Co-Design Approaches for Efficient Deep Neural Networks: Challenges and Opportunities

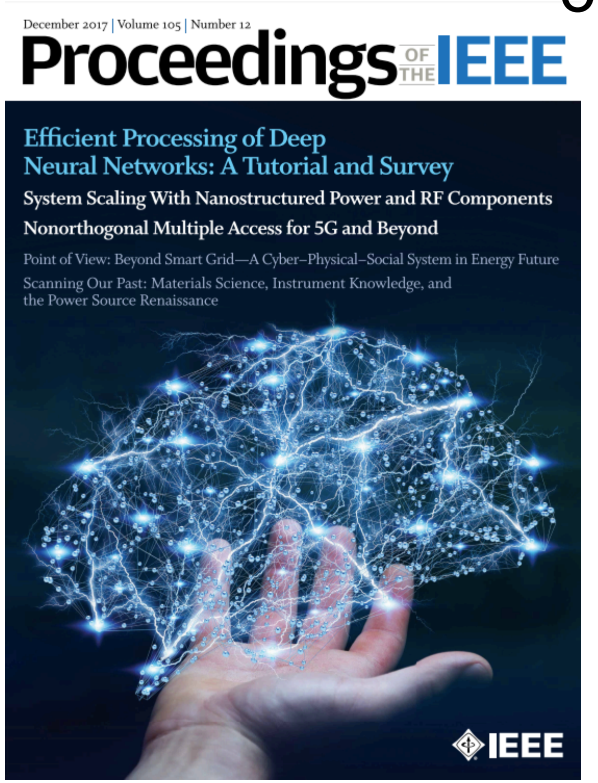
Vivienne Sze (@eems_mit)
Massachusetts Institute of Technology

In collaboration with Yu-Hsin Chen, Joel Emer, Sertac Karaman, Fangchang Ma, Diana Wofk, Yannan Wu, Tien-Ju Yang, Google Mobile Vision Team

Slides available at
https://tinyurl.com/SzeNeurIPS2019

Energy-Efficient Processing of DNNs

A significant amount of algorithm and hardware research on energy-efficient processing of DNNs



December 2017 | Volume 105 | Number 12

Proceedings OF THE IEEE

Efficient Processing of Deep Neural Networks: A Tutorial and Survey

System Scaling With Nanostructured Power and RF Components

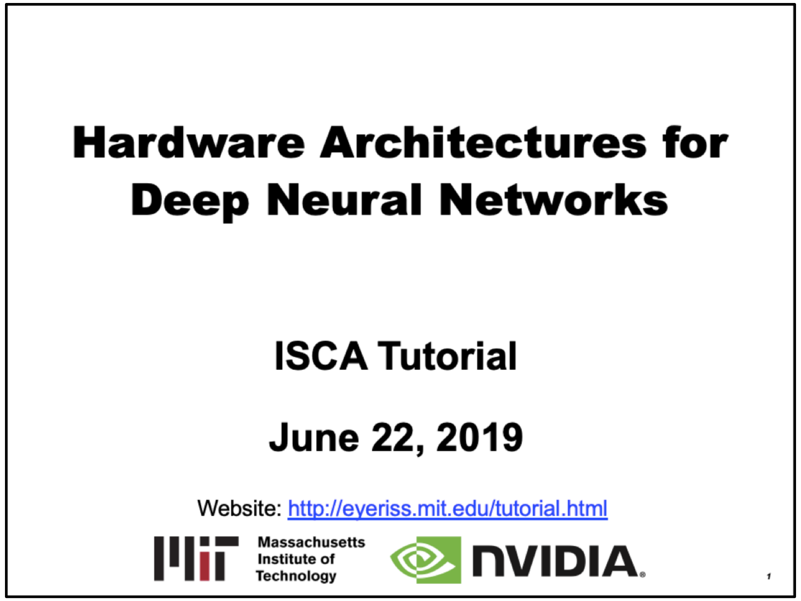
Nonorthogonal Multiple Access for 5G and Beyond

Point of View: Beyond Smart Grid—A Cyber–Physical–Social System in Energy Future

Scanning Our Past: Materials Science, Instrument Knowledge, and the Power Source Renaissance

IEEE

V. Sze, Y.-H. Chen,
T-J. Yang, J. Emer,
"***Efficient Processing of Deep Neural Networks: A Tutorial and Survey***,"
Proceedings of the IEEE, Dec. 2017



Hardware Architectures for Deep Neural Networks

ISCA Tutorial

June 22, 2019

Website: http://eyeriss.mit.edu/tutorial.html

Massachusetts Institute of Technology

NVIDIA

1

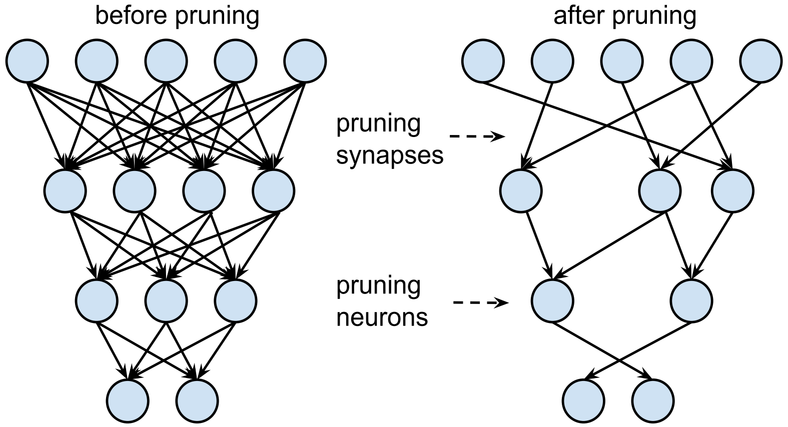
http://eyeriss.mit.edu/tutorial.html

We identified various challenges to existing approaches

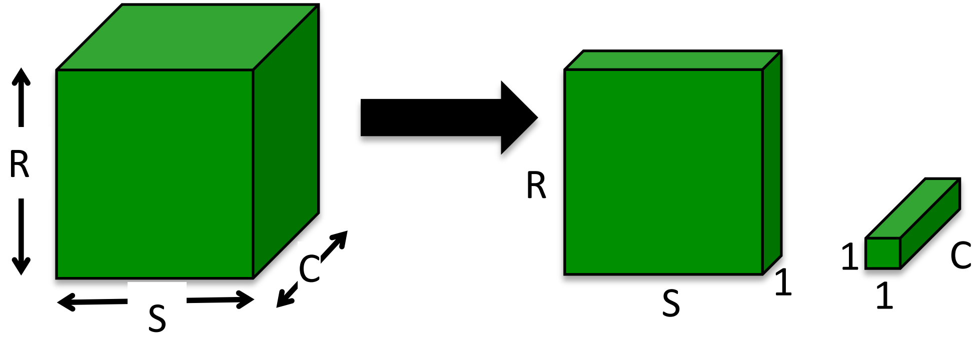
Design of Efficient DNN Algorithms

Popular efficient DNN algorithm approaches

Network Pruning



Efficient Network Architectures

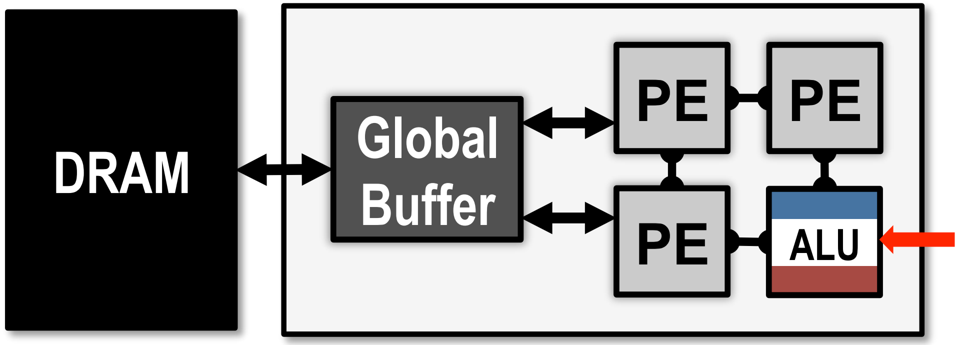


… also reduced precision

□ Focus on reducing number of MACs and weights

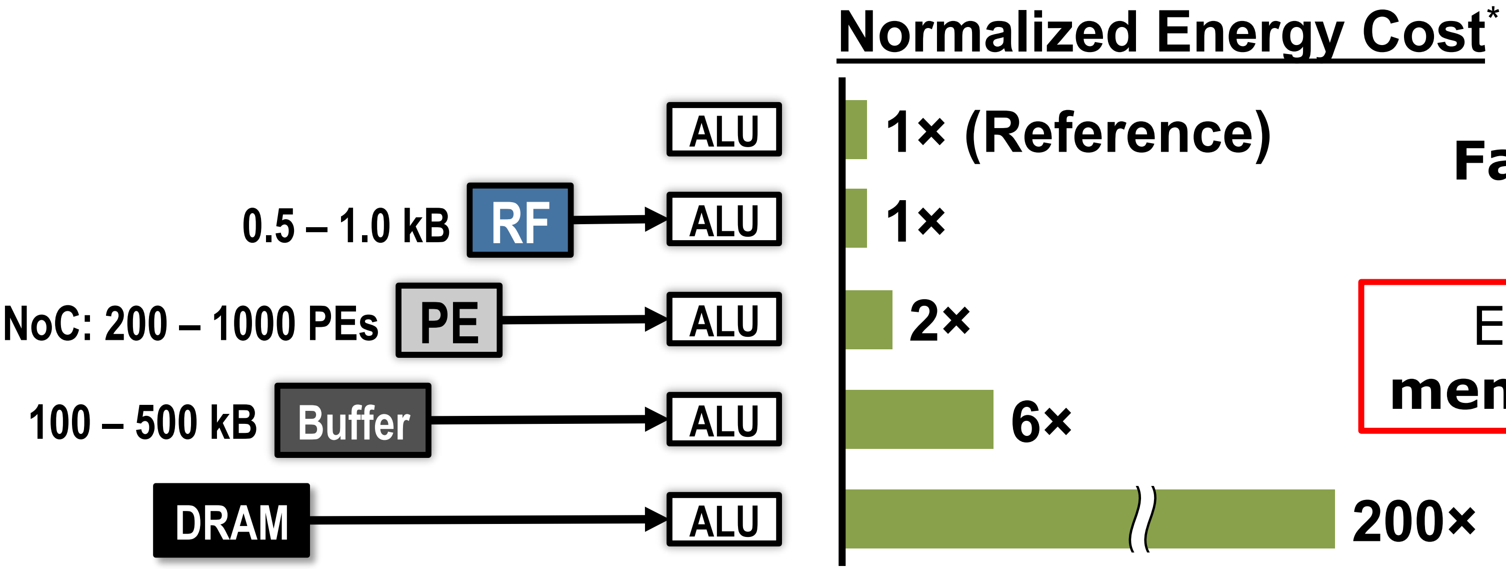
□ **Does it translate to energy savings and reduced latency?**

Data Movement is Expensive



fetch data to run a MAC here

Specialized hardware with small (< 1kB) low cost memory near compute

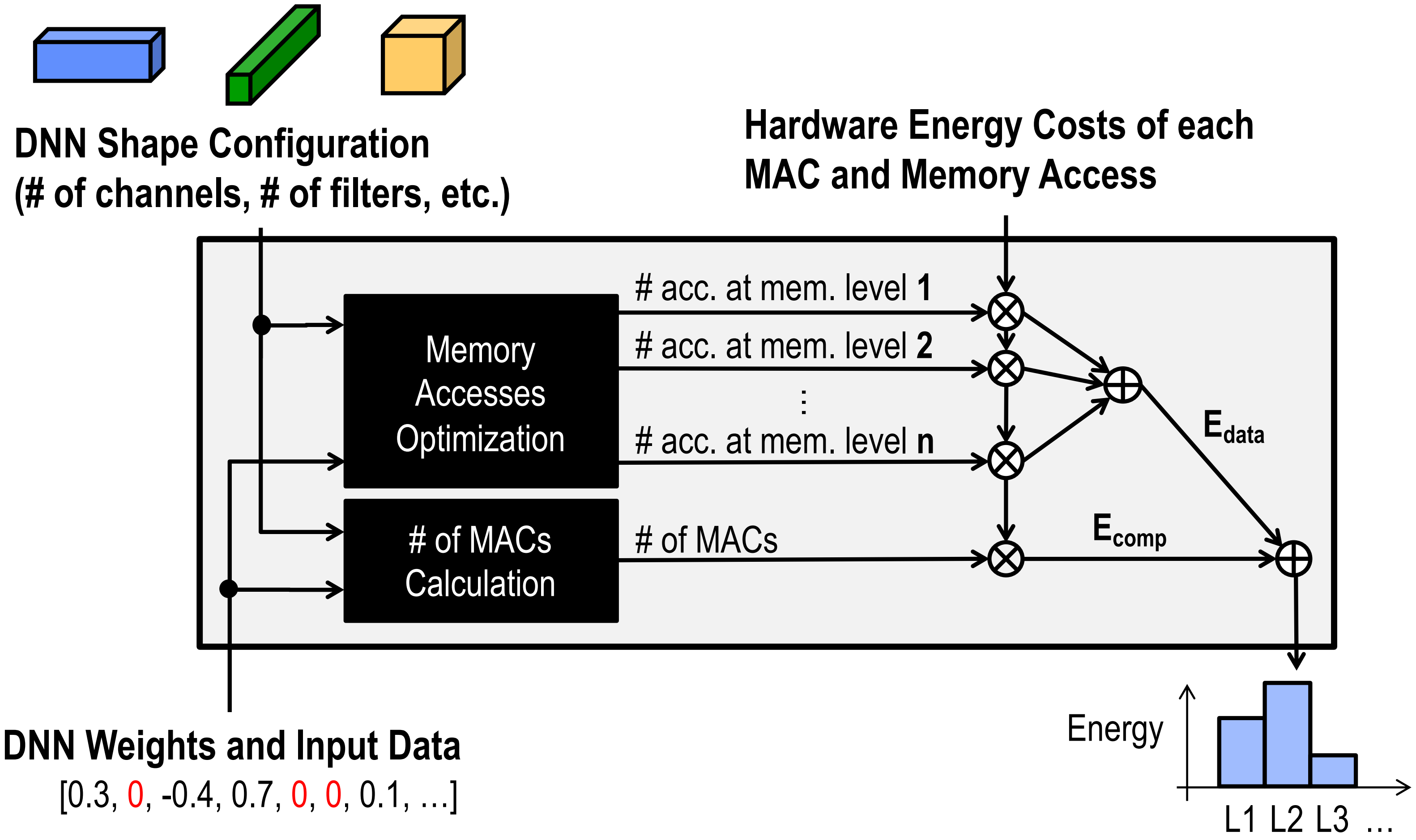


Farther and **larger** memories consume more power

Energy of weight depends on **memory hierarchy** and **dataflow**

* measured from a commercial 65nm process

Energy-Evaluation Methodology



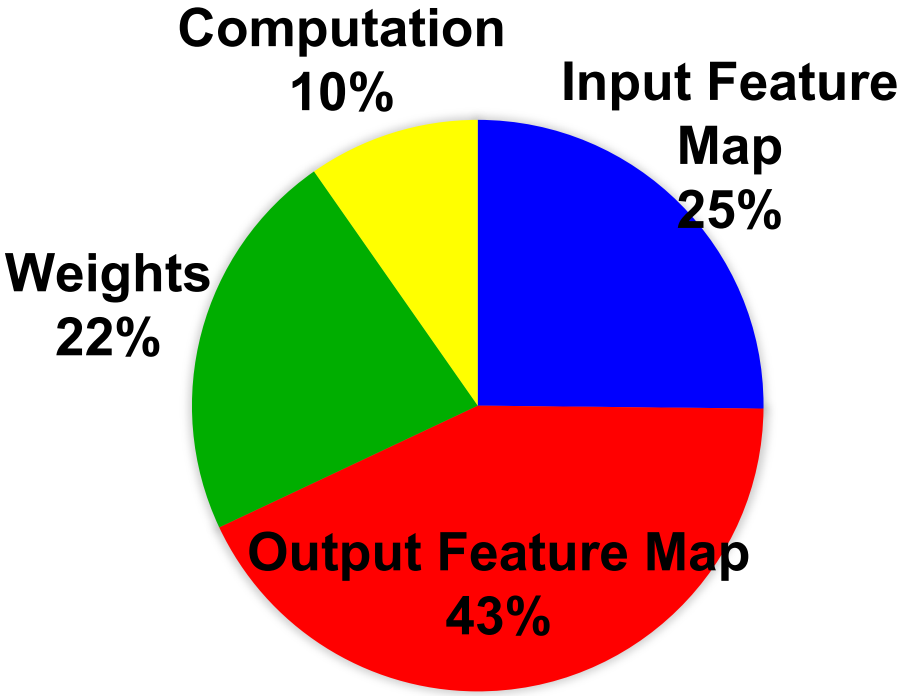
Tool available at https://energyestimation.mit.edu/

