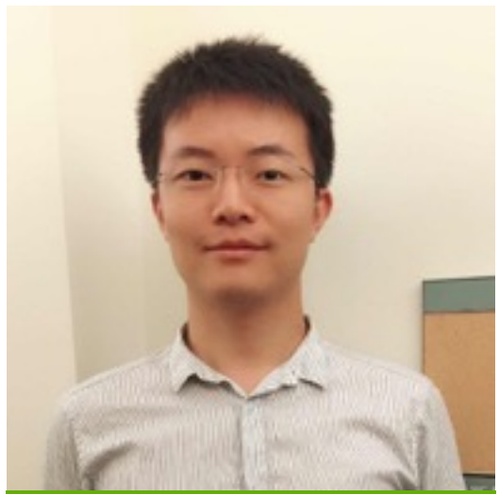
Deep Compression and EIE:

——Deep Neural Network Model Compression and Efficient Inference Engine

Song Han
CVA group, Stanford University
Apr 7, 2016

Intro about me and my advisor



Song Han

- Fourth year PhD with Prof. Bill Dally at Stanford.
- Research interest: deep learning **model compression** and **hardware acceleration**, to make inference more efficient for deployment.
- Recent work on “Deep Compression” and “EIE: Efficient Inference Engine” covered by TheNextPlatform & O’Reilly & TechEmergence & HackerNews

Bill Dally

- Professor at Stanford University and former chairman of CS department, leads the Concurrent VLSI Architecture Group.
- Chief Scientist of NVIDIA.
- Member of the National Academy of Engineering, Fellow of the American Academy of Arts & Sciences, Fellow of the IEEE, Fellow of the ACM and numerous other rewards…

Thanks to my collaborators

- NVIDIA: Jeff Pool, John Tran, Bill Dally
- Stanford: Xingyu Liu, Jing Pu, Ardavan Pedram, Mark Horowitz, Bill Dally
- Tsinghua: Huizi Mao, Song Yao, Yu Wang
- Berkeley: Forrest Iandola, Matthew Moskewicz, Khalid Ashraf, Kurt Keutzer

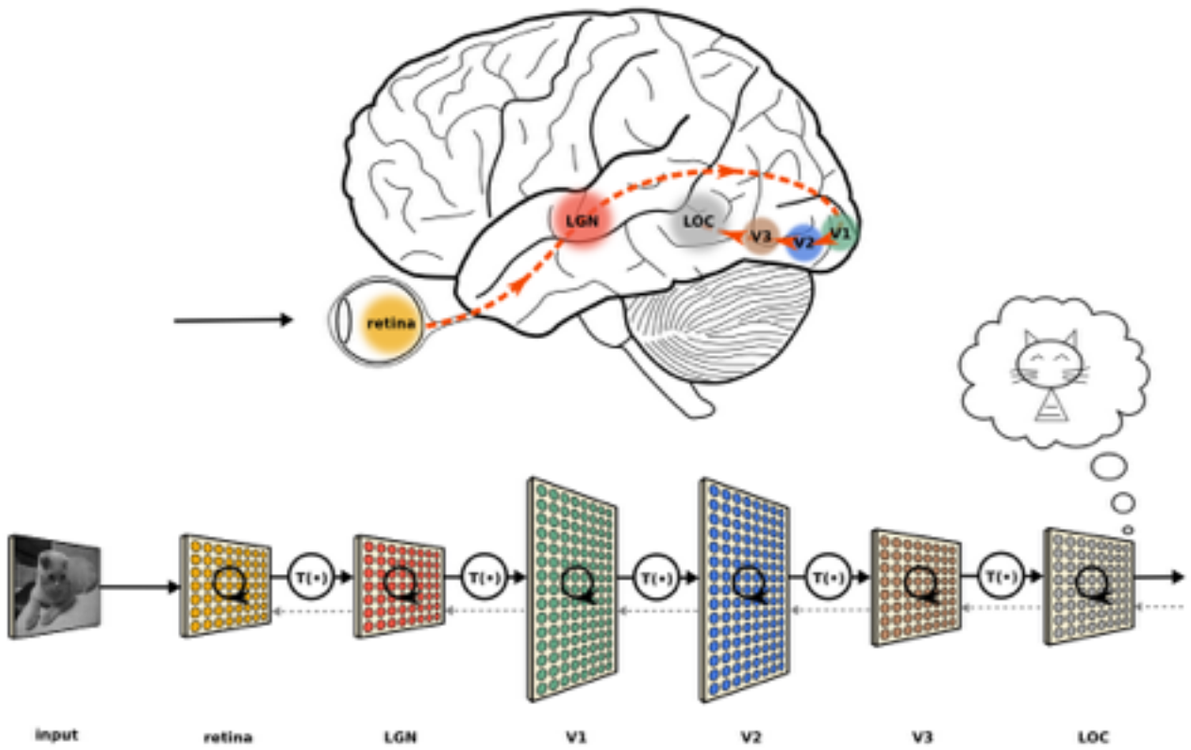
You'll be interested in his GTC talk: S6417 - FireCaffe

This Talk:

- **Deep Compression**[1,2]: A Deep Neural Network Model Compression Pipeline.
- **EIE Accelerator**[3]: Efficient Inference Engine that Accelerates the Compressed Deep Neural Network Model.
- **SqueezeNet++**[4,5]: ConvNet Architecture Design Space Exploration

[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”, 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 <0.5 MB model size” arXiv 2016
[5]. Yao, Han, et.al, “Hardware-friendly convolutional neural network with even-number filter size” ICLR’16 workshop

Deep Learning： Next Wave of AI



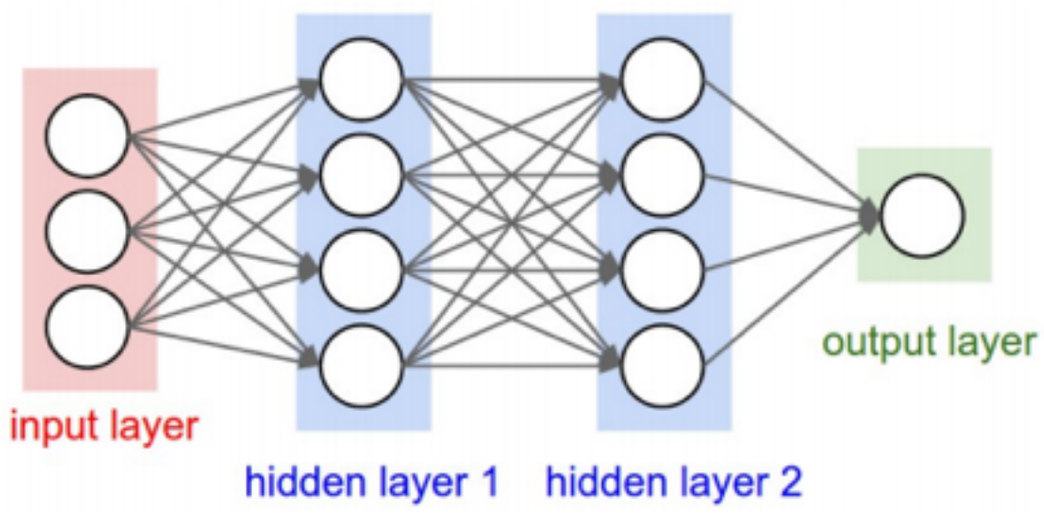
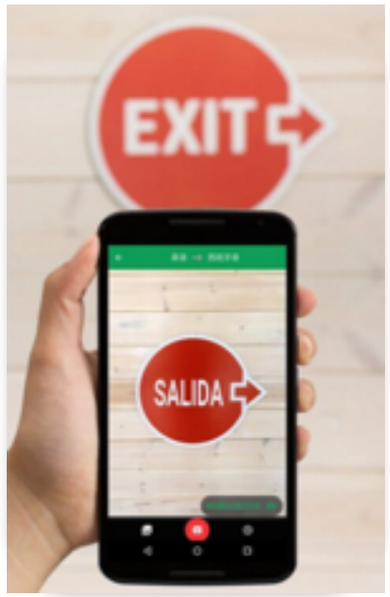


Image Recognition

Speech Recognition

Natural Language Processing

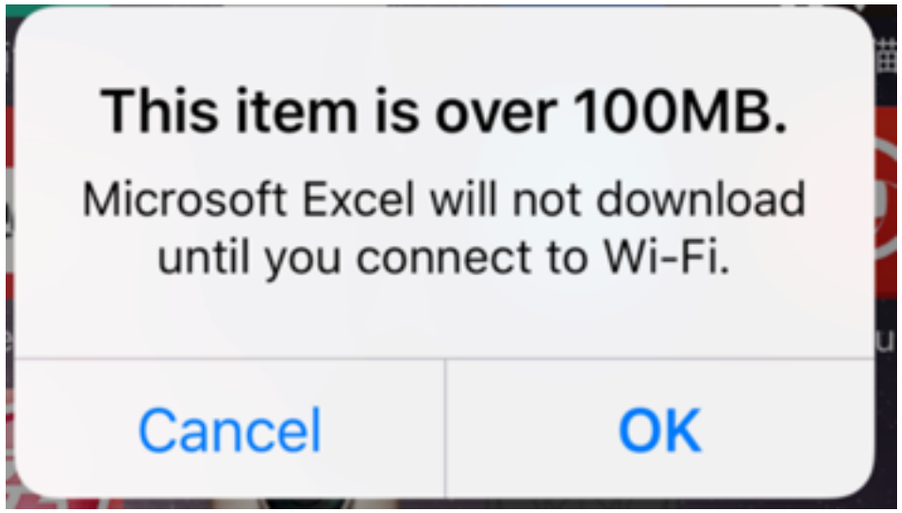
Applications



The Problem:

If Running DNN on Mobile…

App developers suffers from the model size



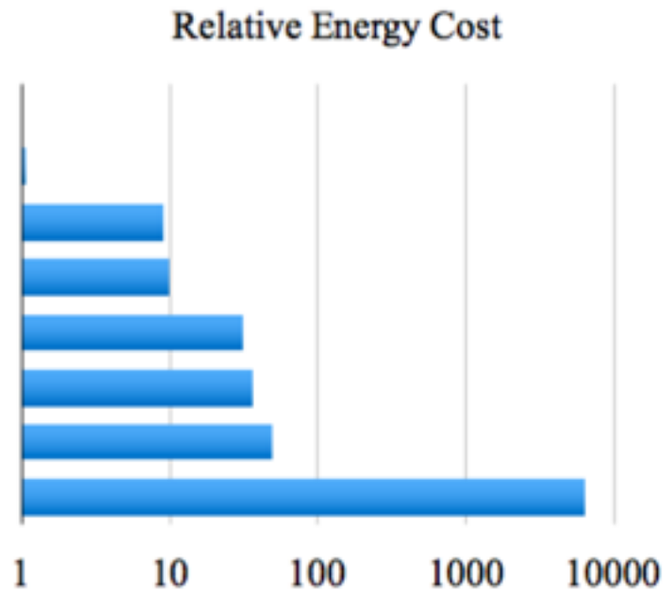
"At Baidu, our #1 motivation for compressing networks is to **bring down the size of the binary file**. As a mobile-first company, we frequently update various apps via different app stores. We've **very sensitive to the size of our binary files**, and a feature that increases the binary size by 100MB will receive much more scrutiny than one that increases it by 10MB." —Andrew Ng

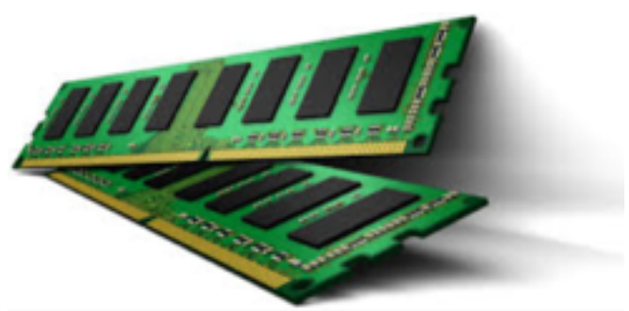
The Problem:

If Running DNN on Mobile…

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**





The Problem:

If Running DNN on the Cloud…

Network Delay

Power Budget

User Privacy

Intelligent but Inefficient

Deep Compression

Problem 1: Model Size
Solution 1: Deep Compression

Smaller Size
Compress Mobile App Size by 35x-50x

Accuracy
no loss of accuracy
improved accuracy

Speedup
make inference faster

EIE Accelerator

Problem 2: Latency, Power, Energy
Solution 2: ASIC accelerator

Offline
No dependency on network connection

Real Time
No network delay high frame rate

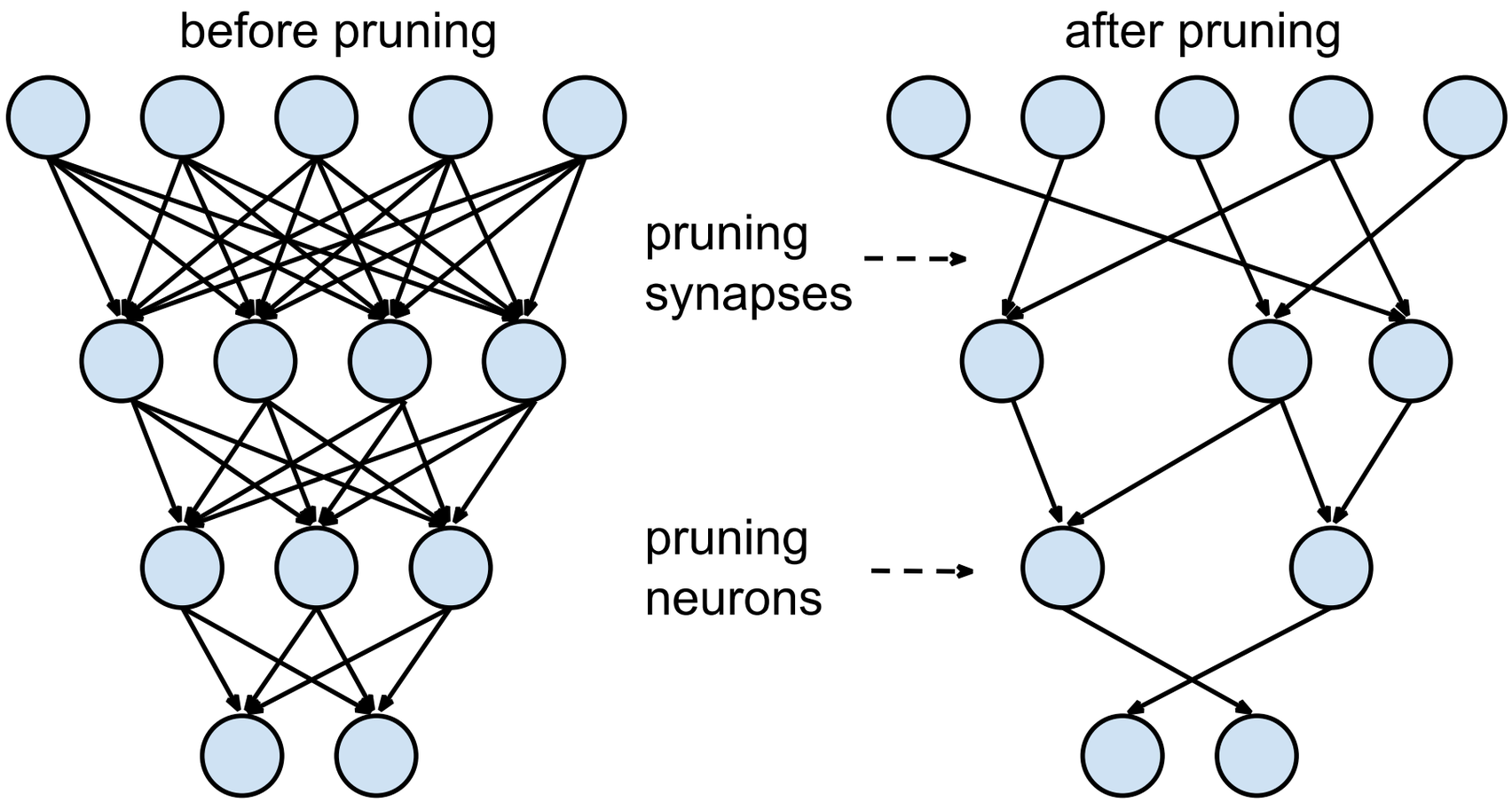
Low Power
High energy efficiency that preserves battery

Part1: Deep Compression

- AlexNet: 35×, 240MB => 6.9MB **=> 0.47MB (510x)**
- VGG16: 49×, 552MB => 11.3MB
- With no loss of accuracy on ImageNet12
- Weights fits on-chip SRAM, taking 120x less energy than DRAM

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, ICLR 2016
3. Iandola, Han, et al. “SqueezeNet: AlexNet-level accuracy with 50x fewer parameters and <0.5MB model size” ECCV submission

1. Pruning



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

Pruning: Motivation

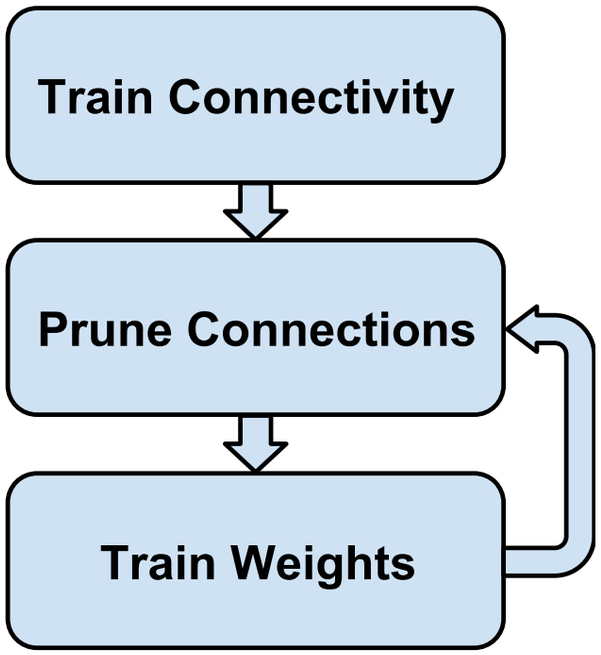
Age	Number of Connections	Stage
at birth	50 Trillion	newly formed
1 year old	1000 Trillion	peak
10 year old	500 Trillion	pruned and stabilized

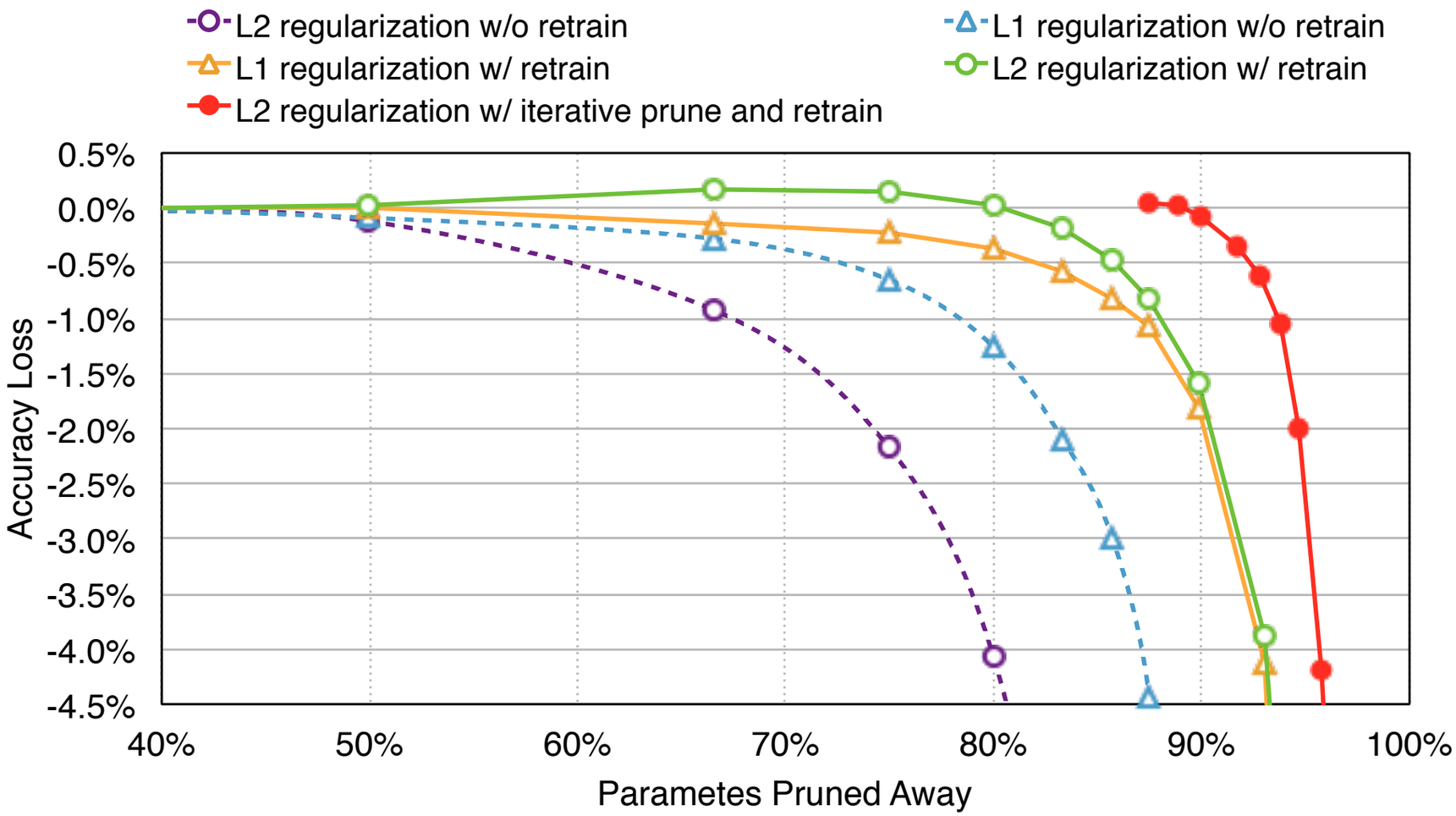
Table 1: The synapses pruning mechanism in human brain development

- 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.

