High-Performance Hardware for Machine Learning

U.C. Berkeley

October 19, 2016

William Dally

NVIDIA Corporation

Stanford University

Machine learning is transforming computing

Speech

Natural Language Understanding

Question Answering

Game Playing (Go)

Vision

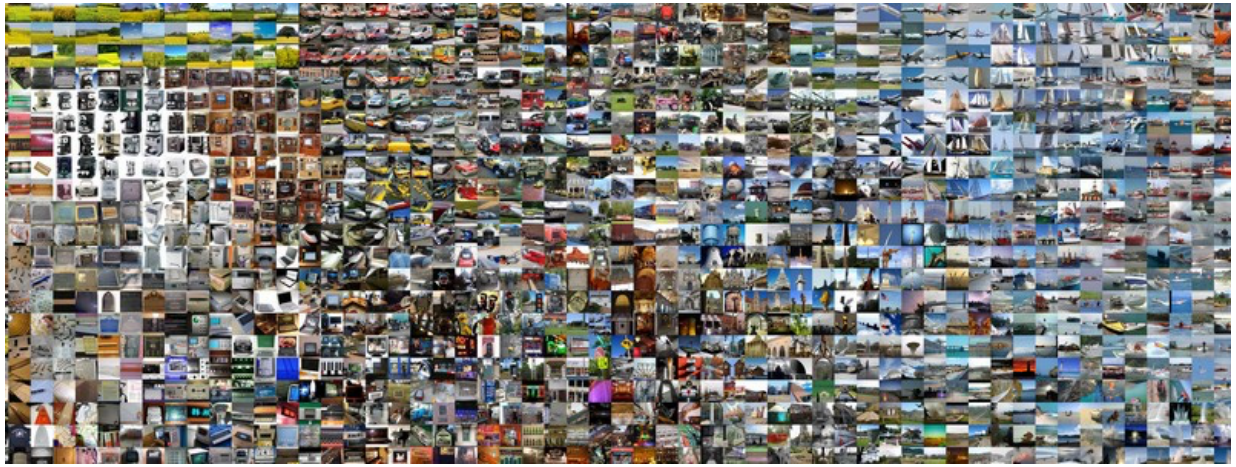
Autonomous Vehicles

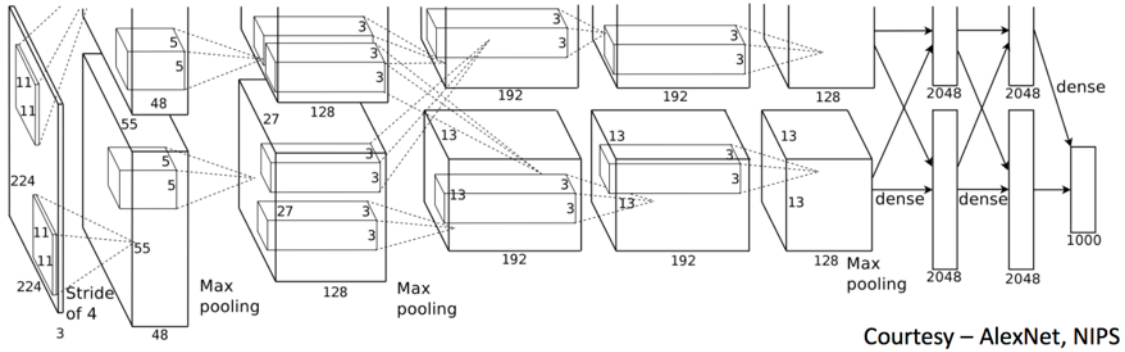
Control

Ad Placement

Whole research fields rendered irrelevant

Hardware and Data enable DNNs





Courtesy – AlexNet, NIPS 2012

The Need for Speed

Important Property of Neural Networks

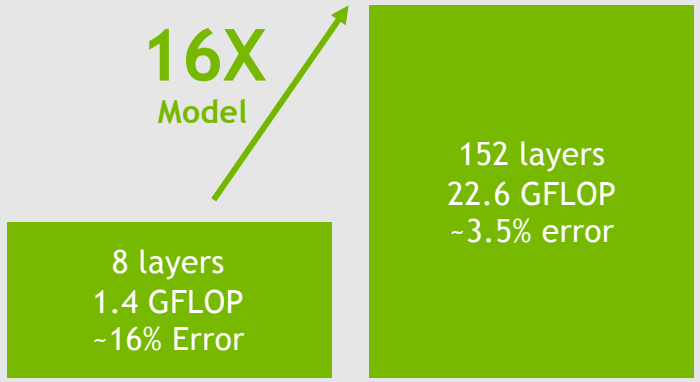
Results get better with

more data +
bigger models +
more computation

(Better algorithms, new insights and improved techniques always help, too!)