[**Yang**, *CVPR* 2017]

Key Observations

- □ Number of weights *alone* is not a good metric for energy
- □ All data types should be considered

Energy Consumption of GoogLeNet



Tool available at https://energyestimation.mit.edu/

[**Yang**, *CVPR* 2017]

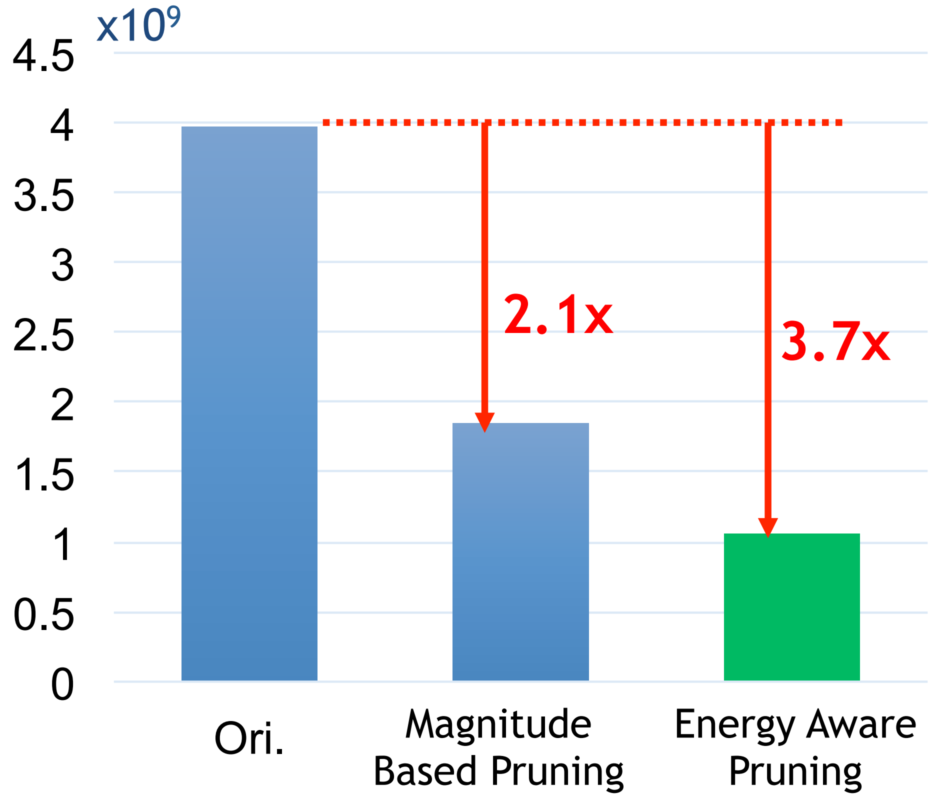
Energy-Aware Pruning

Directly target energy and incorporate it into the optimization of DNNs to provide greater energy savings

- Sort layers based on energy and prune layers that consume the most energy first
- Energy-aware pruning reduces AlexNet energy by **3.7x** and outperforms the previous work that uses magnitude-based pruning by **1.7x**

[**Yang**, *CVPR* 2017]

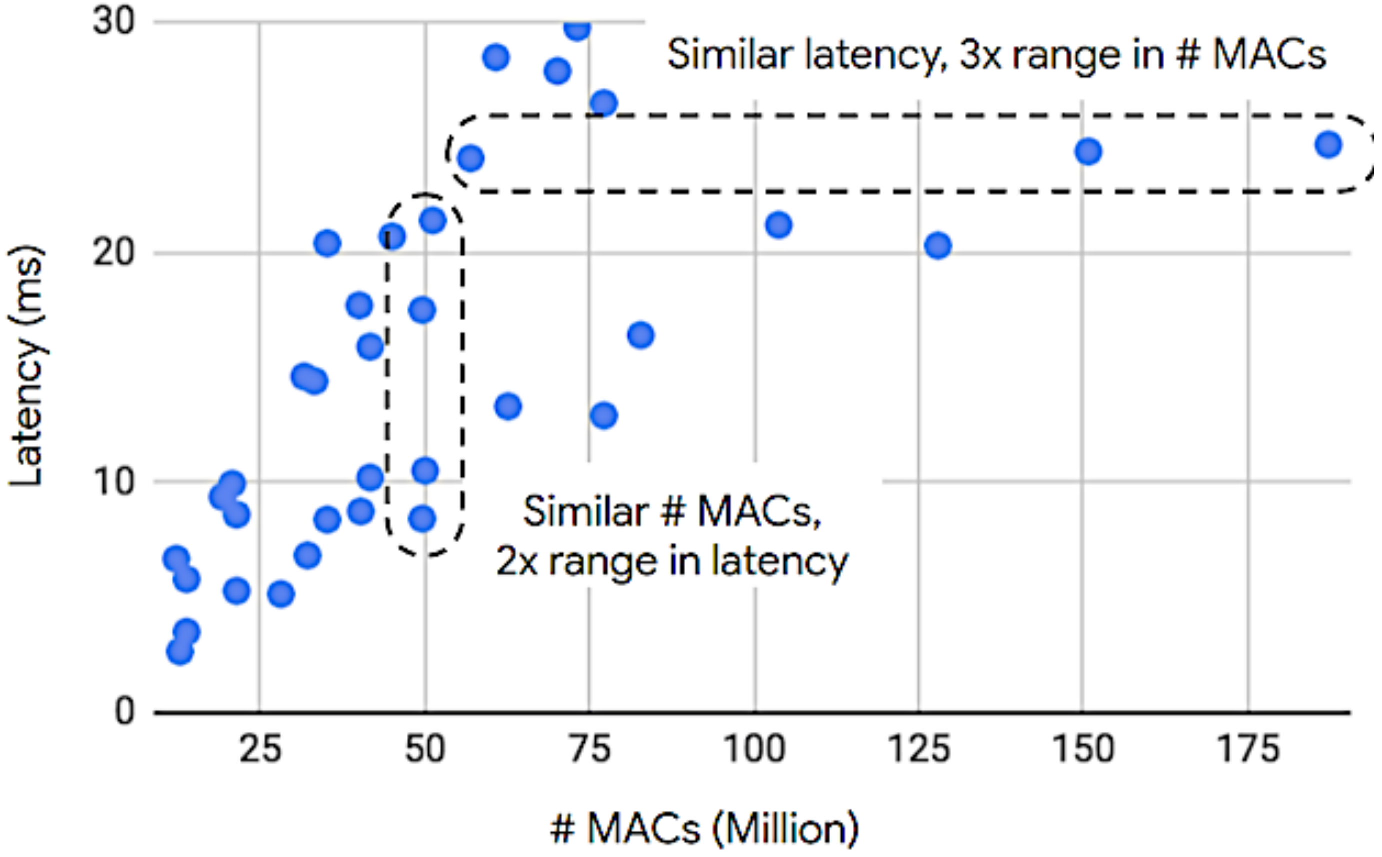
Normalized Energy (AlexNet)



Pruned models available at http://eyeriss.mit.edu/energy.html

# of Operations versus Latency

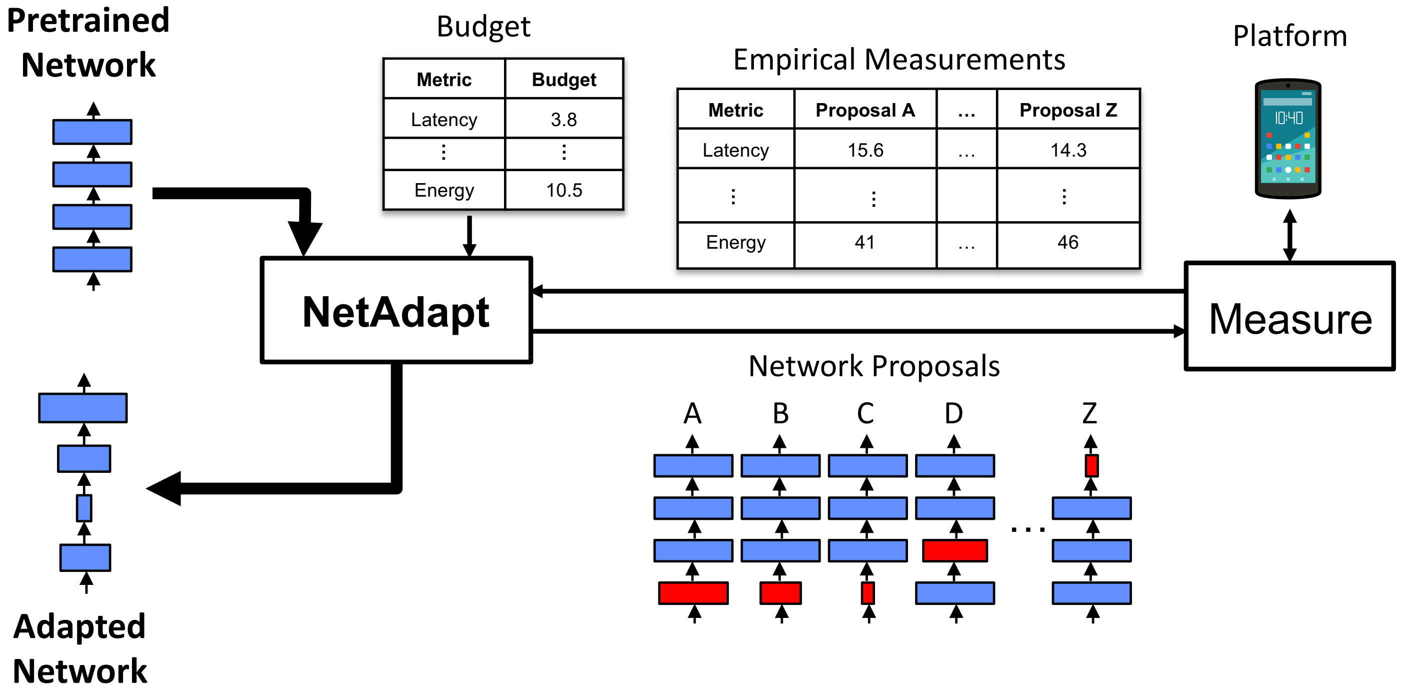
# of operations (MACs) does not approximate latency well



Source: Google (https://ai.googleblog.com/2018/04/introducing-cvpr-2018-on-device-visual.html)

NetAdapt: Platform-Aware DNN Adaptation

- **Automatically adapt DNN** to a mobile platform to reach a target latency or energy budget
- Use **empirical measurements** to guide optimization (avoid modeling of tool chain or platform architecture)
- Requires **very few hyperparameters** to tune



Budget

Metric	Budget
Latency	3.8
⋮	⋮
Energy	10.5

Empirical Measurements

Metric	Proposal A	…	Proposal Z
Latency	15.6	…	14.3
⋮	⋮		⋮
Energy	41	…	46

In collaboration with Google's Mobile Vision Team

Code available at http://netadapt.mit.edu

[**Yang**, *ECCV* 2018]

NetAdapt: Problem Formulation

$$\boxed{\max_{Net} Acc(Net) \;\; subject\ to\ Res_j(Net) \leq Bud_j, j = 1, \cdots, m}$$

↓ Break into a set of simpler problems and solve iteratively

$$\boxed{\max_{Net_i} Acc(Net_i) \;\; subject\ to\ Res_j(Net_i) \leq Res_j(Net_{i-1}) - \Delta R_{i,j}, j = 1, \cdots, m}$$

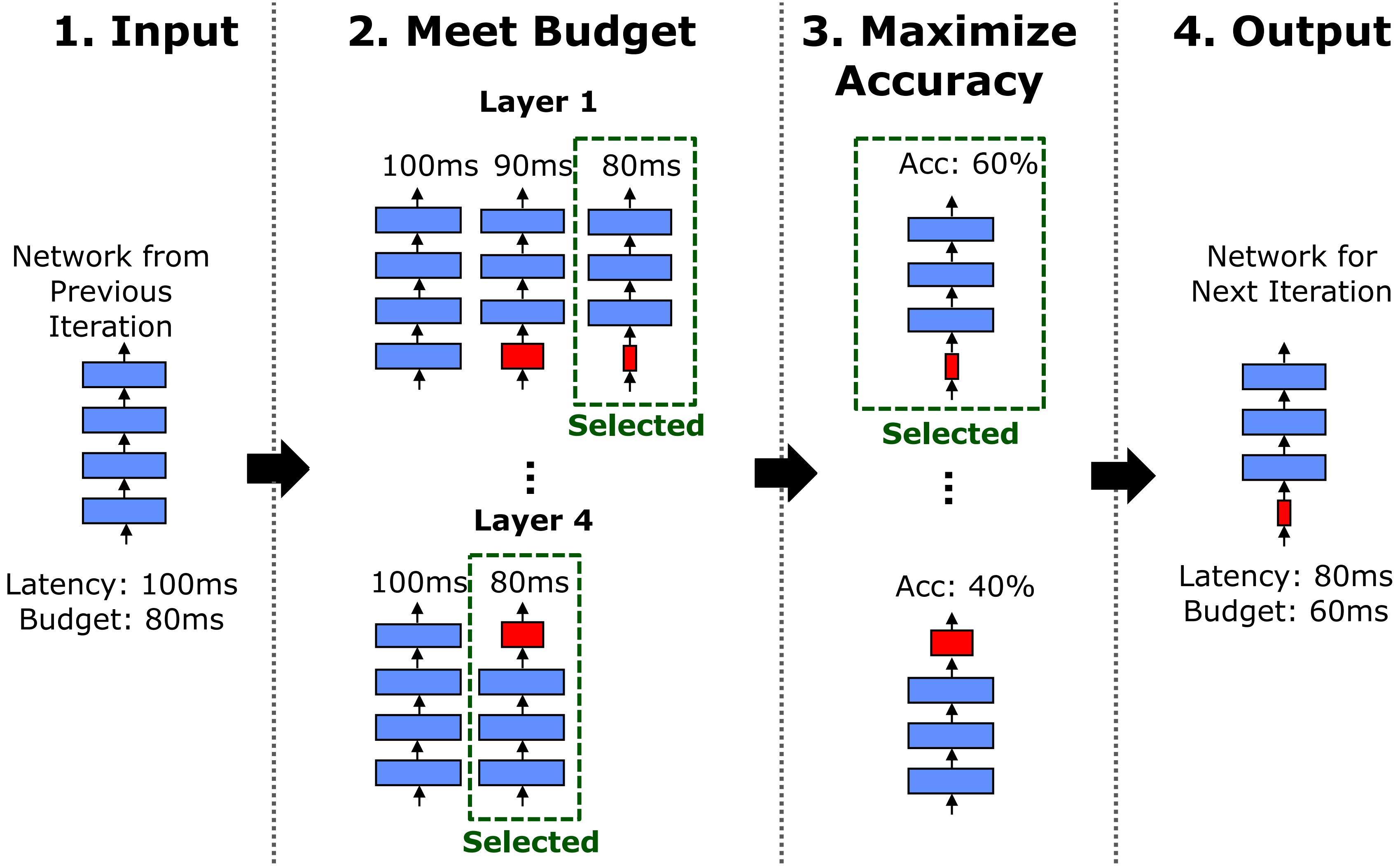
**Acc*: accuracy function, *Res*: resource evaluation function, *Bud*: given budget

