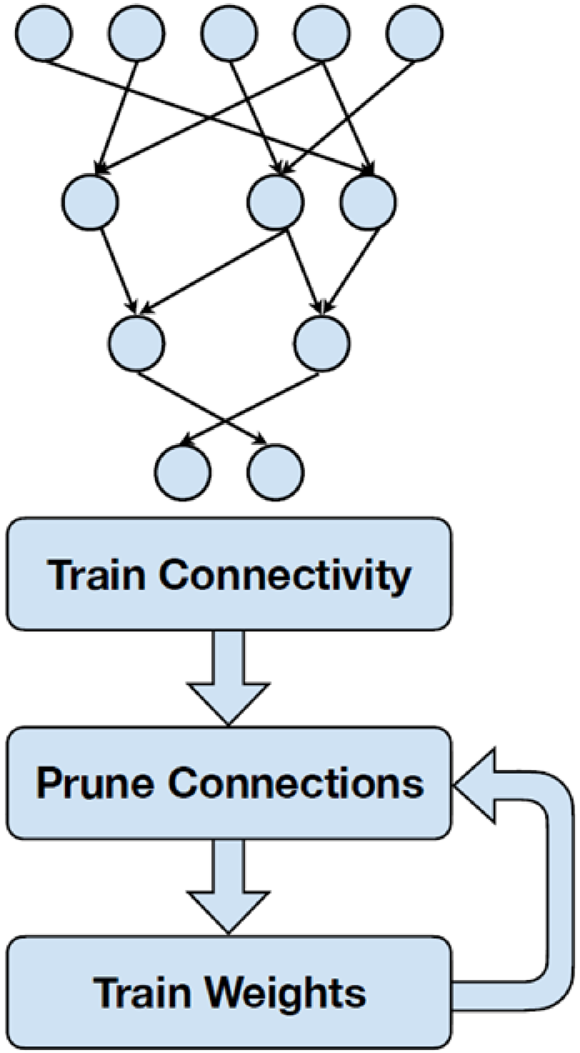
Train Connectivity

Prune Connections

Train Weights



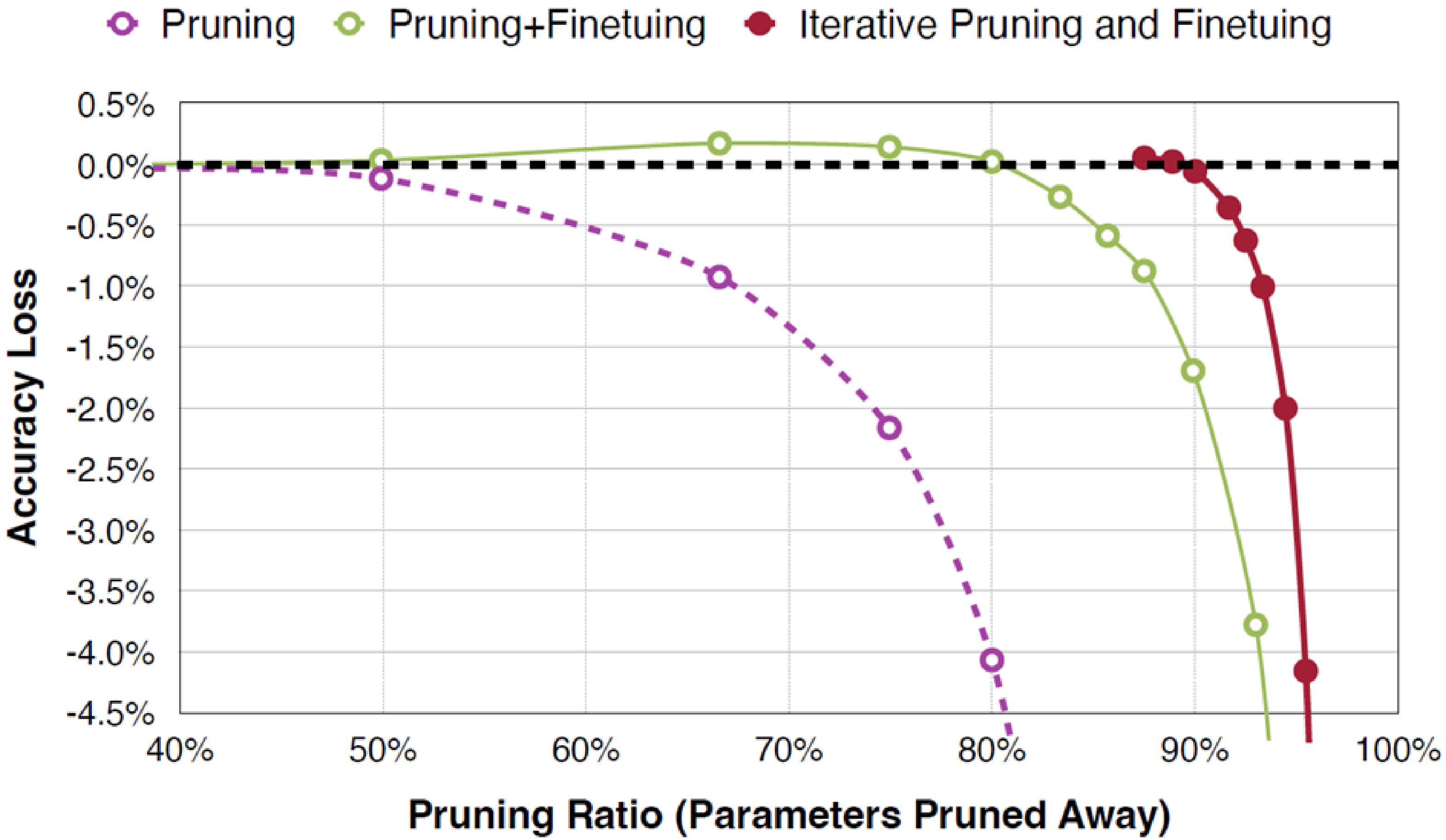
Neural Network Pruning



Train Connectivity

Prune Connections

Train Weights



Neural Network Pruning

Neural Network	# Parameters			MACs
	Before Pruning	After Pruning	**Reduction**	Reduction
AlexNet	61 M	6.7 M	**9 x**	3 x
VGG-16	138 M	10.3 M	**12 x**	5 x
GoogleNet	7 M	2.0 M	**3.5 x**	5 x
ResNet50	26 M	7.47 M	**3.4 x**	6.3 x
SqueezeNet	1M	0.38 M	**3.2 x**	3.5 x

Pruning saves up to 12x parameter storage without accuracy drop

Neural Network Pruning: Example

- Pruning the NeuralTalk LSTM does not hurt image caption quality.

Baseline: a basketball player in a white uniform is playing with a ball .

Pruned 90%: a basketball player in a white uniform is playing with a basketball.

Baseline: a brown dog is running through a grassy field.

Pruned 90%: a brown dog is running through a grassy area.

Baseline: a man is riding a surfboard on a wave.

Pruned 90%: a man in a wetsuit is riding a wave on a beach.

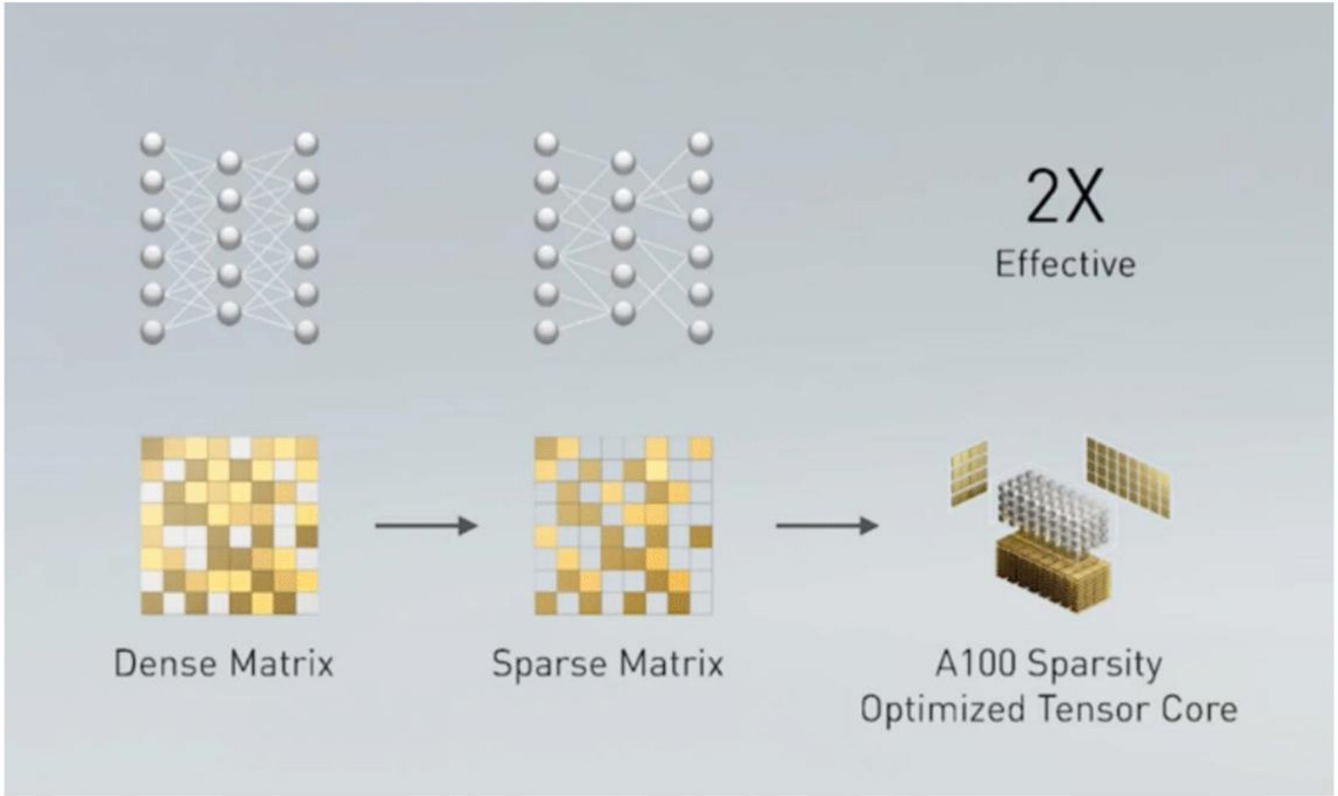
Baseline: a soccer player in red is running in the field.