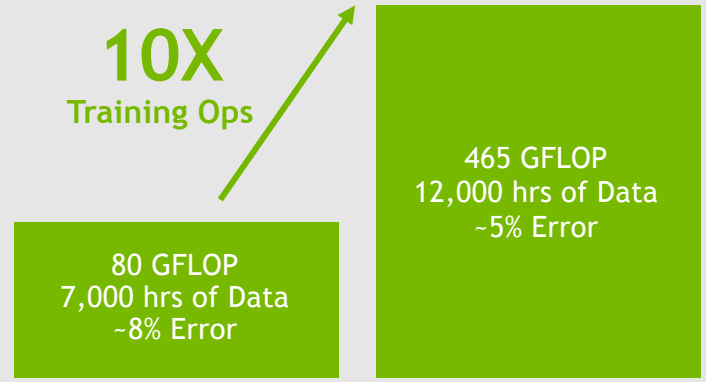
Google

IMAGE RECOGNITION



Microsoft

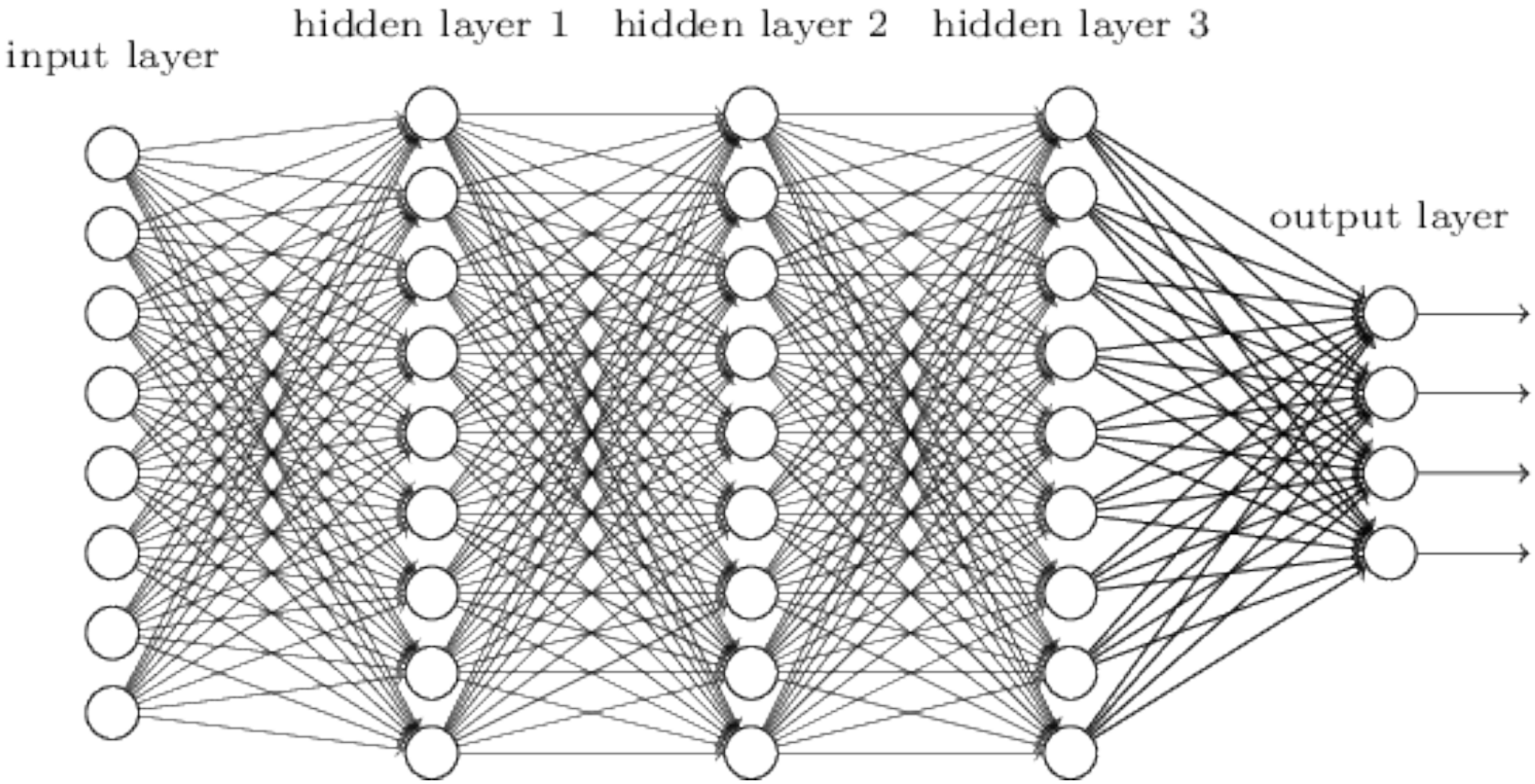
SPEECH RECOGNITION



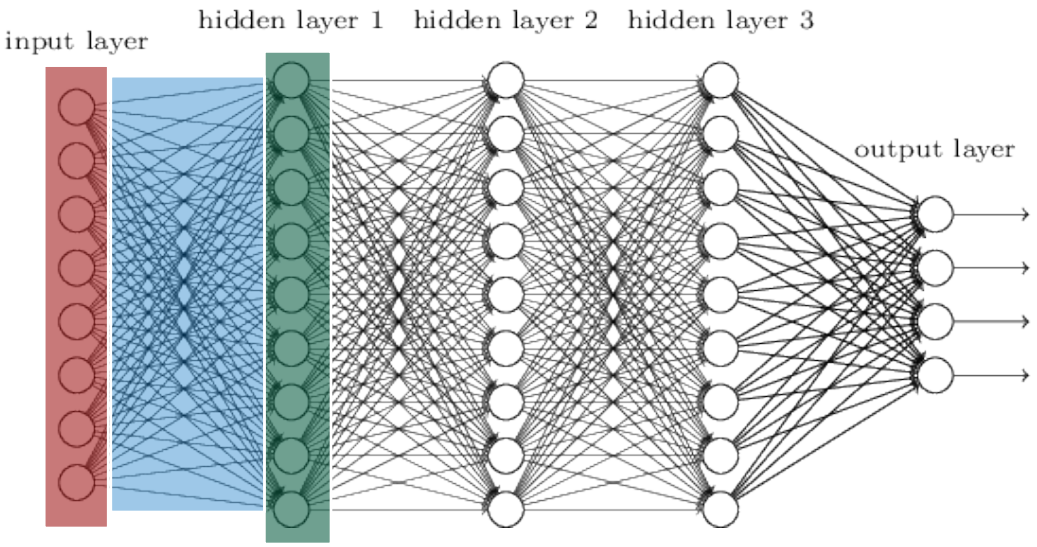
Baidu 百度

DNN primer

WHAT NETWORK? DNNS, CNNS, AND RNNS

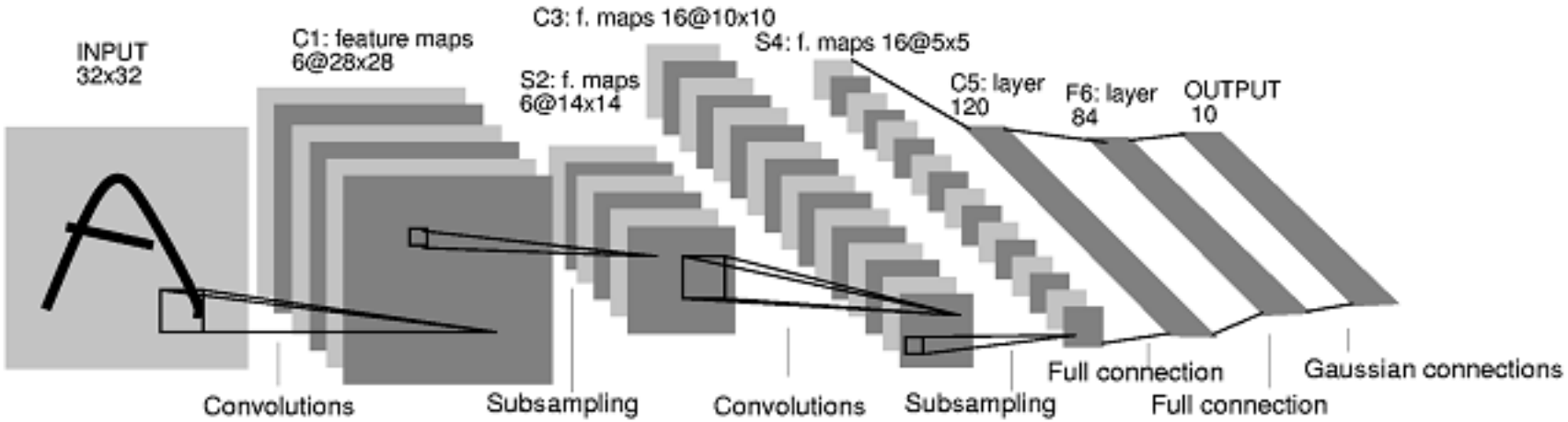


DNN, KEY OPERATION IS DENSE M X V

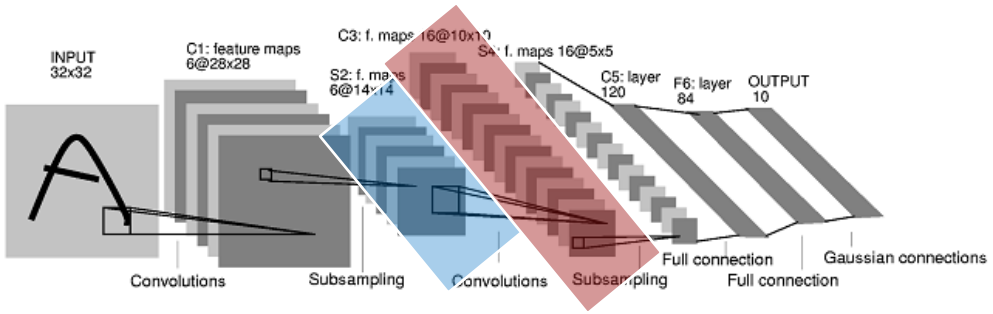


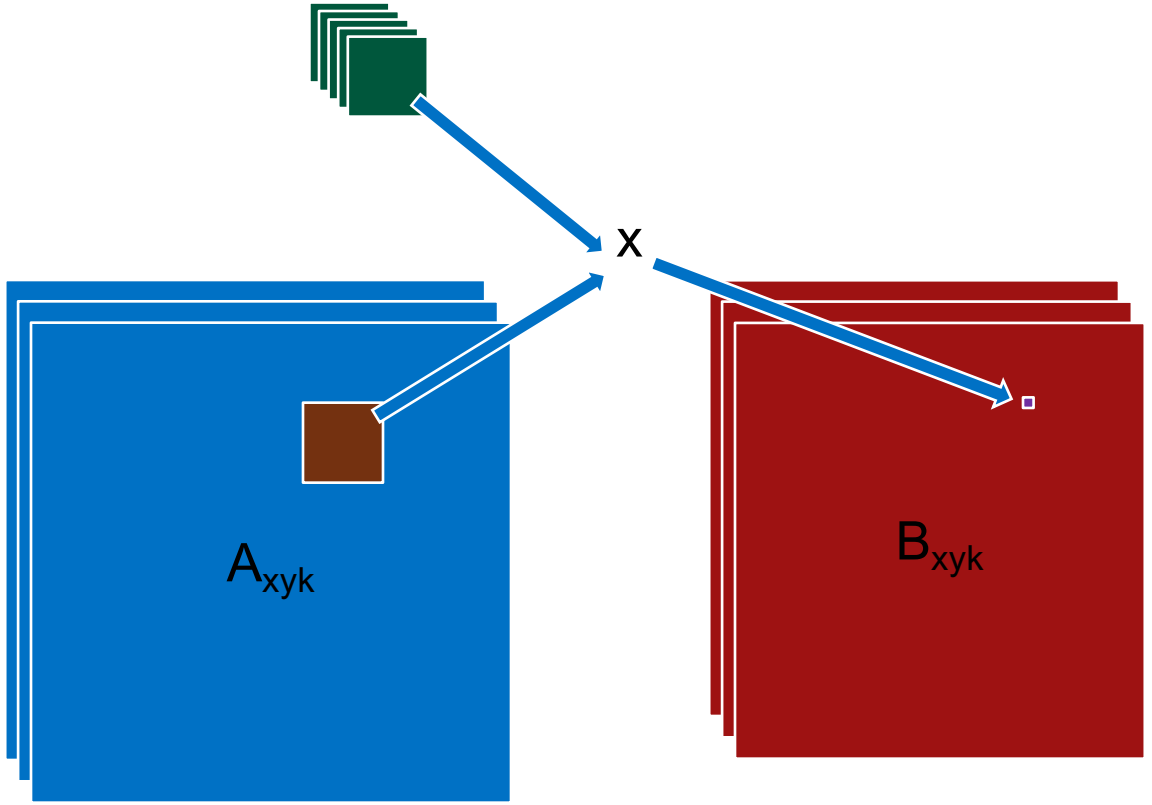
$$b_i = W_{ij} \times a_j$$

CNNS - For image inputs, convolutional stages act as trained feature detectors



CNNS require convolution in addition to M X V





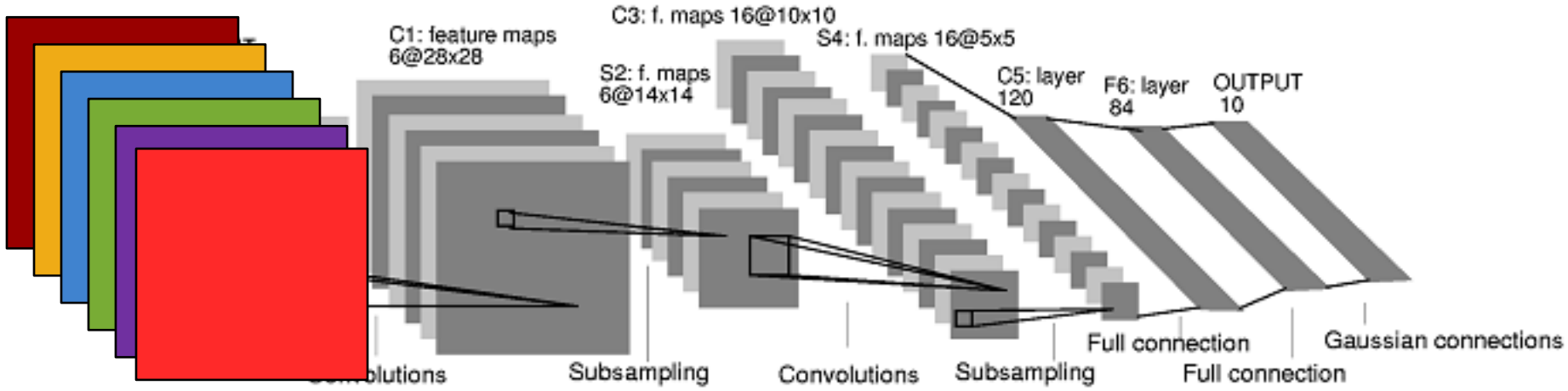
Input maps
A_{xyc}

Output maps
B_{xyk}

4 Distinct Sub-problems

DNNs are Trivially Parallelized

Lots of parallelism in a DNN

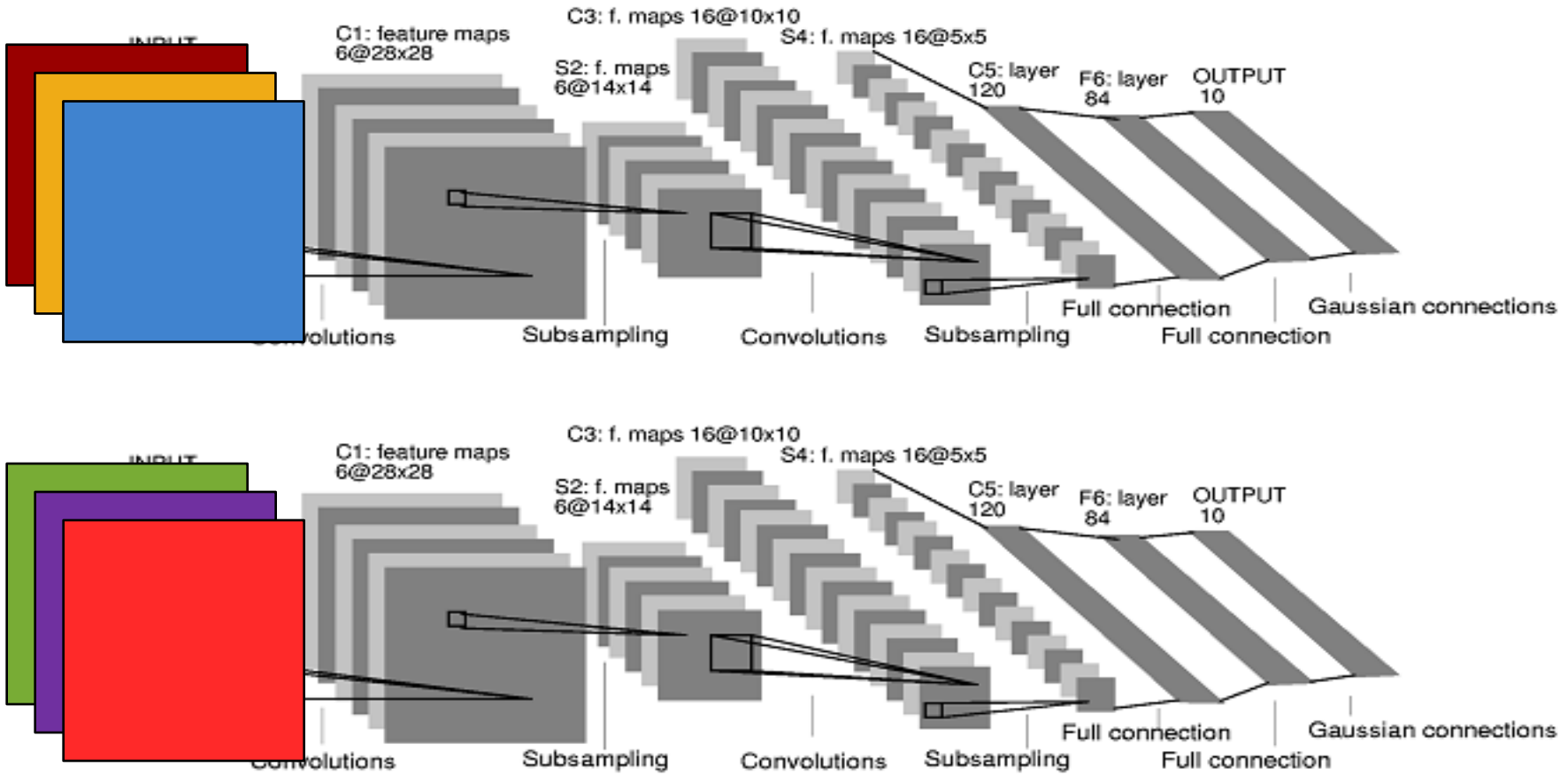


- Inputs
- Points of a feature map
- Filters
- Elements within a filter

- Multiplies within layer are independent
- Sums are reductions
- Only layers are dependent
- No data dependent operations

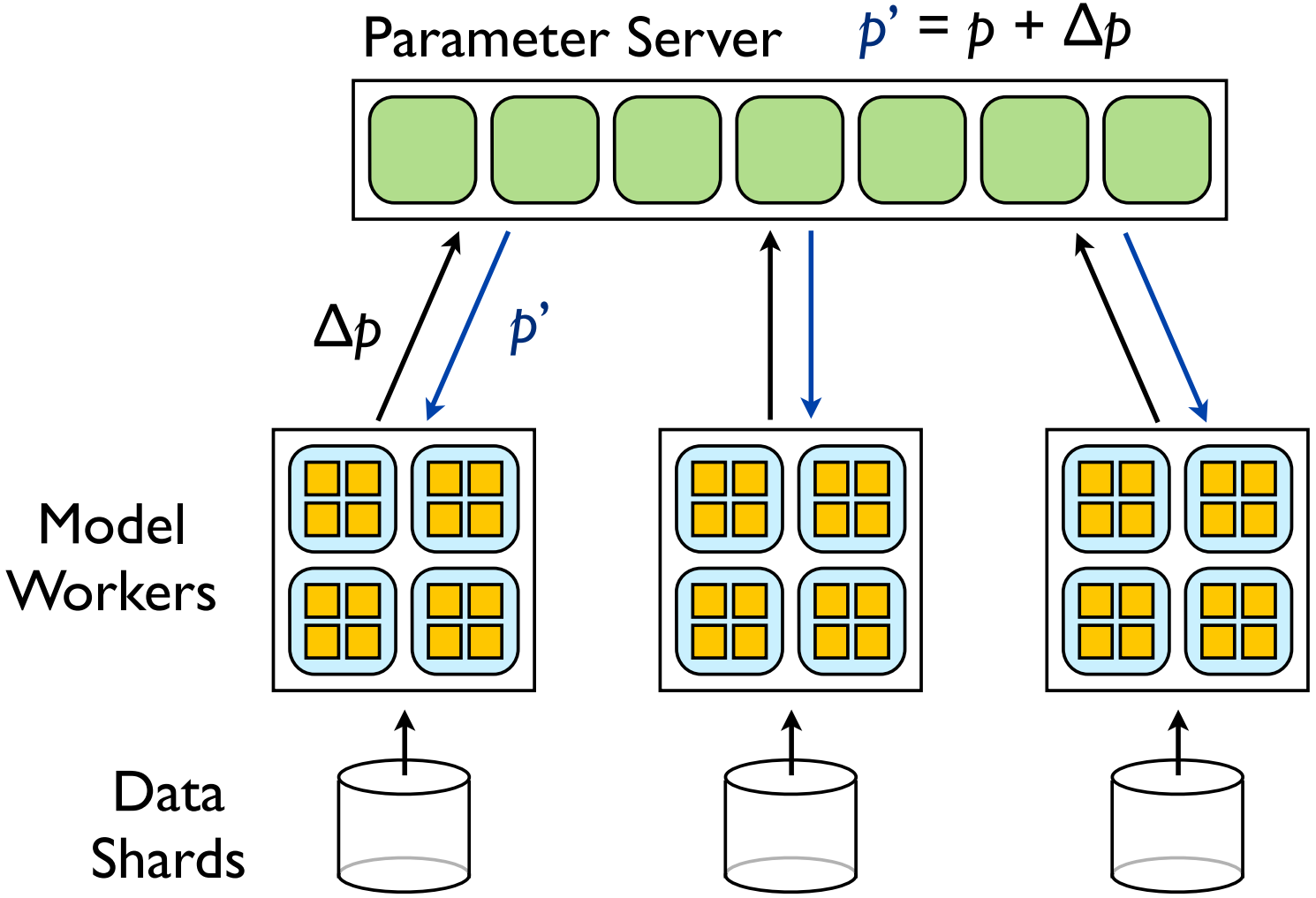
  => can be statically scheduled

Data Parallel – Run multiple inputs in parallel



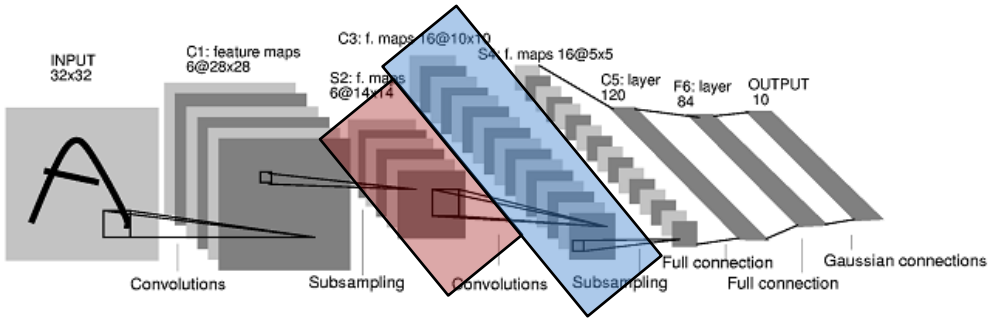
- Doesn’t affect latency for one input
- Requires P-fold larger batch size
- For training requires coordinated weight update

Parameter Update



Large Scale Distributed Deep Networks, Jeff Dean et al., 2013

Model-Parallel Convolution – by output region (x,y)



6D Loop

Forall region XY

For each output map j

For each input map k

For each pixel x,y in XY

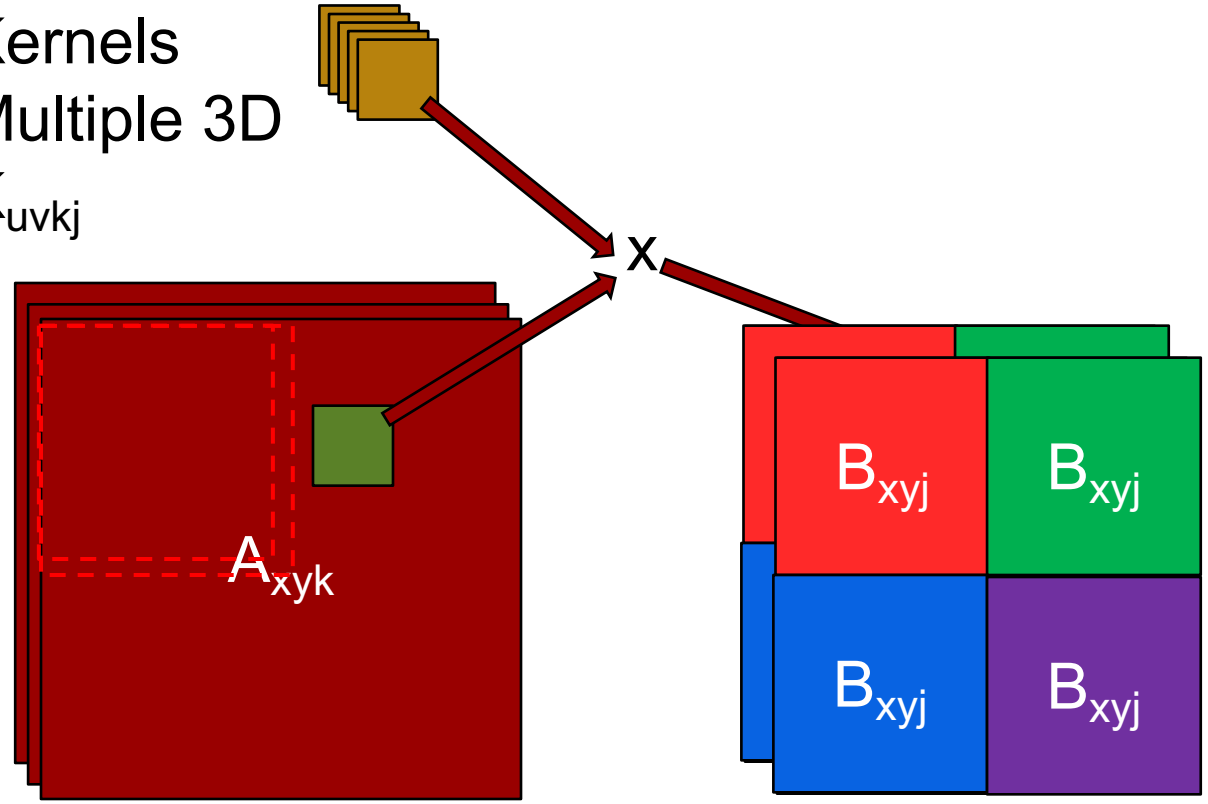
For each kernel element u,v

B_{xyj} += $A_{(x-u)(y-v)k}$ x K_{uvkj}

Kernels

Multiple 3D

K_{uvkj}



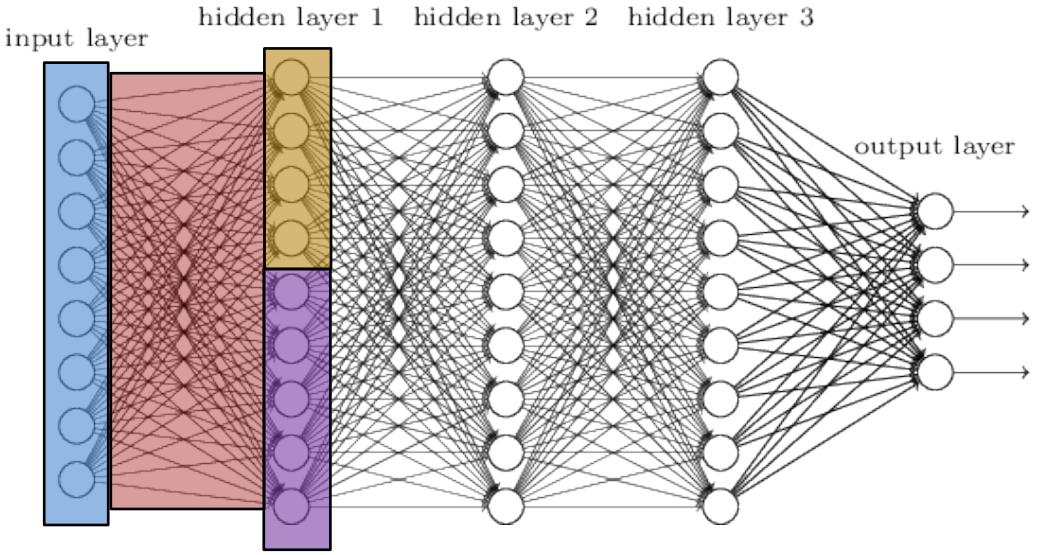
Input maps

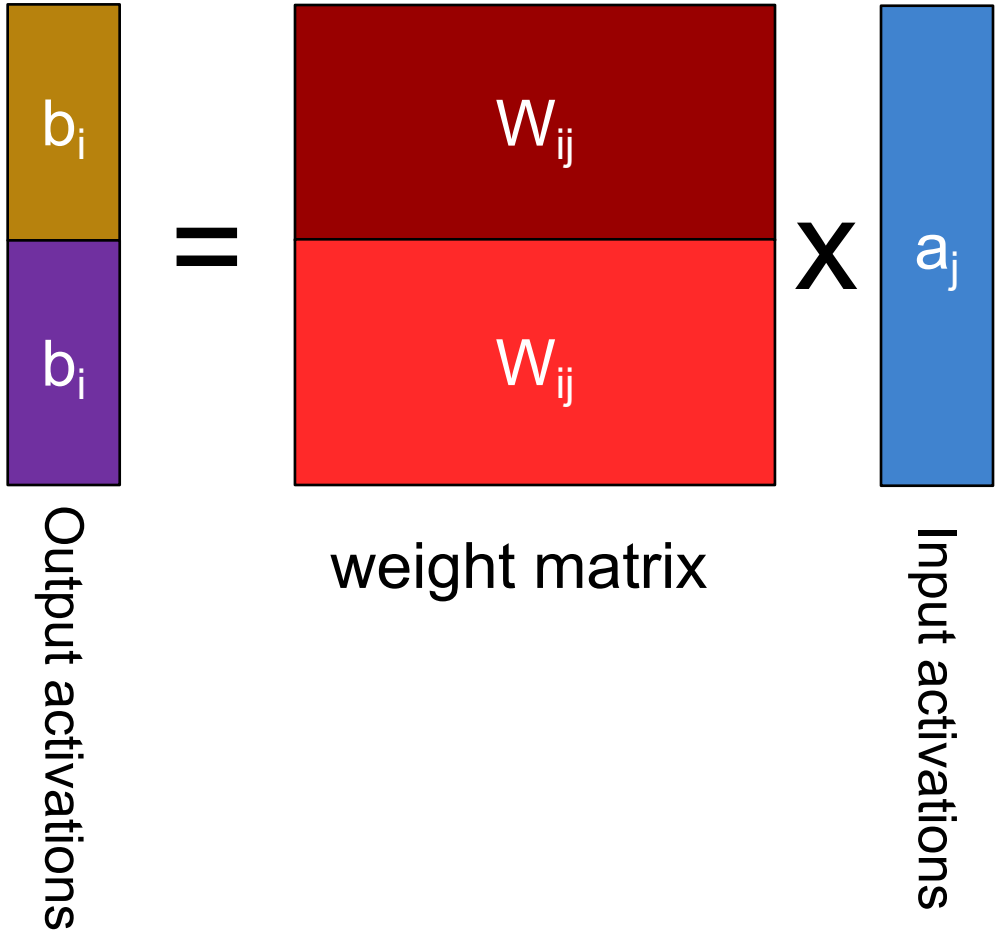
A_{xyk}

Output maps

B_{xyj}

Model Parallel Fully-Connected Layer (M x V)





GPUs

Pascal GP100

- 10 TeraFLOPS FP32
- 20 TeraFLOPS FP16
- 16GB HBM – 750GB/s
- 300W TDP
- 67GFLOPS/W (FP16)
- 16nm process
- 160GB/s NV Link

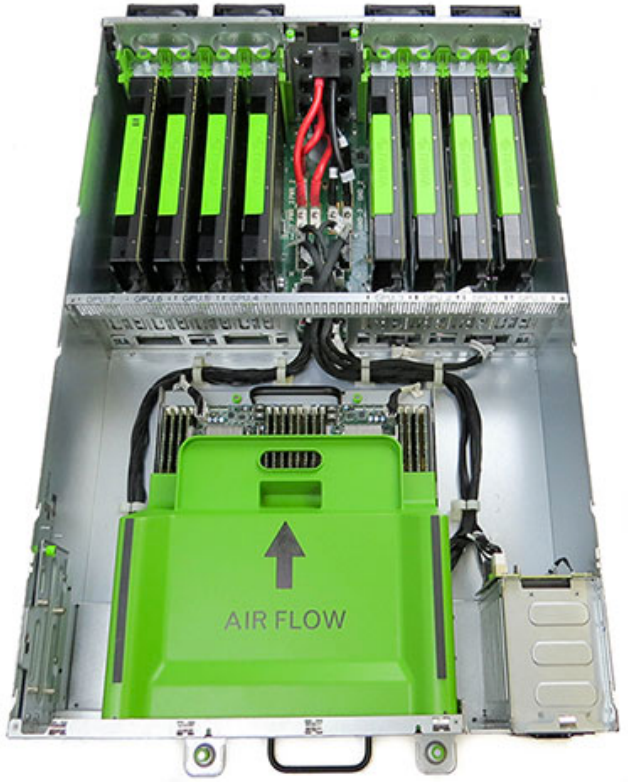
NVIDIA DGX-1

WORLD'S FIRST DEEP LEARNING SUPERCOMPUTER

- 170 TFLOPS
- 8x Tesla P100 16GB
- NVLink Hybrid Cube Mesh
- Optimized Deep Learning Software
- Dual Xeon
- 7 TB SSD Deep Learning Cache
- Dual 10GbE, Quad IB 100Gb
- 3RU – 3200W