***ΔR*: resource reduction, Budget incrementally tightens** $\mathbf{Res_j(Net_{i-1}) - \Delta R_{i,j}}$

Advantages

- Supports multiple resource budgets at the same time
- Guarantees that budget will be satisfied because the resource consumption decreases monotonically
- Generates a family of networks (from each iteration) with different resource versus accuracy trade-offs
- Intuitive and can easily set a few additional hyperparameters ($\Delta R_{i,j}$)

NetAdapt: Simplified Example of One Iteration



1. Input

Network from Previous Iteration

Latency: 100ms
Budget: 80ms

2. Meet Budget

Layer 1

100ms 90ms 80ms

Selected

⋮

Layer 4

100ms 80ms

Selected

3. Maximize Accuracy

Acc: 60%

Selected

⋮

Acc: 40%

4. Output

Network for Next Iteration

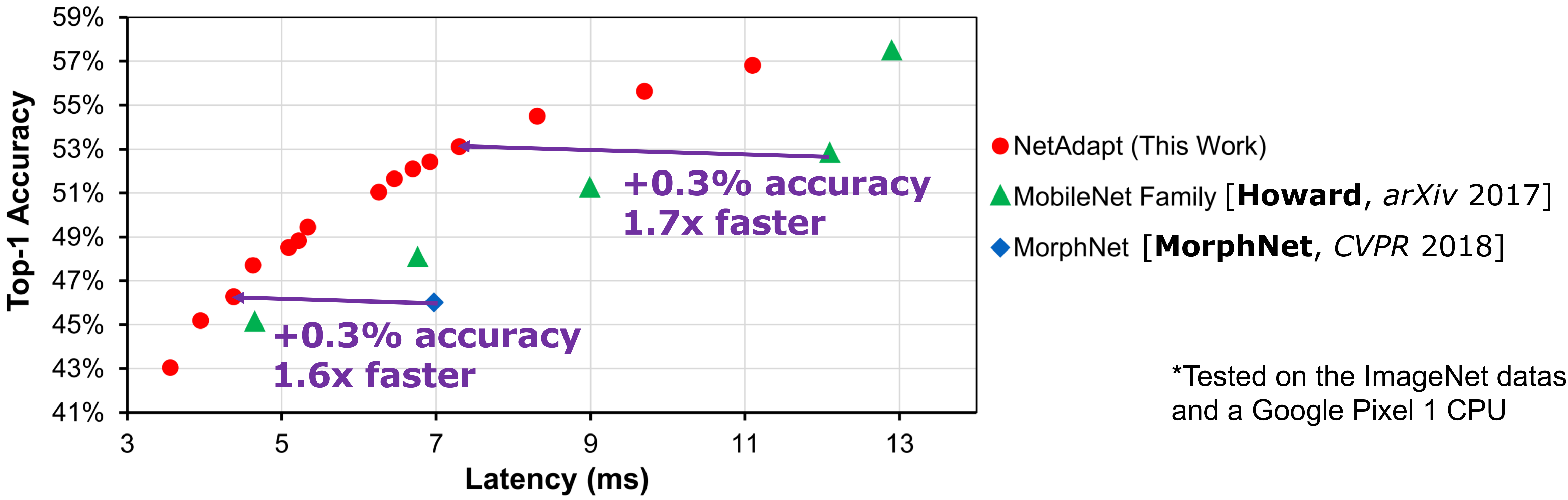
Latency: 80ms
Budget: 60ms

Code available at
http://netadapt.mit.edu

[**Yang**, *ECCV* 2018]

Improved Latency vs. Accuracy Tradeoff

□ NetAdapt boosts the measured inference speed of MobileNet by up to 1.7x with higher accuracy



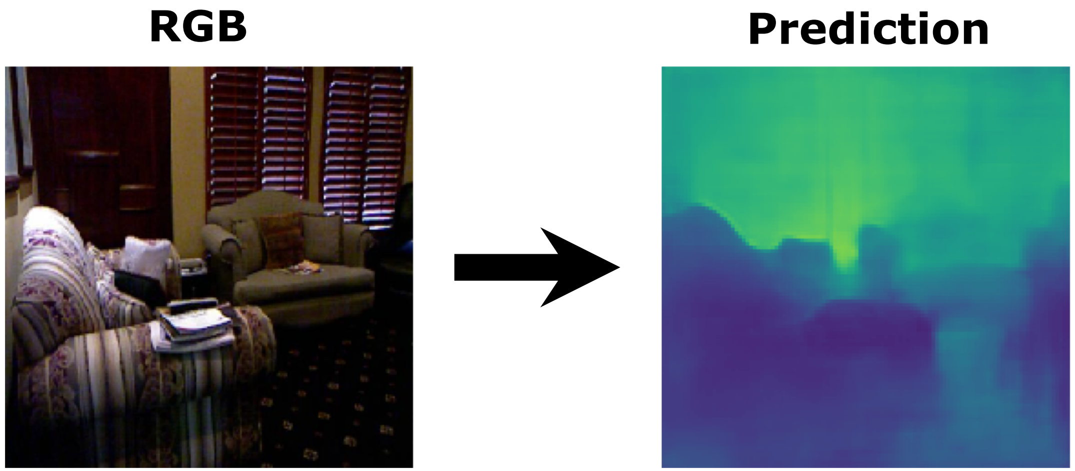
*Tested on the ImageNet dataset and a Google Pixel 1 CPU

Code available at http://netadapt.mit.edu

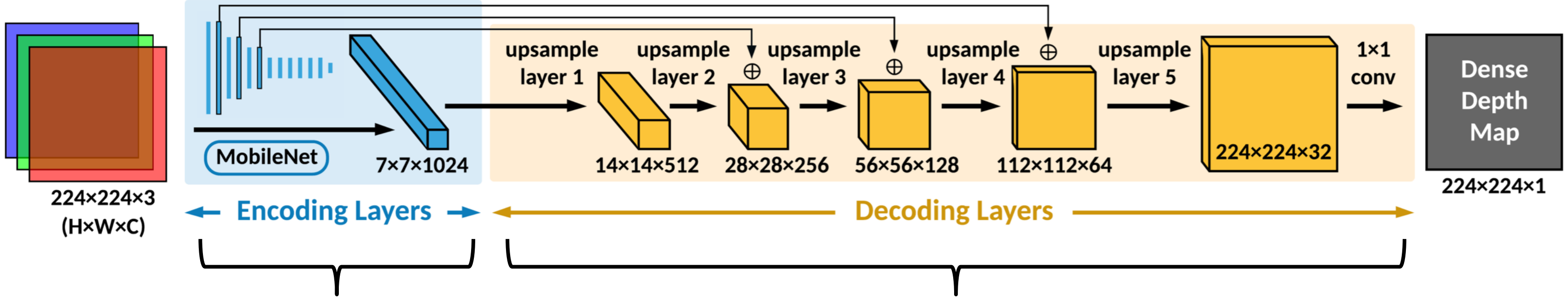
[**Yang**, *ECCV* 2018]

FastDepth: Fast Monocular Depth Estimation

Depth estimation from a single RGB image desirable, due to the relatively low cost and size of monocular cameras



Auto Encoder DNN Architecture (Dense Output)

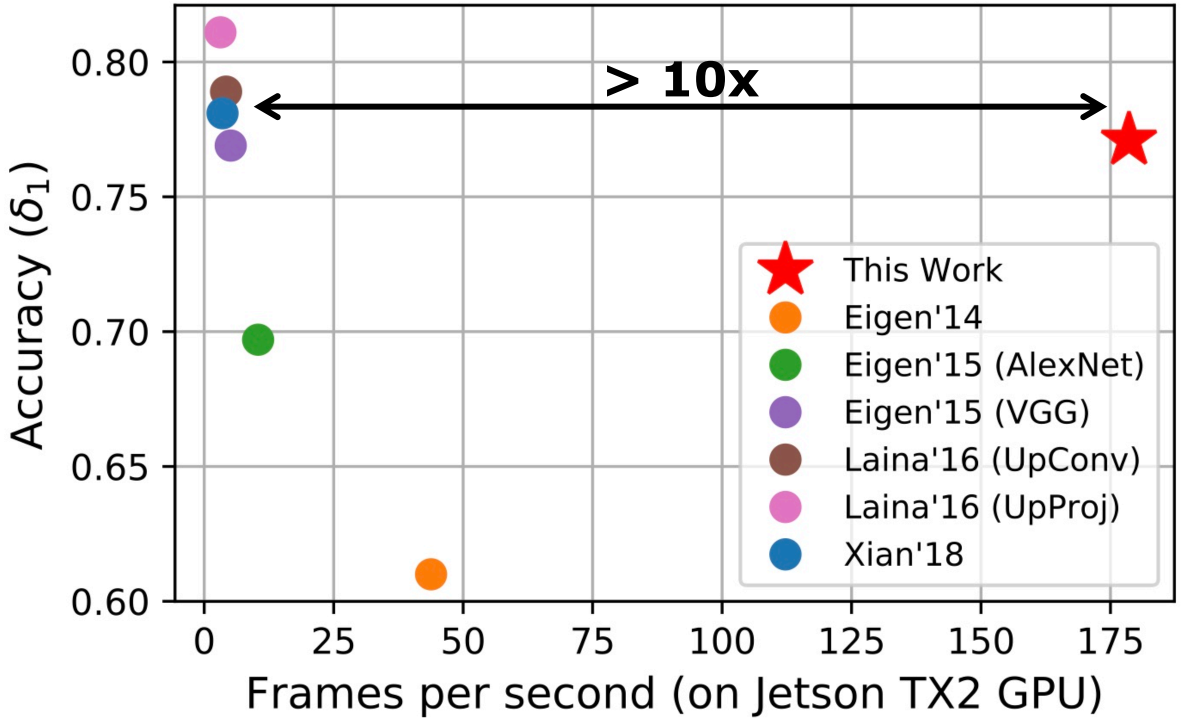


Reduction (similar to classification)

Expansion

FastDepth: Fast Monocular Depth Estimation

Apply *NetAdapt*, *compact network design*, and *depth wise decomposition* to enable depth estimation at **high frame rates on an embedded platform** while maintaining accuracy



Configuration: Batch size of one (32-bit float)

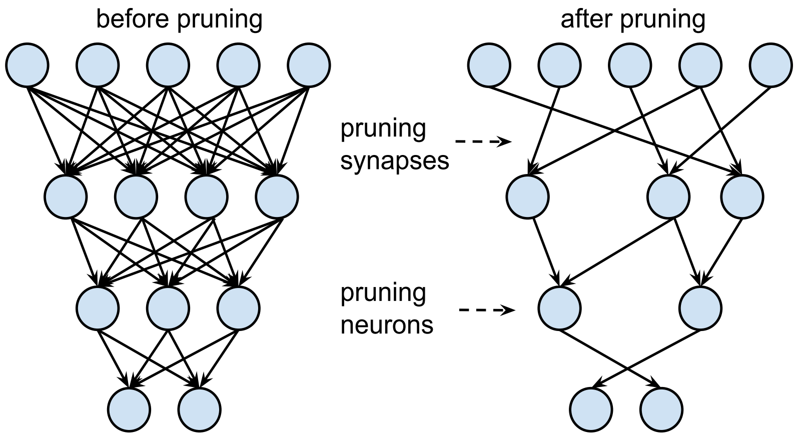
Models available at http://fastdepth.mit.edu

~40fps on an iPhone

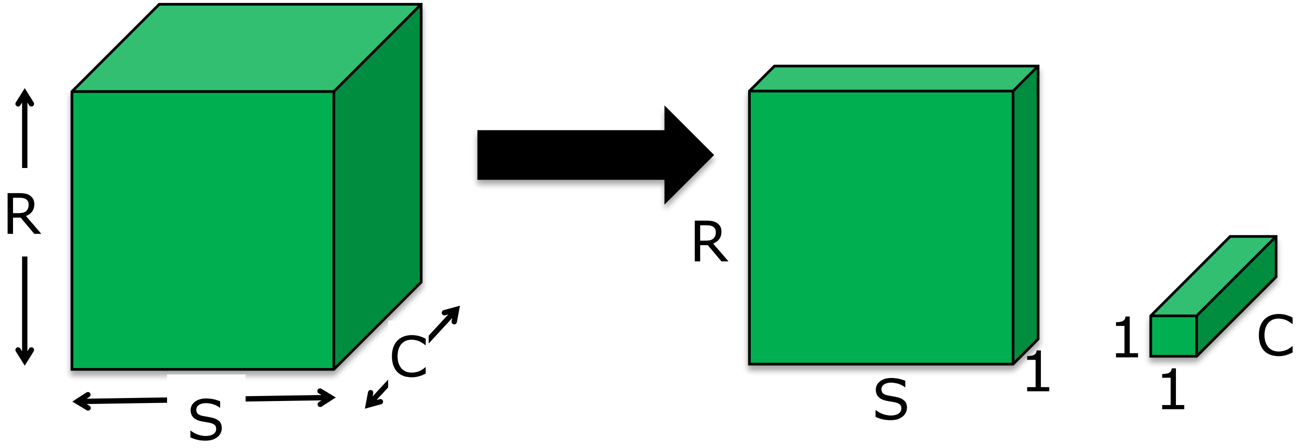
[**Wofk**, *ICRA* 2019]

Many Efficient DNN Design Approaches

Network Pruning



Efficient Network Architectures



Reduce Precision

32-bit float 1 01001010 00000000101000000000100

8-bit fixed 0 1100110

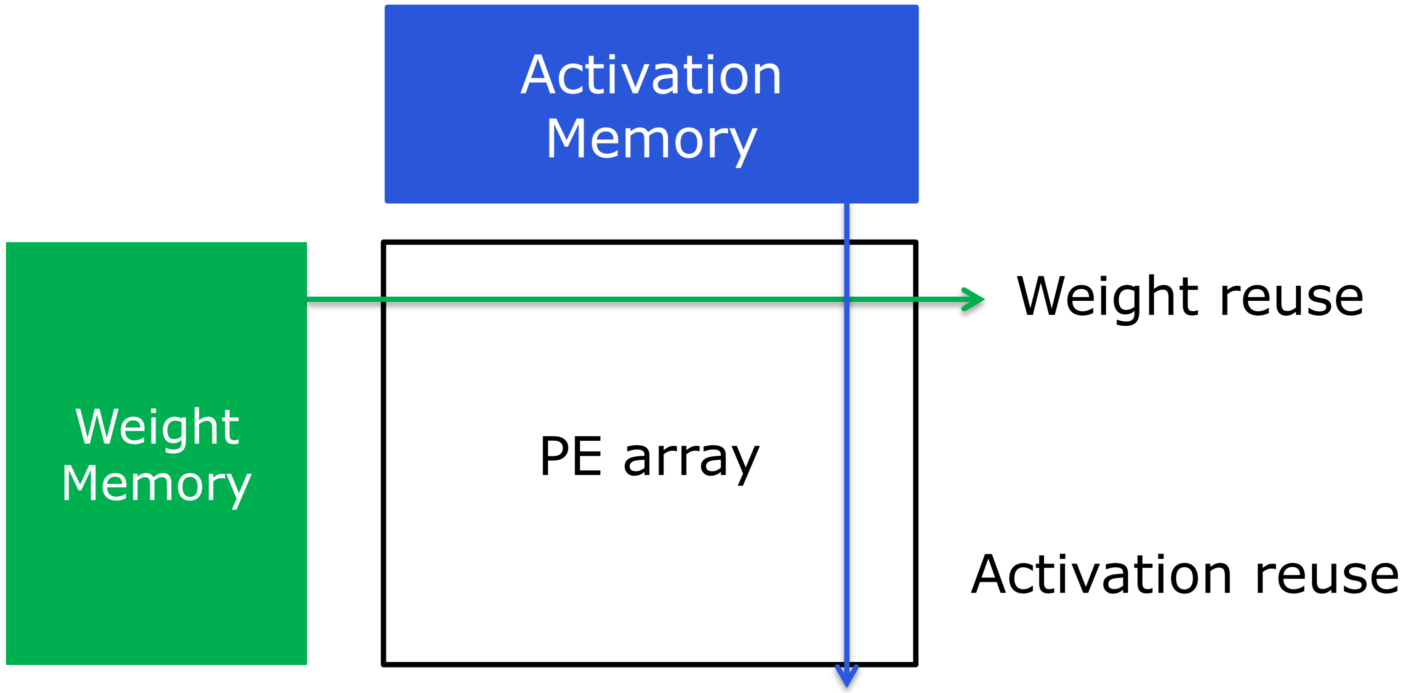
Binary 0

No guarantee that DNN algorithm designer will use a given approach.
Need flexible DNN processor!

