Foundations of Edge AI

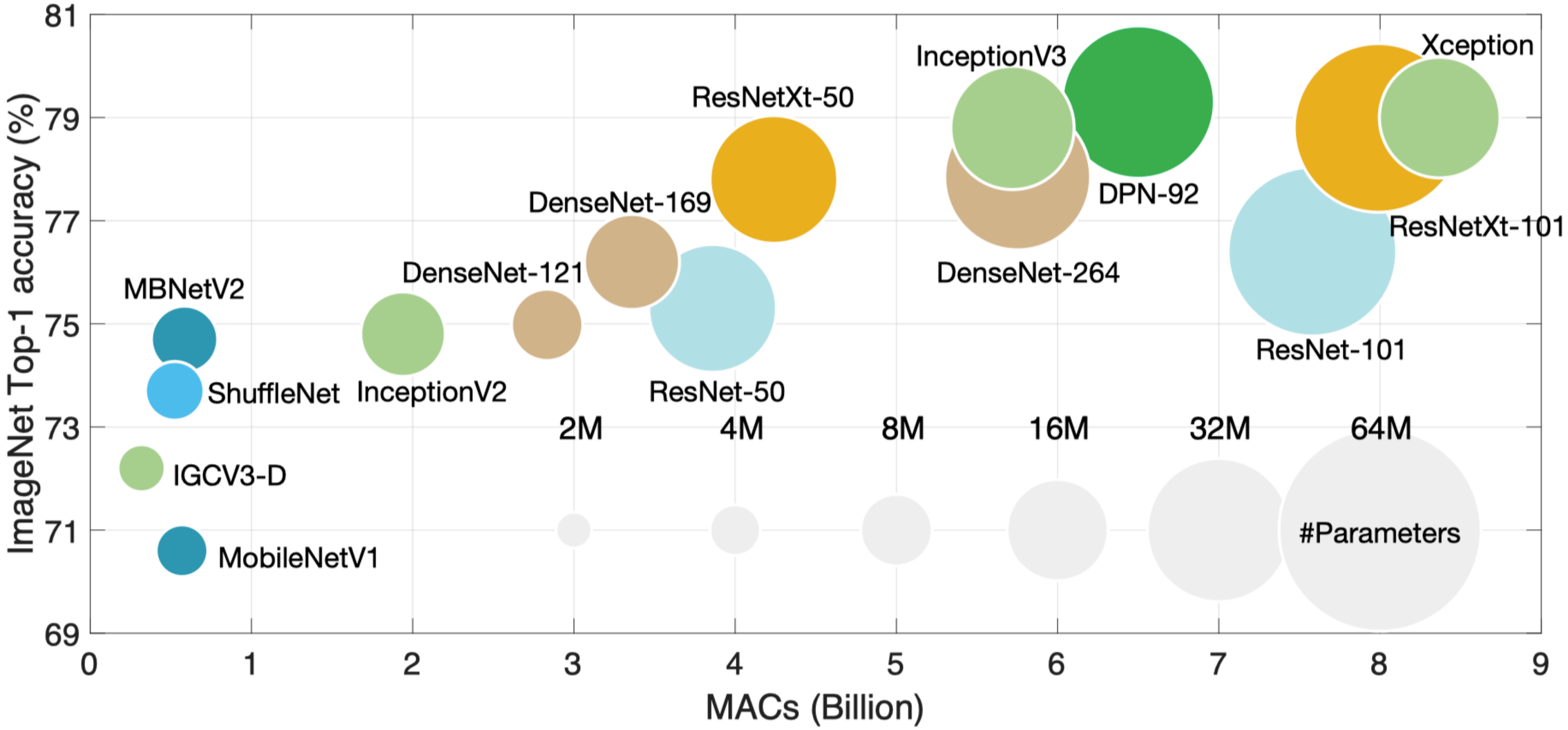
Lecture07

Neural Network Pruning

Lanyu (Lori) Xu

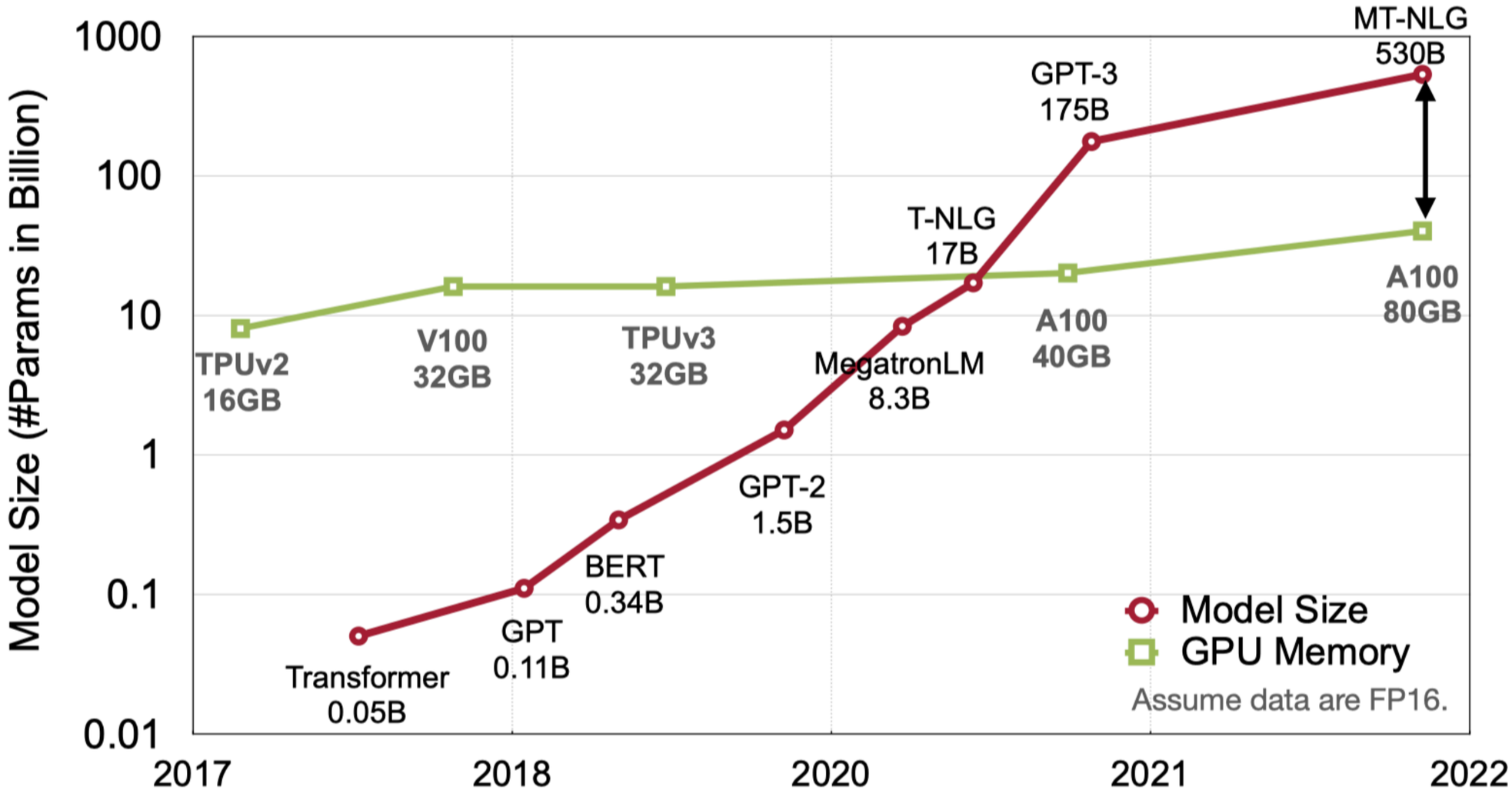
Email: lxu@oakland.edu
Homepage: https://lori930.github.io/
Office: EC 524

High Accuracy with High Computation Cost



Cai, H., Gan, C., Wang, T., Zhang, Z., & Han, S. (2019). Once-for-all: Train one network and specialize it for efficient deployment. arXiv preprint arXiv:1908.09791.

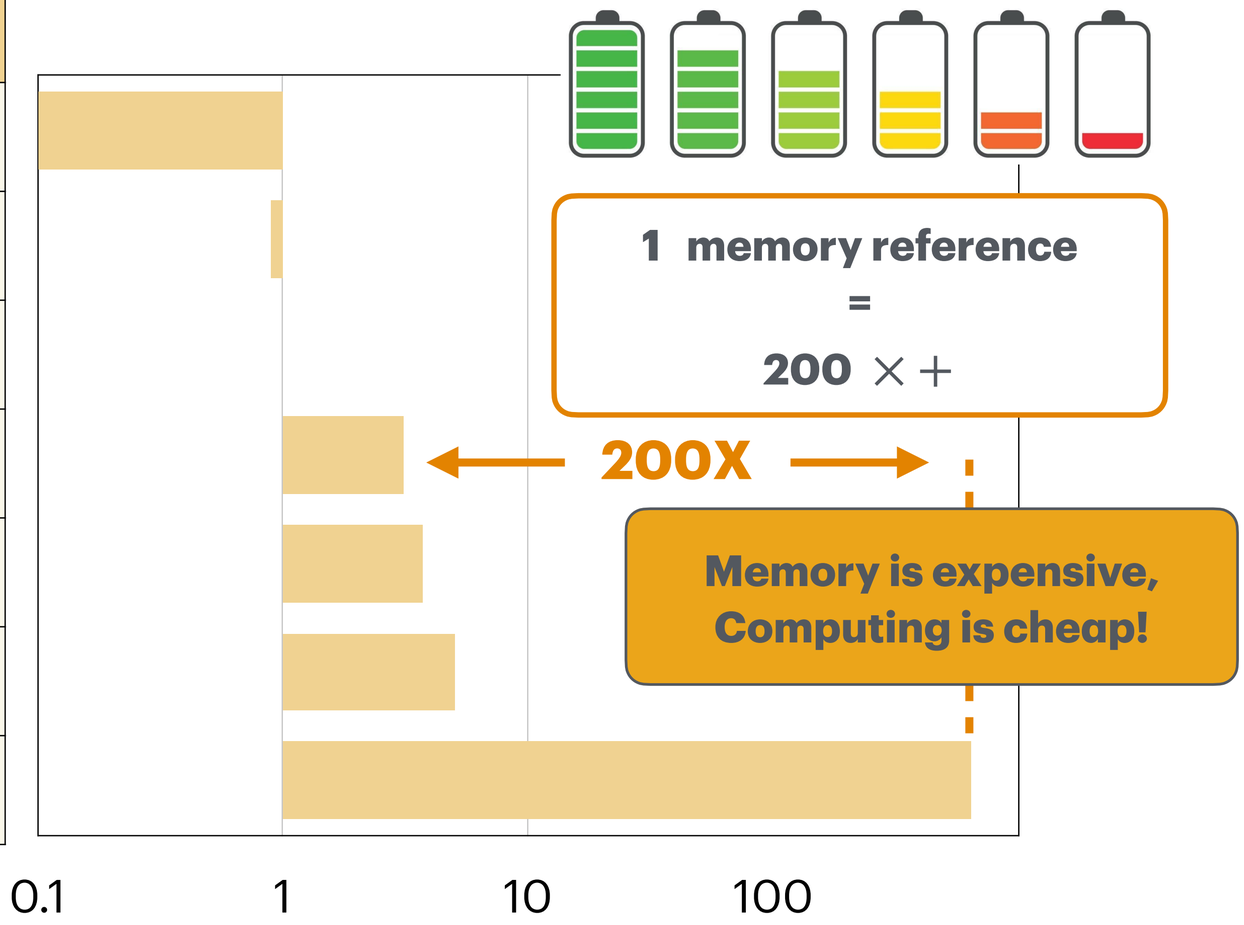
Today's AI is too BIG!



Memory is Expensive

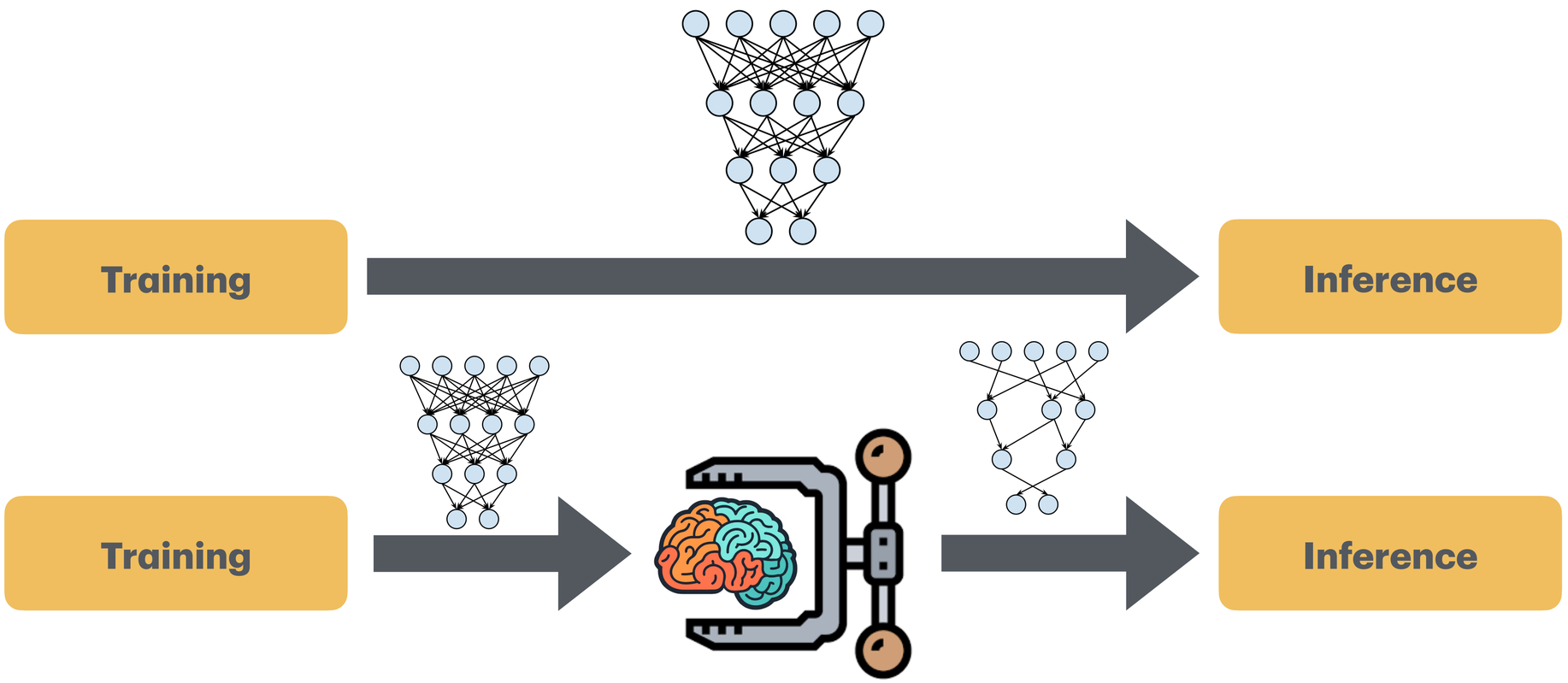
Data movement → More memory reference → More energy

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



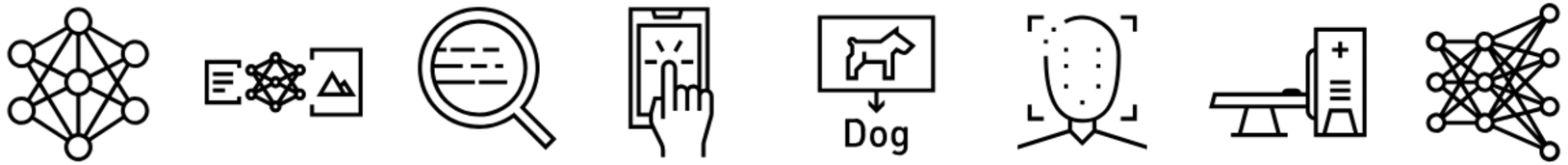
Horowitz, M. (2014, February). 1.1 computing's energy problem (and what we can do about it). In 2014 IEEE international solid-state circuits conference digest of technical papers (ISSCC) (pp. 10-14). IEEE.

Bridge the Gap



Model compression
(pruning, sparsity,
quantization, etc.)

MLPerf



The Olympic Game for AI Computing

- Provide unbiased evaluations of training and inference performance for hardware, software, and services. **Open division** and **closed division**.

- **MLPerf Inference** measures inference performance on nine different benchmarks, including several *large language models (LLMs), text-to-image, natural language processing, recommenders, computer vision,* and *medical image segmentation*.

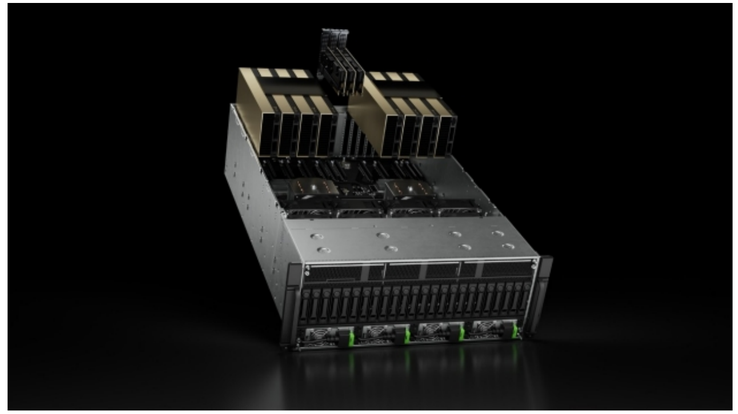
- **MLPerf Training** measures training performance on nine different benchmarks, including *LLM pre-training, LLM fine-tuning, text-to-image, graph neural network (GNN), computer vision, medical image segmentation,* and *recommendation*.

- **MLPerf HPC** measures training performance across four different scientific computing use cases, including *climate atmospheric river identification, cosmology parameter prediction, quantum molecular modeling,* and *protein structure prediction*.

