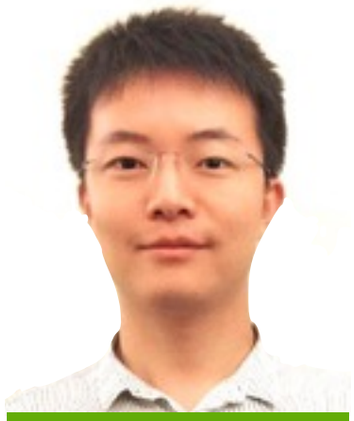
Deep Compression:

——Compressing Deep Neural Networks with Pruning, Trained Quantization and Huffman Coding

Song Han, Huizi Mao, Bill Dally
Stanford University
May 4, 2016

Intro



Song Han

- I'm 4th year PhD with Prof. Bill Dally at Stanford.
- **"Deep Compression"** and **"EIE: Efficient Inference Engine"** covered by TheNextPlatform & O'Reilly & TechEmergence & HackerNews & Embedded-Vision.

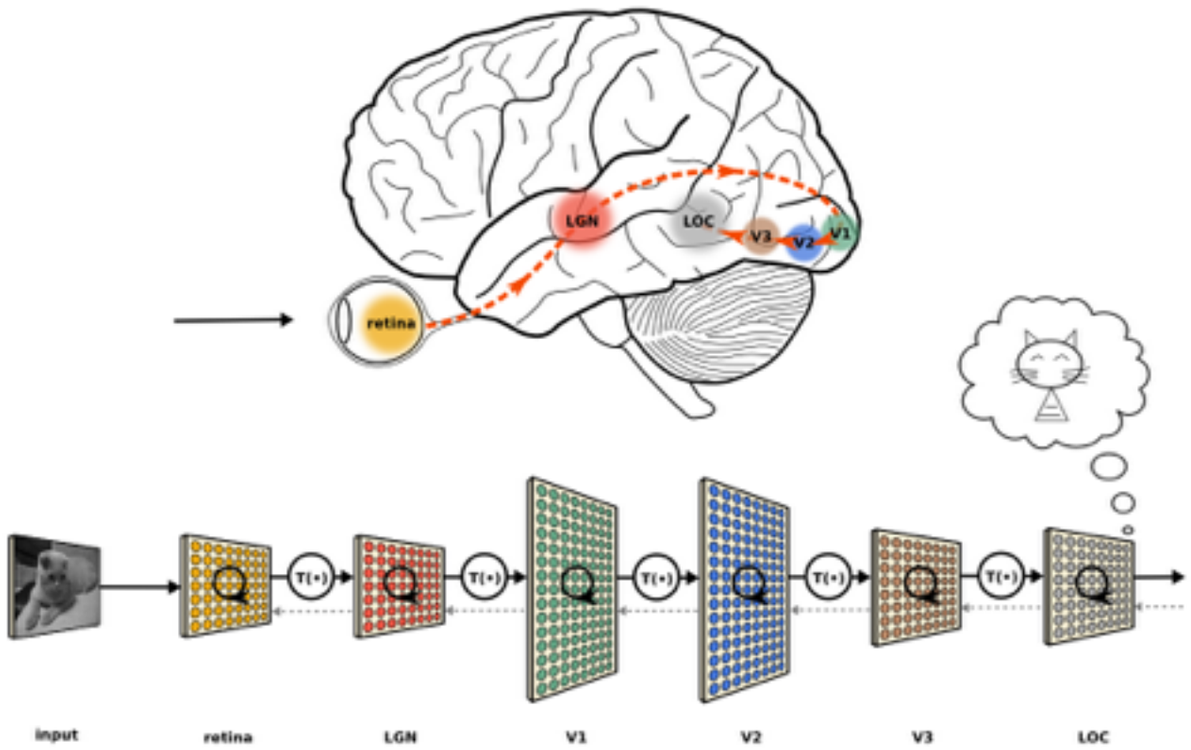
Huizi Mao

- Undergrad student at Tsinghua University.
- Joining us at Stanford this fall.

Bill Dally

- Professor at Stanford University and former chair of CS department, leads the CVA Group.
- Chief Scientist of **NVIDIA**.

Deep Learning： Next Wave of AI



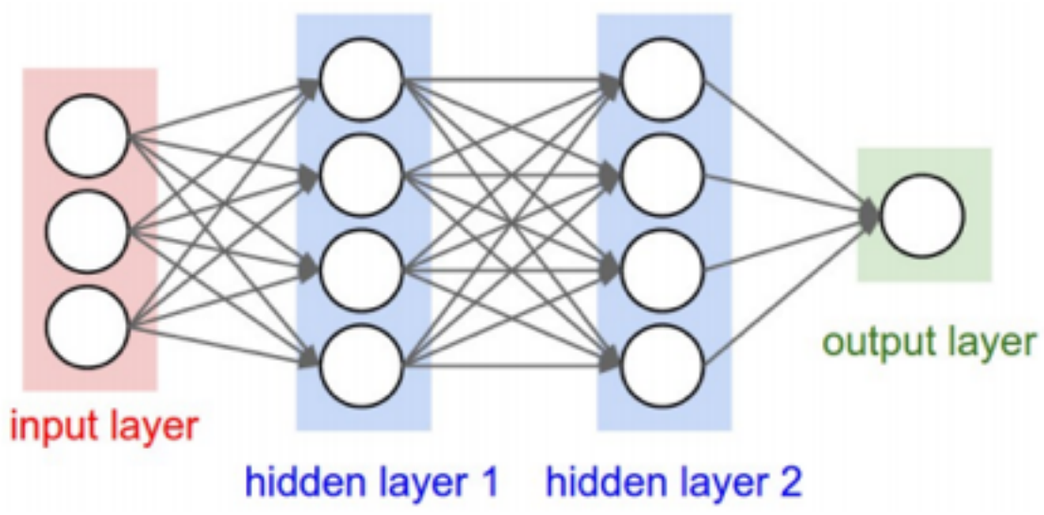
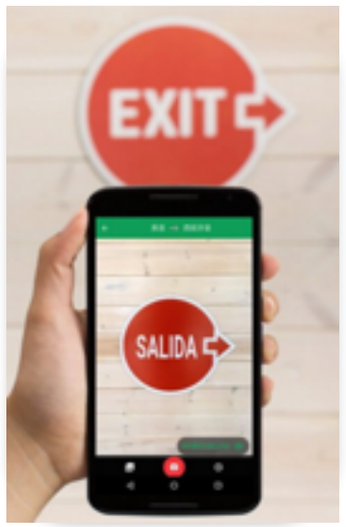


Image Recognition

Speech Recognition

Natural Language Processing

Deep Learning on Mobile



Phones

Drones

Robots

Glasses

Self Driving Cars

where to compute?

DNN on the Cloud…

Network Delay

Power Budget

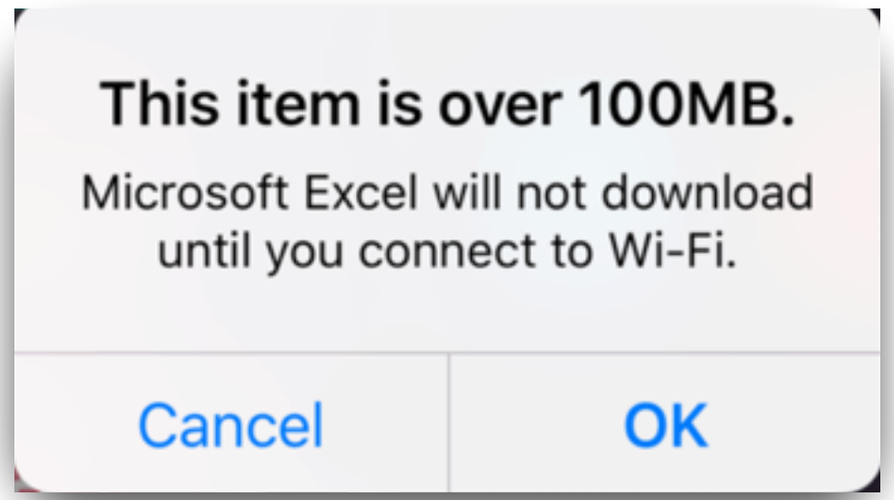
User Privacy

Intelligent but Inefficient

What if Run Deep Learning Locally on Mobile?

Model Size!

App developers suffers from the model size



What if Run Deep Learning Locally on Mobile?

Model Size!

Hardware engineer suffers from the model size (embedded system, limited resource)

Operation	Energy [pJ]	Relative Cost
32 bit int ADD	0.1	1
32 bit float ADD	0.9	9
32 bit Register File	1	10
32 bit int MULT	3.1	31
32 bit float MULT	3.7	37
32 bit SRAM Cache	5	50
32 bit DRAM Memory	**640**	**6400**

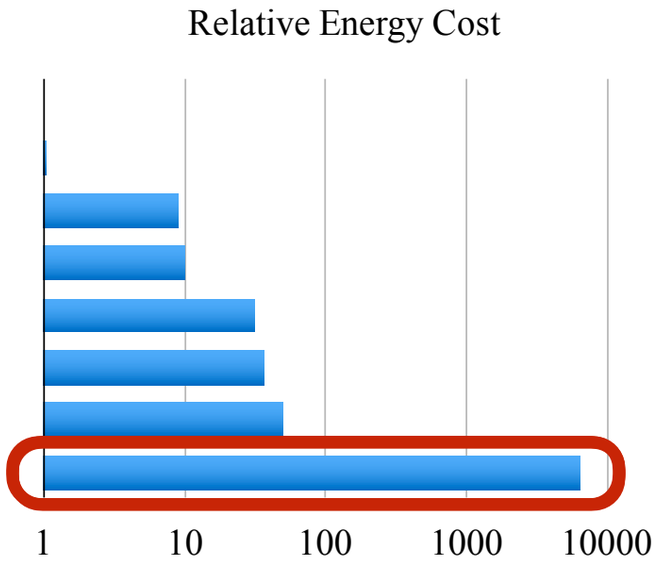
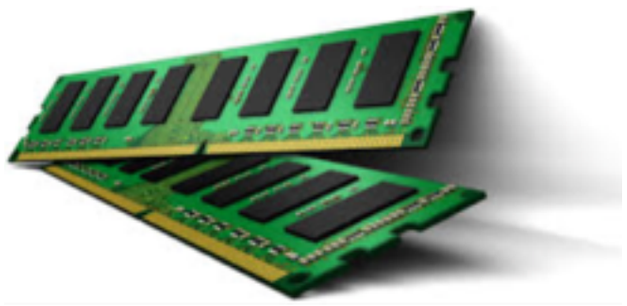
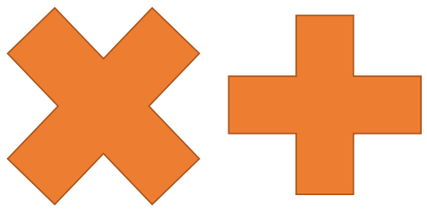


Figure 1: Energy table for 45nm CMOS process. Memory access is 2 orders of magnitude more energy expensive than arithmetic operations.