Retrain to Recover Accuracy





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

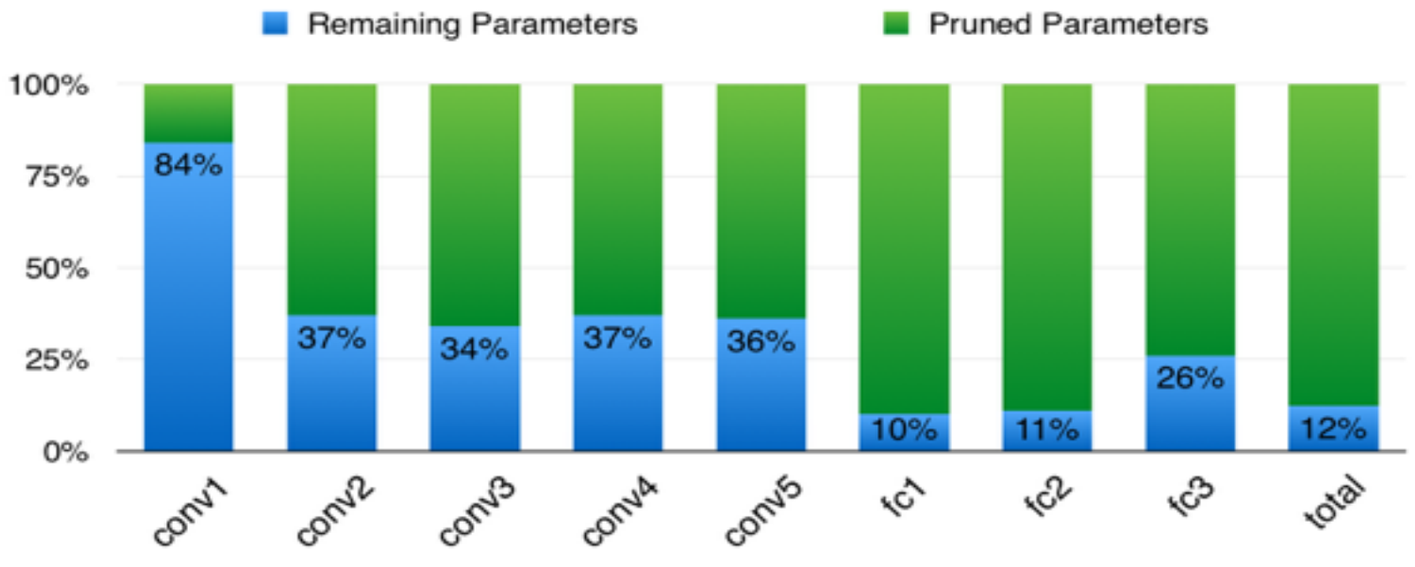
Pruning: Result on 4 Covnets

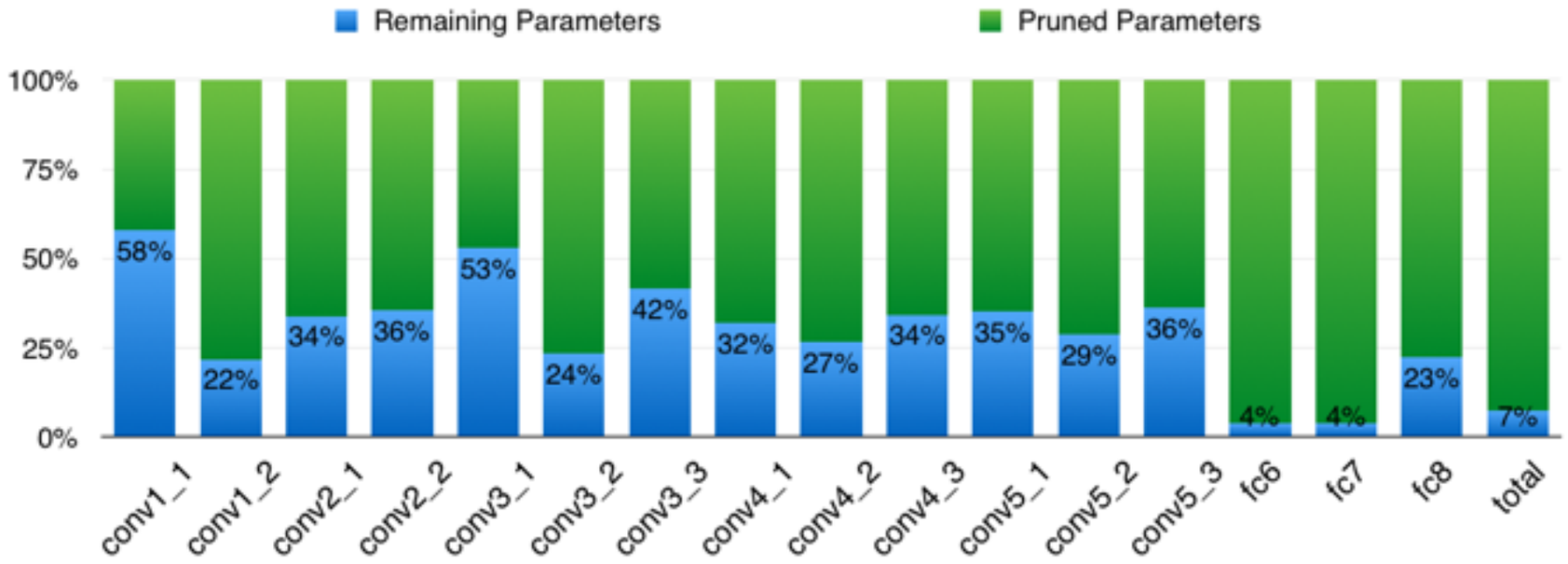
Network	Top-1 Error	Top-5 Error	Parameters	Compression Rate
LeNet-300-100 Ref	1.64%	-	267K	
LeNet-300-100 Pruned	1.59%	-	**22K**	**12**×
LeNet-5 Ref	0.80%	-	431K	
LeNet-5 Pruned	0.77%	-	**36K**	**12**×
AlexNet Ref	42.78%	19.73%	61M	
AlexNet Pruned	42.77%	19.67%	**6.7M**	**9**×
VGG16 Ref	31.50%	11.32%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

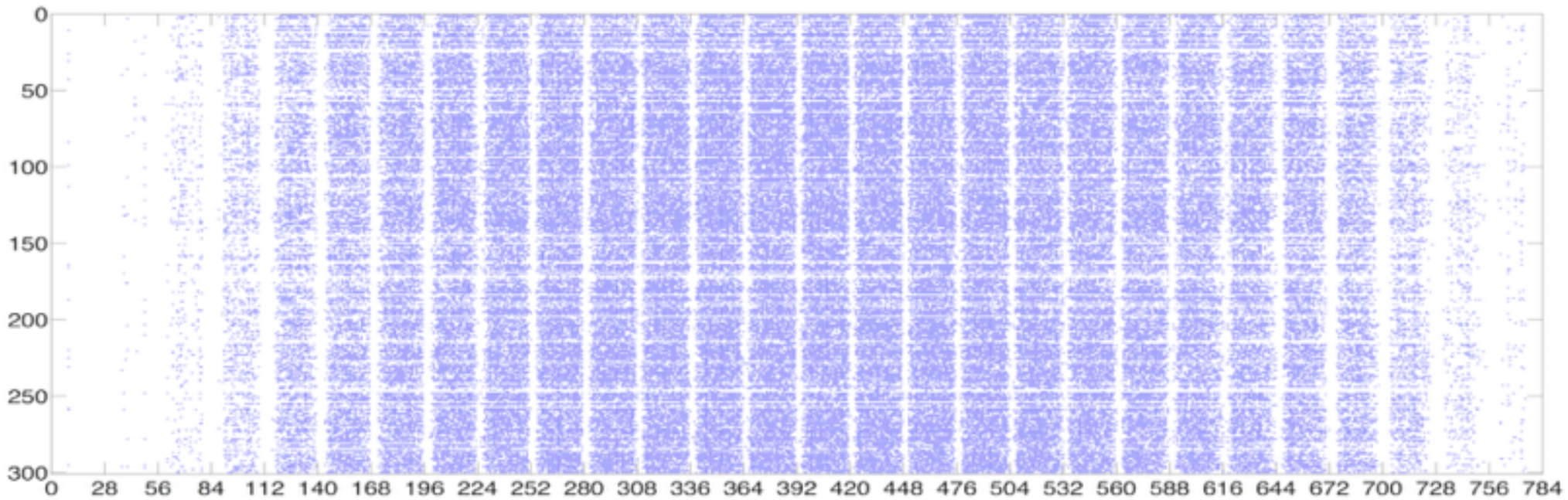
AlexNet & VGGNet





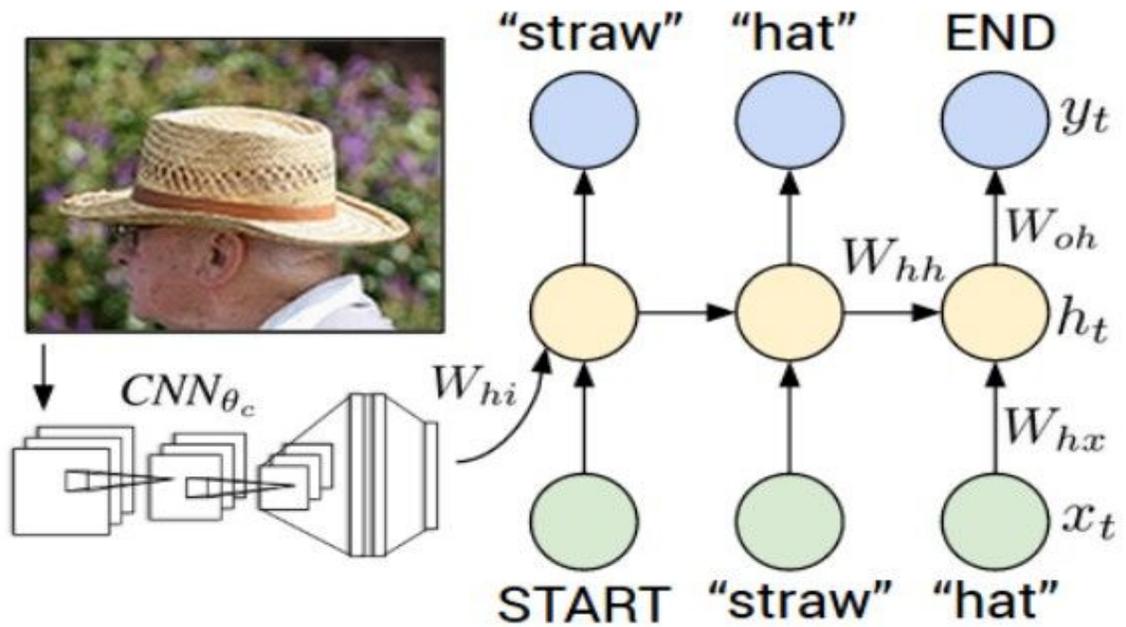
Han et al. Learning both Weights and Connections for Efficient Neural Networks, NIPS 2015

Mask Visualization



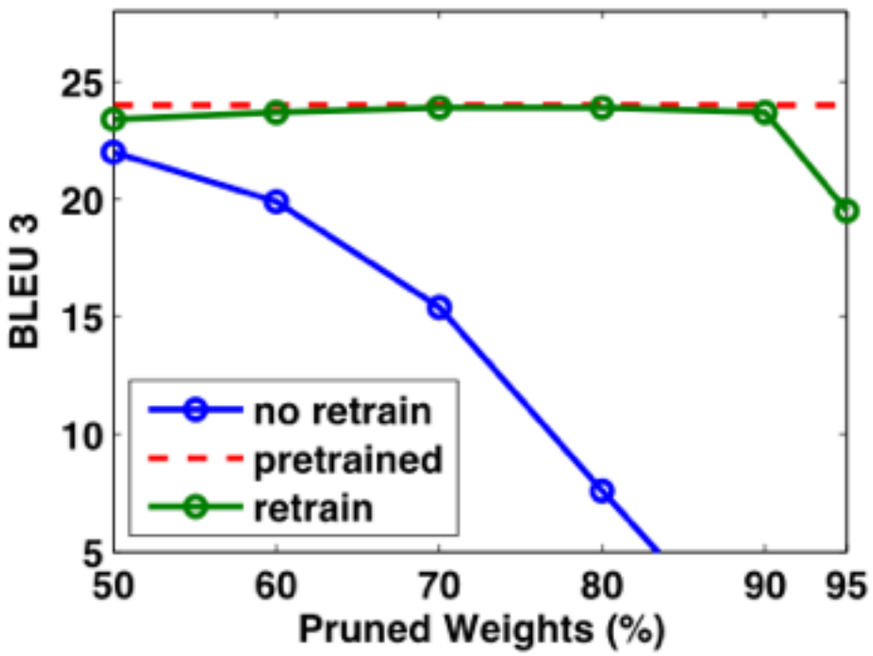
Visualization of the first FC layer's sparsity pattern of Lenet-300-100. It has a banded structure repeated 28 times, which correspond to the un-pruned parameters in the center of the images, since the digits are written in the center.

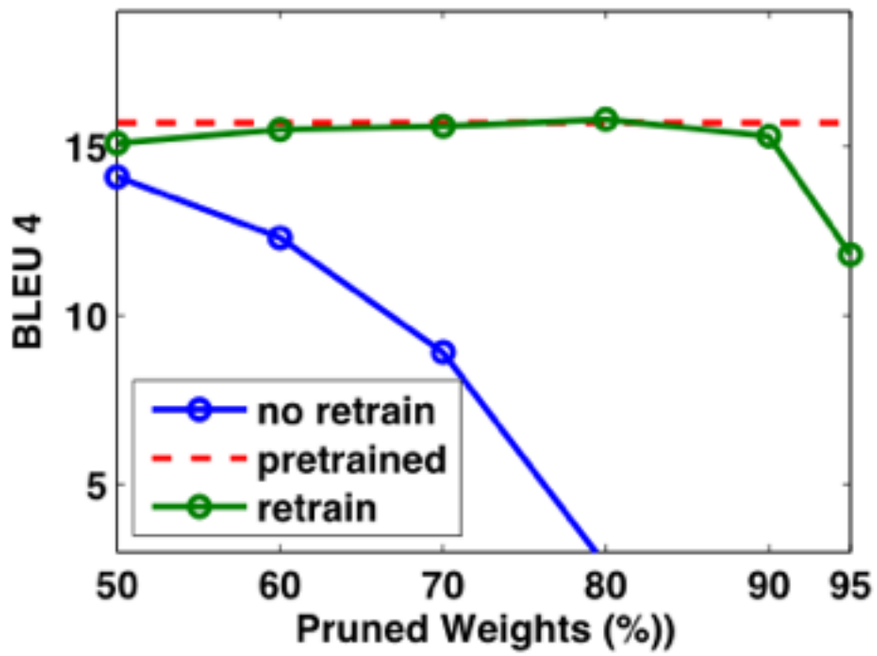
Pruning NeuralTalk and LSTM



Karpathy, Feifei, 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

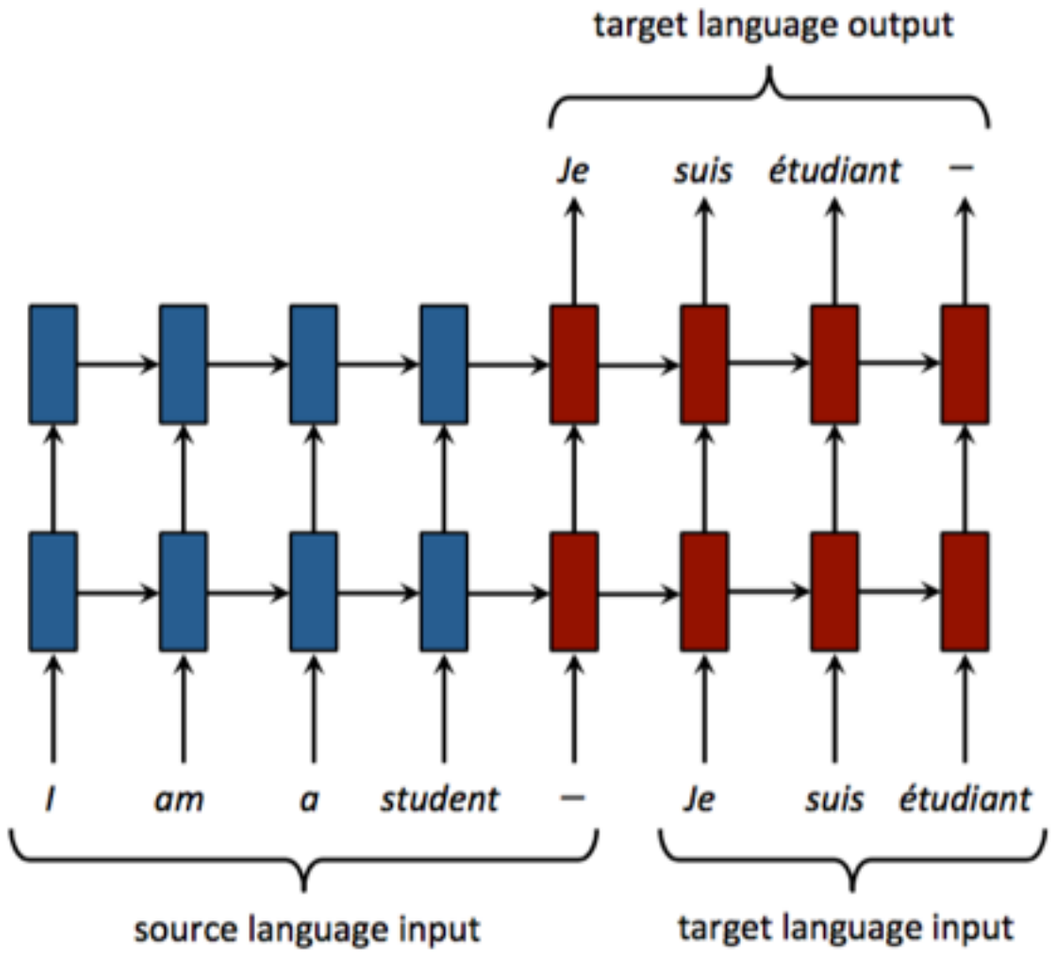
- **Original**: a basketball player in a white uniform is playing with a ball
- **Pruned 90%**: a basketball player in a white uniform is playing with a basketball

