Energy-Efficient Hardware for Embedded Vision and Deep Convolutional Neural Networks

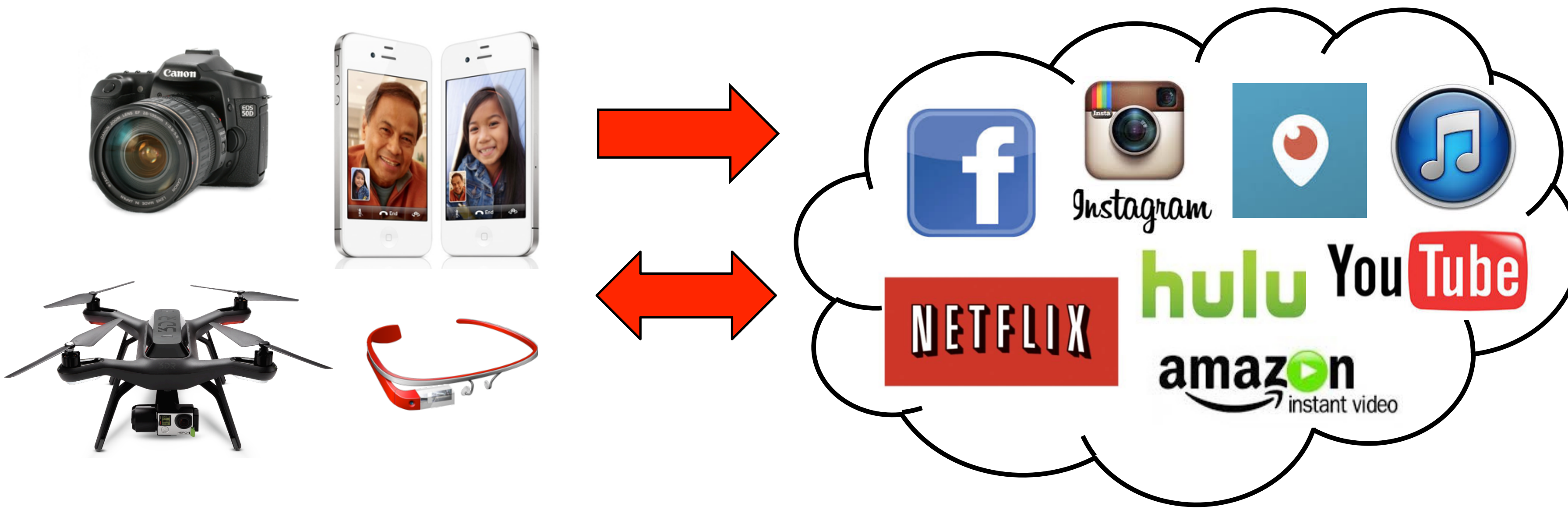
Vivienne Sze

Massachusetts Institute of Technology

Contact Info
email: sze@mit.edu
website: www.rle.mit.edu/eems

Video is the Biggest Big Data

Over 70% of today's Internet traffic is video
Over 300 hours of video uploaded to YouTube **<u>every minute</u>**
Over 500 million hours of video surveillance collected **<u>every day</u>**



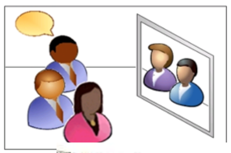
Energy limited due to battery capacity

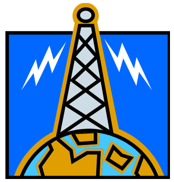
Power limited due to heat dissipation

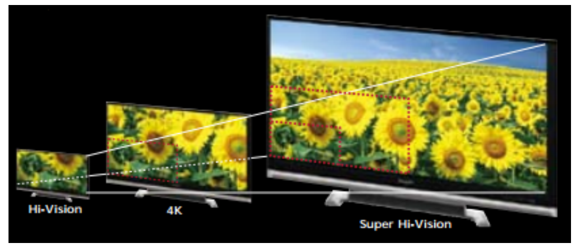
Need energy-efficient pixel processing!

Energy-Efficient Multimedia Systems Group

Next-Generation Video Coding (Compress Pixels)



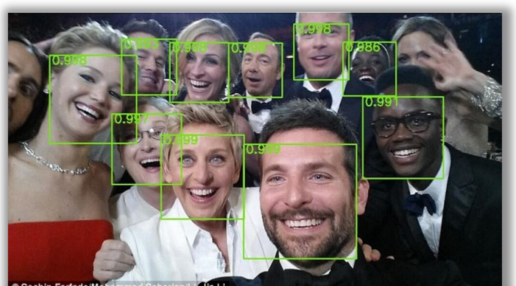




Ultra-HD

Goal: Increase coding efficiency, speed and energy-efficiency

Energy-Efficient Computer Vision & Deep Learning (Understand Pixels)



Recognition

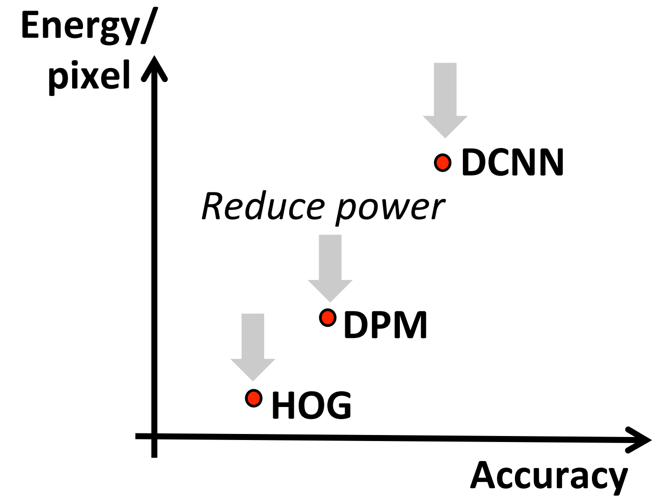
Self-Driving Cars

AI

Goal: Make computer vision as ubiquitous as video coding

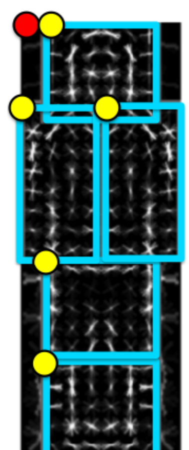
Features for Object Detection/Classification

- **Hand-crafted features**
  - Histogram of Oriented Gradients (HOG)
  - Deformable Parts Model (DPM)
- **Trained features (using machine learning)**
  - Deep Convolutional Neural Nets (DCNN)



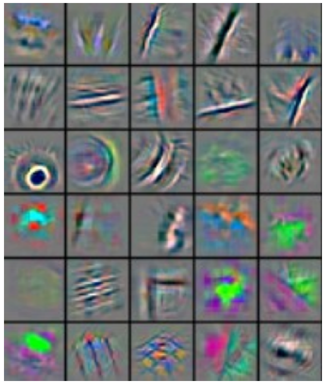
HOG
Rigid Template
based on edges

[Dalal, CVPR 2005]
Cited by 14500



DPM
Flexible Template
based on edges

[Felzenszwalb, PAMI 2010]
Cited by 4063

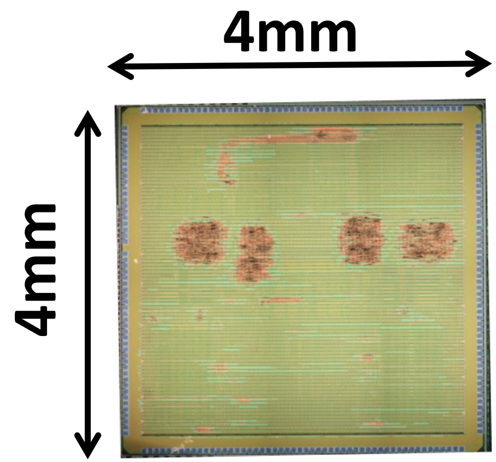


DCNN
High level
Abstraction

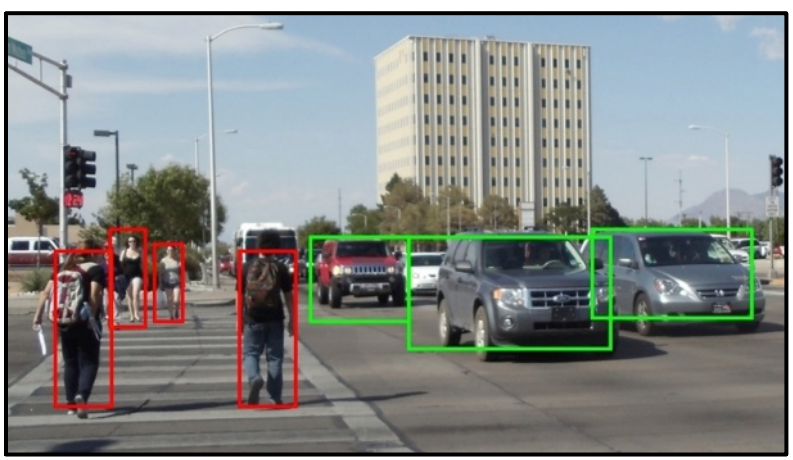
[Krizhevsky, NIPS 2012]
Cited by 4843

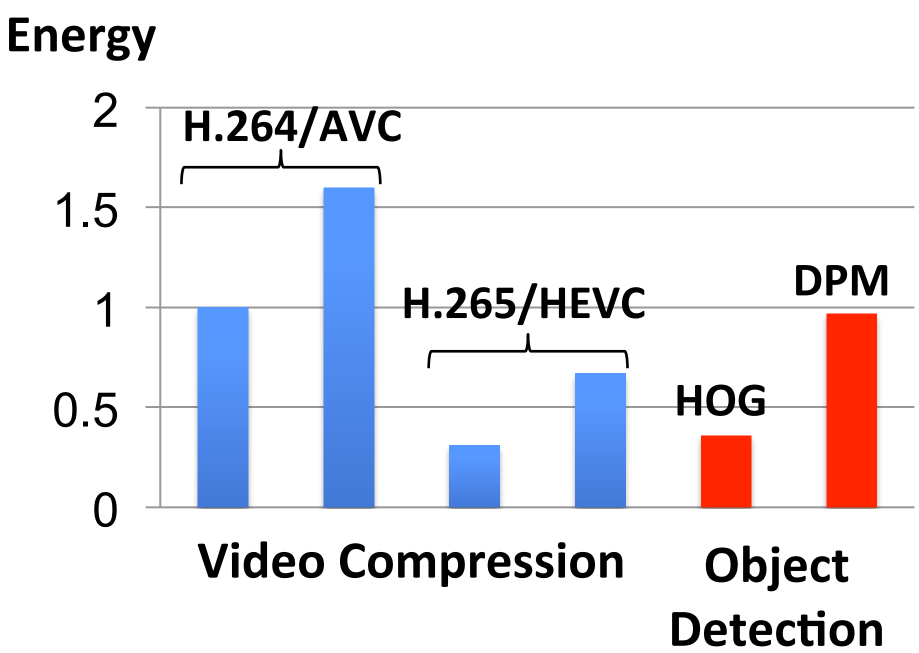
Typical Constraints on Video Coding

- **Area cost**
  - **Memory Size 100-500kB**
- **Power budget**
  - **< 1W for smartphones**
- **Throughput**
  - **Real-time 30 fps**
- **Energy**
  - **~1nJ/pixel**



MIT Object Detection Chip [VLSI 2016] [paper]

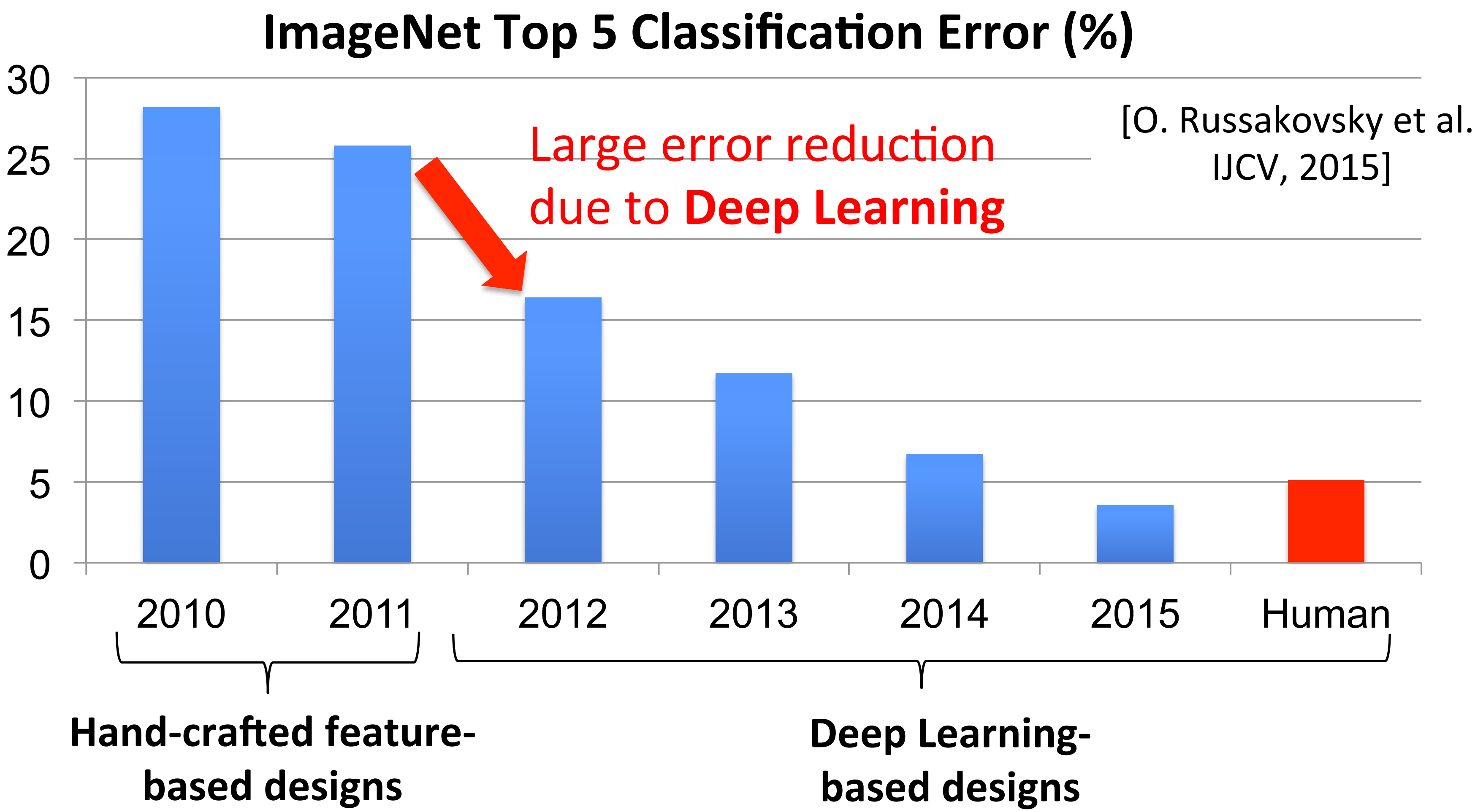




Eyeriss: Energy-Efficient Hardware for DCNNs

Yu-Hsin Chen, Tushar Krishna, Joel Emer, Vivienne Sze, ISSCC 2016 [paper] / ISCA 2016 [paper]

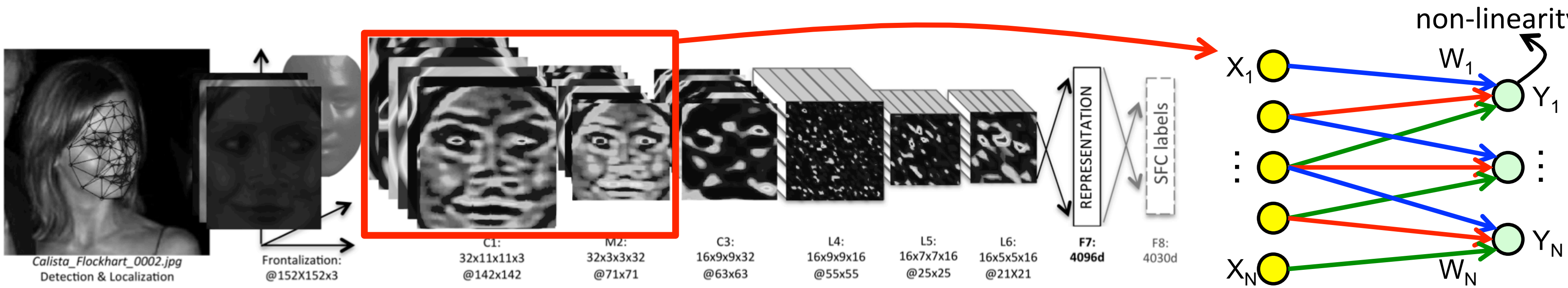
Increased Accuracy with Deep Learning



Deep Learning requires significantly more computation than previous approaches

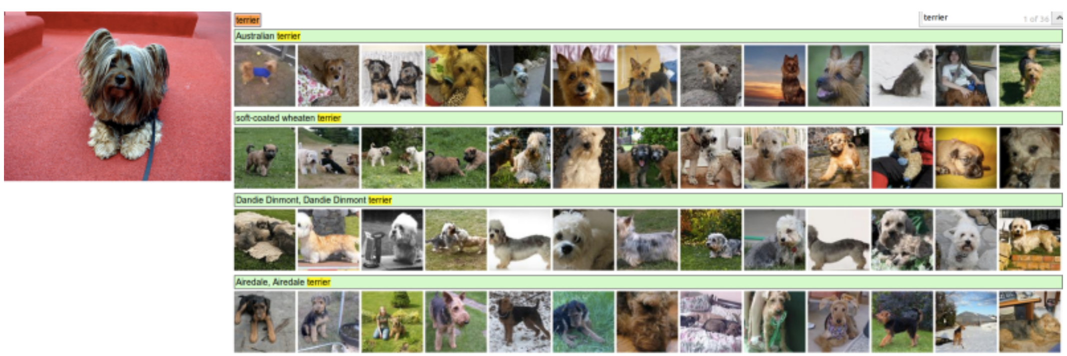
Human or *Superhuman* Accuracy Level

- Face recognition
  - Deep learning accuracy (97.25%) vs. Human accuracy (97.53%)



[Yaniv et al., CVPR 2014]

- Fine grained category recognition (e.g. dogs, monkeys, snakes, birds)
  - Deep learning errors: 7 vs. Human errors: 28



120 species of dogs

[O. Russakovsky et al., IJCV 2015]