1  **= 100** 

Problem: DNN Model Size

Solution: Deep Compression

Smaller Size

Compress DNN model size by 10x-50x

Accuracy

No loss of accuracy / Improved accuracy

Speedup

Make inference faster by specialized HW

Deep Compression Overview

- AlexNet: 35×, 240MB => 6.9MB
- VGG16: 49×, 552MB => 11.3MB
- GoogLeNet: 10x, 28MB => 2.8MB
- SqueezeNet: 10x, 4.8MB => 0.47MB
- No loss of accuracy on ImageNet12
- Weights fits on-chip SRAM cache, taking 120x less energy than DRAM memory

Deep Compression Pipeline

- **Network Pruning**:
  Less Number of Weights

- **Weight Sharing**:
  Reduce Storage for Each Remaining Weight

- **Huffman Coding**:
  Entropy of the Total Remaining Weights

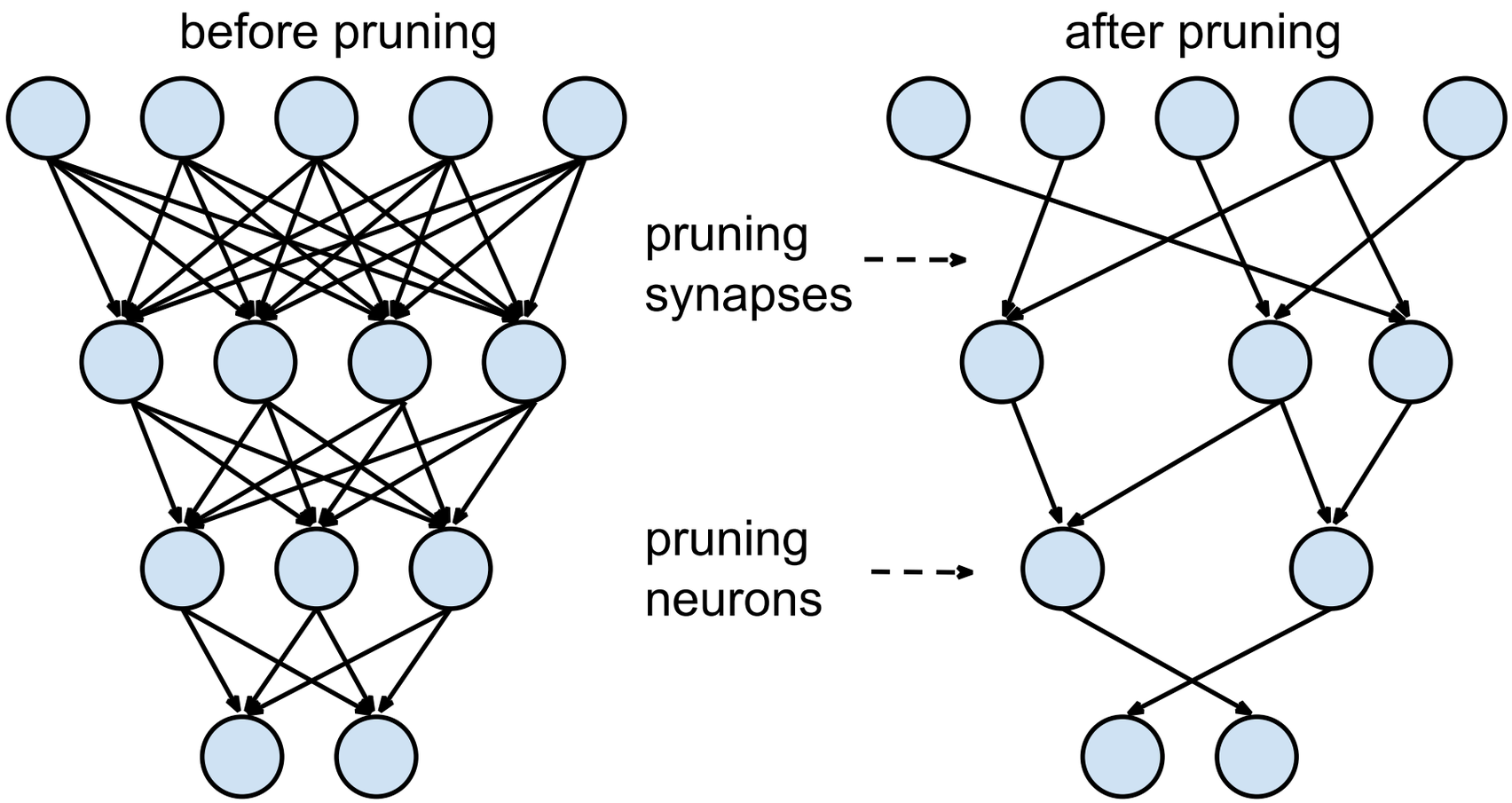
Deep Compression Pipeline

- **Network Pruning**:
  Less Number of Weights

- **Weight Sharing**:
  Reduce Storage for Each Remaining Weight

- **Huffman Coding**:
  Entropy of the Total Remaining Weights

1. Pruning



[1] LeCun et al. Optimal Brain Damage NIPS'90
[2] Hassibi, et al. Second order derivatives for network pruning: Optimal brain surgeon. NIPS'93
[3] Han et al. Learning both Weights and Connections for Efficient Neural Networks, NIPS'15

Pruning: Motivation

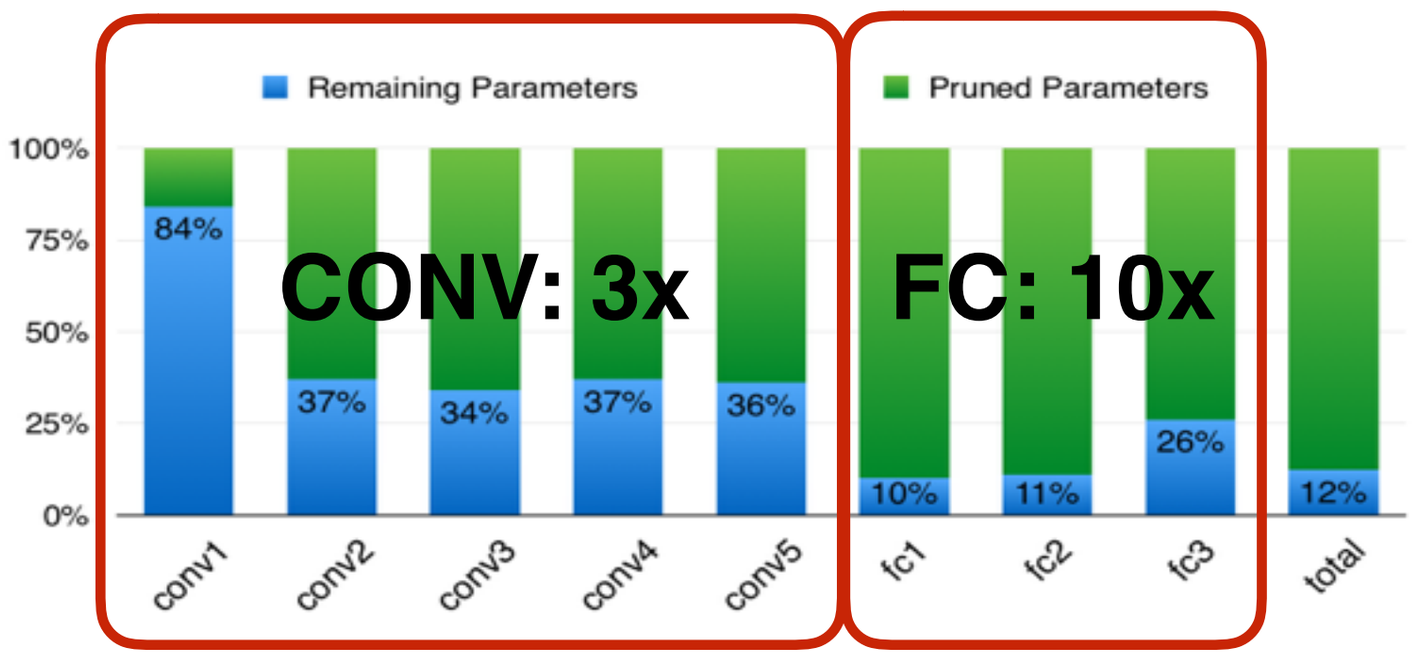
Age	Number of Connections	Stage
at birth	50 Trillion	newly formed
1 year old	1000 Trillion	peak
10 year old	500 Trillion	pruned and stabilized

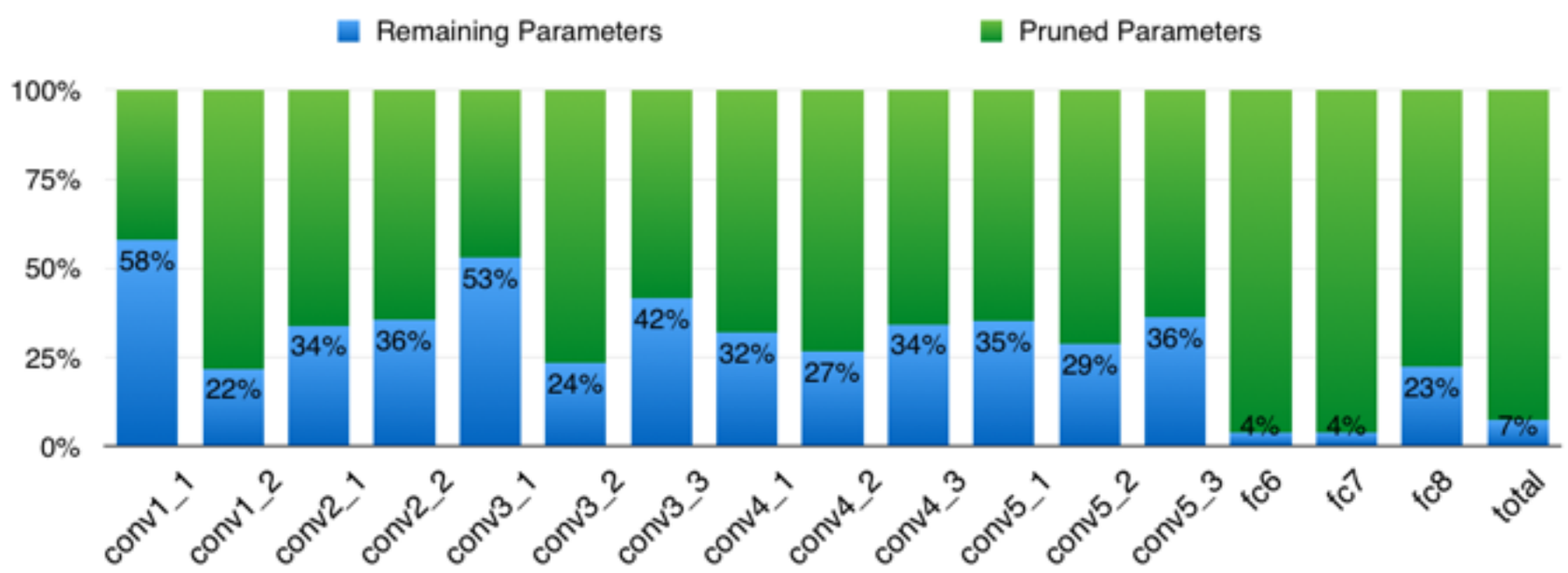
Table 1: The synapses pruning mechanism in human brain development

- Trillion of synapses are generated in the human brain during the first few months of birth.
- **1 year old**, peaked at **1000 trillion**
- Pruning begins to occur.
- **10 years old**, a child has nearly **500 trillion** synapses
- This 'pruning' mechanism removes redundant connections in the brain.

[1] Christopher A Walsh. Peter Huttenlocher (1931-2013). Nature, 502(7470):172–172, 2013.