Facebook's deep learning machine

- Purpose-Built for Deep Learning Training



2x Faster Training for Faster Deployment

2x Larger Networks for Higher Accuracy

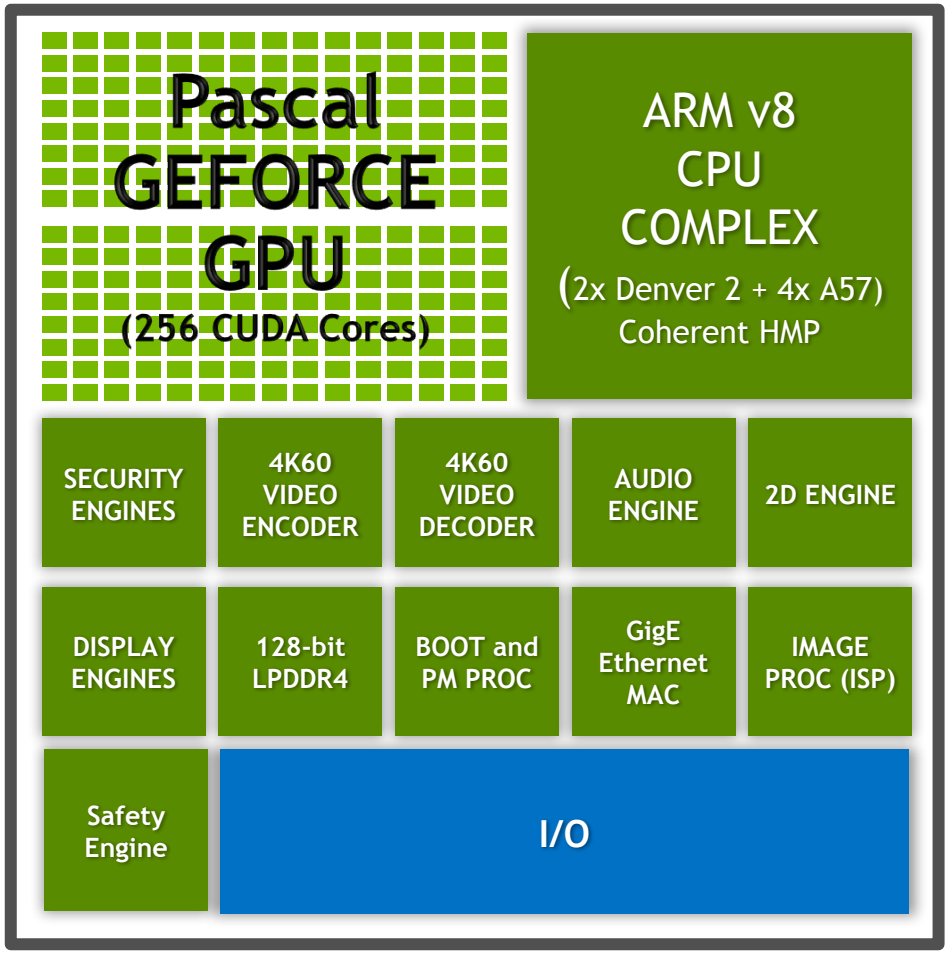
Powered by Eight Tesla M40 GPUs

Open Rack Compliant

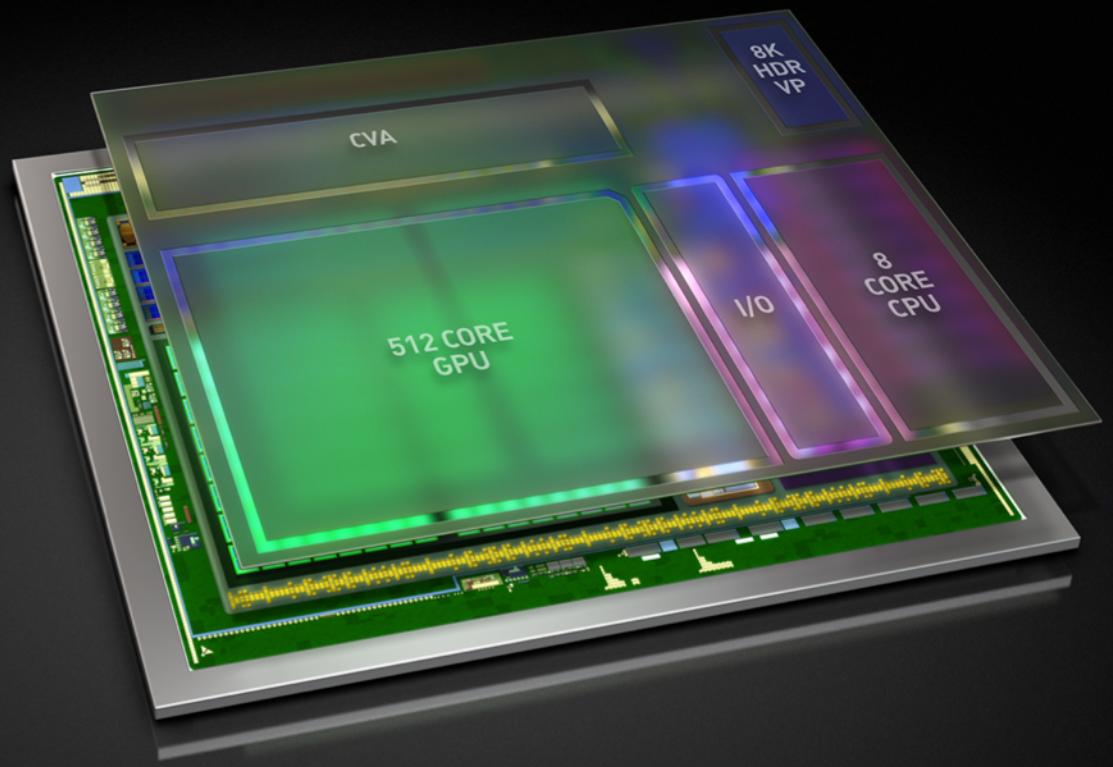
"Most of the major advances in machine learning and AI in the past few years have been contingent on tapping into powerful GPUs and huge data sets to build and train advanced models"

Serkan Piantino
Engineering Director of Facebook AI Research

NVIDIA Parker



- 1.5 Teraflop FP16
- 4GB of LPDDR4 @ 25.6 GB/s
- 15 W TDP (1W idle, <10W typical)
- 100GFLOPS/W (FP16)
- 16nm process



XAVIER

AI SUPERCOMPUTER SOC

7 Billion Transistors 16nm FF

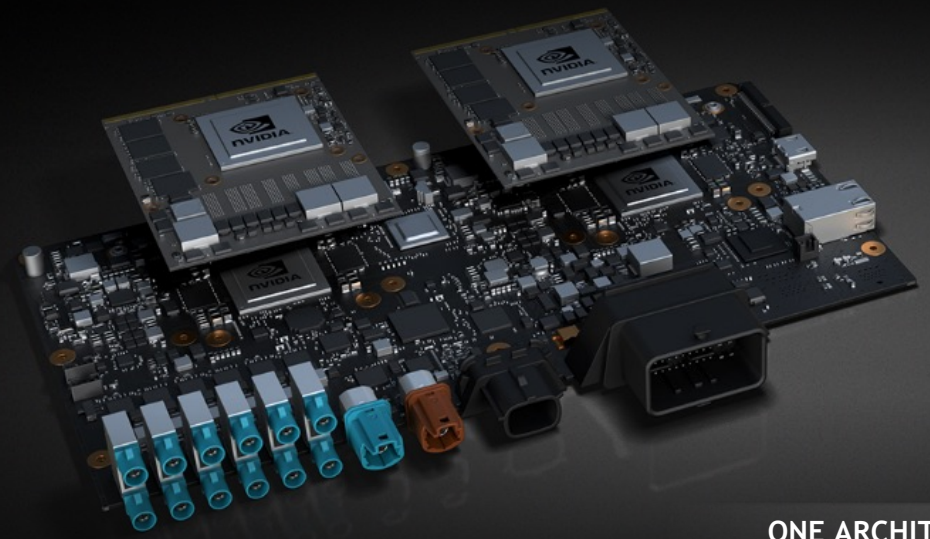
8 Core Custom ARM64 CPU

512 Core Volta GPU

New Computer Vision Accelerator

Dual 8K HDR Video Processors

Designed for ASIL C Functional Safety



XAVIER

AI SUPERCOMPUTER SOC

ONE ARCHITECTURE

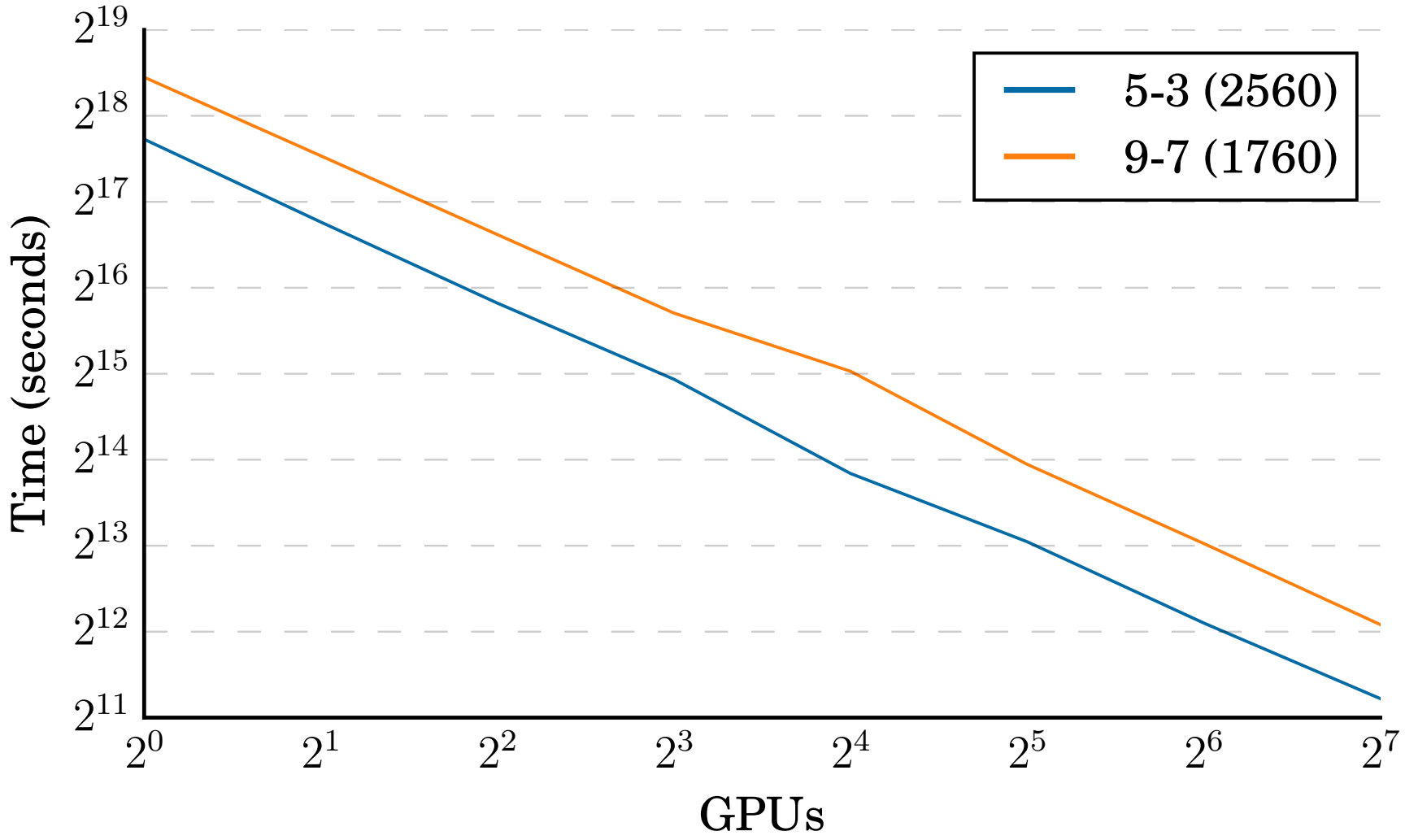
DRIVE PX 2

2 PARKER + 2 PASCAL GPU | 20 TOPS DL | 120 SPECINT | 80W

XAVIER

20 TOPS DL | 160 SPECINT | 20W

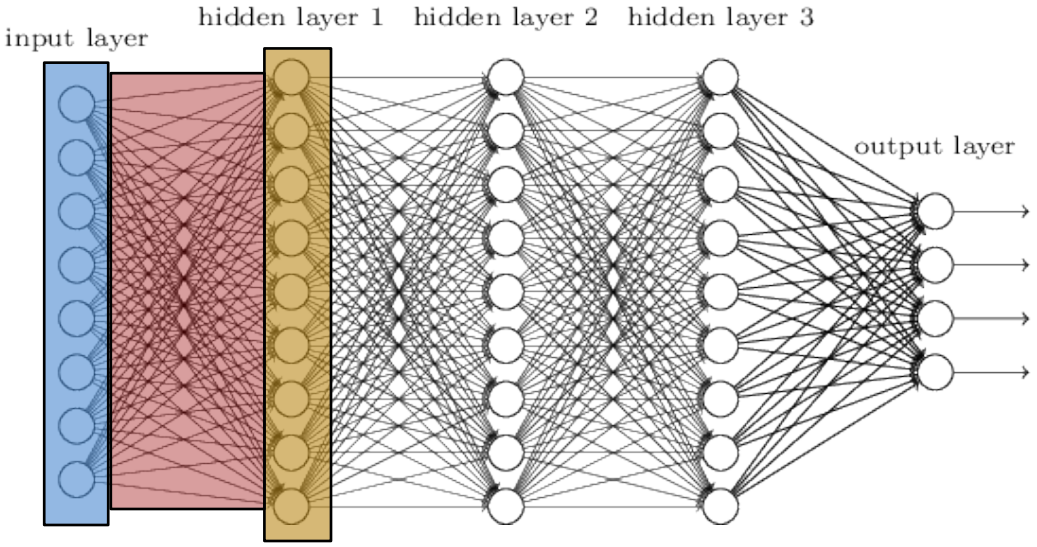
Parallel GPUs on Deep Speech 2



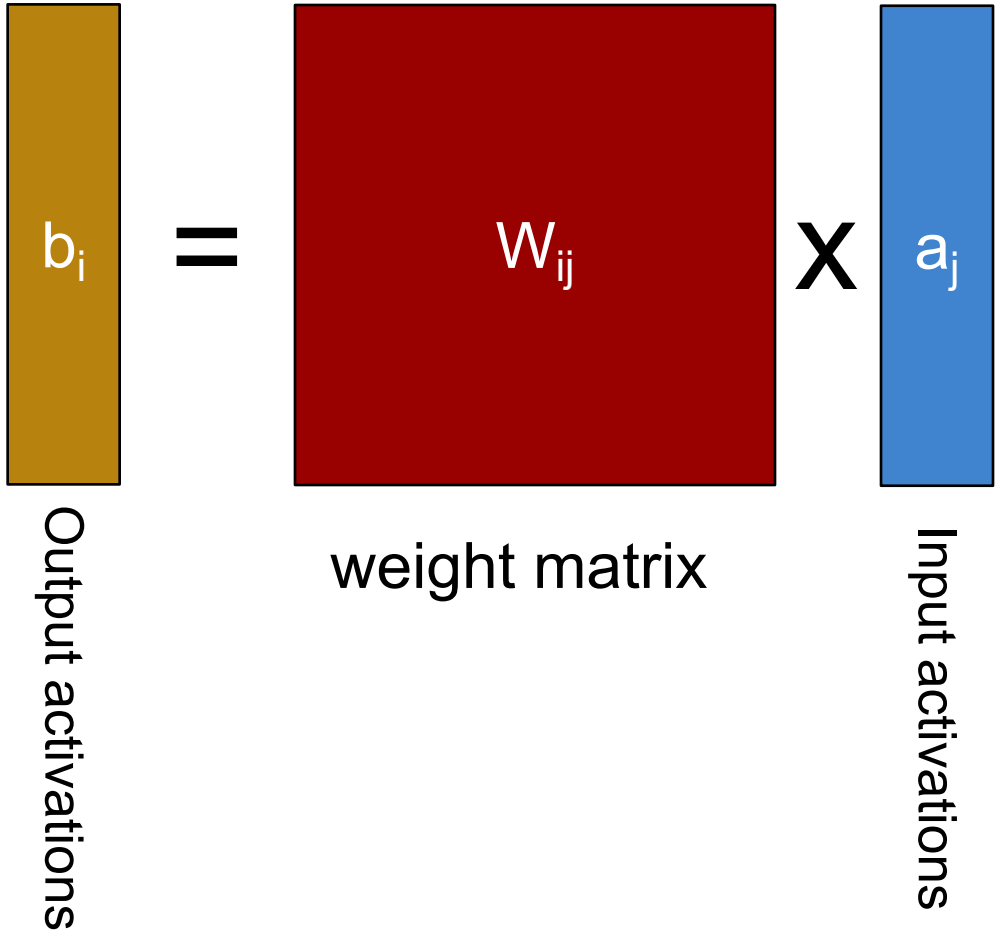
Baidu, Deep Speech 2: End-to-End Speech Recognition in English and Mandarin, 2015

Reduced Precision

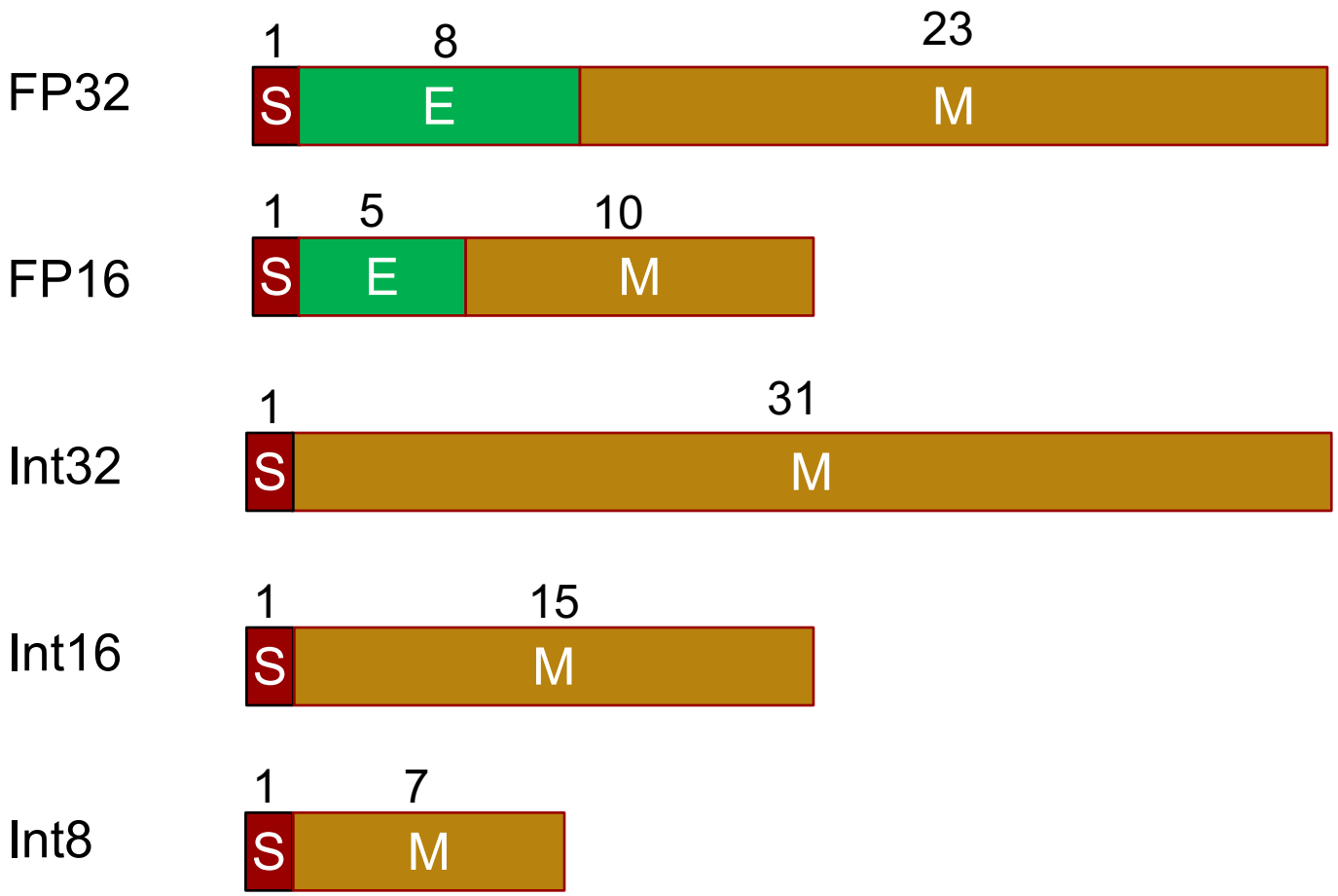
How Much Precision is Needed for Dense M x V?