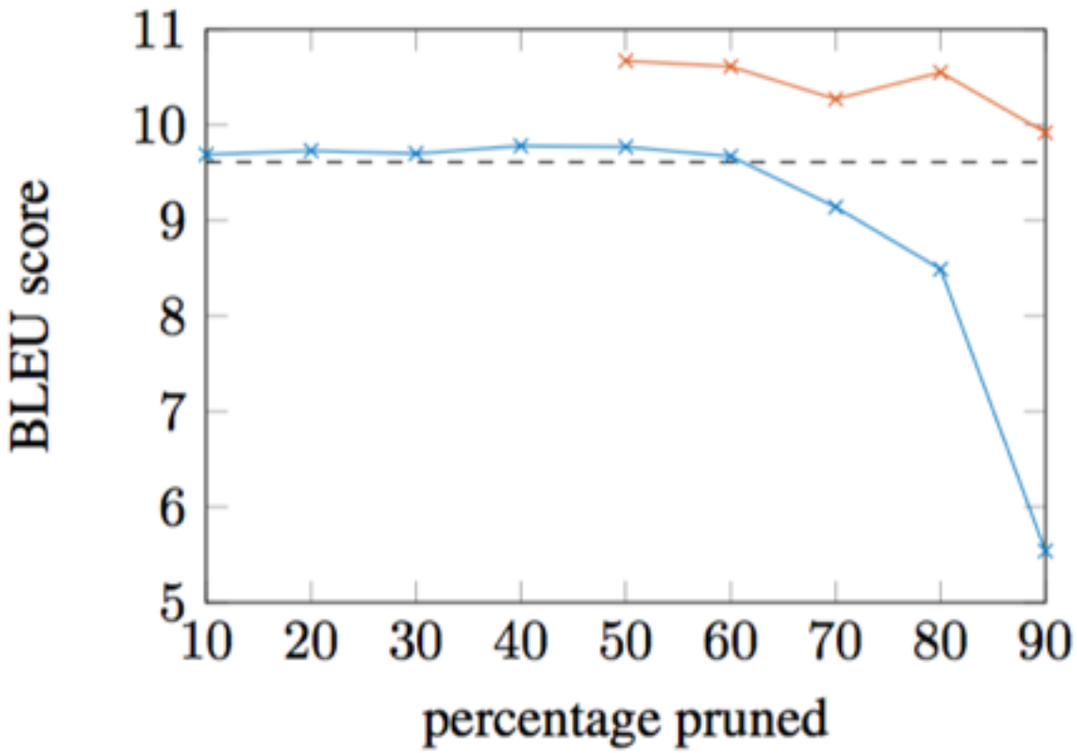
- **Original** : a brown dog is running through a grassy field
- **Pruned 90%**: a brown dog is running through a grassy area

- **Original** : a 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 et al. Learning both Weights and Connections for Efficient Neural Networks, NIPS 2015

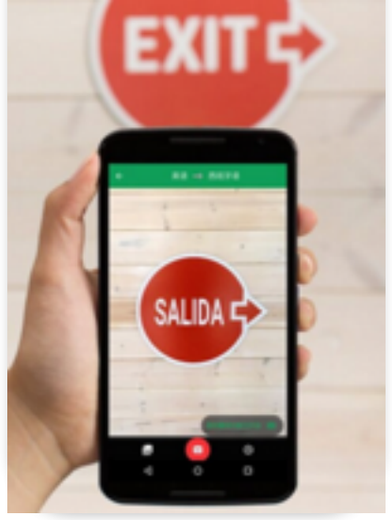
Pruning Neural Machine Translation



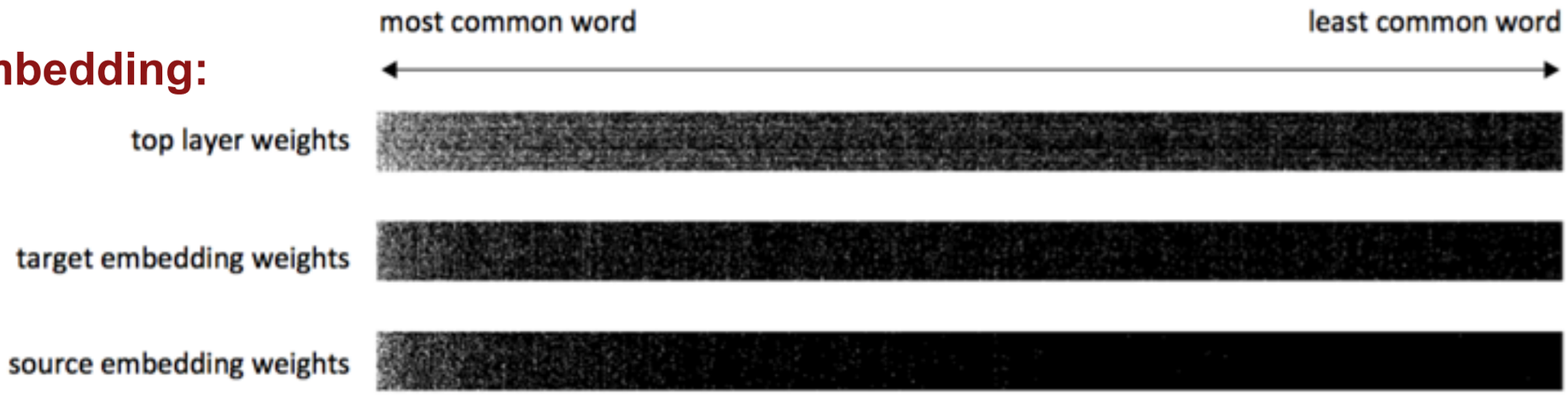


Abi See, “CS224N Final Project: Exploiting the Redundancy in Neural Machine Translation”

Pruning Neural Machine Translation

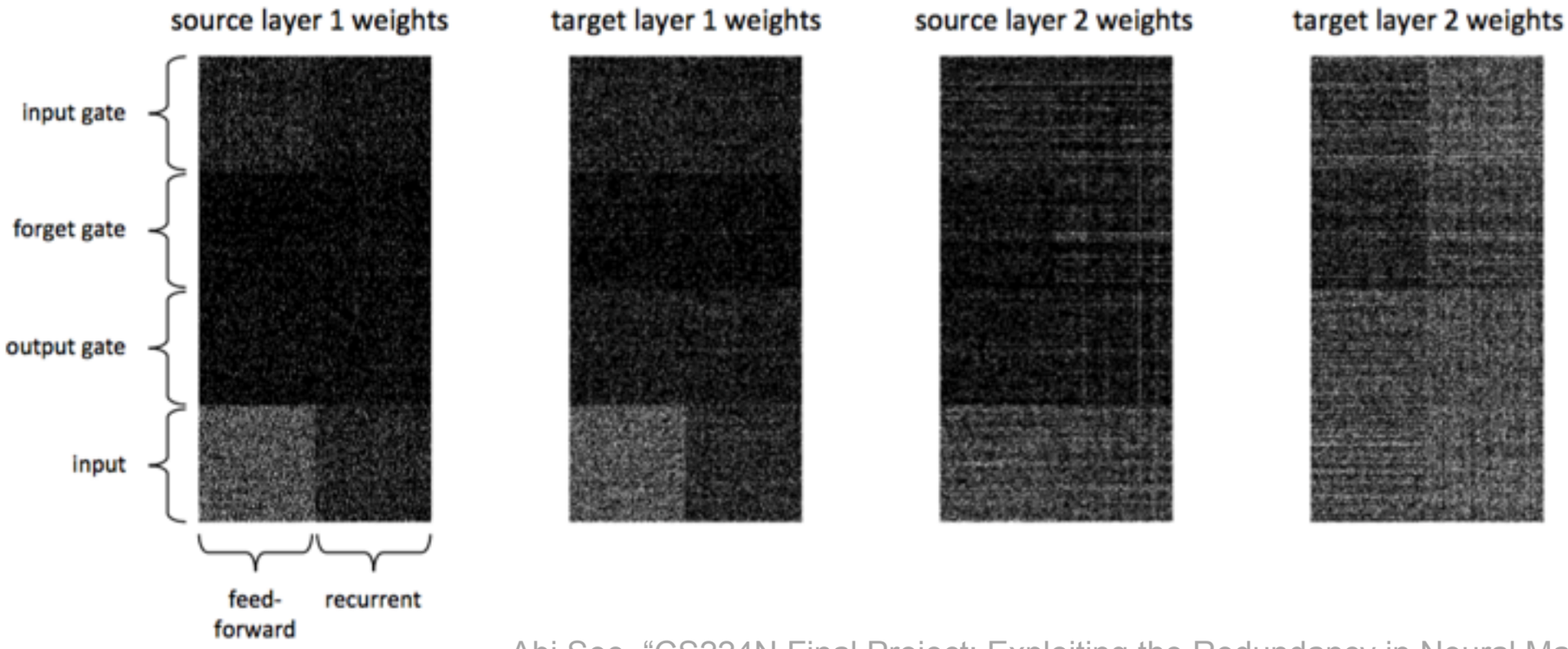


Word Embedding:



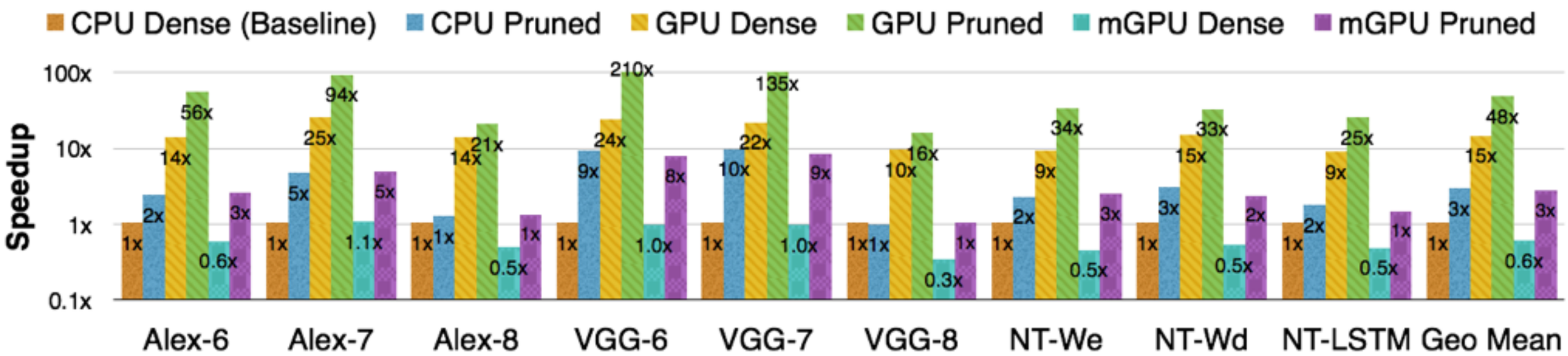
Dark means zero and redundant, White means non-zero and useful

LSTM:



Abi See, “CS224N Final Project: Exploiting the Redundancy in Neural Machine Translation”

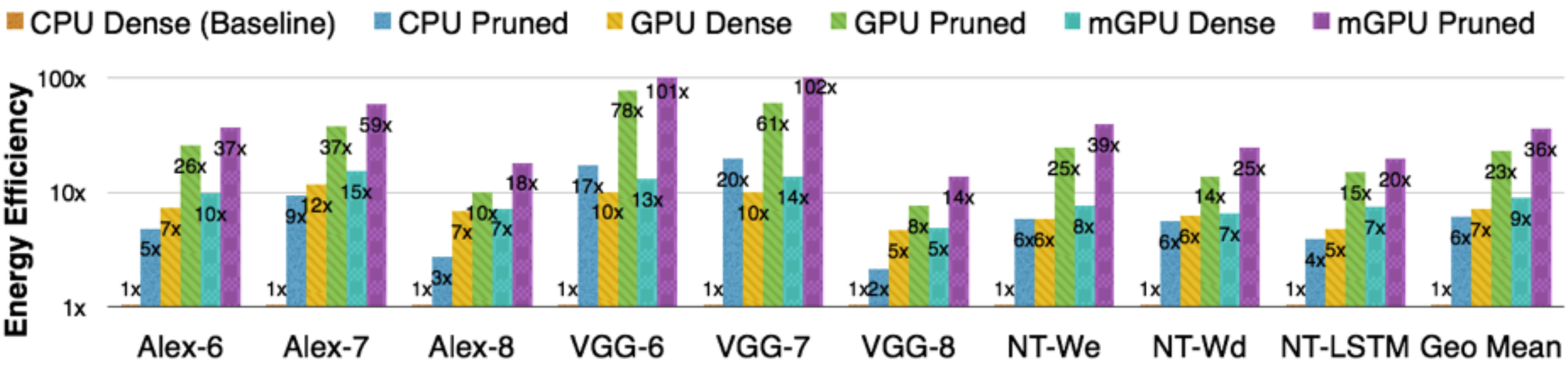
Speedup (FC layer)



- Intel Core i7 5930K: MKL CBLAS GEMV, MKL SPBLAS CSRMV
- NVIDIA GeForce GTX Titan X: cuBLAS GEMV, cuSPARSE CSRMV
- NVIDIA Tegra K1: cuBLAS GEMV, cuSPARSE CSRMV

Han et al. Deep Compression: Compressing Deep Neural Networks with Pruning, Trained Quantization and Huffman Coding, ICLR 2016

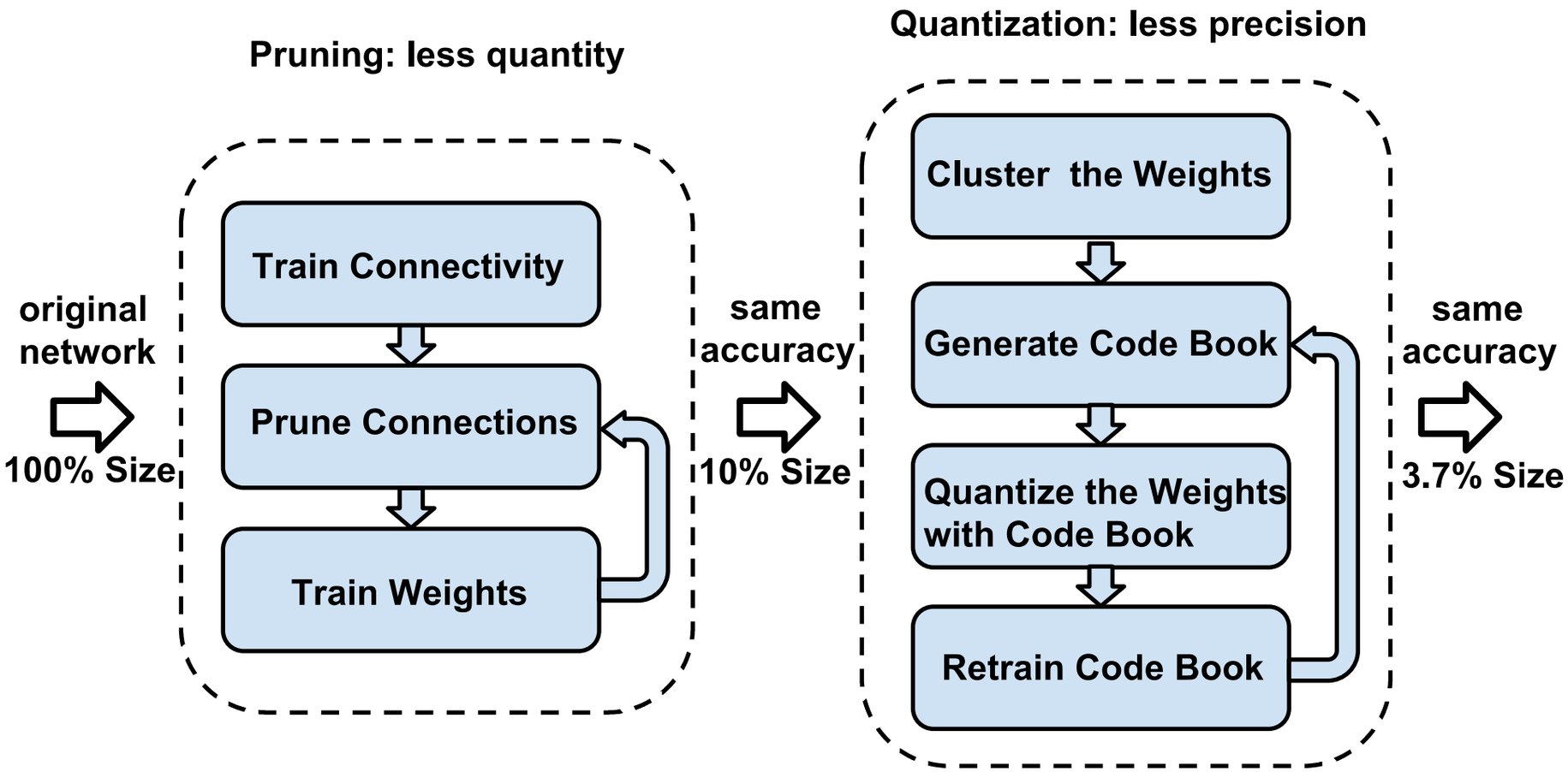
Energy Efficiency (FC layer)



- Intel Core i7 5930K: CPU socket and DRAM power are reported by pcm-power utility
- NVIDIA GeForce GTX Titan X: reported by nvidia-smi utility
- NVIDIA Tegra K1: measured the total power consumption with a power-meter, 15% AC to DC conversion loss, 85% regulator efficiency and 15% power consumed by peripheral components => 60% AP+DRAM power

Han et al. Deep Compression: Compressing Deep Neural Networks with Pruning, Trained Quantization and Huffman Coding, ICLR 2016

2. Weight Sharing
(Trained Quantization)



Han et al. Deep Compression: Compressing Deep Neural Networks with Pruning, Trained Quantization and Huffman Coding, ICLR 2016