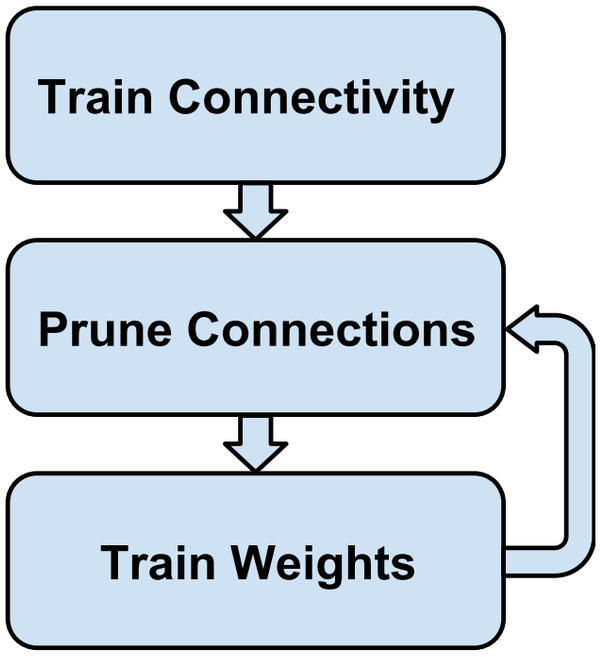
AlexNet & VGGNet

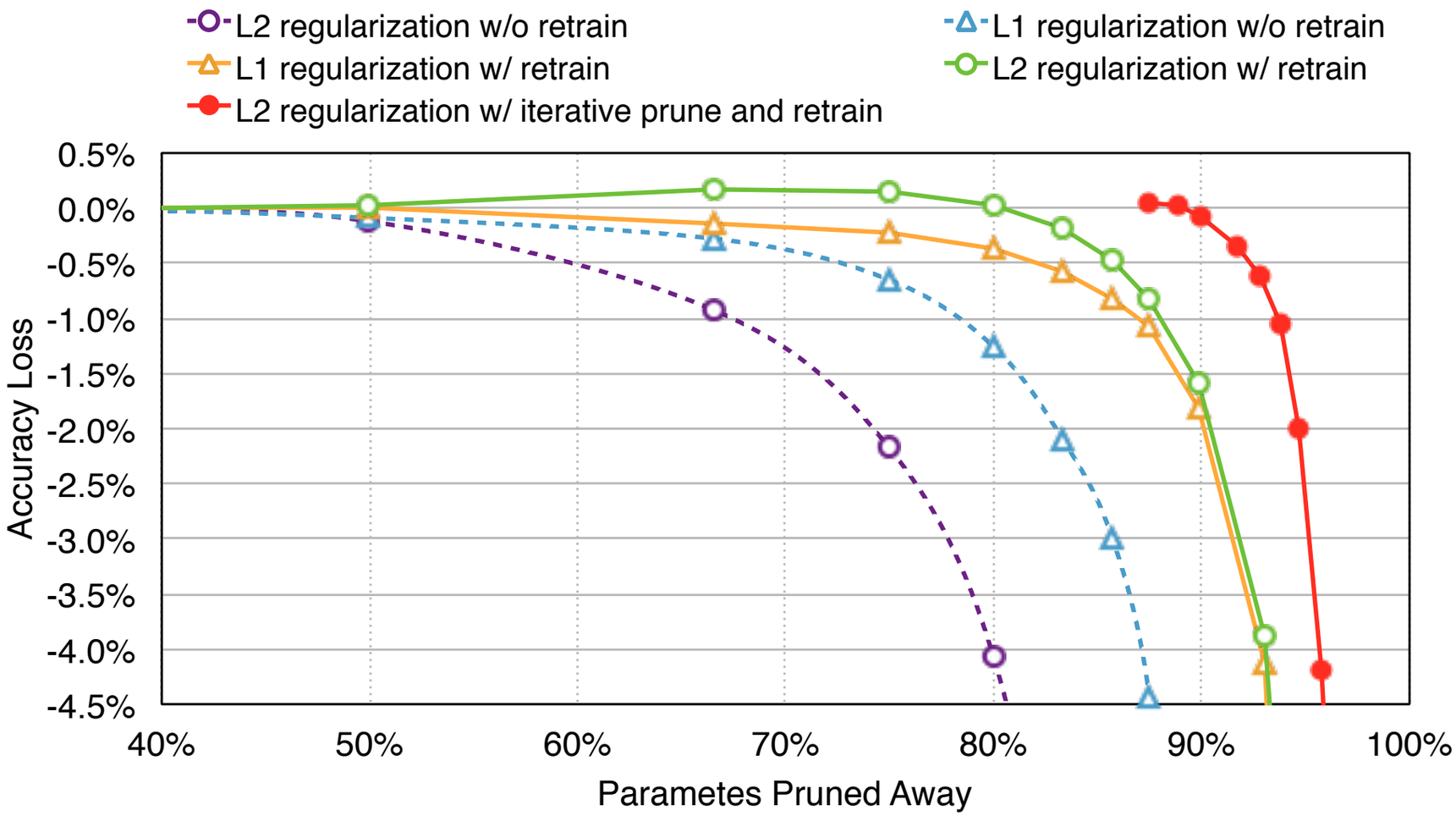




Han et al. Learning both Weights and Connections for Efficient Neural Networks, NIPS 2015

Retrain to Recover Accuracy





Han et al. Learning both Weights and Connections for Efficient Neural Networks, NIPS 2015

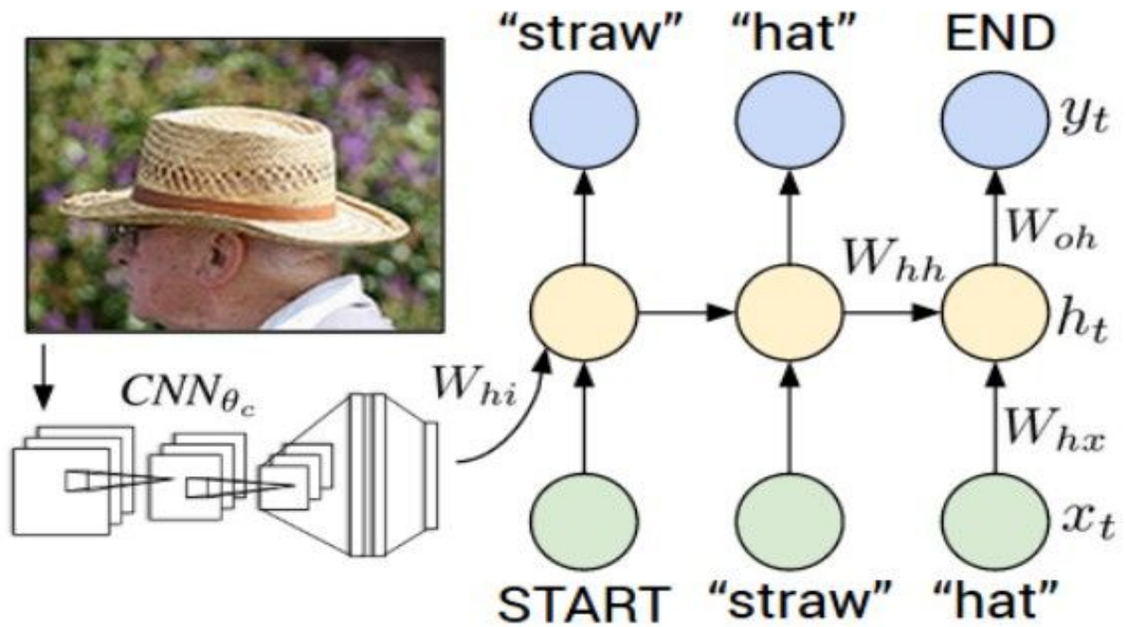
Pruning: Result

Network	Top-1 Error	Top-5 Error	Parameters	Compression Rate
LeNet-300-100 Ref	1.64%	-	267K	
LeNet-300-100 Pruned	1.59%	-	**22K**	**12**×
LeNet-5 Ref	0.80%	-	431K	
LeNet-5 Pruned	0.77%	-	**36K**	**12**×
AlexNet Ref	42.78%	19.73%	61M	
AlexNet Pruned	42.77%	19.67%	**6.7M**	**9**×
VGG16 Ref	31.50%	11.32%	138M	
VGG16 Pruned	31.34%	10.88%	**10.3M**	**13**×

Table 1: Network pruning can save 9× to 13× parameters with no drop in predictive performance

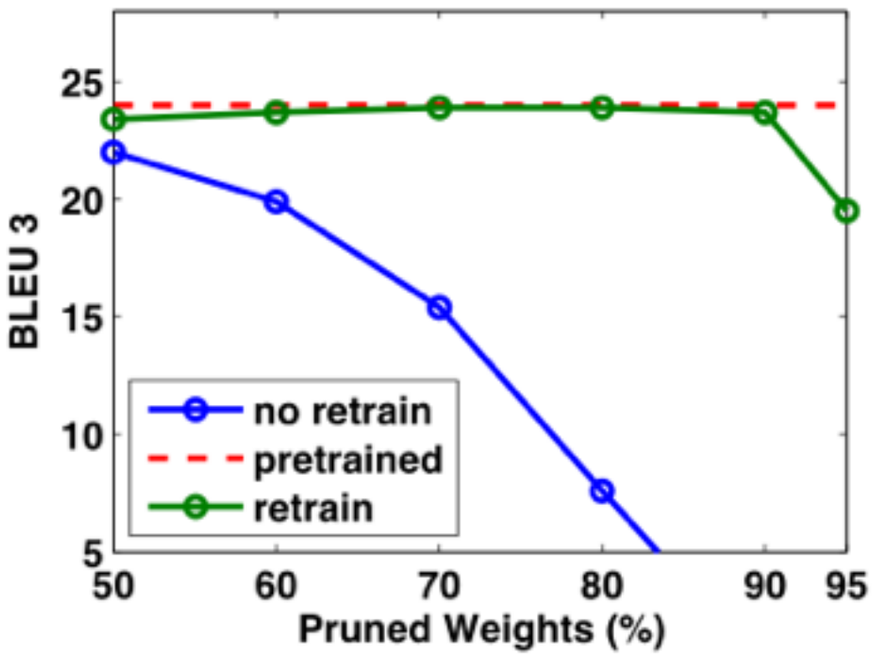
Han et al. Learning both Weights and Connections for Efficient Neural Networks, NIPS 2015

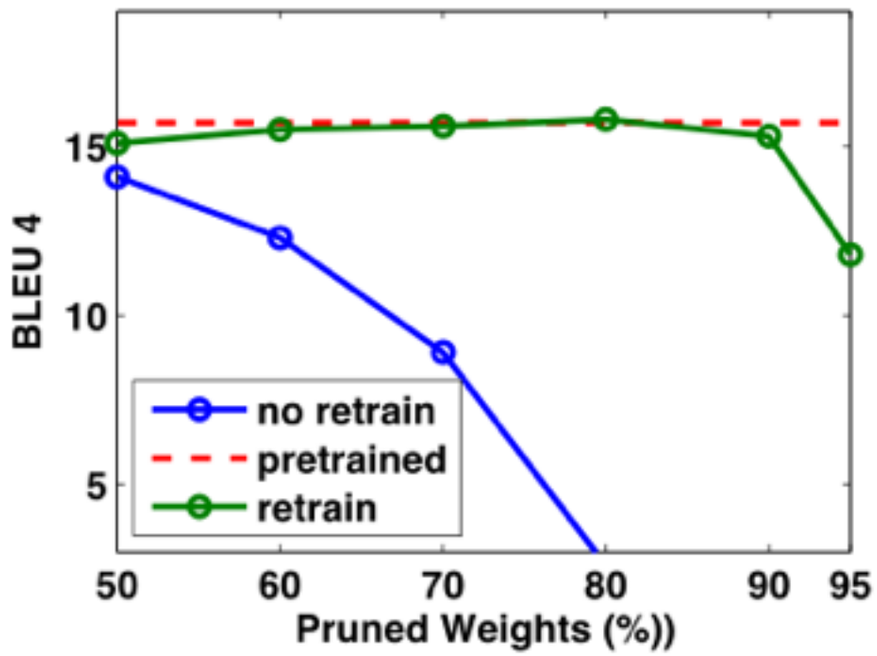
Pruning RNN and LSTM



Karpathy, et al, "Deep Visual-Semantic Alignments for Generating Image Descriptions"

- Pruning away 90% parameters in NeuralTalk doesn't hurt BLUE score with proper retraining





Pruning NeuralTalk and LSTM

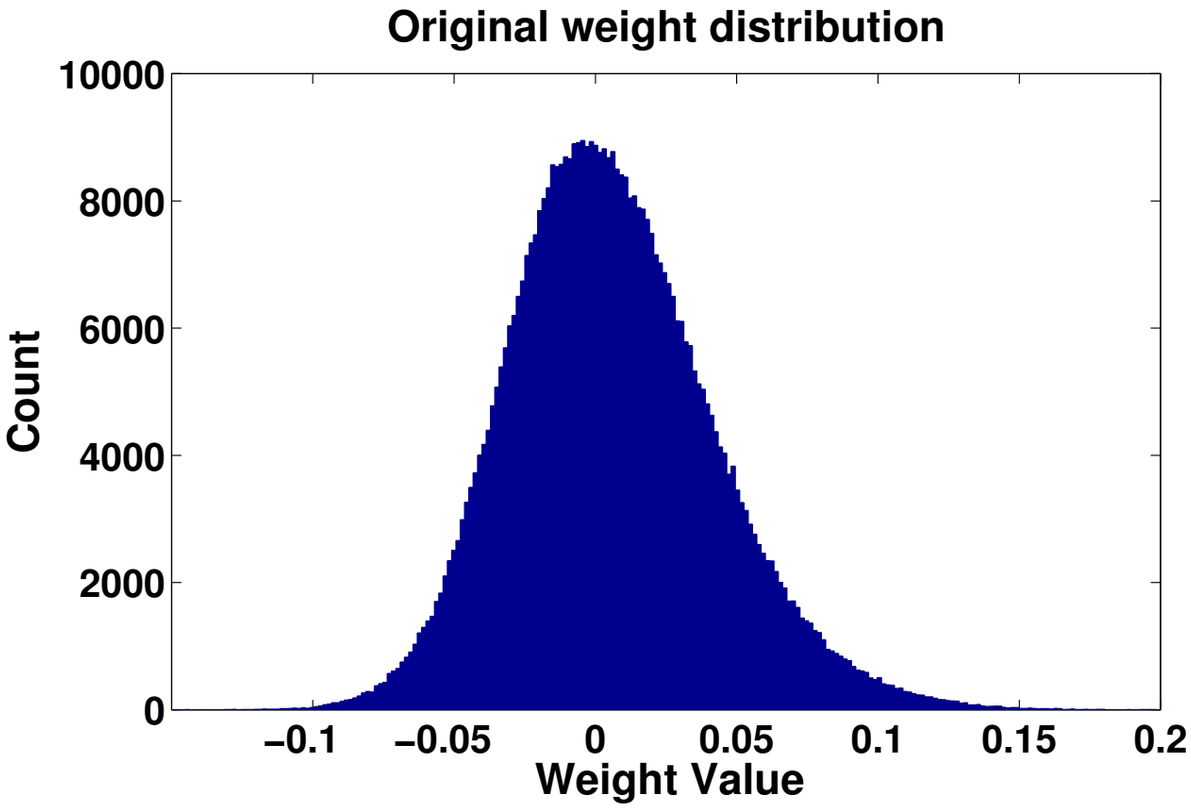
- **Original**: a basketball player in a white uniform is playing with a ball
- **Pruned 90%**: a basketball player in a white uniform is playing with a basketball