Pruned 95%: a man in a red shirt and black and white black shirt is running through a field.

Han S., “Efficient Methods and Hardware for Deep Learning”

Pruning in the Industry

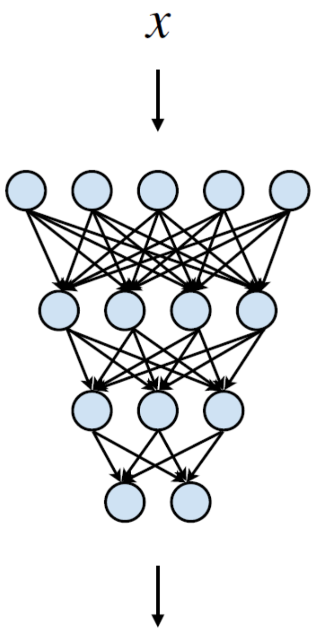
Hardware support for Sparsity for Nvidia A100 GPU



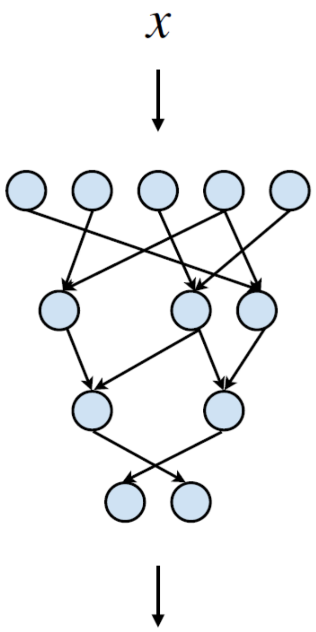
2:4 sparsity in A100 GPU
2X peak performance, 1.5X measured BERT speedup

How Should We Formulate Pruning?

- In general, we could formulate the pruning as follows:
  $\arg\min_{\mathbf{W}_P} L(\mathbf{x}; \mathbf{W}_P)$ subject to $\|\mathbf{W}_p\|_0 < N$
- L represents the objective function for neural network training.
- X is input, W is original weights, W$_p$ is pruned weights.
- $\|W_p\|_0$ calculates the #nonzeros in W$_p$, and N is the target #nonzeros .



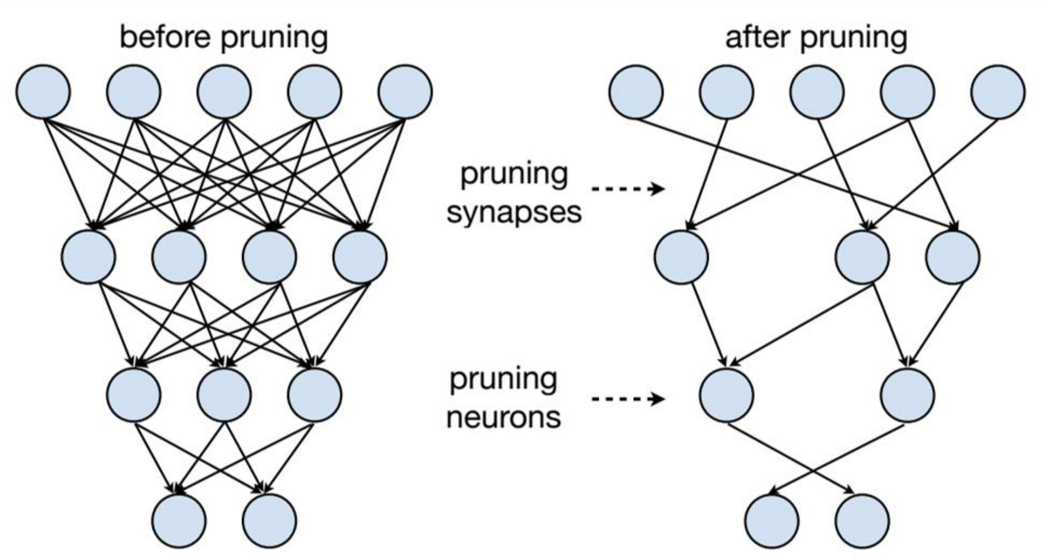
$$\arg\min_{\mathbf{W}} L(\mathbf{x}; \mathbf{W})$$



$$\arg\min_{\mathbf{W}_P} L(\mathbf{x}; \mathbf{W}_P)$$
$$s.t. \|\mathbf{W}_P\|_0 \leq N$$

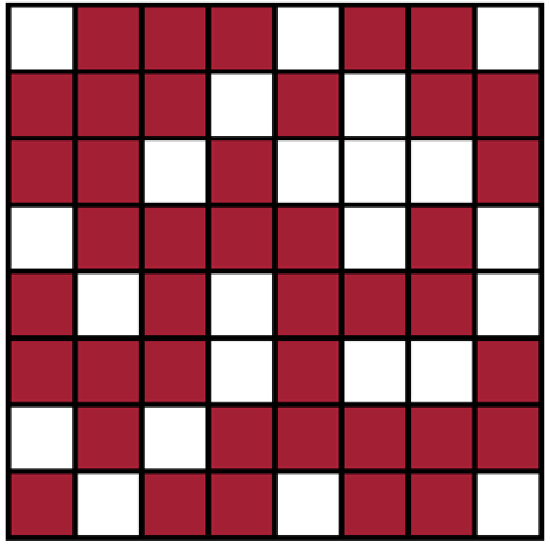
Neural Network Pruning

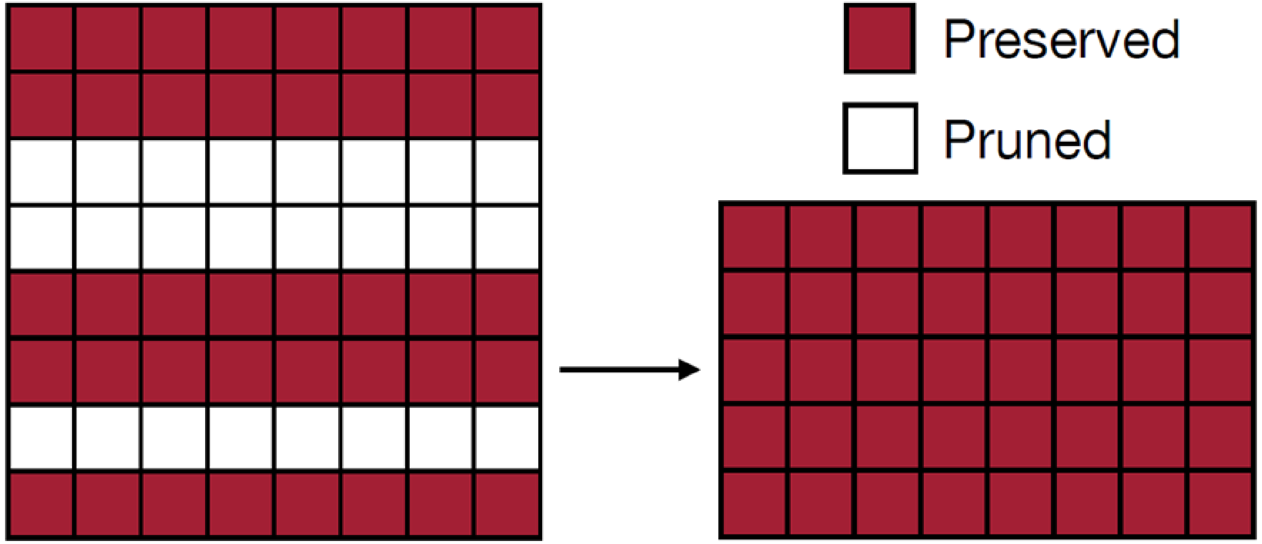
- ~~Introduction to Pruning~~
  - ~~What is pruning?~~
  - ~~How should we formulate pruning?~~
- Determine the Pruning Granularity
  - In what pattern should we prune the neural network?
- Determine the Pruning Criterion
  - What synapses/neurons should we prune?
- Determine the Pruning Ratio
  - What should the target sparsity be for each layer?
- Fine-tune/Train Pruned Neural Network
  - How should we improve the performance of pruned models?



Pruning at Different Granularities

- Pruning is performed at different granularities, from structured to non-structured.





Fine-grained/Unstructured

- More flexible pruning index choice
- Hard to accelerate
  (irregular data expression)

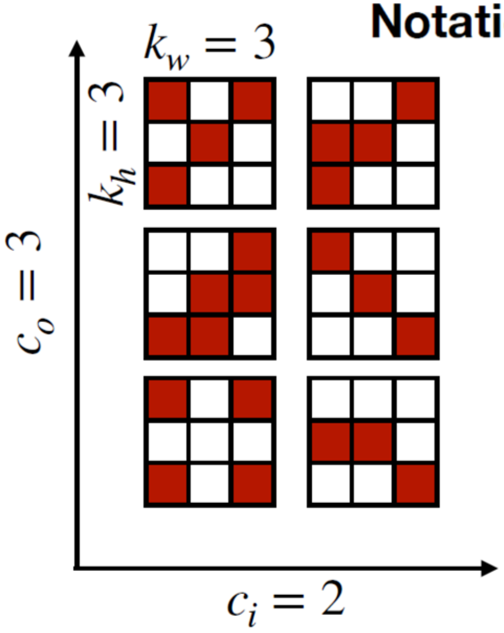
Coarse-grained/Structured

- Less flexible pruning index choice
  (a subset of the fine-grained case)
- Easy to accelerate (just a smaller matrix!)