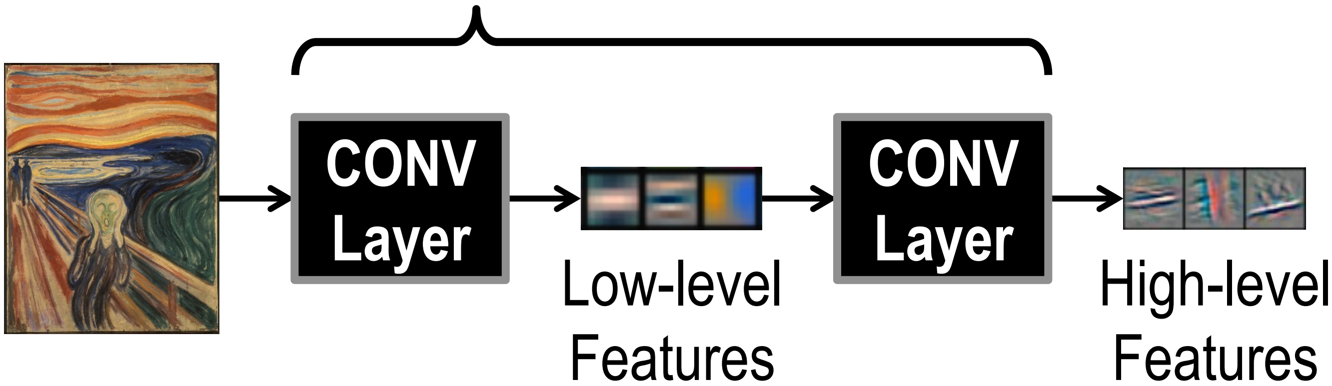
AlphaGo using Deep Learning

Go is exponentially more complex than chess (10^{170} legal positions)

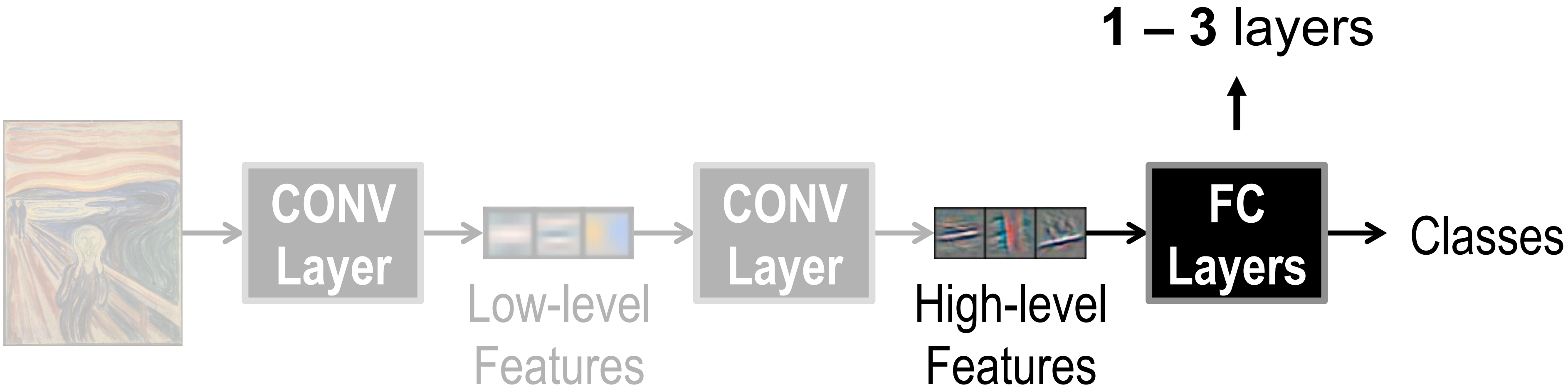
Google's AlphaGo, a computer algorithm, beat Go world champion Lee Sedol 4 to 1

Deep Convolutional Neural Networks

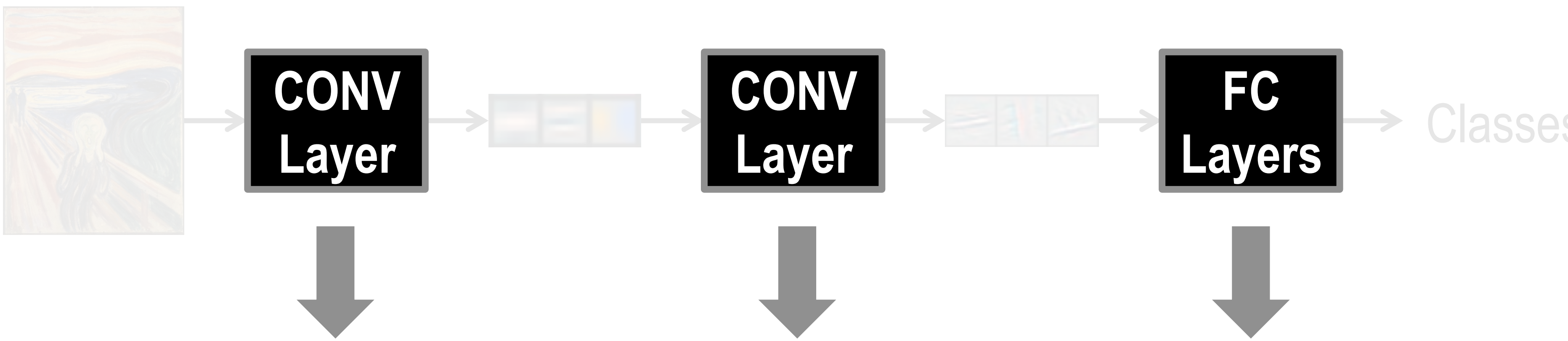
Modern ***deep*** CNN: up to **1000** CONV layers

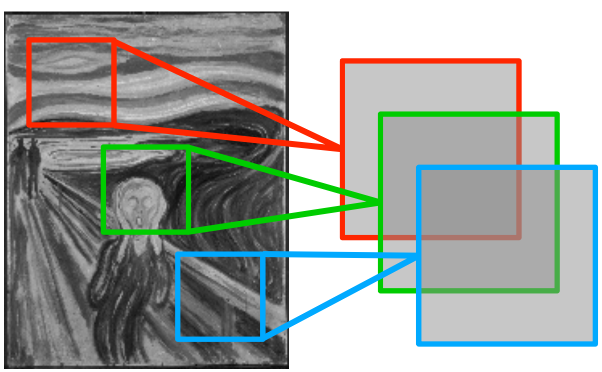


Deep Convolutional Neural Networks



Deep Convolutional Neural Networks



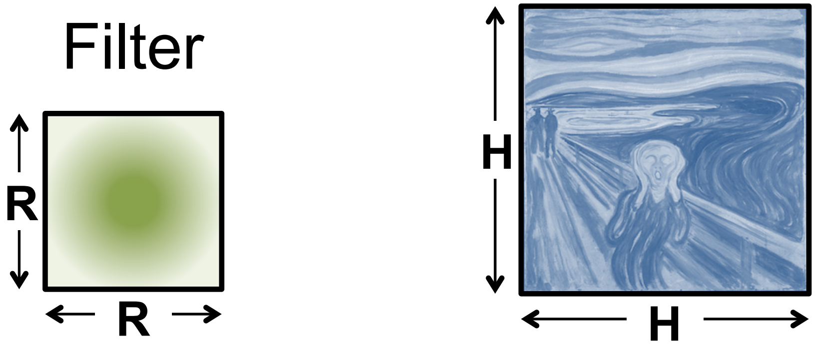


Convolutions account for more than 90% of overall computation, dominating **runtime** and **energy consumption**

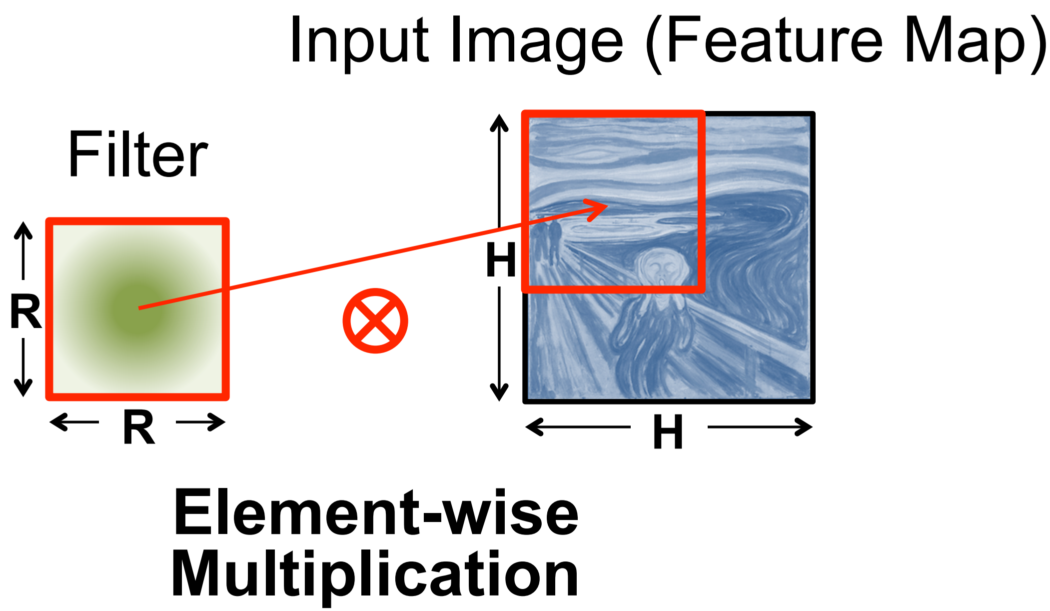
High-Dimensional CNN Convolution

Input Image (Feature Map)

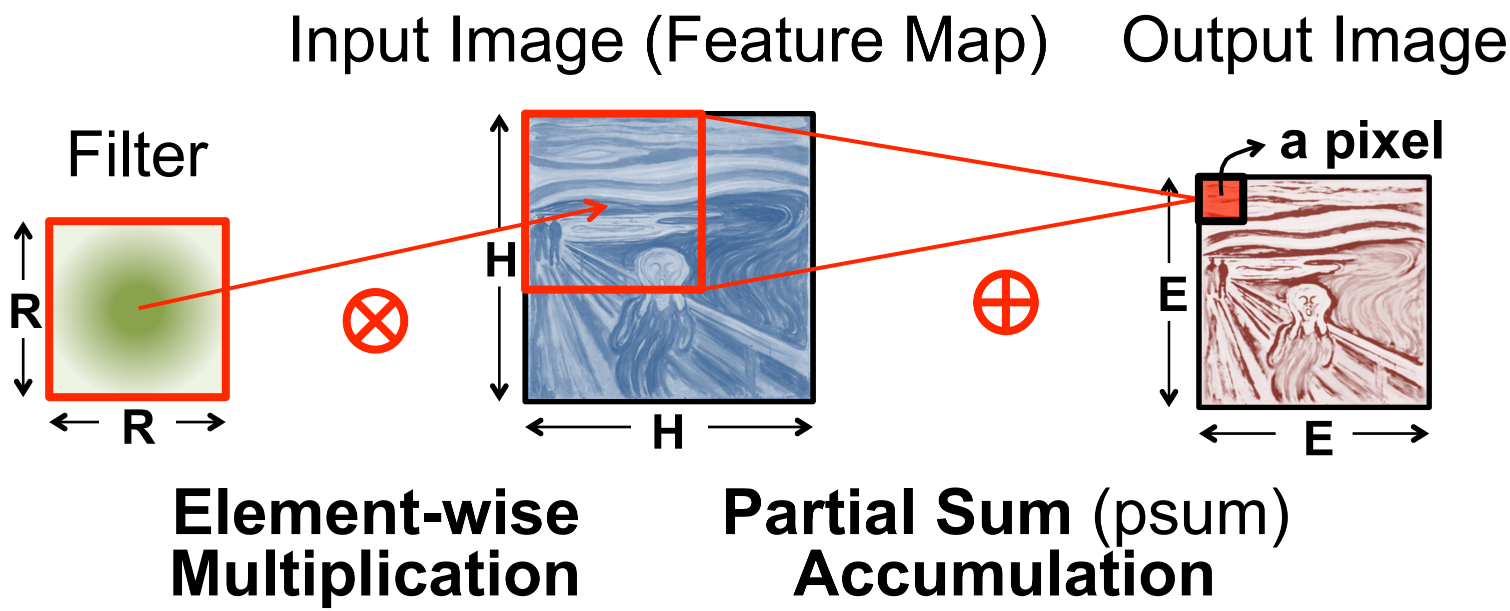
Filter



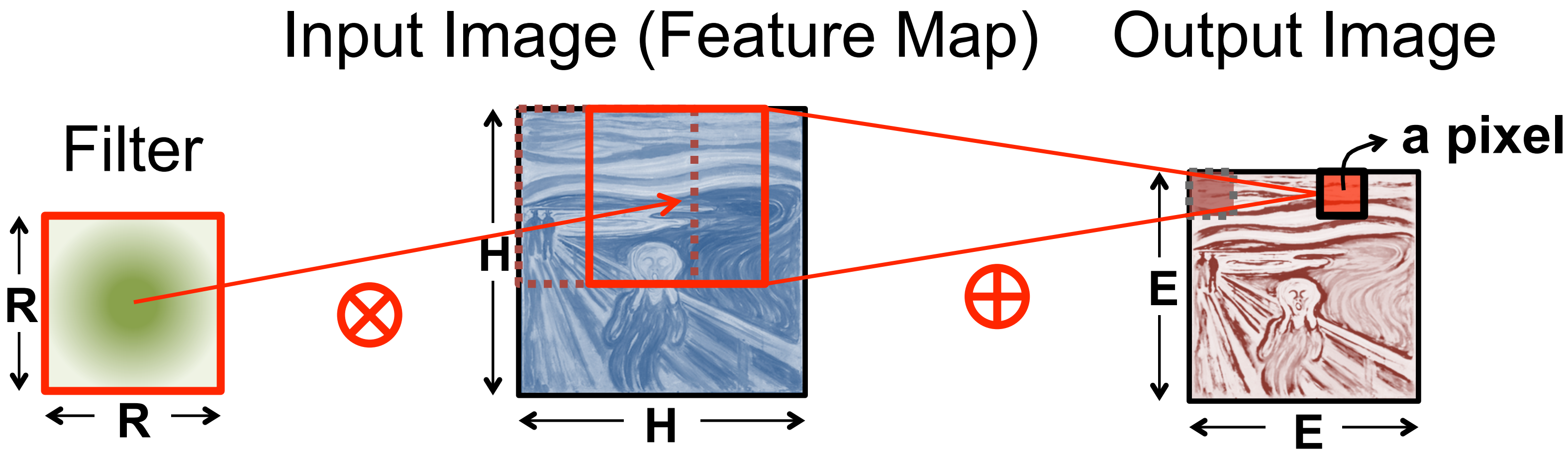
High-Dimensional CNN Convolution



High-Dimensional CNN Convolution

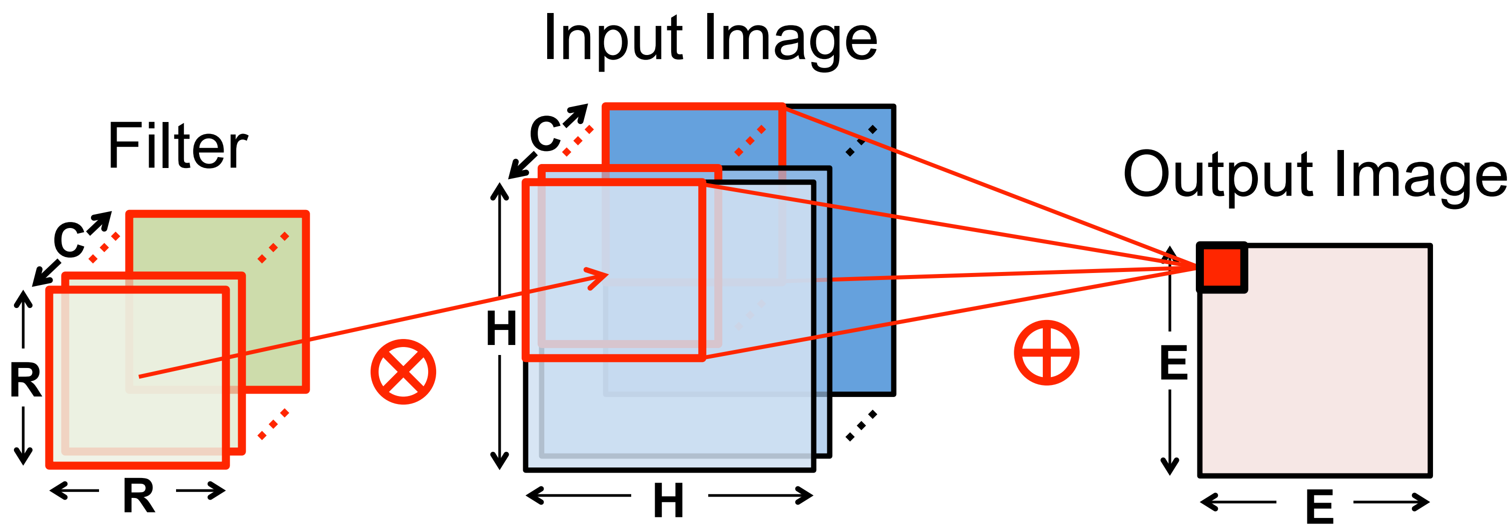


High-Dimensional CNN Convolution



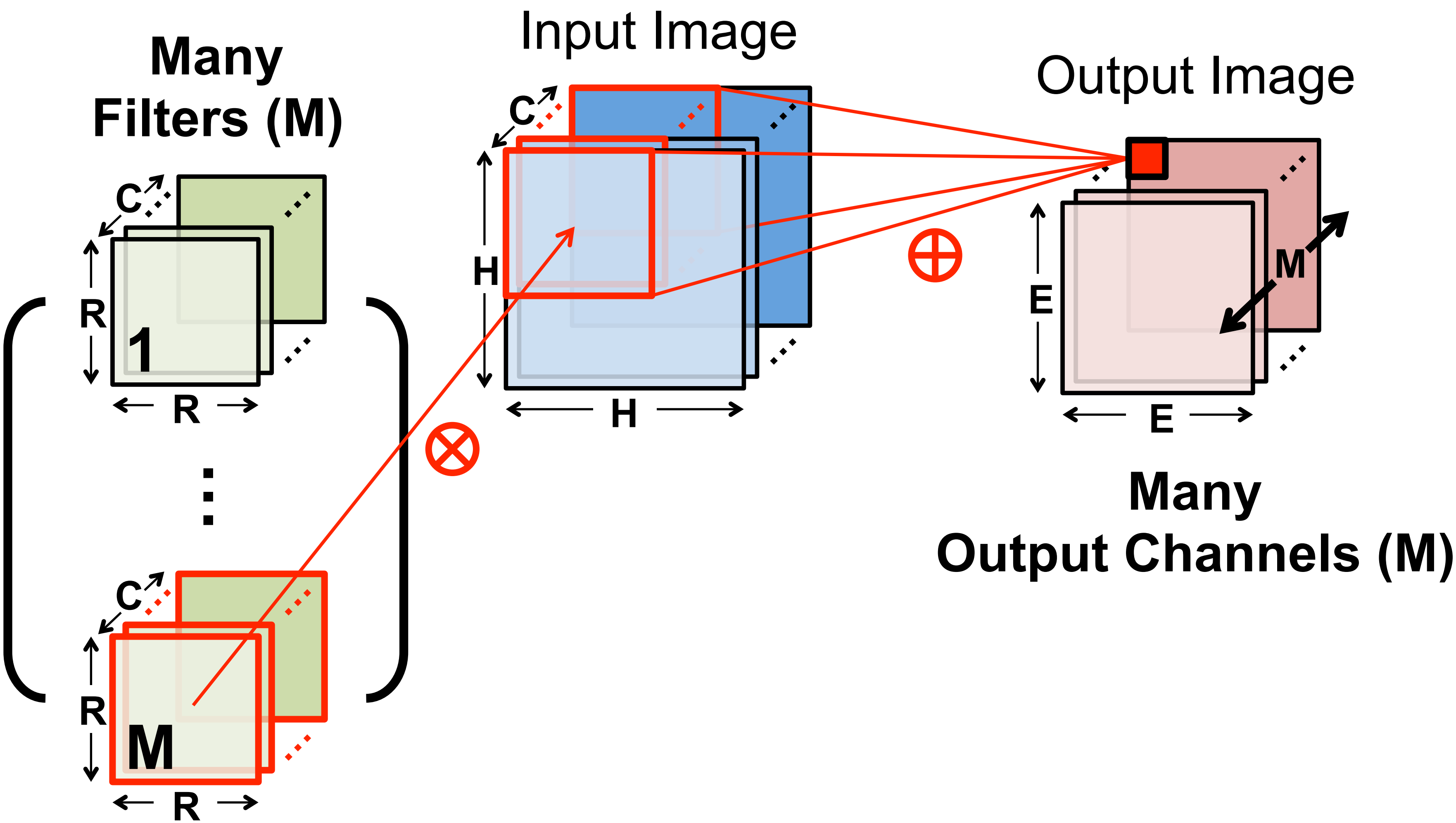
Sliding Window Processing

High-Dimensional CNN Convolution



Many Input Channels (C)