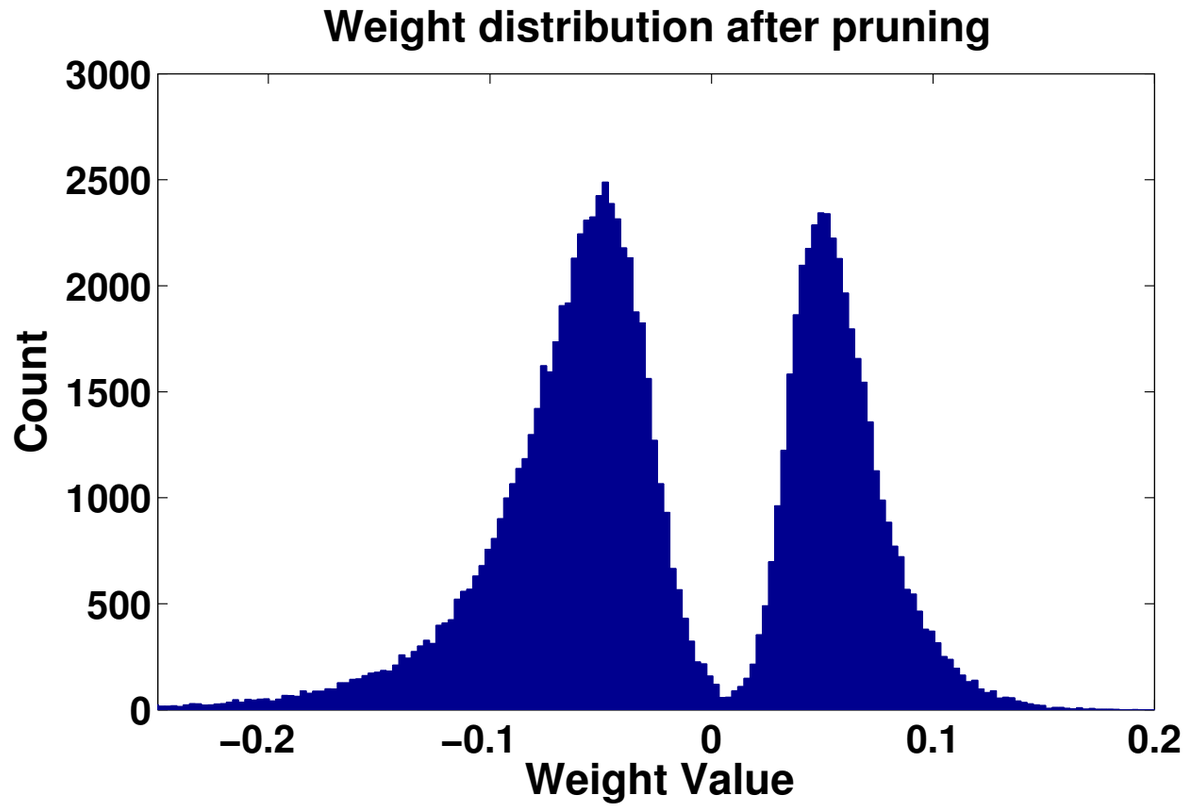
- **Original** : a brown dog is running through a grassy field
- **Pruned 90%**: a brown dog is running through a grassy area

- **Original** : a man is riding a surfboard on a wave
- **Pruned 90%**: a man in a wetsuit is riding a wave on a beach

- **Original** : 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

Weight Distribution

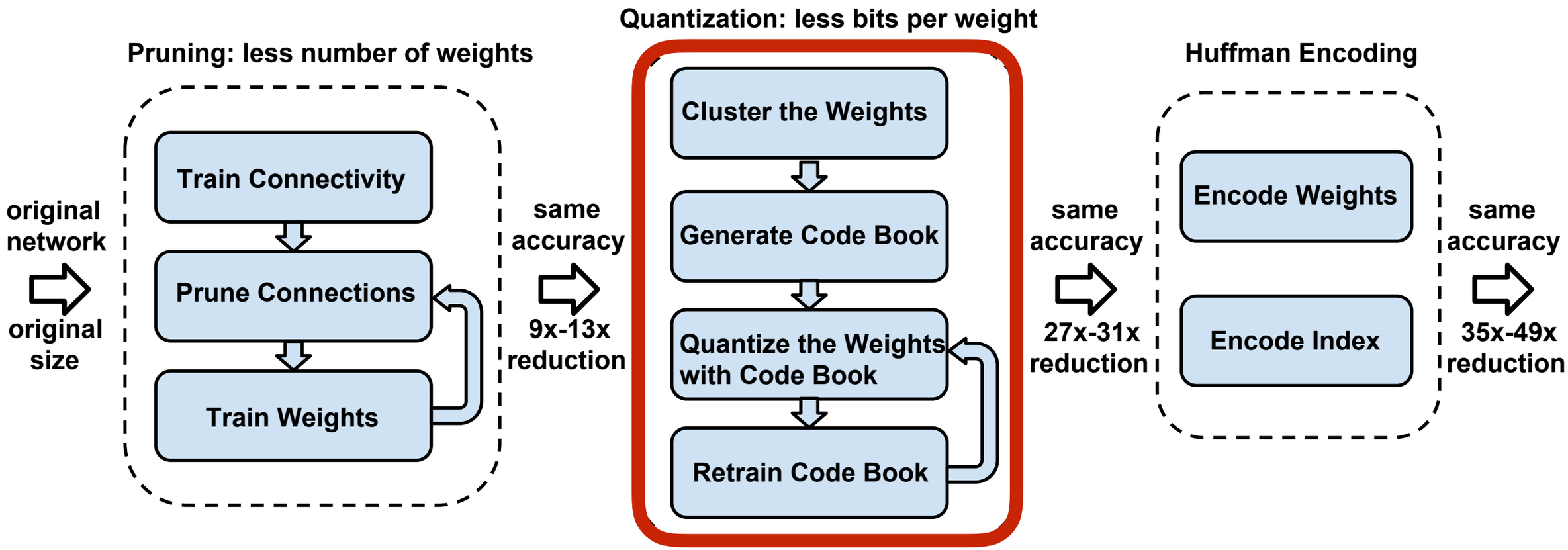




Deep Compression Pipeline

- **Network Pruning**:
  Less Number of Weights
- **Weight Sharing**:
  Reduce Storage for Each Remaining Weight
- **Huffman Coding**:
  Entropy of the Total Remaining Weights

2. Weight Sharing
(Trained Quantization)



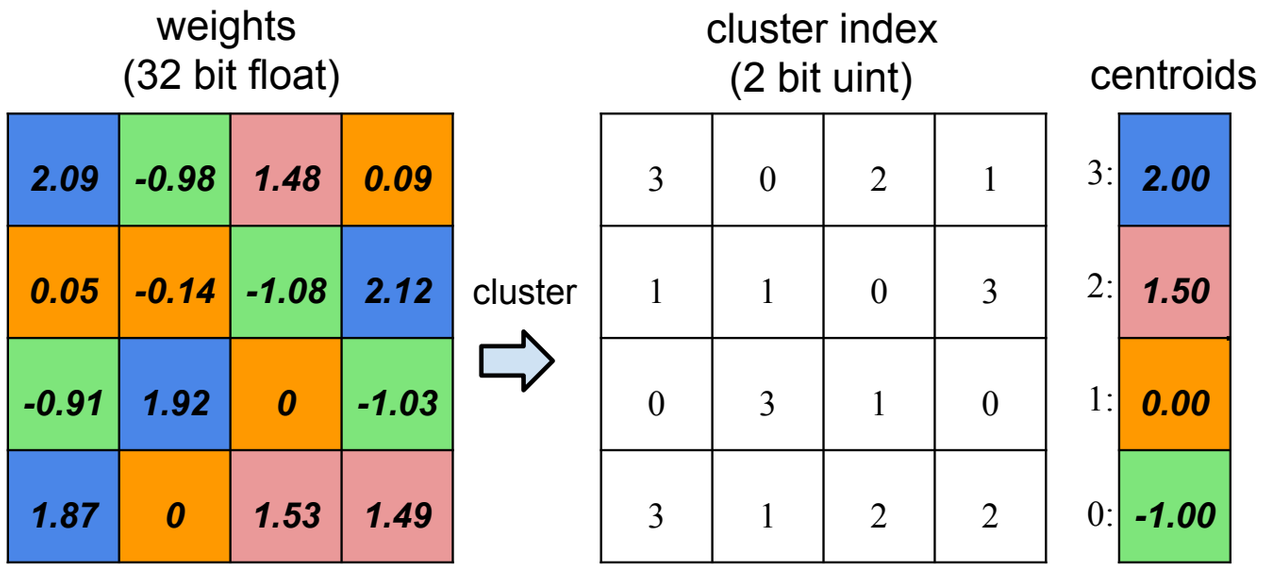
Weight Sharing: Overview

weights
(32 bit float)

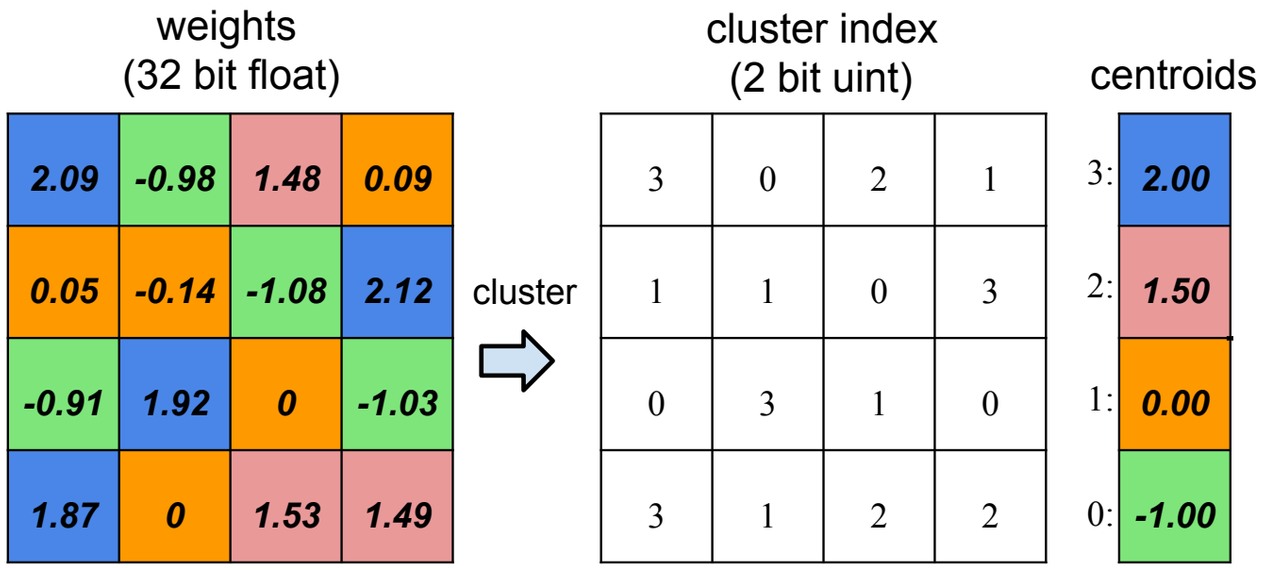
2.09	*-0.98*	*1.48*	*0.09*
0.05	*-0.14*	*-1.08*	*2.12*
-0.91	*1.92*	*0*	*-1.03*
1.87	*0*	*1.53*	*1.49*