Weight Sharing: Overview

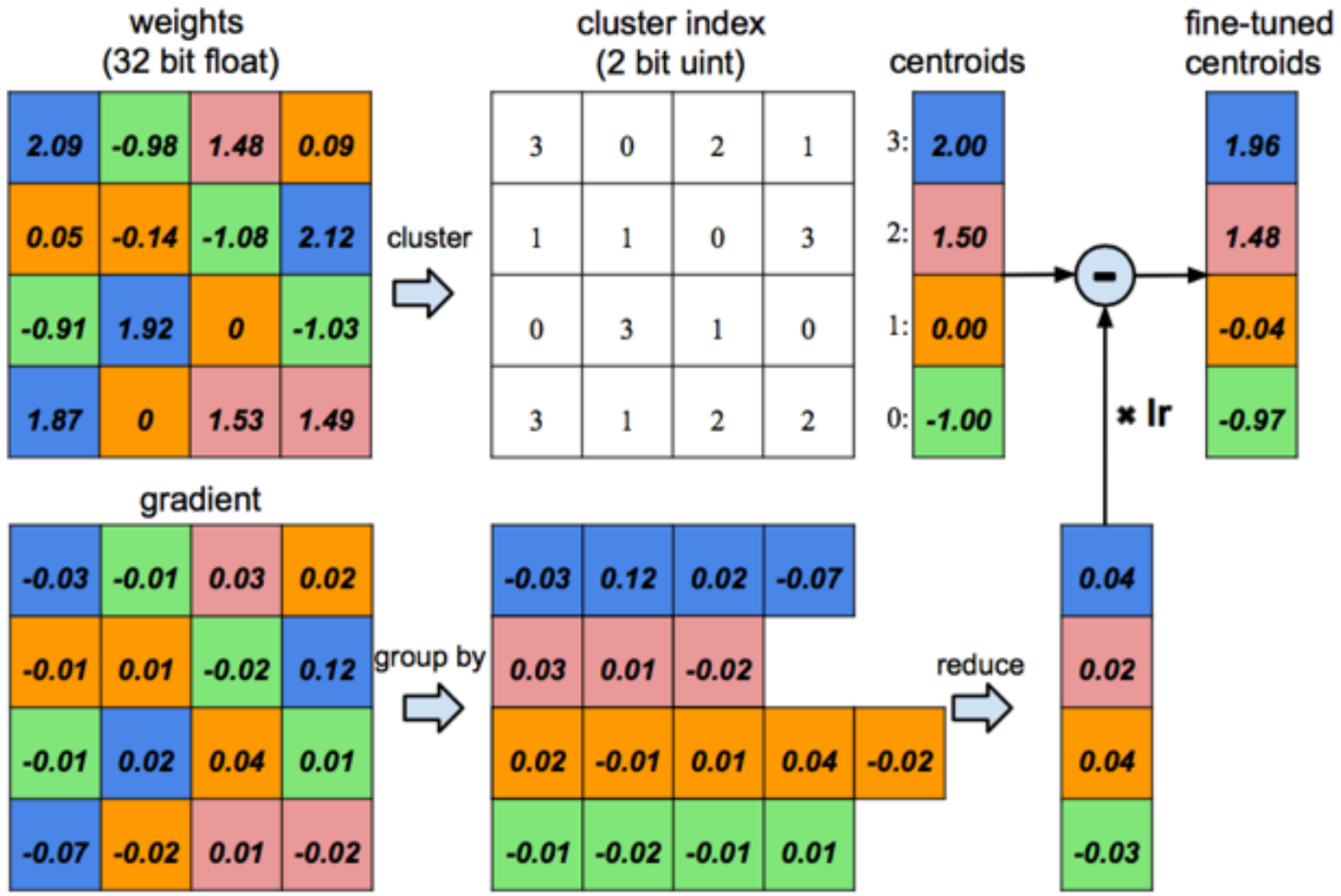
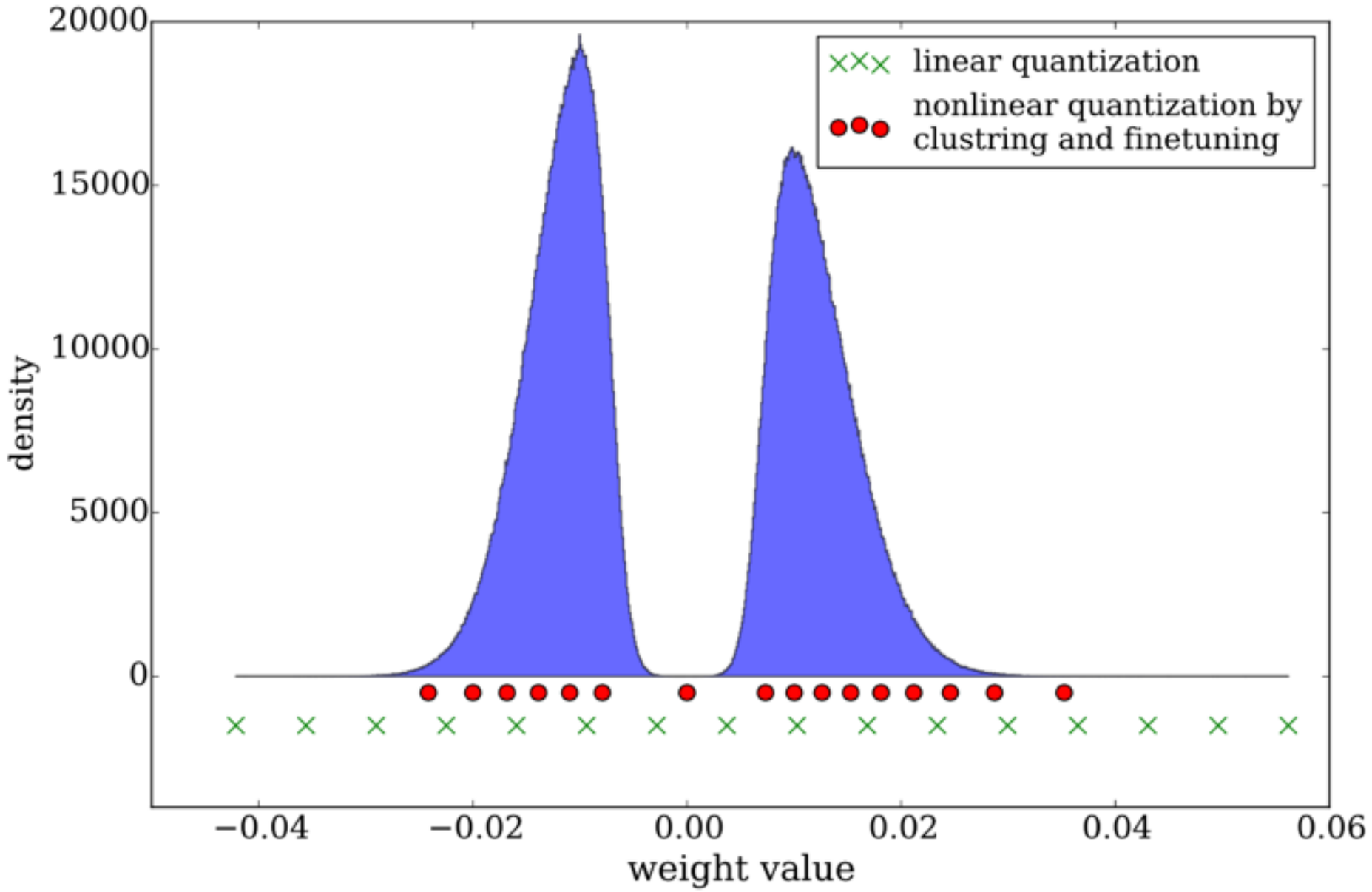


Figure 3: Weight sharing by scalar quantization (top) and centroids fine-tuning (bottom)

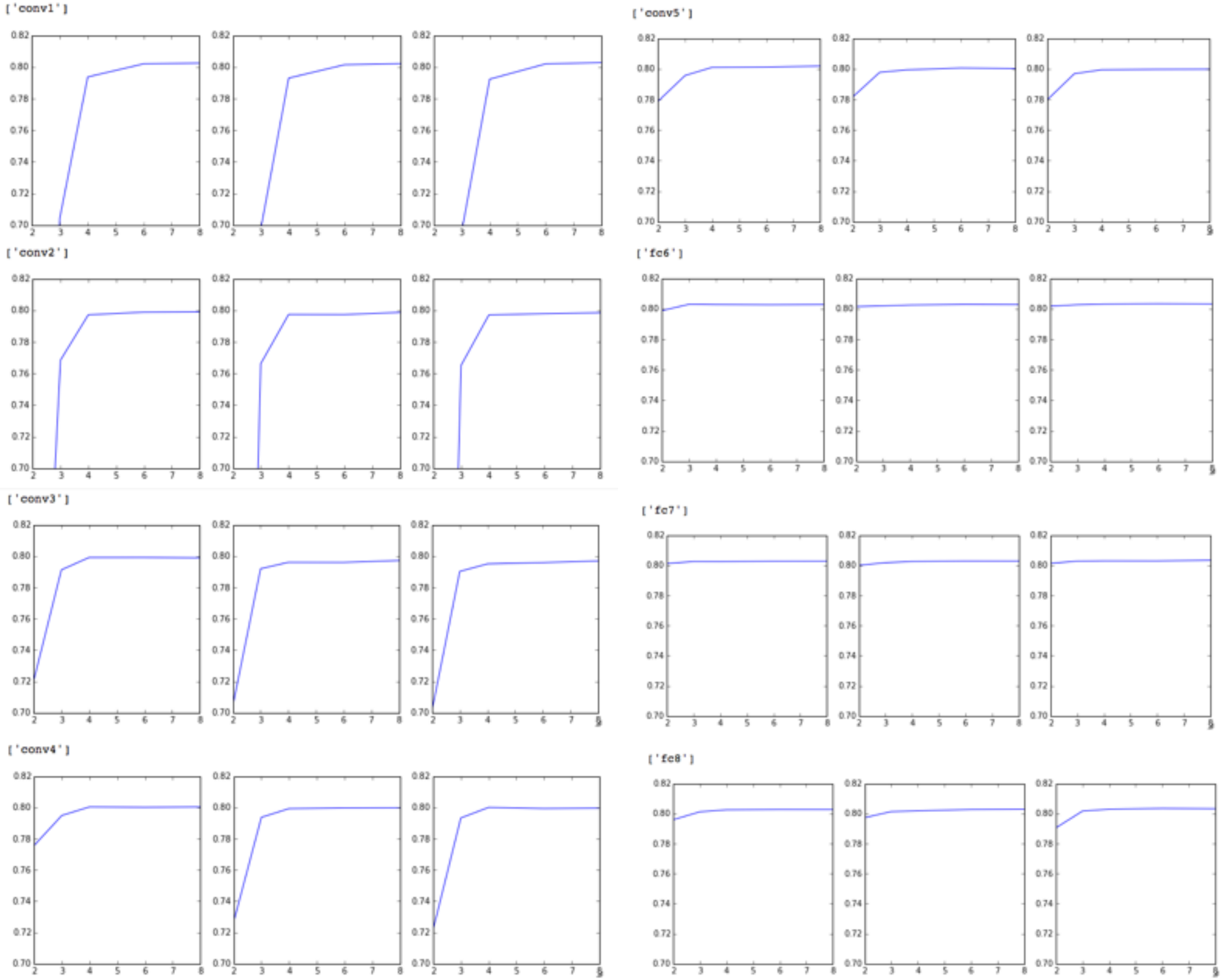
Han et al. Deep Compression: Compressing Deep Neural Networks with Pruning, Trained Quantization and Huffman Coding, ICLR 2016

Finetune Centroids



Han et al. Deep Compression: Compressing Deep Neural Networks with Pruning, Trained Quantization and Huffman Coding, ICLR 2016

Accuracy ~ #Bits on 5 Conv Layers + 3 FC Layers

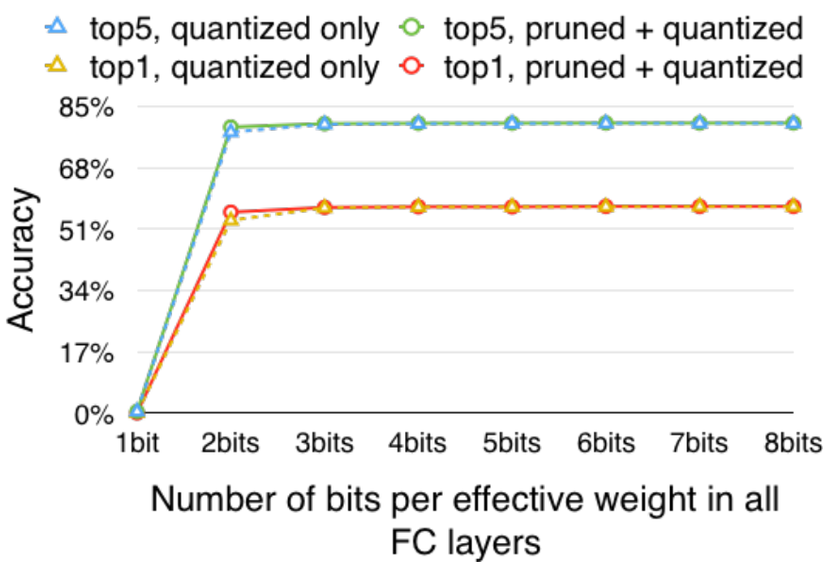


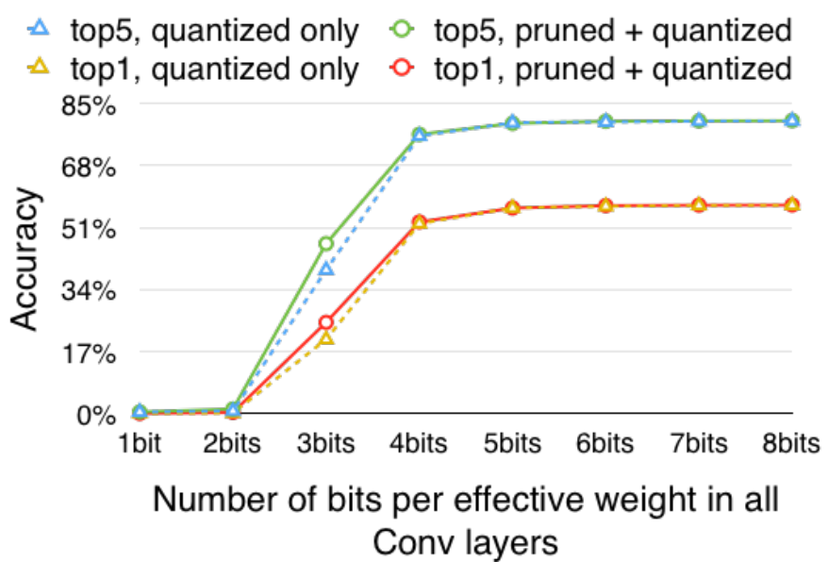
Weight Sharing: Result

- 16 Million => 2^4=16
- 8/5 bit quantization results in **no** accuracy loss
- 8/4 bit quantization results in no top-5 accuracy loss, **0.1%** top-1 accuracy loss
- 4/2 bit quantization results in **-1.99%** top-1 accuracy loss, and **-2.60%** top-5 accuracy loss, not that bad-:

Han et al. Deep Compression: Compressing Deep Neural Networks with Pruning, Trained Quantization and Huffman Coding, ICLR 2016

Pruning and Quantization Works Well Together





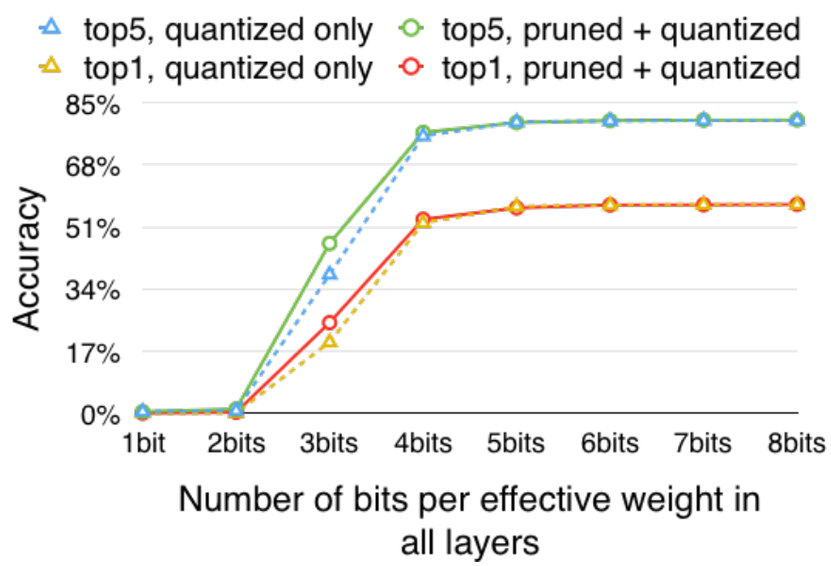
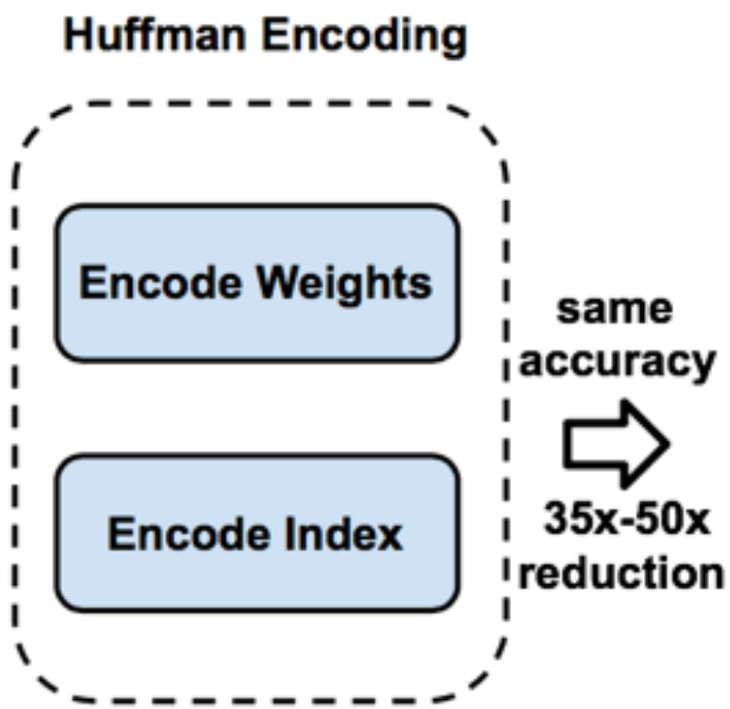


Figure 7: Pruning doesn't hurt quantization. Dashed: quantization on unpruned network. Solid: quantization on pruned network; Accuracy begins to drop at the same number of quantization bits whether or not the network has been pruned. Although pruning made the number of parameters less, quantization still works well, or even better(3 bits case on the left figure) as in the unpruned network.