$$b_i = f\left(\sum_j w_{ij}\, a_i\right)$$

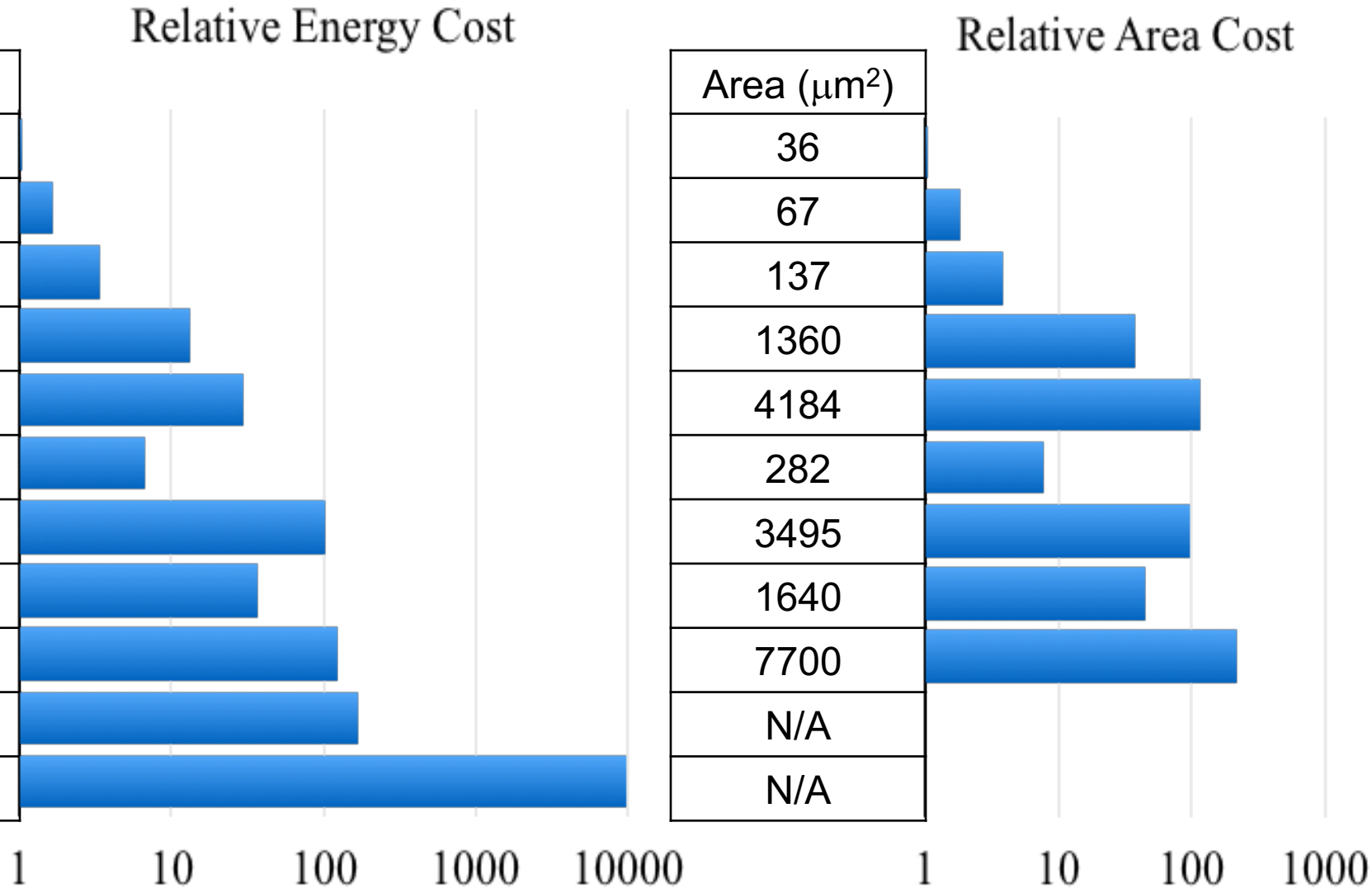


Number Representation



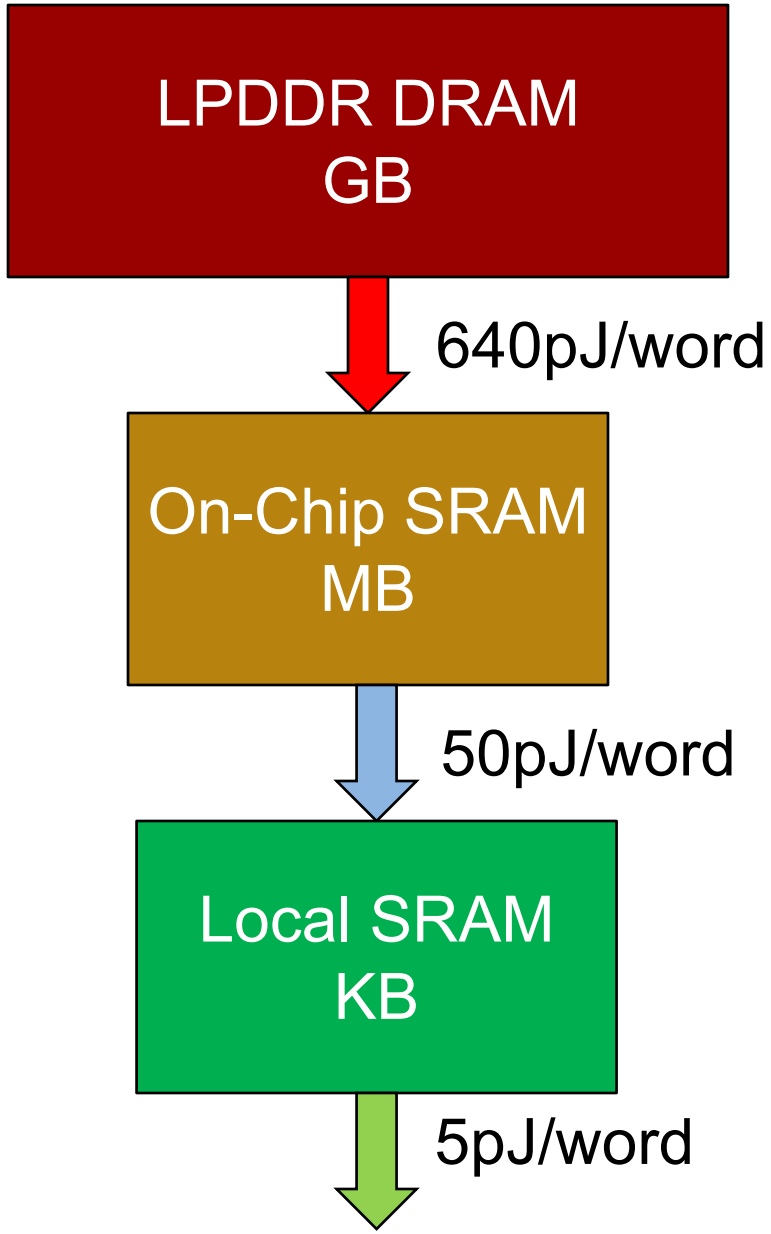
Cost of Operations

Operation:	Energy (pJ)	Area (μm²)
8b Add	0.03	36
16b Add	0.05	67
32b Add	0.1	137
16b FP Add	0.4	1360
32b FP Add	0.9	4184
8b Mult	0.2	282
32b Mult	3.1	3495
16b FP Mult	1.1	1640
32b FP Mult	3.7	7700
32b SRAM Read (8KB)	5	N/A
32b DRAM Read	640	N/A



Energy numbers are from Mark Horowitz "Computing's Energy Problem (and what we can do about it)", ISSCC 2014
Area numbers are from synthesized result using Design Compiler under TSMC 45nm tech node. FP units used DesignWare Library.

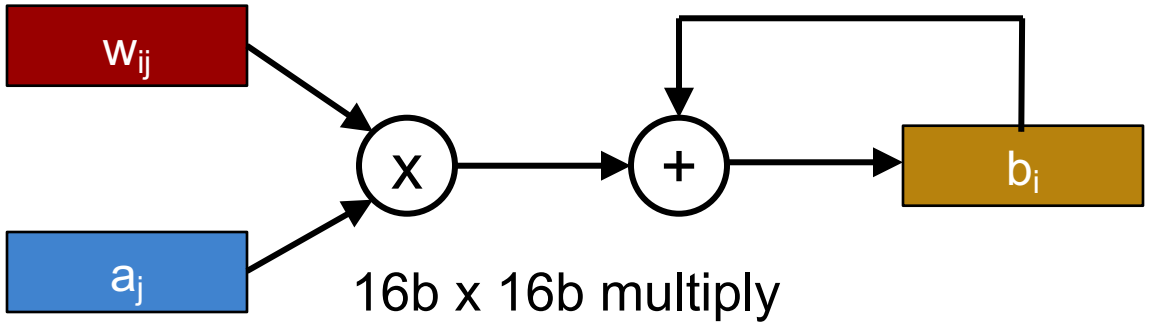
The Importance of Staying Local



Mixed Precision

Store weights as 4b using Trained quantization, decode to 16b

Store activations as 16b



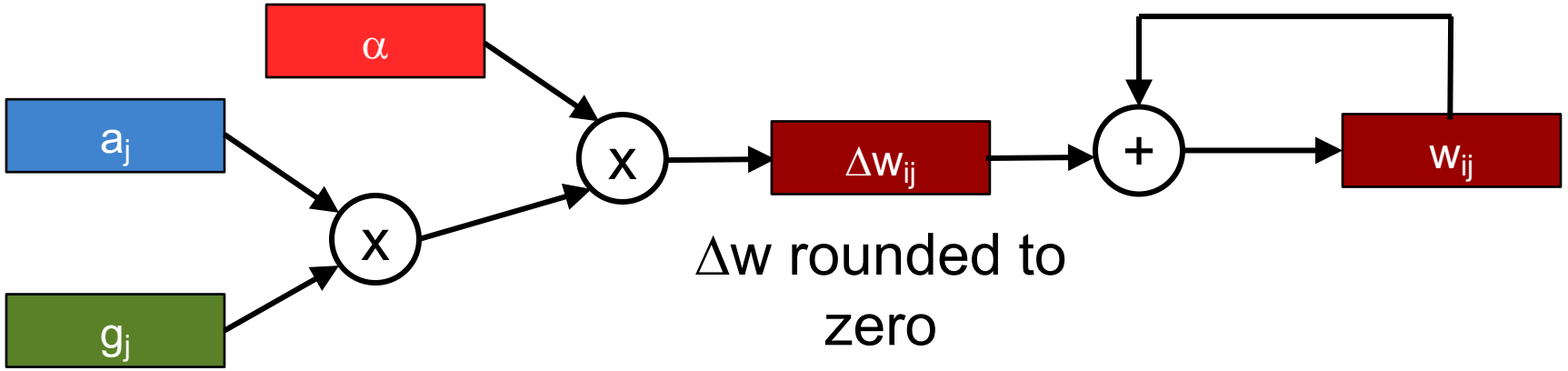
16b x 16b multiply
round result to 16b

accumulate 24b or 32b to avoid saturation

Batch normalization important to ‘center’ dynamic range

Weight Update

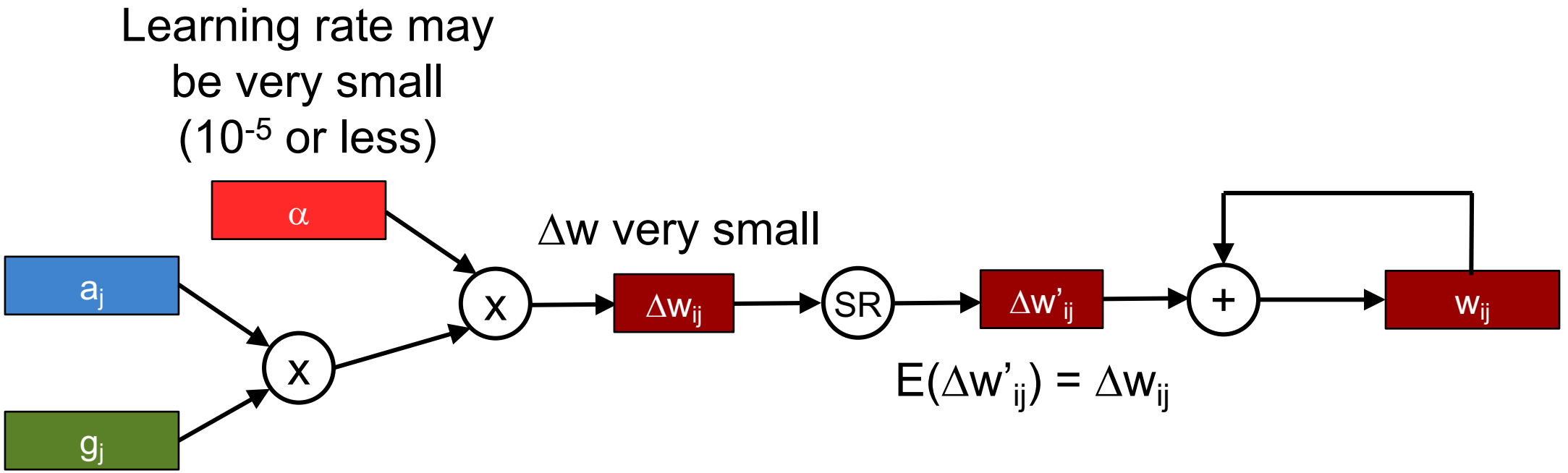
Learning rate may be very small (10^{-5} or less)



Δw rounded to zero

No learning!