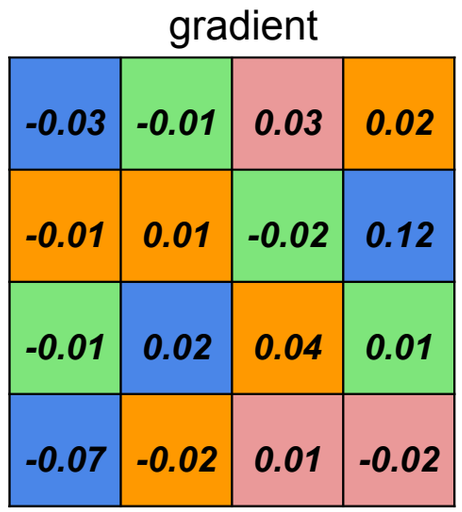
Weight Sharing: Overview

Weight Sharing: Overview

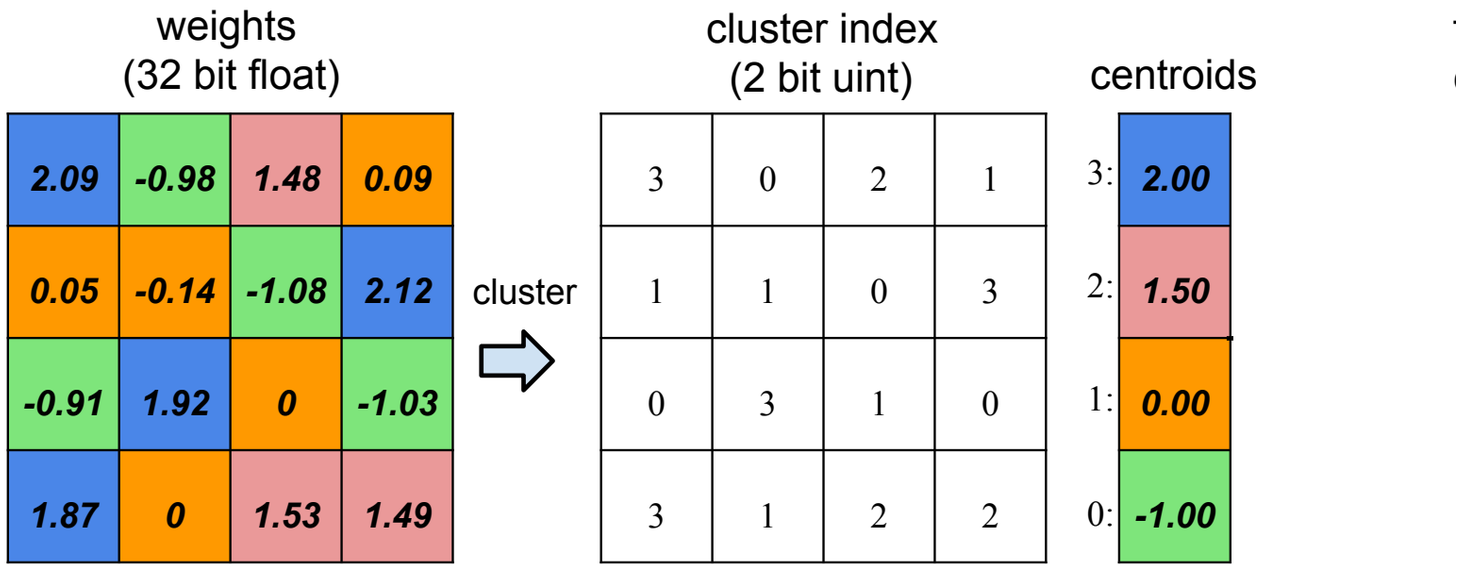


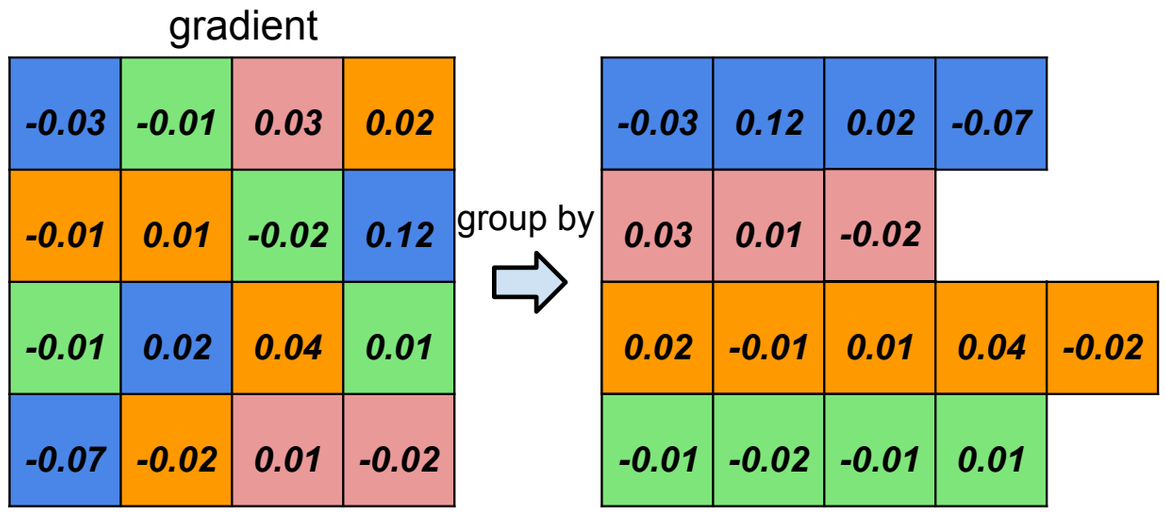
Weight Sharing: Overview



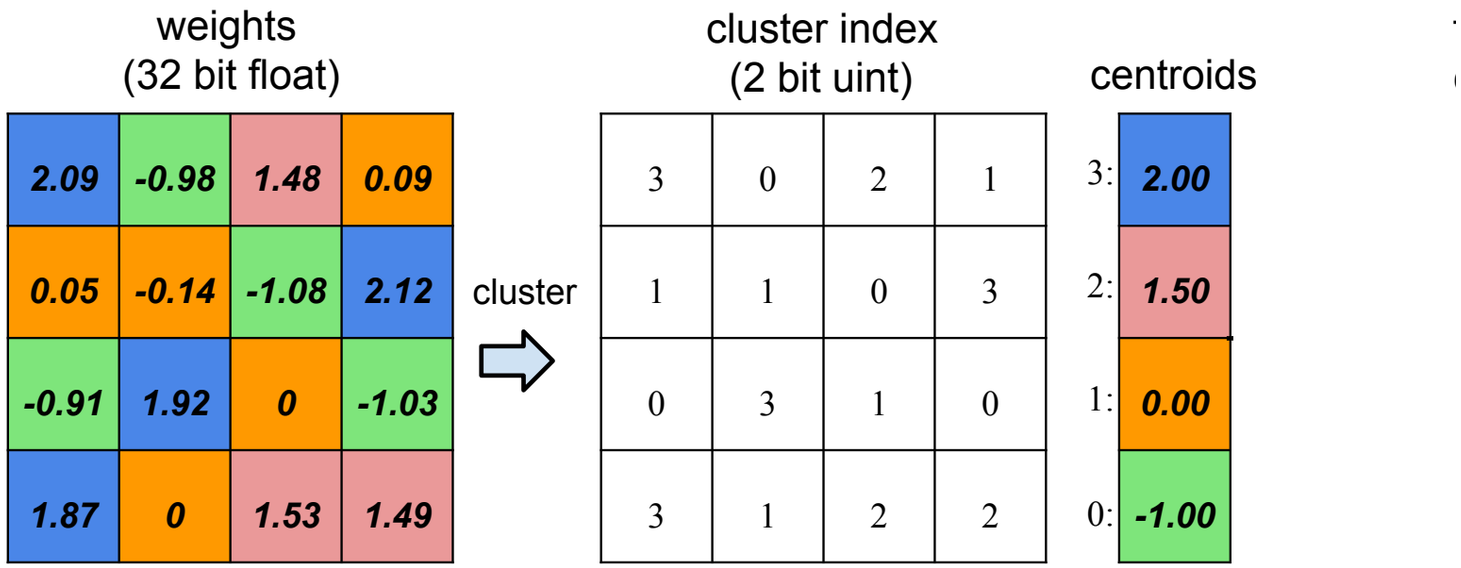


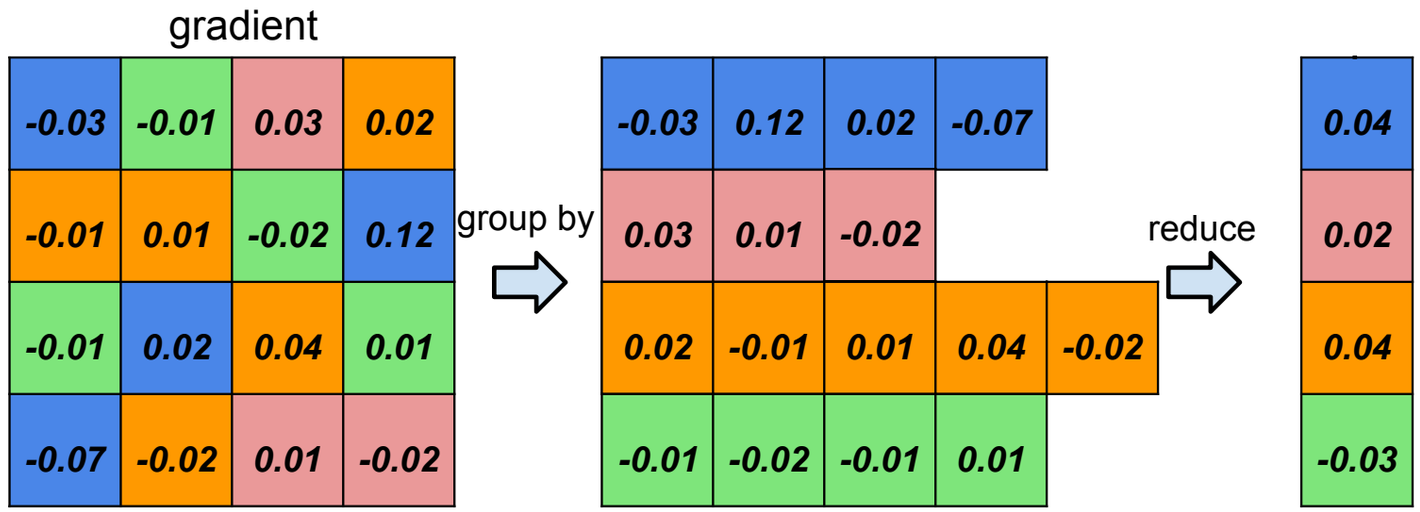
Weight Sharing: Overview



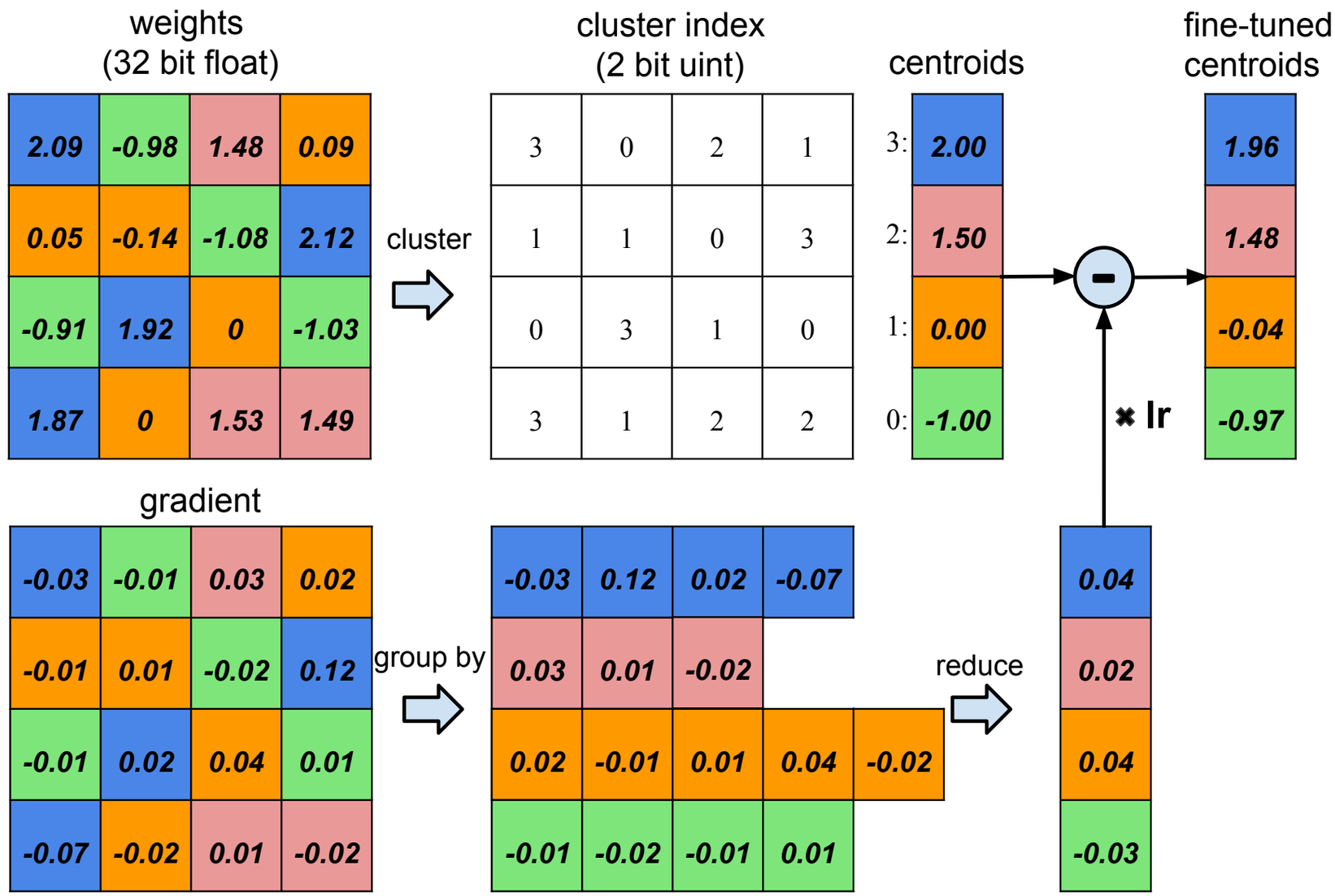


Weight Sharing: Overview

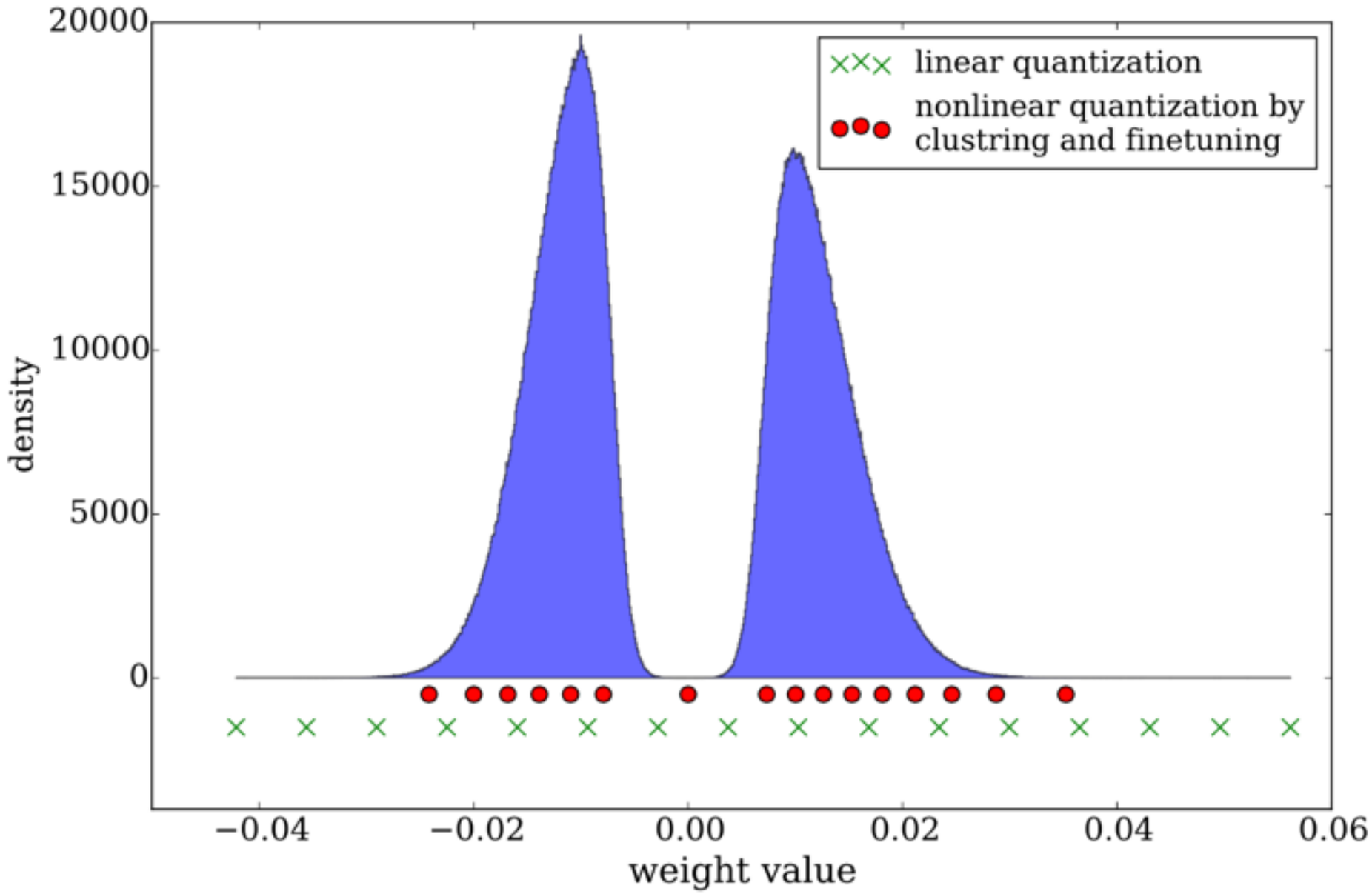