Pruning at Different Granularities

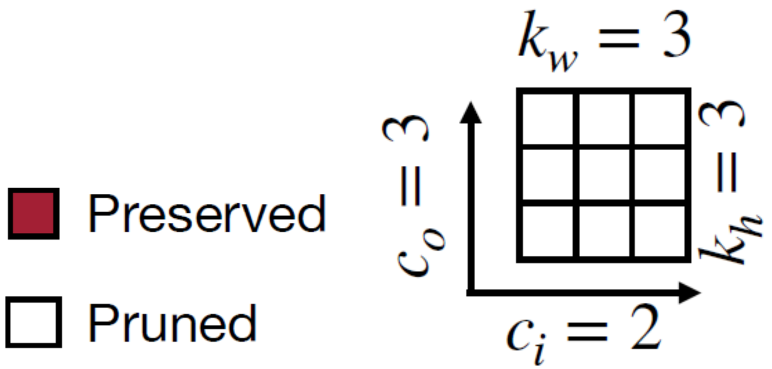
- Convolution layer pruning
  - The weights of convolutional layers have 4 dimensions:
    - c_i : input channels (or channels)
    - c_o : output channels (or filters)
    - k_h : kernel size height
    - k_w : kernel size width
  - The 4 dimensions give us more choices to select pruning granularities.

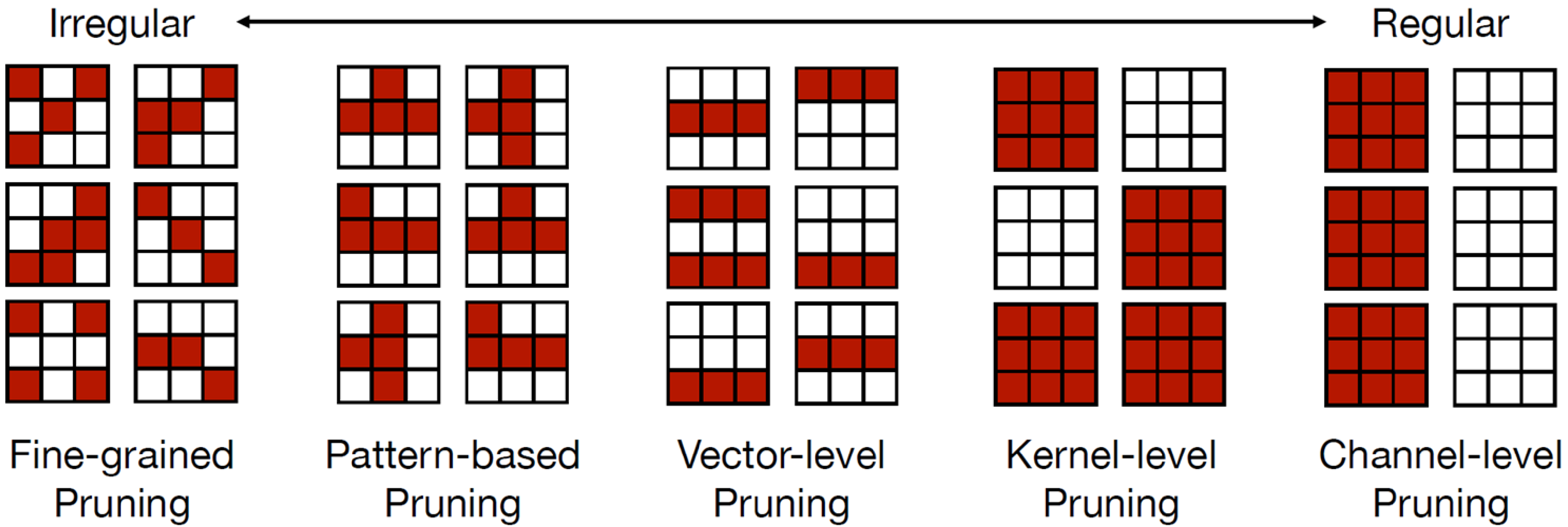
Notations



Pruning at Different Granularities

- Convolution layer pruning



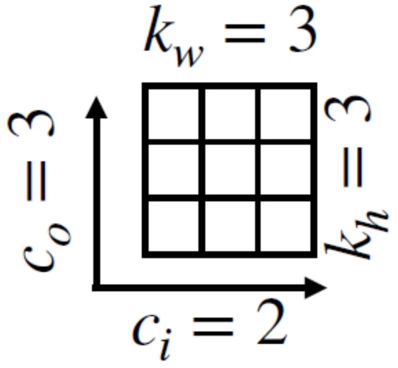


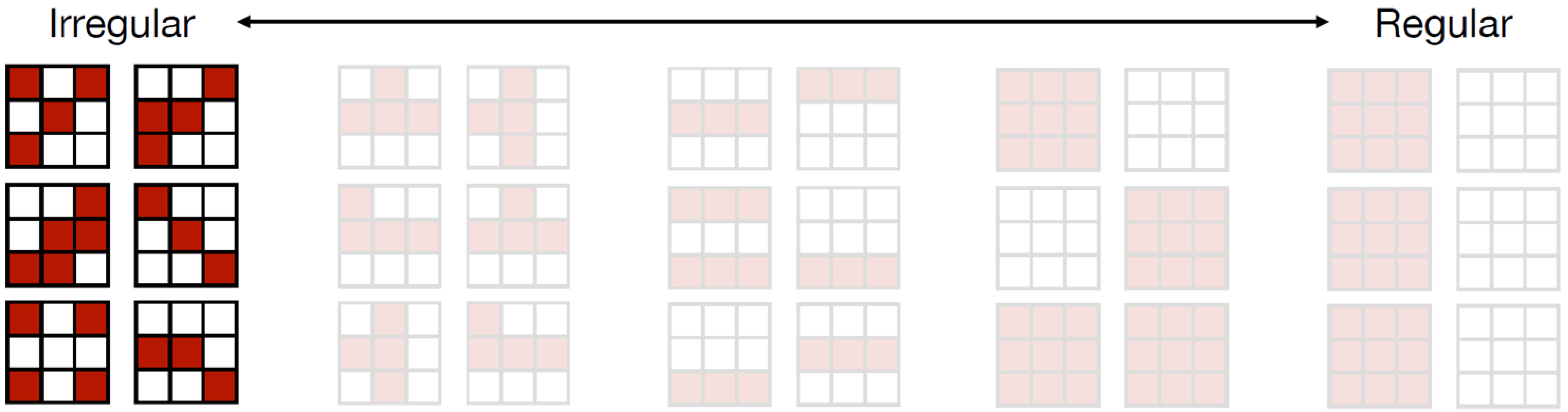
Pruning at Different Granularities

- Convolution layer pruning

🟥 Preserved

□ Pruned





Fine-grained Pruning

Pattern-based Pruning

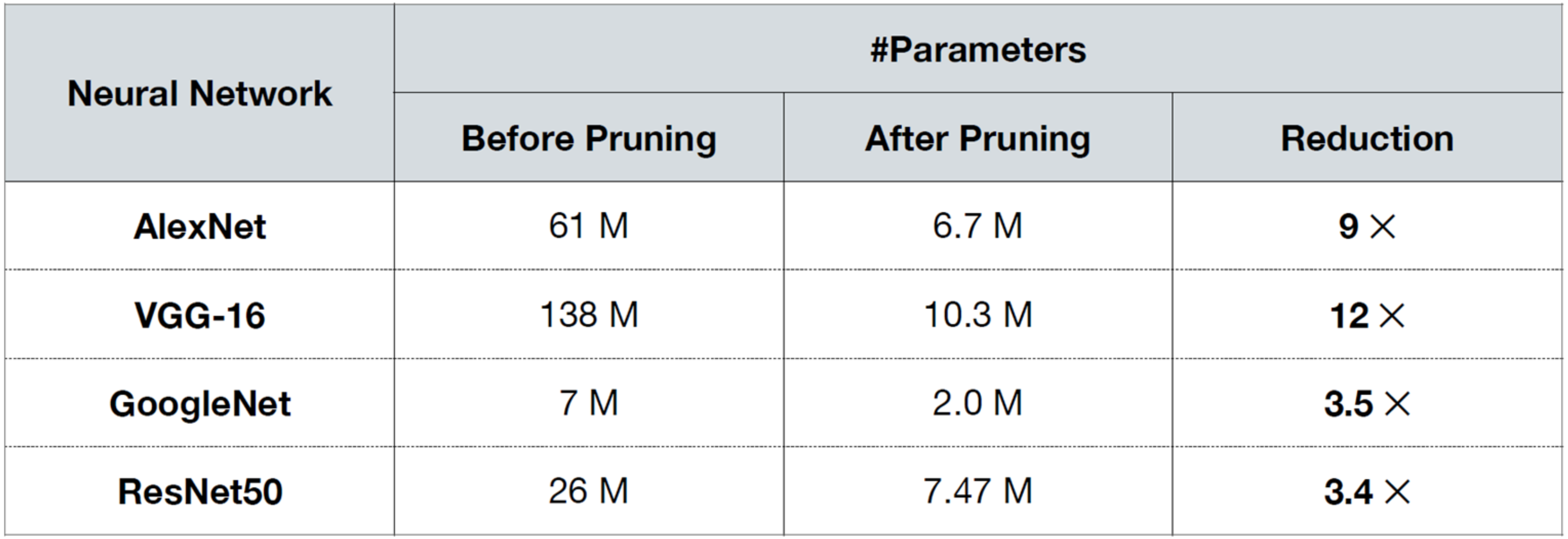
Vector-level Pruning

Kernel-level Pruning

Channel-level Pruning

Fine-grained Pruning

- Flexible pruning indices
- Usually large compression ratio since we can flexibly find “redundant” weights
- Can deliver speed up on some custom hardware (e.g., EIE) but not GPU