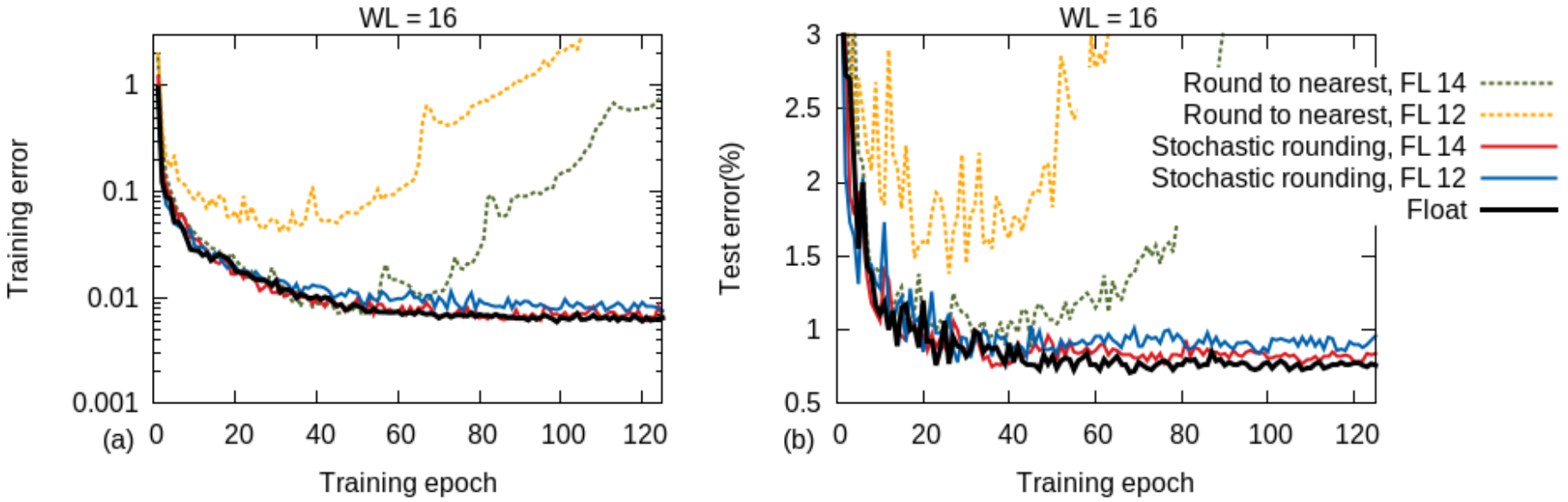
Stochastic Rounding



Reduced Precision For Training

$$b_i = f\left(\sum_j w_{ij}\, a_i\right)$$

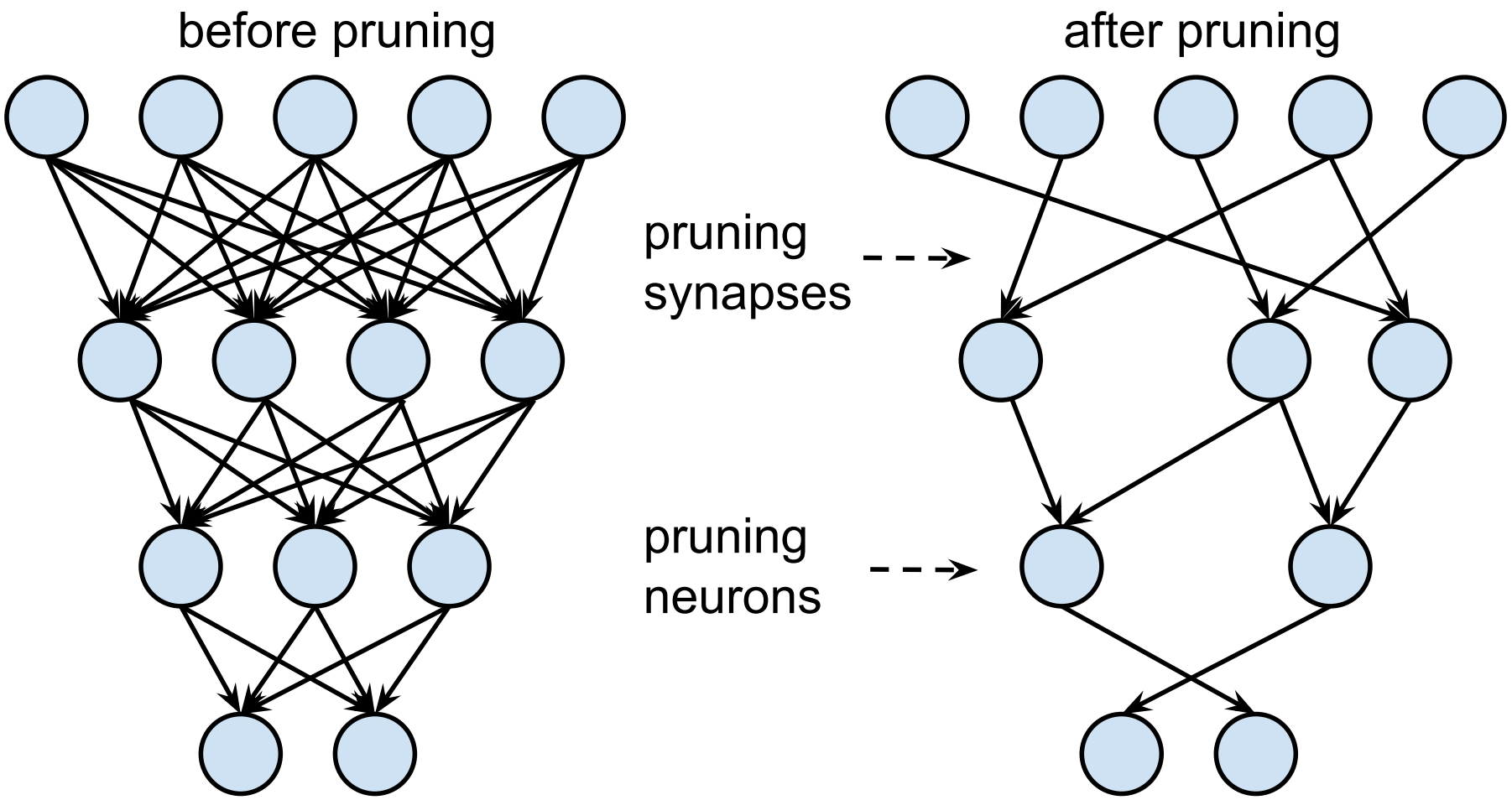
$$w_{ij} = w_{ij} + \alpha a_i g_j$$



S. Gupta et.al “Deep Learning with Limited Numerical

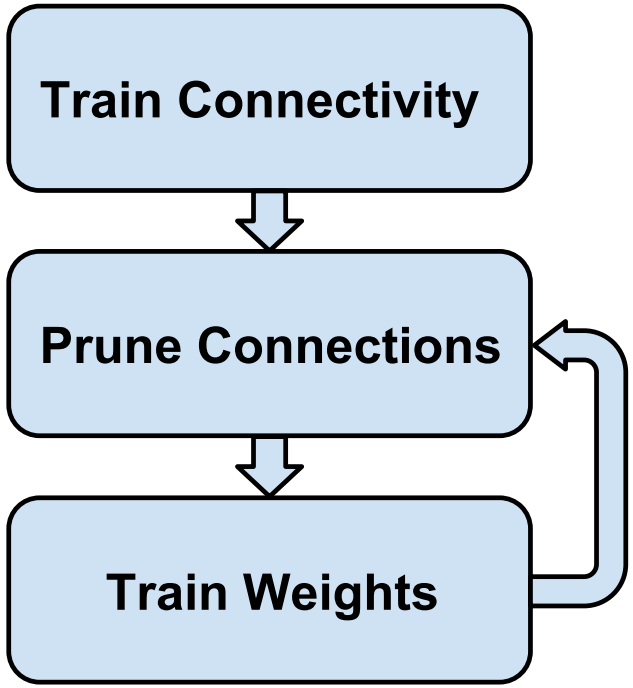
Pruning

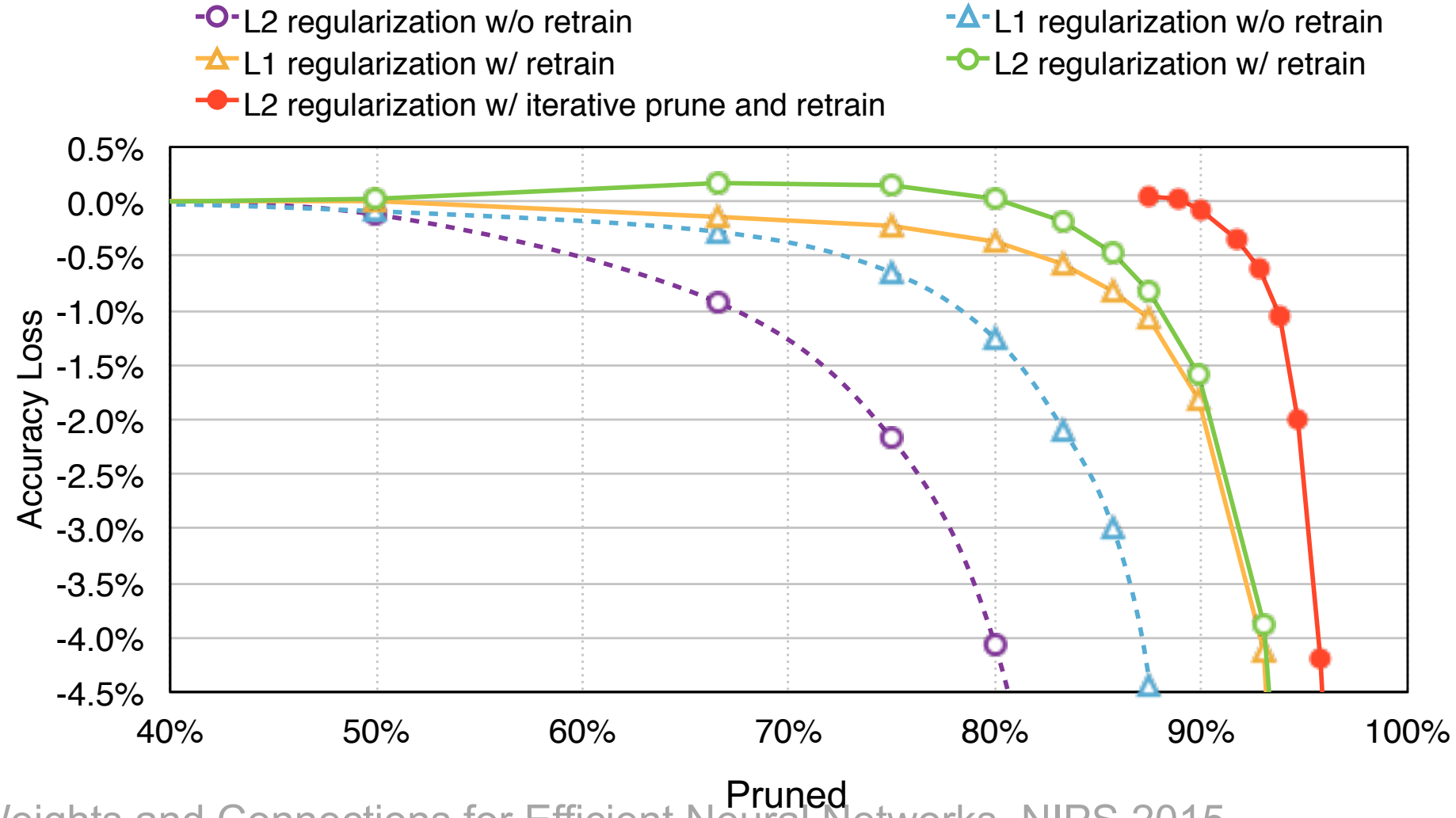
Pruning



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

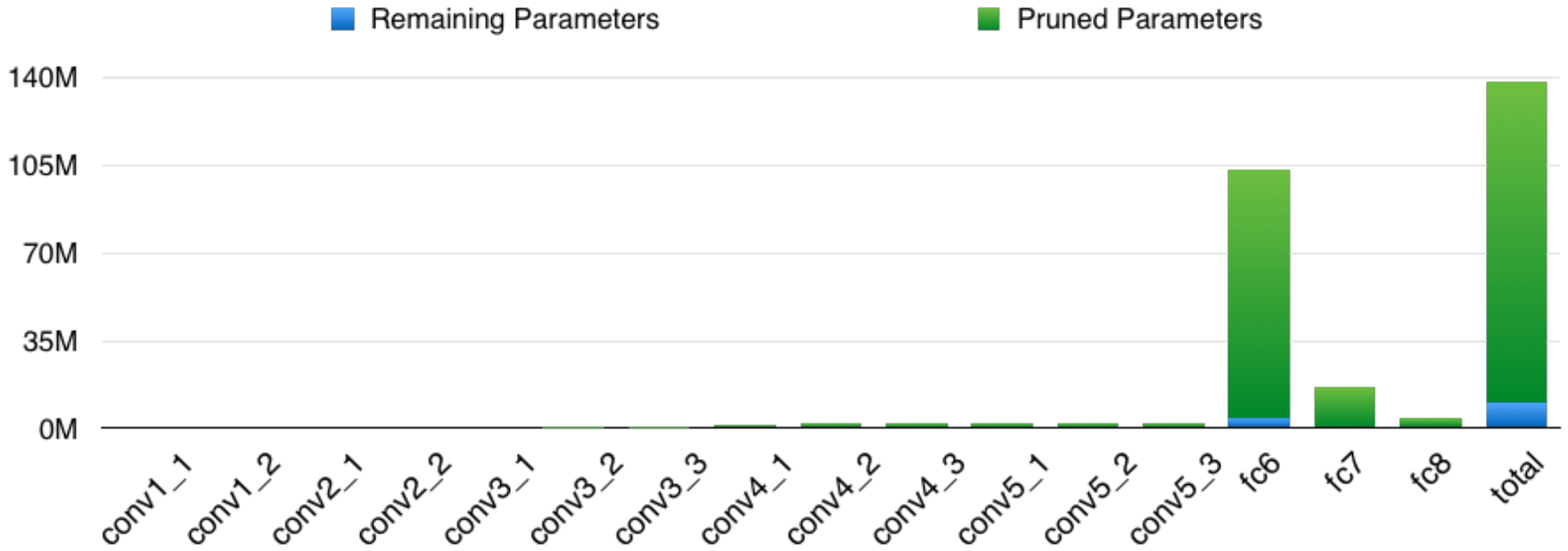
Retrain to Recover Accuracy

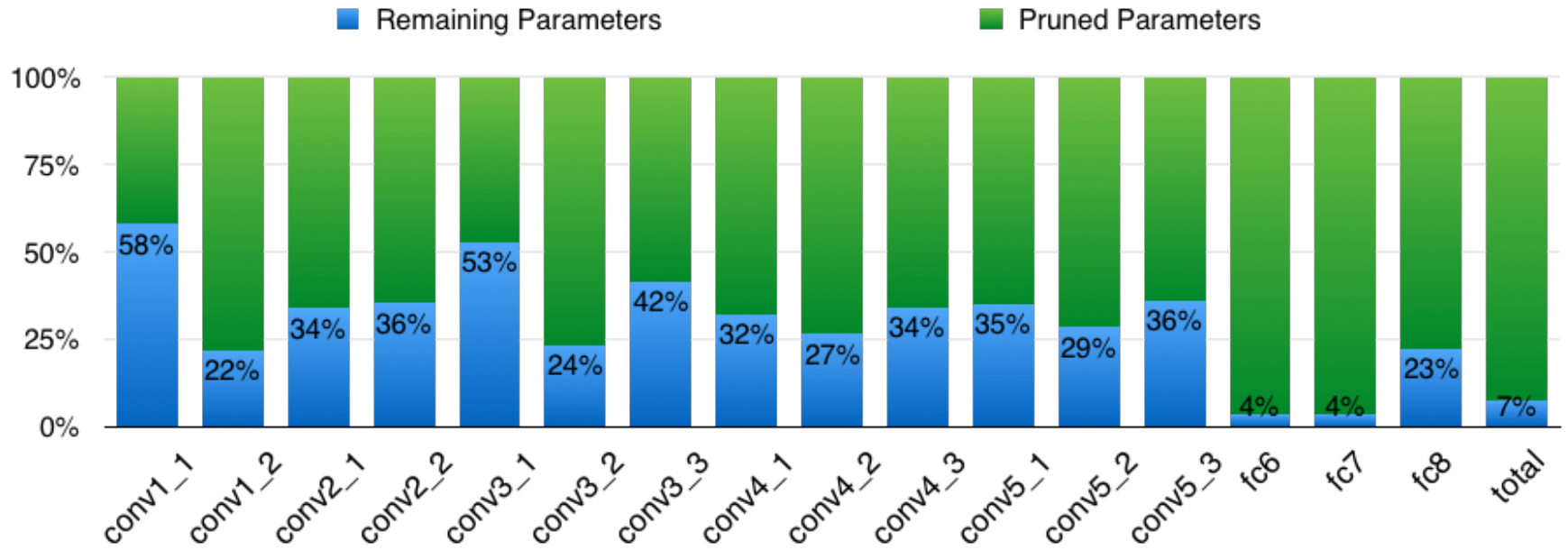




Han et al. Learning both Weights and Connections for Efficient Neural Networks, NIPS 2015

Pruning of VGG-16





Pruning Neural Talk 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

Speedup of Pruning on CPU/GPU

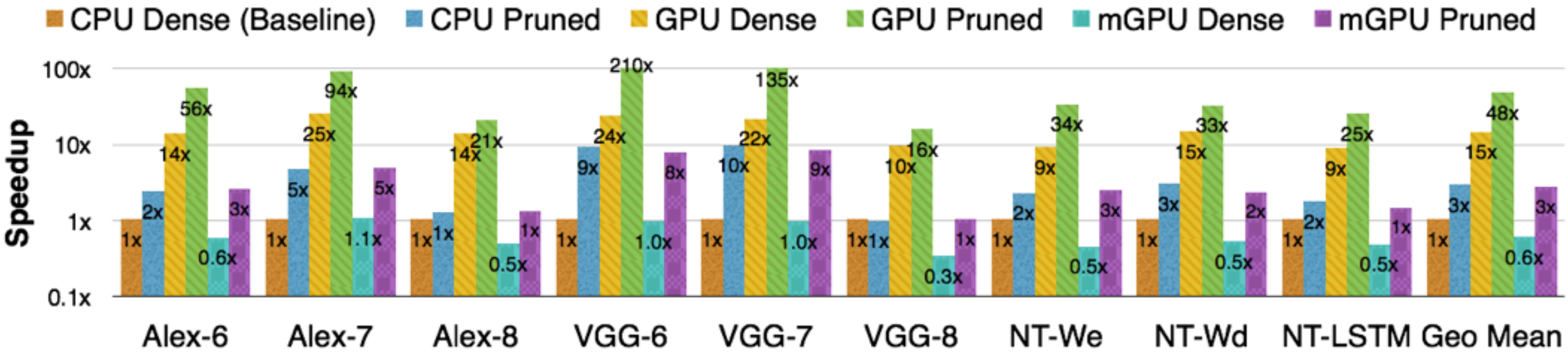
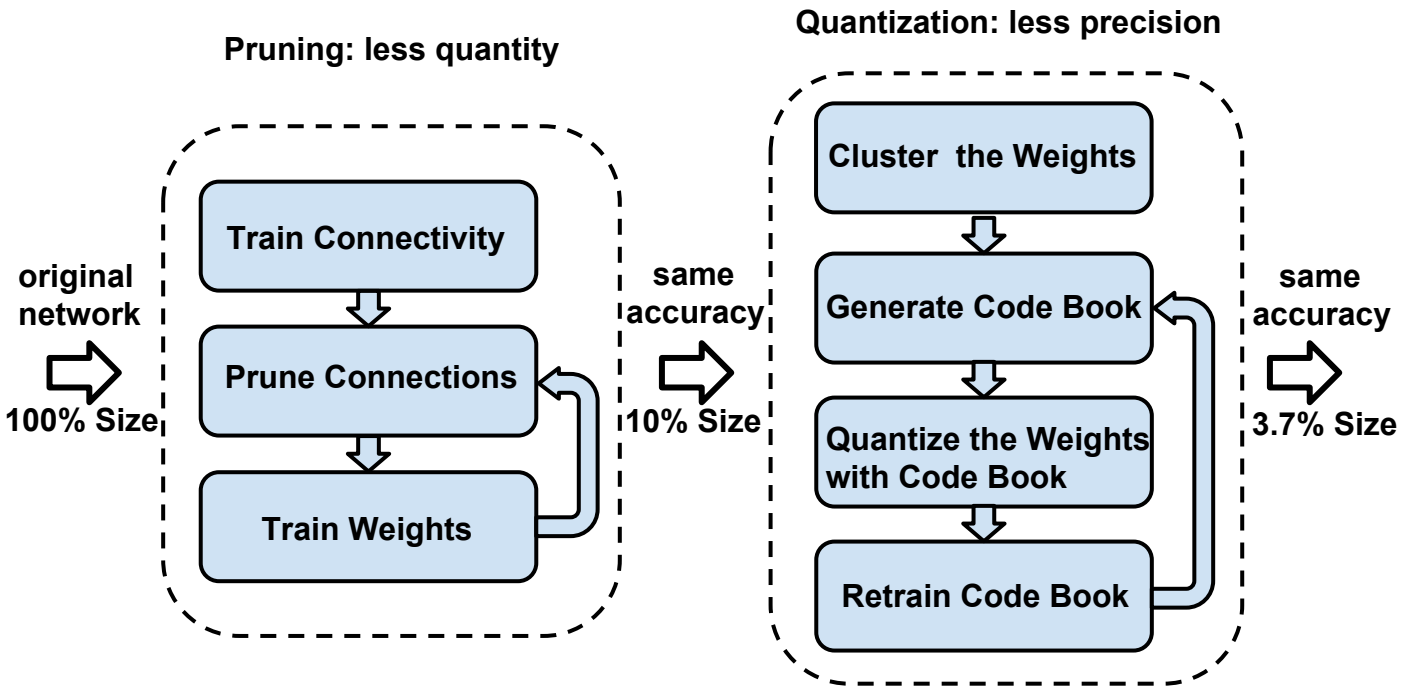


Figure 9: Compared with the original network, pruned network layer achieved 3× speedup on CPU, 3.5× on GPU and 4.2× on mobile GPU on average. Batch size = 1 targeting real time processing. Performance number normalized to CPU.

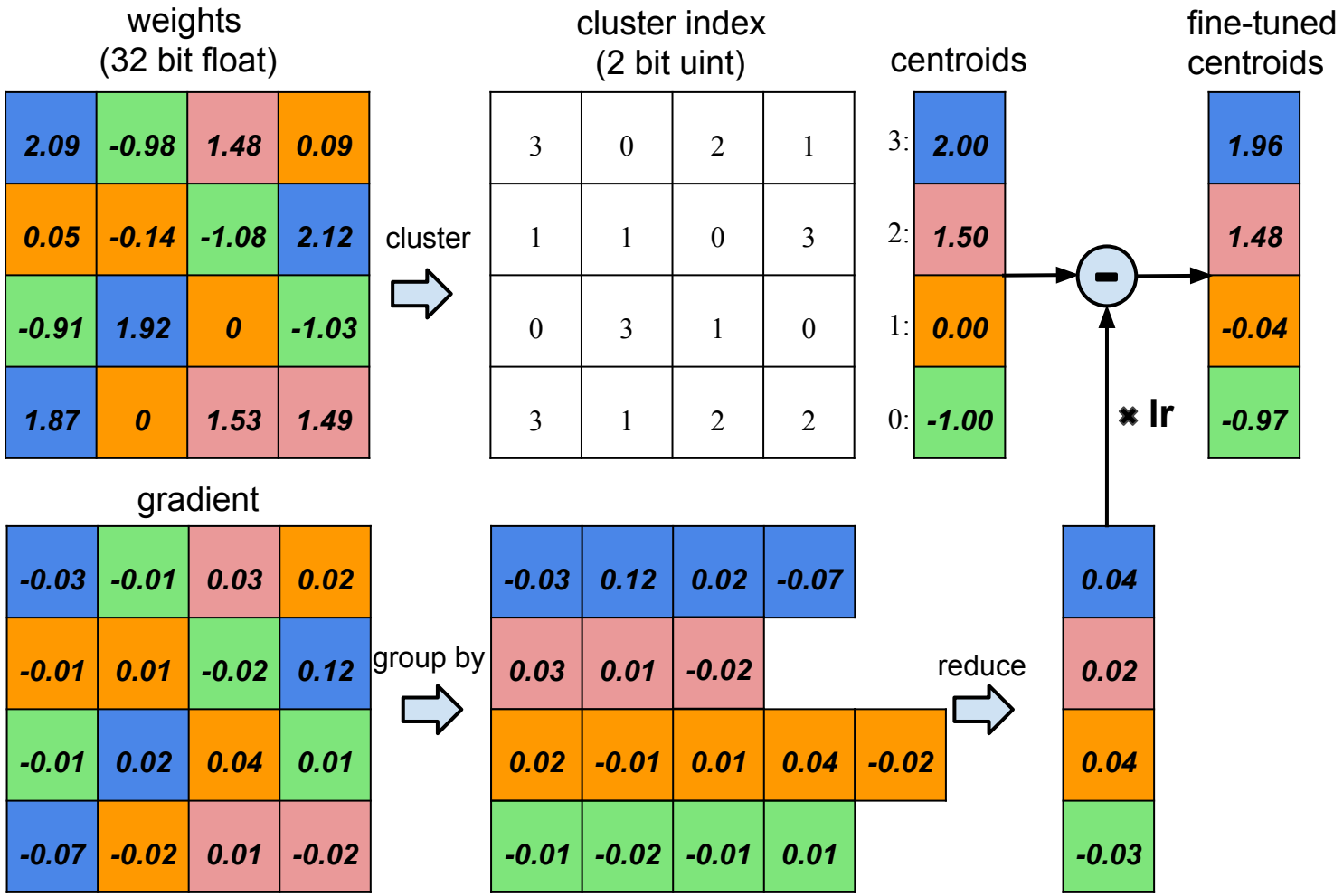
Intel Core i7 5930K: MKL CBLAS GEMV, MKL SPBLAS CSRMV
NVIDIA GeForce GTX Titan X: cuBLAS GEMV, cuSPARSE CSRMV
NVIDIA Tegra K1: cuBLAS GEMV, cuSPARSE CSRMV

Trained Quantization
(Weight Sharing)