High-Dimensional CNN Convolution



AlexNet: 96 – 384 Filters (M)

High-Dimensional CNN Convolution

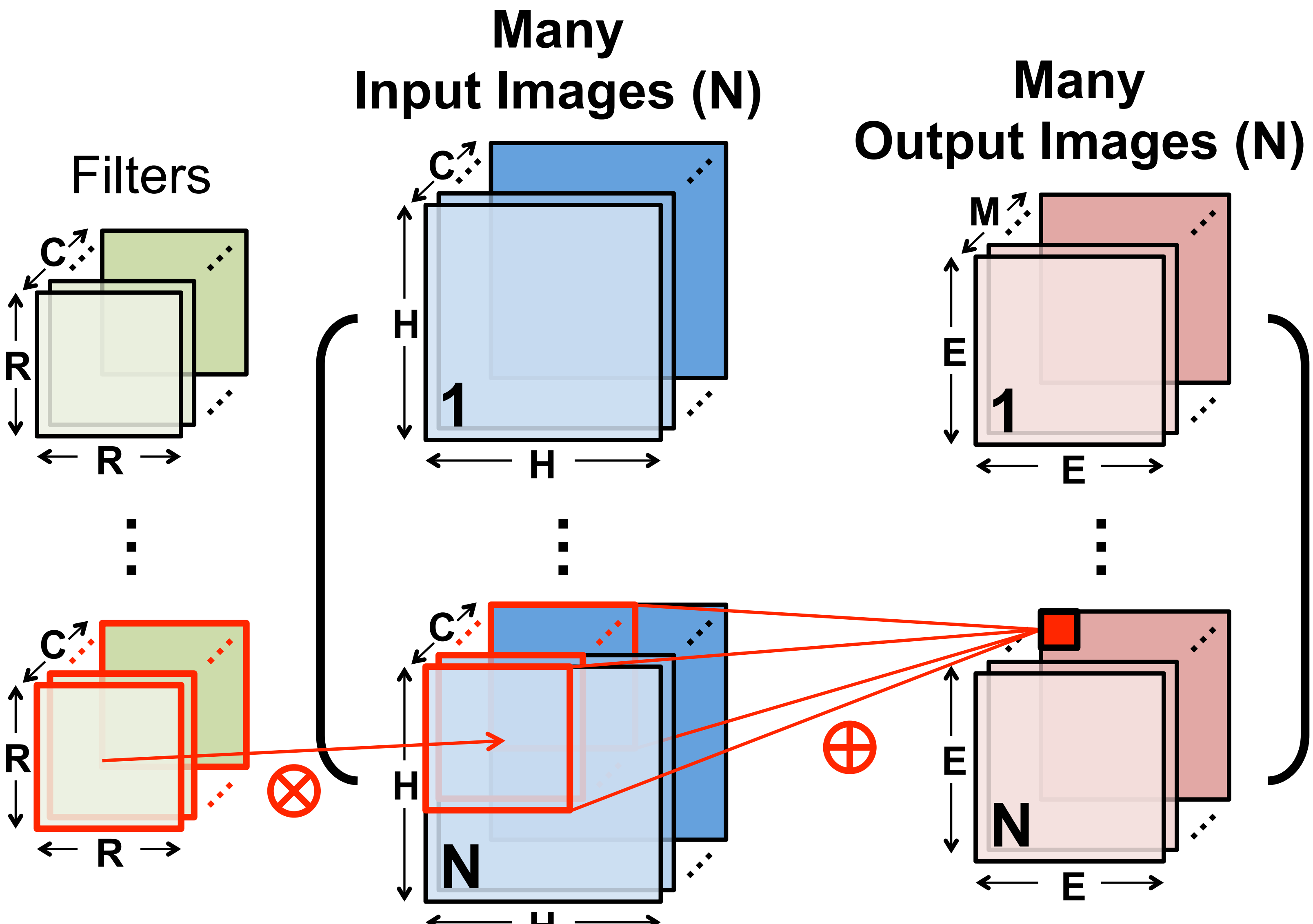


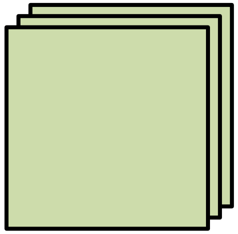
Image batch size: 1 – 256 (N)

Large Sizes with Varying Shapes

AlexNet[1] Convolutional Layer Configurations

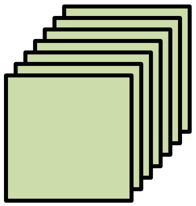
Layer	Filter Size (R)	# Filters (M)	# Channels (C)	Stride
1	11x11	96	3	4
2	5x5	256	48	1
3	3x3	384	256	1
4	3x3	384	192	1
5	3x3	256	192	1

Layer 1



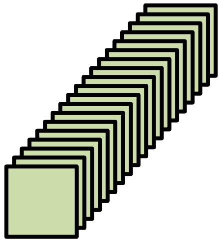
34k Params
105M MACs

Layer 2



307k Params
224M MACs

Layer 3



885k Params
150M MACs

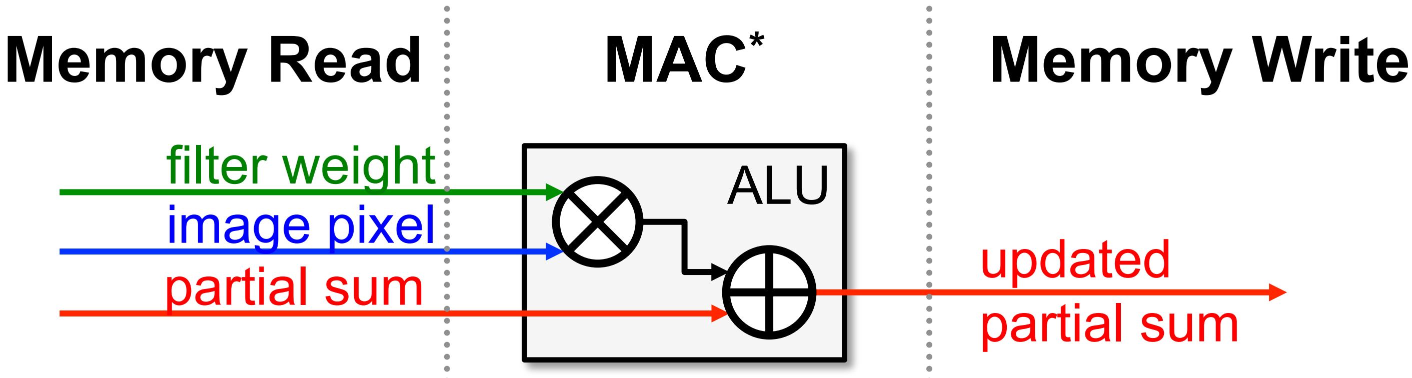
1. [Krizhevsky, NIPS 2012]

Properties We Can Leverage

- Operations exhibit **high parallelism**
  → **high throughput** possible

Properties We Can Leverage

- Operations exhibit **high parallelism**
  → **high throughput** possible
- Memory Access is the Bottleneck

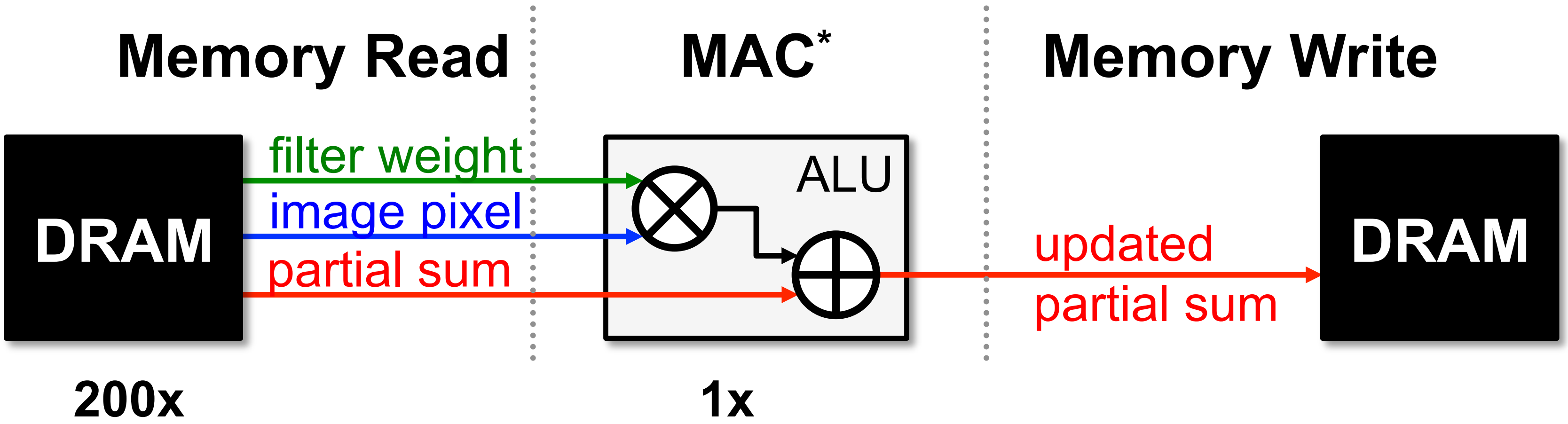


* multiply-and-accumulate

Properties We Can Leverage

- Operations exhibit **high parallelism**
  → **high throughput** possible

- Memory Access is the Bottleneck

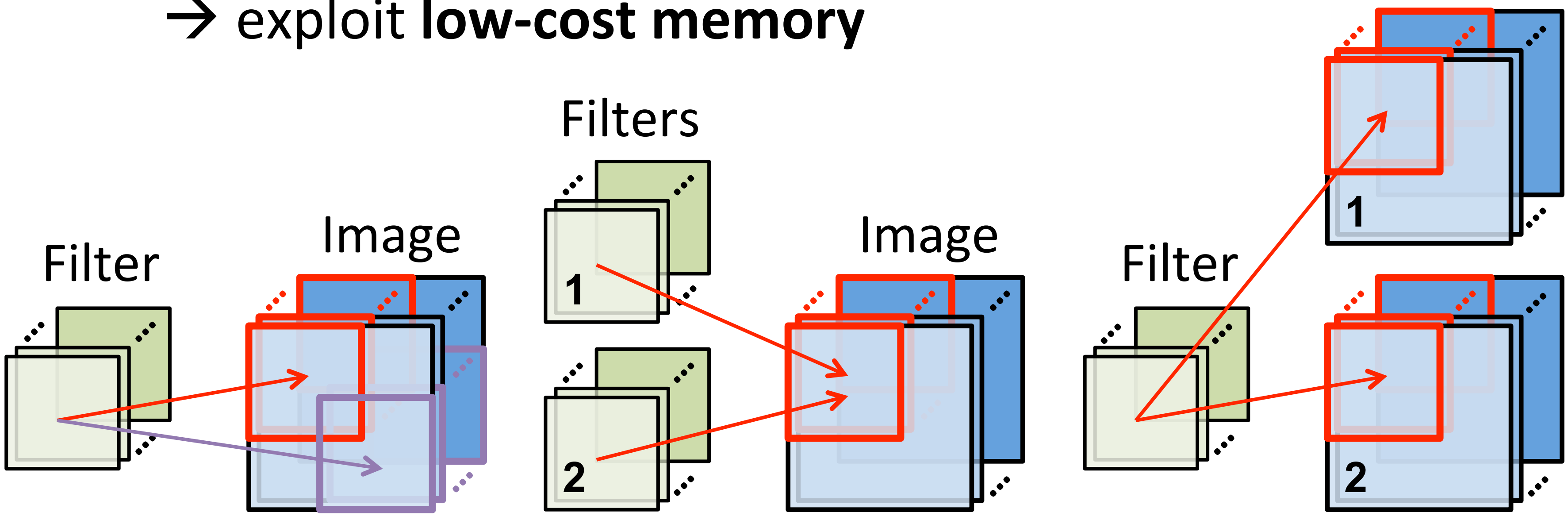


Worst Case: all memory R/W are **DRAM** accesses

- Example: AlexNet [NIPS 2012] has **724M** MACs
  → **2896M** DRAM accesses required

Properties We Can Leverage

- Operations exhibit **high parallelism**
  → **high throughput** possible
- **Input data reuse** opportunities (**up to 500x**)
  → exploit **low-cost memory**



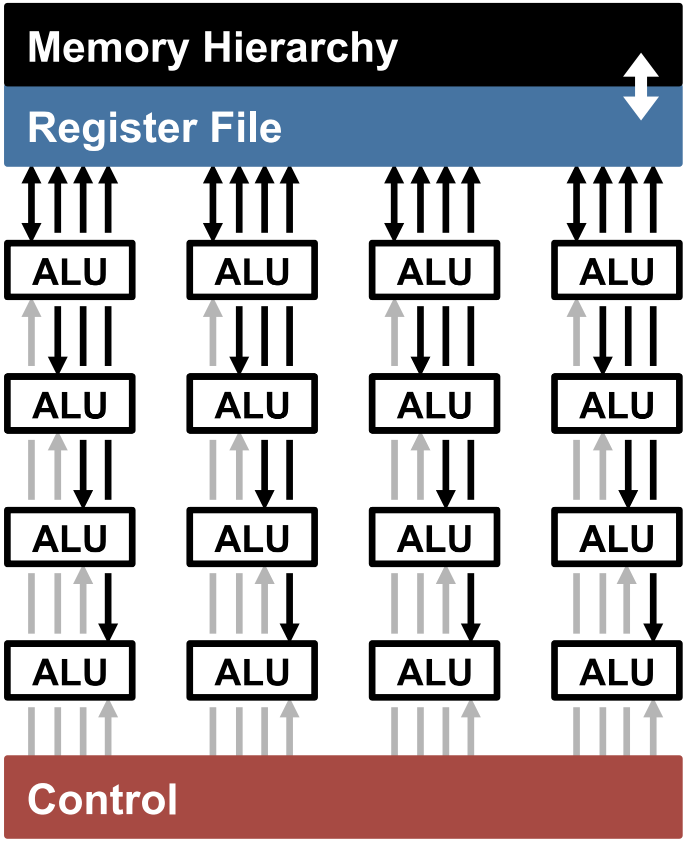
Convolutional Reuse (pixels, weights)

Image Reuse (pixels)

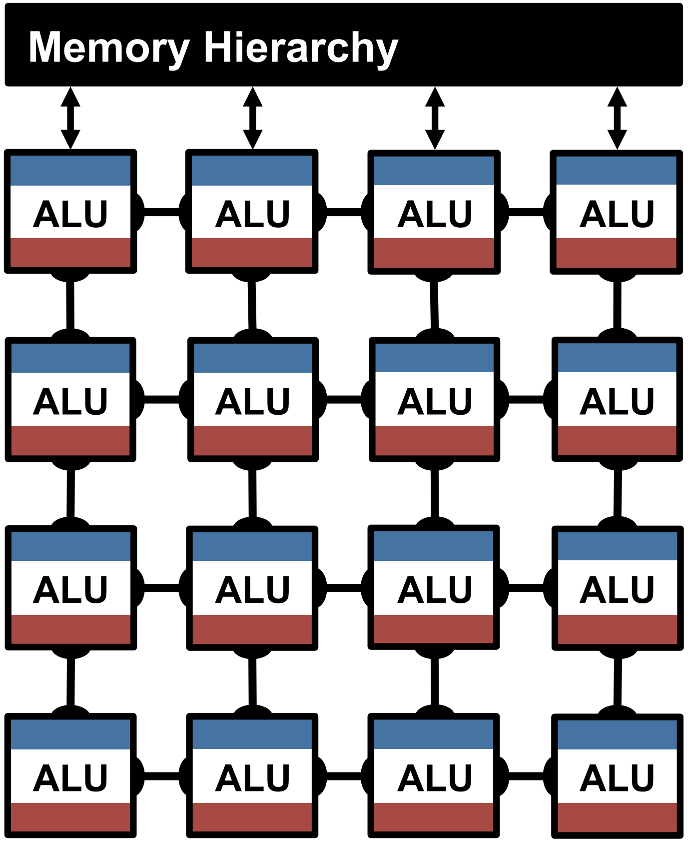
Filter Reuse (weights)