MLPerf

The Olympic Game for AI Computing

LLAMA 2

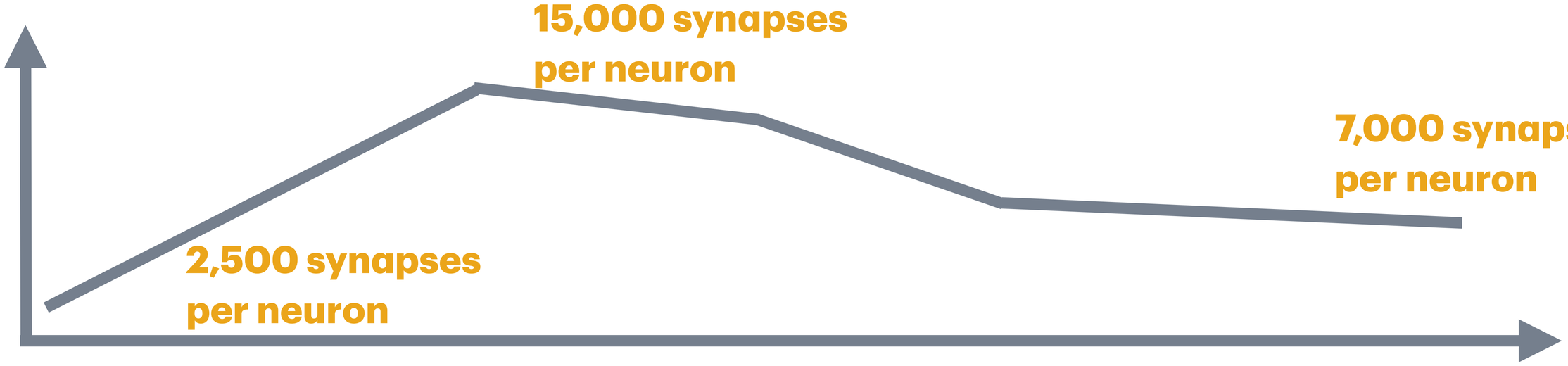


NVIDIA H200

- Llama 2 70B performance on a single NVIDIA H200 GPU (Aug 28, 2024)
- Almost 3x higher throughput (4,488 token/s → 11,189 token/s)
  - **Depth pruning**: 80 layers → 32 layers
  - **Width pruning**: 28,762 intermediate dimensions → 14,336 intermediate dimensions
  - **Fine-tuning** to maintain almost 99% accuracy

NVIDIA Blackwell Platform Sets New LLM Inference Records in MLPerf Inference v4.1

Pruning Happens in Human Brain



Peter Huttenlocher, pediatric neurologist, 1931-2013

Drachman, D. A. (2005). Do we have brain to spare?. Neurology, 64(12), 2004-2005.

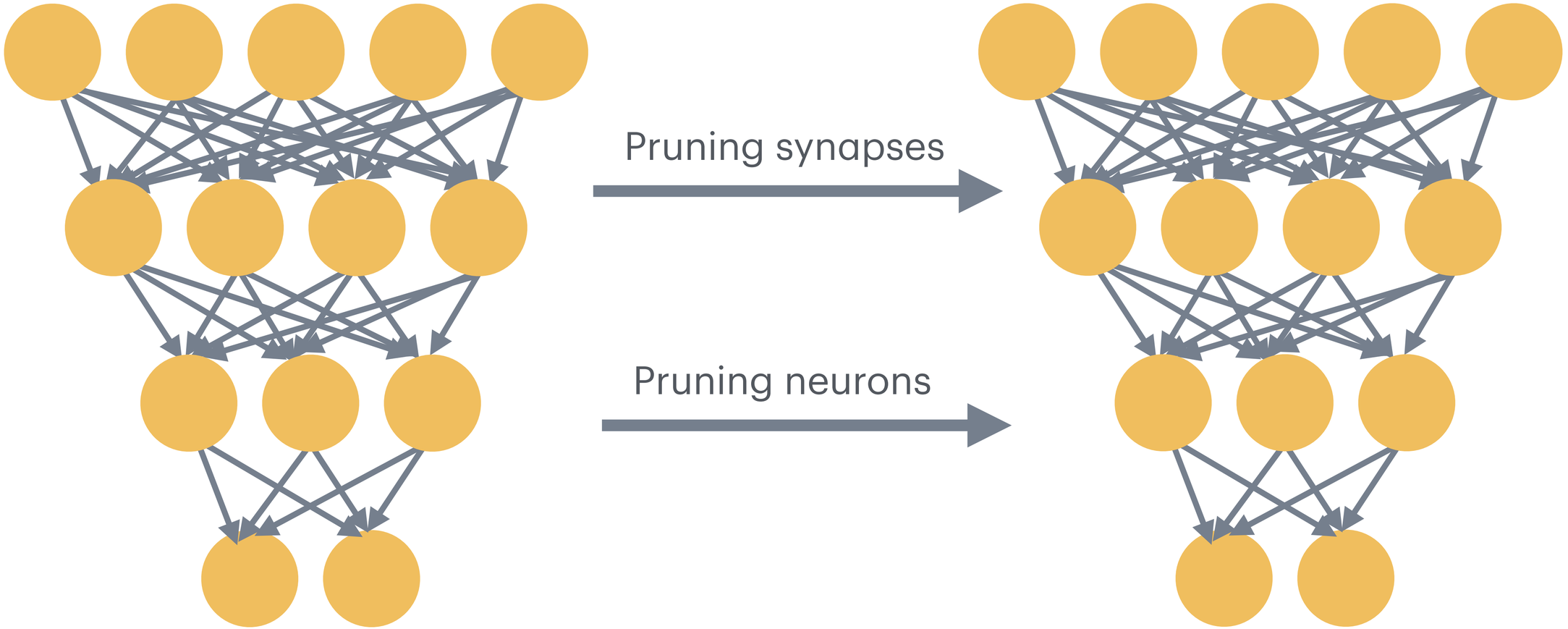
Lecture Plan

Today we will:

- Introduce pruning
  - What is pruning?
  - Granularities of pruning
  - Criteria to select synapses to prune
  - Criteria to select neurons to prune
- Show a Pruning demo

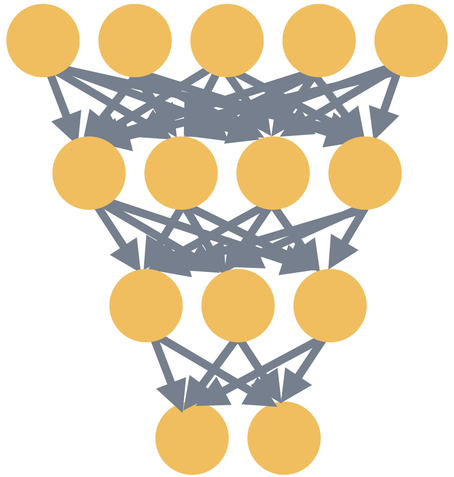
Neural Network Pruning

Make neural network smaller by removing synapses and neurons

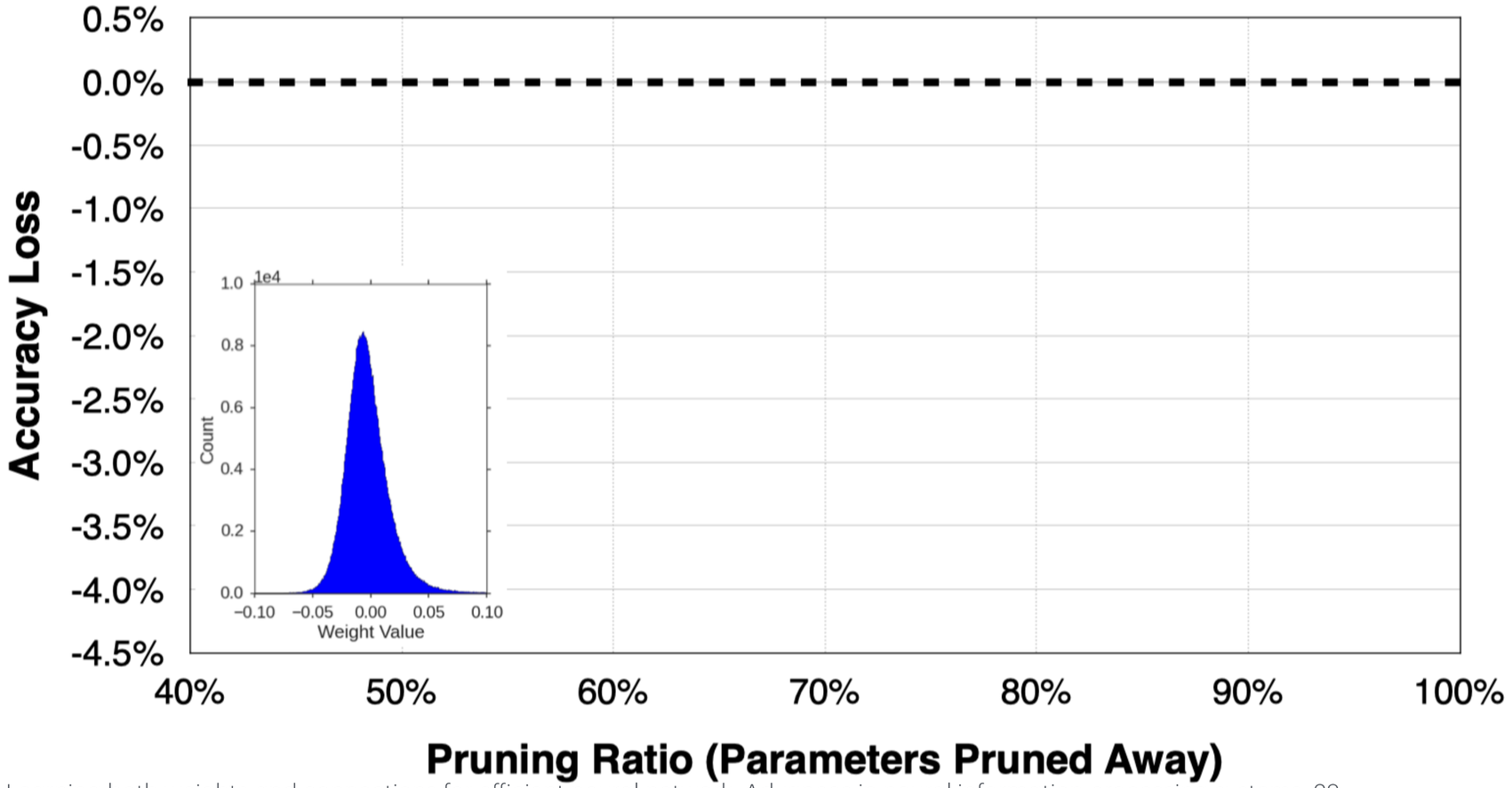


Neural Network Pruning

Make neural network smaller by removing synapses and neurons



Train Connectivity

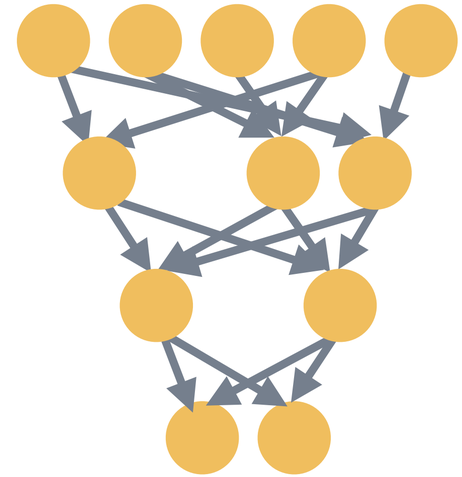


Han, S., Pool, J., Tran, J., & Dally, W. (2015). Learning both weights and connections for efficient neural network. Advances in neural information processing systems, 28.

Figure source

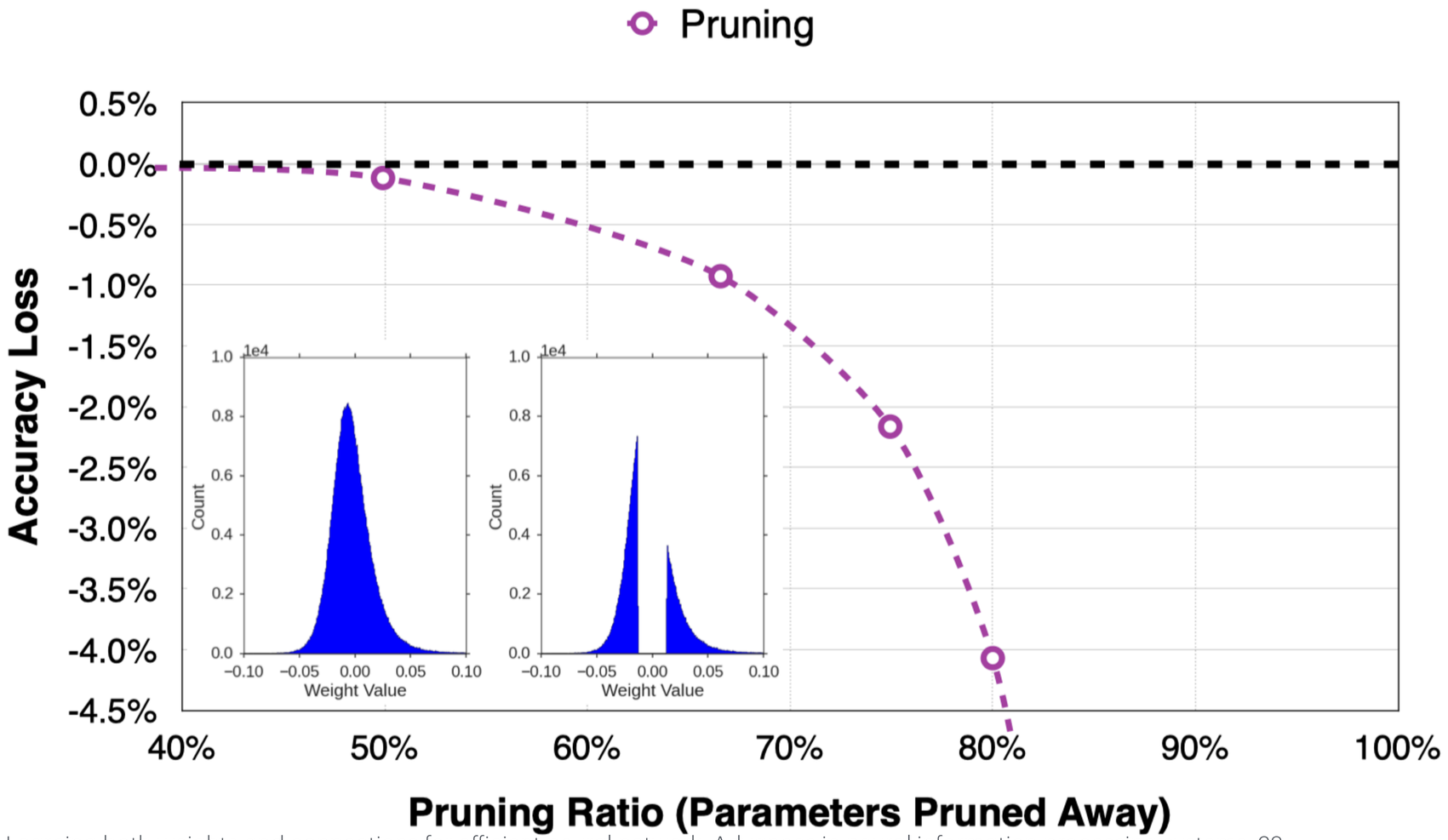
Neural Network Pruning

Make neural network smaller by removing synapses and neurons



Train Connectivity

Prune Connections

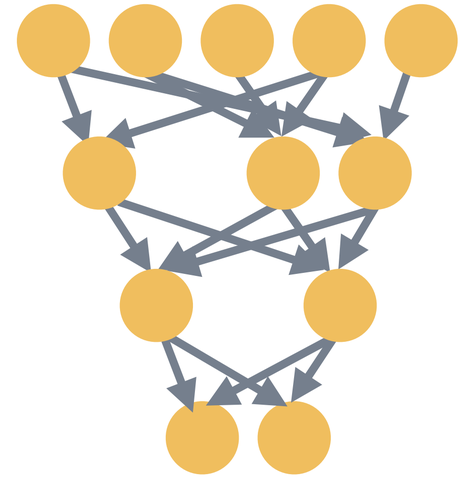


Han, S., Pool, J., Tran, J., & Dally, W. (2015). Learning both weights and connections for efficient neural network. Advances in neural information processing systems, 28.

Figure source

Neural Network Pruning

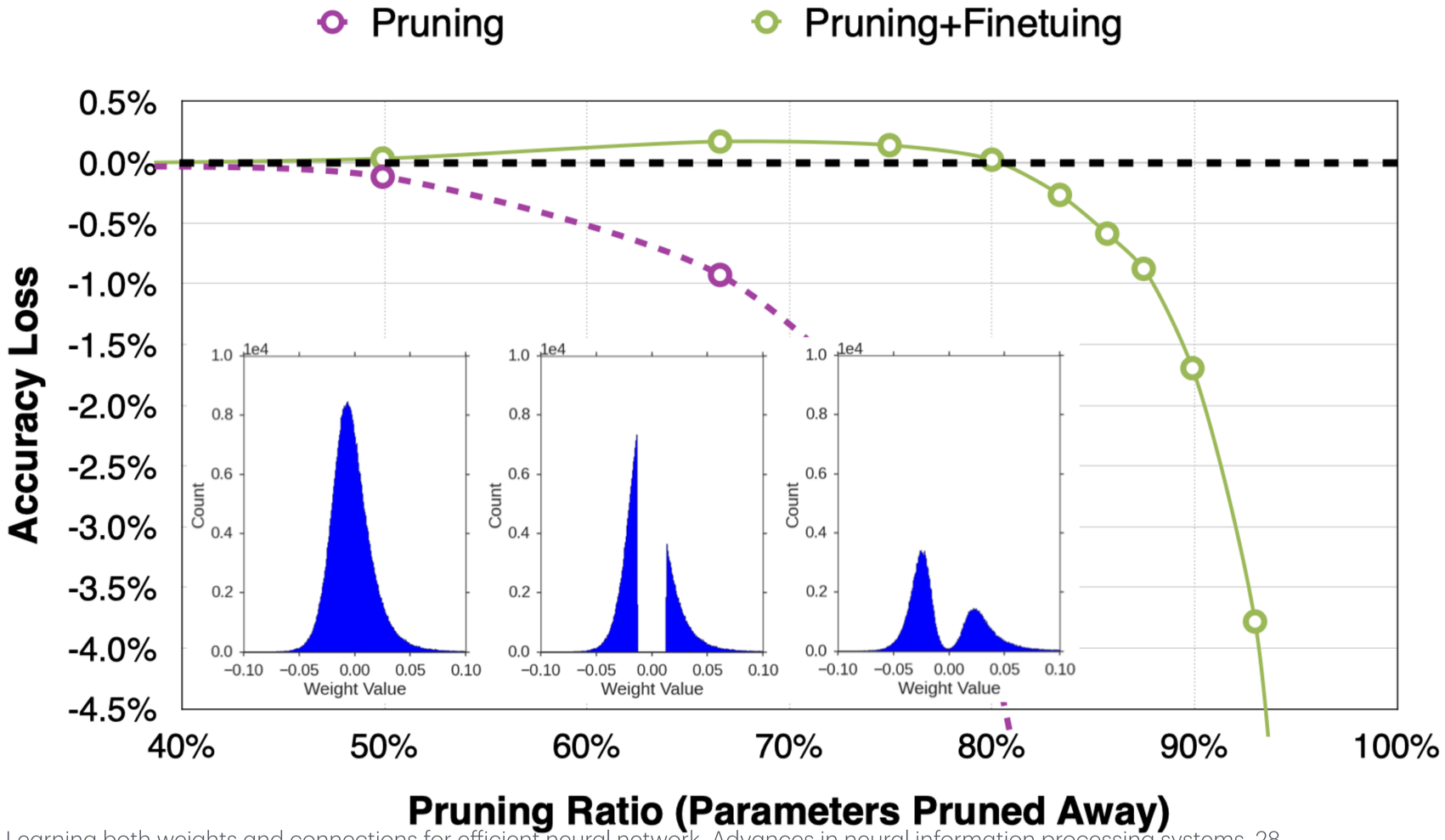
Make neural network smaller by removing synapses and neurons



Train Connectivity

Prune Connections

Train Weights

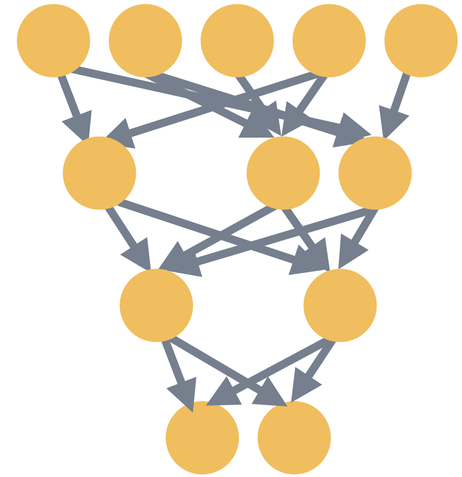


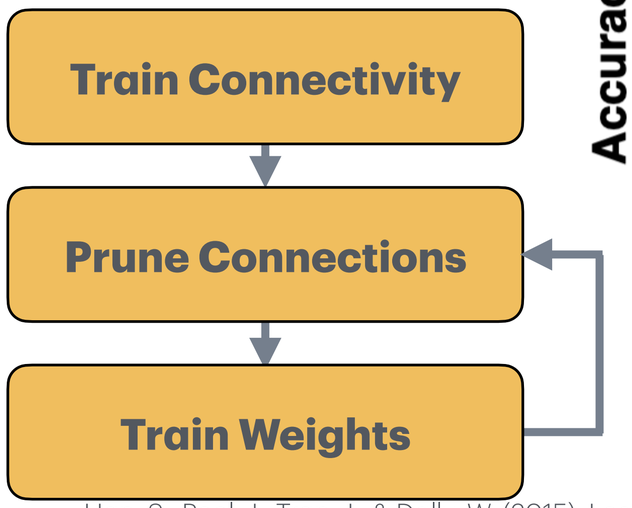
Han, S., Pool, J., Tran, J., & Dally, W. (2015). Learning both weights and connections for efficient neural network. Advances in neural information processing systems, 28.