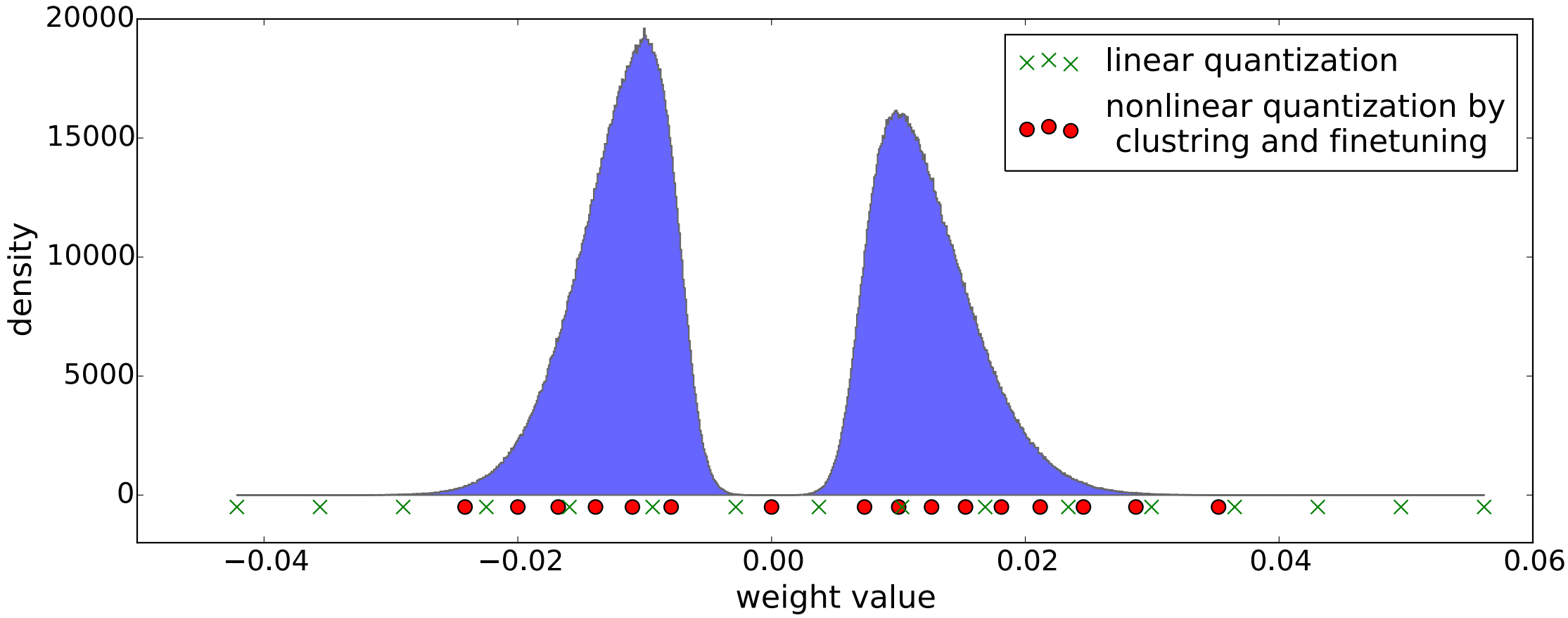
Han et al. Deep Compression: Compressing Deep Neural Networks with Pruning, Trained Quantization and Huffman Coding, arXiv 2015

Weight Sharing via K-Means



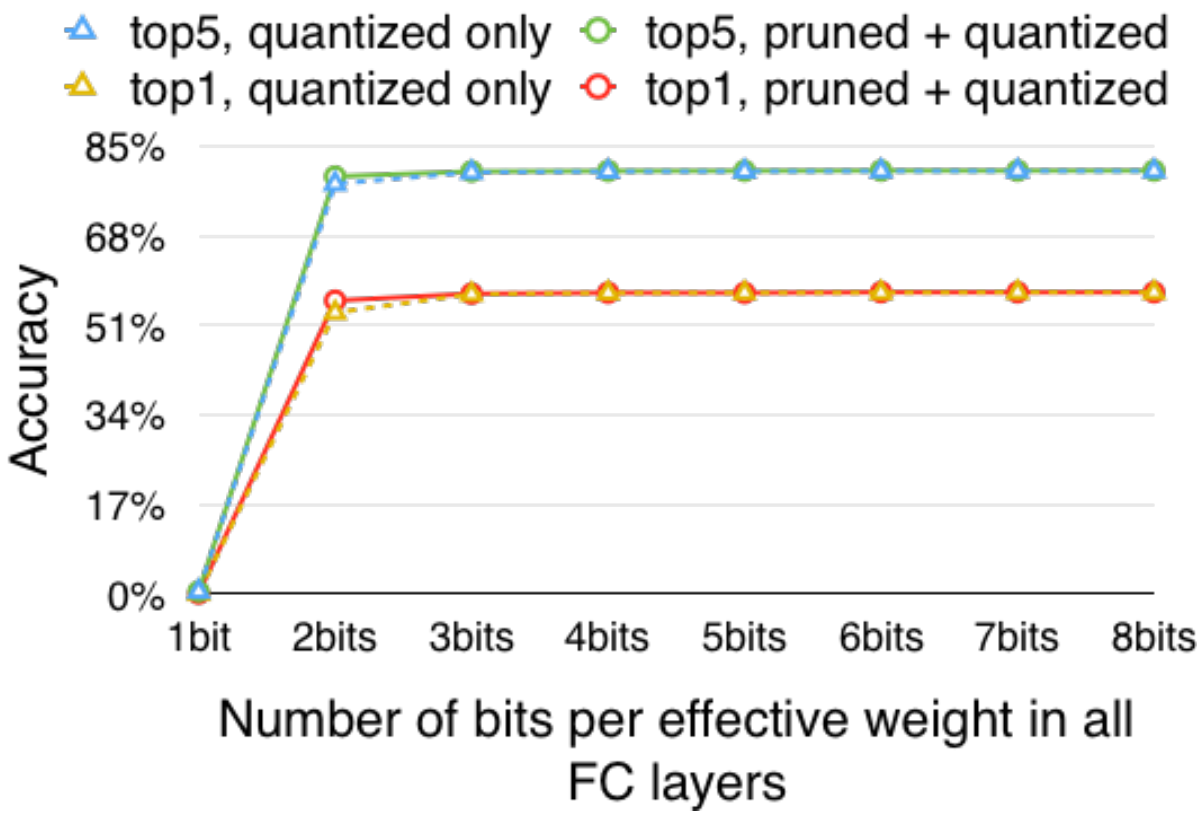
Han et al. Deep Compression: Compressing Deep Neural Networks with Pruning, Trained Quantization and Huffman Coding, arXiv 2015

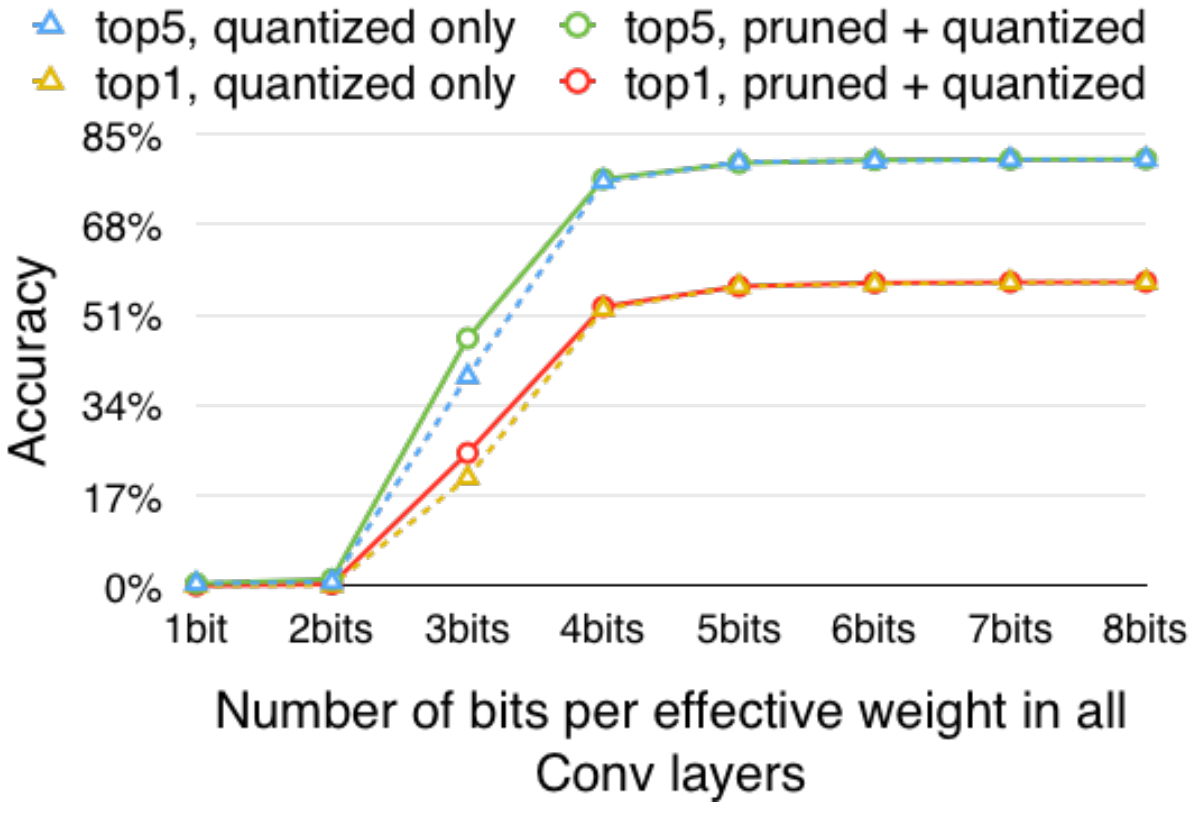
Trained Quantization



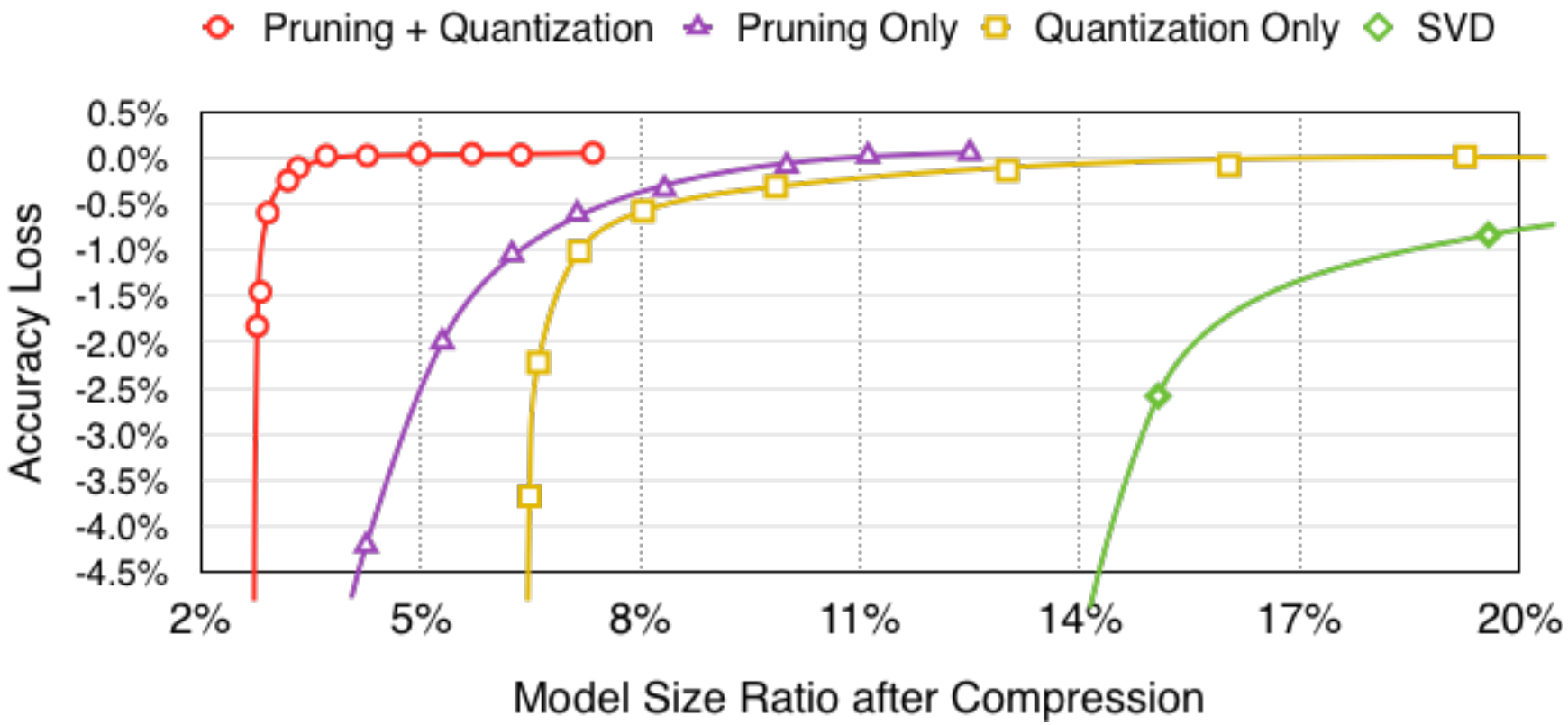
Han et al. Deep Compression: Compressing Deep Neural Networks with Pruning, Trained Quantization and Huffman Coding, arXiv 2015

Bits per Weight





Pruning + Trained Quantization

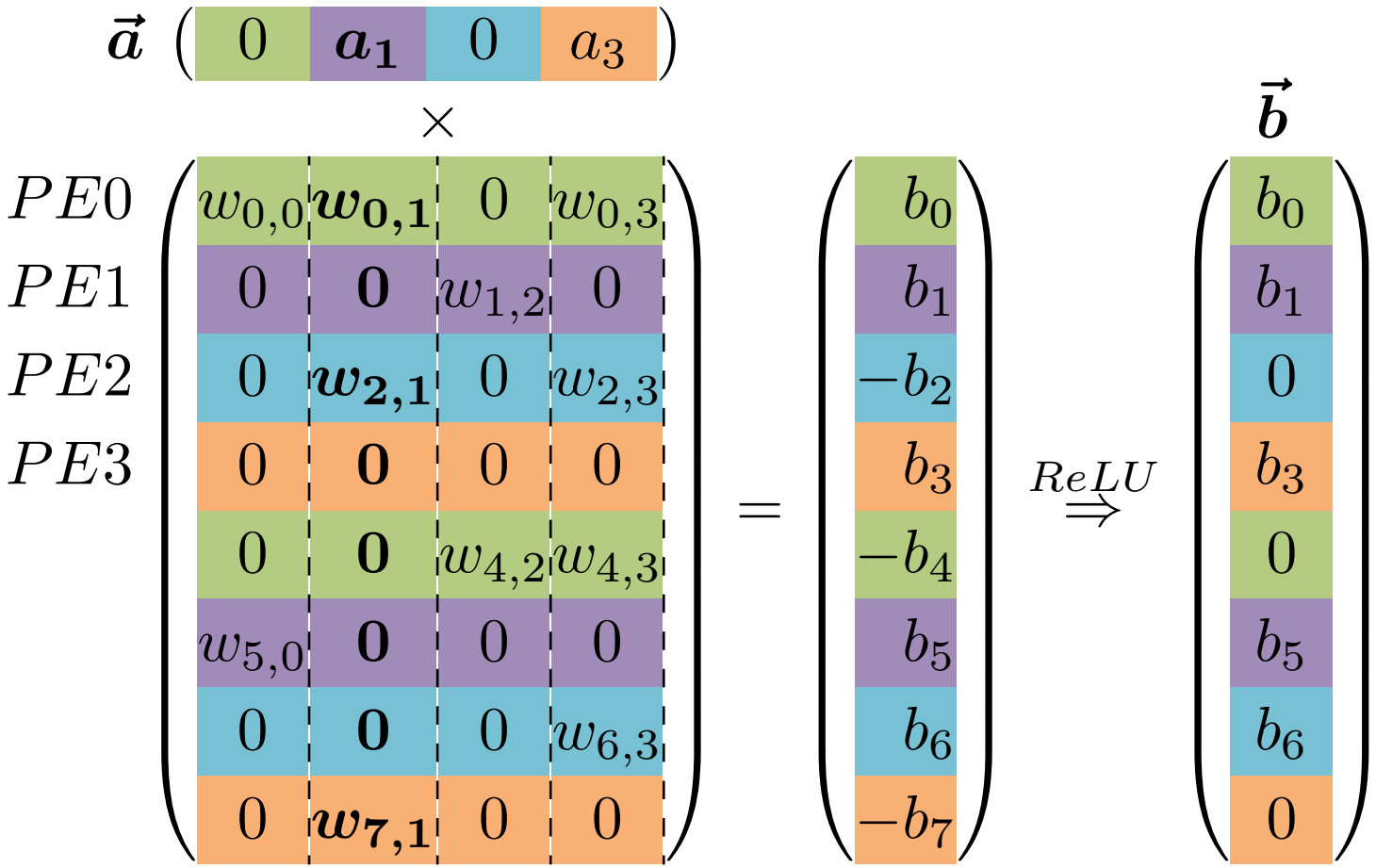


30x – 50x Compression Means

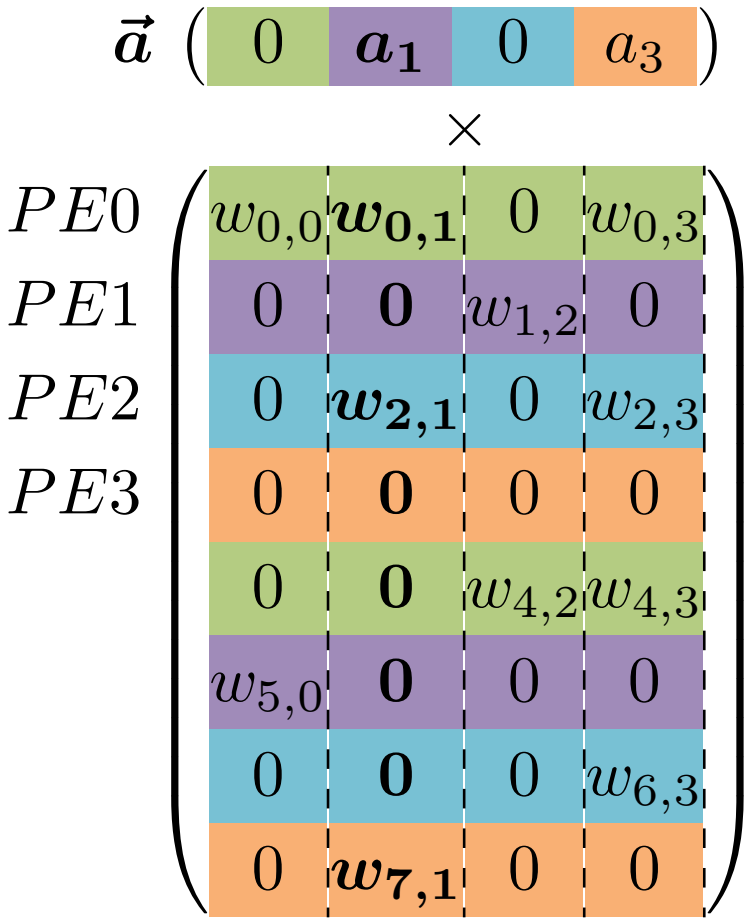
- Complex DNNs can be put in mobile applications (<100MB total)
  - 1GB network (250M weights) becomes 20-30MB
- Memory bandwidth reduced by 30-50x
  - Particuarly for FC layers in real-time applications with no reuse
- Memory working set fits in on-chip SRAM
  - 5pJ/word access vs 640pJ/word