Highly-Parallel Compute Paradigms

Temporal Architecture (SIMD/SIMT)



Spatial Architecture (Dataflow Processing)



Advantages of Spatial Architecture

Temporal Architecture
(SIMD/SIMT)

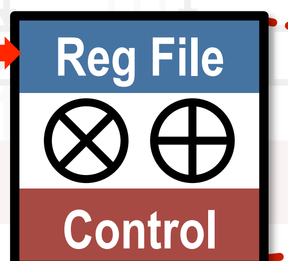
Spatial Architecture
(Dataflow Processing)

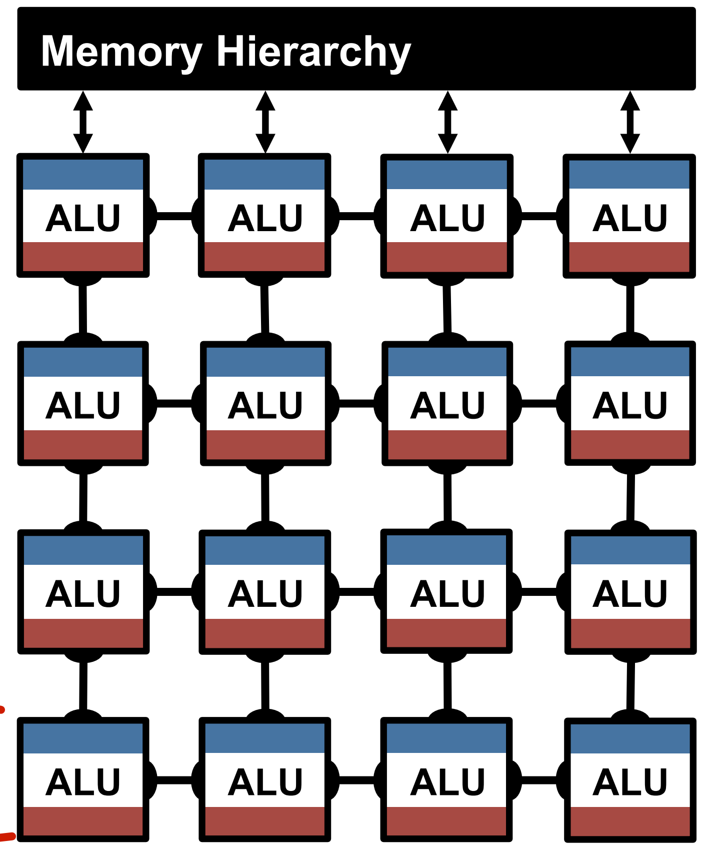
Efficient Data Reuse
Distributed local storage (RF)

Inter-PE Communication
Sharing among regions of PEs

Processing Element (PE)

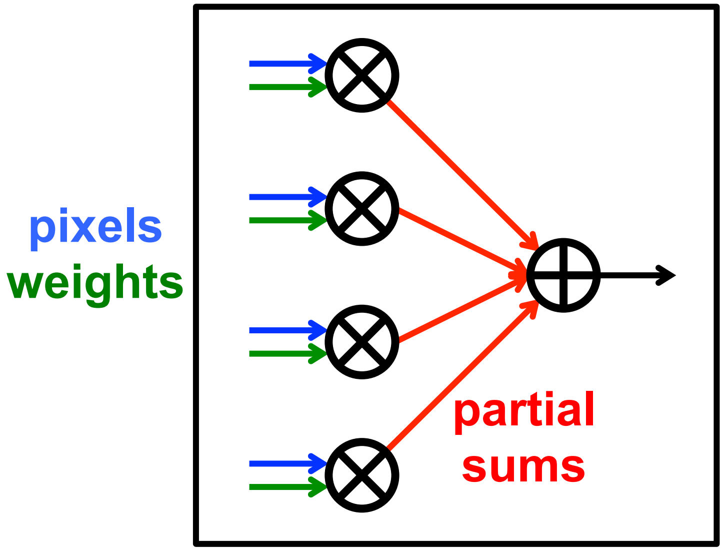
0.5 – 1.0 kB



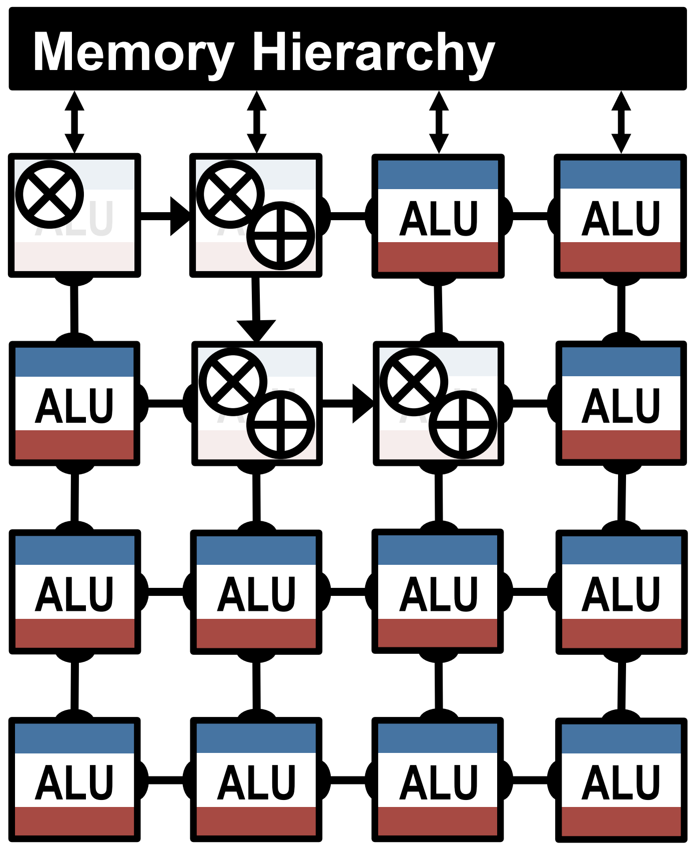


How to Map the Dataflow?

CNN Convolution



Spatial Architecture (Dataflow Processing)



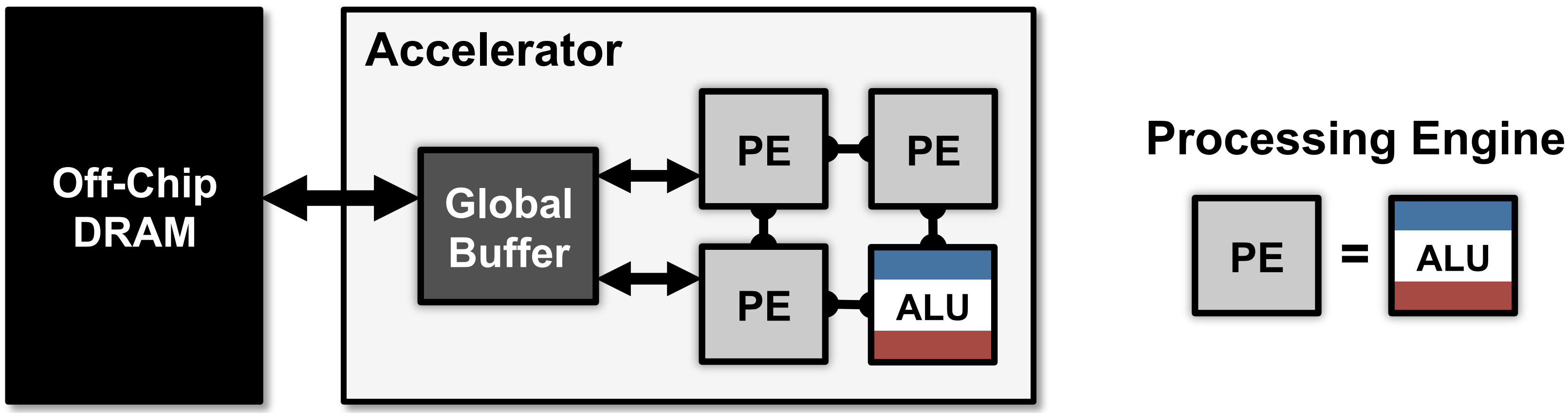
Goal: Increase reuse of input data (**weights** and **pixels**) and local **partial sums** accumulation

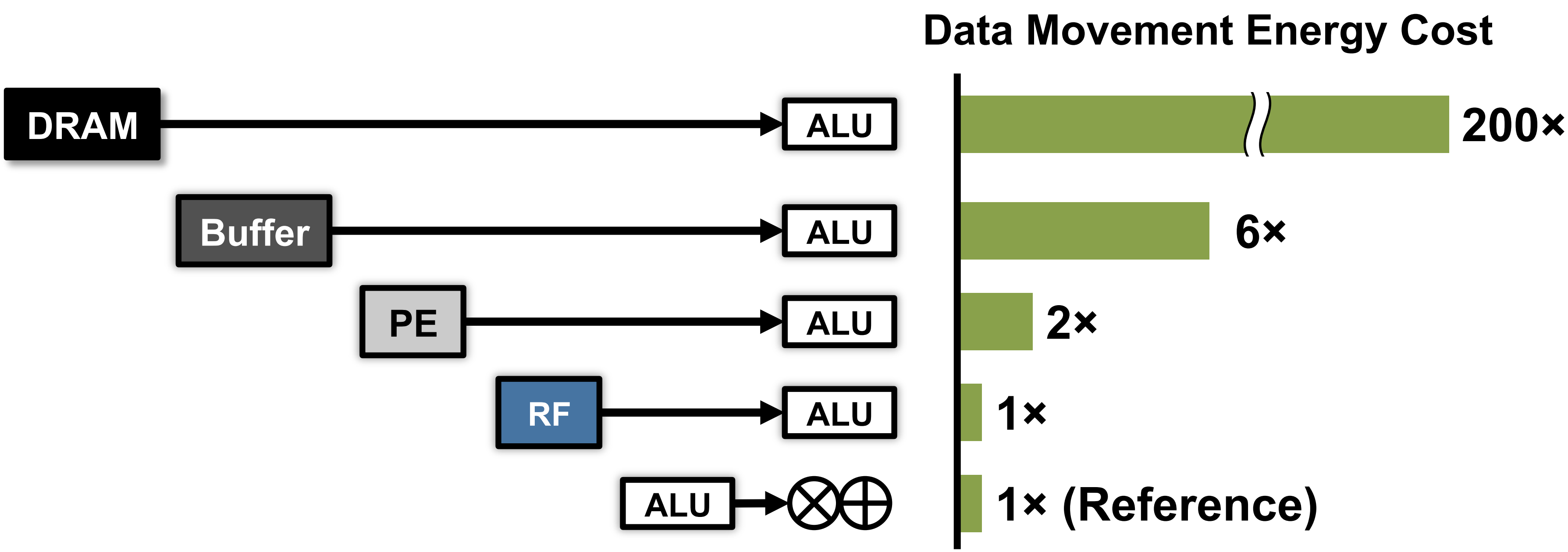
Energy-Efficient Dataflow

Yu-Hsin Chen, Joel Emer, Vivienne Sze, ISCA 2016 [paper]

Maximize data reuse and accumulation at RF

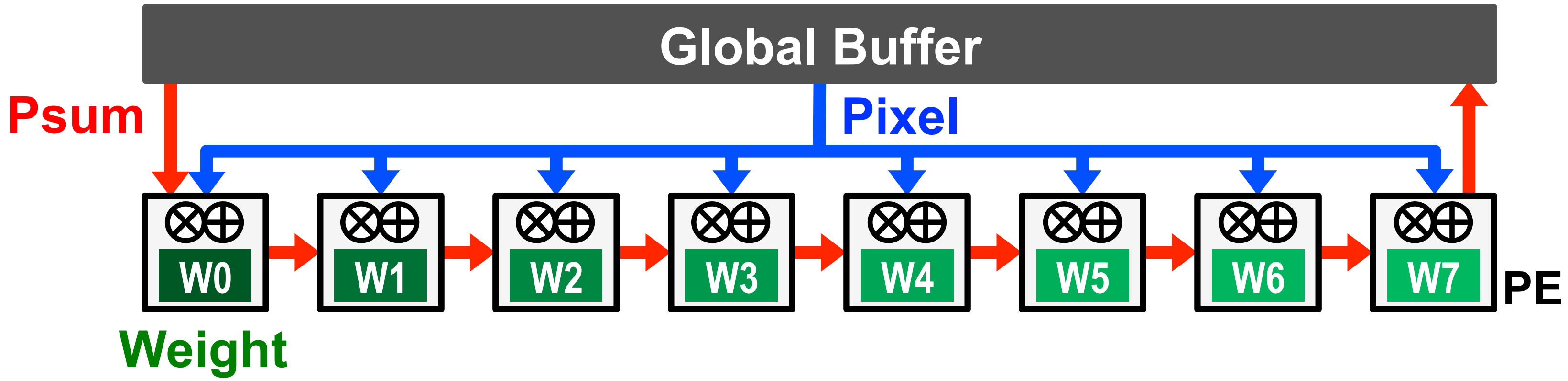
Data Movement is Expensive





Maximize data reuse at lower levels of hierarchy

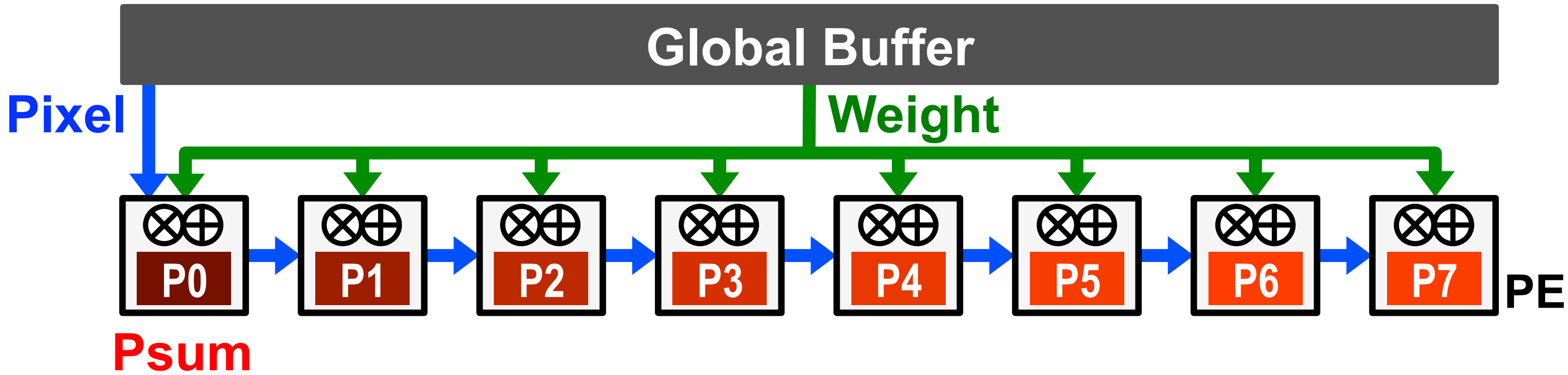
Weight Stationary (WS)



- **Minimize** **weight** read energy consumption
  - maximize convolutional and filter reuse of weights
- **Examples:**

  [**Chakradhar**, *ISCA* 2010] [**nn-X (NeuFlow)**, *CVPRW* 2014]
  [**Park**, *ISSCC* 2015] [**Origami**, *GLSVLSI* 2015]

Output Stationary (OS)



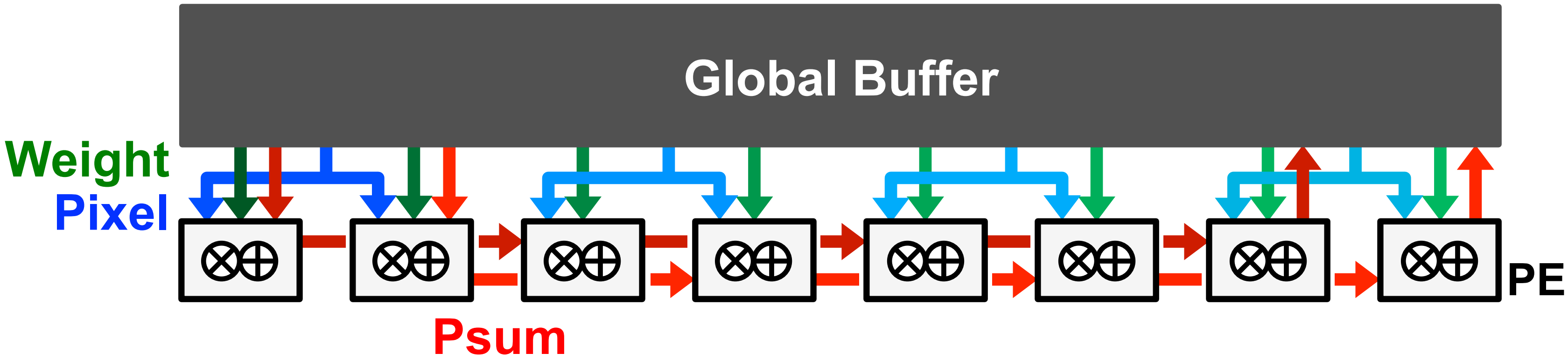
- **Minimize** **partial sum** R/W energy consumption
  - maximize local accumulation

- **Examples:**

  [**Gupta**, *ICML* 2015] [**ShiDianNao**, *ISCA* 2015]

  [**Peemen**, *ICCD* 2013]

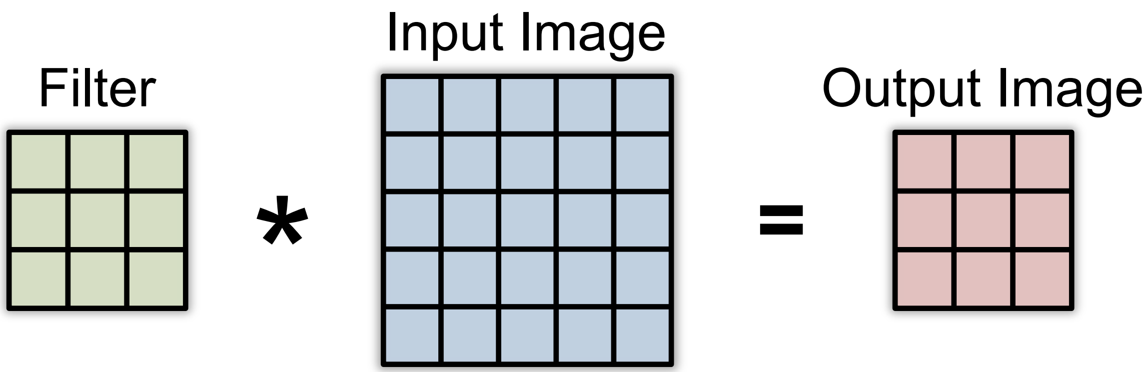
No Local Reuse (NLR)