Han et al. Deep Compression: Compressing Deep Neural Networks with Pruning, Trained Quantization and Huffman Coding, ICLR 2016

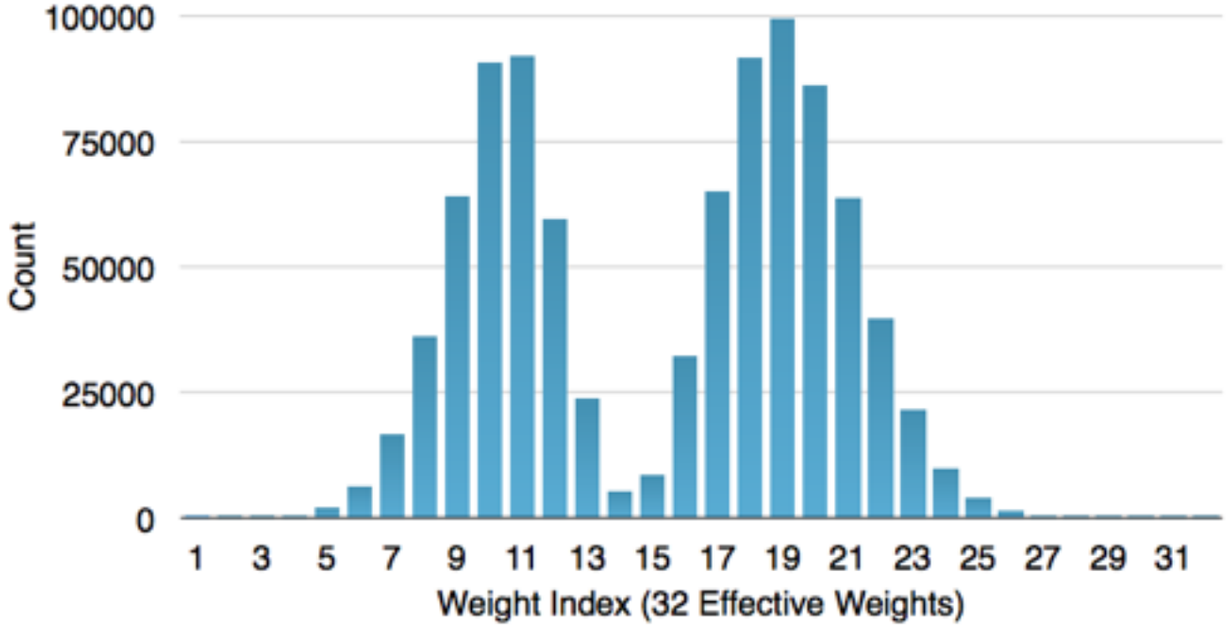
3. Huffman Coding



Han et al. Deep Compression: Compressing Deep Neural Networks with Pruning, Trained Quantization and Huffman Coding, ICLR 2016

Huffman Coding

Huffman code is a type of optimal prefix code that is commonly used for loss-less data compression. It produces a variable-length code table for encoding source symbol. The table is derived from the occurrence probability for each symbol. As in other entropy encoding methods, more common symbols are represented with fewer bits than less common symbols, thus save the total space.



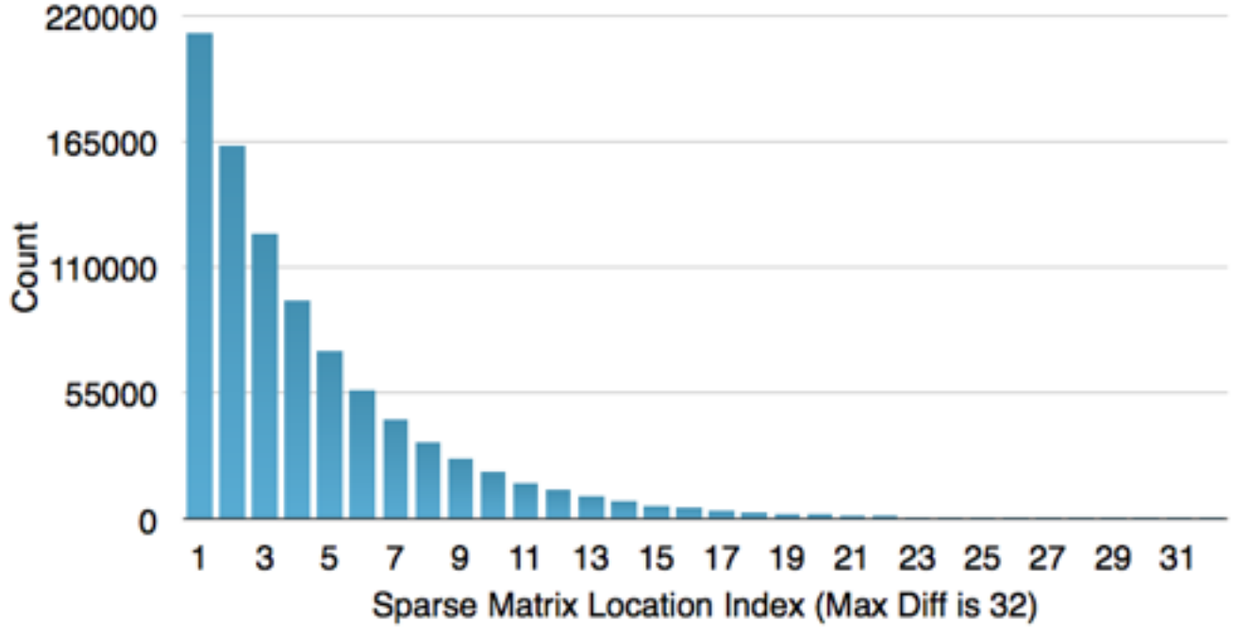


Figure 5: Distribution for weight (Left) and index (Right). The distribution is biased and can be compressed by Huffman encoding

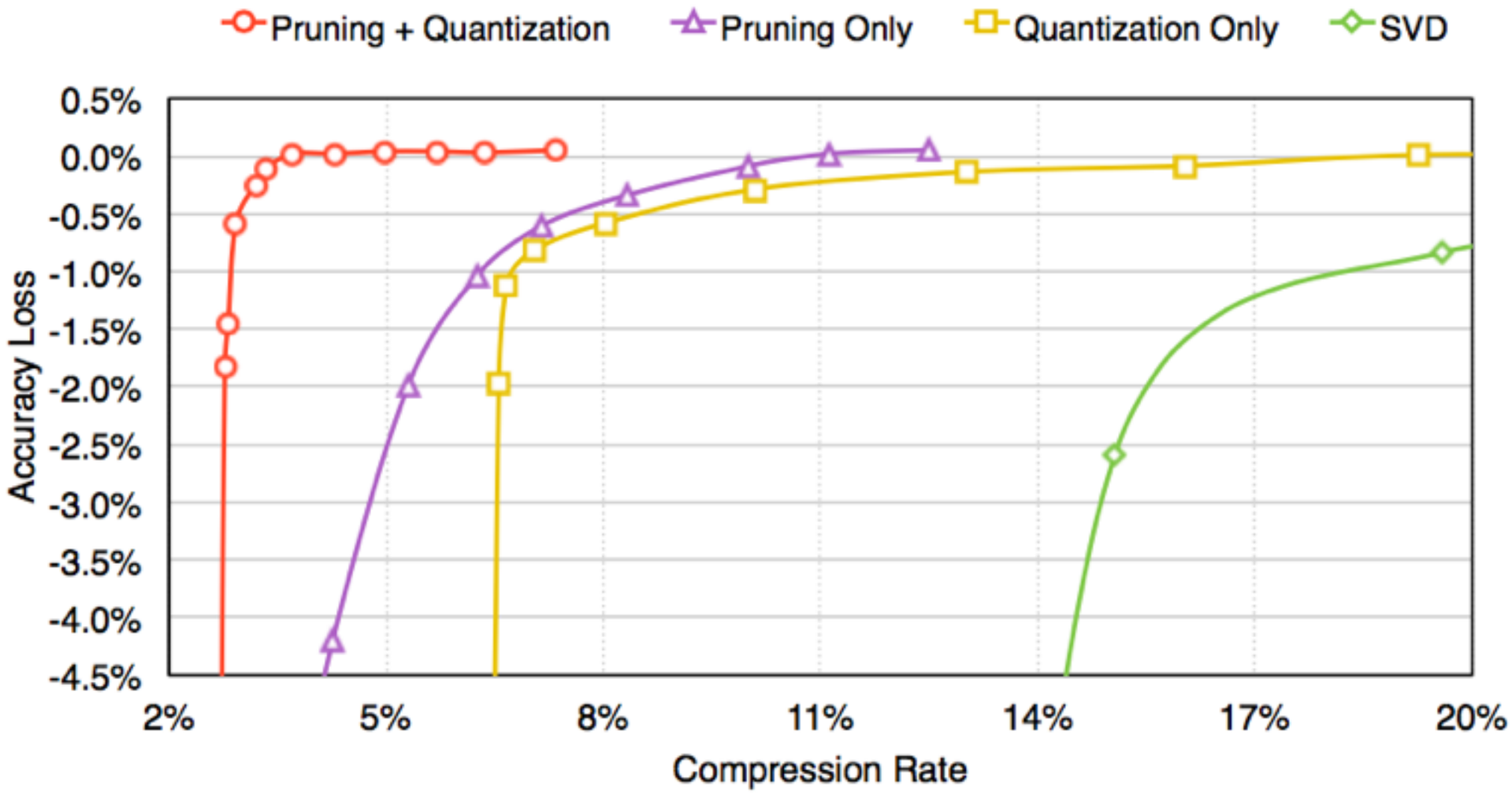
Deep Compression Result on 4 Convnets

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**×
VGG16 Ref	31.50%	11.32%	552 MB	
VGG16 Compressed	31.17%	10.91%	**11.3 MB**	**49**×

Table 1: The compression pipeline can save 35× to 49× parameter storage with no drop in predictive performance

Han et al. Deep Compression: Compressing Deep Neural Networks with Pruning, Trained Quantization and Huffman Coding, ICLR 2016

Result: AlexNet



Han et al. Deep Compression: Compressing Deep Neural Networks with Pruning, Trained Quantization and Huffman Coding, ICLR 2016

AlexNet: Breakdown

Layer	#Weights	Weights% (P)	Weigh bits (P+Q)	Weight bits (+H)	Index bits (P+Q)	Index bits (+H)	Compress rate (P+Q)	Compress rate (P+Q+H)
conv1	35K	84%	8	6.3	4	1.2	32.6%	20.53%
conv2	307K	38%	8	5.5	4	2.3	14.5%	9.43%
conv3	885K	35%	8	5.1	4	2.6	13.1%	8.44%
conv4	663K	37%	8	5.2	4	2.5	14.1%	9.11%
conv5	442K	37%	8	5.6	4	2.5	14.0%	9.43%
fc6	38M	9%	5	3.9	4	3.2	3.0%	2.39%
fc7	17M	9%	5	3.6	4	3.7	3.0%	2.46%
fc8	4M	25%	5	4	4	3.2	7.3%	5.85%
total	61M	11%	5.4	4	4	3.2	3.7%	2.88%

Table 4: Compression Statistics for Alexnet. P: pruning, Q:quantization, H:Huffman Encoding

Han et al. Deep Compression: Compressing Deep Neural Networks with Pruning, Trained Quantization and Huffman Coding, ICLR 2016

The Big Gun:

New Network Topology

+ Deep Compression

Iandola, Han,et al. SqueezeNet: AlexNet-level accuracy with 50x fewer parameters and <1MB model size

New Network Topology + Deep Compression

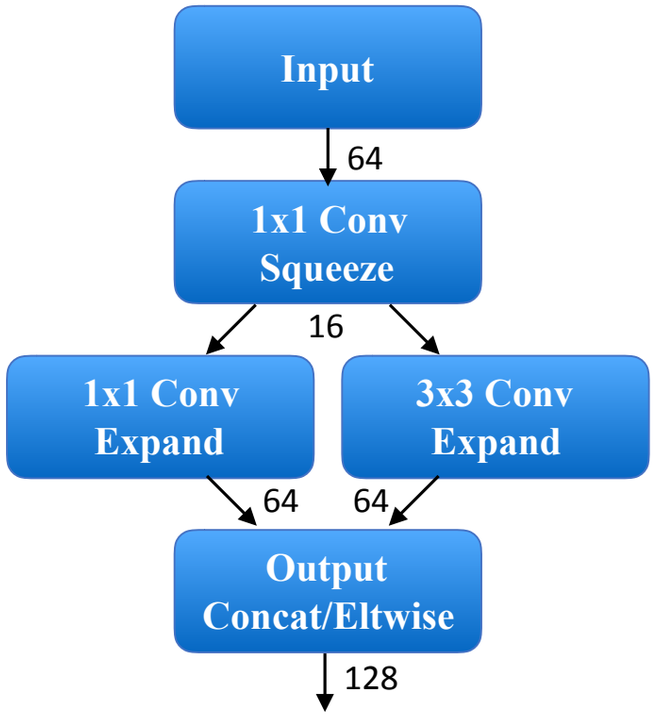


Fig 1: SqueezeNet architecture

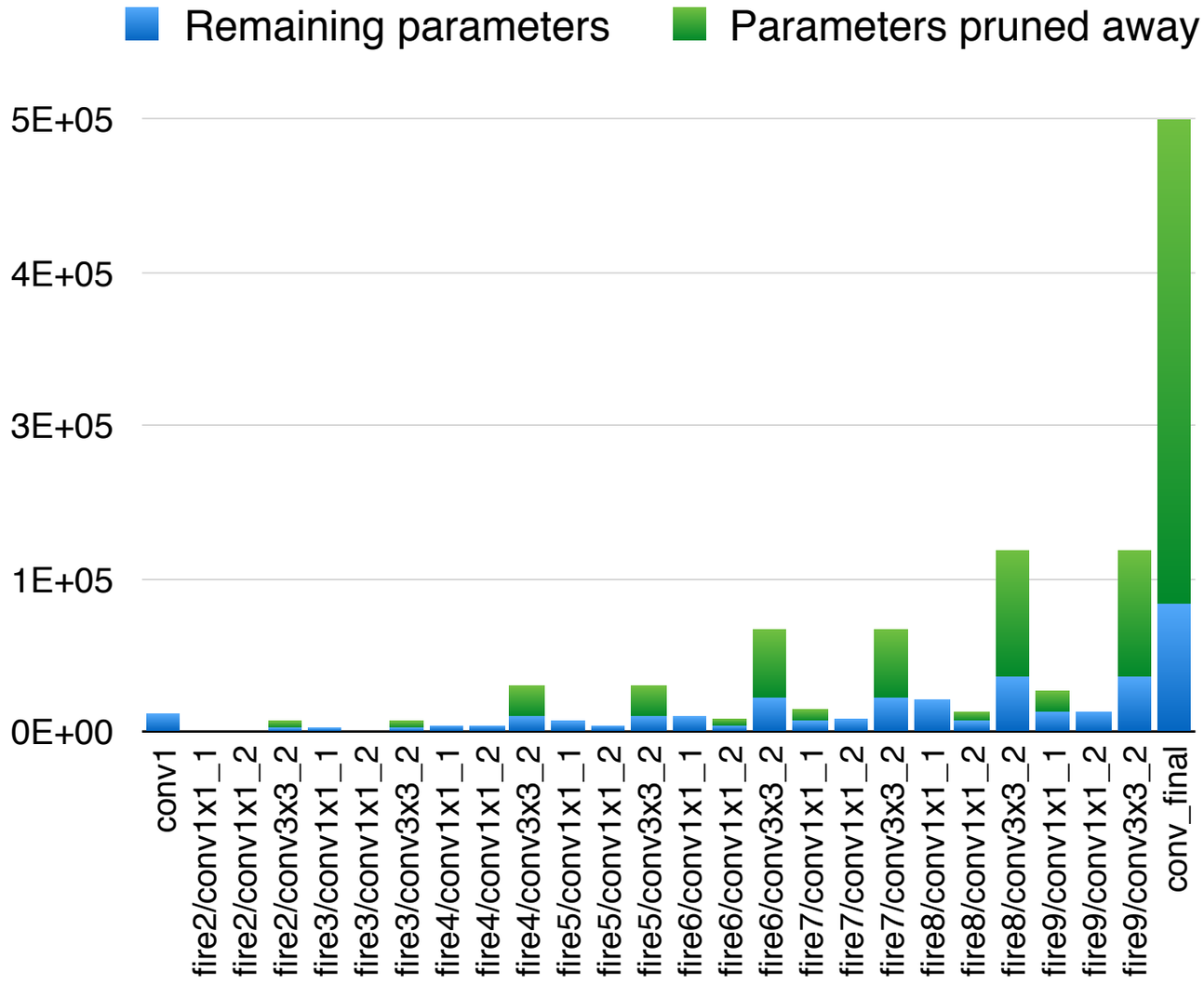
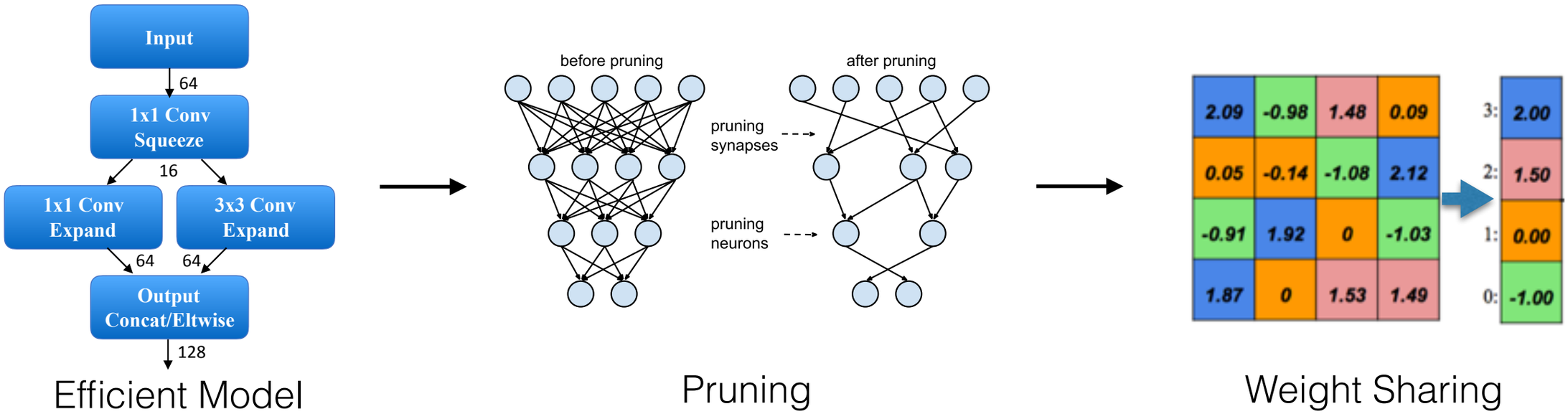


Fig 2. Deep compression is compatible with even extreme efficient network architecture such as SqueezeNet: It can be pruned 3x, quantized to 6bit w/o loss of accuracy.

Iandola, Han,et al. “SqueezeNet: AlexNet-level accuracy with 50x fewer parameters and <1MB model size” arXiv 2016

470KB model, AlexNet-accuracy

CNN architecture	Compression Approach	Data Type	Original → Compressed Model Size	Reduction in Model Size vs. AlexNet	Top-1 ImageNet Accuracy	Top-5 ImageNet Accuracy
AlexNet	None (baseline)	32 bit	240MB	1x	57.2%	80.3%
AlexNet	SVD [3]	32 bit	240MB → 48MB	5x	56.0%	79.4%
AlexNet	Network Pruning [4]	32 bit	240MB → 27MB	9x	57.2%	80.3%
AlexNet	Deep Compression [5]	5-8 bit	240MB → 6.9MB	35x	57.2%	80.3%
SqueezeNet (ours)	None	32 bit	4.8MB	**50x**	57.5%	80.3%
SqueezeNet (ours)	Deep Compression	8 bit	4.8MB → 0.66MB	**363x**	57.5%	80.3%
SqueezeNet (ours)	Deep Compression	6 bit	4.8MB → 0.47MB	**510x**	57.5%	80.3%



Iandola, Han,et al. "SqueezeNet: AlexNet-level accuracy with 50x fewer parameters and <1MB model size" arXiv 2016

470KB model, AlexNet-accuracy

https://github.com/songhan/SqueezeNet_compressed

Iandola, Han,et al. “SqueezeNet: AlexNet-level accuracy with 50x fewer parameters and <1MB model size” arXiv 2016

Smaller DNN means…

(1) Less communication across servers during **distributed training**.

(2) Easier to download from **App Store**.

(3) Less bandwidth to update model to an **autonomous car**.