Efficient Inference Engine

Sparse Matrix Representation

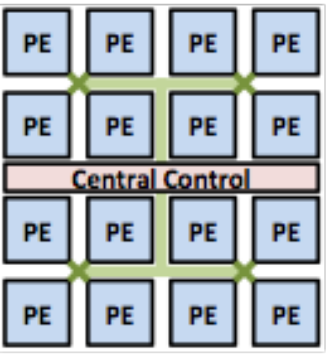


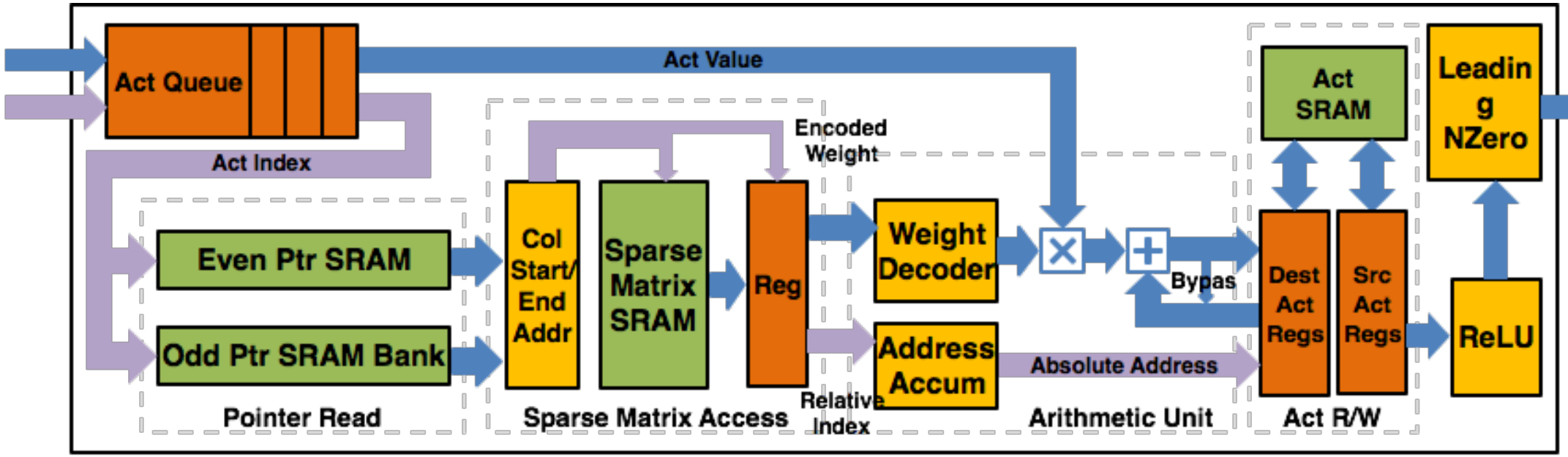
Sparse Matrix Representation



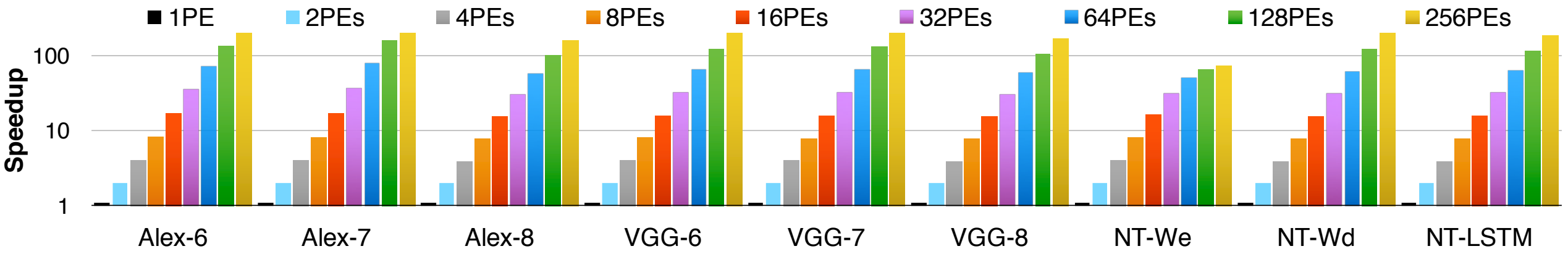
Virtual Weight	$W_{0,0}$	$W_{0,1}$	$W_{4,2}$	$W_{0,3}$	$W_{4,3}$
Relative Index	0	1	2	0	0
Column Pointer	0	1	2	3	

EIE Architecture

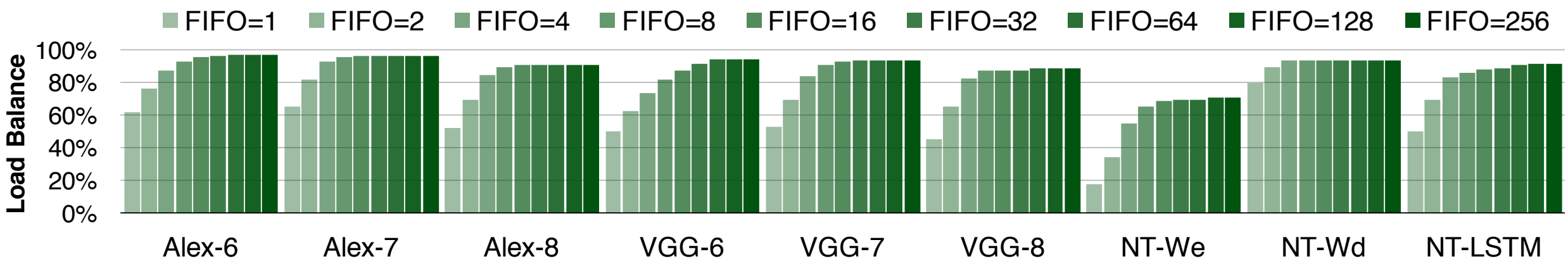




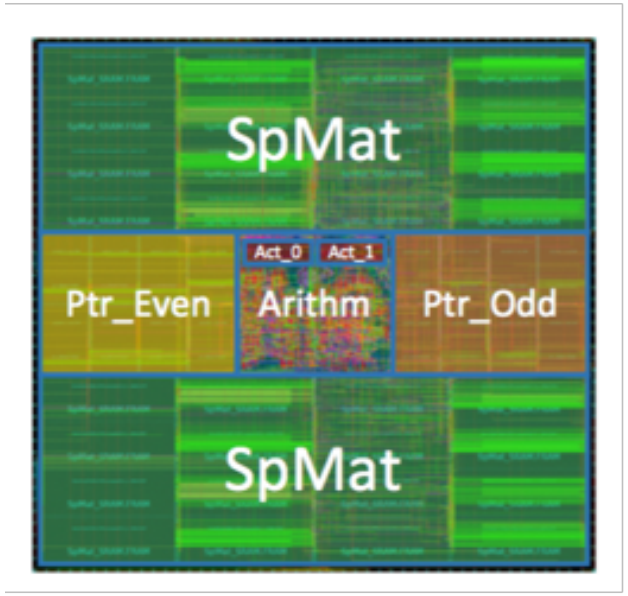
Scalability



Load Balance



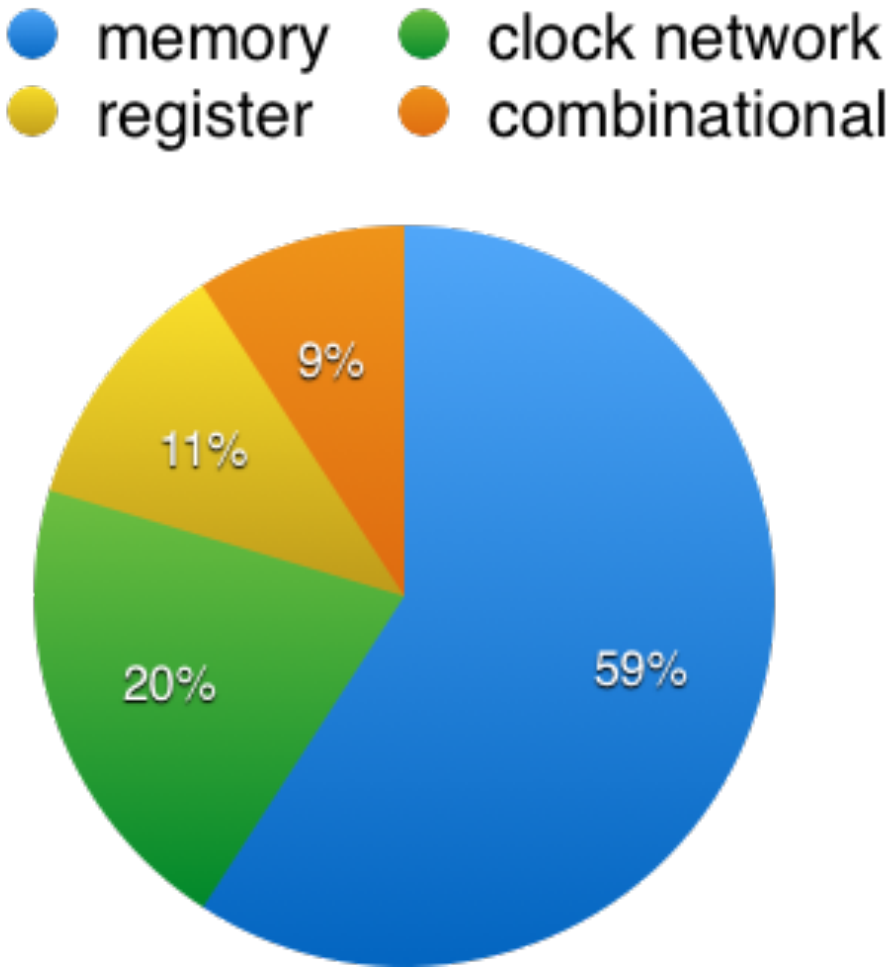
Implementation

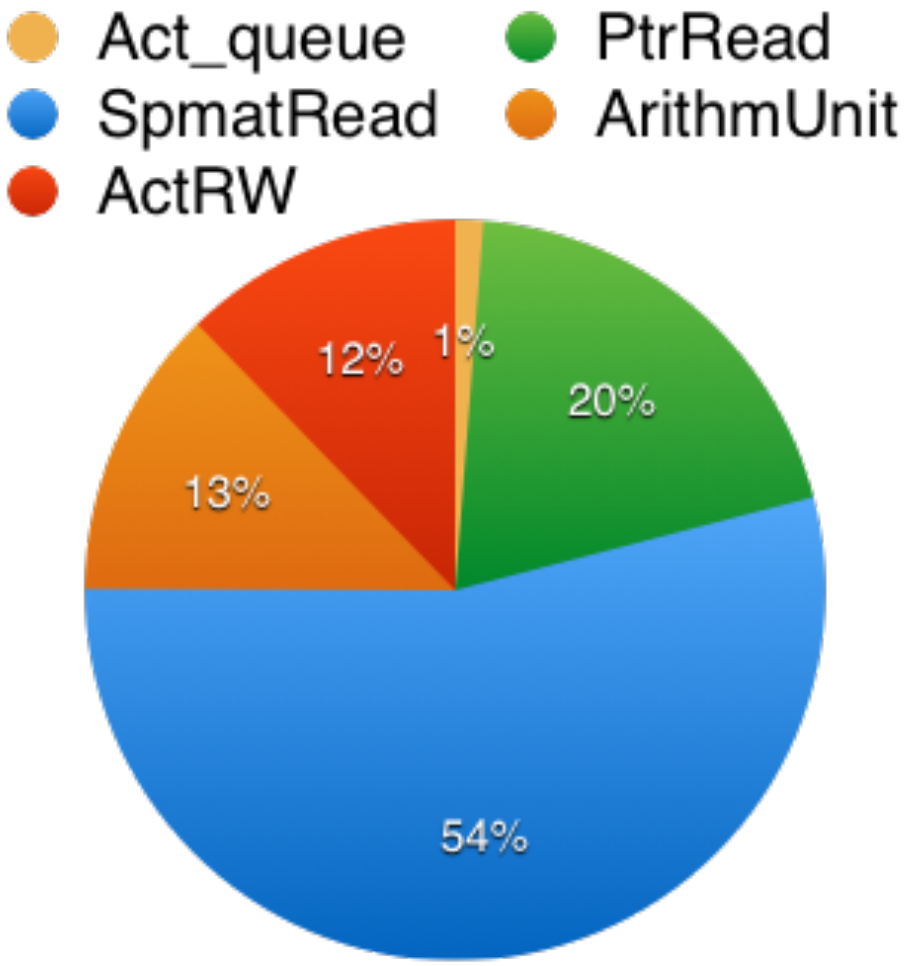


Technology	45 nm
# PEs	64
on-chip SRAM	8 MB
Max Model Size	84 Million
Static Sparsity	10x
Dynamic Sparsity	3x
Quantization	4-bit
ALU Width	16-bit
Area	40.8 mm^2
MxV Throughput	81,967 layers/s
Power	586 mW

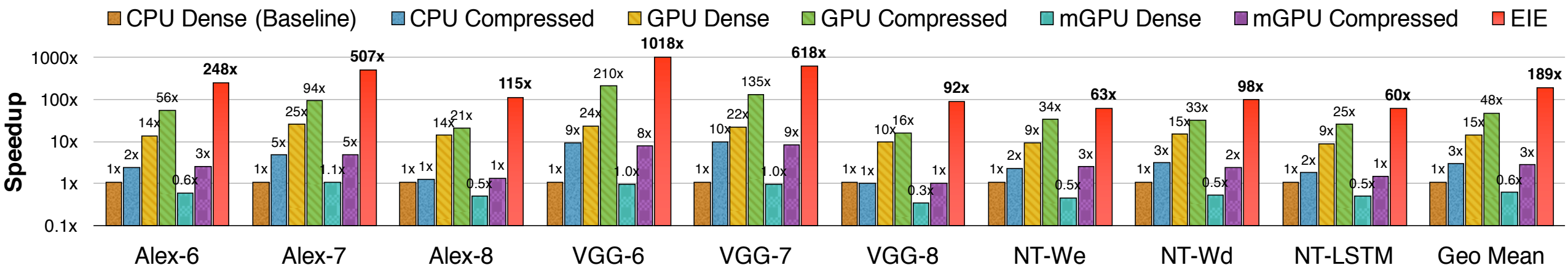
1. Post layout result
2. Throughput on AlexNet FC-7