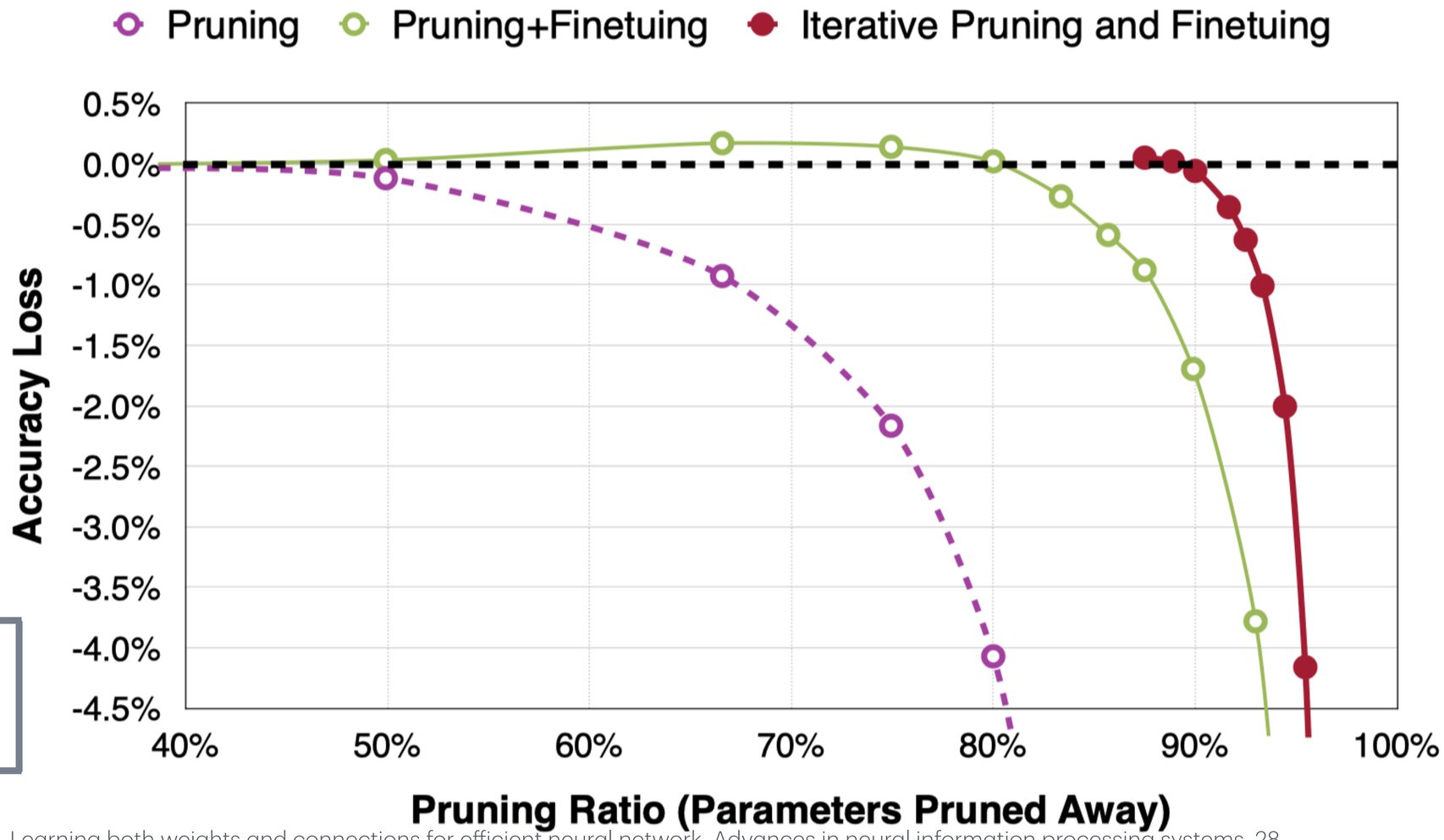
Figure source

Neural Network Pruning

Make neural network smaller by removing synapses and neurons







Han, S., Pool, J., Tran, J., & Dally, W. (2015). Learning both weights and connections for efficient neural network. Advances in neural information processing systems, 28.

Figure source

Neural Network Pruning

Make neural network smaller by removing synapses and neurons

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	1 M	0.38M	**3.2 x**	3.5 x

Han, S. (2017). Efficient methods and hardware for deep learning (Doctoral dissertation, Stanford University).

Neural Network Pruning

Pruning the NeuralTalk LSTM does not harm image caption quality

Baseline: a white bird is flying over water.

Pruned: a white bird is flying over water.

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

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

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

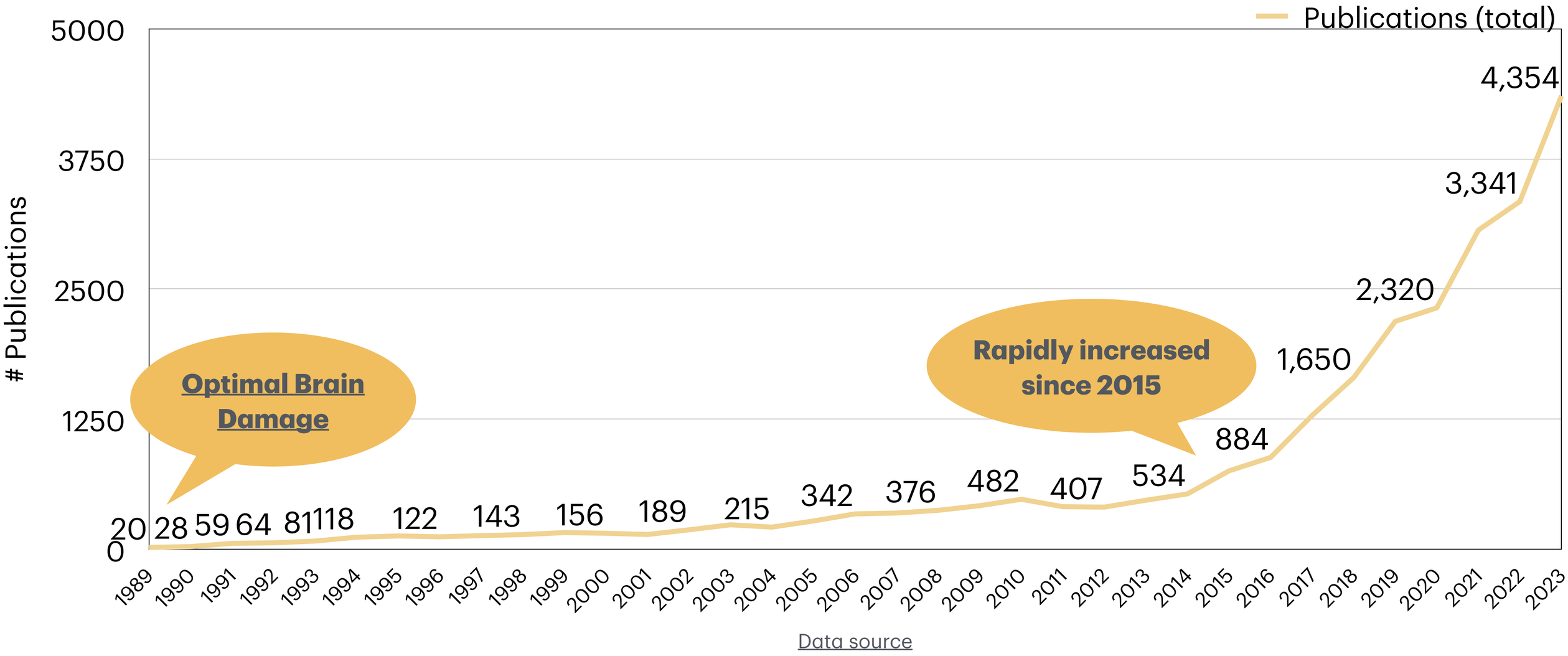
Pruned: a brown dog is running through a grassy area.

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

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

Han, S. (2017). Efficient methods and hardware for deep learning (Doctoral dissertation, Stanford University).

Pruning and Sparsity



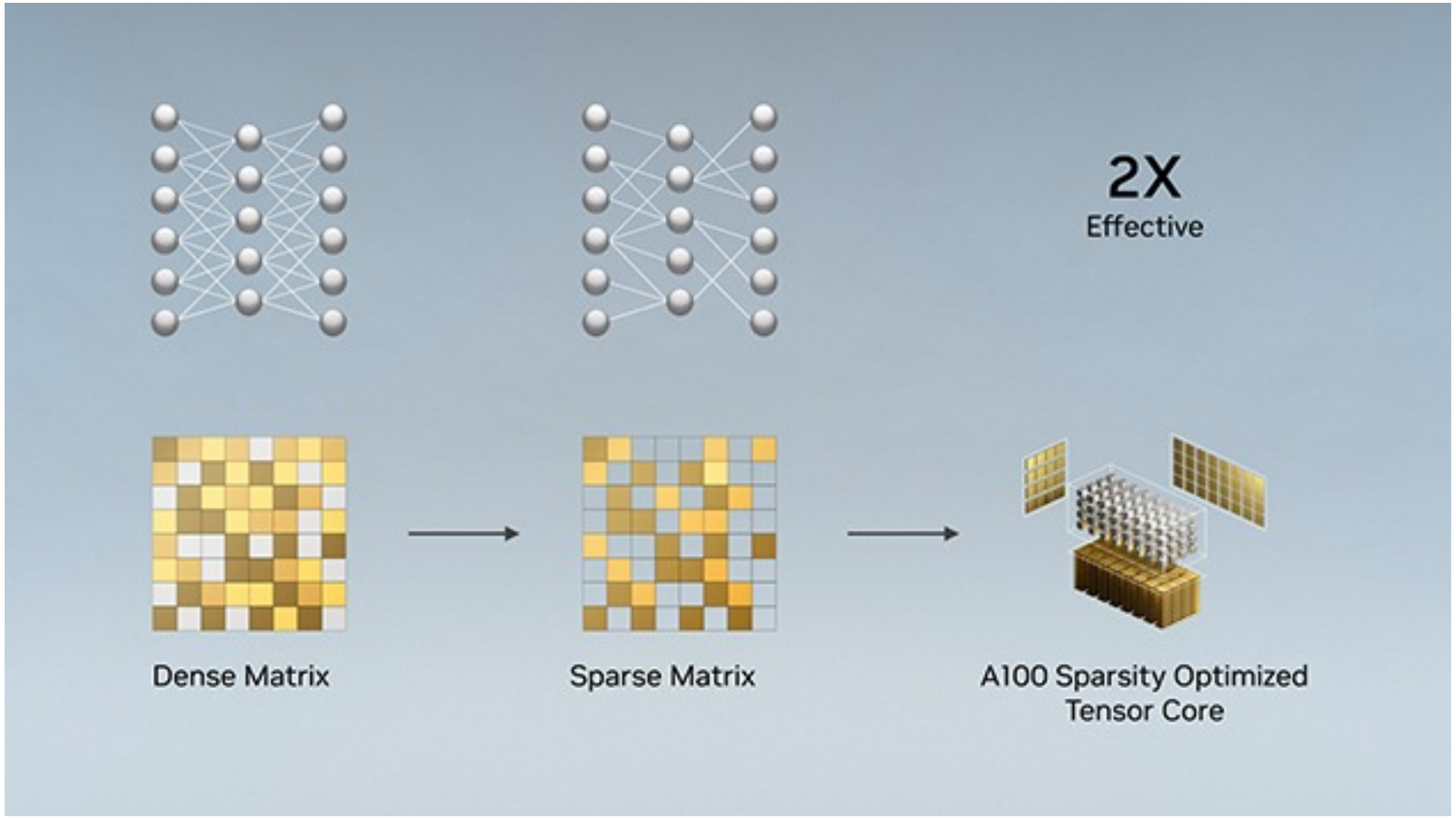
Data source

LeCun, Y., Denker, J., & Solla, S. (1989). Optimal brain damage. Advances in neural information processing systems, 2.