[**Chen**, *SysML* 2018]

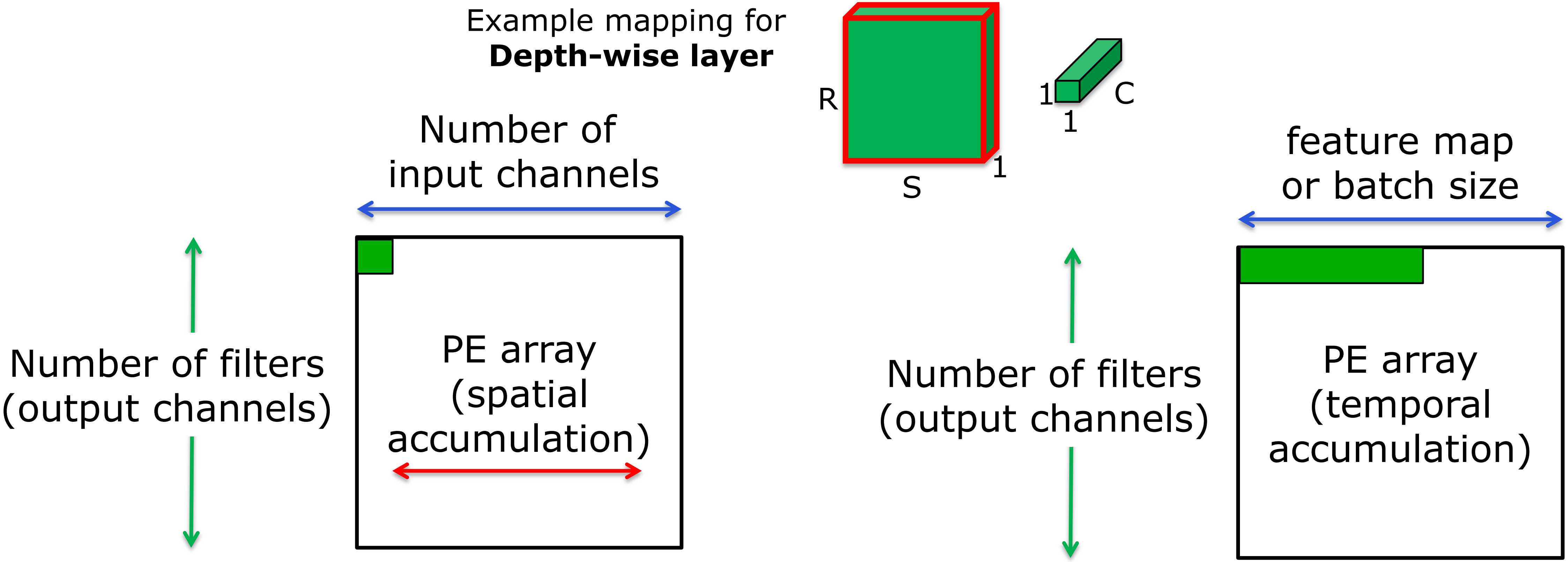
Limitations of Existing DNN Processors

- Specialized DNN processors often rely on certain properties of the DNN model in order to achieve high energy-efficiency
- Example: Reduce memory access by amortizing across PE array



Limitations of Existing DNN Processors

- □ Reuse depends on # of channels, feature map/batch size
  - Not efficient across all DNN models (e.g., efficient network architectures)



Example mapping for
Depth-wise layer

Number of input channels

Number of filters (output channels)

PE array (spatial accumulation)

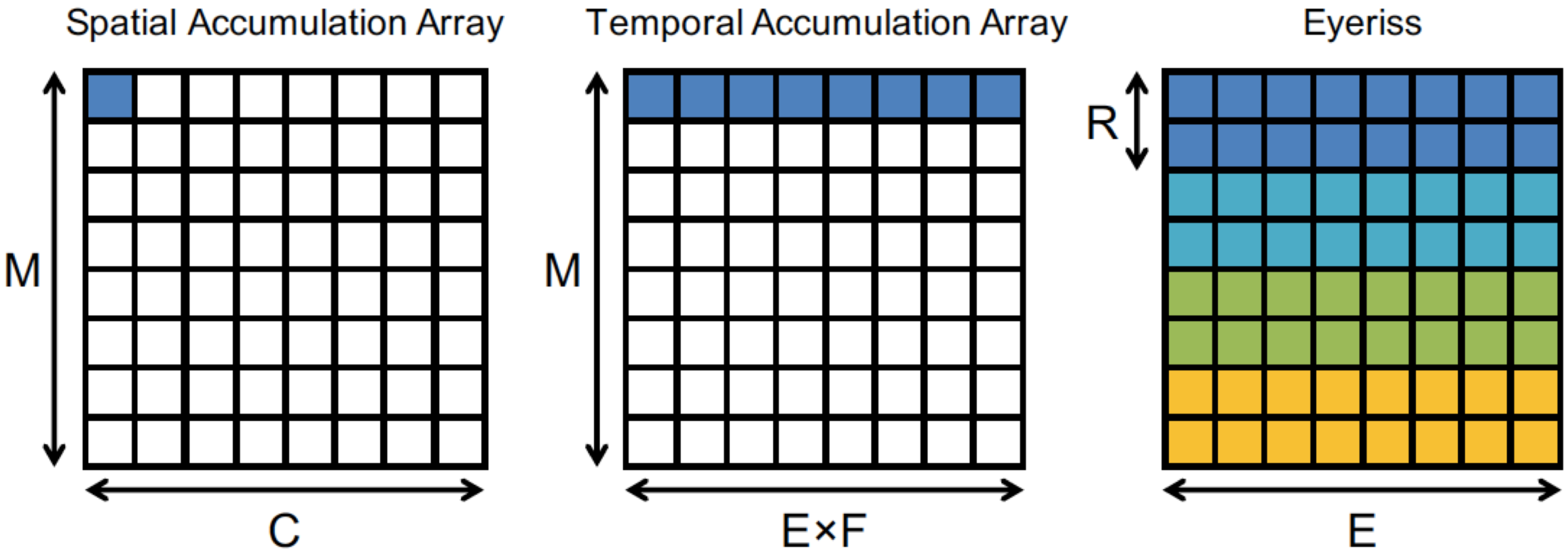
feature map or batch size

Number of filters (output channels)

PE array (temporal accumulation)

Need Flexible Dataflow

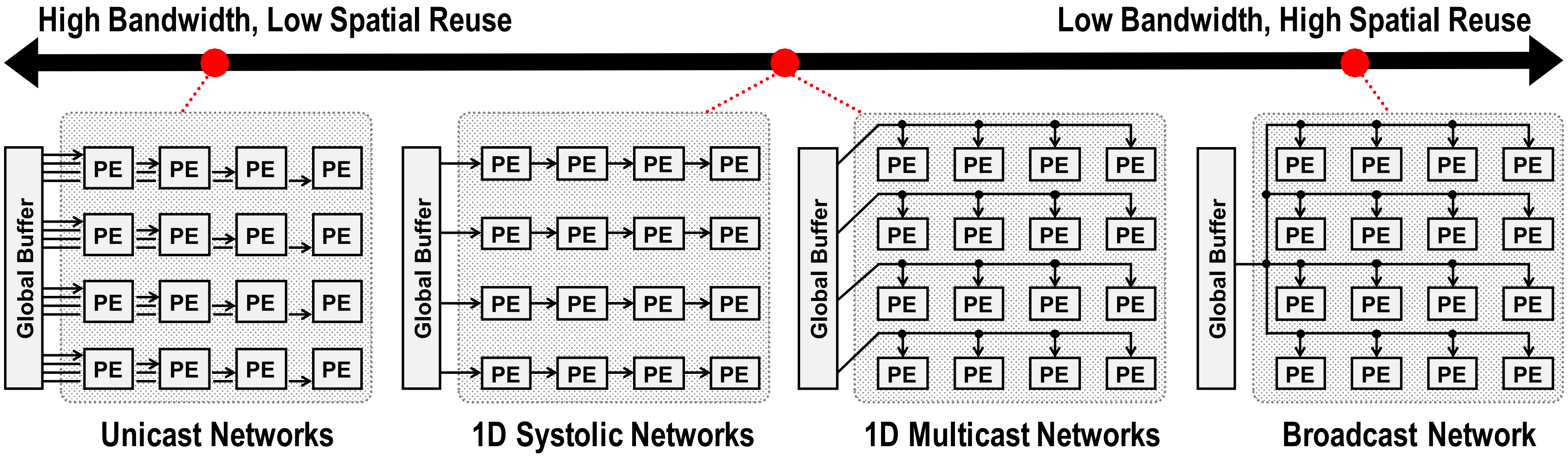
Use flexible dataflow (Row Stationary) to exploit reuse in any dimension of DNN to increase energy efficiency and array utilization



Example: Depth-wise layer

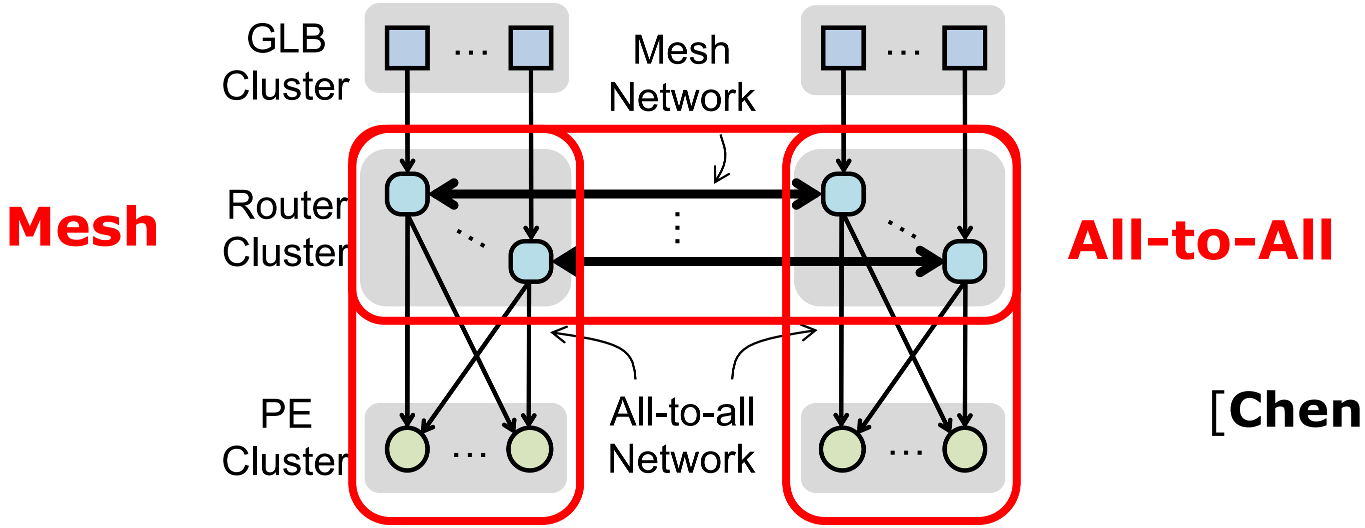
Need Flexible On-Chip Network for Varying Reuse

- □ When reuse available, need multicast to exploit spatial data reuse for energy efficiency and high array utilization
- □ When reuse not available, need unicast for high BW for weights for FC and weights & activations for high PE utilization
- □ An all-to-all on-chip network satisfies above but too expensive and not scalable

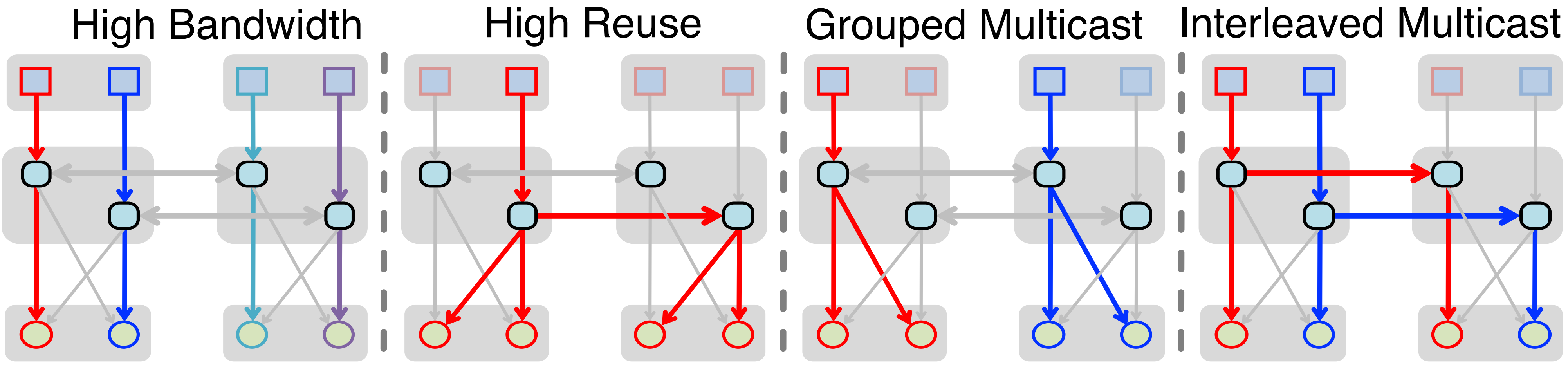


[**Chen**, *JETCAS* 2019]

Hierarchical Mesh



[**Chen**, *JETCAS* 2019]