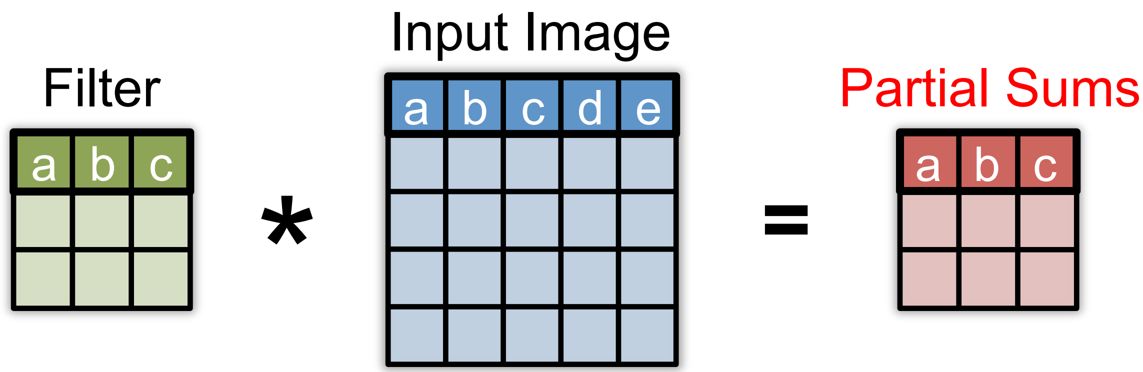
- Use a **large global buffer** as shared storage
  - Reduce **DRAM** access energy consumption
- **Examples:**

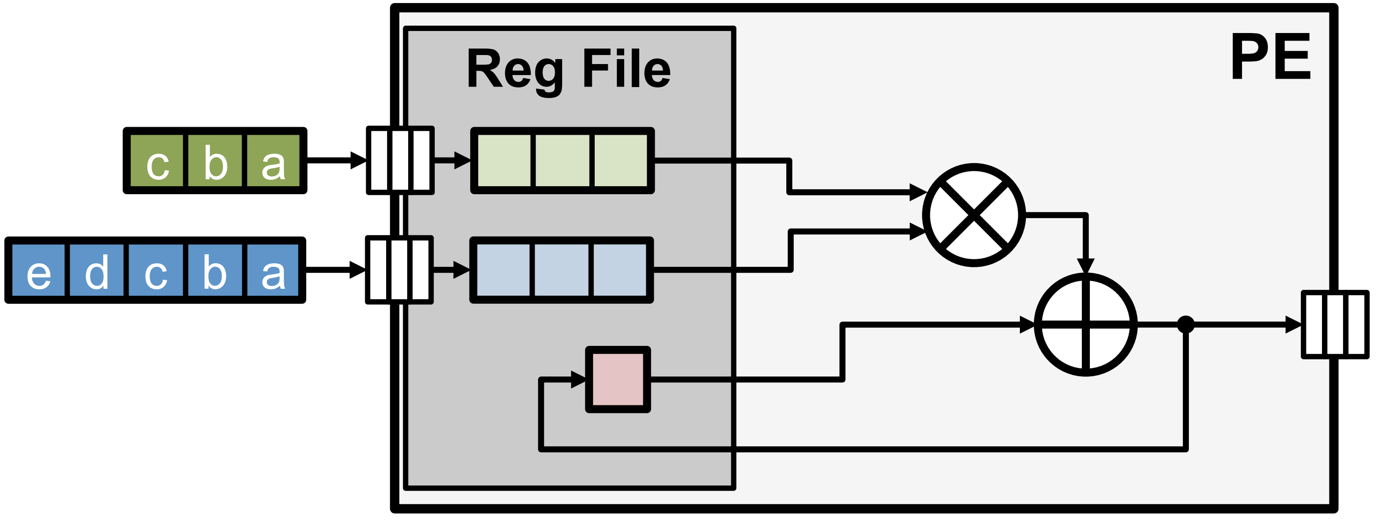
  [**DianNao**, *ASPLOS* 2014] [**DaDianNao**, *MICRO* 2014]
  [**Zhang**, *FPGA* 2015]

Row Stationary: Energy-efficient Dataflow

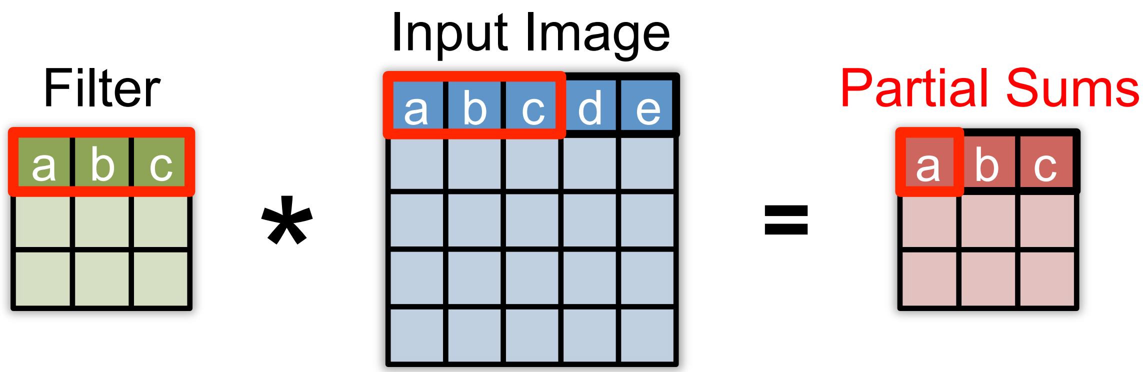


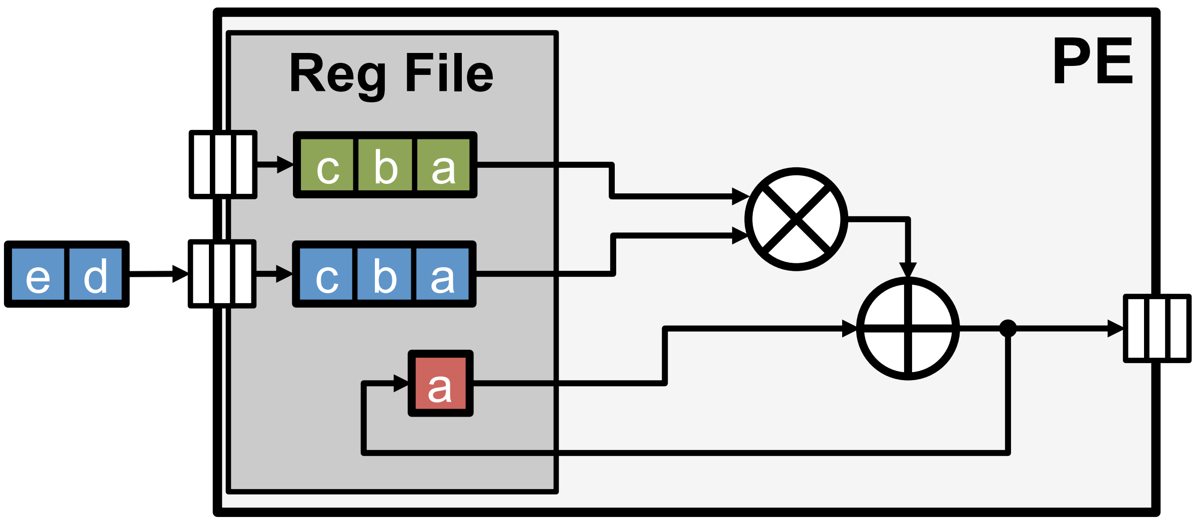
1D Row Convolution in PE



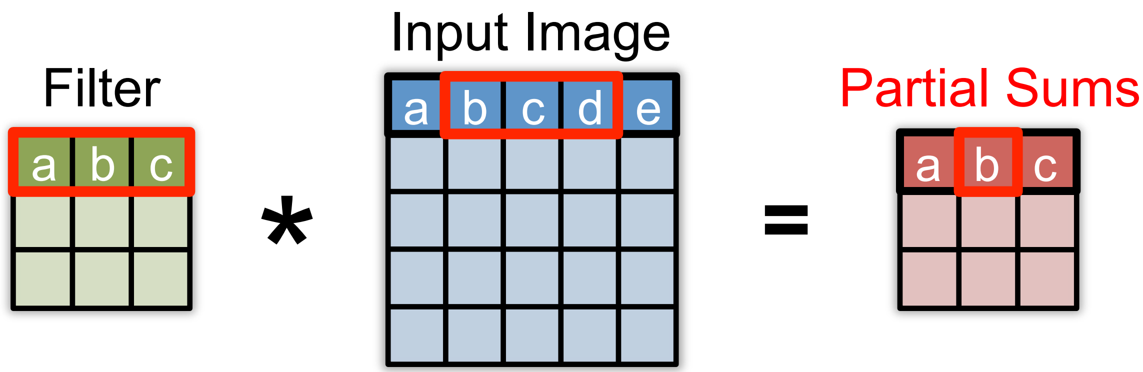


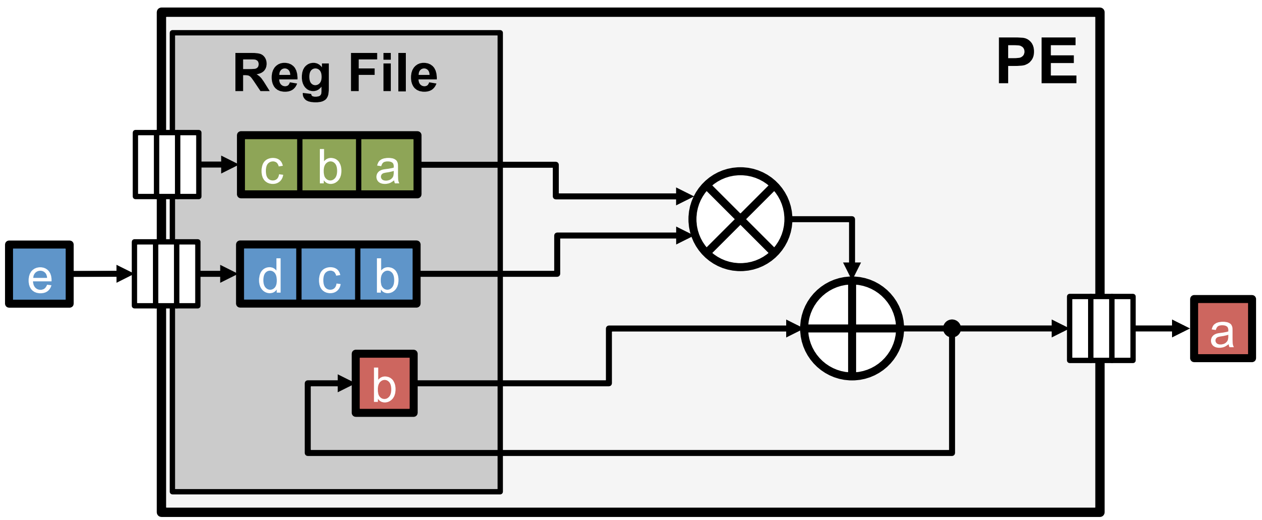
1D Row Convolution in PE



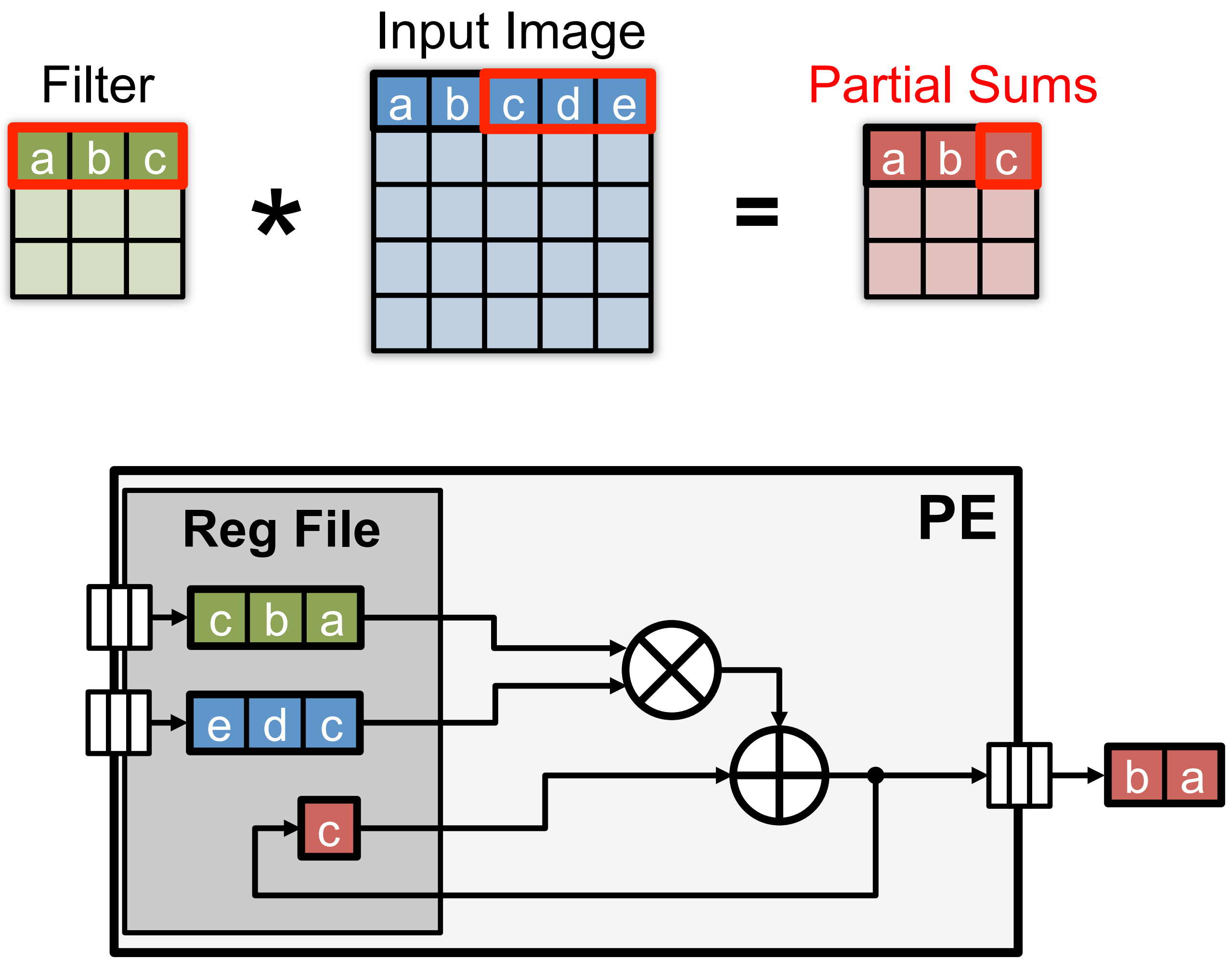


1D Row Convolution in PE



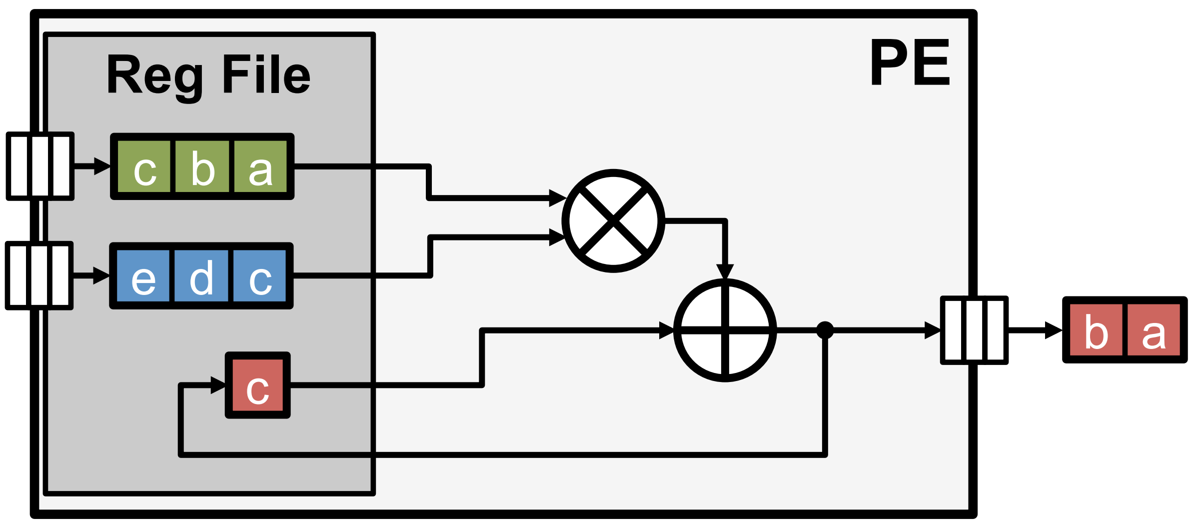


1D Row Convolution in PE

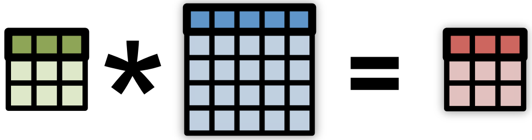


1D Row Convolution in PE

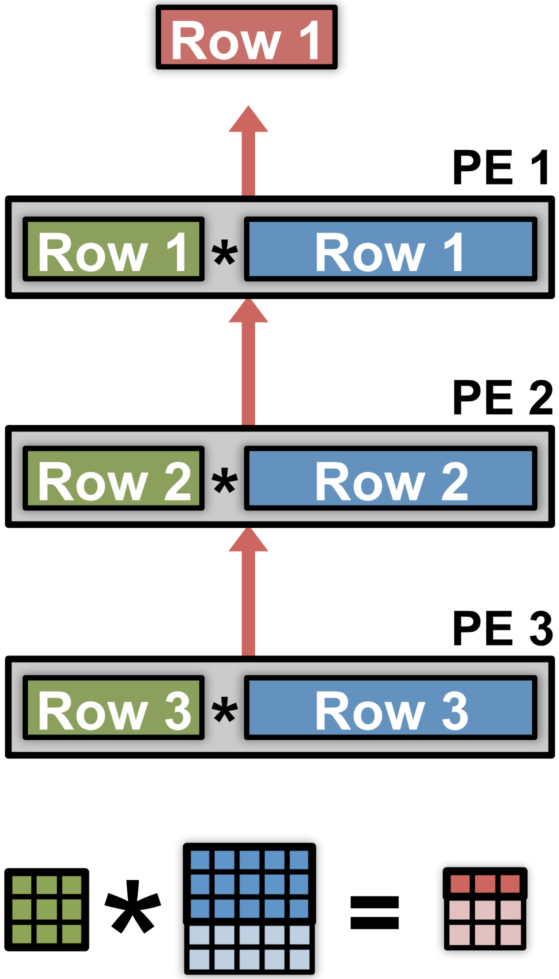
- Maximize row **convolutional reuse** in RF
  - Keep a **filter** row and **image** sliding window in RF
- Maximize row **psum accumulation** in RF