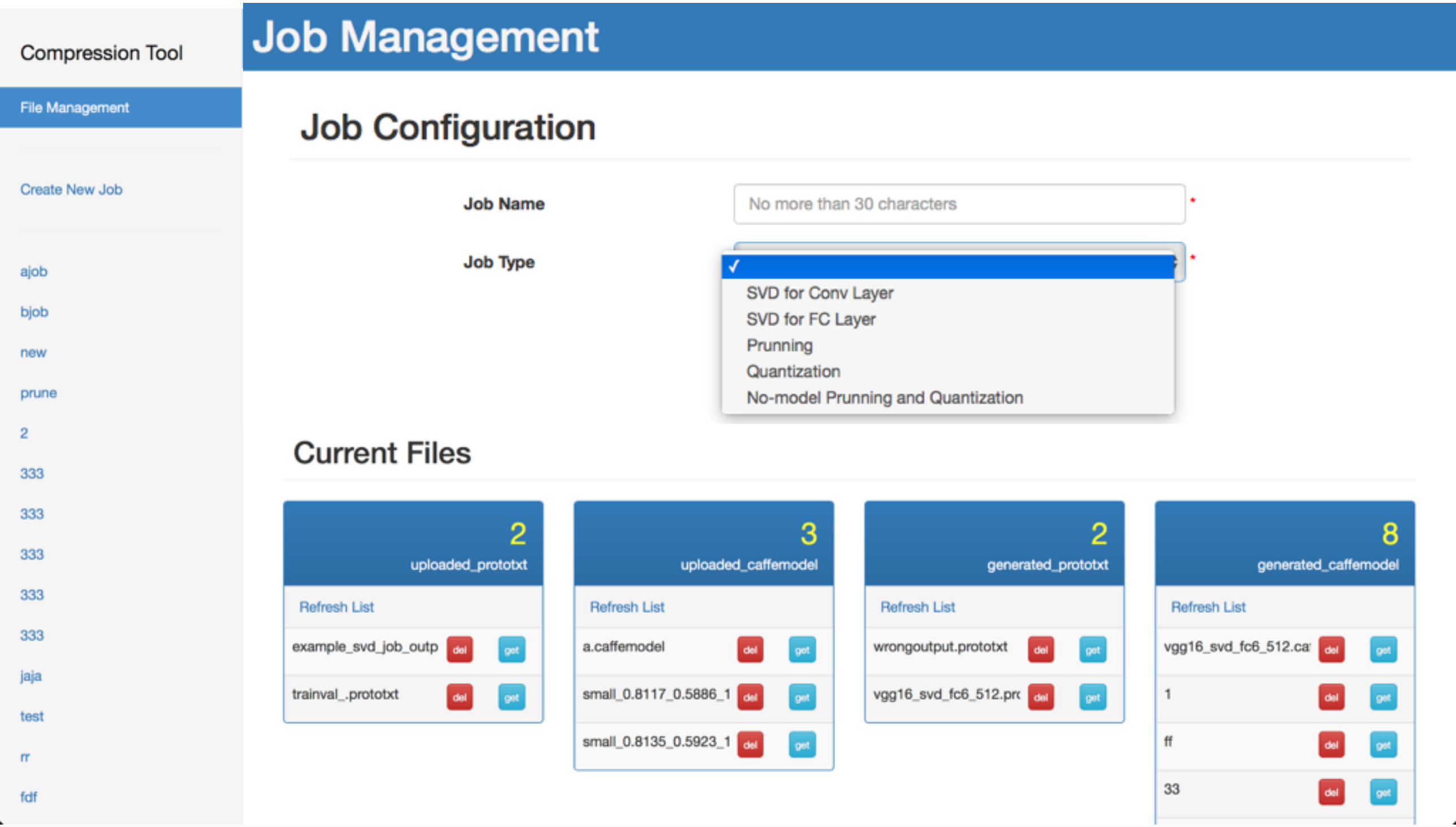
(4) Easier to deploy on **embedded hardware** with limited memory.

A Model Compression Tool for App Developers

deepcompression.net is under construction

- **Easy Version (done)**:
  - ✓ No training needed
  - ✓ Fast (3 minutes)
  - x 5x - 10x compression rate
  - x 1% loss of accuracy
- **Advanced Version (todo):**
  - ✓ 35x - 50x compression rate
  - ✓ no loss of accuracy
  - x Training is needed
  - x Slow

DeepCompression.net



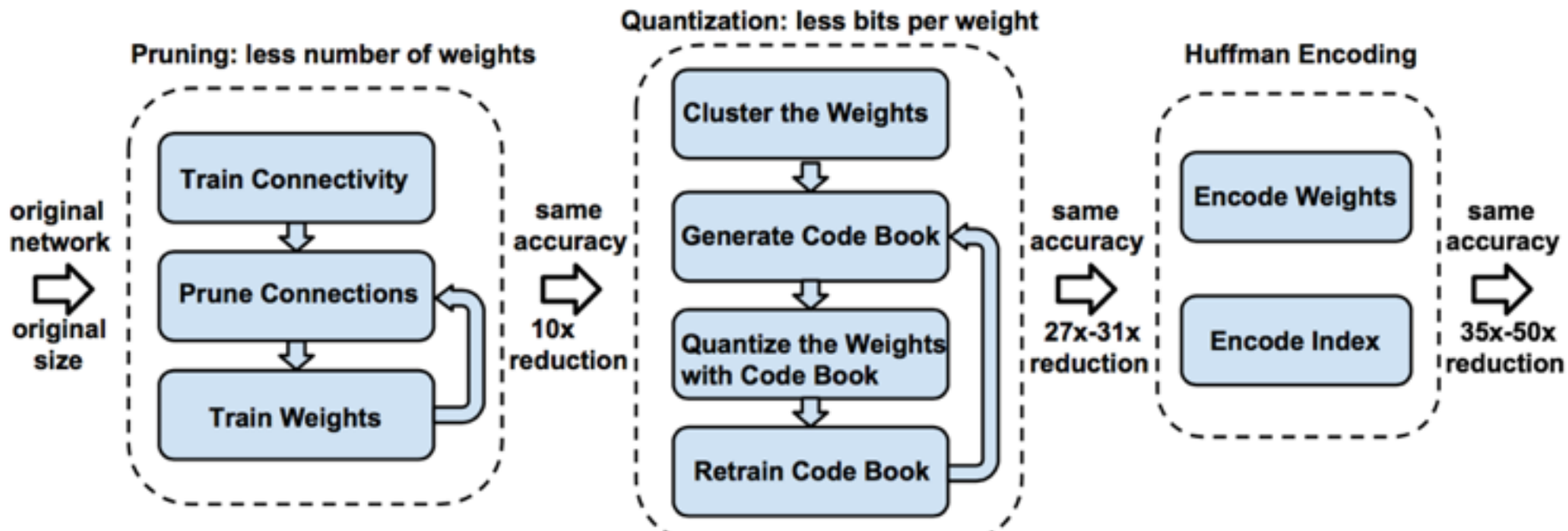
DeepCompression.net

Provides a trial account for GTC attendees:

- Username: deepcompression
- Password: songhan

welcome your feedback!

Conclusion



- We have presented a method to compress neural networks without affecting accuracy by finding the right connections and quantizing the weights.
- Pruning the unimportant connections => quantizing the network and enforce weight sharing => apply Huffman encoding.
- We highlight our experiments on ImageNet, and reduced the weight storage by 35×, VGG16 by 49×, without loss of accuracy.
- Now weights can fit in cache

Part2: SqueezeNet++ ——CNN Design Space Exploration

Song Han
CVA group, Stanford University
Apr 7, 2016

Iandola, Han,et al. “SqueezeNet: AlexNet-level accuracy with 50x fewer parameters and <0.5MB model size”, submitted to ECCV

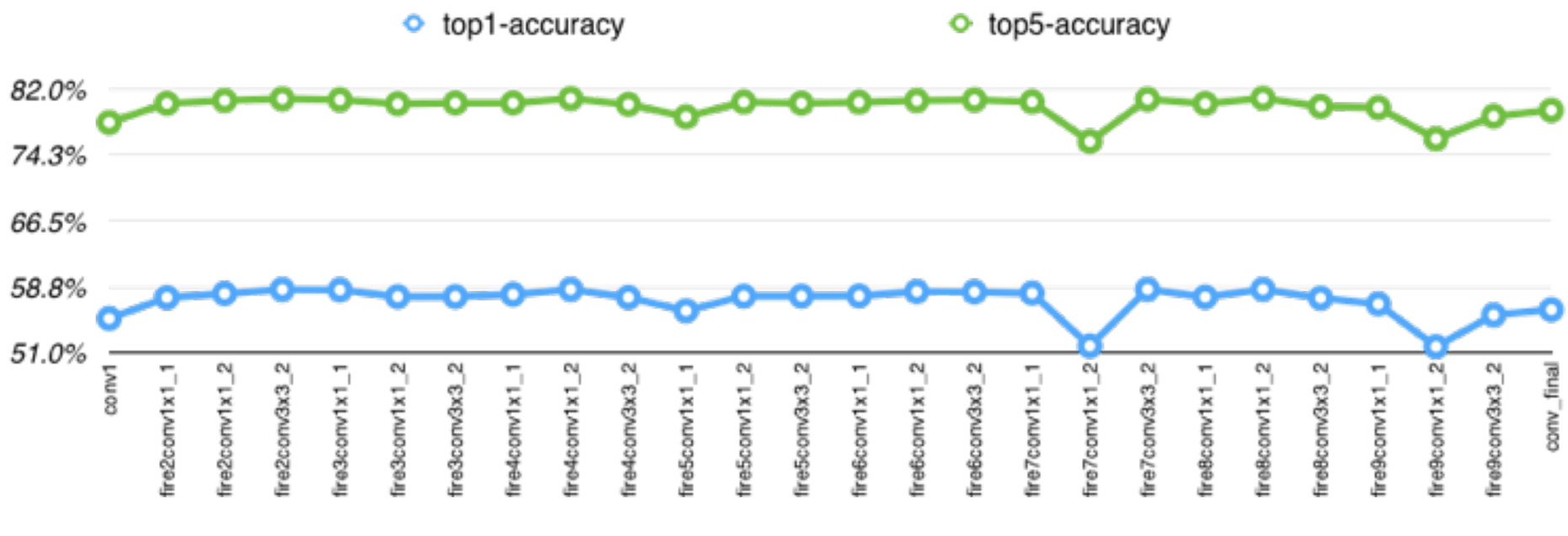
Motivation: How to choose so many architectural dimensions?

layer name/type	output size	filter size / stride (if not a fire layer)	depth	s_{1x1} (#1x1 squeeze)	e_{1x1} (#1x1 expand)	e_{3x3} (#3x3 expand)	s_{1x1} sparsity	e_{1x1} sparsity	e_{3x3} sparsity	# bits	#parameter before pruning	#parameter after pruning
input image	224x224x3										-	-
conv1	111x111x96	7x7/2 (x96)	1				100% (7x7)			6bit	14,208	14,208
maxpool1	55x55x96	3x3/2	0									
fire2	55x55x128		2	16	64	64	100%	100%	**33%**	6bit	11,920	5,746
fire3	55x55x128		2	16	64	64	100%	100%	**33%**	6bit	12,432	6,258
fire4	55x55x256		2	32	128	128	100%	100%	**33%**	6bit	45,344	20,646
maxpool4	27x27x256	3x3/2	0									
fire5	27x27x256		2	32	128	128	100%	100%	**33%**	6bit	49,440	24,742
fire6	27x27x384		2	48	192	192	100%	**50%**	**33%**	6bit	104,880	44,700
fire7	27x27x384		2	48	192	192	**50%**	100%	**33%**	6bit	111,024	46,236
fire8	27x27x512		2	64	256	256	100%	**50%**	**33%**	6bit	188,992	77,581
maxpool8	13x12x512	3x3/2	0									
fire9	13x13x512		2	64	256	256	**50%**	100%	**30%**	6bit	197,184	77,581
conv10	13x13x1000	1x1/1 (x1000)	1				**20%** (3x3)			6bit	513,000	103,400
avgpool10	1x1x1000	13x13/1	0									
	activations		parameters				compression info				1,248,424 (total)	**421,098** (total)

Table 1. SqueezeNet architectural dimensions.

Micro-Architecture DSX

Use sensitive analysis

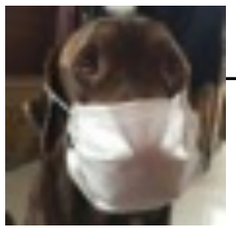


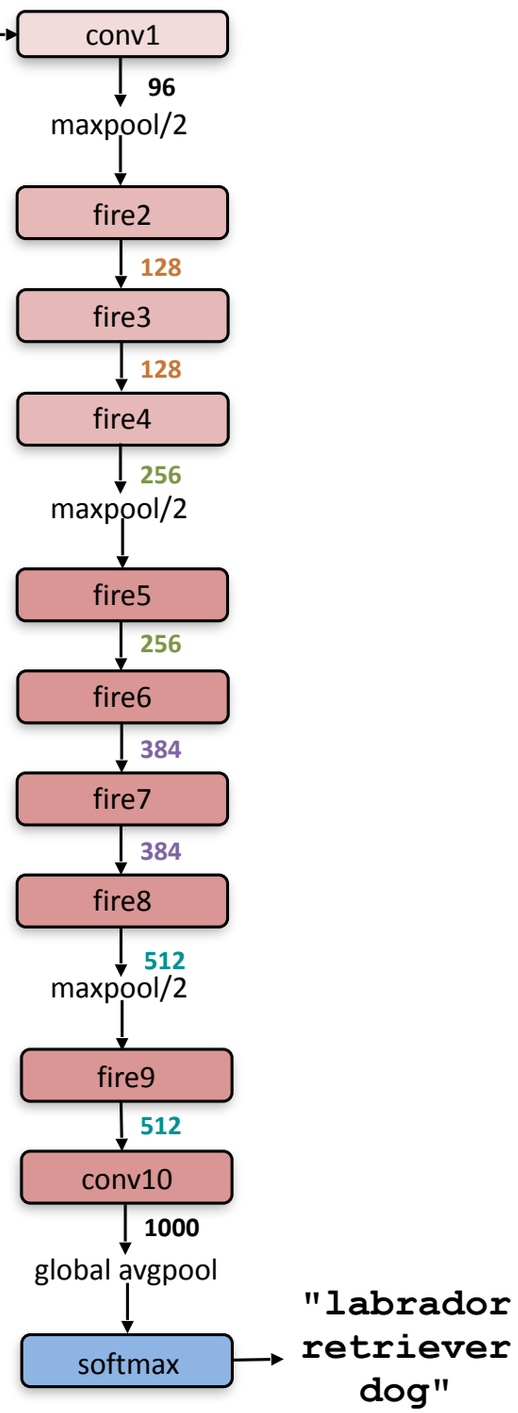
- Micro-Architecture: how to size the layers: 64? 128? 256?
- Sensitivity: Pruning 50% weights for a single layer and measure the accuracy.
- => Sensitivity analysis helps sizing the number of parameters in a layer.

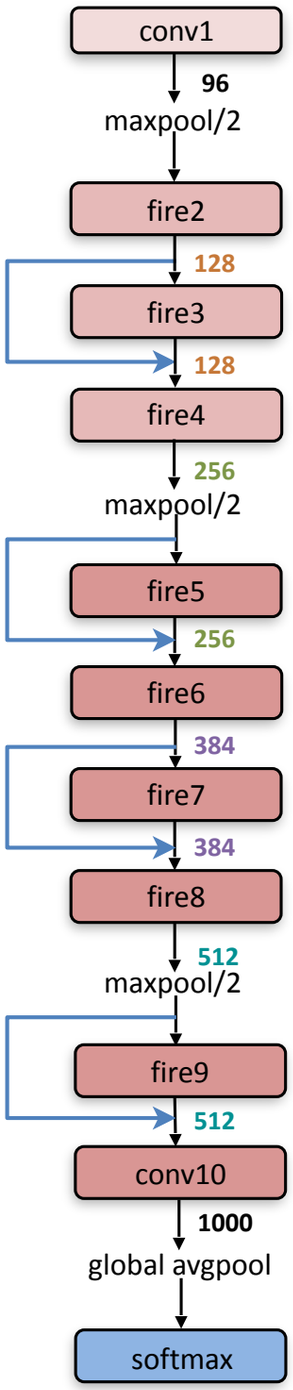
Architecture	Top-1 Accuracy	Top-5 Accuracy	Model Size
SqueezeNet	57.5%	80.3%	4.8MB
SqueezeNet++	**59.5%**	**81.5%**	7.1MB

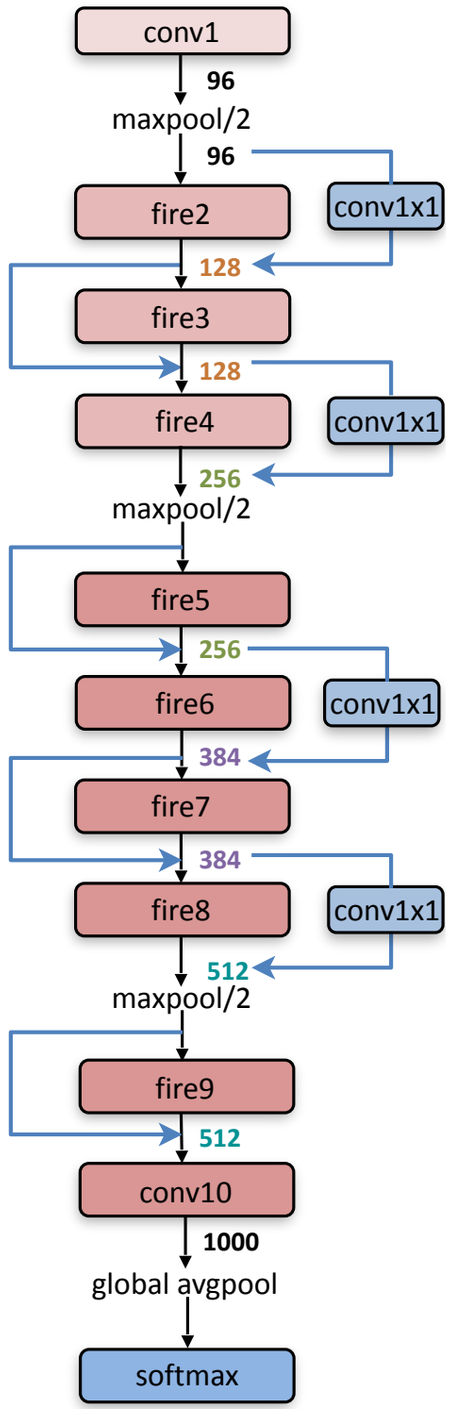
Iandola, Han,et al. “SqueezeNet: AlexNet-level accuracy with 50x fewer parameters and <0.5MB model size”, submitted to ECCV

Macro-Architecture DSX









Iandola, Han,et al. “SqueezeNet: AlexNet-level accuracy with 50x fewer parameters and <0.5MB model size”, submitted to ECCV