Pruning in the Industry

Hardware support for sparsity

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



How Sparsity Adds Umph to AI Inference

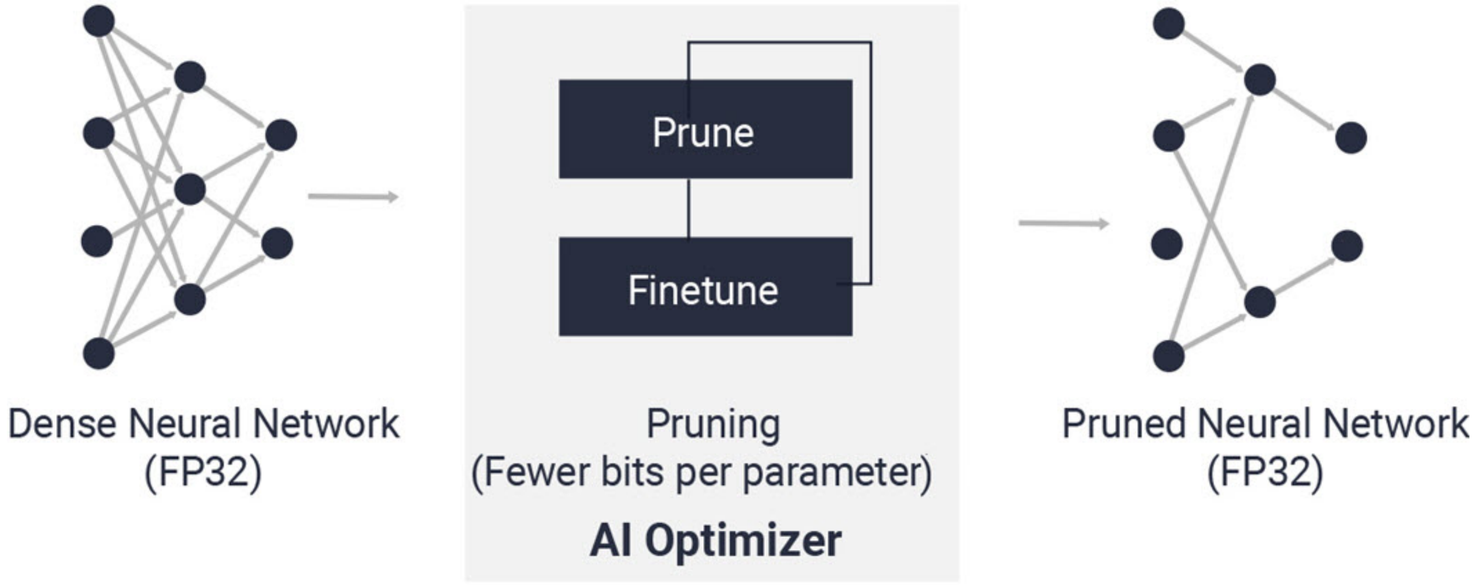
Pruning in the Industry

Hardware support for sparsity

- Reduce model complexity by 5X to 50X with minimal accuracy impact

XILINX VITIS™

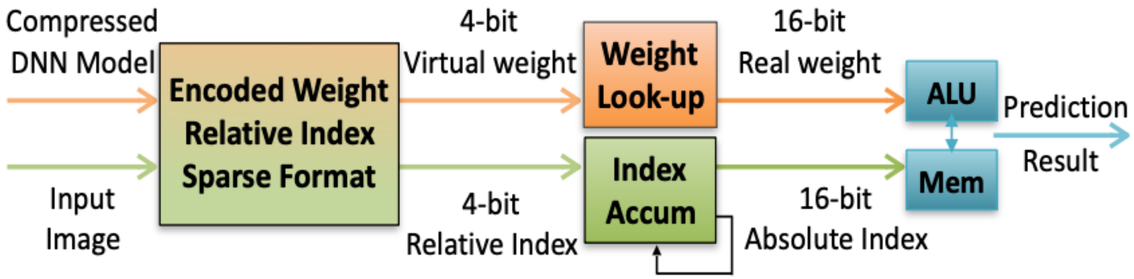
AMD XILINX



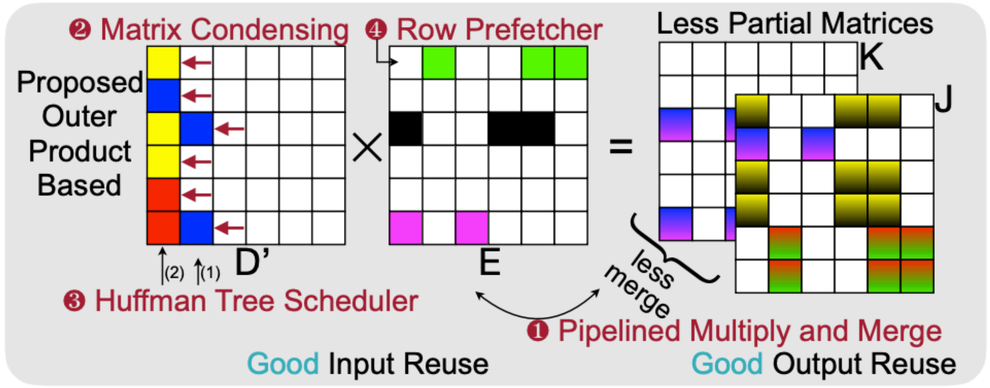
Vitis AI Optimizer

Pruning in the Industry

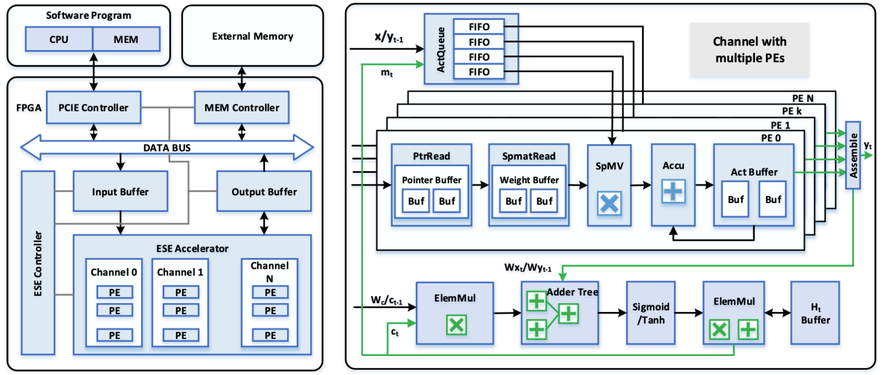
Hardware support for sparsity



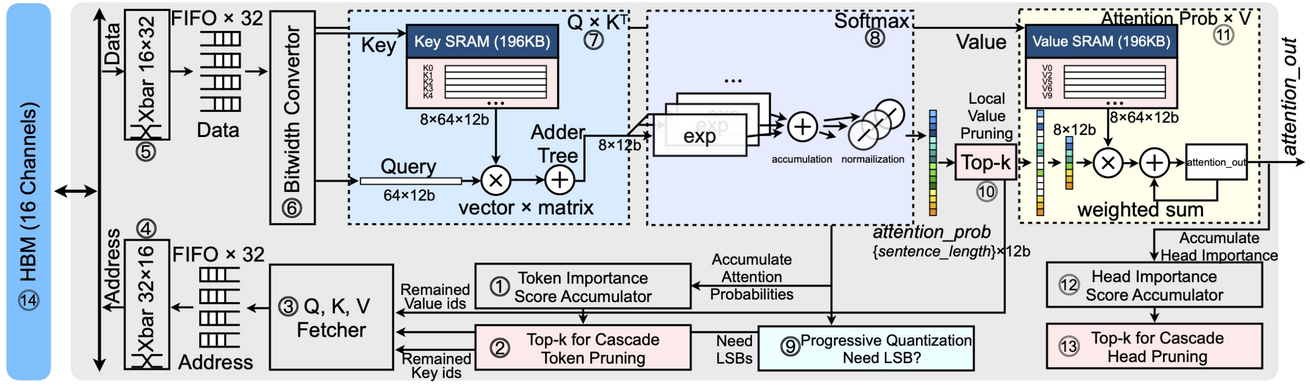
Han, S., Liu, X., Mao, H., Pu, J., Pedram, A., Horowitz, M. A., & Dally, W. J. (2016). EIE: Efficient inference engine on compressed deep neural network. ACM SIGARCH Computer Architecture News, 44(3), 243-254.



Zhang, Z., Wang, H., Han, S., & Dally, W. J. (2020, February). Sparch: Efficient architecture for sparse matrix multiplication. In 2020 IEEE International Symposium on High Performance Computer Architecture (HPCA) (pp. 261-274). IEEE.



Han, S., Kang, J., Mao, H., Hu, Y., Li, X., Li, Y., ... & Dally, W. B. J. (2017, February). Ese: Efficient speech recognition engine with sparse lstm on fpga. In Proceedings of the 2017 ACM/SIGDA International Symposium on Field-Programmable Gate Arrays (pp. 75-84).



Wang, H., Zhang, Z., & Han, S. (2021, February). Spatten: Efficient sparse attention architecture with cascade token and head pruning. In 2021 IEEE International Symposium on High-Performance Computer Architecture (HPCA) (pp. 97-110). IEEE.

How to formulate pruning?

Formulate Pruning

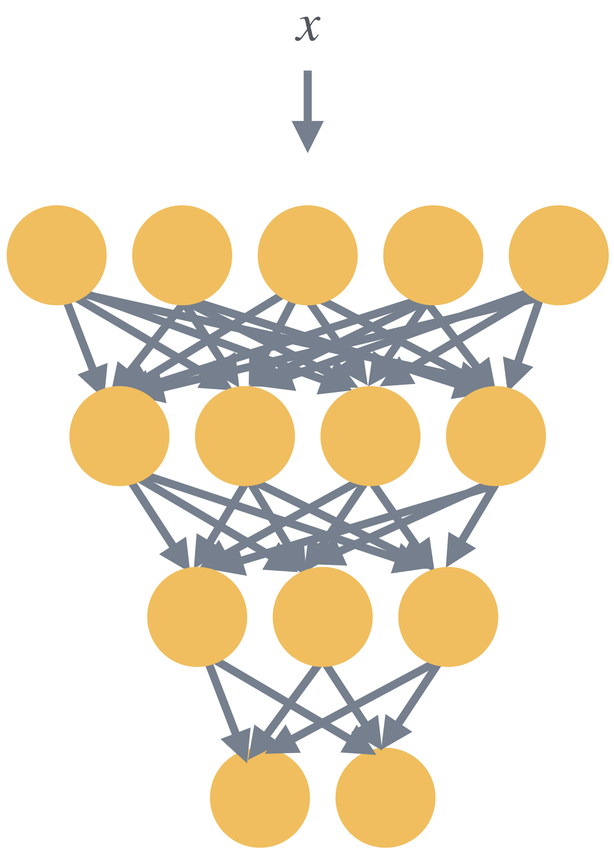
- In general, we could formulate the pruning as follows:

$$arg \min_{W_P} L(X; W_P)$$

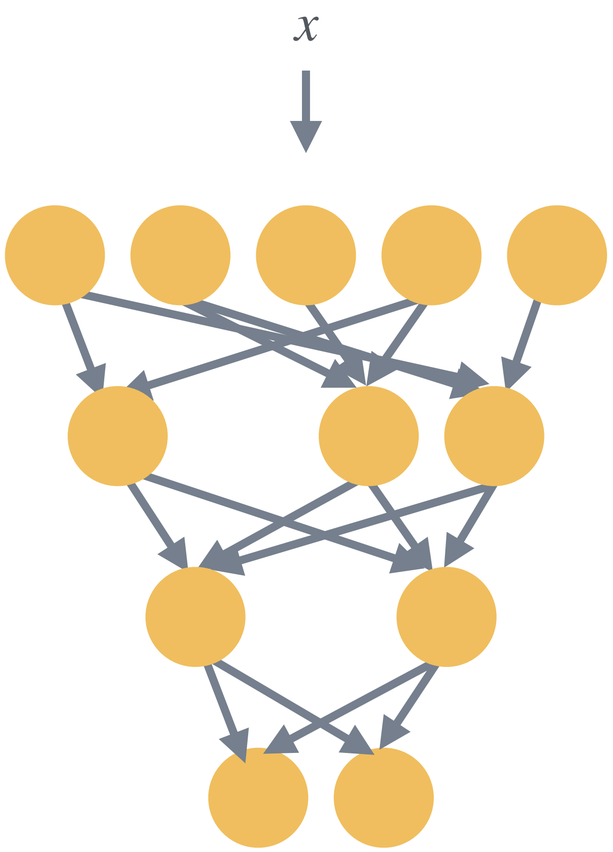
Subject to

$$\| W_P \|_0 \leq N$$

- L represents the objective function for neural network training
- X is input, W is the original weights, W_P is pruned weights
- $\| W_P \|_0$ calculates the #nonzeros in W_P, N is the target #nonzeros

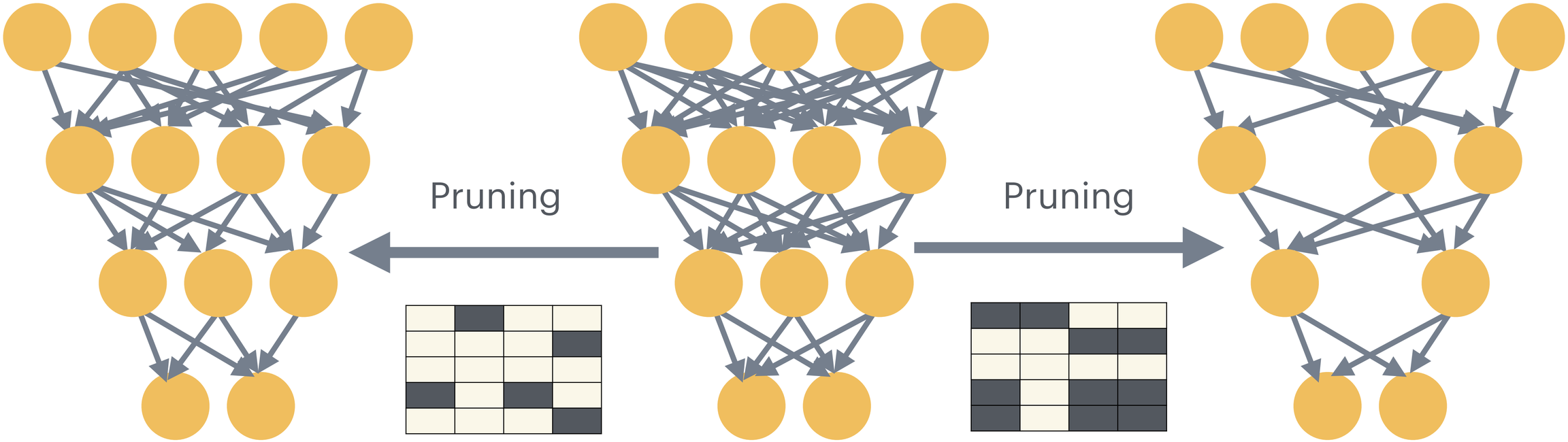


$$\arg \min_{W} L(X; W)$$



$$\arg \min_{W_P} L(X; W_P)$$
$$s.t. \| W_P \|_0 \leq N$$

Neural Network Pruning



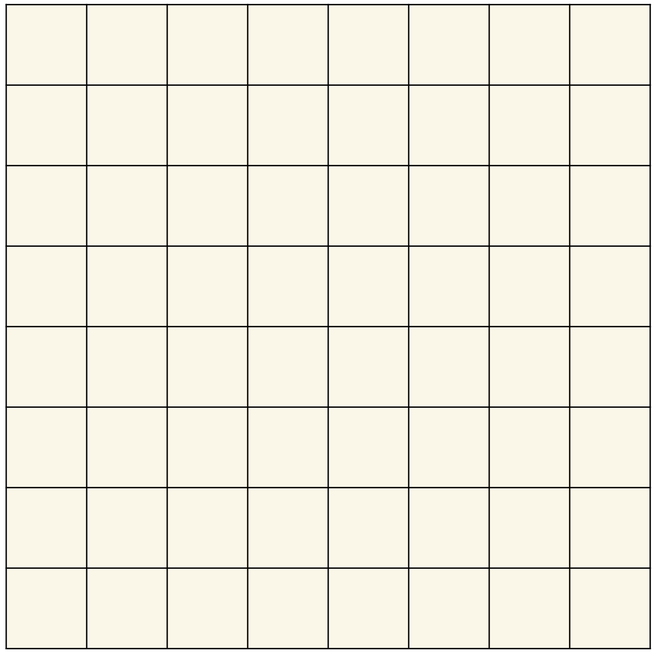
In what pattern should we prune the neural network?
Pruning 30%? 50%? 80%?
Which synapses? Which neurons?

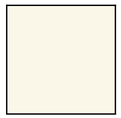
Pruning Granularity

Pruning can be performed at different granularities, from structured to non-structured

Pruning at Different Granularities

A simple example of 2D weight matrix



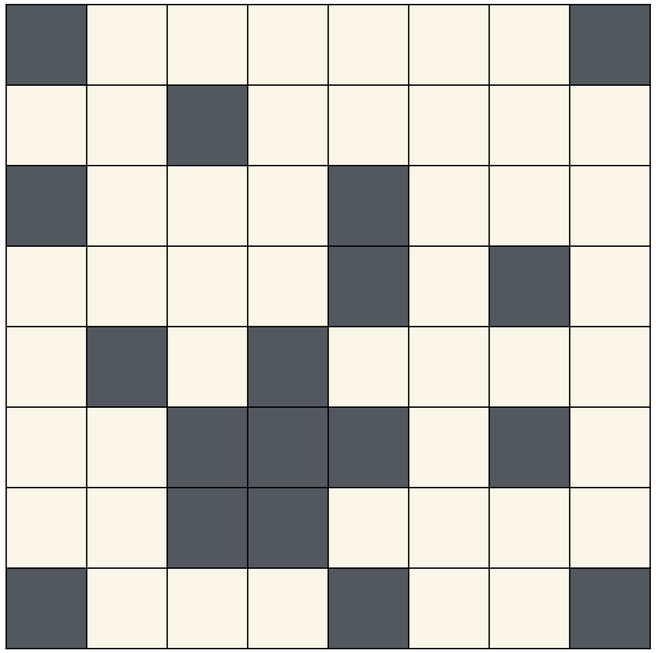


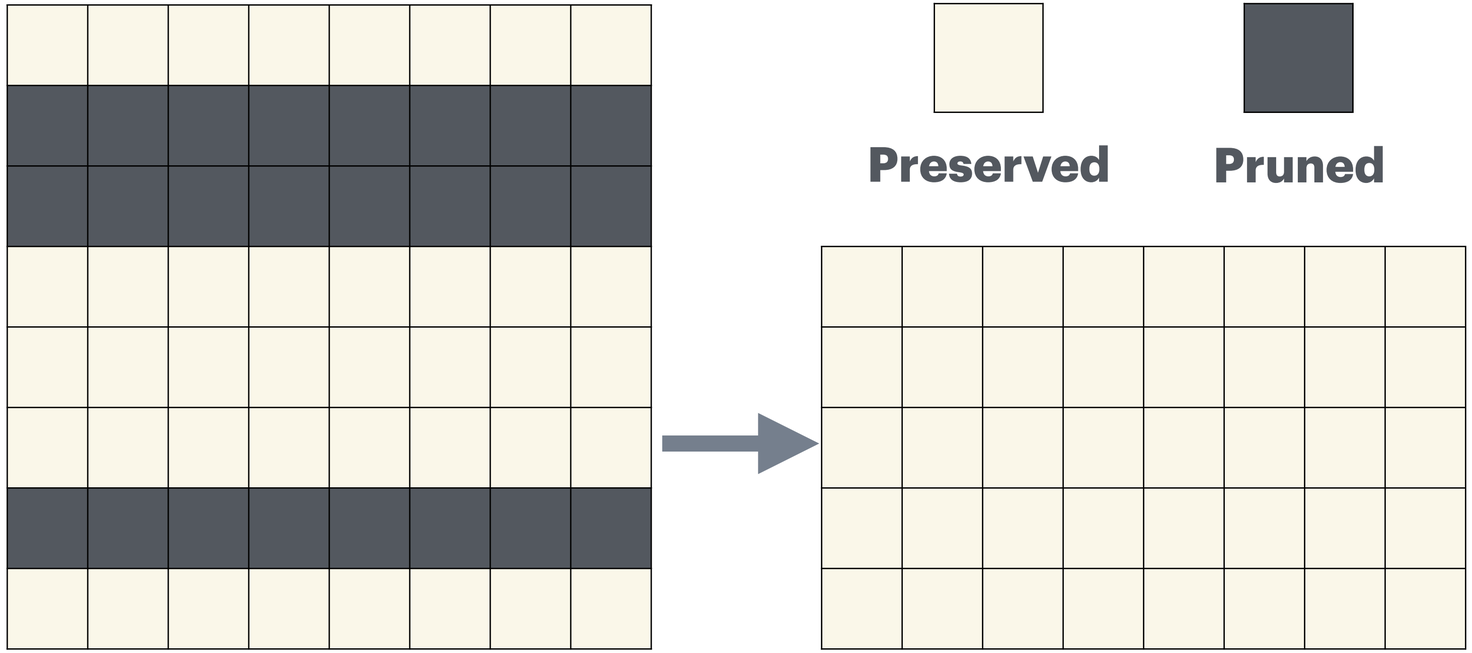
Preserved

Pruned

Pruning at Different Granularities

A simple example of 2D weight matrix



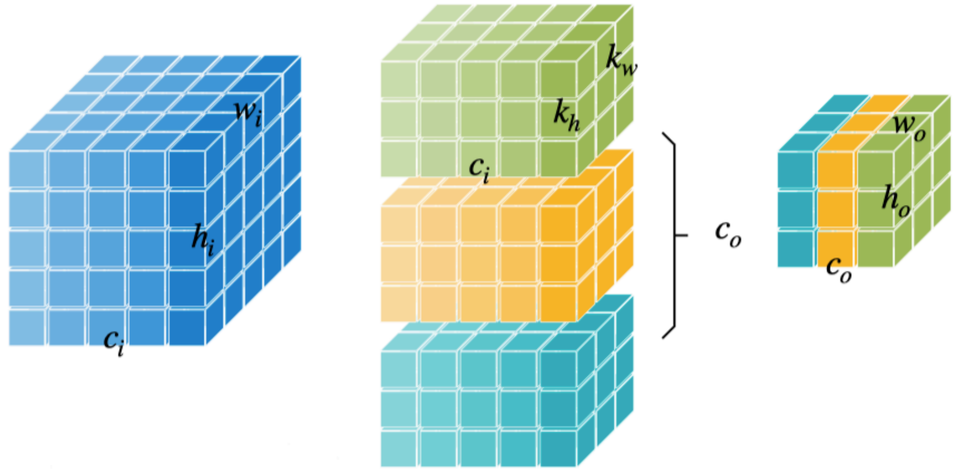


- **Fine-grained/Unstructured**
  - More flexible pruning index choice
  - Hard to accelerate (irregular)
- **Coarse-grained/Structured**
  - Less flexible pruning index choice (a subset of the fine-grained case)
  - Easy to accelerate (just a smaller matrix!)