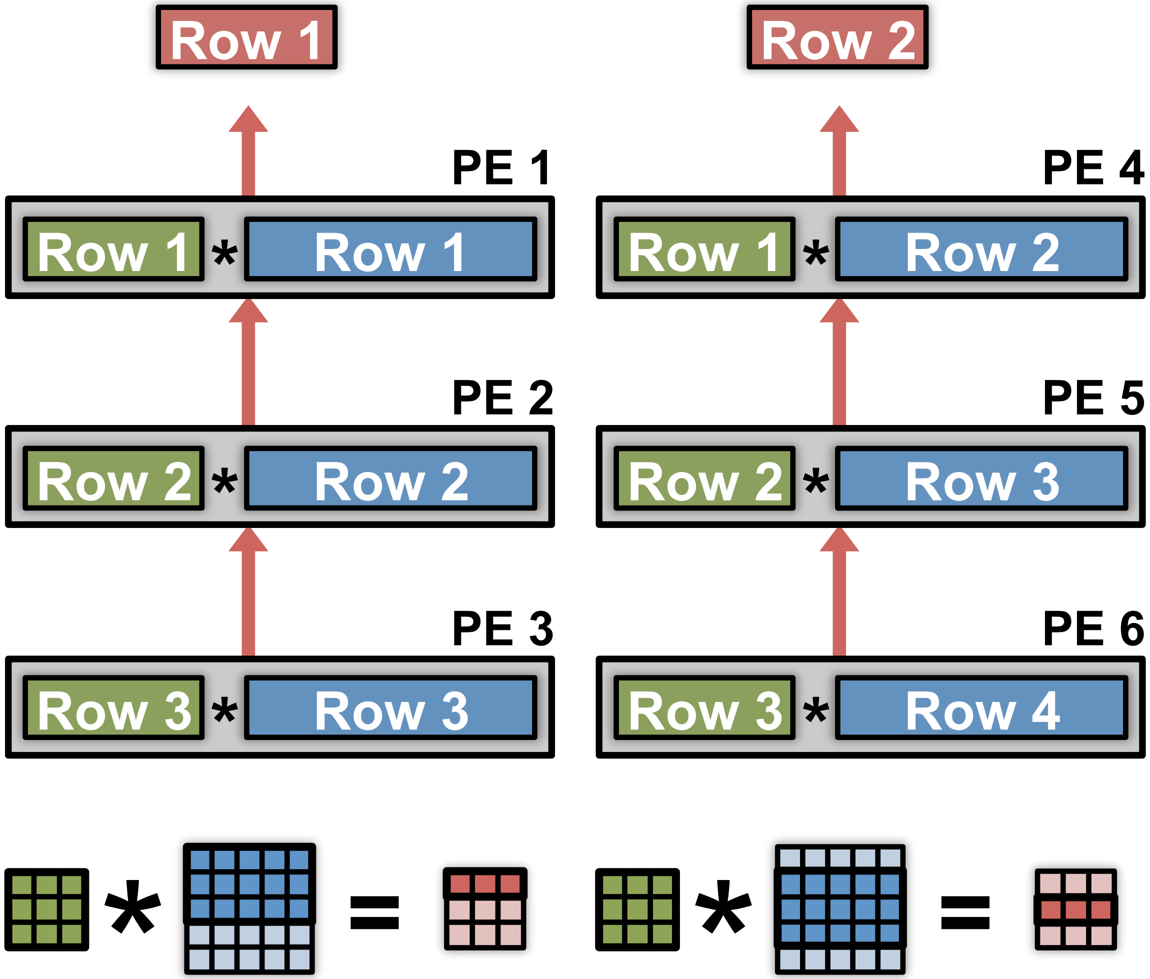
2D Convolution in PE Array



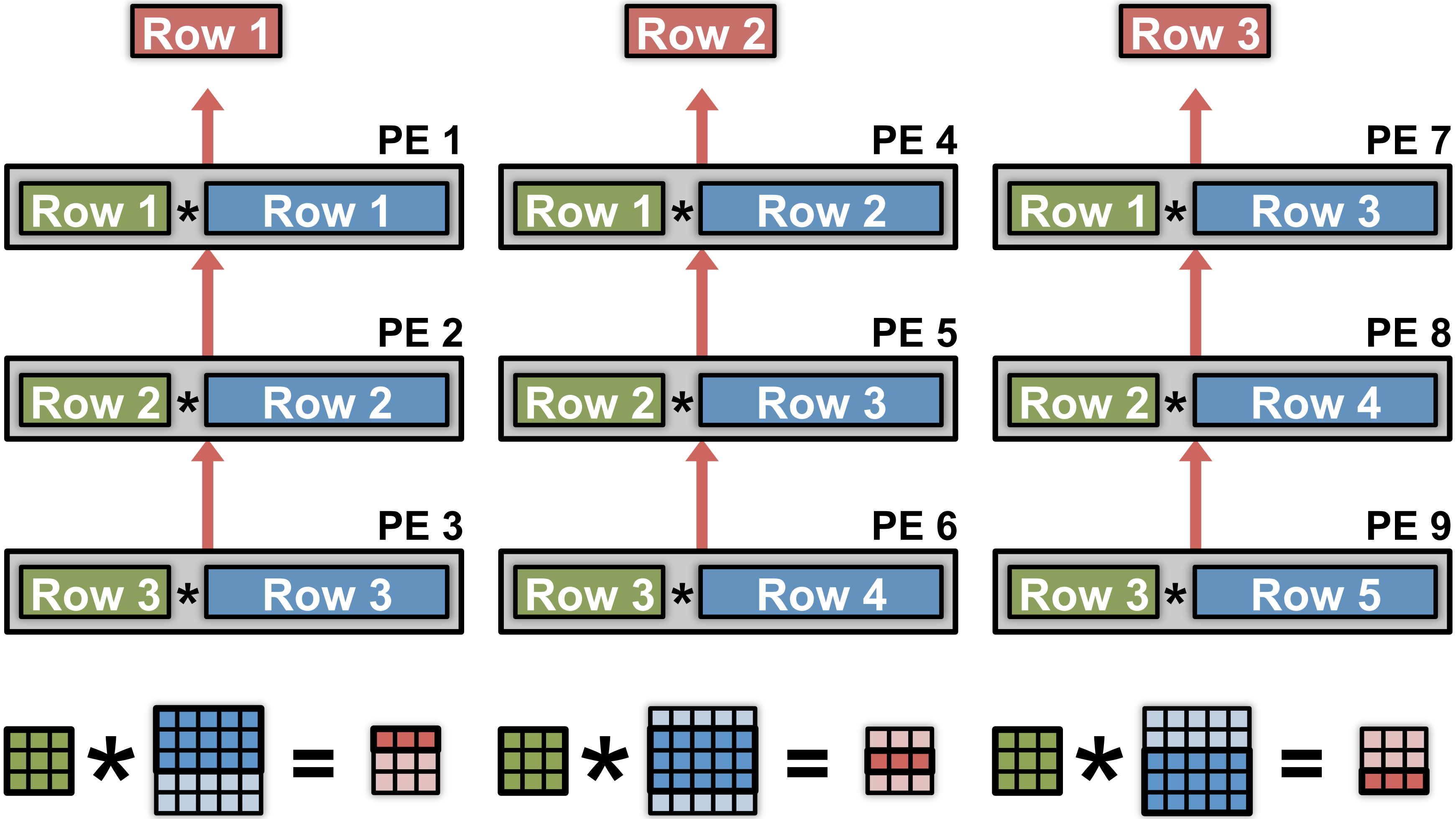
2D Convolution in PE Array



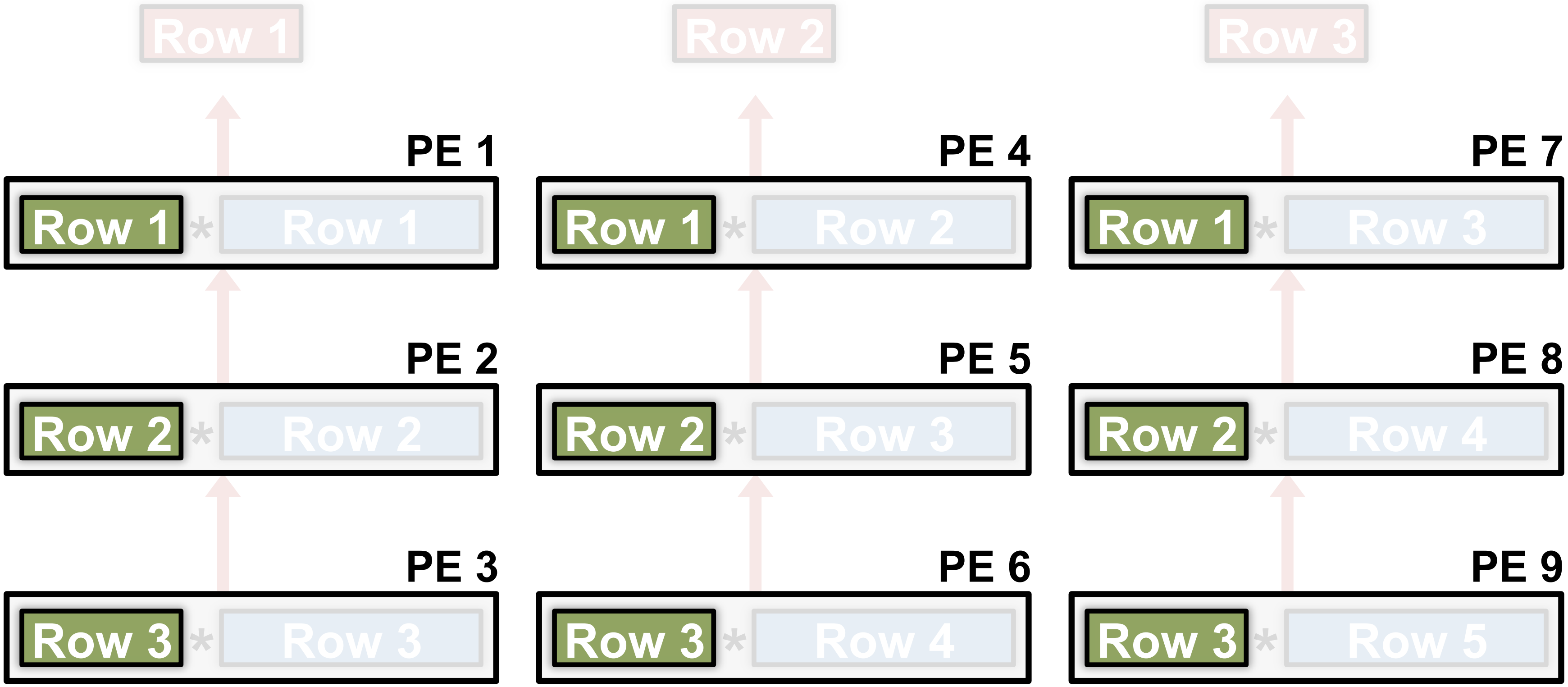
2D Convolution in PE Array



2D Convolution in PE Array



Convolutional Reuse Maximized



Filter rows are reused across PEs **horizontally**

Convolutional Reuse Maximized

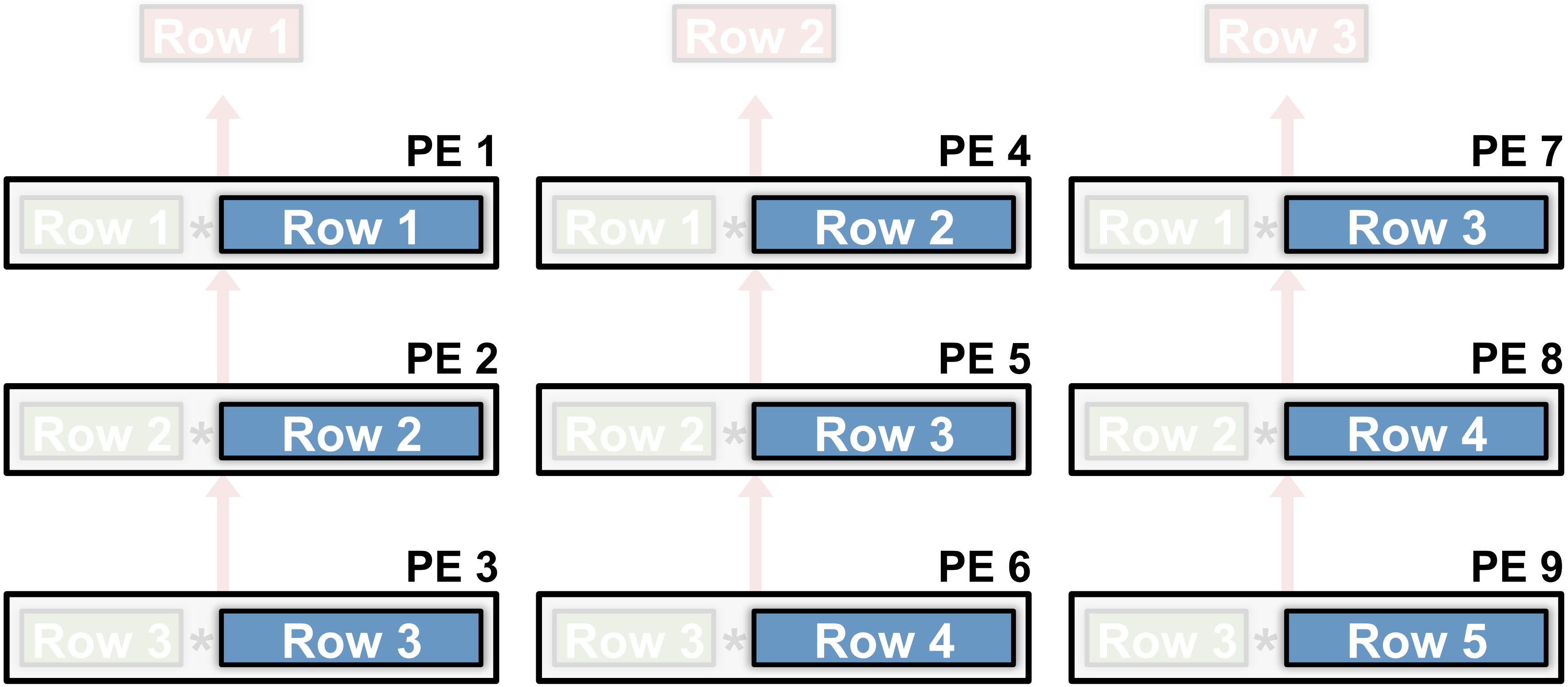
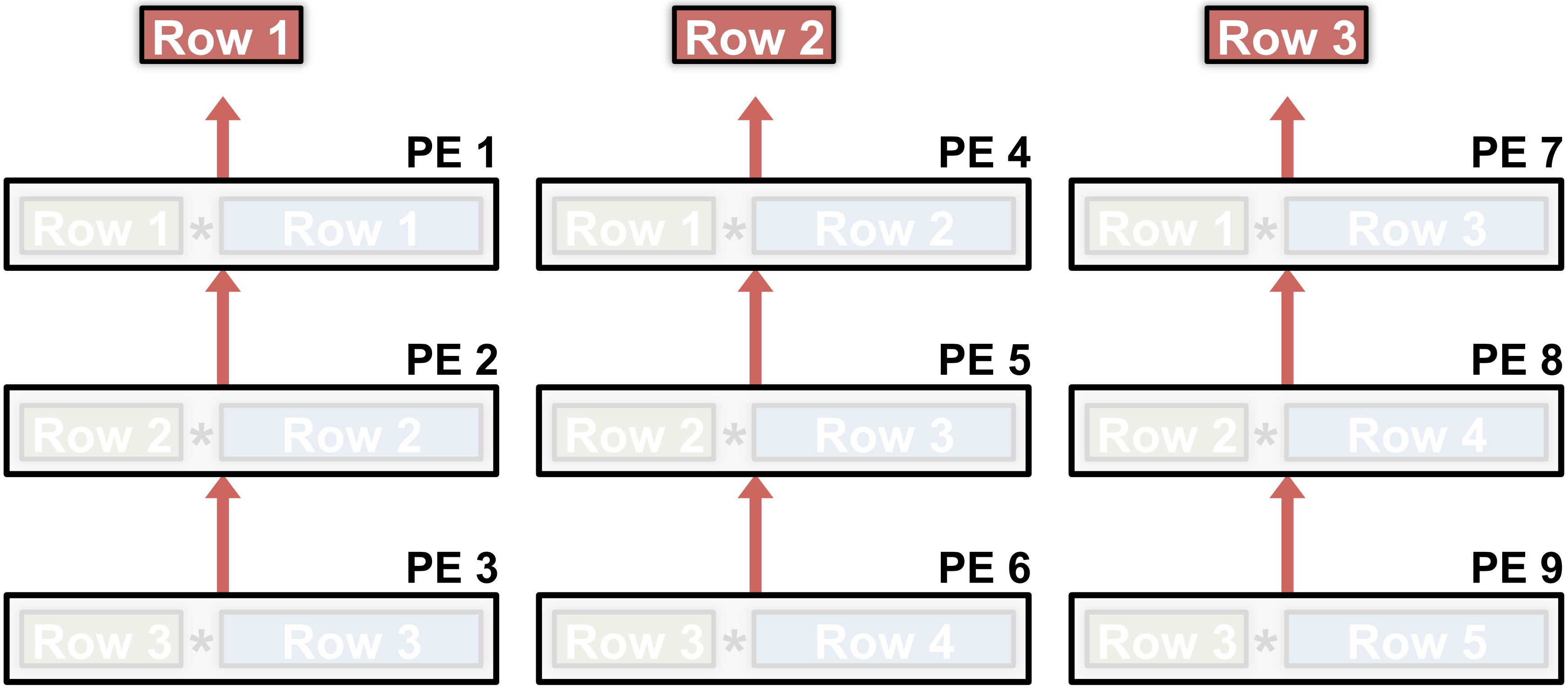