Neural Network	#Parameters		
	Before Pruning	After Pruning	Reduction
AlexNet	61 M	6.7 M	**9** ×
VGG-16	138 M	10.3 M	**12** ×
GoogleNet	7 M	2.0 M	**3.5** ×
ResNet50	26 M	7.47 M	**3.4** ×

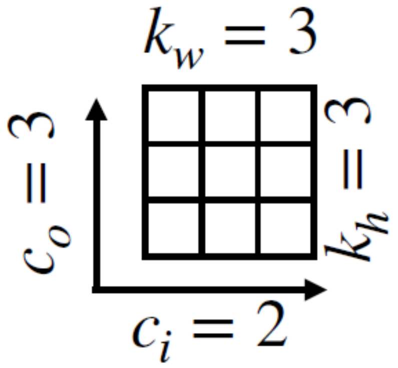
Han et al., “EIE: efficient inference engine on compressed deep neural network”, ISCA 2016

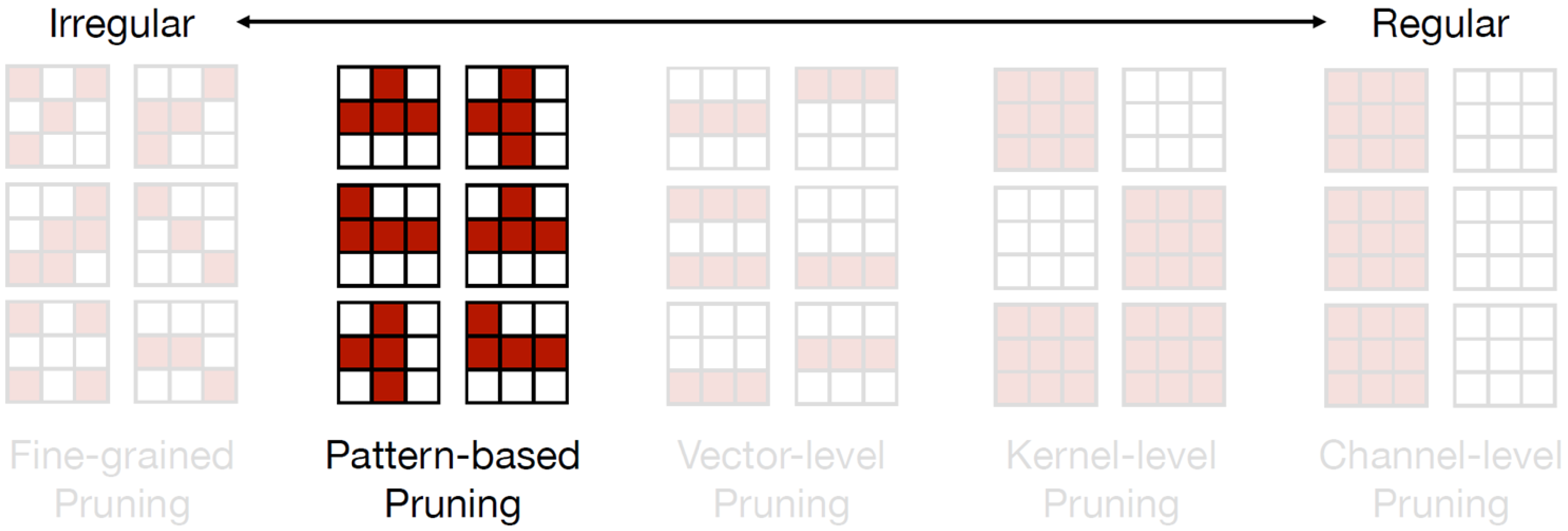
Pruning at Different Granularities

- Convolution layer pruning

🟥 Preserved

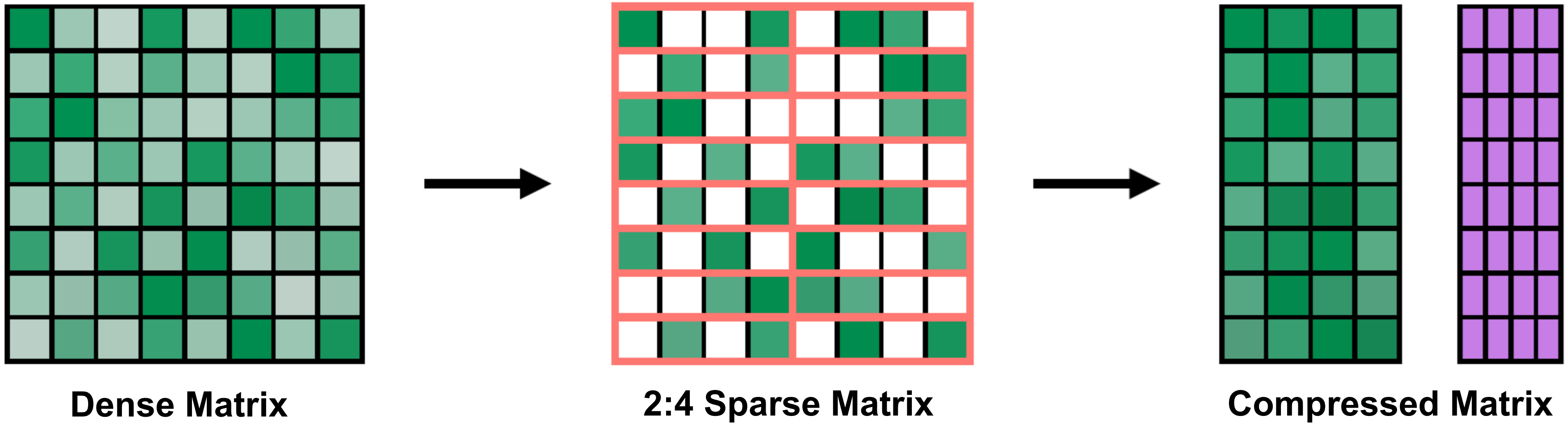
□ Pruned





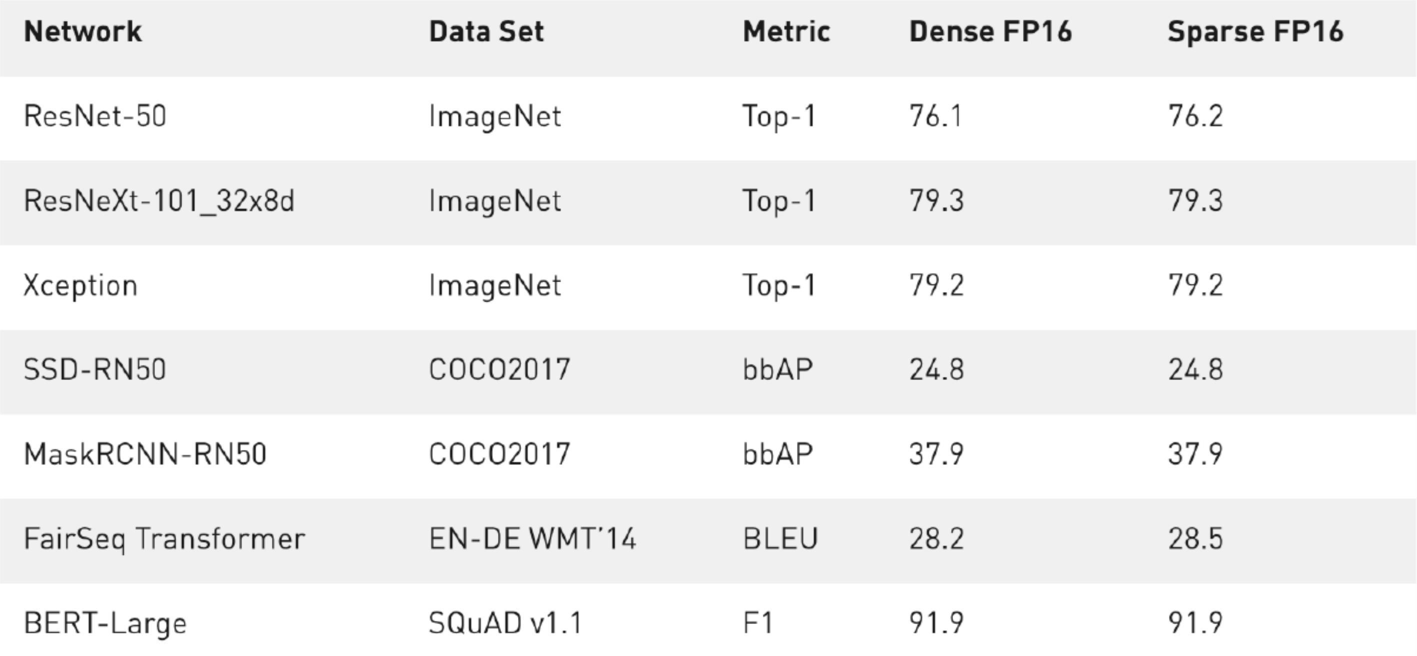
Pattern-based Pruning

- N:M sparsity
  - N:M sparsity means that in each contiguous M elements, N of them are pruned.
  - A classic case in 2:4 sparsity (50% sparsity).
  - It is supported by NVIDIA’s Ampere GPU Architecture, which delivers up to 2x speed up.
  - Usually maintains accuracy (tested on varieties of tasks)