Pruning at Different Granularities

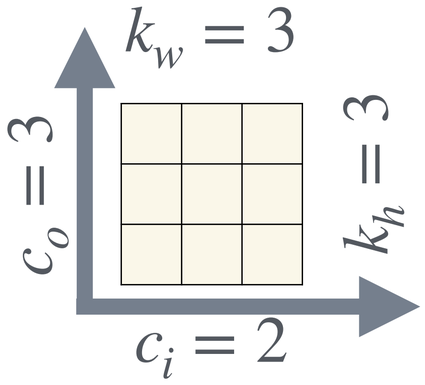
The case of convolutional layers

- The weights of convolutional layers have 4 dimensions $[c_o, c_i, k_h, k_w]$
  - c_i : input channels
  - c_o : output channels (or filters)
  - k_h : kernel size height
  - k_w : kernel size wideth
- The 4 dimensions give us more choices to select pruning granualrities



Pruning at Different Granularities

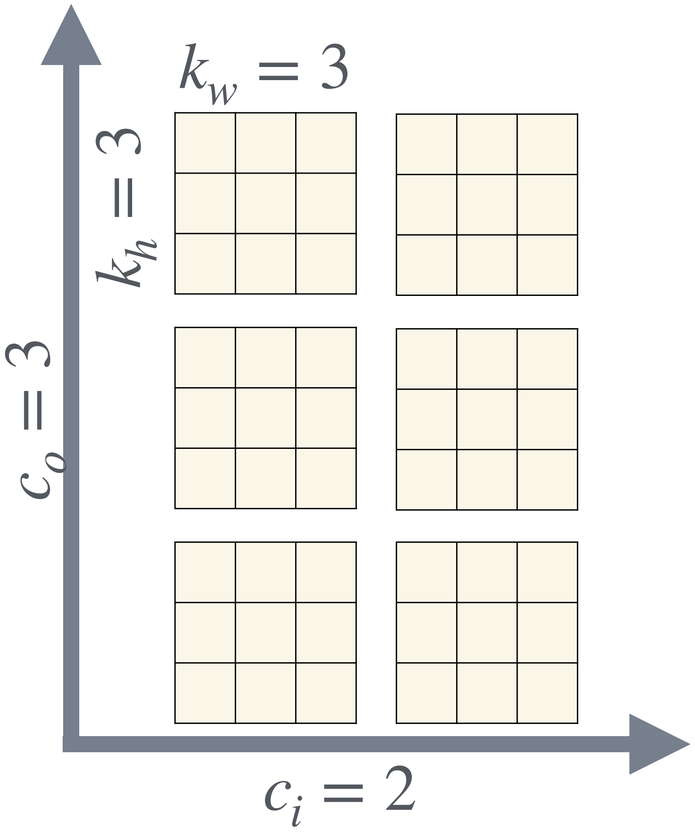
The case of convolutional layers

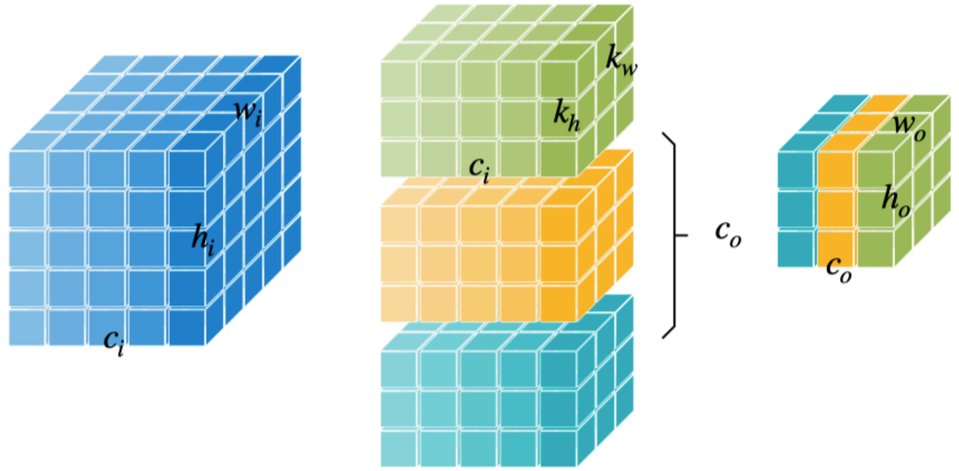


Preserved

Pruned

Notations

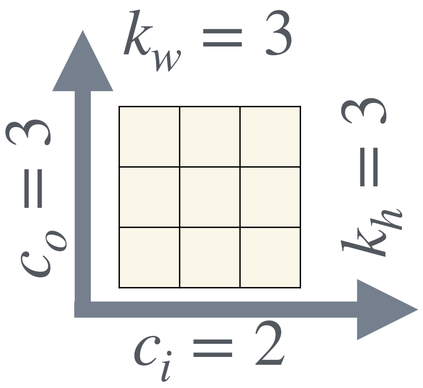




Pruning at Different Granularities

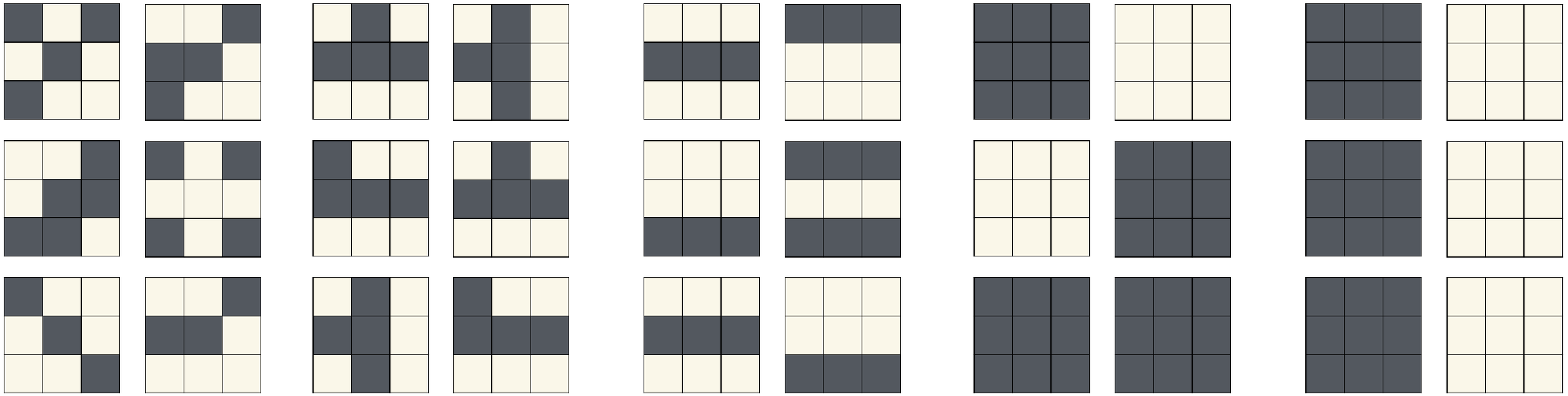
The case of convolutional layers

- Some of the commonly used pruning granularities



Preserved **Pruned**

Irregular **Regular**



Fine-grained Pruning **Pattern-based Pruning** **Vector-level Pruning** **Kernel-level Pruning** **Channel-level Pruning**

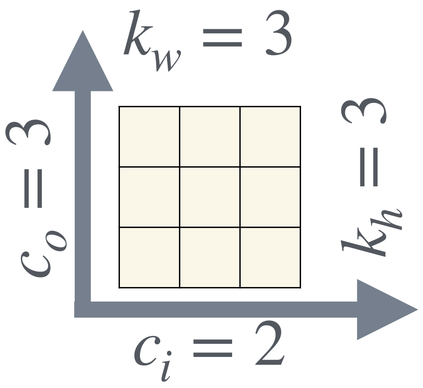
Mao, H., Han, S., Pool, J., Li, W., Liu, X., Wang, Y., & Dally, W. J. (2017). Exploring the granularity of sparsity in convolutional neural networks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops (pp. 13-20).

Pruning at Different Granularities

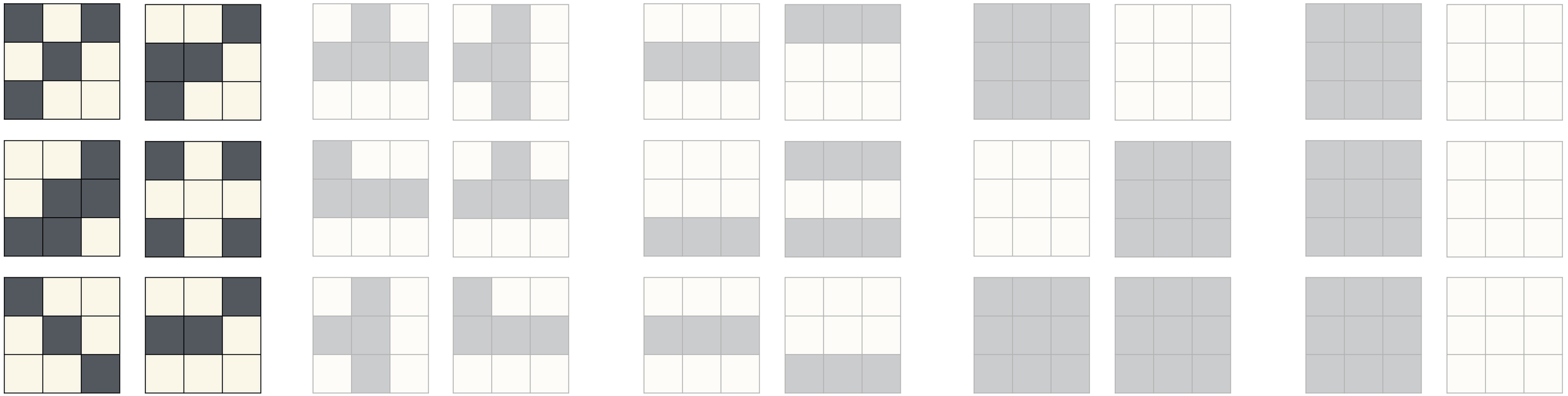
The case of convolutional layers

- Some of the commonly used pruning granularities

Pros and Cons?



Irregular ⟵ ⟶ **Regular**



Fine-grained Pruning | Pattern-based Pruning | Vector-level Pruning | Kernel-level Pruning | Channel-level Pruning

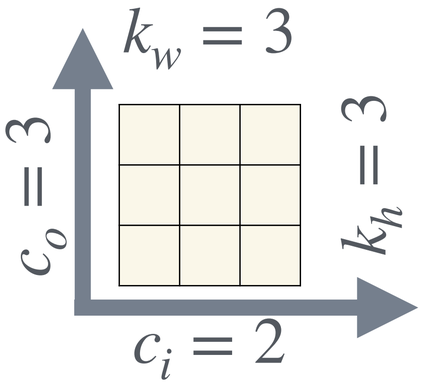
Mao, H., Han, S., Pool, J., Li, W., Liu, X., Wang, Y., & Dally, W. J. (2017). Exploring the granularity of sparsity in convolutional neural networks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops (pp. 13-20).

Fine-Grained Pruning

- Flexible pruning indices
- Usually with a larger compression ratio since it is flexible to find redundant weights
  - Note: the reduced #params does not directly translate to a speed up
- Can deliver speed up on some custom hardware but not easily on off-the-shelf hardware (e.g. GPU)

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

Pruning at Different Granularities

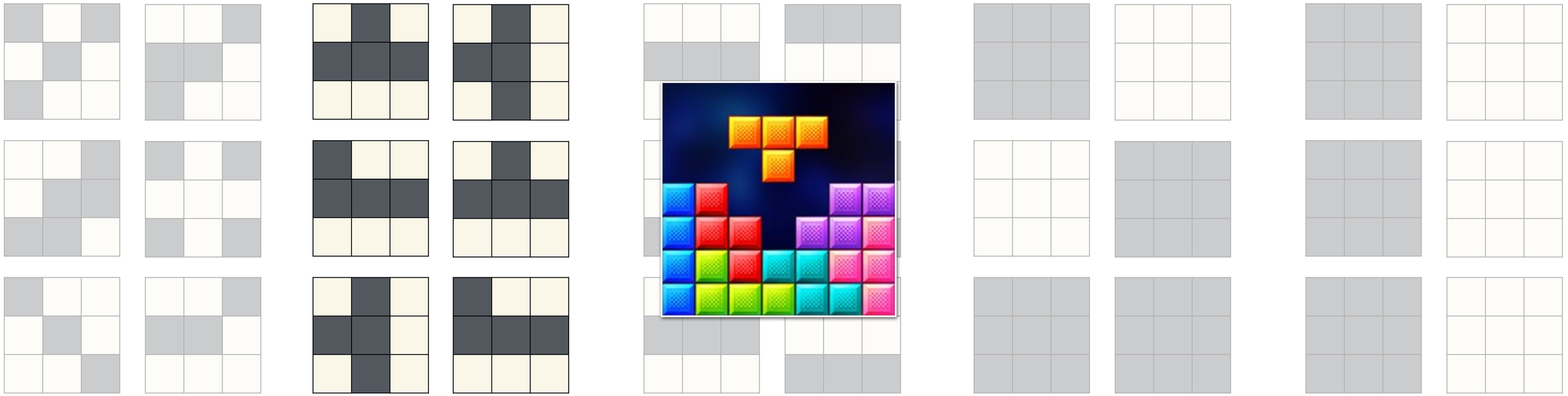


The case of convolutional layers

- Some of the commonly used pruning granularities

Pros and Cons?

Irregular ⟵ ⟶ **Regular**

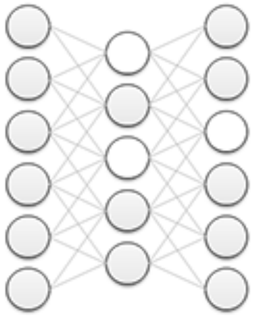


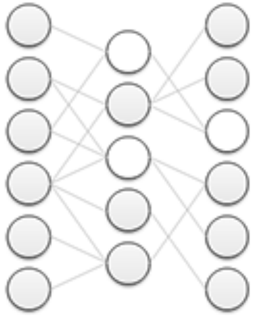
Fine-grained Pruning | **Pattern-based Pruning** | Vector-level Pruning | Kernel-level Pruning | Channel-level Pruning

Mao, H., Han, S., Pool, J., Li, W., Liu, X., Wang, Y., & Dally, W. J. (2017). Exploring the granularity of sparsity in convolutional neural networks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops (pp. 13-20).

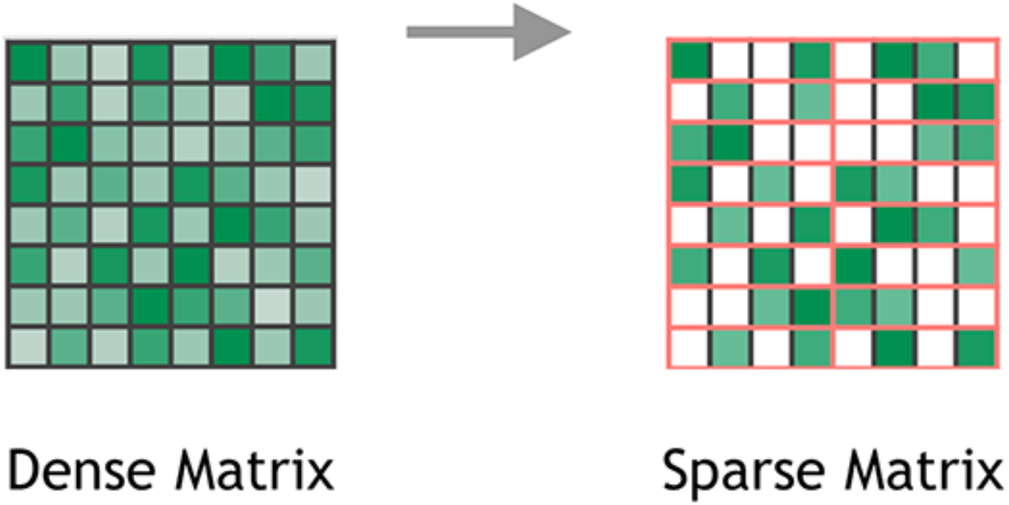
Pattern-Based Pruning

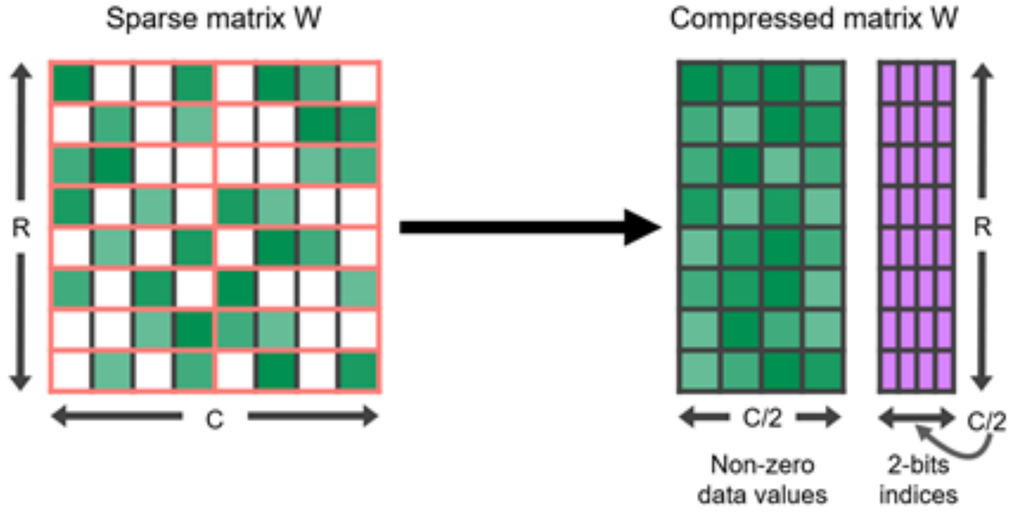
N:M sparsity





- For each contiguous M elements, N of them is pruned
- A classic case is 2:4 sparsity (50% sparsity)

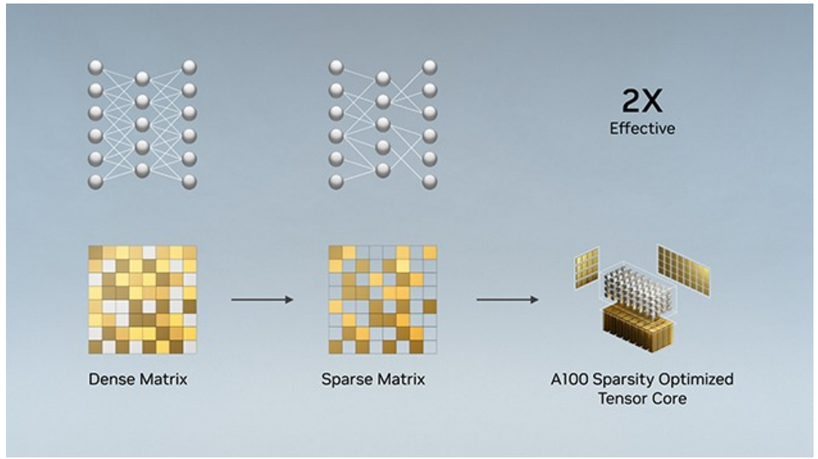




Accelerating Inference with Sparsity Using the NVIDIA Ampere Architecture and NVIDIA TensorRT

Pattern-Based Pruning

N:M sparsity

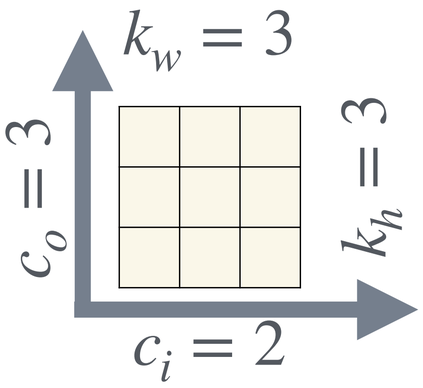


- For each contiguous M elements, N of them is pruned
- A classic case is 2:4 sparsity (50% sparsity)
- It is supported by NVIDIA's Ampere GPU Architecture (~2x speed up)
- Usually maintains accuracy

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

Accelerating Inference with Sparsity Using the NVIDIA Ampere Architecture and NVIDIA TensorRT

Pruning at Different Granularities

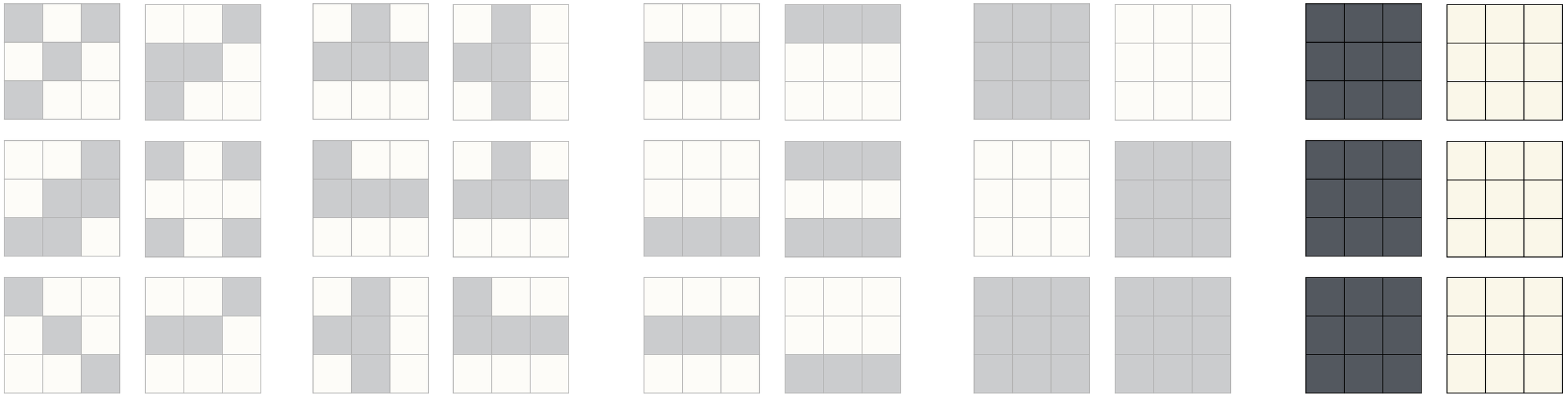


The case of convolutional layers

- Some of the commonly used pruning granularities

Pros and Cons?