Energy Distribution

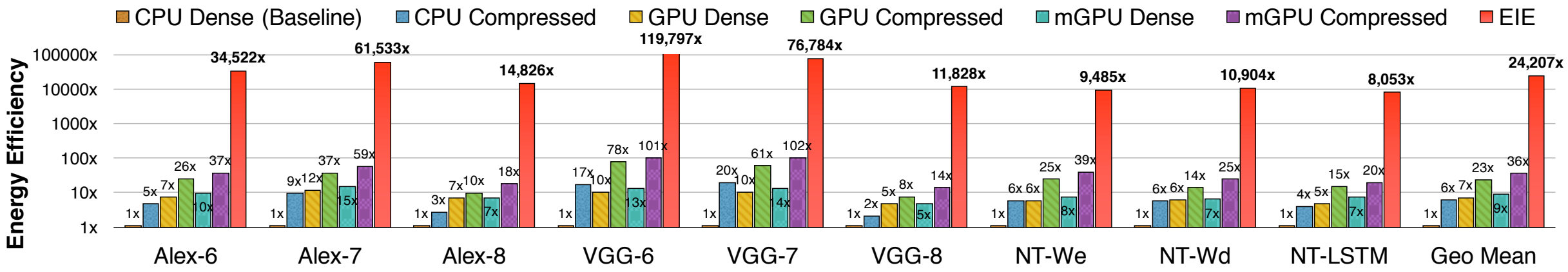




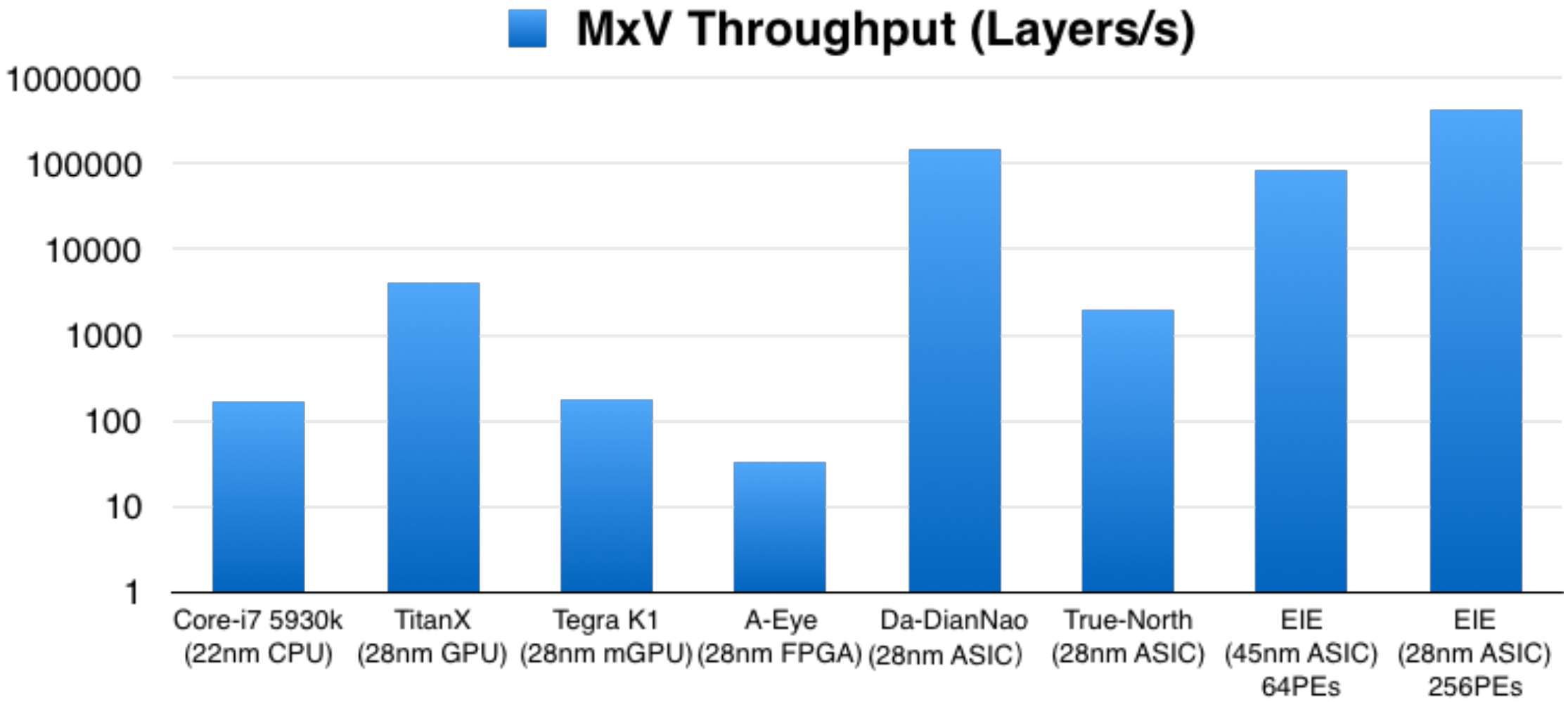
FC Layer: Speedup on EIE



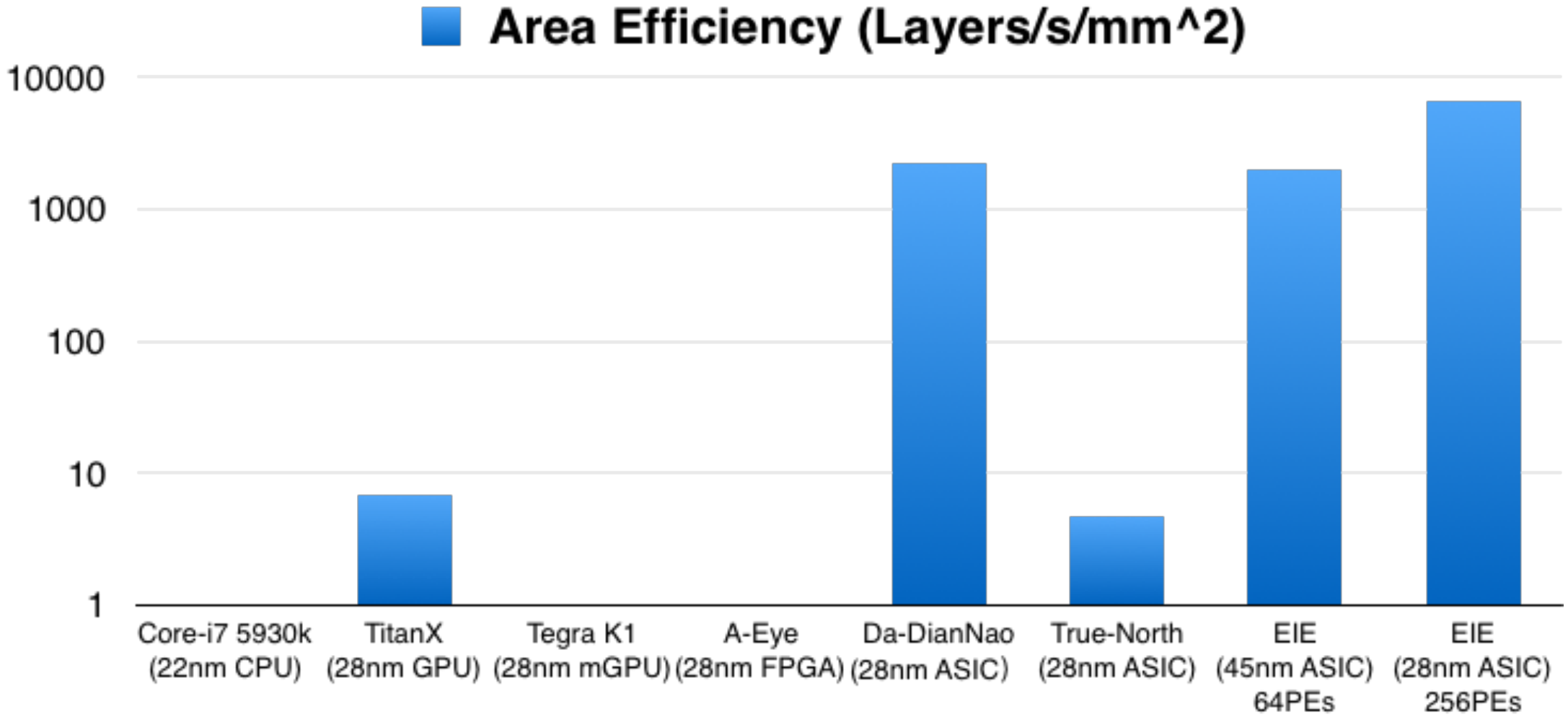
FC Layer: Energy Efficiency on EIE



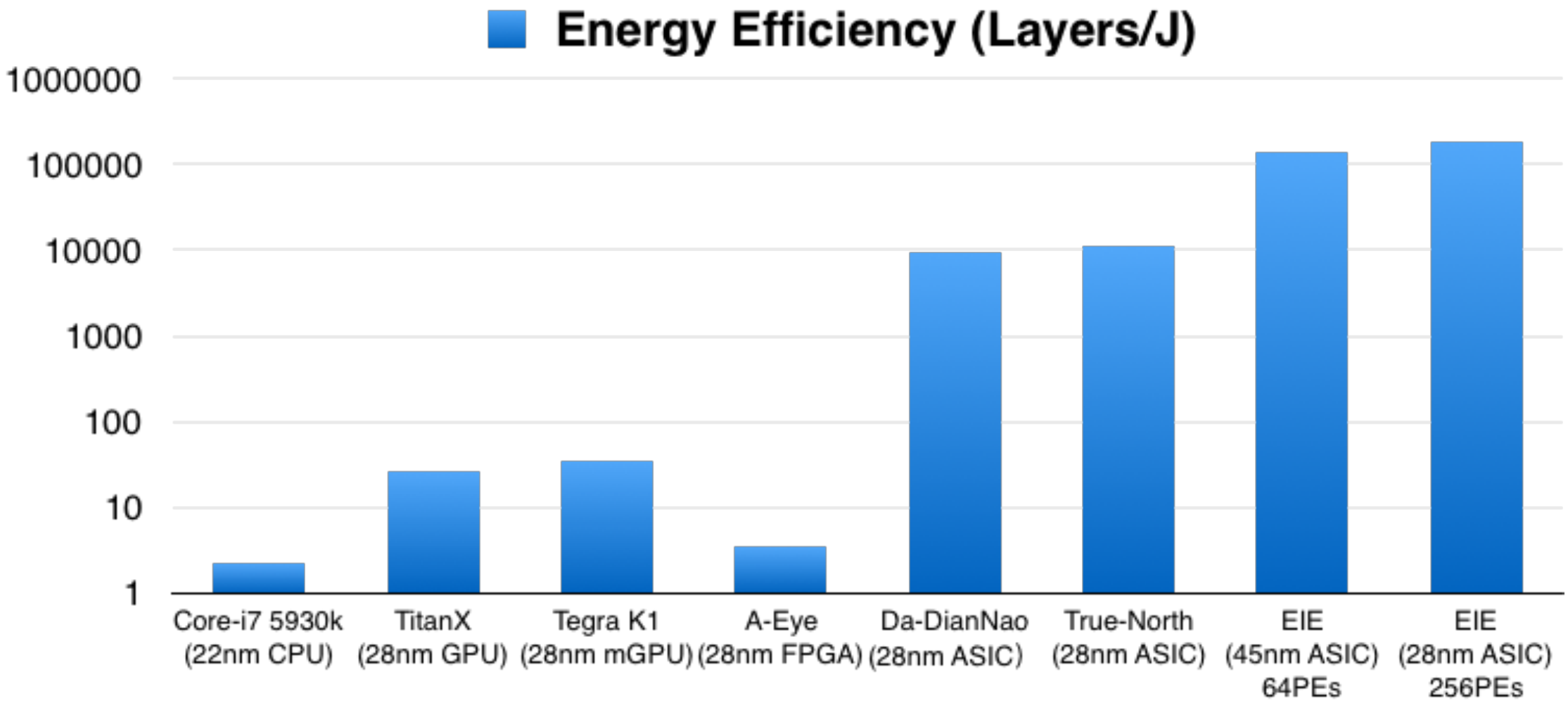
Comparison: Throughput



Comparison: Area Efficiency



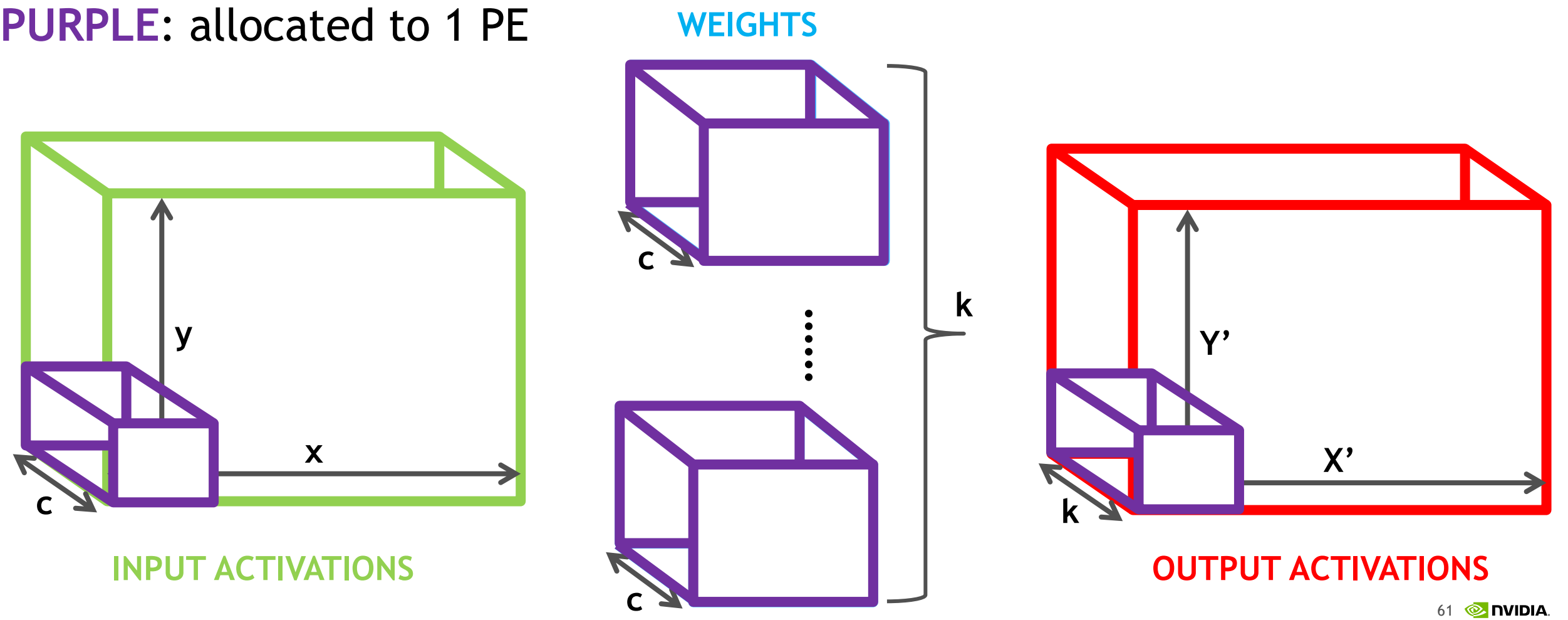
Comparison: Energy Efficiency



Sparse Convolutional Accelerator

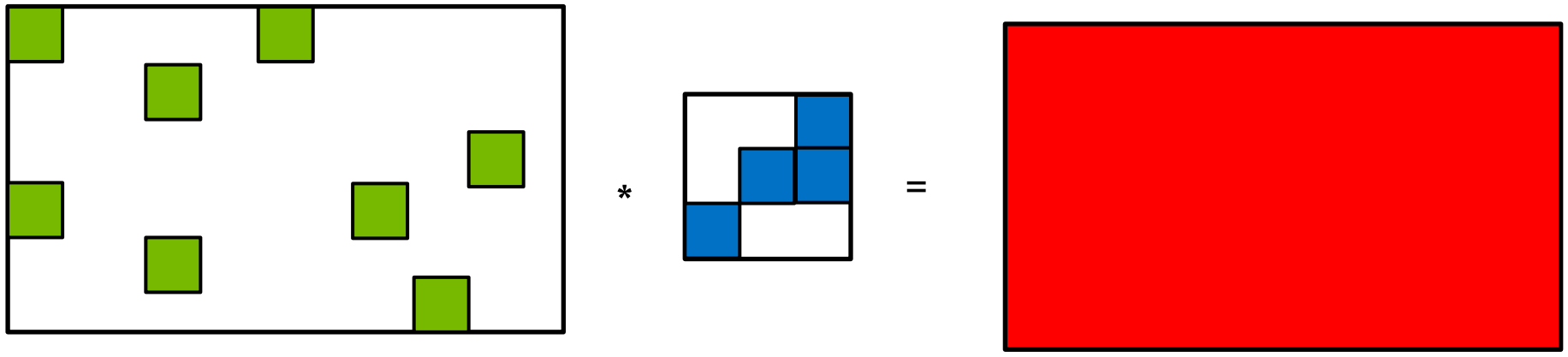
Blocking CNN Inference

PURPLE: allocated to 1 PE

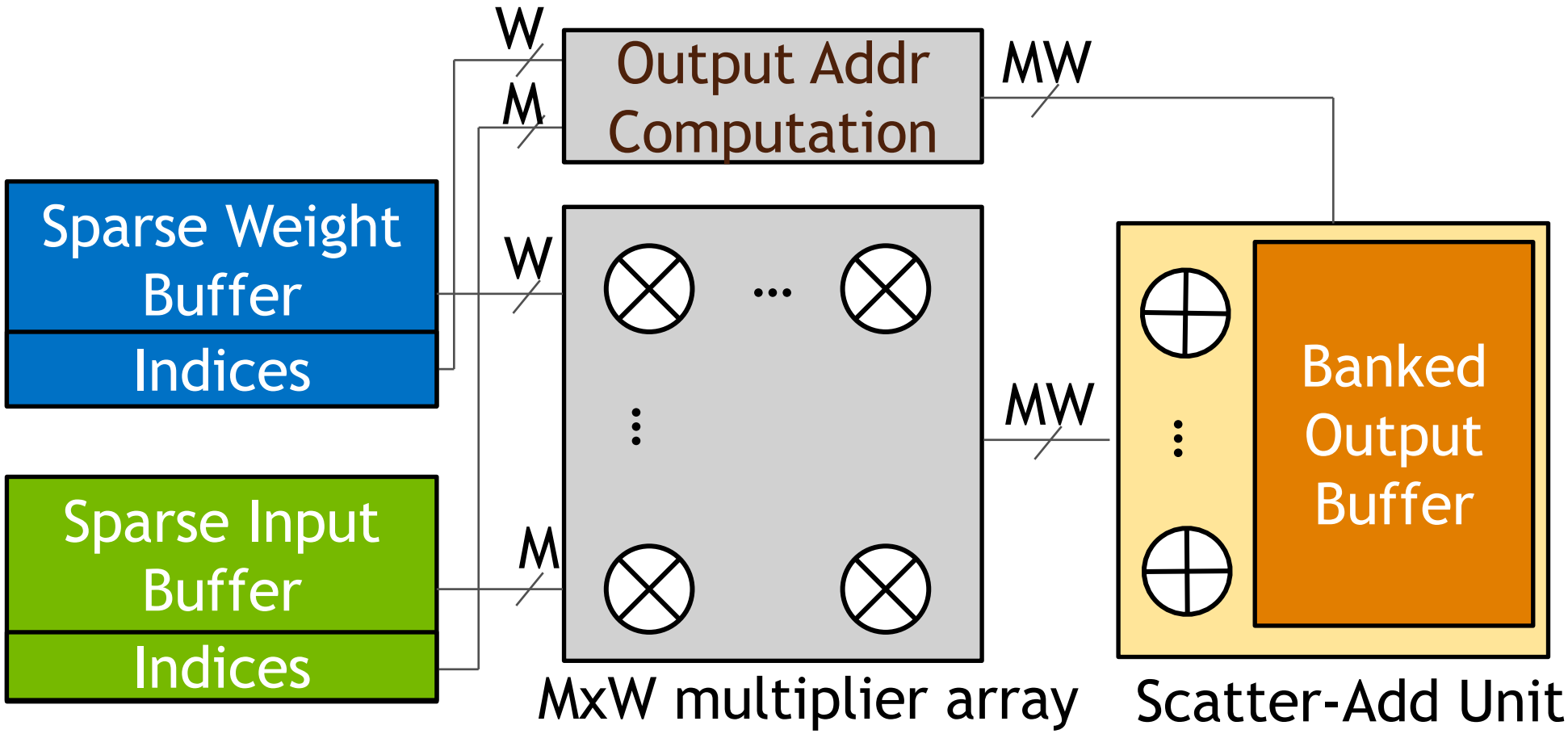


Sparse Convolution

- Only compute where both operands are nonzero
- 10-30x Reduction in work

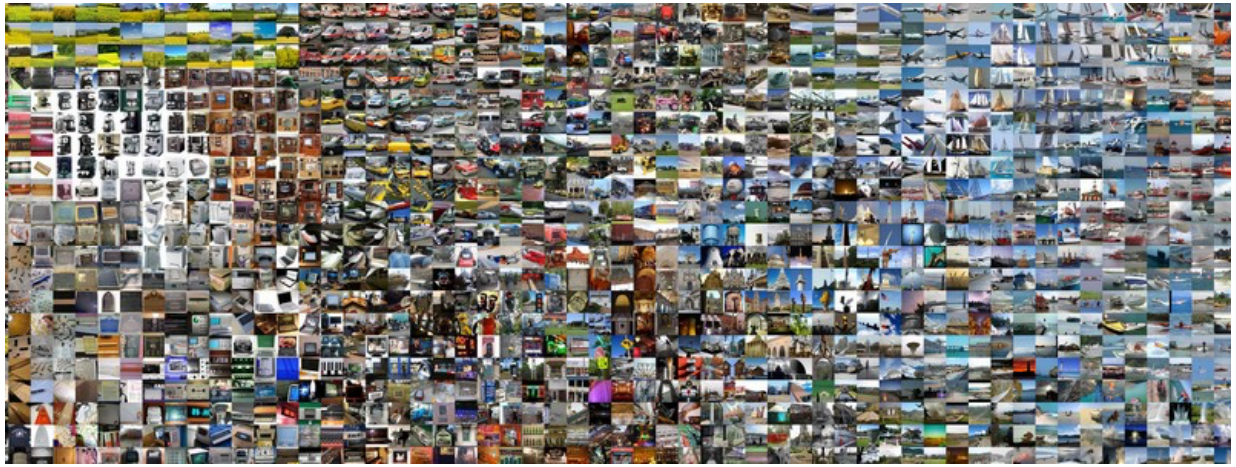


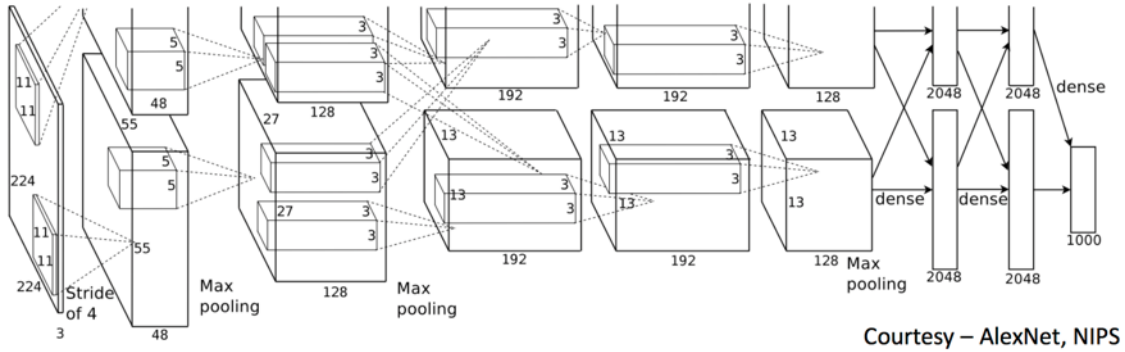
Sparse Convolution Engine



Conclusion

Hardware and Data enable DNNs





Courtesy – AlexNet, NIPS 2012

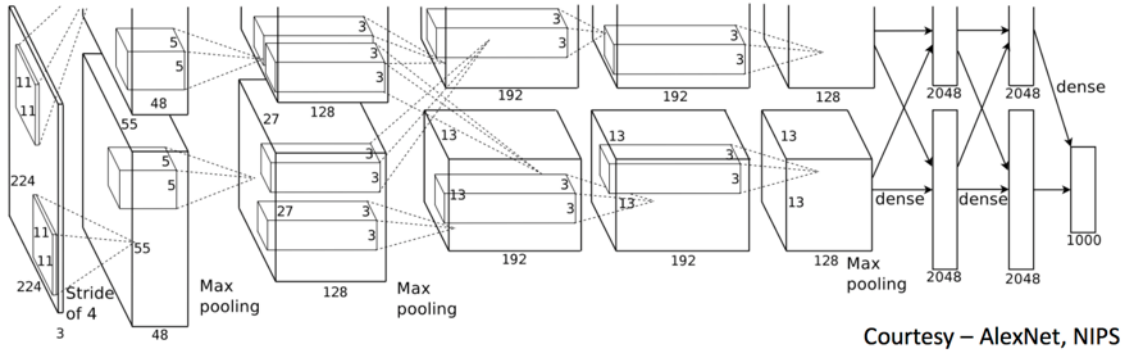
Summary

- **Hardware** has enabled the current resurgence of DNNs
  - And limits the size of today's networks
- **Inference**
  - Dynamically **sparse** activations x statically **sparse** weights
  - **8b weights** sufficient (can be compressed to **2-4b)**
  - Energy dominated by **data movement** and **buffering**
  - **Fixed-function hardware** will dominate **inference**
- **Training**
  - Only **dynamic sparsity** (3x activations, 2x dropout)
  - Medium precision (**FP16** – for weights)
  - **Large memory footprint** (batch x retained activations) – can be 10s – 100s of GB
  - Parallelism to 10PF today 100PF in near future (Communication BW)
  - **GPUs** will dominate **training**

4 Distinct Sub-problems

	Training	Inference	
Convolutional	32b FP Batch Activation Storage GPUs ideal Comm for Parallelism	Low-Precision Compressed Latency-Sensitive Fixed-Function HW Arithmetic Dominated	B x S Weight Reuse Act Dominated
Fully-Conn.	32b FP Batch Weight Storage GPUs ideal Comm. for Parallelism	Low-Precision Compressed Latency-Sensitive No weight reuse Fixed-Function HW Storage dominated	B Weight Reuse Weight Dominated
	32b FP - large batches Minimize Training Time Enables larger networks	8b Int - small (unit) batches Meet real-time constraint	

Thank You



Courtesy – AlexNet, NIPS 2012