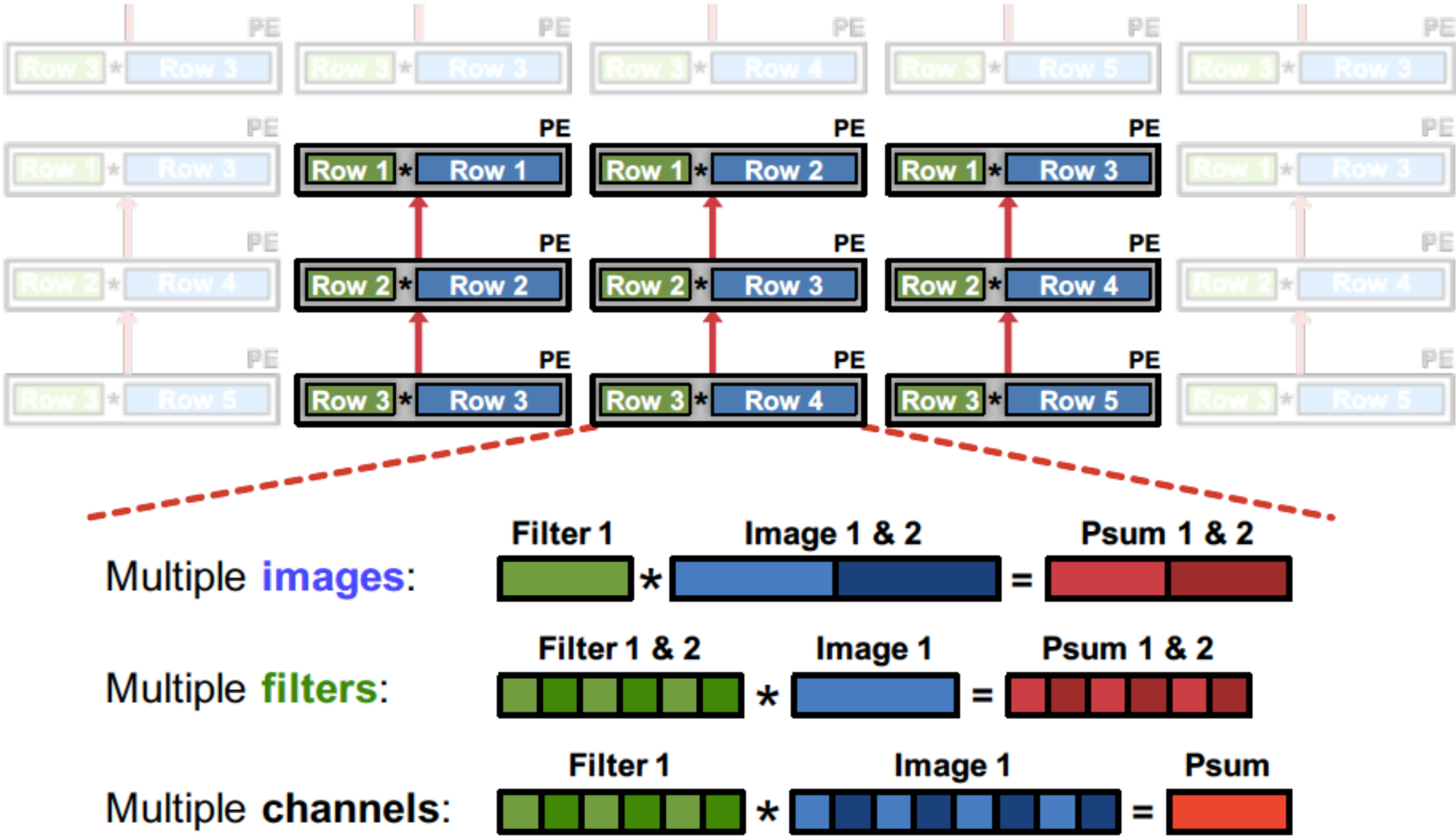
Image rows are reused across PEs **diagonally**

Maximize 2D Accumulation in PE Array



Partial sums accumulate across PEs **vertically**

CNN Convolution – The Full Picture



Map rows from **multiple** images, filters and **channels** to same PE to exploit other forms of reuse and local accumulation

Evaluate Reuse in Different Dataflows

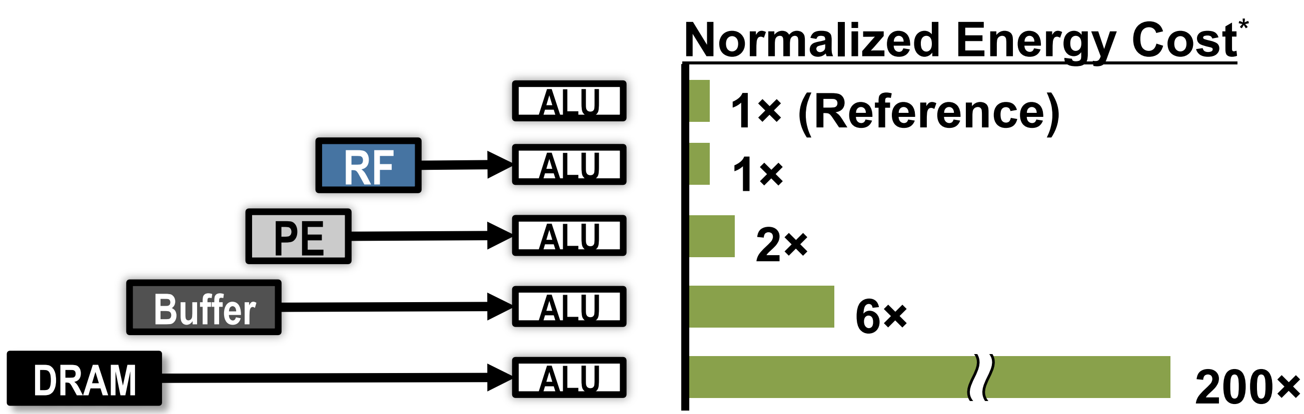
- **Weight Stationary**
  - Minimize movement of filter weights
- **Output Stationary**
  - Minimize movement of partial sums
- **No Local Reuse**
  - Don't use any local PE storage. Maximize global buffer size.
- **Row Stationary**

Evaluate Reuse in Different Dataflows

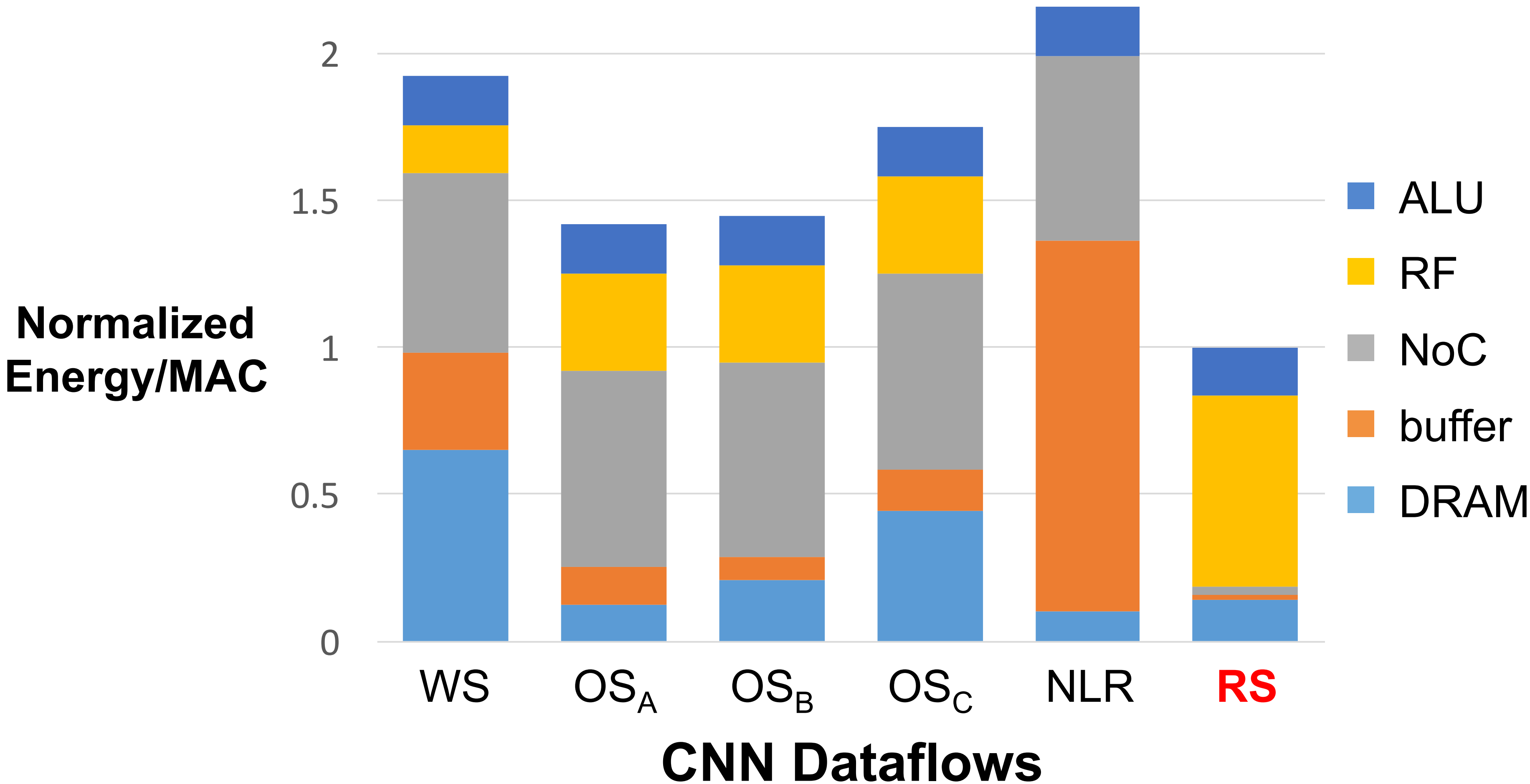
- **Weight Stationary**
  - Minimize movement of filter weights
- **Output Stationary**
  - Minimize movement of partial sums
- **No Local Reuse**
  - Don't use any local PE storage. Maximize global buffer size.
- **Row Stationary**

Evaluation Setup

- Same Total Area
- AlexNet
- 256 PEs
- Batch size = 16

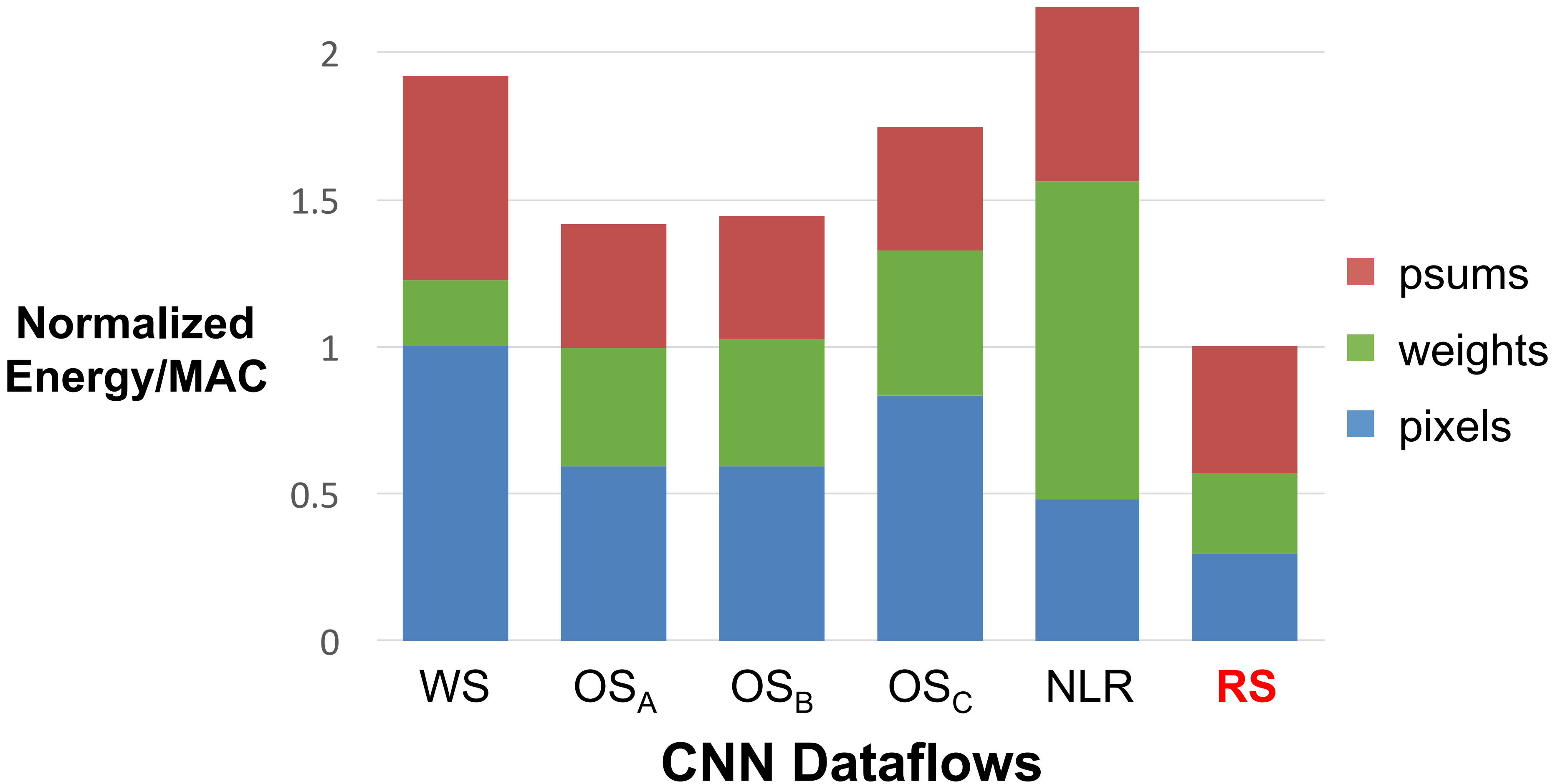


Dataflow Comparison: CONV Layers



RS uses **1.4× – 2.5× lower** energy than other dataflows

Dataflow Comparison: CONV Layers



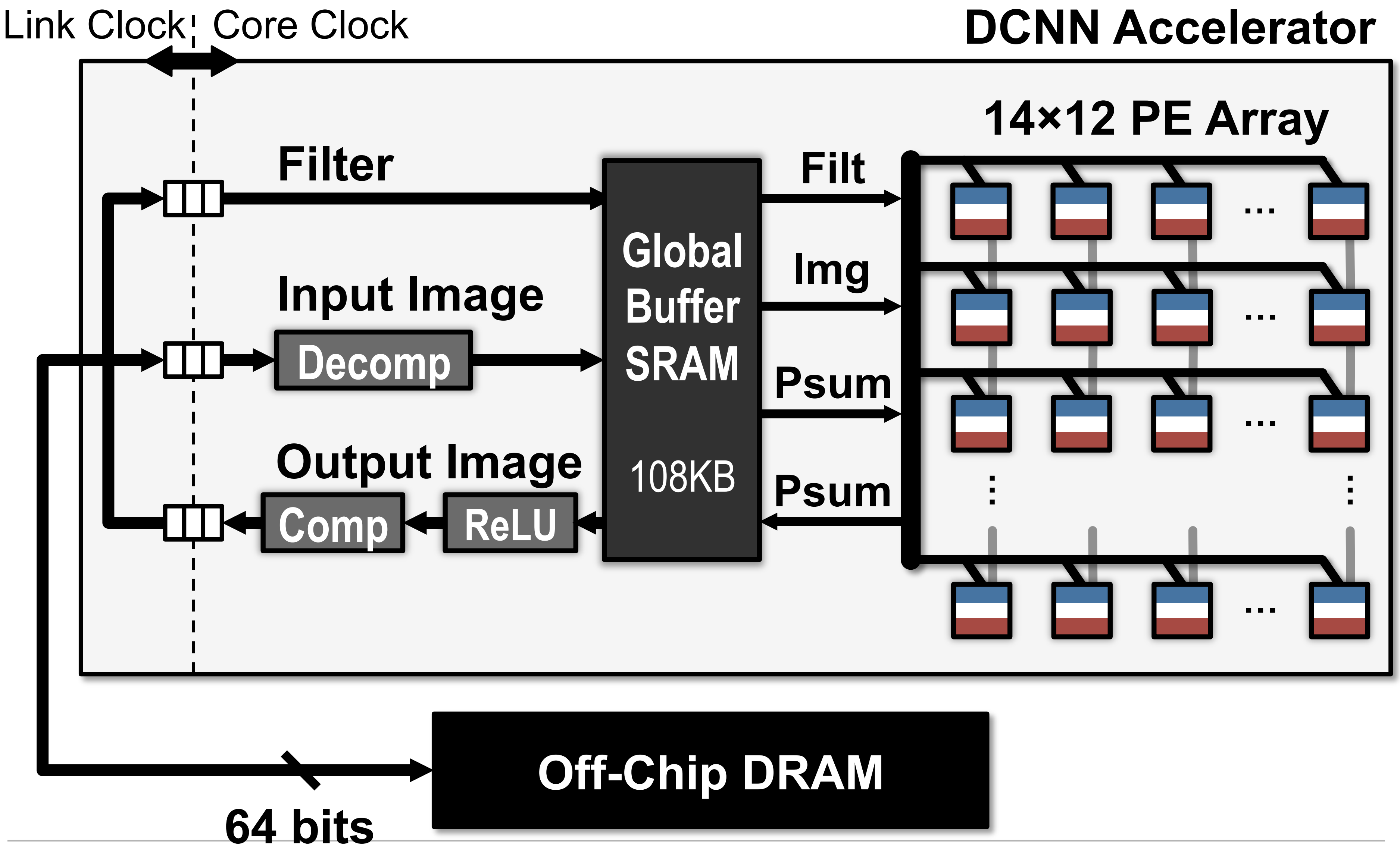
RS optimizes for the best **overall** energy efficiency

Energy-Efficient Accelerator

Yu-Hsin Chen, Tushar Krishna, Joel Emer, Vivienne Sze, ISSCC 2016 [paper]