Pattern-based Pruning

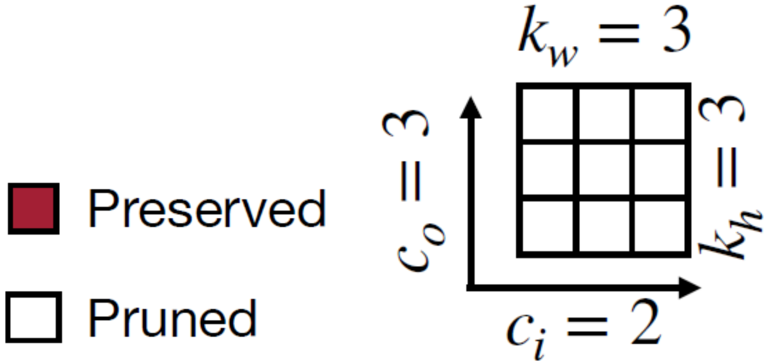
- N:M sparsity
  - N:M sparsity means that in each contiguous M elements, N of them are pruned.
  - A classic case in 2:4 sparsity (50% sparsity).
  - It is supported by NVIDIA's Ampere GPU Architecture, which delivers up to 2x speed up.
  - Usually maintains accuracy (tested on varieties of tasks)

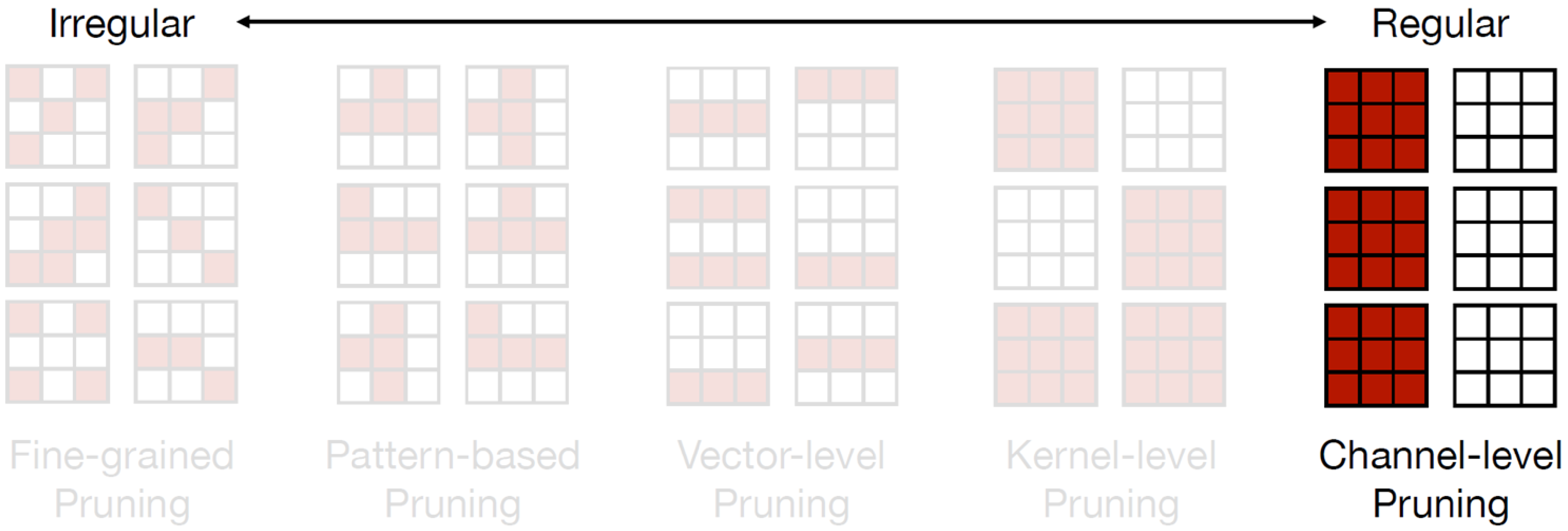


Network	Data Set	Metric	Dense FP16	Sparse FP16
ResNet-50	ImageNet	Top-1	76.1	76.2
ResNeXt-101_32x8d	ImageNet	Top-1	79.3	79.3
Xception	ImageNet	Top-1	79.2	79.2
SSD-RN50	COCO2017	bbAP	24.8	24.8
MaskRCNN-RN50	COCO2017	bbAP	37.9	37.9
FairSeq Transformer	EN-DE WMT'14	BLEU	28.2	28.5
BERT-Large	SQuAD v1.1	F1	91.9	91.9

Pruning at Different Granularities

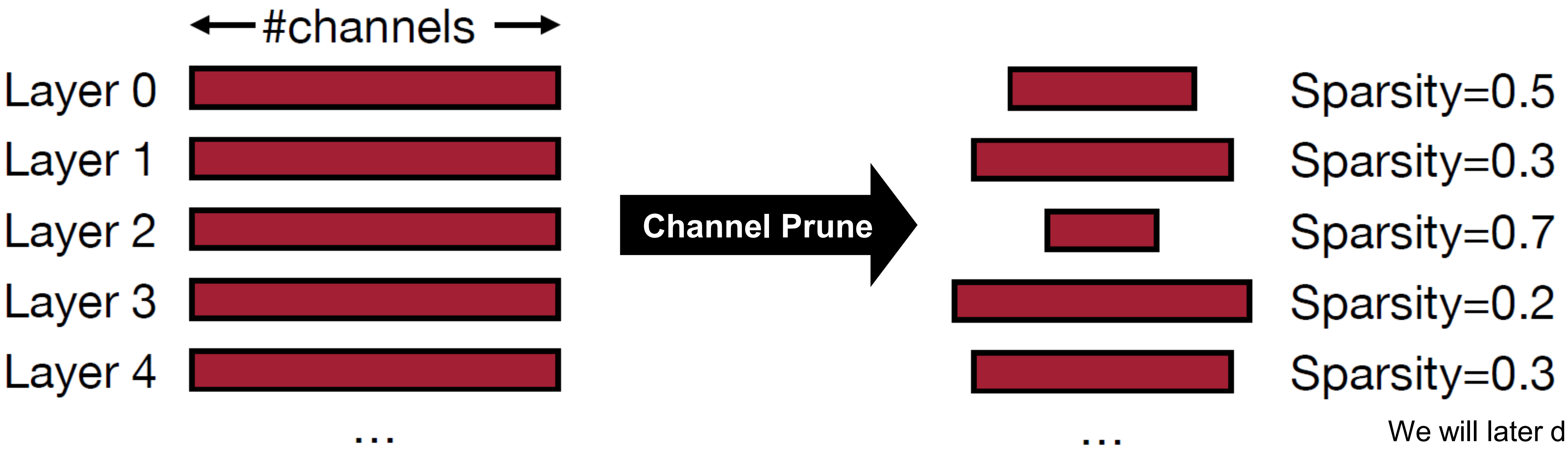
- Convolution layer pruning





Channel-level Pruning

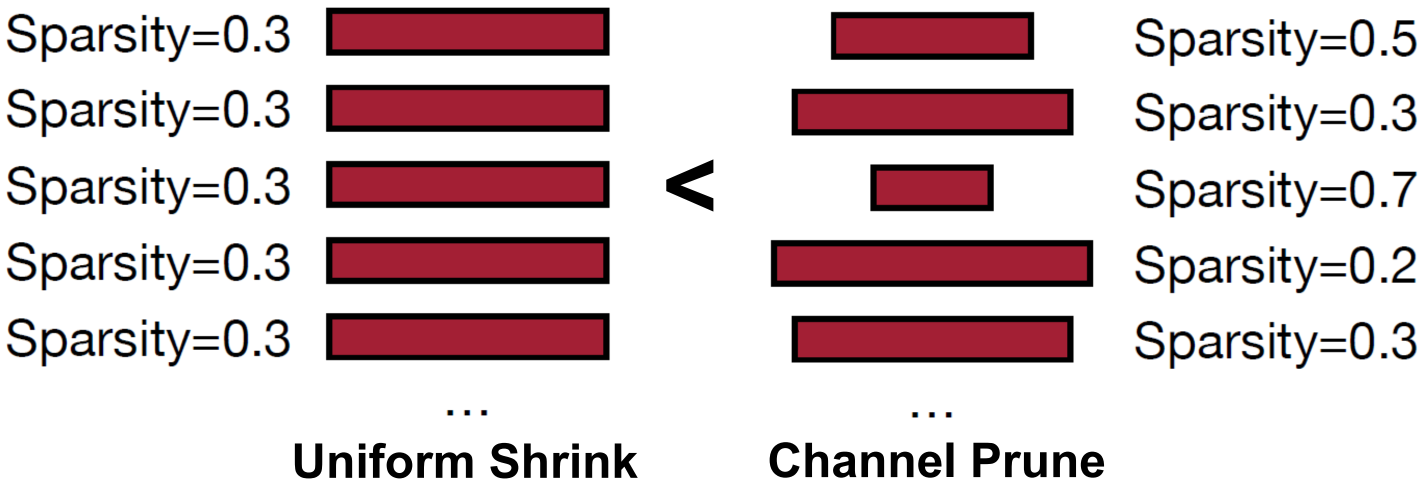
- Direct speed up due to reduced channel numbers (leading to an NN with smaller #channels)
- Smaller compression ratio than fine-grained pruning



We will later discuss how to determine sparsity ratios.

Channel-level Pruning

- Channel pruning has larger compression ratio than uniform shrink.



Sparsity=0.3

Sparsity=0.3

Sparsity=0.3

Sparsity=0.3

Sparsity=0.3

...