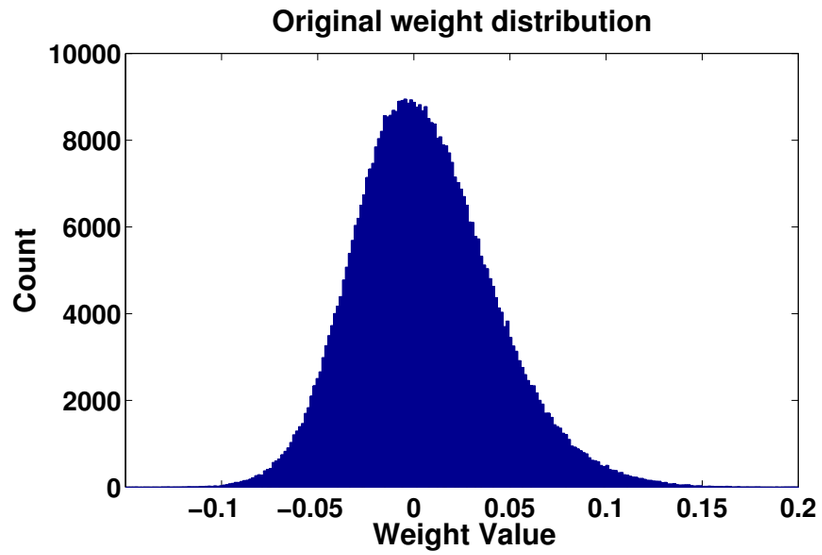
Weight Sharing: Overview

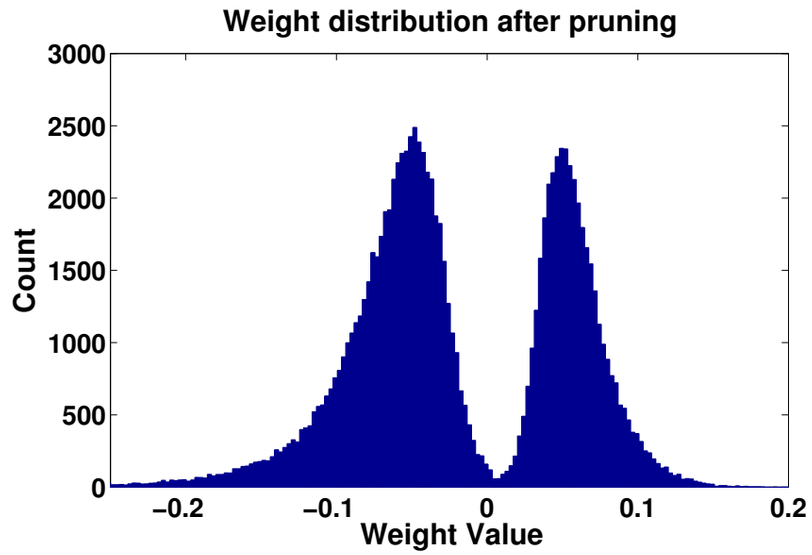


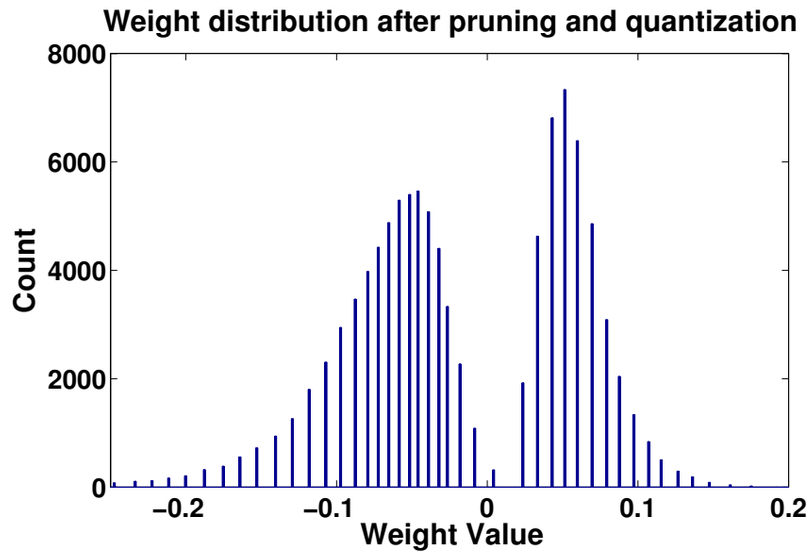
Finetune Centroids



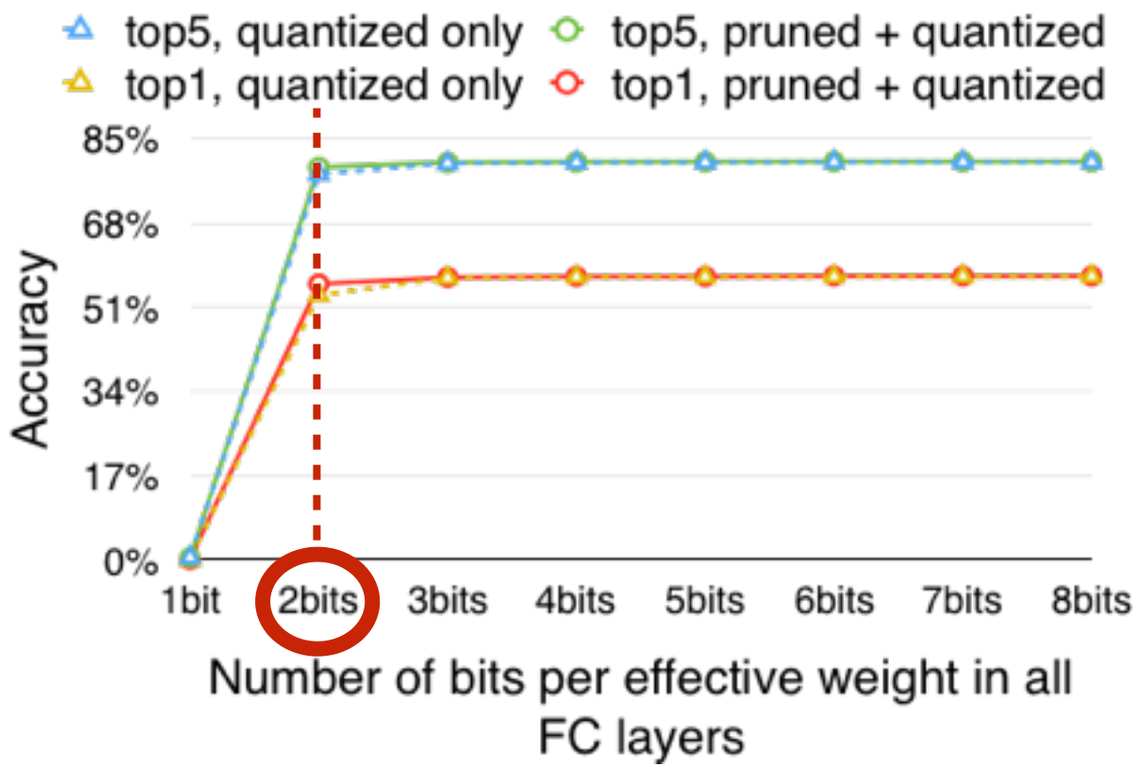
Weight Distribution

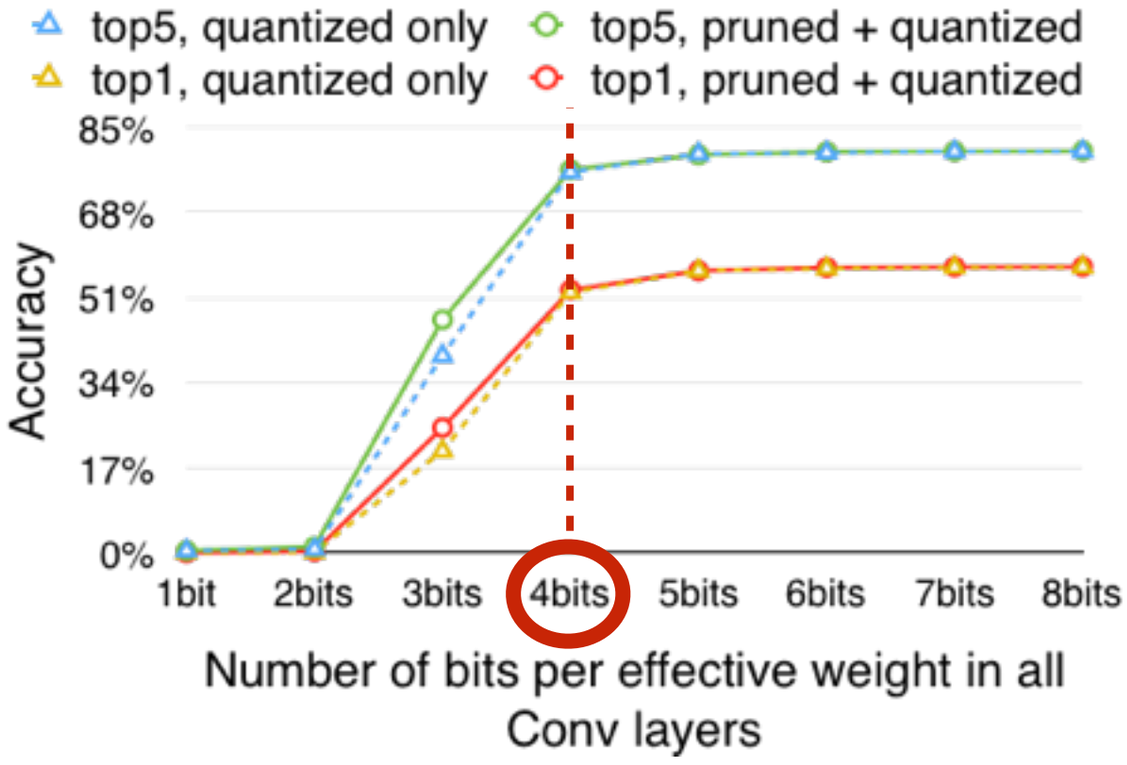






Bits Per Weight

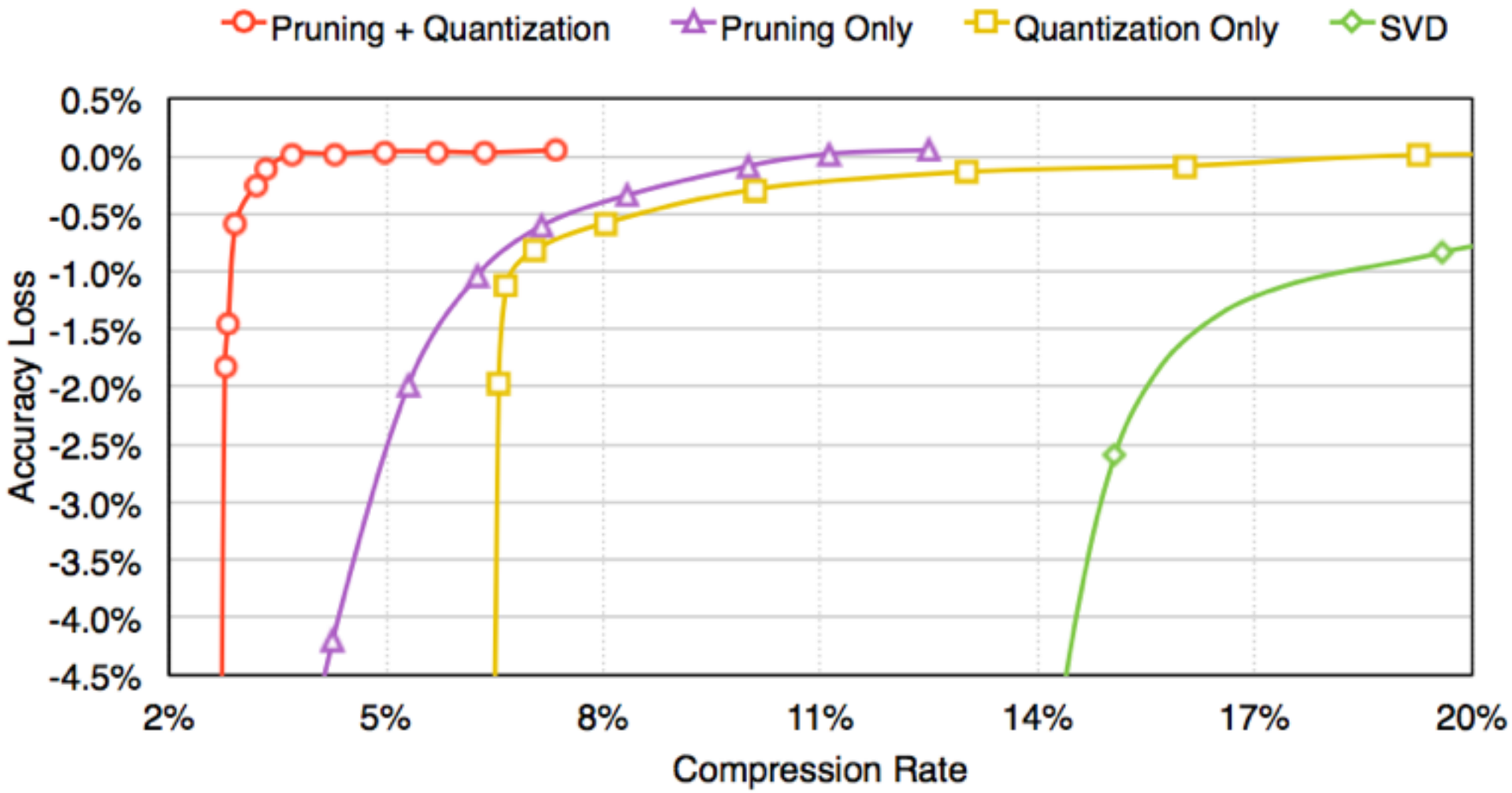




Pruning + Trained Quantization

#CONV bits / #FC bits	Top-1 Error	Top-5 Error	Top-1 Error Increase	Top-5 Error Increase