Macro-Architecture DSX

vanilla Fire module

simple bypass

complex bypass with 1x1 Conv

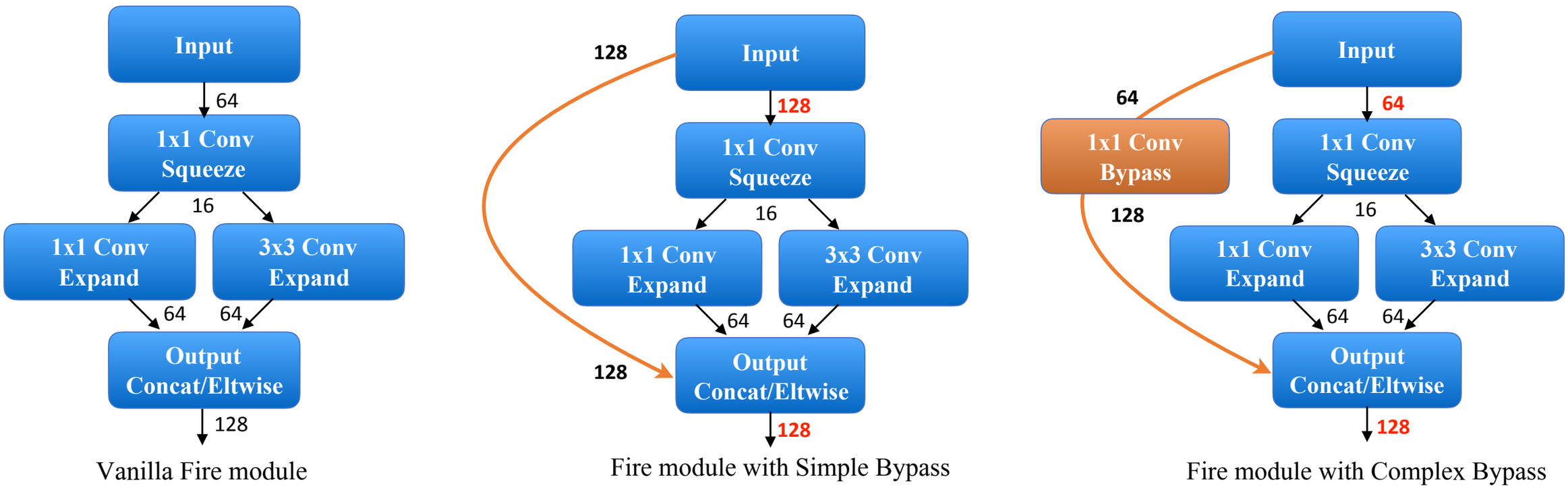


Table 3. SqueezeNet accuracy and model size using different macroarchitecture

Architecture	Top-1 Accuracy	Top-5 Accuracy	Model Size
Vanilla SqueezeNet	57.5%	80.3%	4.8MB
SqueezeNet + Simple Bypass	**60.4%**	**82.5%**	4.8MB
SqueezeNet + Complex Bypass	58.8%	82.0%	7.7MB

Iandola, Han,et al. “SqueezeNet: AlexNet-level accuracy with 50x fewer parameters and <0.5MB model size”, submitted to ECCV

DSD Training (Dense-Sparse-Dense) improves accuracy

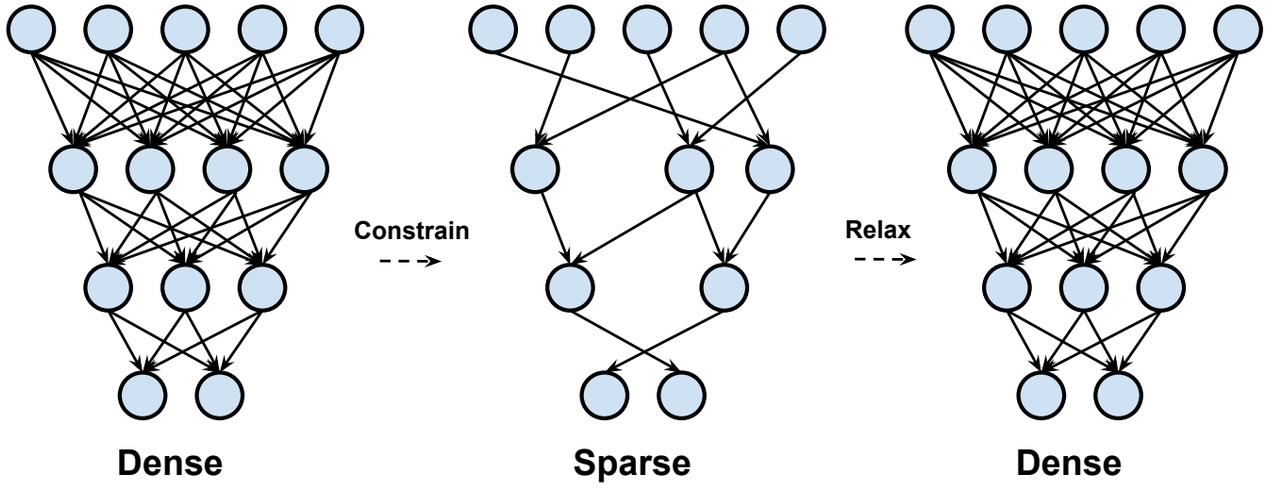


Table 5. Improving accuracy with dense→sparse→dense (DSD) training.

Architecture	Top-1 Accuracy	Top-5 Accuracy	Model Size
SqueezeNet	57.5%	80.3%	4.8MB
SqueezeNet (DSD)	**61.8%**	**83.5%**	4.8MB

- dense→sparse→dense (DSD) training yielded 4.3% higher accuracy.
- Sparsity is a form of regularization. Once the network arrives at a local minimum given the sparsity constraint, relaxing the constraint gives the network more freedom to escape the saddle point and arrive at a higher-accuracy local minimum.
- Regularizing models by intermittently pruning parameters throughout training would be an interesting area of future work.

Iandola, Han,et al. “SqueezeNet: AlexNet-level accuracy with 50x fewer parameters and <0.5MB model size”, submitted to ECCV

Design Space Exploration Conclusion

- SqueezeNet++: sizing the layers with sensitivity analysis
- Use even kernel
- SqueezeNet +simple + complex bypass layer
- DSD training: improves accuracy by 4.3%

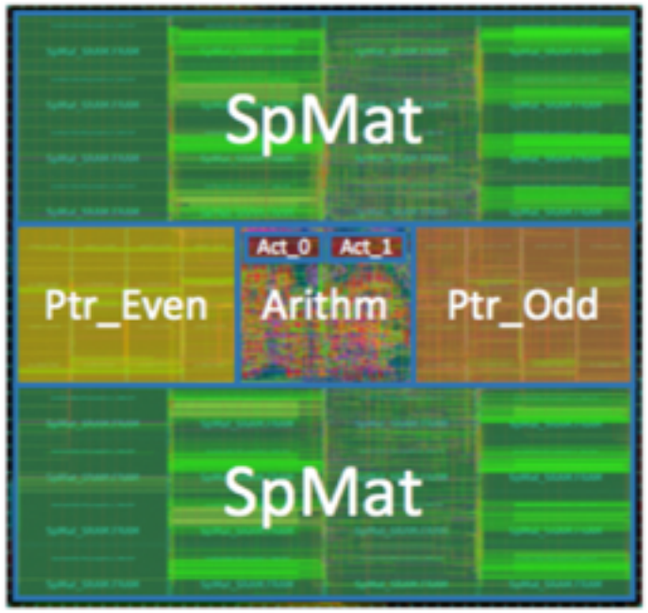
Part 3:

EIE: Efficient Inference Engine on Compressed Deep Neural Network

Song Han
CVA group, Stanford University
Apr 7, 2016

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

ASIC Accelerator on Compressed DNN



- **sparse, indirectly indexed, weight shared** MxV accelerator.

Offline
No dependency on network connection

Real Time
No network delay high frame rate

Low Power
High energy efficiency that preserves battery

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

Distribute Storage and Processing

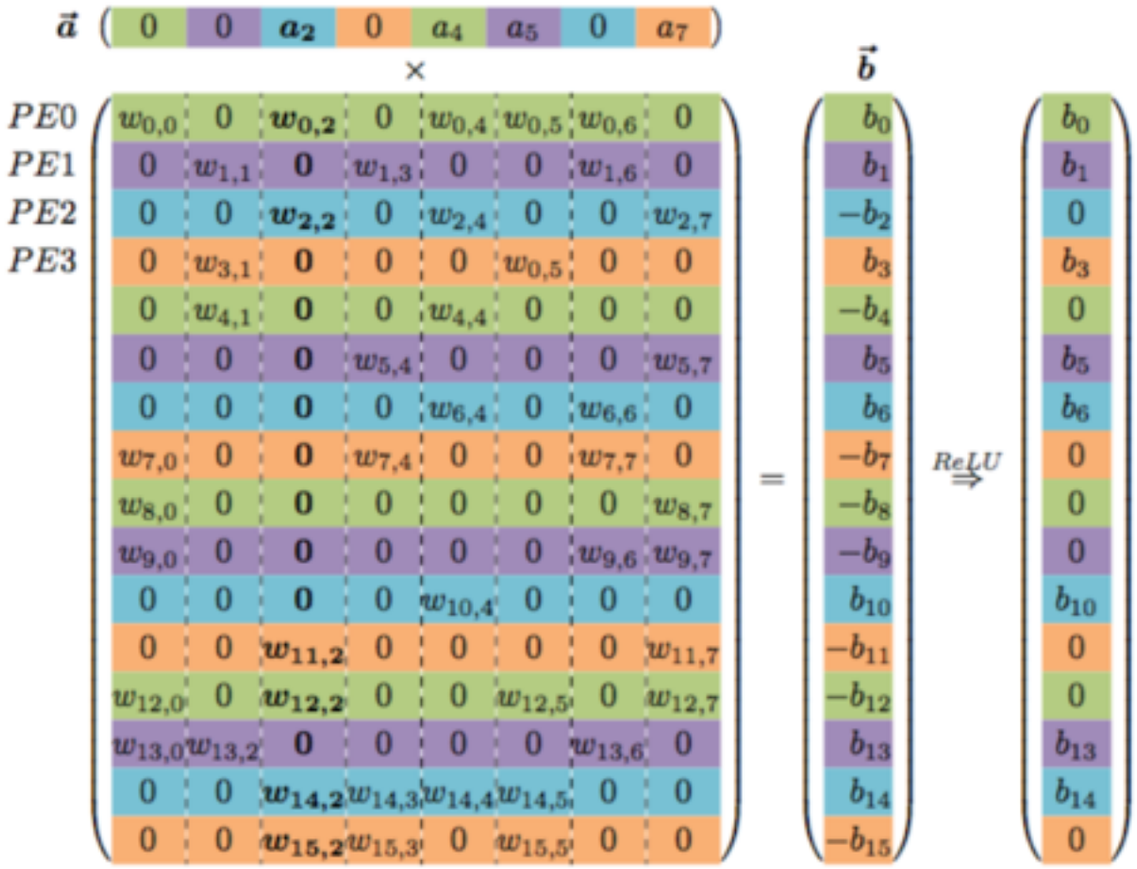
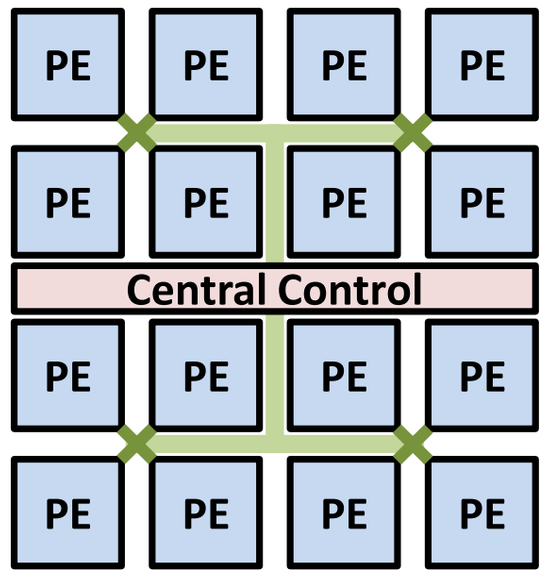
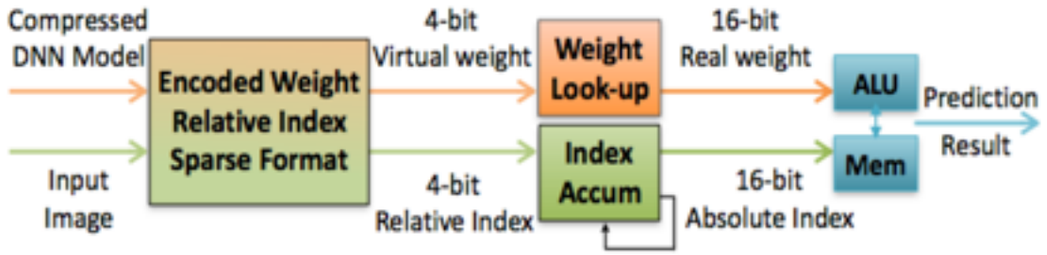


Figure 2: Matrix W and vectors a and b are interleaved over 4 PEs. Elements of the same color are stored in the same PE.





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

Evaluation

1. Cycle-accurate C++ simulator. Two abstract methods: Propagate and Update. Used for DSE and verification.

2. RTL in Verilog, verified its output result with the golden model in Modelsim.

3. Synthesized EIE using the Synopsys Design Compiler (DC) under the TSMC 45nm GP standard VT library with worst case PVT corner.

4. Placed and routed the PE using the Synopsys IC compiler (ICC). We used Cacti to get SRAM area and energy numbers.

5. Annotated the toggle rate from the RTL simulation to the gate-level netlist, which was dumped to switching activity interchange format (SAIF), and estimated the power using Prime-Time PX.

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

Layout of an EIE PE

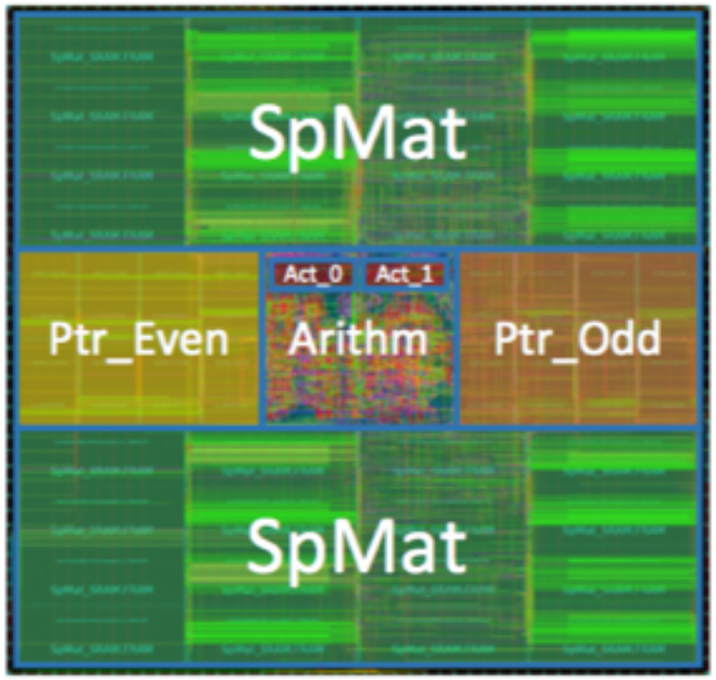


Figure 7: Layout of one PE in EIE under TSMC 45nm process.

	Power (mW)	(%)	Area (μm^2)	(%)
Total	9.157		638,024	
memory	5.416	(59.15%)	594,786	(93.22%)
clock network	1.874	(20.46%)	866	(0.14%)
register	1.026	(11.20%)	9,465	(1.48%)
combinational	0.841	(9.18%)	8,946	(1.40%)
filler cell			23,961	(3.76%)
Act_queue	0.112	(1.23%)	758	(0.12%)
PtrRead	1.807	(19.73%)	121,849	(19.10%)
SpmatRead	4.955	(54.11%)	469,412	(73.57%)
ArithmUnit	1.162	(12.68%)	3,110	(0.49%)
ActRW	1.122	(12.25%)	18,934	(2.97%)
filler cell			23,961	(3.76%)

Table 2: The implementation results of one PE in EIE and the breakdown by component type (line 3-7), by module (line 8-13). The critical path of EIE is 1.15*ns*

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

Baseline and Benchmark

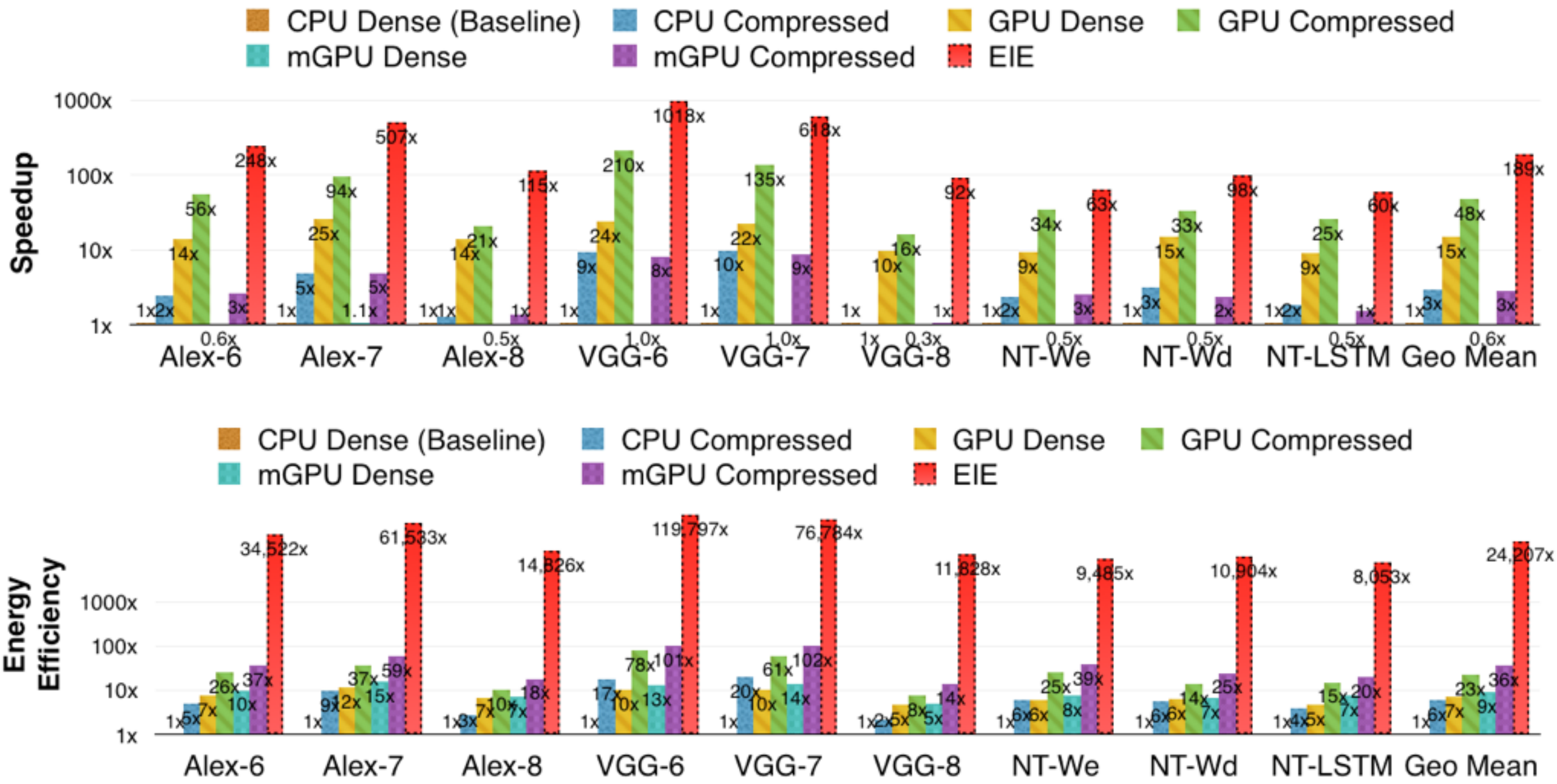
- CPU: Intel Core-i7 5930k
- GPU: NVIDIA TitanX GPU
- Mobile GPU: Jetson TK1 with NVIDIA

Table 3: Benchmark from state-of-the-art DNN models

Layer	Size	Weight%	Act%	FLOP%	Description
Alex-6	9216, 4096	9%	35.1%	3%	Compressed AlexNet [1] for large scale image classification
Alex-7	4096, 4096	9%	35.3%	3%	
Alex-8	4096, 1000	25%	37.5%	10%	
VGG-6	25088, 4096	4%	18.3%	1%	Compressed VGG-16 [3] for large scale image classification and object detection
VGG-7	4096, 4096	4%	37.5%	2%	
VGG-8	4096, 1000	23%	41.1%	9%	
NT-We	4096, 600	10%	100%	10%	Compressed NeuralTalk [7] with RNN and LSTM for automatic image captioning
NT-Wd	600, 8791	11%	100%	11%	
NTLSTM	1201, 2400	10%	100%	11%	

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

Result: Speedup / Energy Efficiency



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

Comparison with other Platforms

Table V
COMPARISON WITH EXISTING HARDWARE PLATFORMS FOR DNNS

Platform	Core i7	Titan X	Tegra K1	A-Eye [36]	DaDianNao [5]	TrueNorth [11]	EIE (ours)
Year	2014	2015	2014	2015	2014	2016	2016
Platform Type	CPU	GPU	mGPU	FPGA	ASIC	ASIC	ASIC
Technology	22nm	28nm	28nm	28nm	28nm	28nm	45nm
Clock (MHz)	3500	1075	852	150	606	Async	800
Memory type	DRAM	DRAM	DRAM	DRAM	eDRAM	SRAM	SRAM
Max DNN model size	<16G	<3G	<500M	<500M	18M	256M	84M
Quantization Stategy	32-bit float	32-bit float	32-bit float	16-bit fixed	16-bit fixed	1-bit fixed	4-bit fixed
Area (mm^2)	356	601	-	-	67.7	430	40.8
M×V Throughput (Frames/s)	162	4,115	173	33	36,900	1,191	82,000
Power(W)	140	250	8.0	9.63	15.97	0.23	0.59
Energy Efficiency (Frames/s/W)	1.16	20.5	21.6	3.4	2,310	5,178	138,983

Where are the savings from?