Uniform Shrink

<

Sparsity=0.5

Sparsity=0.3

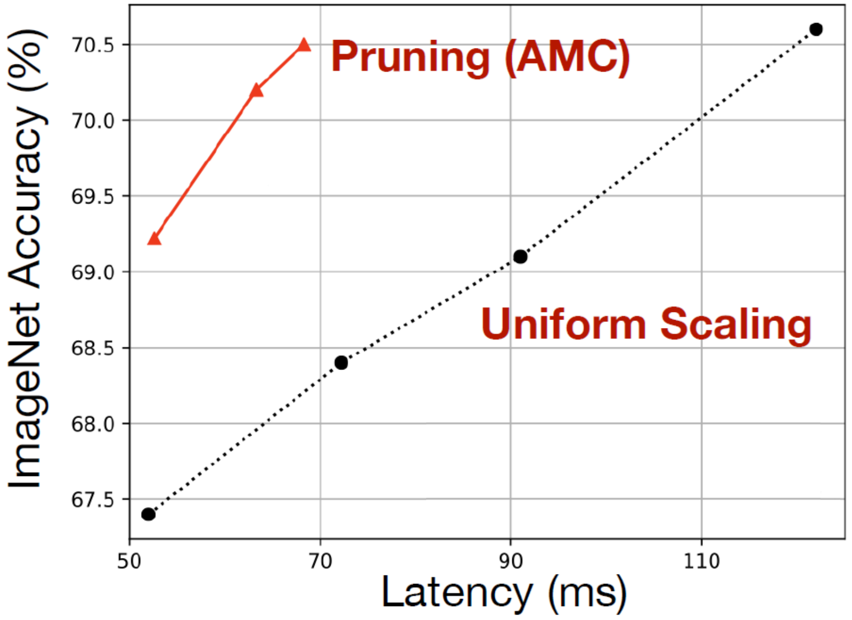
Sparsity=0.7

Sparsity=0.2

Sparsity=0.3

...

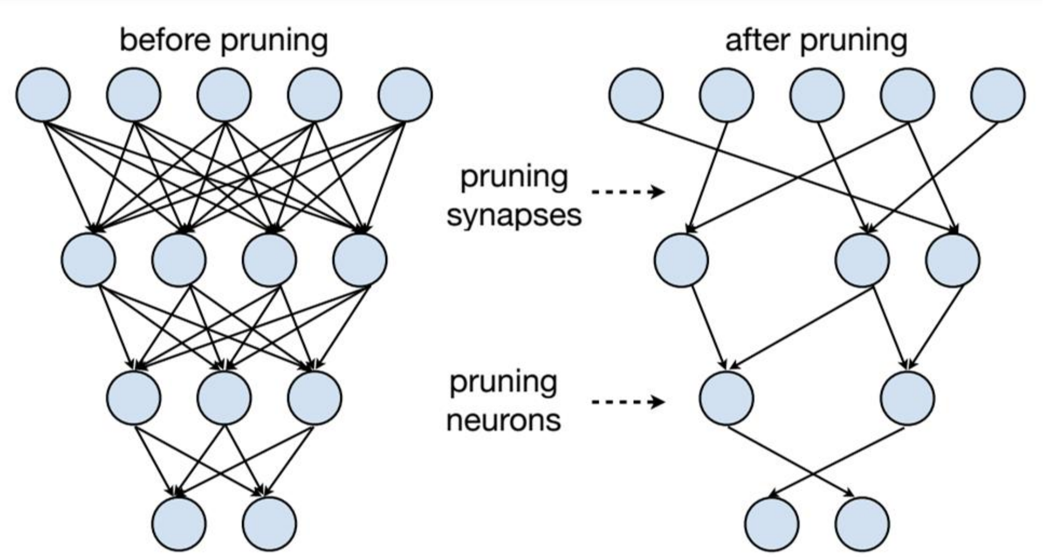
Channel Prune



He *et al* , “AMC: Automl for Model Compression and Acceleration on Mobile Devices”, ECCV 2018

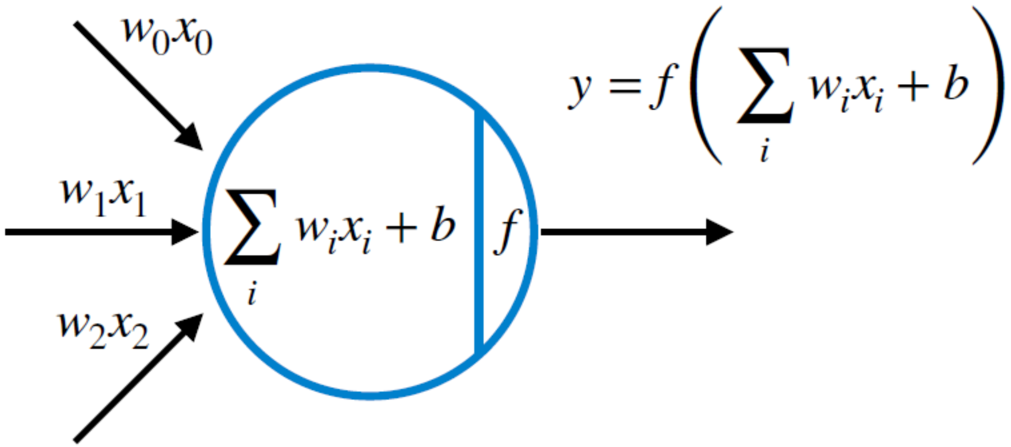
Neural Network Pruning

- ~~Introduction to Pruning~~
  - ~~What is pruning?~~
  - ~~How should we formulate pruning?~~
- ~~Determine the Pruning Granularity~~
  - ~~In what pattern should we prune the neural network?~~
- Determine the Pruning Criterion
  - What synapses/neurons should we prune?
- Determine the Pruning Ratio
  - What should the target sparsity be for each layer?
- Fine-tune/Train Pruned Neural Network
  - How should we improve the performance of pruned models?



Selection of Synapses to Prune

- When removing parameters from a neural network model,
  - The less important the parameters being removed,
    the better the performance of the pruned network.



Example

$f(\cdot) = \text{ReLU}(\cdot),\ \ W = [10, -8, 0.1]$

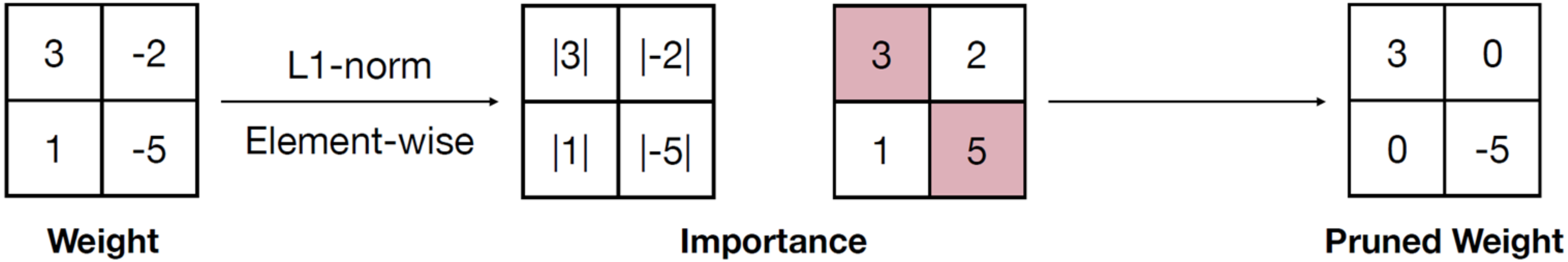
➡ $y = \text{ReLU}(10x_0 - 8x_1 + 0.1x_2)$

If only one weight will be removed,
Which one? Why?

Magnitude-based Pruning

- Magnitude-based pruning considers weights with larger absolute values are more important than other weights.
  - For element-wise pruning,

$$Importance = |W|$$



3	-2
1	-5

Weight

|3|	|-2|
|1|	|-5|

3	2
1	5

Importance

3	0
0	-5

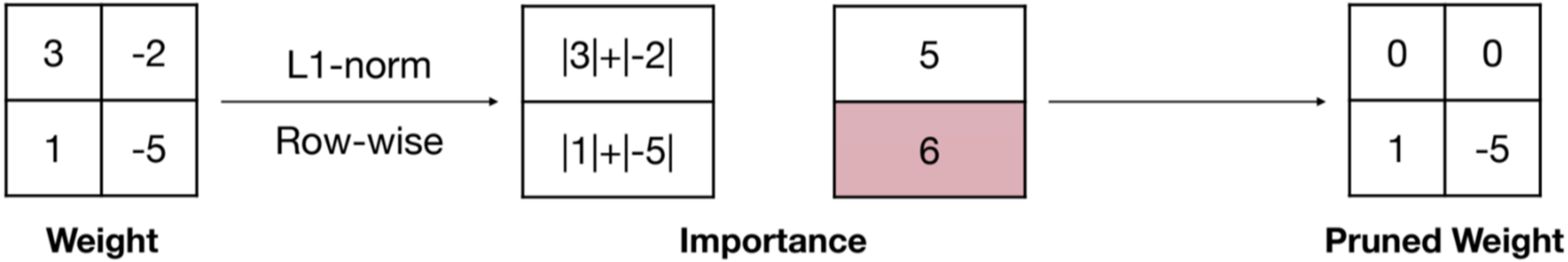
Pruned Weight

Han *et al* , "Learning Both Weights and Connections for Efficient Neural Network", NeurIPS 2015

Magnitude-based Pruning

- Magnitude-based pruning considers weights with larger absolute values are more important than other weights.
  - For row-wise pruning, the L1-norm magnitude can be defined as

$$Importance = \sum_{i \in S} |w_i|, \text{ where } \mathbf{W}^{(S)} \text{ is the structural set } S \text{ of parameters } \mathbf{W}$$