32bits / 32bits	42.78%	19.73%	-	-
8 bits / 5 bits	42.78%	19.70%	0.00%	-0.03%
8 bits / 4 bits	42.79%	19.73%	0.01%	0.00%
4 bits / 2 bits	44.77%	22.33%	1.99%	2.60%

Pruning + Trained Quantization



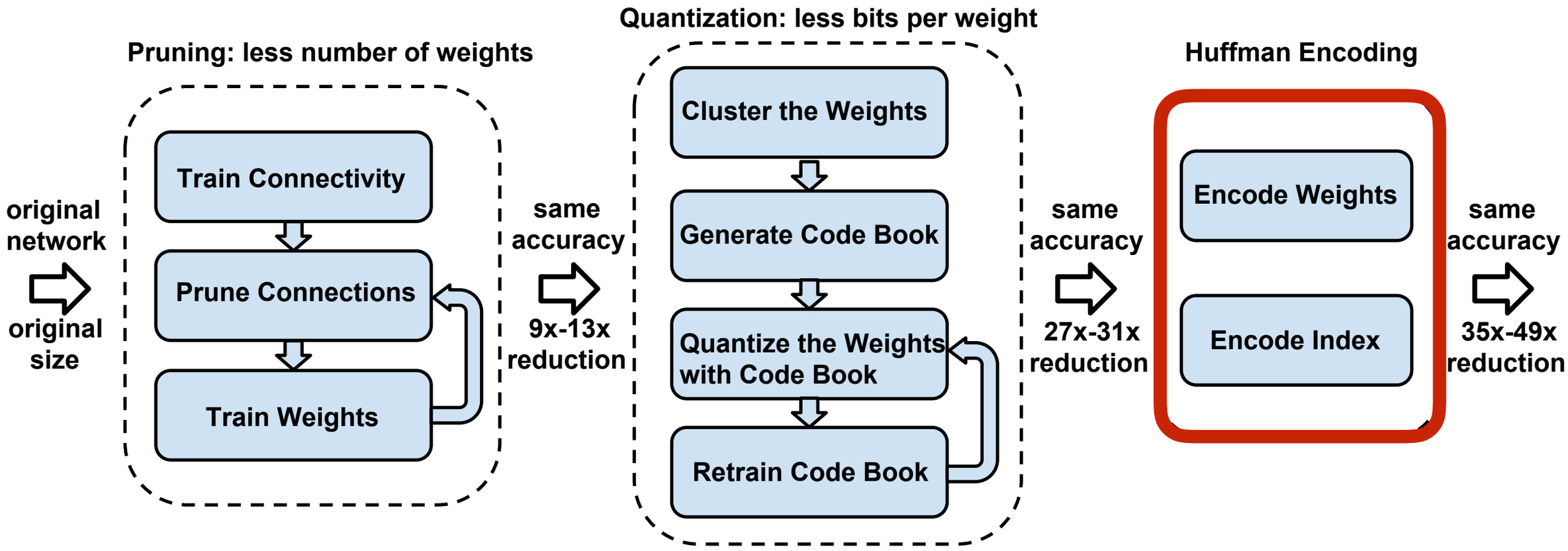
Deep Compression Pipeline

- **Network Pruning**:
  Less Number of Weights

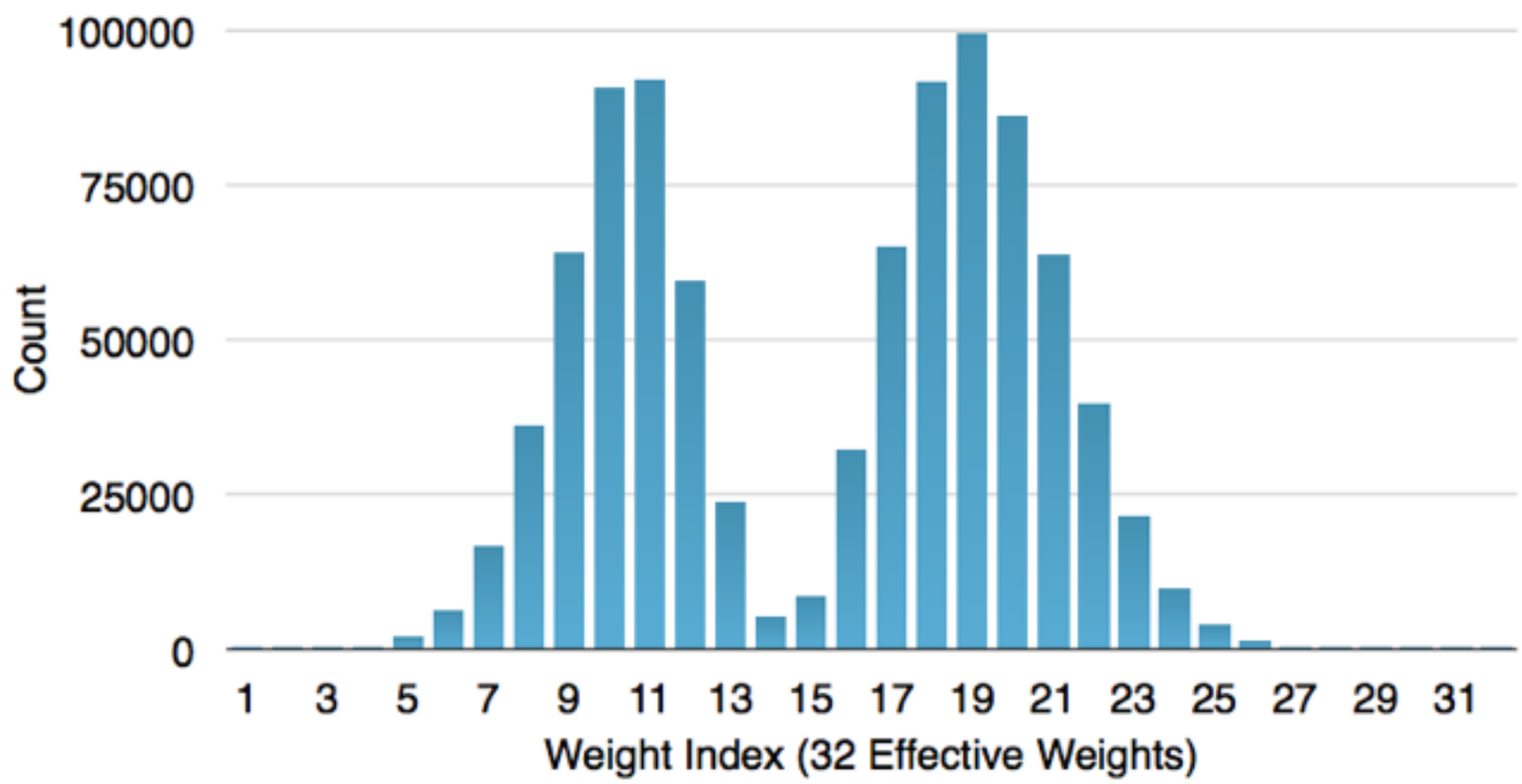
- **Weight Sharing:**
  Reduce Storage for Each Remaining Weight