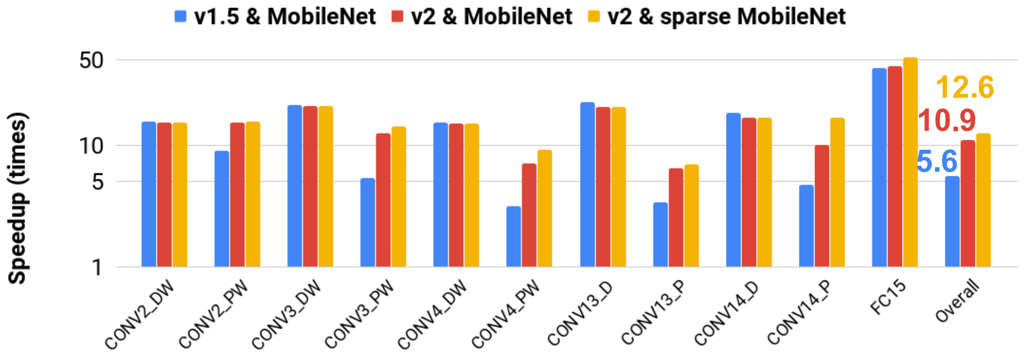


Eyeriss v2: Balancing Flexibility and Efficiency

Efficiently supports

- □ Wide range of filter shapes
  - ■ Large and Compact
- □ Different Layers
  - ■ CONV, FC, depth wise, etc.
- □ Wide range of sparsity
  - ■ Dense and Sparse
- □ Scalable architecture

Over an order of magnitude faster and more energy efficient than Eyeriss v1



Speed up over Eyeriss v1 scales with number of PEs

# of PEs	256	1024	16384
AlexNet	17.9x	71.5x	1086.7x
GoogLeNet	10.4x	37.8x	448.8x
MobileNet	15.7x	57.9x	873.0x

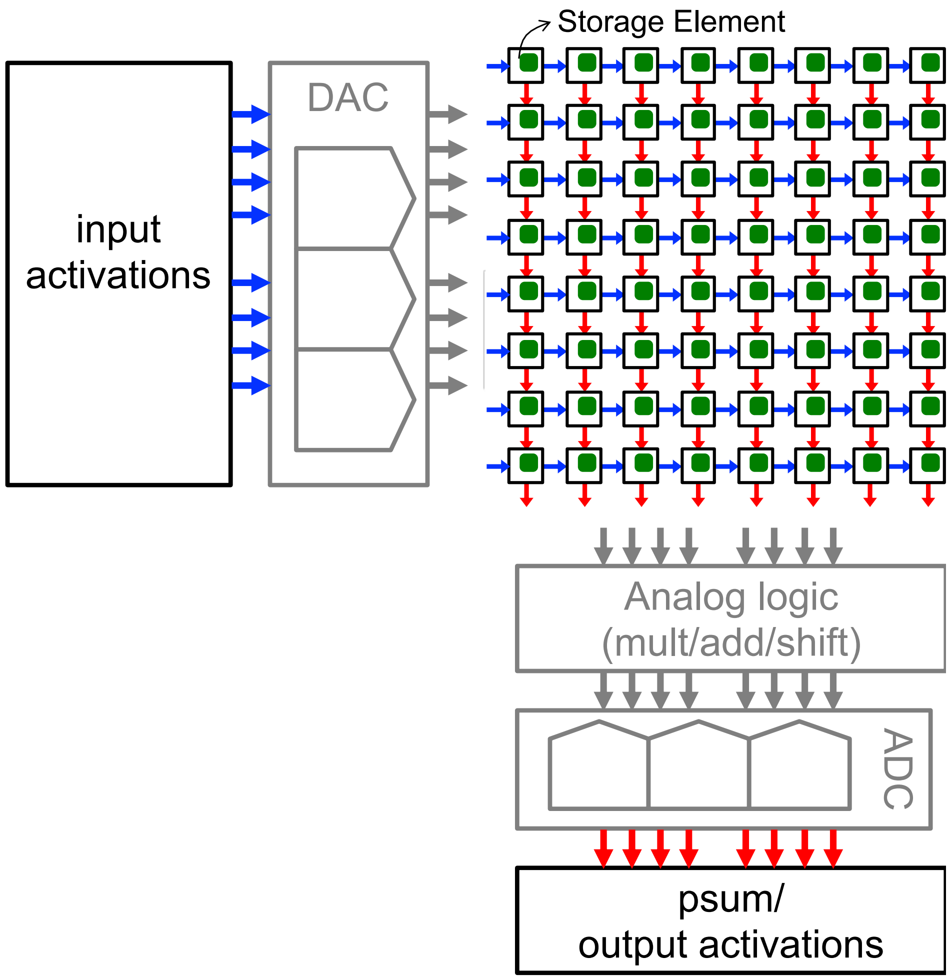
[**Chen**, *JETCAS* 2019]

Processing In Memory / In Memory Compute

- □ **Reduce weight data movement** by moving compute into the memory

- □ Implement as **matrix-vector multiply**

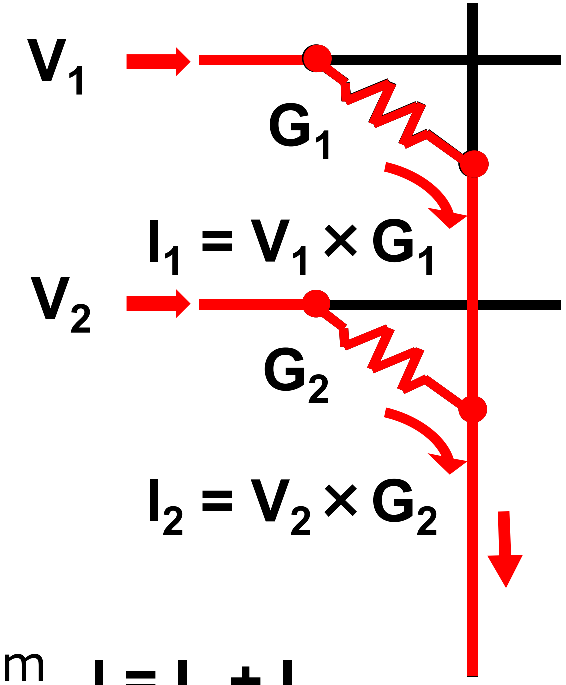
- □ **Increase weight bandwidth and amount of parallel MACs**



Design Considerations for PIM Accelerators

- **Prediction Accuracy**
  - **non-idealities of analog compute**
    - per chip training → expensive in practice
  - **lower bit widths for data and computation**
    - multiple devices per weight → decrease area density
    - bit serial processing → increase cycles per MAC
- **Hardware Efficiency**
  - **Data movement into/from array**
    - A/D and D/A conversion increase energy consumption and reduce area density
  - **Array utilization**
    - Large array size can amortize conversion cost → increase area density and data reuse → DNNs need to take advantage of this property

Activation is input voltage (V_i)
Weight is resistor conductance (G_i)



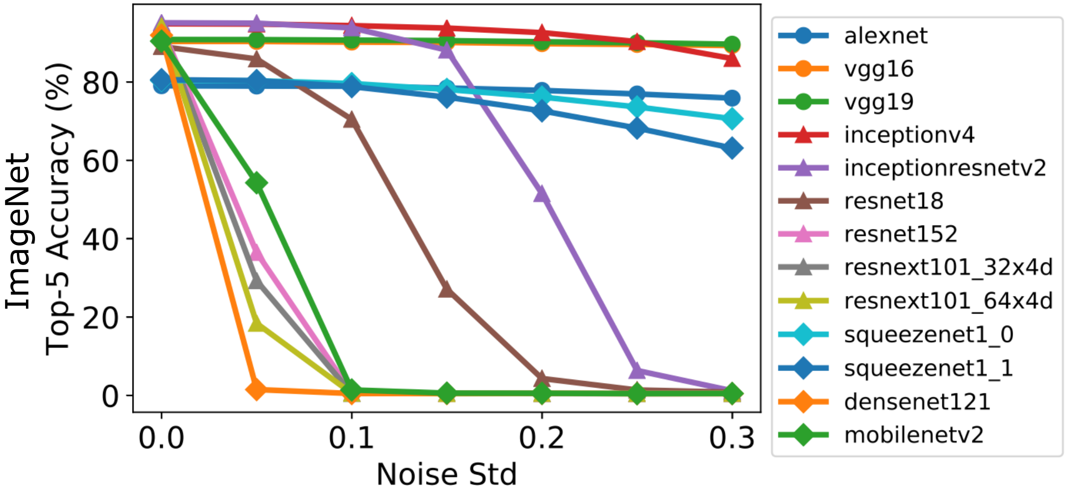
Partial sum is output current

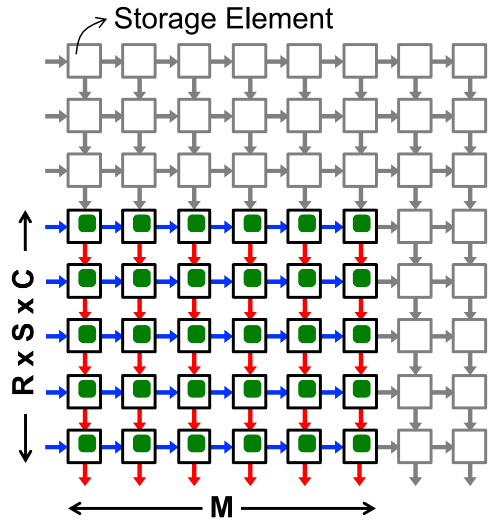
$$I = I_1 + I_2$$
$$= V_1 \times G_1 + V_2 \times G_2$$

Image Source: [**Shafiee**, *ISCA* 2016]

Design Considerations for DNNs on PIM

- □ Designing DNNs for PIM may differ from DNNs for digital processors
- □ Highest accuracy DNN on digital processor may be different on PIM
  - ■ Accuracy drops based on robustness to non-idealities
- □ Reducing number of weights is less desirable
  - ■ Since PIM is weight stationary, may be better to reduce number of activations
  - ■ PIM tend to have larger arrays → fewer weights may lead to low utilization on PIM
- □ Current trend is deeper and smaller filters
  - ■ For PIM, may be preferable to do shallower and larger filters





[**Yang**, *IEDM* 2019]

How to Evaluate Efficient DNN Approaches

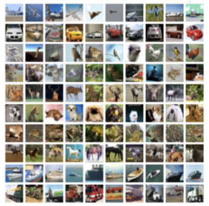
NeurIPS Tutorial: https://slideslive.com/38921492

Key Metrics: Much more than OPS/W!

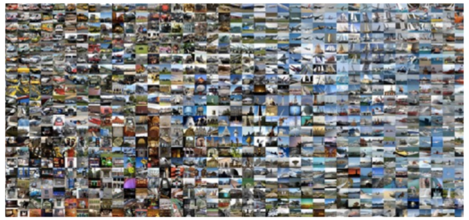
- **Accuracy**
  - Quality of result
- **Throughput**
  - Analytics on high volume data
  - Real-time performance (e.g., video at 30 fps)
- **Latency**
  - For interactive applications (e.g., autonomous navigation)
- **Energy and Power**
  - Embedded devices have limited battery capacity
  - Data centers have a power ceiling due to cooling cost
- **Hardware Cost**
  - $$$
- **Flexibility**
  - Range of DNN models and tasks
- **Scalability**
  - Scaling of performance with amount of resources

MNIST

CIFAR-10



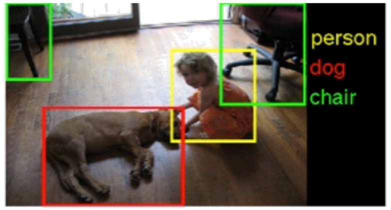
ImageNet



Embedded Device

Data Center

Computer Vision



Speech Recognition

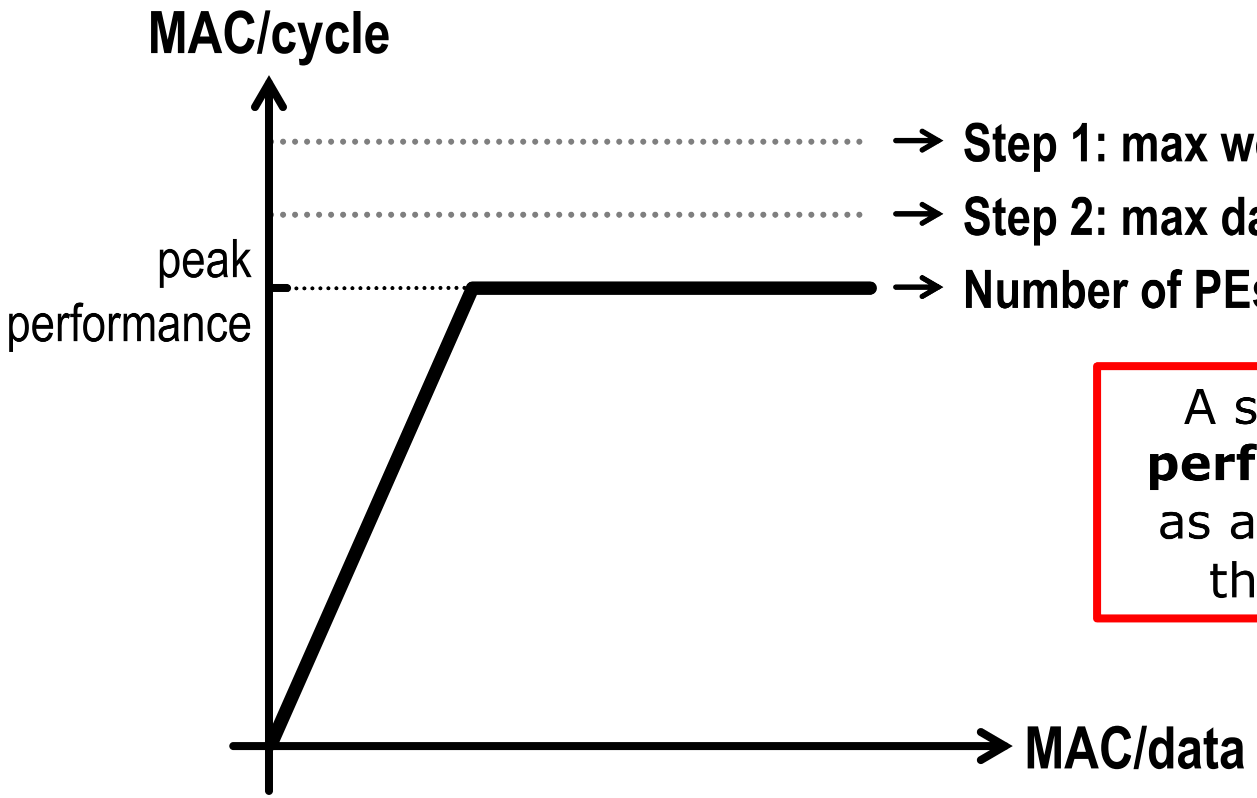
[**Sze**, *CICC* 2017]

Key Design Objectives of DNN Processor

- **Increase Throughput and Reduce Latency**
  - Reduce time per MAC
    - Reduce critical path → increase clock frequency
    - Reduce instruction overhead
  - Avoid unnecessary MACs (save cycles)
  - Increase number of processing elements (PE) → more MACs in parallel
    - Increase area density of PE or area cost of system
  - Increase PE utilization* → keep PEs busy
    - Distribute workload to as many PEs as possible
    - Balance the workload across PEs
    - Sufficient memory bandwidth to deliver workload to PEs (reduce idle cycles)
- Low latency has an additional constraint of **small batch size**

*(100% = peak performance)

Eyexam: Performance Evaluation Framework



→ **Step 1: max workload parallelism** (Depends on DNN Model)