Irregular ⟵ ⟶ **Regular**



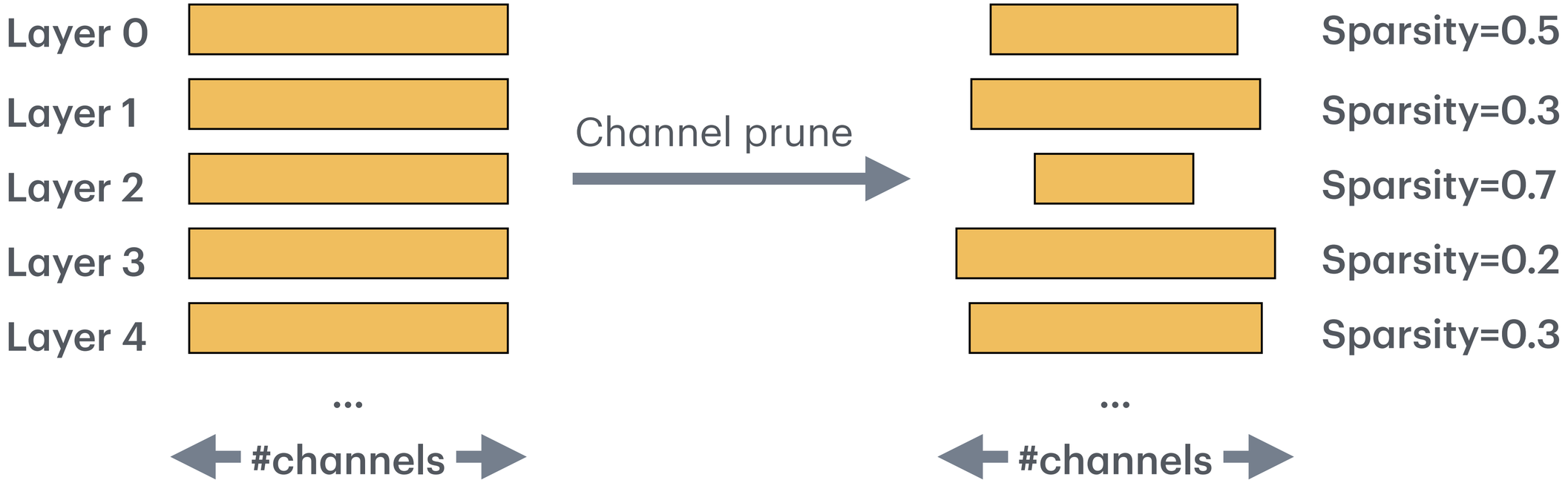
Fine-grained Pruning | Pattern-based Pruning | Vector-level Pruning | Kernel-level Pruning | **Channel-level Pruning**

Mao, H., Han, S., Pool, J., Li, W., Liu, X., Wang, Y., & Dally, W. J. (2017). Exploring the granularity of sparsity in convolutional neural networks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops (pp. 13-20).

Channel-Level Pruning

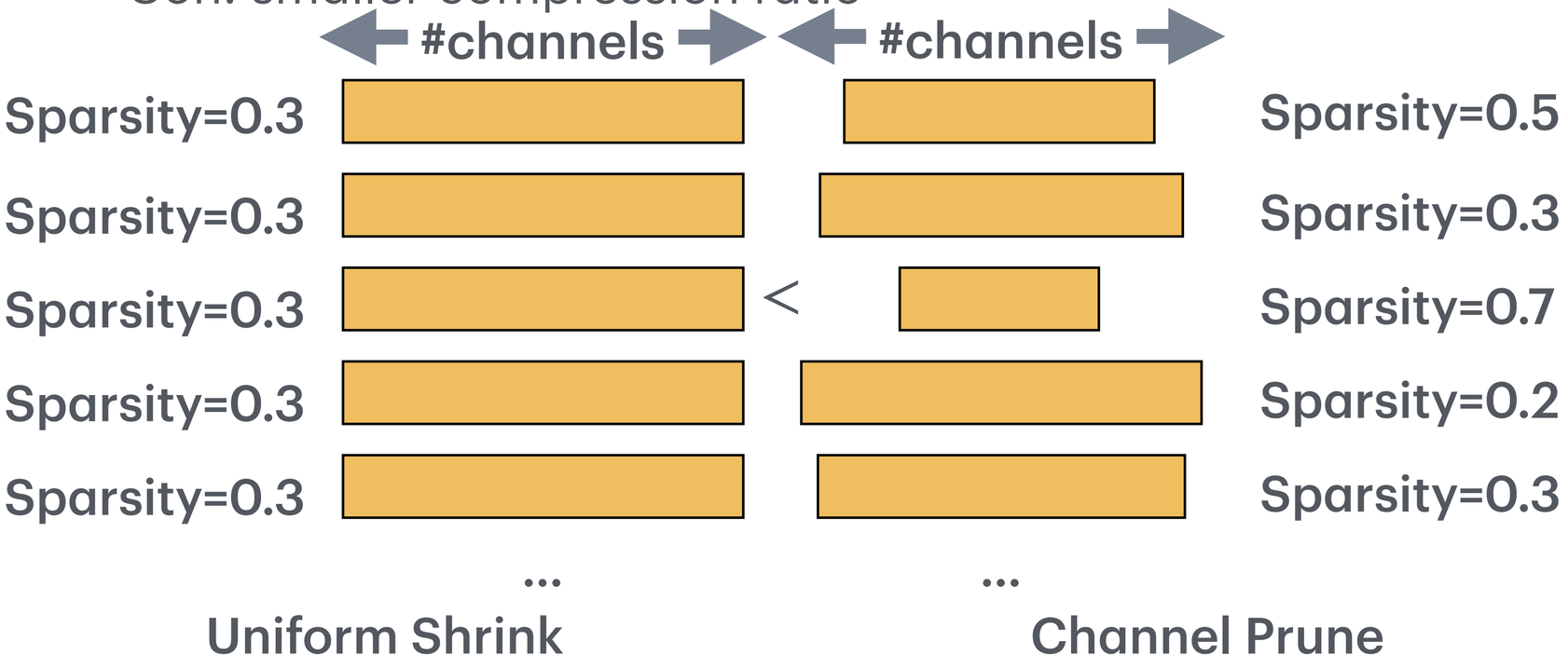
- Pro: Direct speed up due to reduced channel numbers (leading to an NN with smaller #chanels)

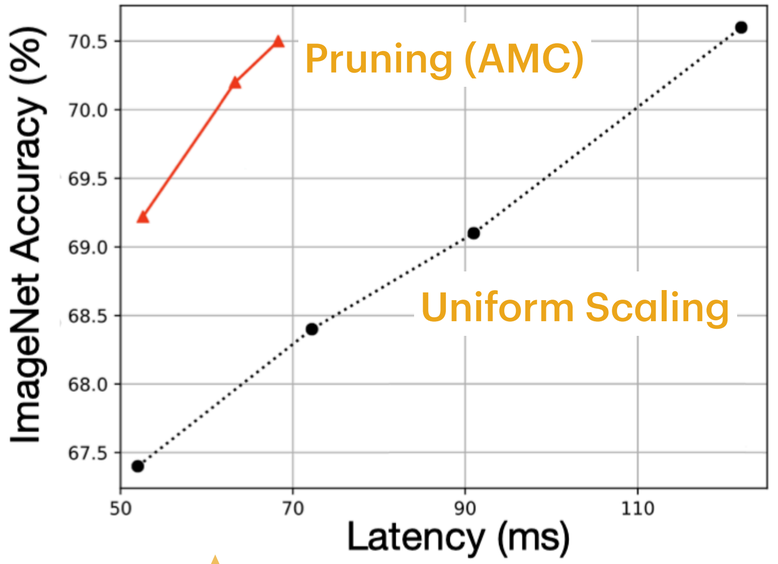
- Con: smaller compression ratio



Channel-Level Pruning

- Pro: Direct speed up due to reduced channel numbers (leading to an NN with smaller #chanels)
- Con: smaller compression ratio

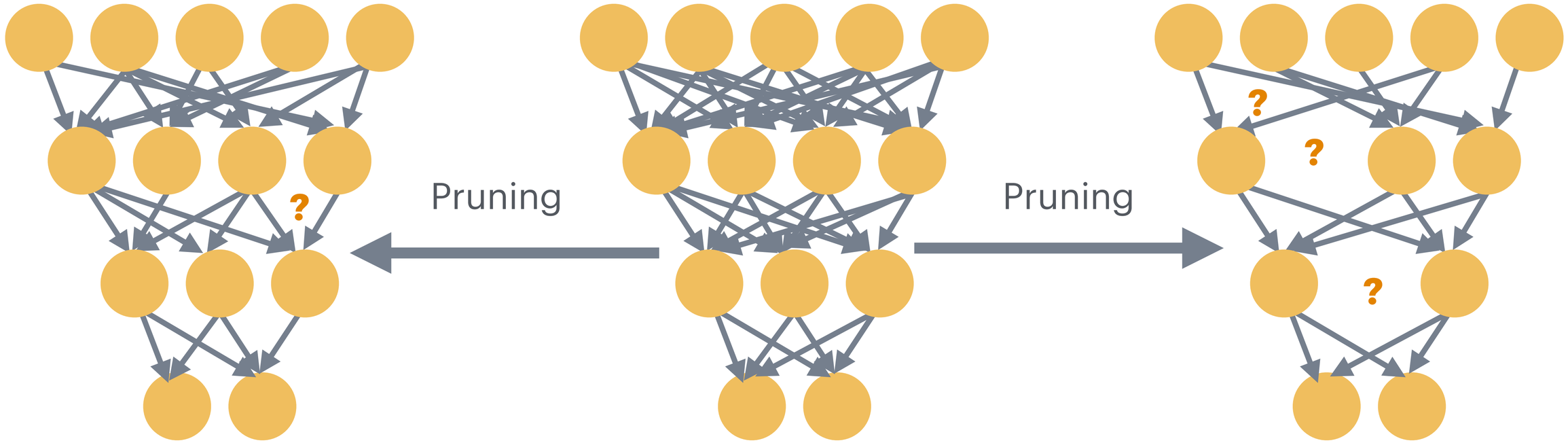




Wisely choose the pruning ratio for different layer is better than uniformly sample the different layers.

He, Y., Lin, J., Liu, Z., Wang, H., Li, L. J., & Han, S. (2018). AMC: Automl for model compression and acceleration on mobile devices. In Proceedings of the European conference on computer vision (ECCV) (pp. 784-800).

Neural Network Pruning



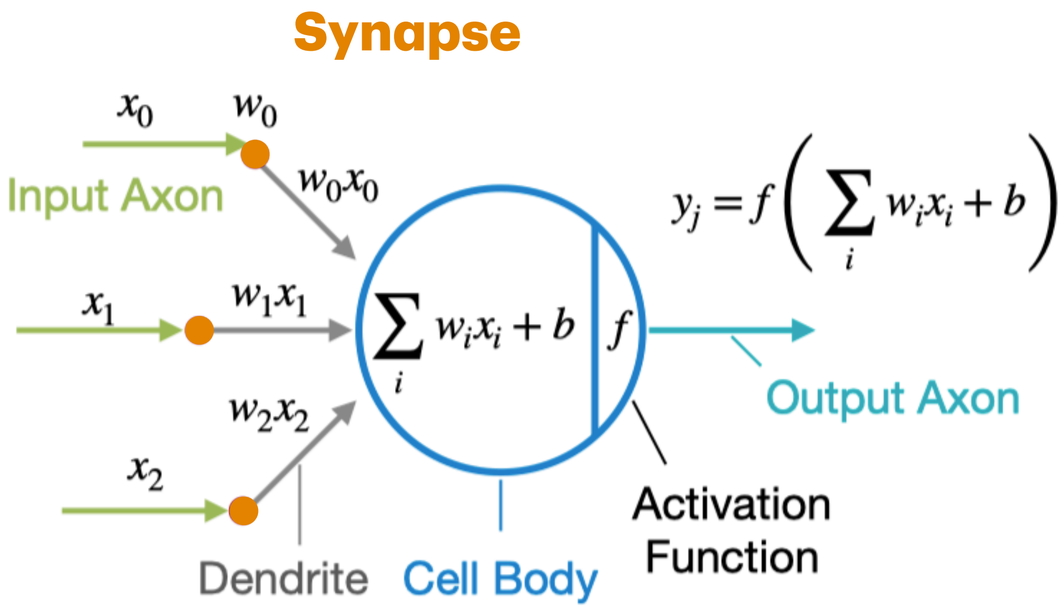
In what pattern should we prune the neural network?
Which synapses? Which neurons?

Pruning Criterion

What synapses and neurons should we prune?

Selection of Synapses to Prune

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

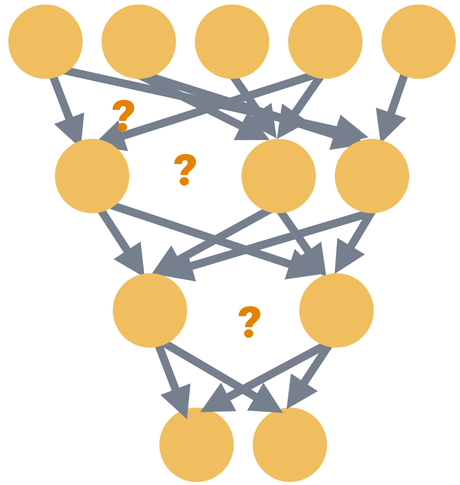


Example

$$f(\cdot) = \text{ReLU}(\cdot), W = [10, -8, 0.1]$$
$$\rightarrow y = \text{ReLU}(10x_0 - 8x_1 + 0.1x_2)$$

If one weight will be removed, which one?

0.1

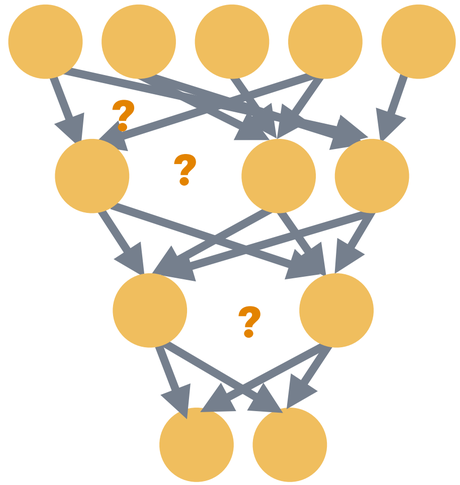


Selection of Synapses to Prune

Magnitude-based

Scaling-based

Second-Order-based



Selection of Synapses to Prune

Magnitude-based

Scaling-based

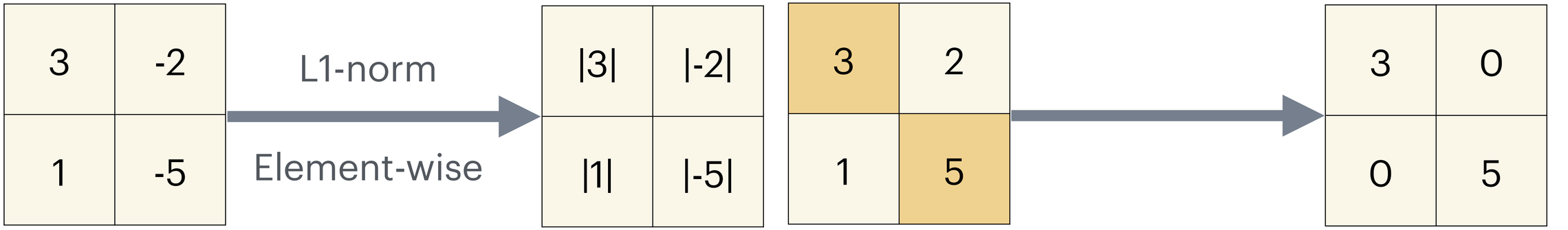
Second-Order-based

Magnitude-Based Pruning

A heuristic pruning criterion

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

$$\text{Importance} = |W|$$



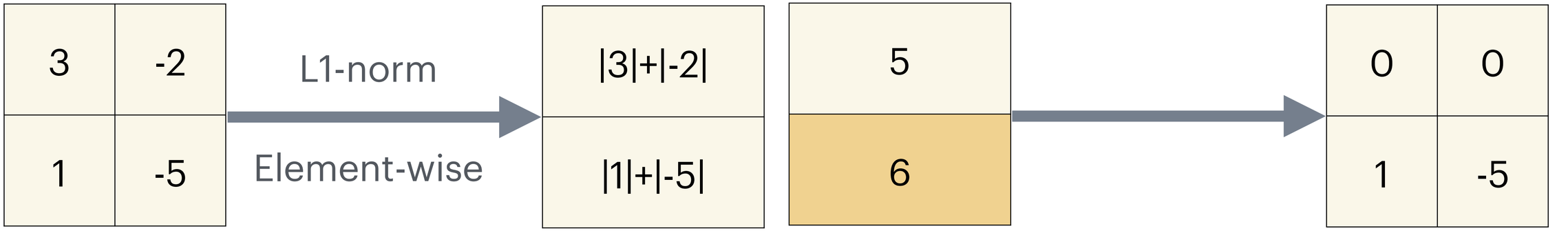
Han, S., Pool, J., Tran, J., & Dally, W. (2015). Learning both weights and connections for efficient neural network. Advances in neural information processing systems, 28.