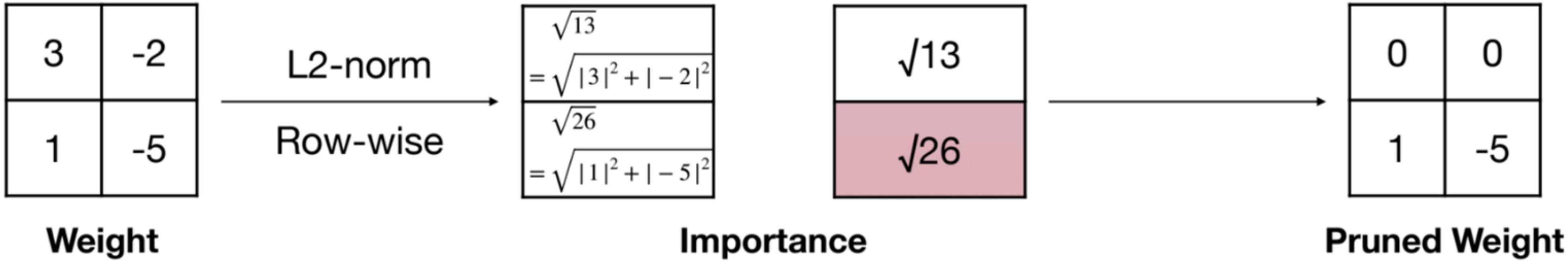
$$\begin{bmatrix} 3 & -2 \\ 1 & -5 \end{bmatrix} \xrightarrow[\text{Row-wise}]{\text{L1-norm}} \begin{bmatrix} |3|+|-2| \\ |1|+|-5| \end{bmatrix} \quad \begin{bmatrix} 5 \\ 6 \end{bmatrix} \longrightarrow \begin{bmatrix} 0 & 0 \\ 1 & -5 \end{bmatrix}$$

Han *et al* , "Learning Both Weights and Connections for Efficient Neural Network", NeurIPS 2015

Magnitude-based Pruning

- Magnitude-based pruning considers weights with larger absolute values are more important than other weights.
  - For row-wise pruning, the L2-norm magnitude can be defined as

$$Importance = \sqrt{\sum_{i \in S} |w_i|^2}, \text{ where } \mathbf{W}^{(S)} \text{ is the structural set } S \text{ of parameters } \mathbf{W}$$



3	-2
1	-5

Weight

$\xrightarrow[\text{Row-wise}]{\text{L2-norm}}$

$\sqrt{13}$ $= \sqrt{\|3\|^2 + \|-2\|^2}$
$\sqrt{26}$ $= \sqrt{\|1\|^2 + \|-5\|^2}$

√13
√26

Importance

0	0
1	-5

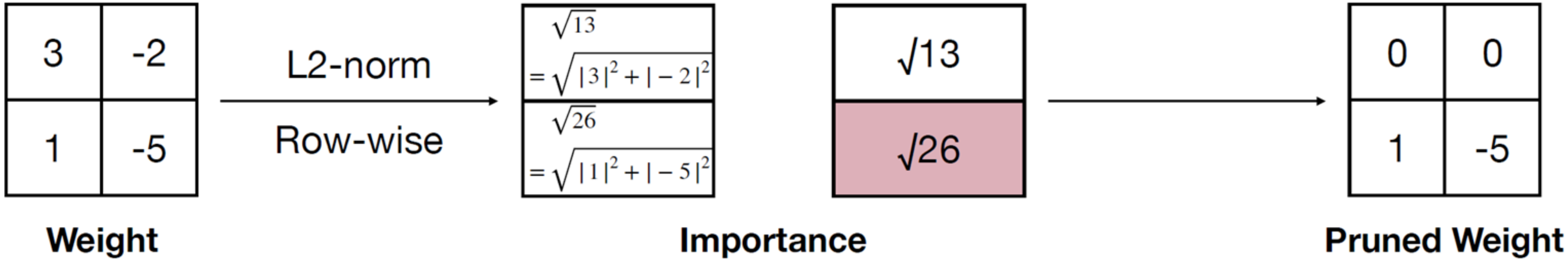
Pruned Weight

Han *et al* , "Learning Both Weights and Connections for Efficient Neural Network", NeurIPS 2015

Magnitude-based Pruning

- Magnitude-based pruning considers weights with larger absolute values are more important than other weights.
  - For row-wise pruning, the L_p-norm magnitude can be defined as

$$Importance = \|\mathbf{W}^{(S)}\|_p = \left(\sum_{i \in S} |w_i|^p\right)^{\frac{1}{p}}$$, where $\mathbf{W}^{(S)}$ is a structural set of parameters



Weight

3	-2
1	-5

$\xrightarrow[\text{Row-wise}]{\text{L2-norm}}$

Importance

$\sqrt{13} = \sqrt{\lvert 3 \rvert^2 + \lvert -2 \rvert^2}$
$\sqrt{26} = \sqrt{\lvert 1 \rvert^2 + \lvert -5 \rvert^2}$

√13
√26

$\longrightarrow$

Pruned Weight

0	0
1	-5

Han *et al* , "Learning Structured Sparsity in Deep Neural Networks", NeurIPS 2016

Taylor Expansion Analysis on Pruning Error

- Evaluate pruning error induced by pruned synapses
- The task of training NN is to minimize the objective function $L(\mathbf{x}; \mathbf{W})$.
  - The importance of a parameter can be quantified by the error on loss function induced by removing it.
- The induced error can be approximated by a Taylor series.

$$\delta L = L(\mathbf{x}; \mathbf{W}) - L(\mathbf{x}; \mathbf{W}_P = \mathbf{W} - \delta \mathbf{W}) = \sum_i g_i \delta w_i + \frac{1}{2} \sum_i h_{ii} \delta w_i^2 + \frac{1}{2} \sum_{i \neq j} h_{ij} \delta w_i \delta w_j + O(\|\delta \mathbf{W}\|^3)$$

where $g_i = \frac{\partial L}{\partial w_i}, h_{i,j} = \frac{\partial^2 L}{\partial w_i \partial w_j}$

Liu *et al* , "Learning Efficient Convolutional Networks Through Network Slimming", ICCV 2017

Second-Order-based Pruning

- Prunes by minimization of the second-order loss change approximation

$$\delta L = L(\mathbf{x};\mathbf{W}) - L(\mathbf{x};\mathbf{W}_P = \mathbf{W} - \delta\mathbf{W}) = \sum_i \xcancel{g_i \delta w_i} + \frac{1}{2}\sum_i h_{ii}\delta w_i^2 + \frac{1}{2}\sum_{i \neq j} h_{ij}\xcancel{\delta w_i}\delta w_j + O(\xcancel{\|\delta}\mathbf{W}\|^3)$$

where $g_i = \dfrac{\partial L}{\partial w_i}, h_{i,j} = \dfrac{\partial^2 L}{\partial w_i \partial w_j}$

LeCun *et al* , “Optimal Brain Damage”, NeurIPS 1989

Second-Order-based Pruning

- Optimal Brain Damage assumes that
  - The objective function L is nearly quadratic: the last term is neglected
  - The neural network training has converged: first-order terms are neglected
  - The error caused by deleting each parameter is independent: cross terms are neglected

$$\delta L_i = L(\mathbf{x}; \mathbf{W}) - L(\mathbf{x}; \mathbf{W}_P \,|\, w_i = 0) \approx \frac{1}{2} h_{ii} w_i^2$$

  - The synapses with smaller induced error $|\delta L_i|$ will be removed; that is to say,

$$importance_{w_i} = |\, \delta L_i \,| = \frac{1}{2} h_{ii} w_i^2$$

* h_{ii} is non-negative

LeCun *et al* , "Optimal Brain Damage", NeurIPS 1989

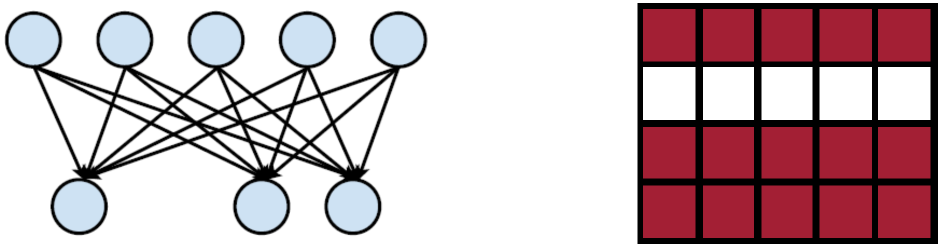
Selection of Neurons to Prune

- When removing neurons from a neural network model, the less useful the neurons being removed the better the performance of the pruned network.

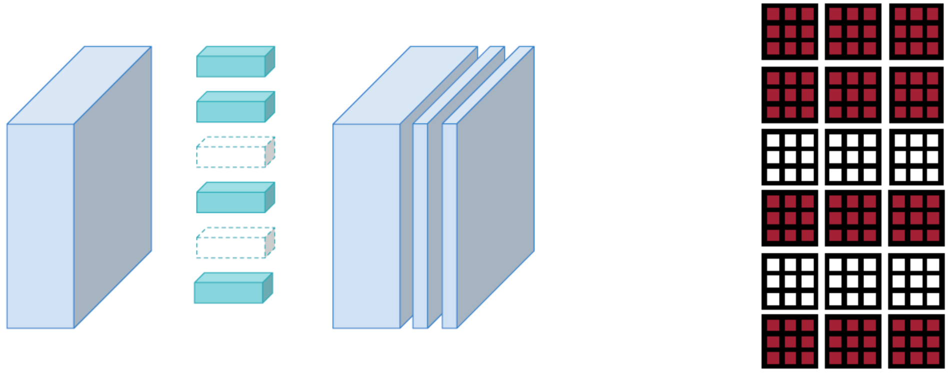
Neuron pruning is coarse-grained pruning.