- Three factors for energy saving:

- **Matrix is compressed by 35×;**
  less work to do; less bricks to carry

- **DRAM => SRAM, no need to go off-chip: 120×;**
  carry bricks from Stanford to Berkeley => Stanford to Palo Alto

- **Sparse activation: 3×;**
  lighter bricks to carry

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

Load Balancing and Scalability

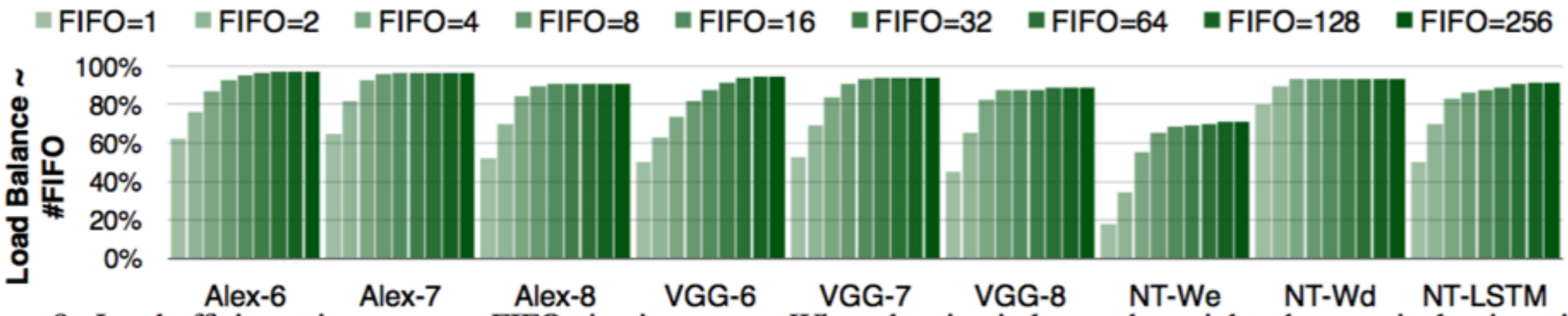


Figure 8: Load efficiency improves as FIFO size increases. When the size is larger than eight, the marginal gain quickly diminishes. So we choose FIFO depth to be eight.

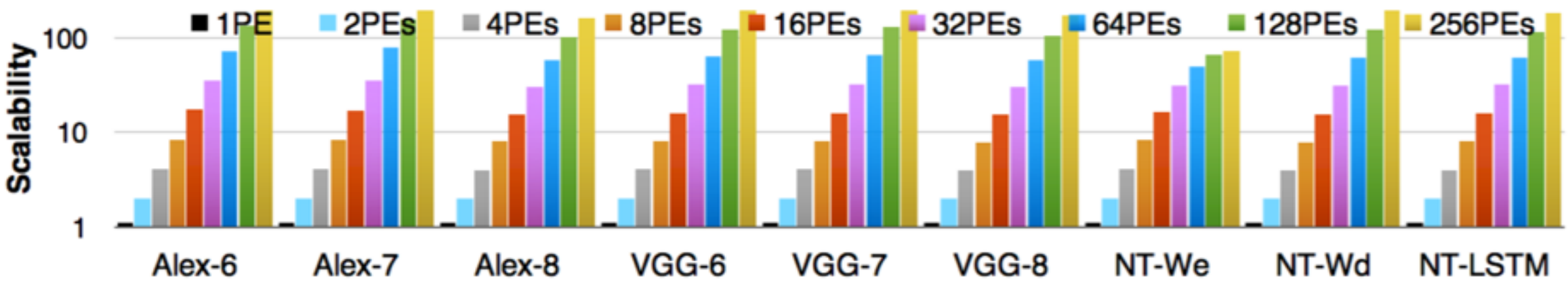
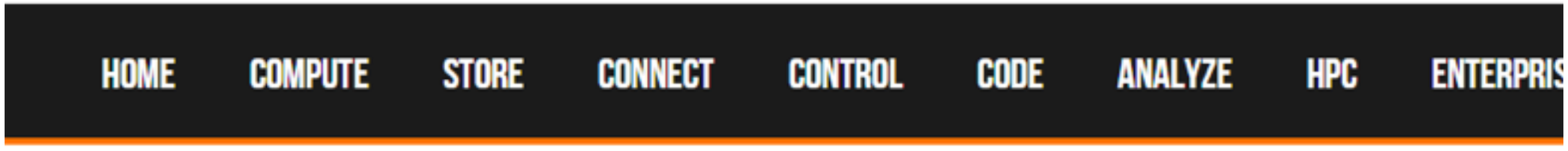


Figure 11: System scalability. The average efficiency of single PE finally decreases as the number of PEs increases. On some very sparse layers, having more PEs initially increases the efficiency a bit.

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

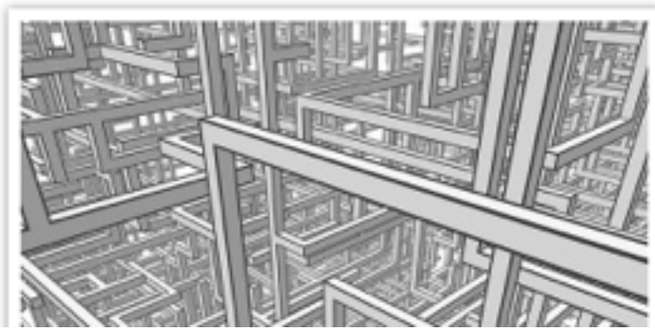
media coverage

TheNextPlatform



EMERGENT CHIP VASTLY ACCELERATES DEEP NEURAL NETWORKS

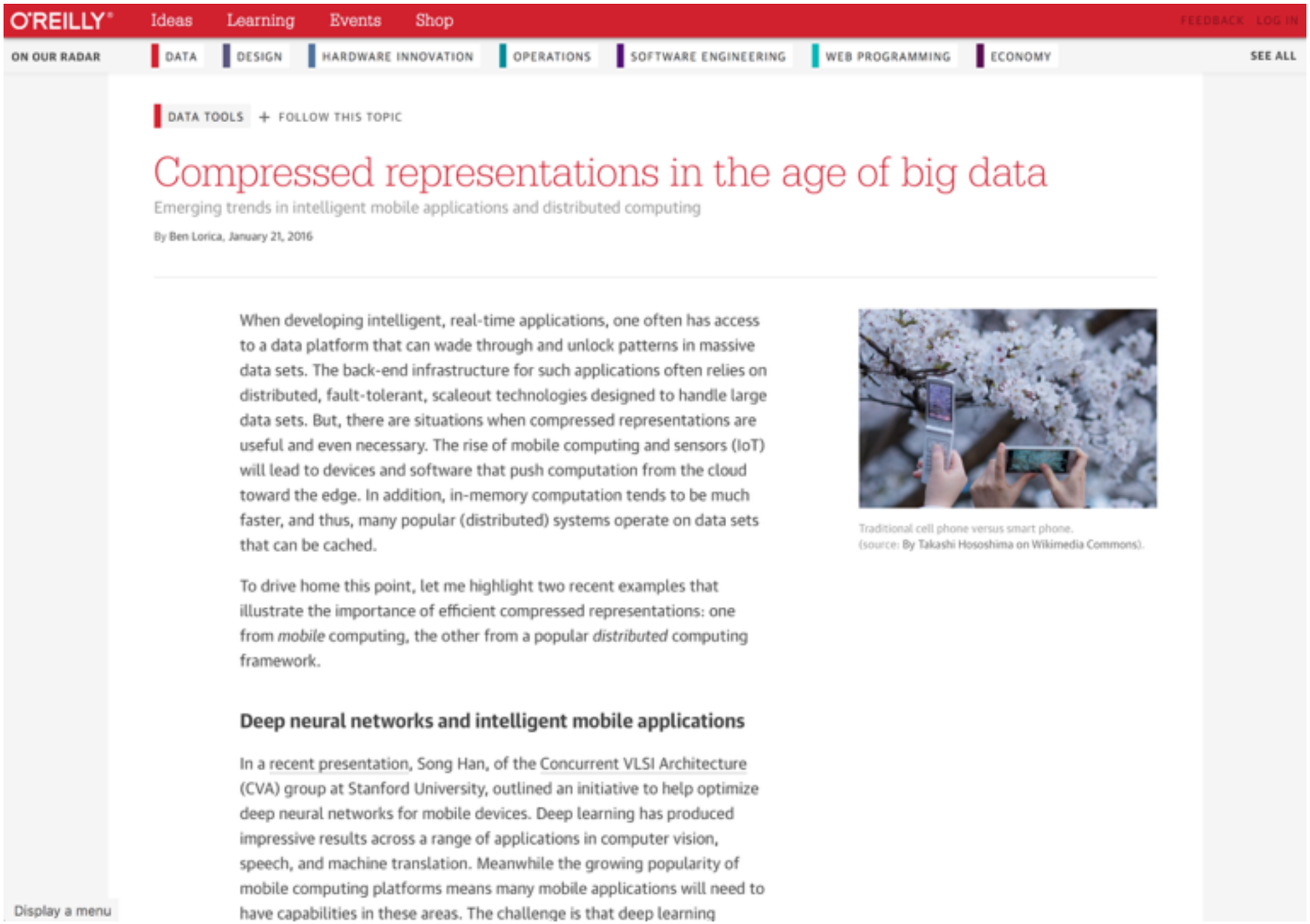
December 8, 2015 Nicole Hemsoth



Stanford University PhD candidate, Song Han, who works under advisor and networking pioneer, Dr. Bill Dally, responded in a most soft-spoken and thoughtful way to the question of whether the coupled software and hardware architecture he developed might change the world.

http://www.nextplatform.com/2015/12/08/emergent-chip-vastly-accelerates-deep-neural-networks/

O'Reilly



O'REILLY® Ideas Learning Events Shop FEEDBACK LOG IN

ON OUR RADAR DATA DESIGN HARDWARE INNOVATION OPERATIONS SOFTWARE ENGINEERING WEB PROGRAMMING ECONOMY SEE ALL

DATA TOOLS + FOLLOW THIS TOPIC

Compressed representations in the age of big data

Emerging trends in intelligent mobile applications and distributed computing

By Ben Lorica, January 21, 2016

When developing intelligent, real-time applications, one often has access to a data platform that can wade through and unlock patterns in massive data sets. The back-end infrastructure for such applications often relies on distributed, fault-tolerant, scaleout technologies designed to handle large data sets. But, there are situations when compressed representations are useful and even necessary. The rise of mobile computing and sensors (IoT) will lead to devices and software that push computation from the cloud toward the edge. In addition, in-memory computation tends to be much faster, and thus, many popular (distributed) systems operate on data sets that can be cached.

To drive home this point, let me highlight two recent examples that illustrate the importance of efficient compressed representations: one from *mobile* computing, the other from a popular *distributed* computing framework.

Deep neural networks and intelligent mobile applications

In a recent presentation, Song Han, of the Concurrent VLSI Architecture (CVA) group at Stanford University, outlined an initiative to help optimize deep neural networks for mobile devices. Deep learning has produced impressive results across a range of applications in computer vision, speech, and machine translation. Meanwhile the growing popularity of mobile computing platforms means many mobile applications will need to have capabilities in these areas. The challenge is that deep learning

Traditional cell phone versus smart phone.
(source: By Takashi Hososhima on Wikimedia Commons).

Display a menu

https://www.oreilly.com/ideas/compressed-representations-in-the-age-of-big-data

TechEmergence

techemergence

Search

TECHNOLOGY INTERVIEWS BUSINESS RESEARCH SUBSCRIBE ABOUT

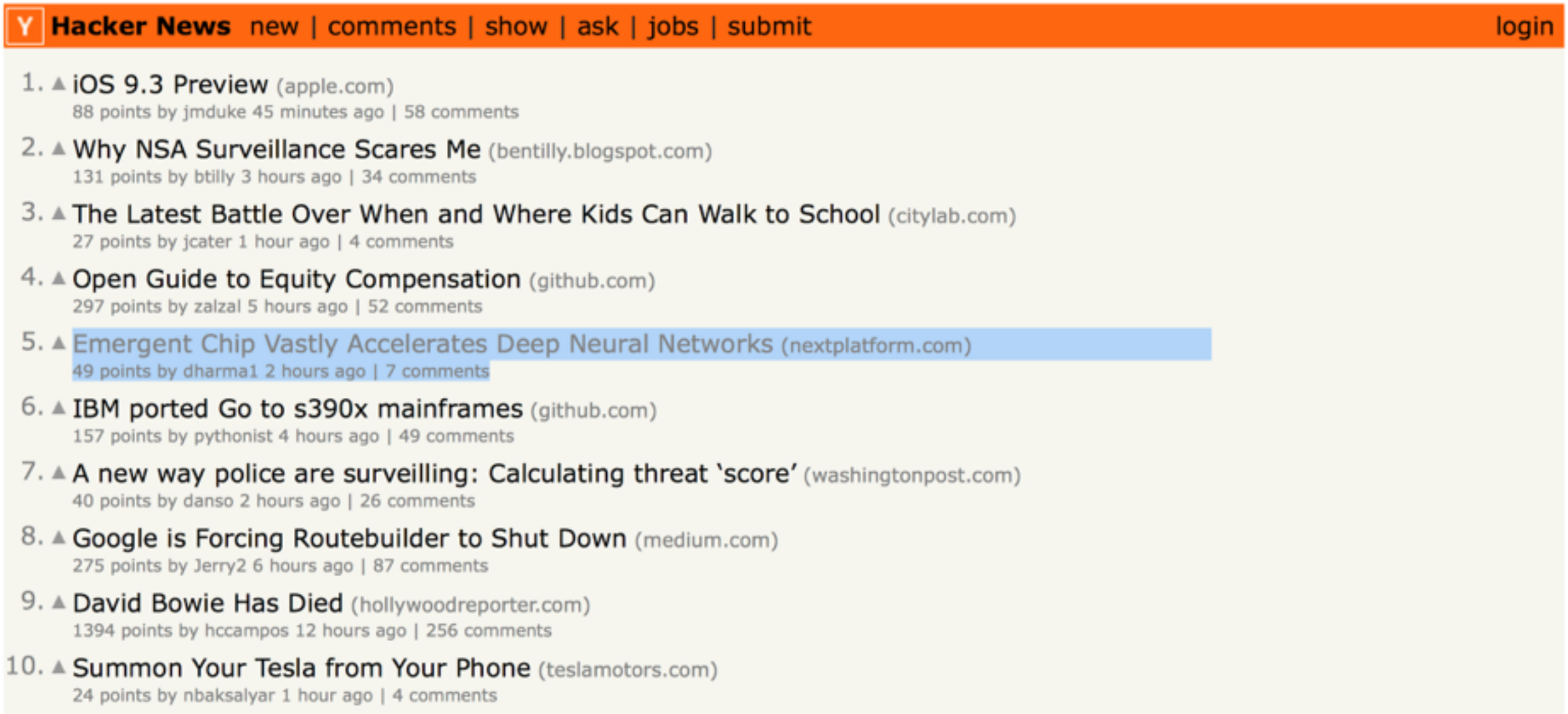
Like a "Limitless" Pill for Deep Neural Networks

DYLLAN FURNESS ✖ DECEMBER 31, 2015

Song Han may have done just that. Once an intern at Google, now a PhD candidate at Stanford University, Han is currently researching under the advisory of networking pioneer Dr Bill Dally. Under Dally's guidance, Han is working to develop a small chip called EIE which is intended to increase the role of static random access memory (SRAM) while scaling down networks to more manageable and efficient sizes in a technique called deep compression – i.e. EIE is sort of like a "*Limitless*" pill for deep neural networks.

http://techemergence.com/a-limitless-pill-for-deep-neural-networks/

Hacker News

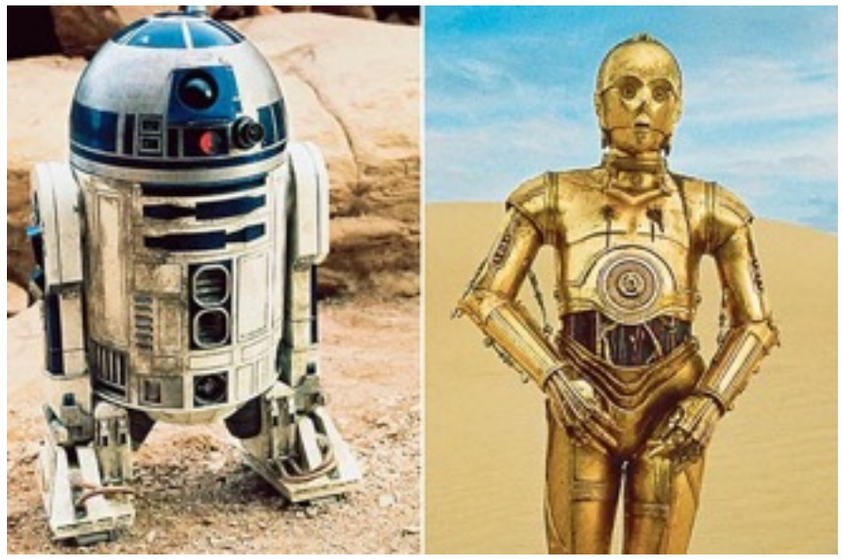


https://news.ycombinator.com/item?id=10881683

Conclusion

- We present EIE, an energy-efficient engine optimized to operate on compressed deep neural networks.
- By leveraging sparsity in both the activations and the weights, EIE reduces the energy needed to compute a typical FC layer by 3,000×.
- With wrapper logic on top of EIE, 1x1 convolution and 3x3 convolution is possible.

Hardware for Deep Learning



PC

Computation

Mobile

Mobile Computation

Intelligent Mobile

Intelligent Mobile Computation

Recap

Model Compression

[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”, ICLR 2016, Deep Learning Symposium, NIPS 2015

Hardware Acceleration

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

CNN Architecture Design Space Exploration

[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 2016 workshop

Thank you!

songhan@stanford.edu

[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”, 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” ECCV’16 submission
[5]. Yao, Han, et.al, “Hardware-friendly convolutional neural network with even-number filter size” ICLR’16 workshop