- **Huffman Coding:**
  Entropy of the Total Remaining Weights

3. Huffman Coding



Huffman Coding

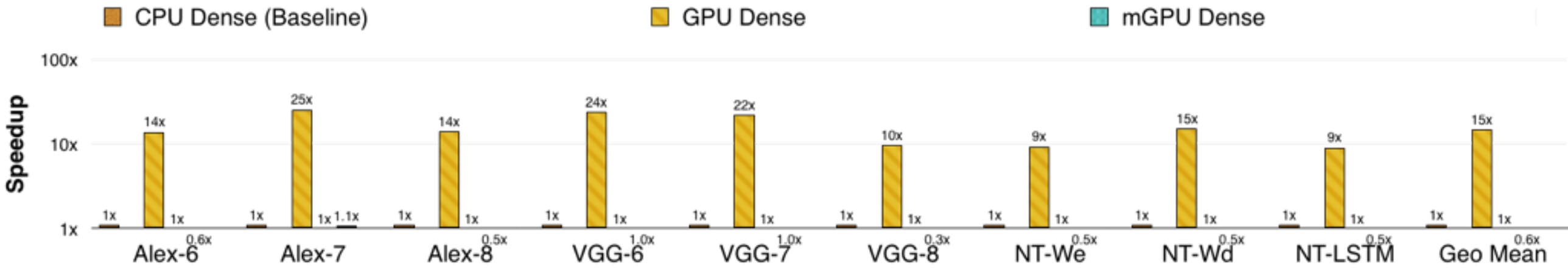


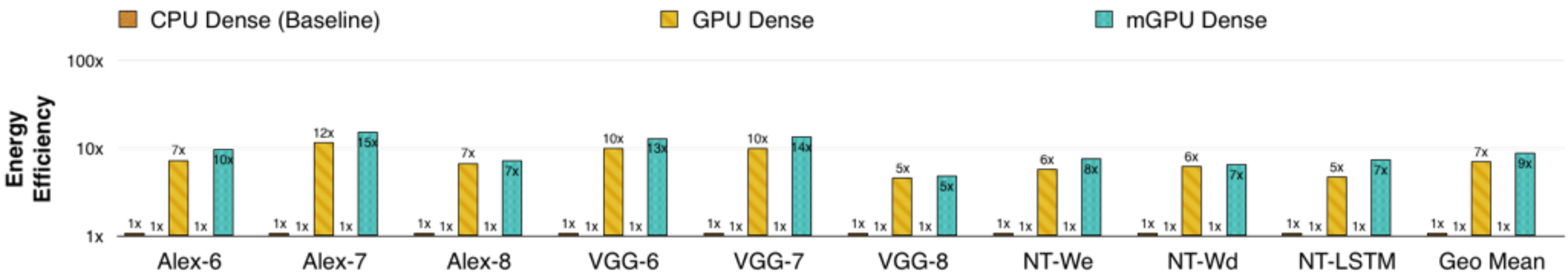
Deep Compression Result on 4 Convnets

Network	Top-1 Error	Top-5 Error	Parameters	Compress Rate
LeNet-300-100 Ref	1.64%	-	1070 KB	
LeNet-300-100 Compressed	1.58%	-	**27 KB**	**40**×
LeNet-5 Ref	0.80%	-	1720 KB	
LeNet-5 Compressed	0.74%	-	**44 KB**	**39**×
AlexNet Ref	42.78%	19.73%	240 MB	
AlexNet Compressed	42.78%	19.70%	**6.9 MB**	**35**×
VGG-16 Ref	31.50%	11.32%	552 MB	
VGG-16 Compressed	31.17%	10.91%	**11.3 MB**	**49**×
SqueezeNet Ref	42.5%	19.7%	4.8 MB	
SqueezeNet Compressed	42.5%	19.7%	**0.47MB**	**10**×
GoogLeNet Ref	31.30%	11.10%	28 MB	
GoogLeNet Compressed	31.26%	11.08%	**2.8 MB**	**10**×

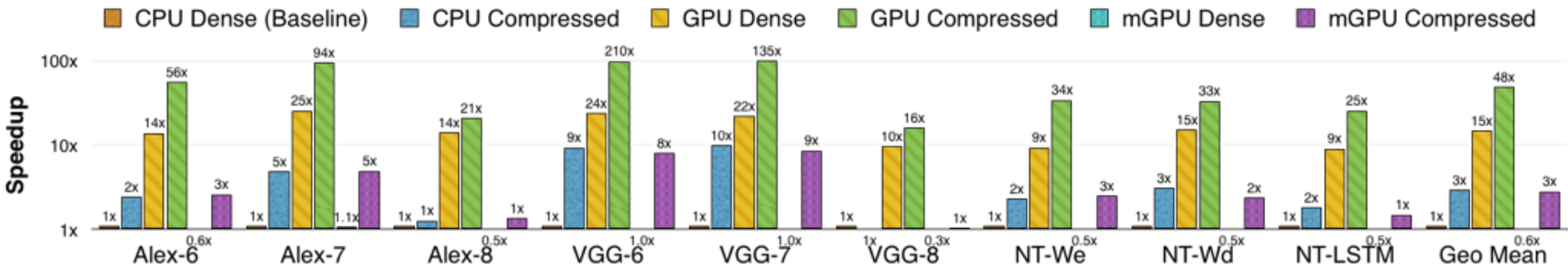
- Just Pruning and Quantization, no Huffman Coding
- Inception Model is Really Efficient
- Still an Order of Magnitude Smaller
- Fit in SRAM Cache

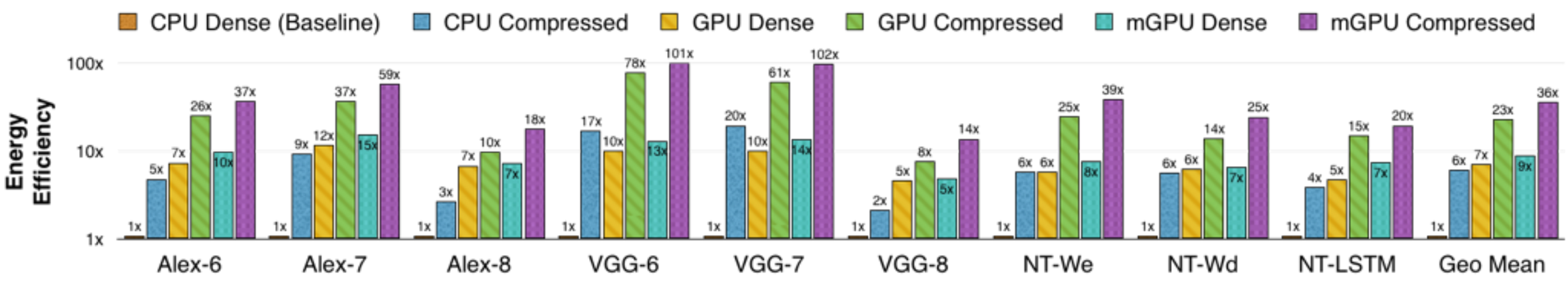
Speedup/Energy Efficiency on CPU/GPU





Speedup/Energy Efficiency on CPU/GPU



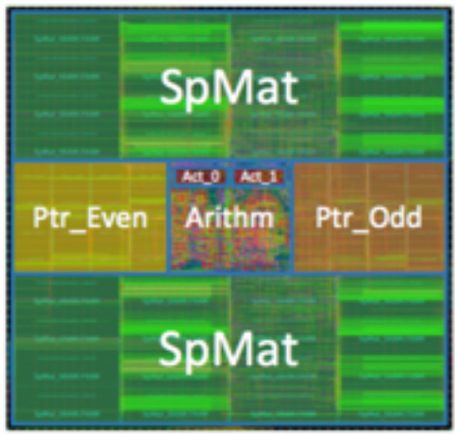


EIE: Efficient Inference Engine

64PEs, 43mm^2,

600mW, 128GOPS @1GHz

Scalable to 256PEs



Sparse Matrix

Sparse Vector

Weight Sharing

Fully fits in SRAM

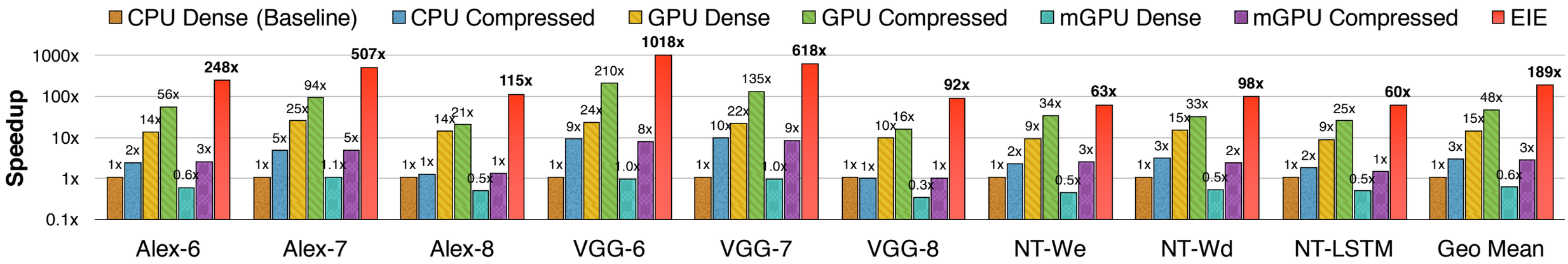


Figure 6. Speedups of GPU, mobile GPU and EIE compared with CPU running uncompressed DNN model. There is no batching in all cases.

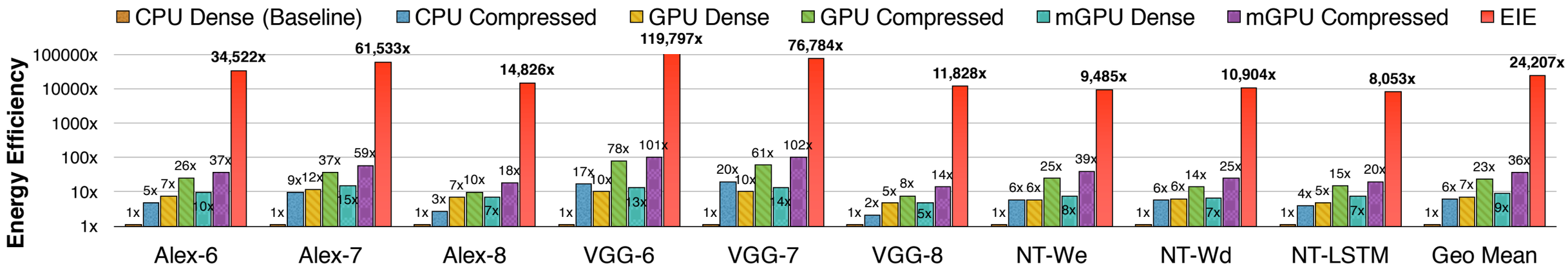


Figure 7. Energy efficiency of GPU, mobile GPU and EIE compared with CPU running uncompressed DNN model. There is no batching in all cases.

Han et al. “EIE: Efficient Inference Engine on Compressed Deep Neural Network”, ISCA 2016

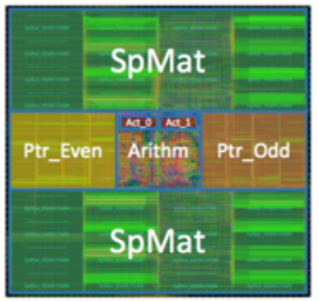
Conclusion

10x-50x Compression Means:

- **Complex DNNs can be put in mobile applications (<10MB total)**
  - 500MB network (125M weights) becomes 10MB
  - 10MB network (2.5M weights) becomes 1MB

- **Memory bandwidth reduced by 10-50x**
  - Particularly for FC layers in real-time applications with no reuse

- **Memory working set fits in on-chip SRAM**
  - 5pJ/word access vs 640pJ/word, save 120x energy



Slide credited from Bill Dally, NIPS'15 Tutorial: High-Performance Hardware for Machine Learning

Acknowledgment

- Thanks Bill Dally, Mark Horowitz, Andrew Ng and Fei-Fei Li for guidance and useful discussions.

Thank you!

https://github.com/songhan

[1]. Han et al. “Learning both Weights and Connections for Efficient Neural Networks”, NIPS 2015

[2]. Han et al. “Deep Compression: Compressing Deep Neural Networks with Pruning, Trained Quantization and Huffman Coding”, Deep Learning Symposium 2015, ICLR 2016

[3]. Han et al. “EIE: Efficient Inference Engine on Compressed Deep Neural Network”, ISCA 2016

[4]. Iandola, Han,et al. “SqueezeNet: AlexNet-level accuracy with 50x fewer parameters and <0.5MB model size” arXiv’16

[5]. Yao, Han, et.al, “Hardware-friendly convolutional neural network with even-number filter size” ICLR workshop 2016