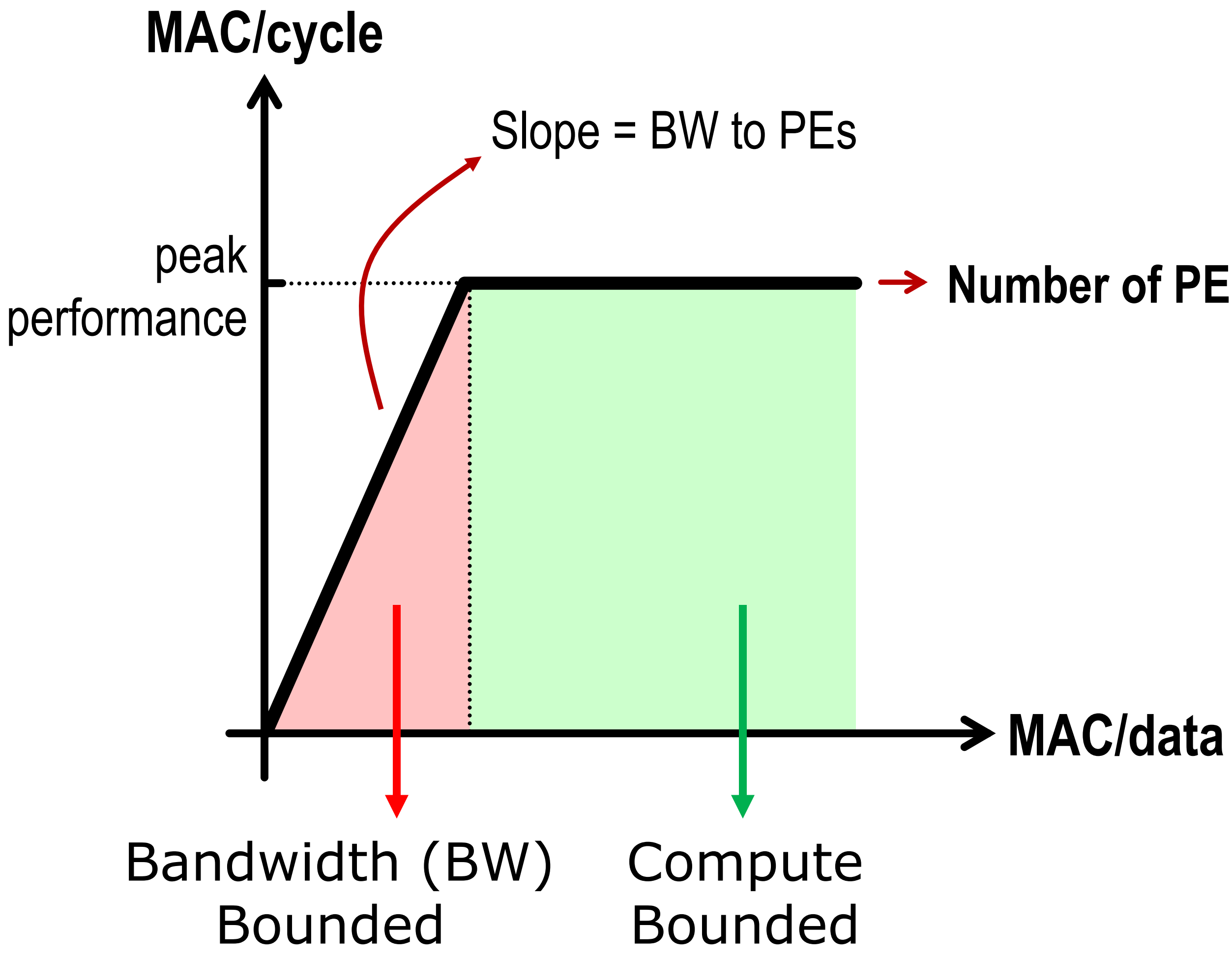
→ **Step 2: max dataflow parallelism**

→ **Number of PEs** (Theoretical Peak Performance)

A systematic way of understanding the **performance limits for DNN hardware** as a function of specific characteristics of the DNN model and hardware design

[**Chen**, *arXiv* 2019: https://arxiv.org/abs/1807.07928]

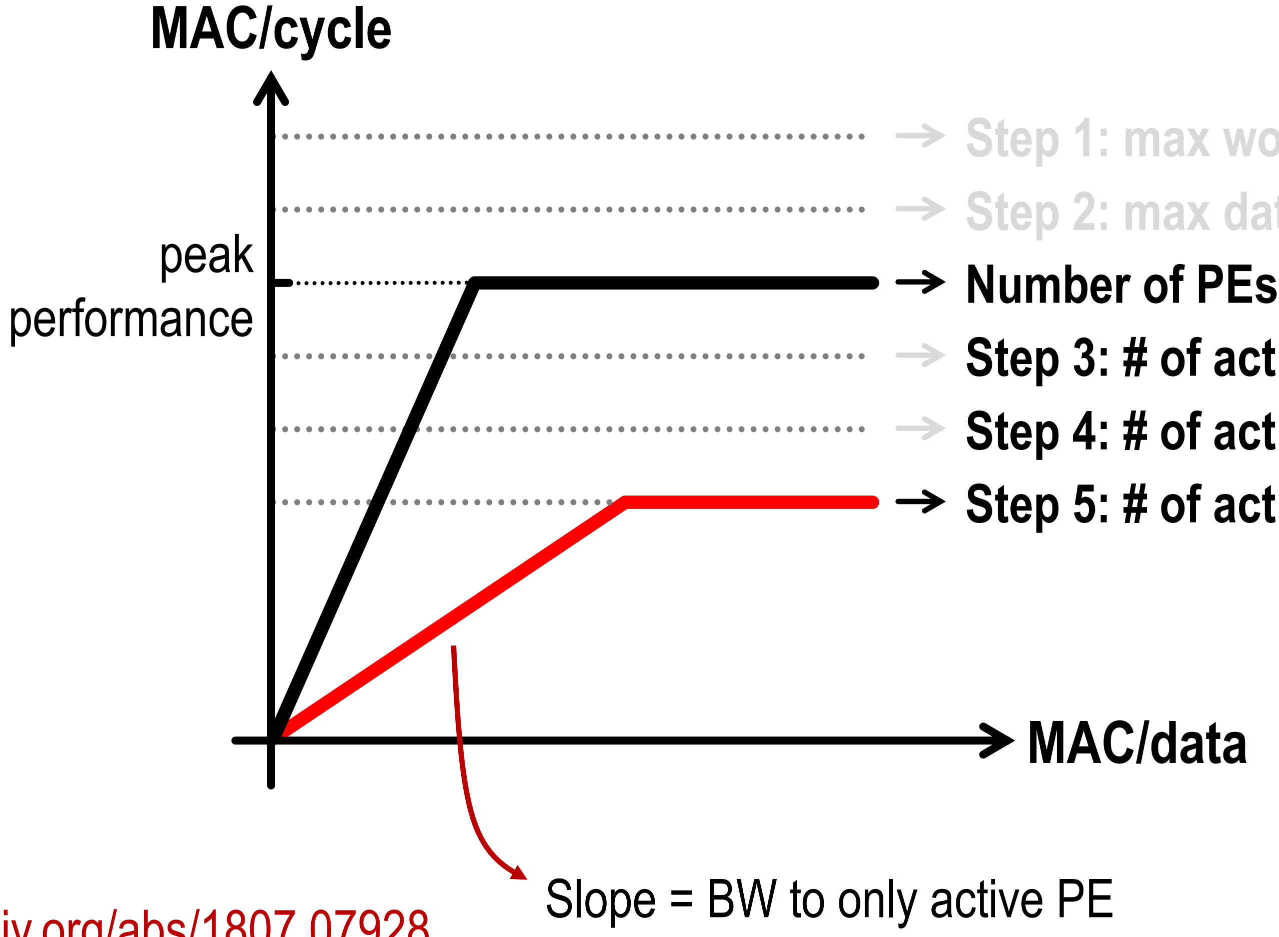
Eyexam: Performance Evaluation Framework



Based on Roofline Model

[**Williams**, *CACM* 2009]

Eyexam: Performance Evaluation Framework



→ Step 1: max workload parallelism

→ Step 2: max dataflow parallelism

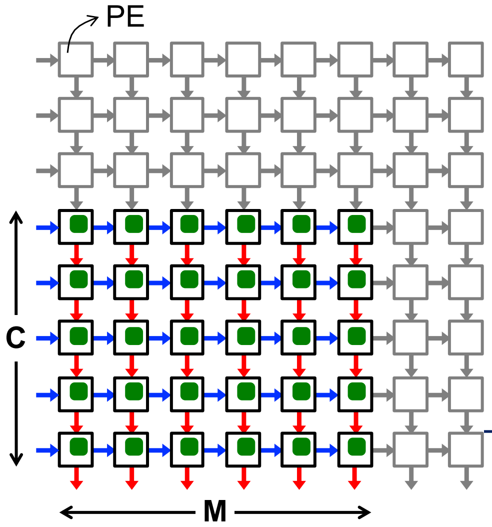
→ Number of PEs (Theoretical Peak Performance)

→ Step 3: # of active PEs under a finite PE array size

→ Step 4: # of active PEs under fixed PE array dimension

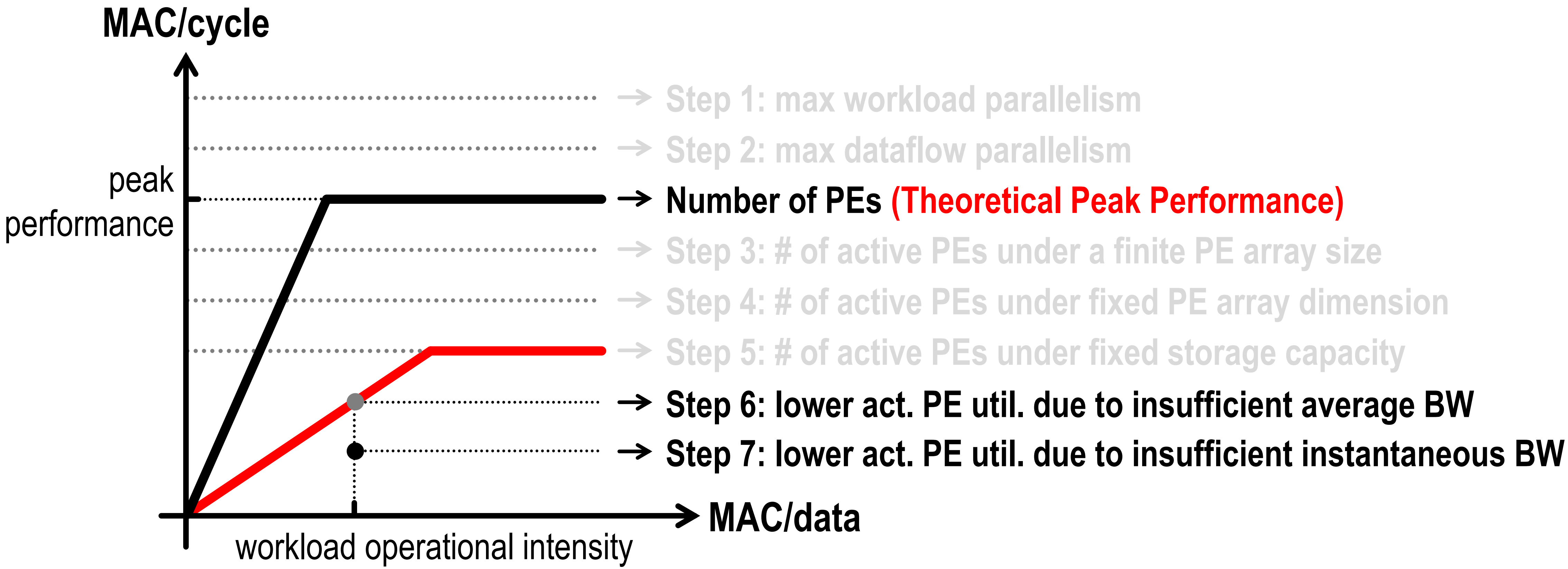
→ Step 5: # of active PEs under fixed storage capacity

Slope = BW to only active PE



https://arxiv.org/abs/1807.07928

Eyexam: Performance Evaluation Framework



→ Step 1: max workload parallelism

→ Step 2: max dataflow parallelism

→ Number of PEs (Theoretical Peak Performance)

→ Step 3: # of active PEs under a finite PE array size

→ Step 4: # of active PEs under fixed PE array dimension

→ Step 5: # of active PEs under fixed storage capacity

→ **Step 6: lower act. PE util. due to insufficient average BW**

→ **Step 7: lower act. PE util. due to insufficient instantaneous BW**

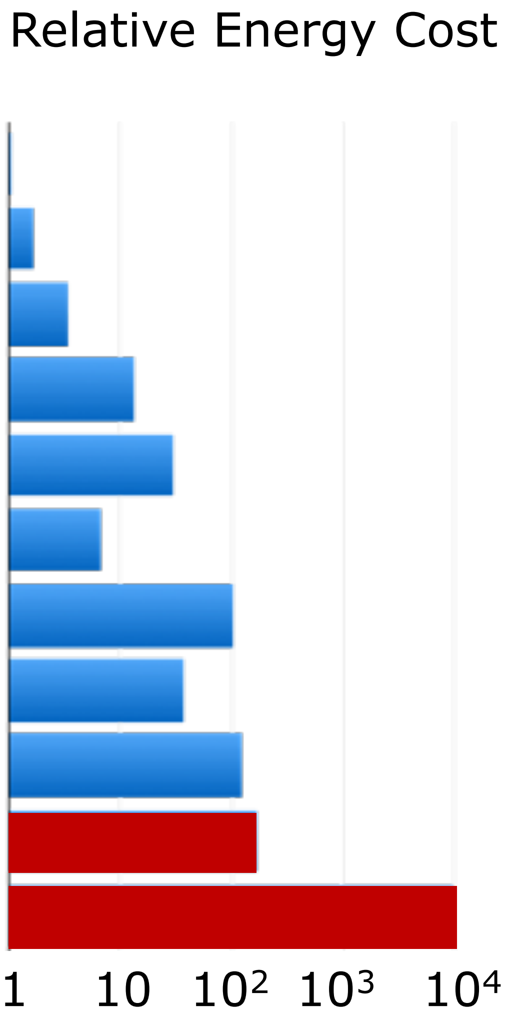
https://arxiv.org/abs/1807.07928

Key Design Objectives of DNN Processor

- □ **Reduce Energy and Power Consumption**
  - ■ Reduce data movement as it dominates energy consumption
    - □ Exploit data reuse
  - ■ Reduce energy per MAC
    - □ Reduce switching activity and/or capacitance
    - □ Reduce instruction overhead
  - ■ Avoid unnecessary MACs

- □ Power consumption is limited by heat dissipation, which limits the **maximum # of MACs in parallel** (i.e., throughput)