Magnitude-Based Pruning

A heuristic pruning criterion

- 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:

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



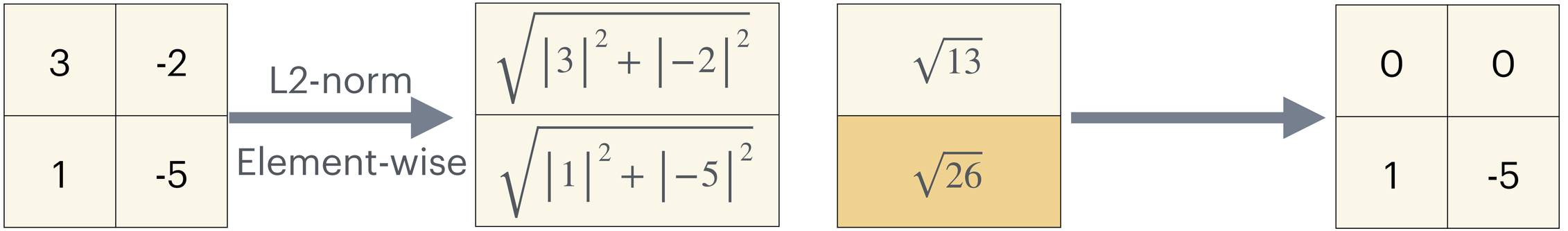
Han, S., Pool, J., Tran, J., & Dally, W. (2015). Learning both weights and connections for efficient neural network. Advances in neural information processing systems, 28.

Magnitude-Based Pruning

A heuristic pruning criterion

- 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:

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



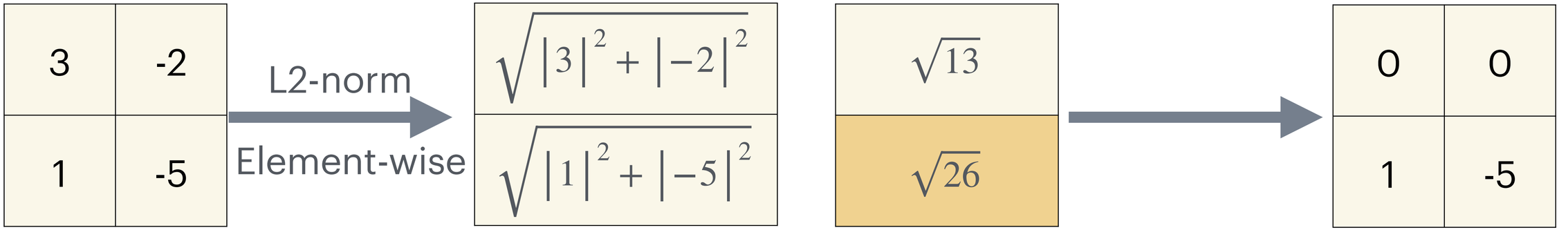
Han, S., Pool, J., Tran, J., & Dally, W. (2015). Learning both weights and connections for efficient neural network. Advances in neural information processing systems, 28.

Magnitude-Based Pruning

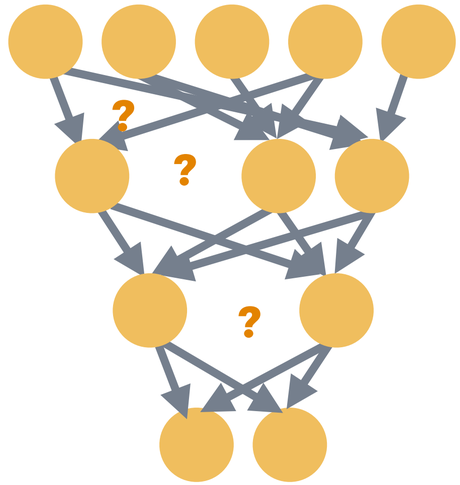
A heuristic pruning criterion

- 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:

$$\| W^{(S)} \|_p = \left(\sum_{i \in S} \left| w_i \right|^2 \right)^{\frac{1}{p}}, \text{ where } W^{(S)} \text{ is the structural set of parameters}$$



Wen, W., Wu, C., Wang, Y., Chen, Y., & Li, H. (2016). Learning structured sparsity in deep neural networks. Advances in neural information processing systems, 29.



Selection of Synapses to Prune

Magnitude-based

Scaling-based

Second-Order-based

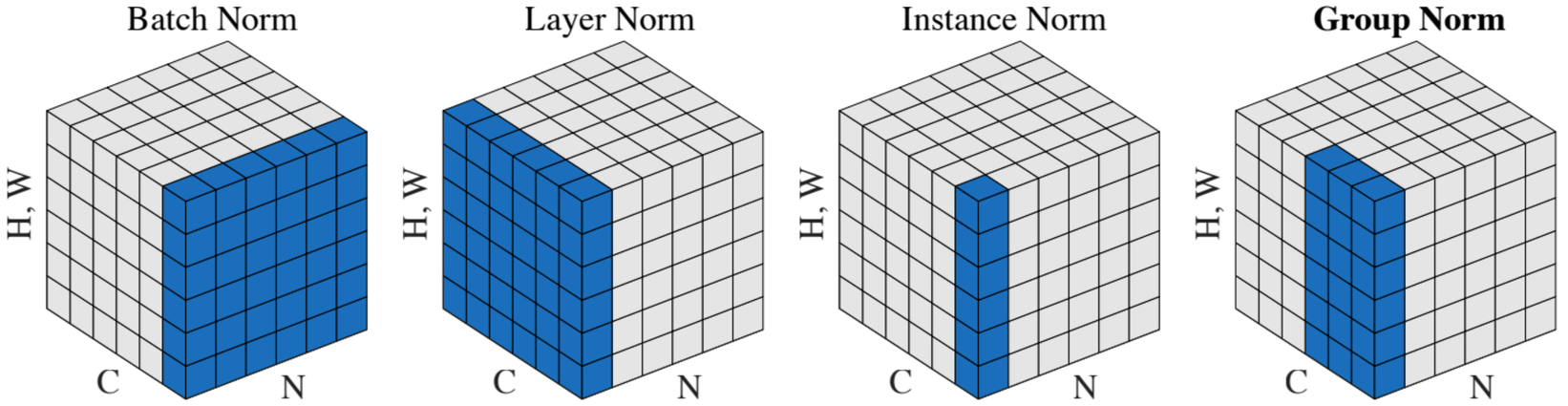
Normalization Layer

$$\mu_i = \frac{1}{m}\sum_{k\in\mathcal{S}_i} x_k$$

$$\sigma_i = \sqrt{\frac{1}{m}\sum_{k\in\mathcal{S}_i}(x_k-\mu_i)^2 + \epsilon}$$

Normalizing the features makes optimization faster

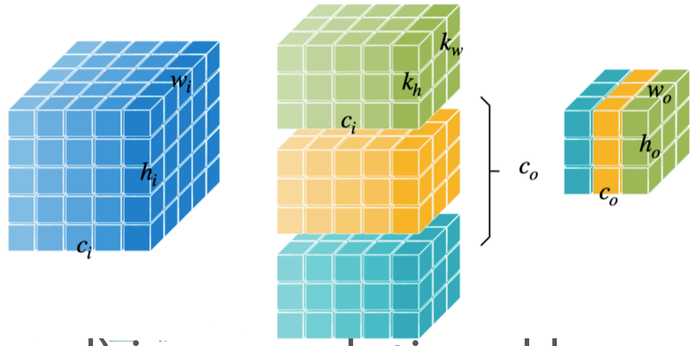
- Normalize the features according to: $\hat{x}_i = \frac{1}{\sigma}(x_i - \mu_i)$
  - μ_i is the mean, and σ_i is the standard deviation (std) over the set of pixels $\mathcal{S}_i$
- Then, learn a per-channel linear transform (trainable scale γ and shift β) to compensate for the possible loss of representational ability as follows: $y = \gamma_{i_c}\hat{x}_i + \beta_{i_c}$



Different normalization use different definitions of the set $\mathcal{S}_i$ (colored in blue)

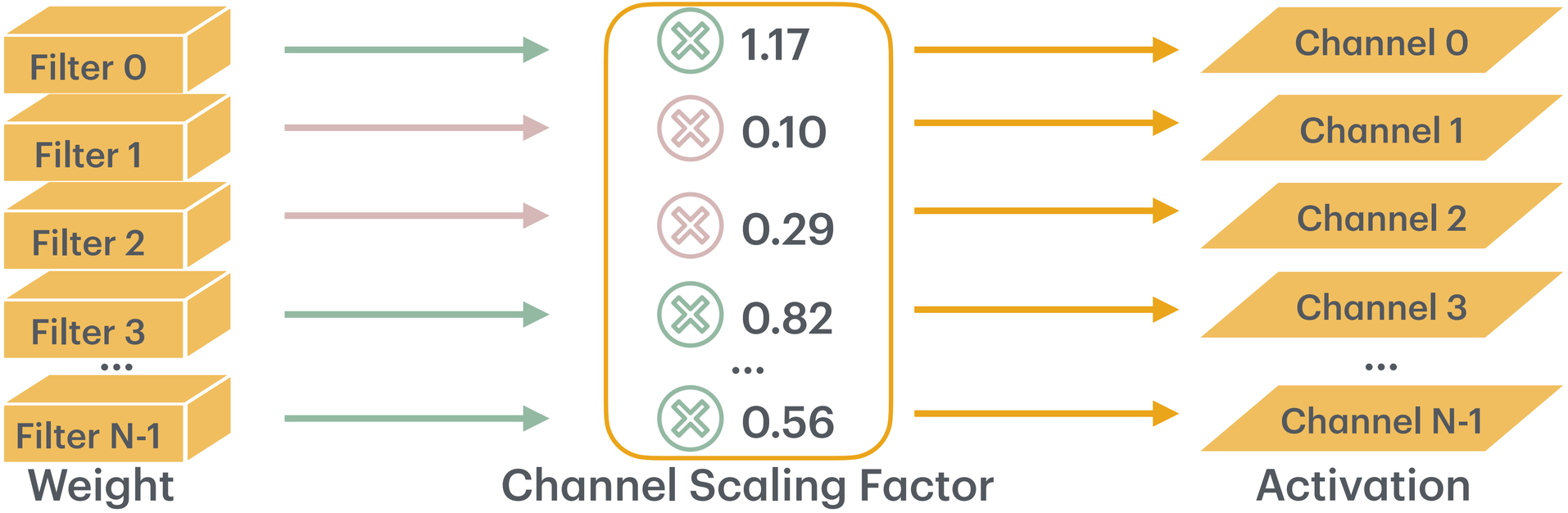
Ioffe, S. (2015). Batch normalization: Accelerating deep network training by reducing internal covariate shift. arXiv preprint arXiv:1502.03167.

Scaling-Based Pruning



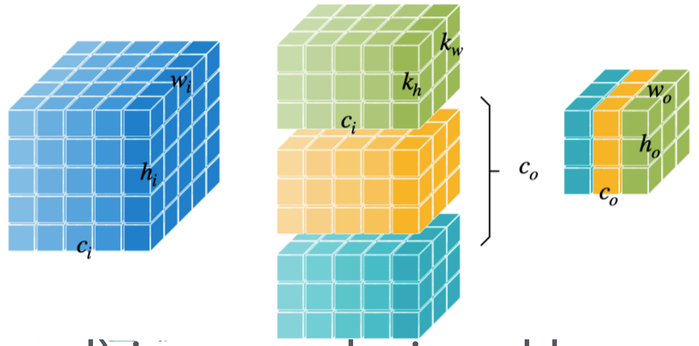
Pruning criterion for filter pruning

- A **scaling factor** is associated with each filter (i.e., output channel) in convolutional layers
  - The scaling factor is multiplied by the output of that channel
  - The scaling factor is *trainable* parameter



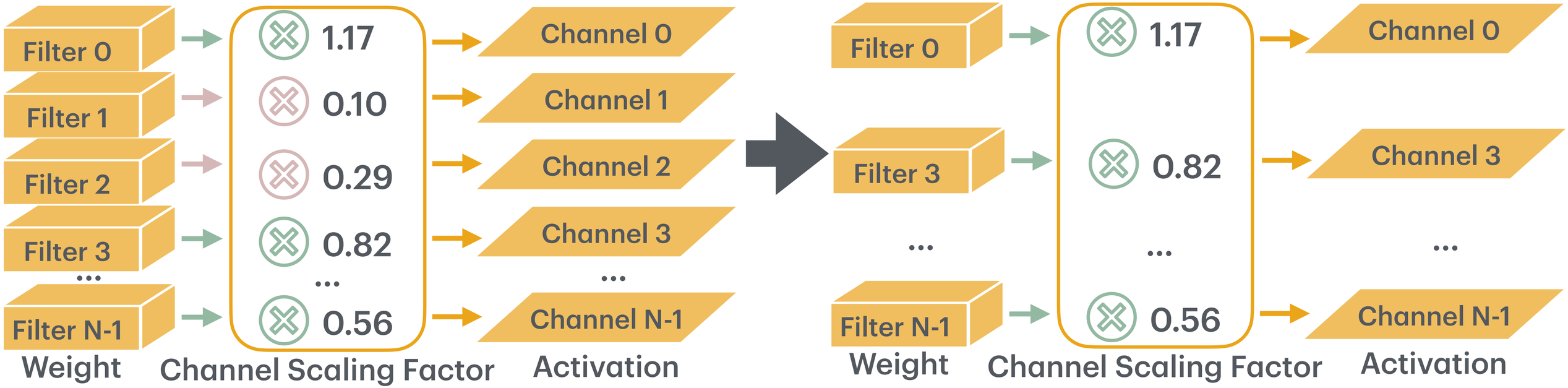
Liu, Z., Li, J., Shen, Z., Huang, G., Yan, S., & Zhang, C. (2017). Learning efficient convolutional networks through network slimming. In Proceedings of the IEEE international conference on computer vision (pp. 2736-2744).

Scaling-Based Pruning



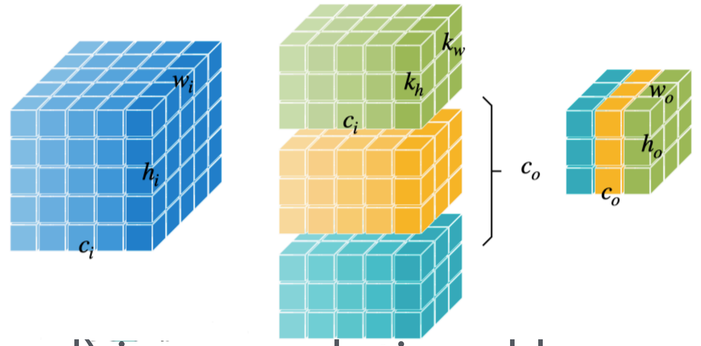
Pruning criterion for filter pruning

- A **scaling factor** is associated with each filter (i.e., output channel) in convolutional layers
- The filters/output channels with **small scaling factor** magnitude will be pruned



Liu, Z., Li, J., Shen, Z., Huang, G., Yan, S., & Zhang, C. (2017). Learning efficient convolutional networks through network slimming. In Proceedings of the IEEE international conference on computer vision (pp. 2736-2744).

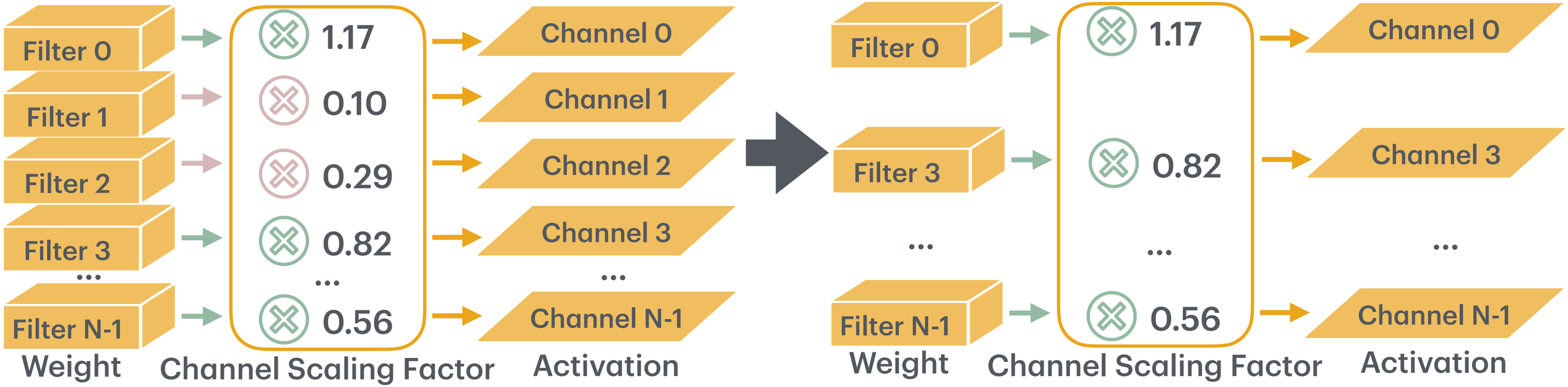
Scaling-Based Pruning



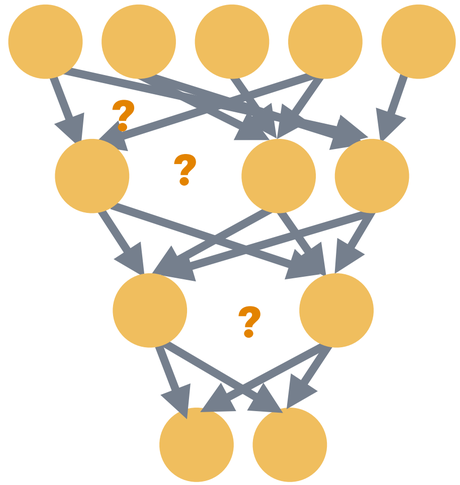
Pruning criterion for filter pruning

- A scaling factor is associated with each filter (i.e., output channel) in convolutional layers
- The **scaling factors** can be reused from the batch normalization layer

$$z_o = \gamma \frac{z_i - \mu_{\mathcal{B}}}{\sqrt{\sigma_{\mathcal{B}}^2 + \epsilon}} + \beta$$



Liu, Z., Li, J., Shen, Z., Huang, G., Yan, S., & Zhang, C. (2017). Learning efficient convolutional networks through network slimming. In Proceedings of the IEEE international conference on computer vision (pp. 2736-2744).



Selection of Synapses to Prune

Magnitude-based

Scaling-based

Second-Order-based

Second-Order-Based Pruning

Minimize the error on loss function introduced by pruning synapses

- The induced error can be approximated by a Taylor series

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

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

LeCun, Y., Denker, J., & Solla, S. (1989). Optimal brain damage. Advances in neural information processing systems, 2.