Weight Matrix

Neuron pruning in Linear Layer

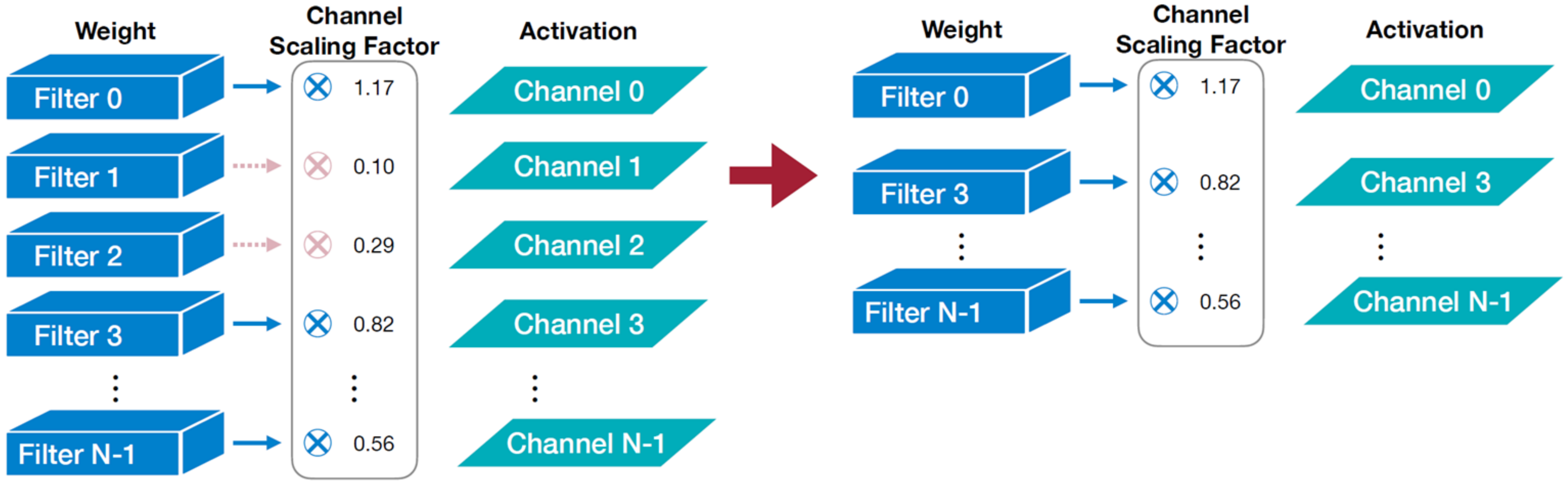


Channel pruning in Convolution Layer



Scaling-based Pruning

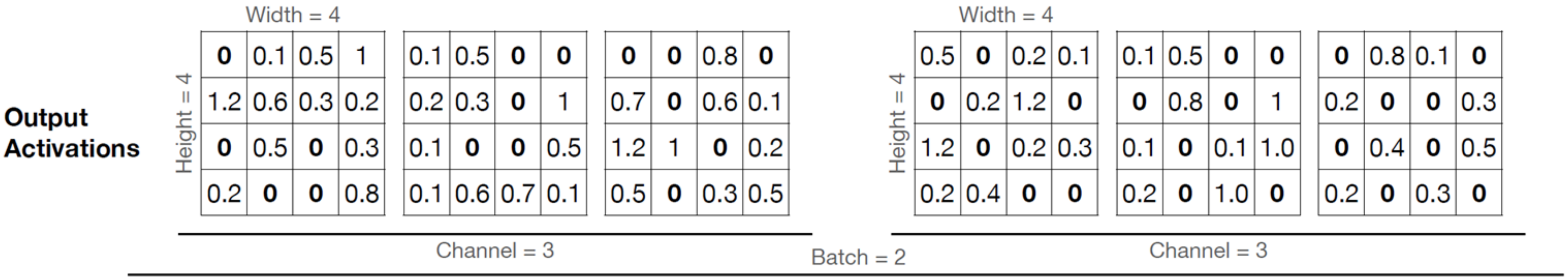
- A scaling factor is associated with each filter (i.e. output channel) in conv layers.
  - The scaling factor is multiplied to the output of that channel
  - The scaling factors are trainable parameters
- The filters/output channels with small scaling factor magnitude will be pruned.



Liu *et al* , "Learning Efficient Convolutional Networks Through Network Slimming", ICCV 2017

Percentage-of-Zero-Based Pruning

- ReLU activation will generate zeros in the output activation.
- Similar to magnitude of weights,
  the Average Percentage of Zero activations measures importance of the neurons.



Output Activations

0	0.1	0.5	1
1.2	0.6	0.3	0.2
0	0.5	**0**	0.3
0.2	**0**	**0**	0.8

0.1	0.5	**0**	**0**
0.2	0.3	**0**	1
0.1	**0**	**0**	0.5
0.1	0.6	0.7	0.1

0	**0**	0.8	**0**
0.7	**0**	0.6	0.1
1.2	1	**0**	0.2
0.5	**0**	0.3	0.5

0.5	**0**	0.2	0.1
0	0.2	1.2	**0**
1.2	**0**	0.2	0.3
0.2	0.4	**0**	**0**

0.1	0.5	**0**	**0**
0	0.8	**0**	1
0.1	**0**	0.1	1.0
0.2	**0**	1.0	**0**

0	0.8	0.1	**0**
0.2	**0**	**0**	0.3
0	0.4	**0**	0.5
0.2	**0**	0.3	**0**

Average Percentage of Zeros (APoZ)

$$= \frac{5+6}{2\cdot4\cdot4} = \frac{11}{32}$$

Channel 0

$$= \frac{5+7}{2\cdot4\cdot4} = \frac{12}{32}$$

Channel 1

$$= \xcancel{\frac{6+8}{2\cdot4\cdot4}} = \frac{14}{32}$$

Channel 2

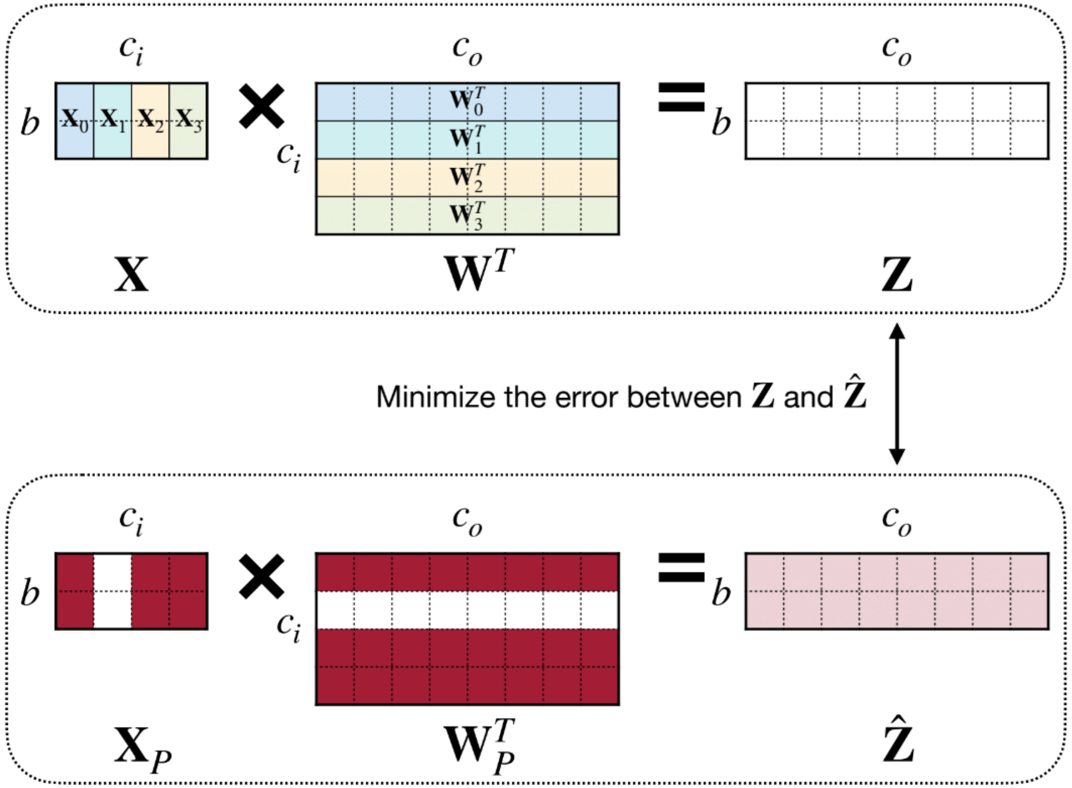
Regression-based Pruning

- Instead of considering the pruning error of the objective function $L(\mathbf{x}; \mathbf{W})$, regression-based pruning minimizes the reconstruction error of the corresponding layer's outputs.

$$\mathbf{Z} = \mathbf{X}\mathbf{W}^T = \sum_{c=0}^{c_i-1} \mathbf{X}_c \mathbf{W}_c^T$$

$$\underset{\mathbf{W},\,\beta}{\arg\min} \|\mathbf{Z} - \hat{\mathbf{Z}}\|_F^2 = \|\mathbf{Z} - \sum_{c=0}^{c_i-1} \beta_c \mathbf{X}_c \mathbf{W}_c^T\|_F^2$$

$$\|\beta\|_0 \leq N_c$$



Minimize the error between $\mathbf{Z}$ and $\hat{\mathbf{Z}}$

- β is coefficient vector for channel selection. $\beta_c = 0$ means channel c is pruned.
- N_c is the number of nonzero channels

- Fix W, solve β for channel selection
- Fix β, solve W to minimize reconstruction error

Summary

- In this lecture, we learned:
  - What is pruning
  - Granularities of pruning
  - Criteria to select weights to prune

- In the next lecture, we will cover:
  - How to find pruning ratio for each layer
  - How to train/fine-tune the pruned layer
  - Lottery ticket hypothesis