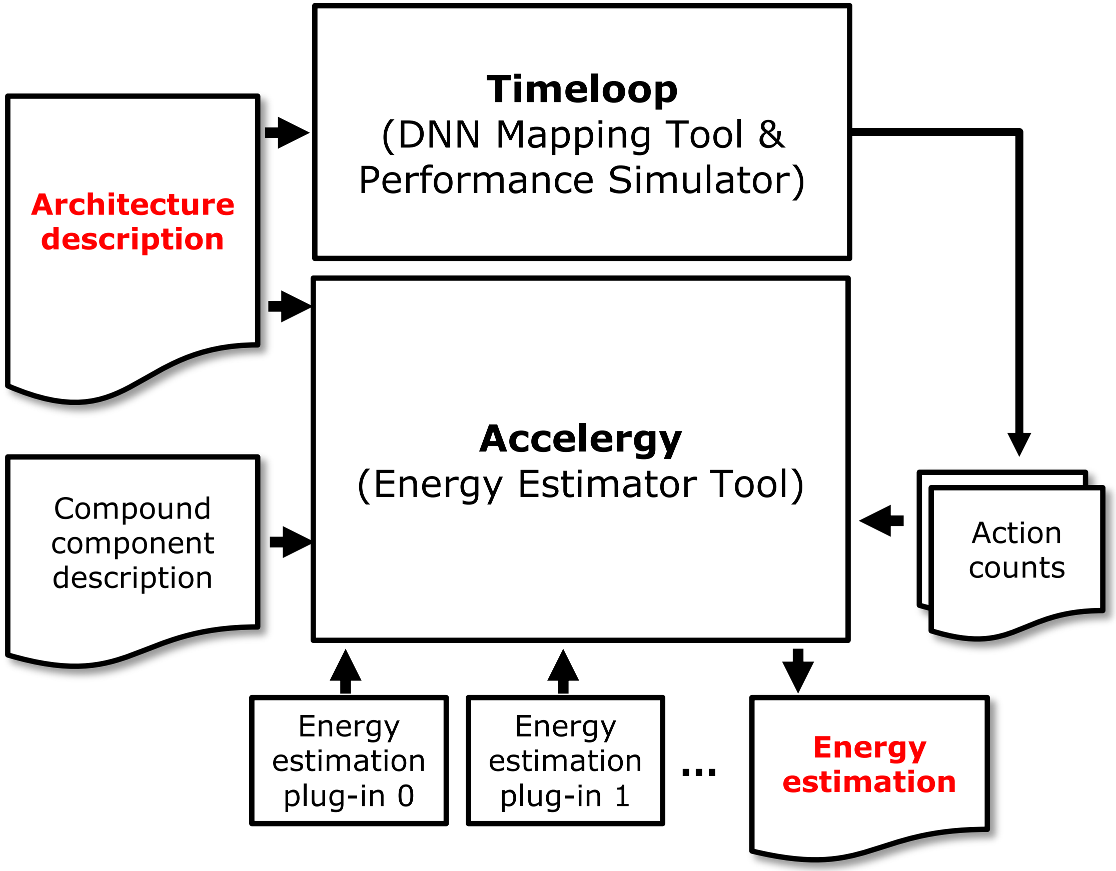
Operation:	Energy (pJ)
8b Add	0.03
16b Add	0.05
32b Add	0.1
16b FP Add	0.4
32b FP Add	0.9
8b Multiply	0.2
32b Multiply	3.1
16b FP Multiply	1.1
32b FP Multiply	3.7
32b SRAM Read (8KB)	5
32b DRAM Read	640



[**Horowitz**, *ISSCC* 2014]

DNN Processor Evaluation Tools

- Require systematic way to
  - Evaluate and compare wide range of DNN processor designs
  - Rapidly explore design space
- **Accelergy** [**Wu**, *ICCAD* 2019]
  - Early stage energy estimation tool at the architecture level
    - Estimate energy consumption based on architecture level components (e.g., # of PEs, memory size, on-chip network)
  - Evaluate architecture level energy impact of emerging devices
    - Plug-ins for different technologies
- **Timeloop** [**Parashar**, *ISPASS* 2019]
  - DNN mapping tool
  - Performance Simulator → Action counts

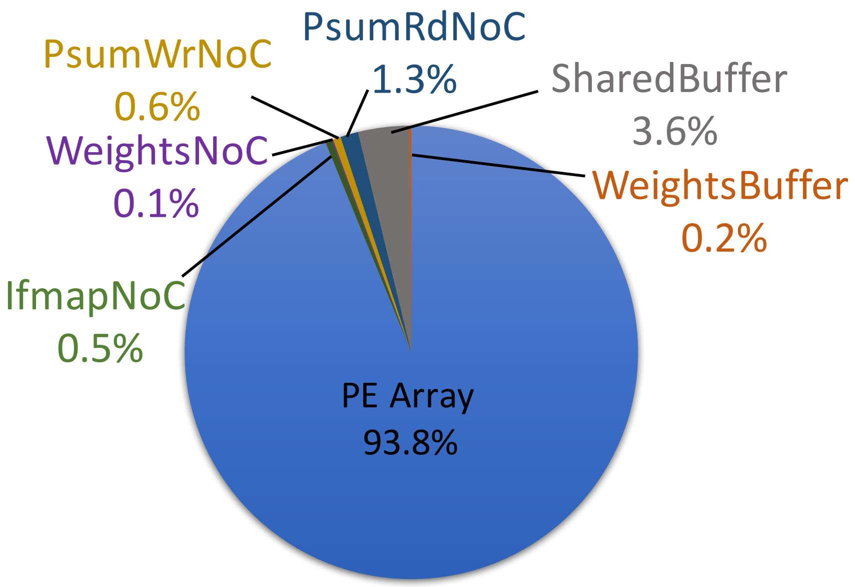


Architecture description
Timeloop
(DNN Mapping Tool & Performance Simulator)
Accelergy
(Energy Estimator Tool)
Compound component description
Action counts
Energy estimation plug-in 0
Energy estimation plug-in 1
…
Energy estimation

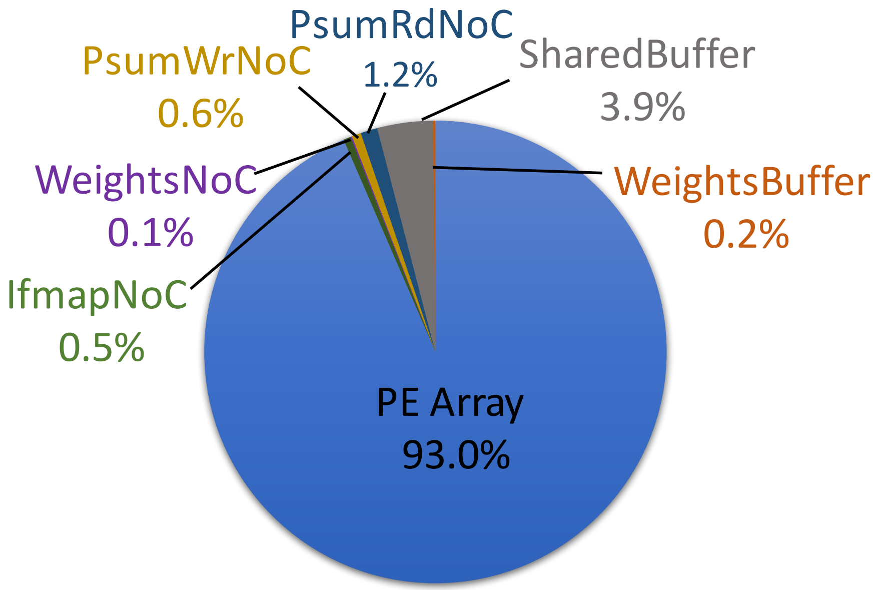
Open-source code available at:
http://accelergy.mit.edu

Accelergy Estimation Validation

- □ Validation on Eyeriss [**Chen**, *ISSCC* 2016]
  - Achieves 95% accuracy compared to post-layout simulations
  - Can accurately captures energy breakdown at different granularities



Ground Truth Energy Breakdown

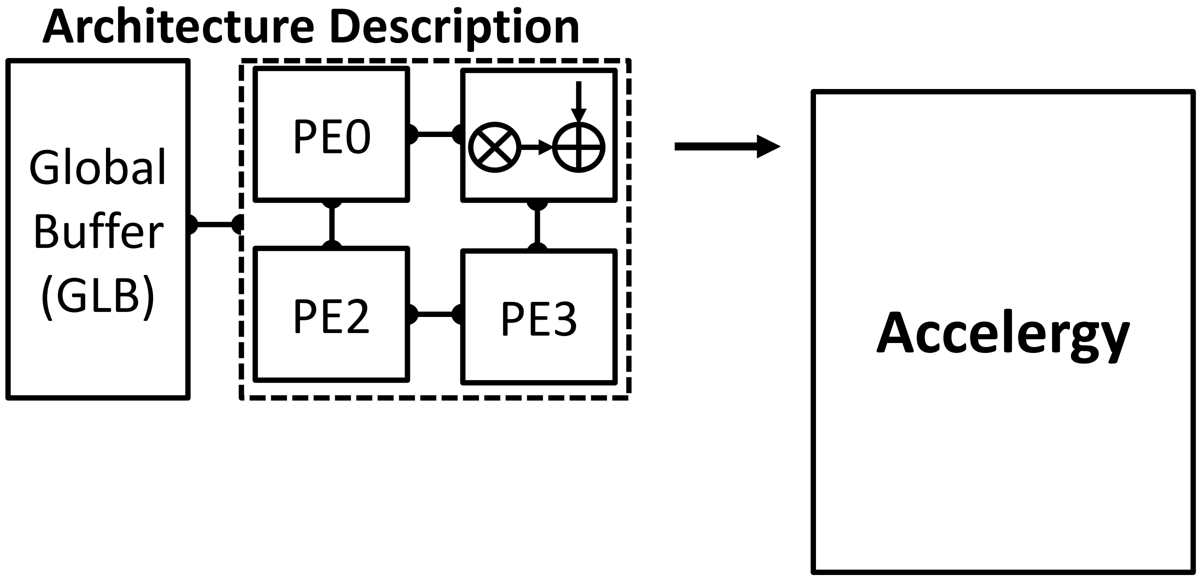


Accelergy Energy Breakdown

Open-source code available at: http://accelergy.mit.edu

[**Wu**, *ICCAD* 2019]

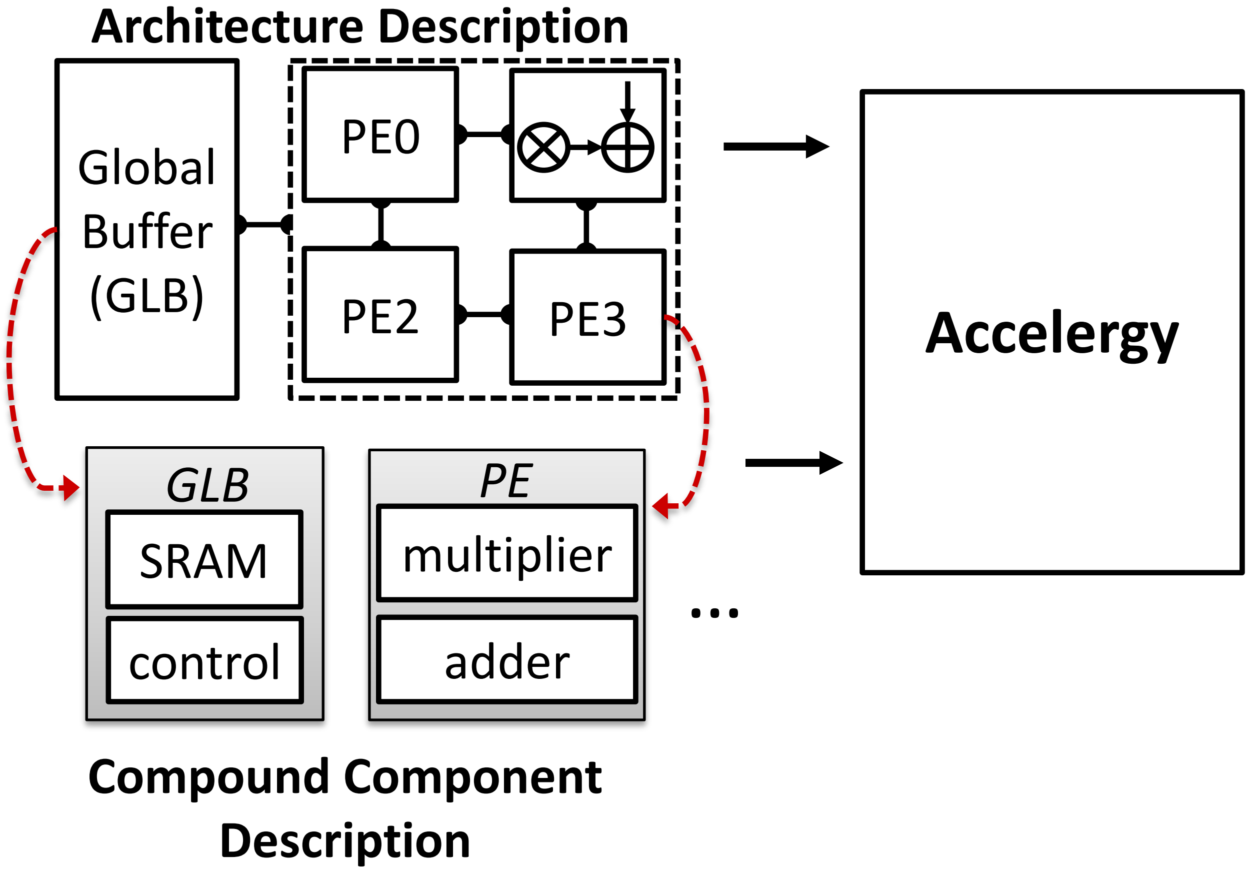
Accelergy Infrastructure



Open-source code available at: http://accelergy.mit.edu

[**Wu**, *ICCAD* 2019]

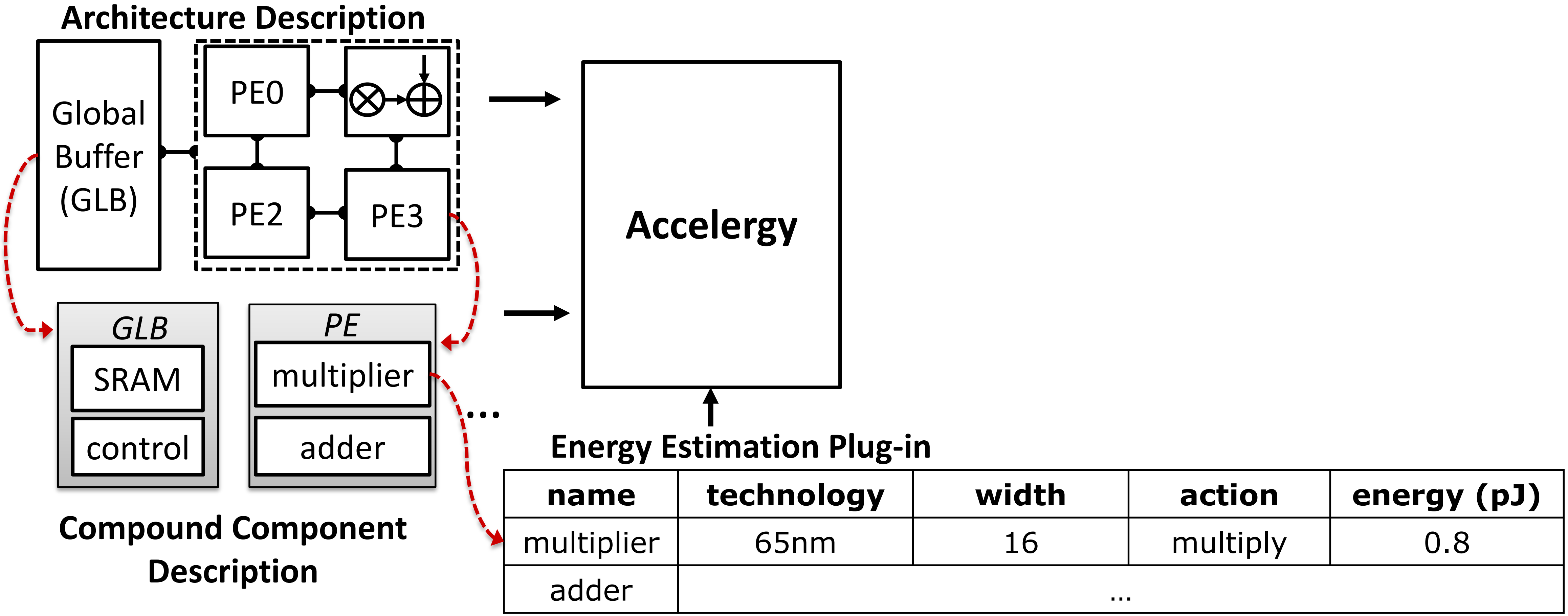
Accelergy Infrastructure



Open-source code available at: http://accelergy.mit.edu

[**Wu**, *ICCAD* 2019]