Second-Order-Based Pruning

Minimize the error on loss function introduced by pruning synapses

- The induced error can be approximated by a Taylor series

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

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

- Optimal Brain Damanage assumes that
  - The objective function L is nearly quadratic: the last term is neglected

LeCun, Y., Denker, J., & Solla, S. (1989). Optimal brain damage. Advances in neural information processing systems, 2.

Second-Order-Based Pruning

Minimize the error on loss function introduced by pruning synapses

- The induced error can be approximated by a Taylor series

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

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

- Optimal Brain Damanage 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

LeCun, Y., Denker, J., & Solla, S. (1989). Optimal brain damage. Advances in neural information processing systems, 2.

Second-Order-Based Pruning

Minimize the error on loss function introduced by pruning synapses

- The induced error can be approximated by a Taylor series

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

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

- Optimal Brain Damanage 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

LeCun, Y., Denker, J., & Solla, S. (1989). Optimal brain damage. Advances in neural information processing systems, 2.

Second-Order-Based Pruning

Minimize the error on loss function introduced by pruning synapses

- Optimal Brain Damanage 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

$$\delta L = L(X; W) - L(X; W_P \,|\, w_i = 0) \approx \frac{1}{2} h_{ii} w_i^2, \text{ where } h_{ii} = \frac{\partial^2 L}{\partial w_i \partial w_j}$$

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

$$\text{importance}_{w_i} = \left|\partial L_i\right| = \frac{1}{2} h_{ii} w_i^2$$

Hessian Matrix H is difficult to compute.

* h_{ii} is non-negative

LeCun, Y., Denker, J., & Solla, S. (1989). Optimal brain damage. Advances in neural information processing systems, 2.

From Weights to Neurons

Selection of Neurons to Prune

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

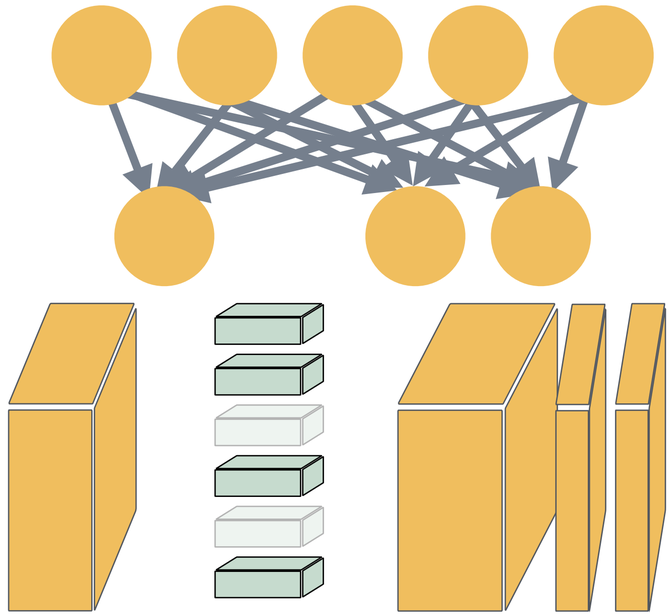
- When removing **neurons** from a neural network model
  - The **less useful** the neurons being removed are, the better the performance of pruned neural network is.

Selection of Neurons to Prune

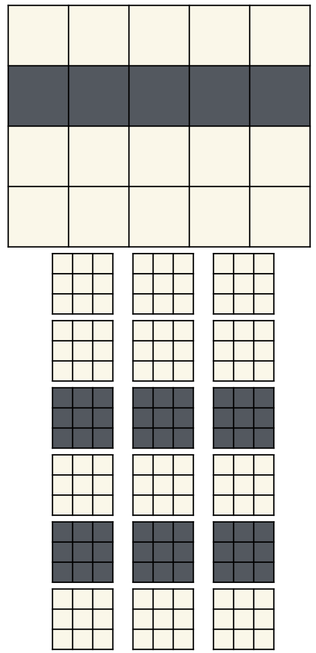
- When removing **neurons** from a neural network model
  - The **less useful** the neurons being removed are, the better the performance of the pruned neural network is.
- Neuron pruning is coarse-grained weight pruning

Neuron Pruning in Linear Layer

Channel Pruning in Convolution Layer



Weight Matrix



Selection of Neurons to Prune

Percentage-of-Zero-based

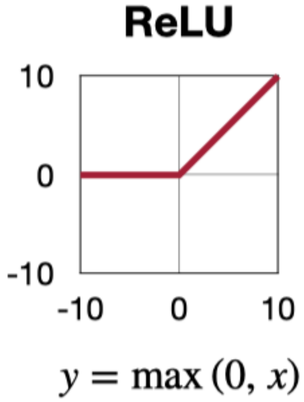
Regression-based

Selection of Neurons to Prune

Percentage-of-Zero-based

Regression-based

Percentage-of-Zero-Based Pruning



- ReLU activation will generate zeros in the output activation

Output Activations

Width=4, Height=4, Channel = 3, Batch = 2

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

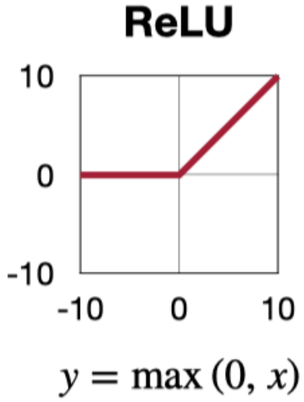
0.5	**0**	0.2	0.1
0	0.2	1.2	**0**
1.2	**0**	0.2	0.3
0.2	0.4	**0**	**0**

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**

Hu, H. (2016). Network trimming: A data-driven neuron pruning approach towards efficient deep architectures. arXiv preprint arXiv:1607.03250.

Percentage-of-Zero-Based Pruning



- ReLU activation will generate zeros in the output activation
- The Average Percentage of Zero activations (APoZ) can be exploited to measure the importance of the neurons

Output Activations

Width=4, Height=4, Channel = 3, Batch = 2

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)

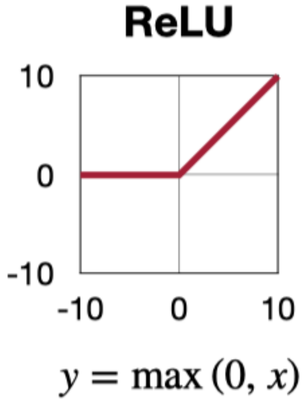
Channel 0: $= \frac{5+6}{2 \cdot 4 \cdot 4} = \frac{11}{32}$

Channel 1: $= \frac{5+7}{2 \cdot 4 \cdot 4} = \frac{12}{32}$

Channel 2: $= \frac{6+8}{2 \cdot 4 \cdot 4} = \frac{14}{32}$

Hu, H. (2016). Network trimming: A data-driven neuron pruning approach towards efficient deep architectures. arXiv preprint arXiv:1607.03250.

Percentage-of-Zero-Based Pruning



- ReLU activation will generate zeros in the output activation
- The Average Percentage of Zero activations (APoZ) can be exploited to measure the importance of the neurons
- The smaller APoZ is, the more importance the neuron has

Output Activations

Width=4, Height=4, Channel = 3, Batch = 2

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)

$$= \frac{5+6}{2 \cdot 4 \cdot 4} = \frac{11}{32}$$

Channel 0

$$= \frac{5+7}{2 \cdot 4 \cdot 4} = \frac{12}{32}$$

Channel 1

$$= \frac{6+8}{2 \cdot 4 \cdot 4} = \frac{14}{32}$$

Channel 2

Hu, H. (2016). Network trimming: A data-driven neuron pruning approach towards efficient deep architectures. arXiv preprint arXiv:1607.03250.

Selection of Neurons to Prune

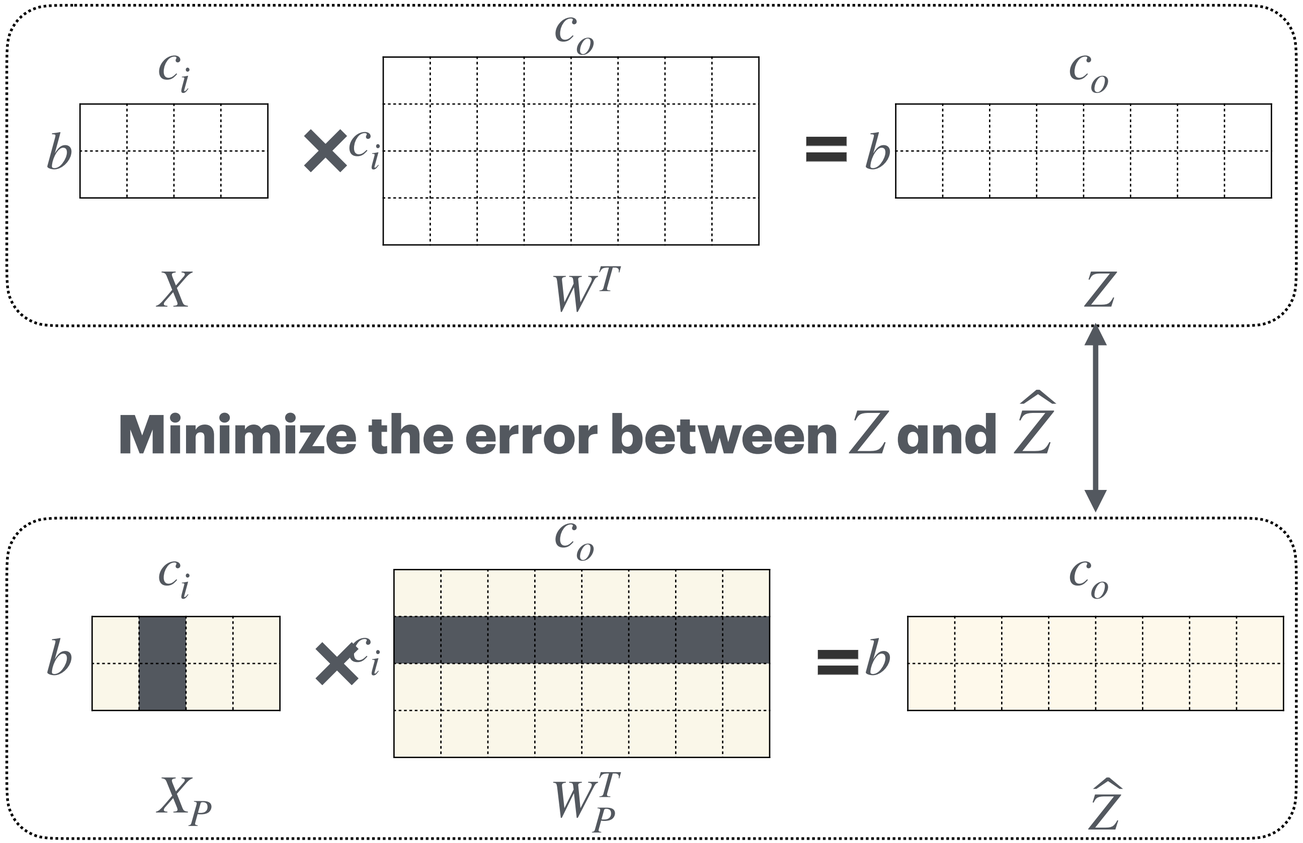
Percentage-of-Zero-based

Regression-based

Regression-Based Pruning

Minimize reconstruction error of the corresponding layer's outputs

- Instead of considering the pruning error of the objective function $L(X; W)$, regression-based pruning minimizes the reconstruction error of the **corresponding layer**'s outputs.



He, Y., Zhang, X., & Sun, J. (2017). Channel pruning for accelerating very deep neural networks. In Proceedings of the IEEE international conference on computer vision (pp. 1389-1397).

Regression-Based Pruning

Minimize reconstruction error of the corresponding layer's outputs

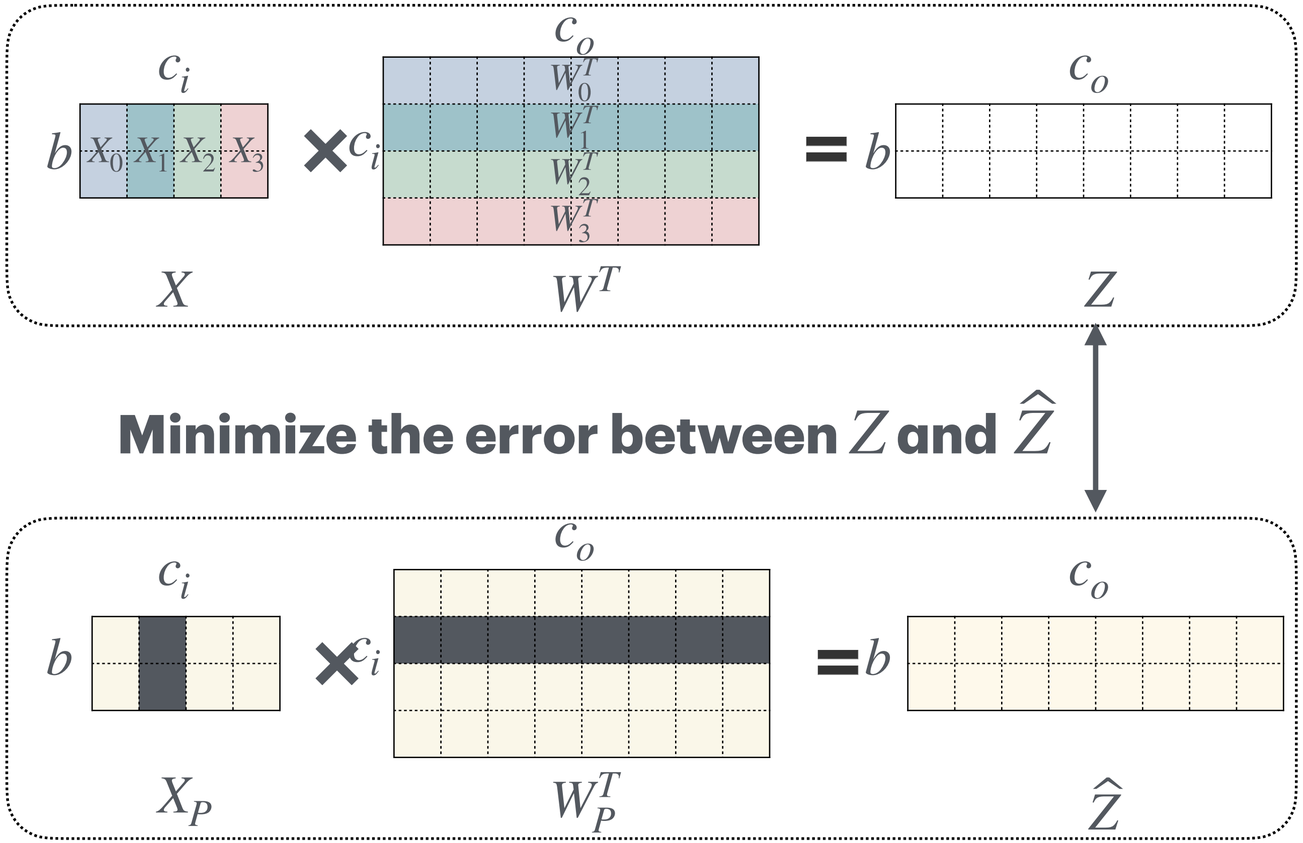
- Let $$Z = XW^T = \sum_{c=0}^{c_i-1} X_c W_c^T$$

- The problem can be formulated as

$$\arg\min_{W,\beta} \left\| Z - \hat{Z} \right\|_F^2 = \left\| Z - \sum_{c=0}^{c_i-1} \beta_c X_c W_c^T \right\|_F^2$$

$$\text{subject to } \left\| \beta \right\|_0 \leq N_c$$

- β is coefficient vector of length c_i for channel selection. $\beta_c = 0$ means channel c is pruned.
- N_c is the number of nonzero channels.
- Solve the problem by
  - Fix W, solve β for channel selection
  - Fix β, solve W to minimize reconstruction error



He, Y., Zhang, X., & Sun, J. (2017). Channel pruning for accelerating very deep neural networks. In Proceedings of the IEEE international conference on computer vision (pp. 1389-1397).

Pruning Demo

The reference for your Lab 2.

References

- Pruning and Sparsity [MIT 6.5940]
- Cai, H., Gan, C., Wang, T., Zhang, Z., & Han, S. (2019). Once-for-all: Train one network and specialize it for efficient deployment. arXiv preprint arXiv:1908.09791.
- Horowitz, M. (2014, February). 1.1 computing's energy problem (and what we can do about it). In 2014 IEEE international solid-state circuits conference digest of technical papers (ISSCC) (pp. 10-14). IEEE.
- Drachman, D. A. (2005). Do we have brain to spare?. Neurology, 64(12), 2004-2005.
- Han, S., Pool, J., Tran, J., & Dally, W. (2015). Learning both weights and connections for efficient neural network. Advances in neural information processing systems, 28.
- Han, S. (2017). Efficient methods and hardware for deep learning (Doctoral dissertation, Stanford University).
- LeCun, Y., Denker, J., & Solla, S. (1989). Optimal brain damage. Advances in neural information processing systems, 2.
- Han, S., Liu, X., Mao, H., Pu, J., Pedram, A., Horowitz, M. A., & Dally, W. J. (2016). EIE: Efficient inference engine on compressed deep neural network. ACM SIGARCH Computer Architecture News, 44(3), 243-254.
- Han, S., Kang, J., Mao, H., Hu, Y., Li, X., Li, Y., ... & Dally, W. B. J. (2017, February). Ese: Efficient speech recognition engine with sparse lstm on fpga. In Proceedings of the 2017 ACM/SIGDA International Symposium on Field-Programmable Gate Arrays (pp. 75-84).
- Zhang, Z., Wang, H., Han, S., & Dally, W. J. (2020, February). Sparch: Efficient architecture for sparse matrix multiplication. In 2020 IEEE International Symposium on High Performance Computer Architecture (HPCA) (pp. 261-274). IEEE.
- Wang, H., Zhang, Z., & Han, S. (2021, February). Spatten: Efficient sparse attention architecture with cascade token and head pruning. In 2021 IEEE International Symposium on High-Performance Computer Architecture (HPCA) (pp. 97-110). IEEE.
- Mao, H., Han, S., Pool, J., Li, W., Liu, X., Wang, Y., & Dally, W. J. (2017). Exploring the granularity of sparsity in convolutional neural networks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops (pp. 13-20).
- He, Y., Lin, J., Liu, Z., Wang, H., Li, L. J., & Han, S. (2018). AMC: Automl for model compression and acceleration on mobile devices. In Proceedings of the European conference on computer vision (ECCV) (pp. 784-800).
- Wen, W., Wu, C., Wang, Y., Chen, Y., & Li, H. (2016). Learning structured sparsity in deep neural networks. Advances in neural information processing systems, 29.
- Liu, Z., Li, J., Shen, Z., Huang, G., Yan, S., & Zhang, C. (2017). Learning efficient convolutional networks through network slimming. In Proceedings of the IEEE international conference on computer vision (pp. 2736-2744).
- Hu, H. (2016). Network trimming: A data-driven neuron pruning approach towards efficient deep architectures. arXiv preprint arXiv:1607.03250.
- He, Y., Zhang, X., & Sun, J. (2017). Channel pruning for accelerating very deep neural networks. In Proceedings of the IEEE international conference on computer vision (pp. 1389-1397).