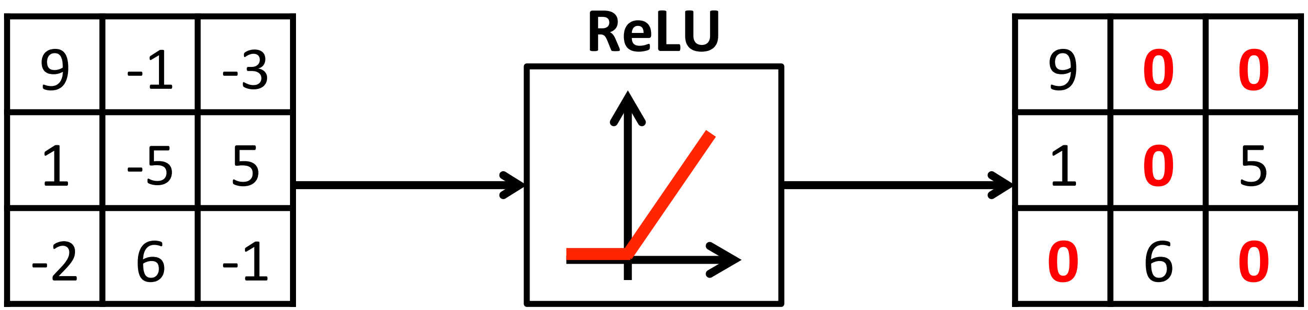
Exploit data statistics

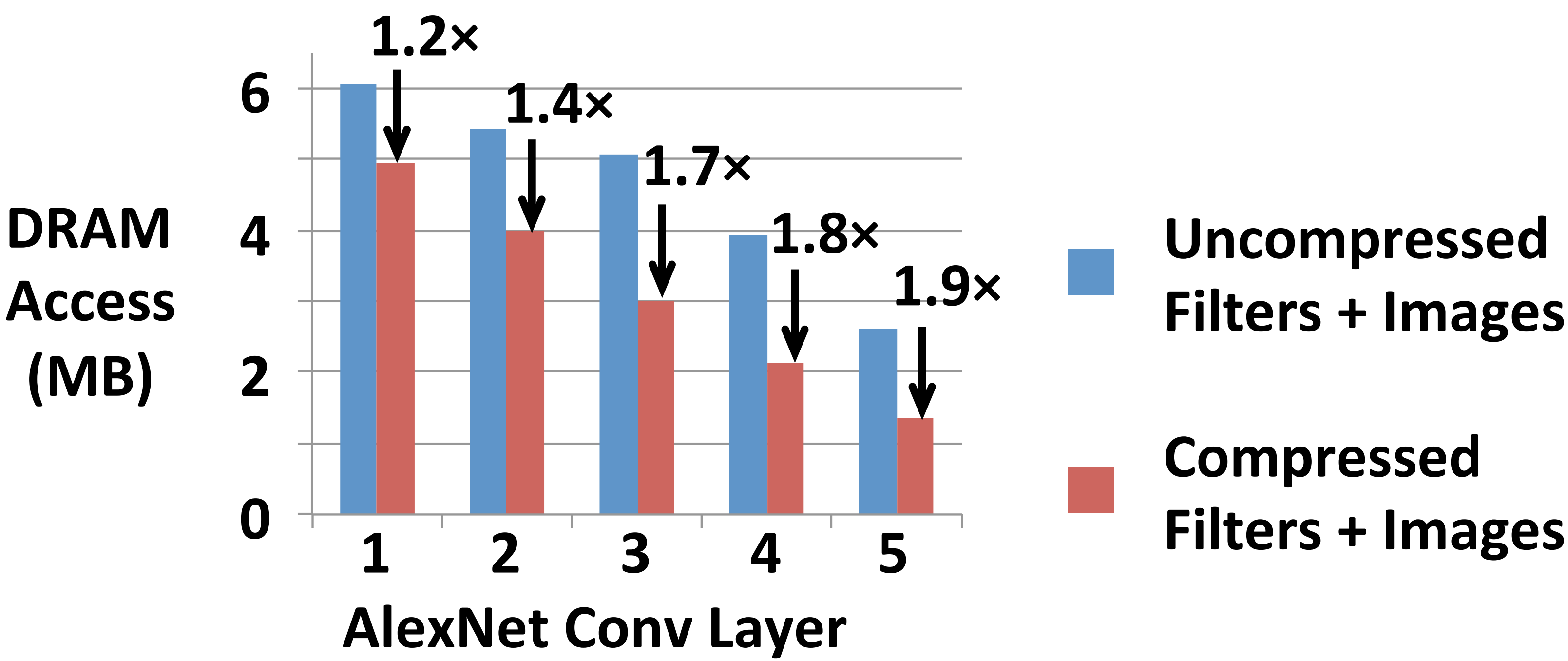
Eyeriss Deep CNN Accelerator



Data Compression Saves DRAM BW

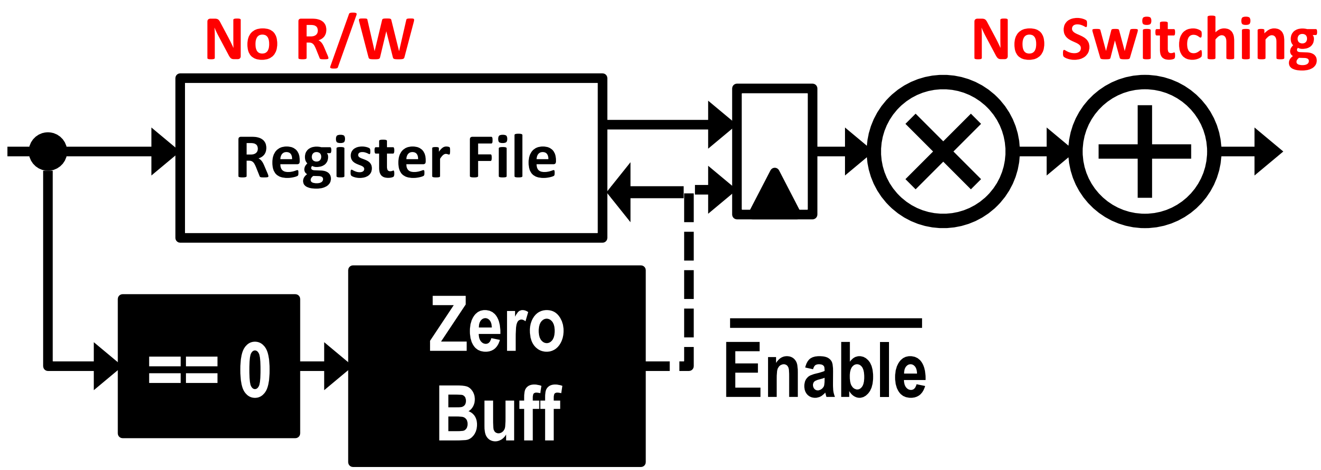
Apply Non-Linearity (**ReLU**) on Filtered Image Data





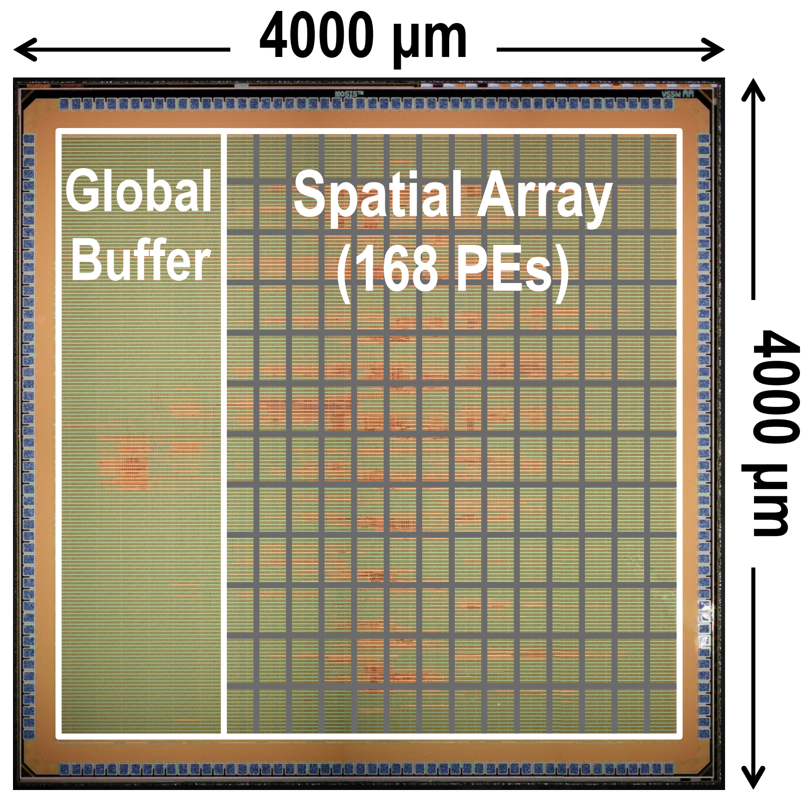
Zero Data Processing Gating

- Skip PE local **memory access**
- Skip MAC **computation**
- Save PE processing power by 45%



Chip Spec & Measurement Results[1]

Technology	TSMC 65nm LP 1P9M
On-Chip Buffer	108 KB
# of PEs	168
Scratch Pad / PE	0.5 KB
Core Frequency	100 – 250 MHz
Peak Performance	33.6 – 84.0 GOPS
Word Bit-width	16-bit Fixed-Point
Natively Supported CNN Shapes	Filter Width: 1 – 32 Filter Height: 1 – 12 Num. Filters: 1 – 1024 Num. Channels: 1 – 1024 Horz. Stride: 1–12 Vert. Stride: 1, 2, 4



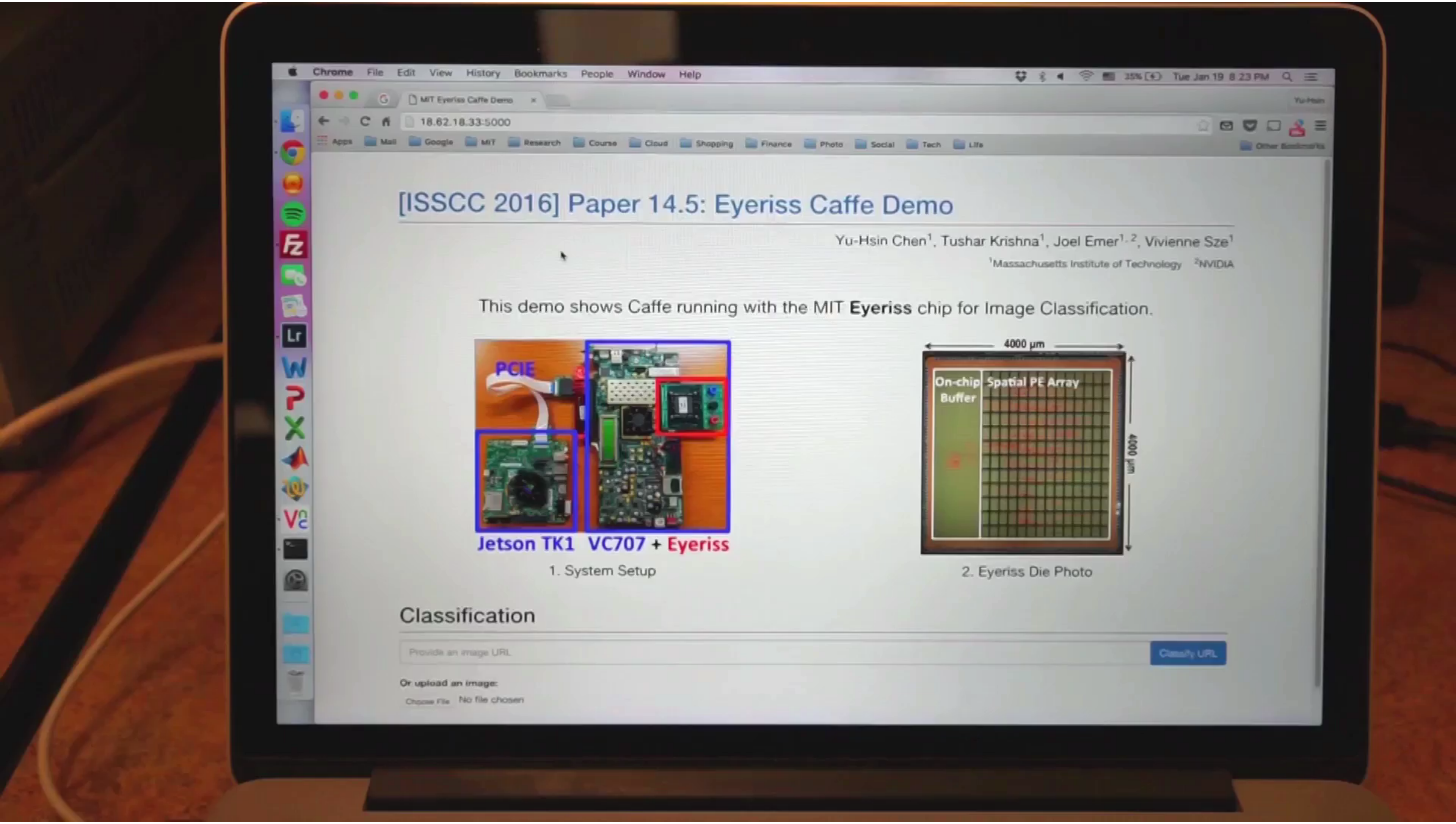
AlexNet: For 2.66 GMACs [8 billion 16-bit inputs (**16GB**) and 2.7 billion outputs (**5.4GB**)], only requires **208.5MB** (buffer) and **15.4MB** (DRAM)

Comparison with GPU

	This Work	NVIDIA TK1 (Jetson Kit)
Technology	65nm	28nm
Clock Rate	200MHz	852MHz
# Multipliers	168	192
On-Chip Storage	Buffer: 108KB Spad: 75.3KB	Shared Mem: 64KB Reg File: 256KB
Word Bit-Width	16b Fixed	32b Float
Throughput[1]	34.7 fps	68 fps
Measured Power	278 mW	Idle/Active[2]: 3.7W/10.2W
DRAM Bandwidth	127 MB/s	1120 MB/s [3]

1. AlexNet Convolutional Layers Only
2. Board Power
3. Modeled from [Tan, SC11]

Demo of Image Classification on Eyeriss



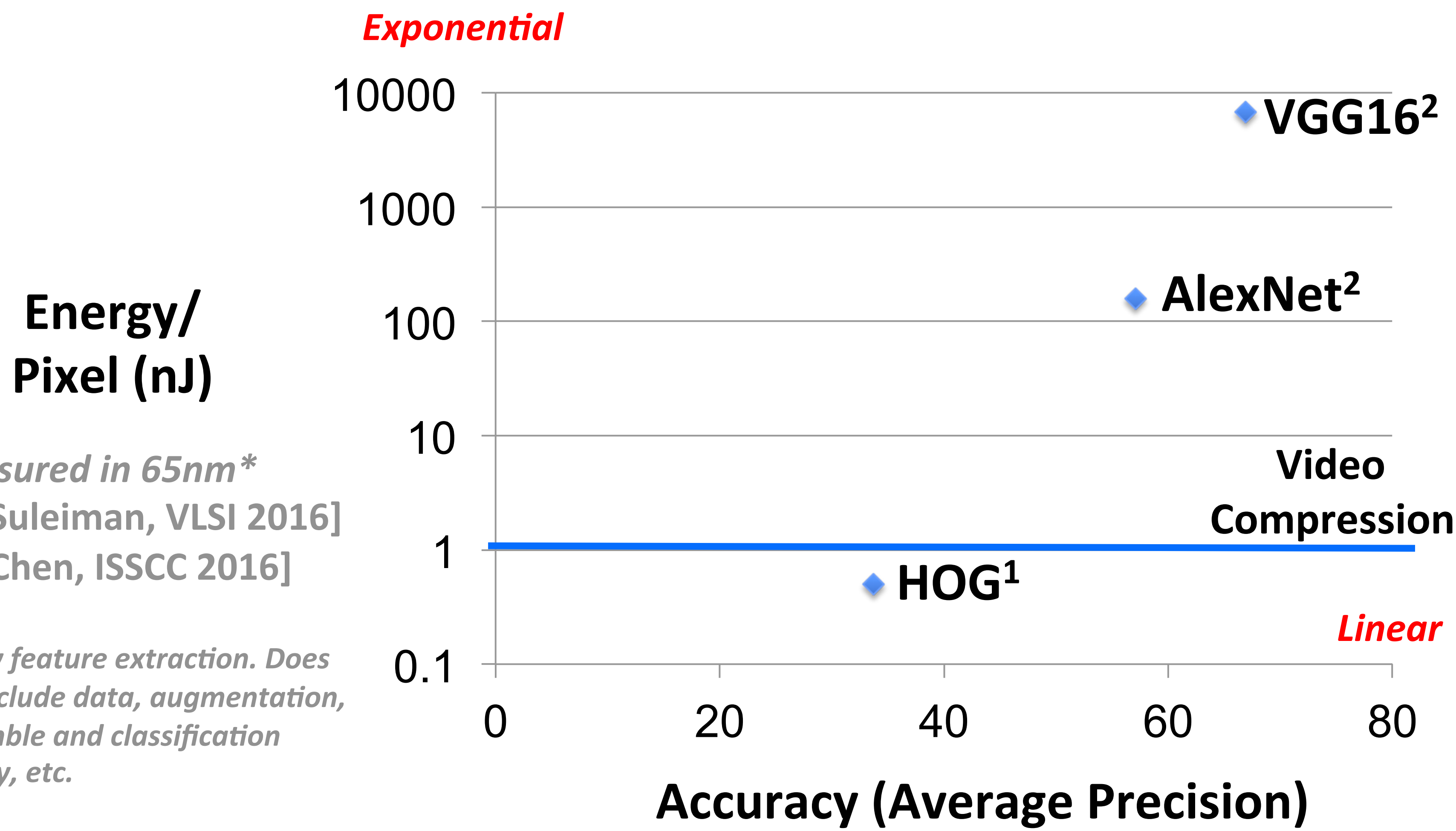
https://vimeo.com/154012013

Integrated with BVLC Caffe DL Framework

Summary of Eyeriss Deep CNN

- **Eyeriss:** a **reconfigurable** accelerator for state-of-the-art deep CNNs **at below 300mW**
- Energy-efficient **dataflow to reduce data movement**
- **Exploit data statistics** for high energy efficiency
- **Integrated** with the **Caffe DL framework** and demonstrated an image classification system

Features: Energy vs. Accuracy



*Measured in 65nm**

1. [Suleiman, VLSI 2016]
2. [Chen, ISSCC 2016]

** Only feature extraction. Does not include data, augmentation, ensemble and classification energy, etc.*

Measured in on VOC 2007 Dataset

1. DPM v5 [Girshick, 2012]
2. Fast R-CNN [Girshick, CVPR 2015]

Acknowledgements

This work is funded by the DARPA YFA grant, MIT Center for Integrated Circuits & Systems, and a gift from Intel.

More info about **Eyeriss** and **Tutorial on DNN Architectures** at

http://eyeriss.mit.edu

More info about research in the **Energy-Efficient Multimedia Systems Group @ MIT**

http://www.rle.mit.edu/eems

Follow @eems_mit

for updates