Accelergy Infrastructure

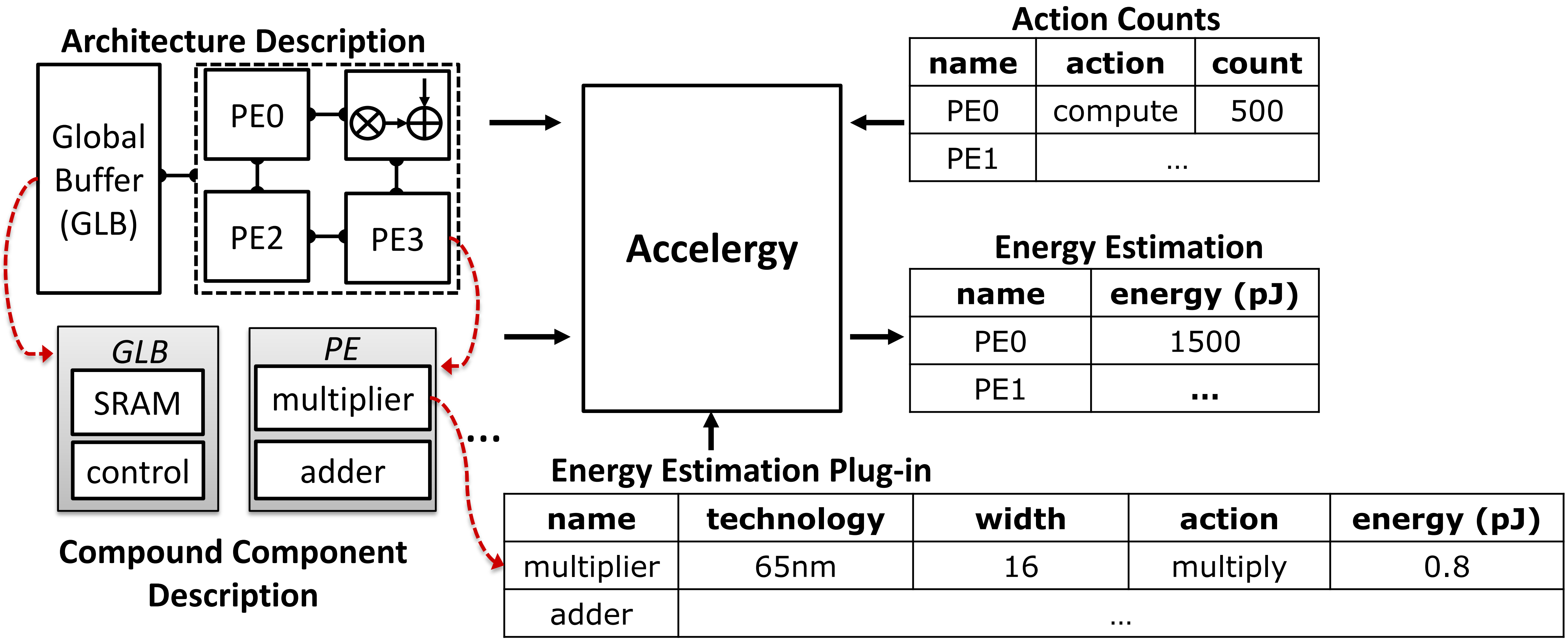


name	technology	width	action	energy (pJ)
multiplier	65nm	16	multiply	0.8
adder	…			

Open-source code available at: http://accelergy.mit.edu

[**Wu**, *ICCAD* 2019]

Accelergy Infrastructure



Action Counts

name	action	count
PE0	compute	500
PE1	…	

Energy Estimation

name	energy (pJ)
PE0	1500
PE1	…

Energy Estimation Plug-in

name	technology	width	action	energy (pJ)
multiplier	65nm	16	multiply	0.8
adder	…			

Open-source code available at: http://accelergy.mit.edu

[**Wu**, *ICCAD* 2019]

Key Design Objectives of DNN Processor

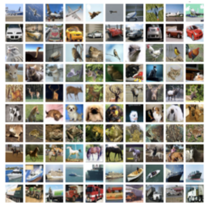
- **Flexibility**
  - Reduce overhead of supporting flexibility
  - Maintain efficiency across wide range of DNN models
    - Different layer shapes impact the amount of
      - Required storage and compute
      - Available data reuse that can be exploited
    - Different precision across layers & data types (weight, activation, partial sum)
    - Different degrees of sparsity (number of zeros in weights or activations)
    - Types of DNN layers and computation beyond MACs (e.g., activation functions)
- **Scalability**
  - Increase how performance (i.e., throughput, latency, energy, power) scales with increase in amount of resources (e.g., number of PEs, amount of memory, etc.)

Specifications to Evaluate Metrics

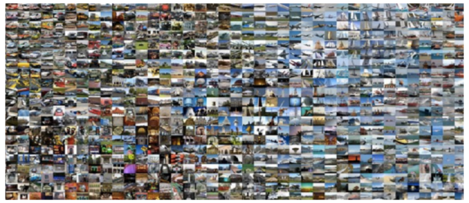
- □ **Accuracy**
  - ■ Difficulty of dataset and/or task should be considered
  - ■ Difficult tasks typically require more complex DNN models
- □ **Throughput**
  - ■ Number of PEs with utilization (not just peak performance)
  - ■ Runtime for running specific DNN models
- □ **Latency**
  - ■ Batch size used in evaluation
- □ **Energy and Power**
  - ■ Power consumption for running specific DNN models
  - ■ Off-chip memory access (e.g., DRAM)
- □ **Hardware Cost**
  - ■ On-chip storage, # of PEs, chip area + process technology
- □ **Flexibility**
  - ■ Report performance across a wide range of DNN models
  - ■ Define range of DNN models that are efficiently supported

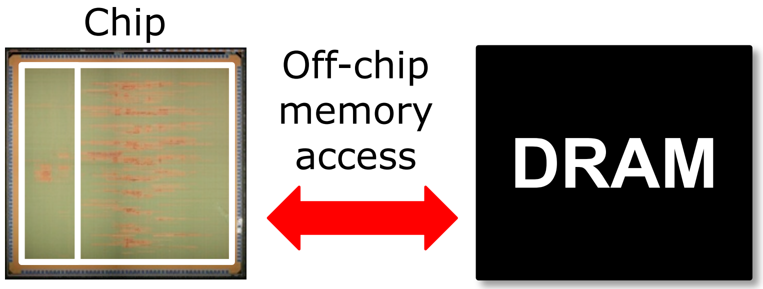
MNIST

CIFAR-10

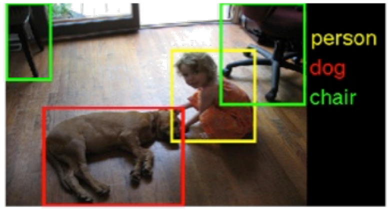


ImageNet





Computer Vision



Speech Recognition

[**Sze**, *CICC* 2017]

Comprehensive Coverage for Evaluation

- All metrics should be reported for fair evaluation of design tradeoffs

- Examples of what can happen if a certain metric is omitted:
  - **Without the accuracy** given for a specific dataset and task, one could run a simple DNN and claim low power, high throughput, and low cost – however, the processor might not be usable for a meaningful task
  - **Without reporting the off-chip memory access**, one could build a processor with *only* MACs and claim low cost, high throughput, high accuracy, and low chip power – however, when evaluating system power, the off-chip memory access would be substantial

- Are results measured or simulated? On what test data?

Example Evaluation Process

The evaluation process for whether a DNN processor is a viable solution for a given application might go as follows:

1. **Accuracy** determines if it can perform the given task
2. **Latency and throughput** determine if it can run fast enough and in real-time
3. **Energy and power consumption** will primarily dictate the form factor of the device where the processing can operate
4. **Cost**, which is primarily dictated by the chip area, determines how much one would pay for this solution
5. **Flexibility** determines the range of tasks it can support

Design Considerations for Co-Design

- **Impact on accuracy**
  - Consider quality of baseline (initial) DNN model, difficulty of task and dataset
  - Sweep curve of accuracy versus latency/energy to see the full tradeoff

- **Does hardware cost exceed benefits?**
  - Need extra hardware to support variable precision and shapes or to identify sparsity
  - Granularity impacts hardware overhead as well as accuracy

- **Evaluation**
  - Avoid only evaluating impact based on number of weights or MACs as they may not be sufficient for evaluating energy consumption and latency

Design Considerations for Co-Design

- **Time required to perform co-design**
  - e.g., Difficulty of tuning affected by
    - Number of hyperparameters
    - Uncertainty in relationship between hyperparameters and impact on performance
- **Other aspects that affect accuracy, latency or energy**
  - Type of data augmentation and preprocessing
  - Optimization algorithm, hyperparameters, learning rate schedule, batch size
  - Training and finetuning time
  - Deep learning libraries and quality of the code
- **How does the approach perform on different platforms?**
  - Is the approach a general method, or applicable on specific hardware?

Summary

- **The number of weights and MACs are not sufficient for evaluating the energy consumption and latency of DNNs**
  - Designers of efficient DNN algorithms should directly target direct metrics such as energy and latency and incorporate into the design

- **Many of the existing DNN processors rely on certain properties of the DNN which cannot be guaranteed as the wide range of efficient DNN algorithm design techniques has resulted in a diverse set of DNNs**
  - DNN hardware used to process these DNNs should be sufficiently flexible to support a wide range of techniques efficiently

- **Evaluate DNN hardware on a comprehensive set of benchmarks and metrics**

Acknowledgements

Joel Emer

Thomas Heldt

Sertac Karaman

Research conducted in the **MIT Energy-Efficient Multimedia Systems Group** would not be possible without the support of the following organizations:

3M AFOSR AIR FORCE OFFICE OF SCIENTIFIC RESEARCH ANALOG DEVICES BROADCOM. DARPA

Google intel IBM NSF NVIDIA.

QUALCOMM® SAMSUNG SRC TEXAS INSTRUMENTS tsmc

For updates on our research

Follow @eems_mit

Additional Resources

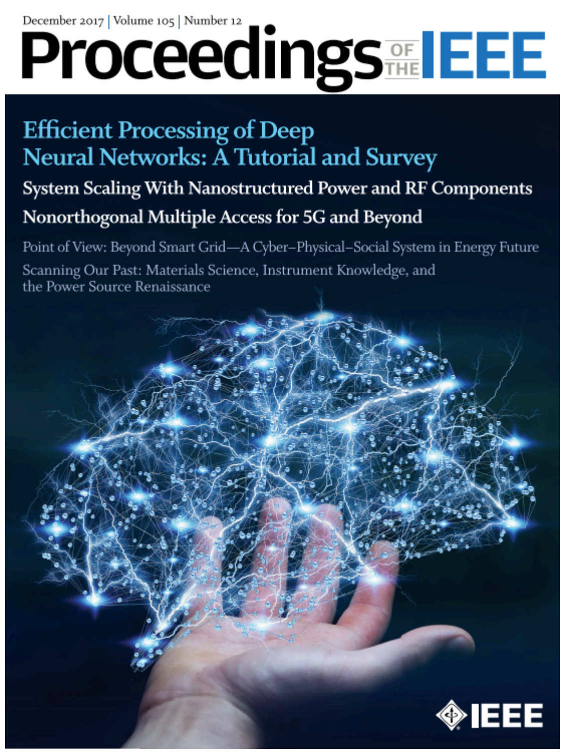
V. Sze, Y.-H. Chen, T-J. Yang, J. Emer,
"***Efficient Processing of Deep Neural Networks: A Tutorial and Survey***," Proceedings of the IEEE, Dec. 2017

Book Coming Soon!

NeurIPS Tutorial: https://slideslive.com/38921492
DNN tutorial website: http://eyeriss.mit.edu/tutorial.html

MIT Professional Education Course on
"Designing Efficient Deep Learning Systems"
http://professional-education.mit.edu/deeplearning

More info about our research on efficient computing for DNNs, robotics, and health care
http://sze.mit.edu



December 2017 | Volume 105 | Number 12

Proceedings OF THE IEEE

Efficient Processing of Deep Neural Networks: A Tutorial and Survey

System Scaling With Nanostructured Power and RF Components

Nonorthogonal Multiple Access for 5G and Beyond

Point of View: Beyond Smart Grid—A Cyber–Physical–Social System in Energy Future

Scanning Our Past: Materials Science, Instrument Knowledge, and the Power Source Renaissance

IEEE

For updates

EEMS Mailing List

Follow @eems_mit

References

- □ **Limitations of Existing Efficient DNN Approaches**
  - Y.-H. Chen*, T.-J. Yang*, J. Emer, V. Sze, "Understanding the Limitations of Existing Energy-Efficient Design Approaches for Deep Neural Networks," SysML Conference, February 2018.
  - V. Sze, Y.-H. Chen, T.-J. Yang, J. Emer, "Efficient Processing of Deep Neural Networks: A Tutorial and Survey," Proceedings of the IEEE, vol. 105, no. 12, pp. 2295-2329, December 2017.
  - Hardware Architecture for Deep Neural Networks: http://eyeriss.mit.edu/tutorial.html

- □ **Co-Design of Algorithms and Hardware for Deep Neural Networks**
  - T.-J. Yang, Y.-H. Chen, V. Sze, "Designing Energy-Efficient Convolutional Neural Networks using Energy-Aware Pruning," IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2017.
  - Energy estimation tool: http://eyeriss.mit.edu/energy.html
  - T.-J. Yang, A. Howard, B. Chen, X. Zhang, A. Go, V. Sze, H. Adam, "NetAdapt: Platform-Aware Neural Network Adaptation for Mobile Applications," European Conference on Computer Vision (ECCV), 2018. http://netadapt.mit.edu/
  - D. Wofk*, F. Ma*, T.-J. Yang, S. Karaman, V. Sze, "FastDepth: Fast Monocular Depth Estimation on Embedded Systems," IEEE International Conference on Robotics and Automation (ICRA), May 2019. http://fastdepth.mit.edu/

References

Energy-Efficient Hardware for Deep Neural Networks

- Project website: http://eyeriss.mit.edu
- Y.-H. Chen, T. Krishna, J. Emer, V. Sze, “Eyeriss: An Energy-Efficient Reconfigurable Accelerator for Deep Convolutional Neural Networks,” IEEE Journal of Solid State Circuits (JSSC), ISSCC Special Issue, Vol. 52, No. 1, pp. 127-138, January 2017.
- Y.-H. Chen, J. Emer, V. Sze, “Eyeriss: A Spatial Architecture for Energy-Efficient Dataflow for Convolutional Neural Networks,” International Symposium on Computer Architecture (ISCA), pp. 367-379, June 2016.
- Y.-H. Chen, T.-J. Yang, J. Emer, V. Sze, “Eyeriss v2: A Flexible Accelerator for Emerging Deep Neural Networks on Mobile Devices,” IEEE Journal on Emerging and Selected Topics in Circuits and Systems (JETCAS), June 2019.
- Eyexam: https://arxiv.org/abs/1807.07928

Processing In Memory

- T.-J. Yang, V. Sze, “Design Considerations for Efficient Deep Neural Networks on Processing-in-Memory Accelerators,” IEEE International Electron Devices Meeting (IEDM), Invited Paper, December 2019.

DNN Processor Evaluation Tools

- Wu et al., “Accelergy: An Architecture-Level Energy Estimation Methodology for Accelerator Designs,” ICCAD 2019, http://accelergy.mit.edu
- Parashar et al., “Timeloop: A Systematic Approach to DNN Accelerator Evaluation,” ISPASS 2019