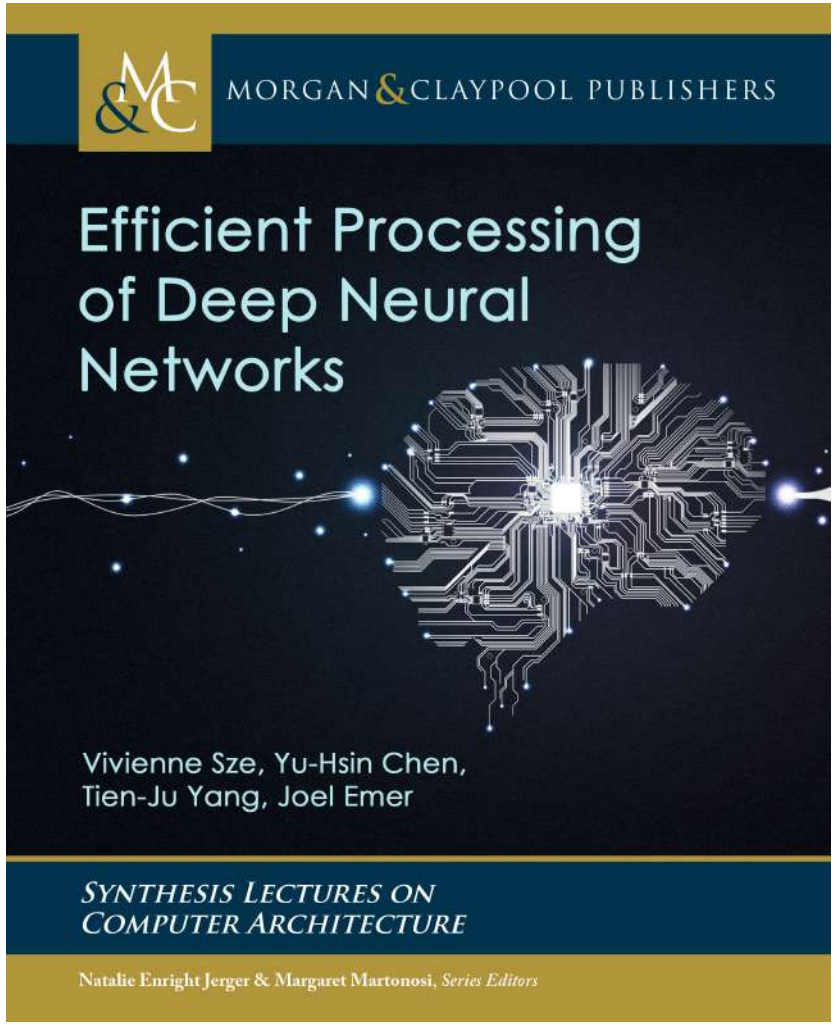
How to Evaluate Efficient Deep Neural Network Approaches

Vivienne Sze (@eems_mit)
Massachusetts Institute of Technology

In collaboration with Yu-Hsin Chen, Joel Emer, Yannan Wu, Tien-Ju Yang, Google Mobile Vision Team

Slides available at
https://tinyurl.com/SzeMITDL2020

Book on Efficient Processing of DNNs



Part I Understanding Deep Neural Networks

Introduction

Overview of Deep Neural Networks

Part II Design of Hardware for Processing DNNs

Key Metrics and Design Objectives

Kernel Computation

Designing DNN Accelerators

Operation Mapping on Specialized Hardware

Part III Co-Design of DNN Hardware and Algorithms

Reducing Precision

Exploiting Sparsity

Designing Efficient DNN Models

Advanced Technologies

https://tinyurl.com/EfficientDNNBook

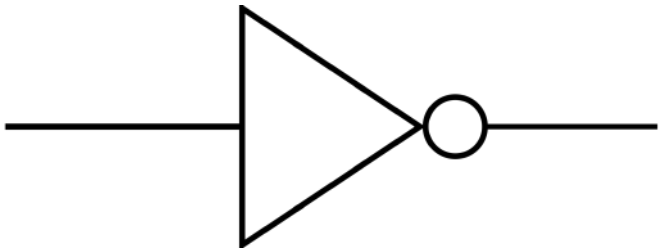
How to Evaluate these DNN Approaches?

- Many Deep Neural Networks (DNN) accelerators and approaches for efficient DNN processing.
  - **Too many to cover!**

- We will focus on how to **evaluate** approaches for efficient processing of DNNs
  - Approaches include the design of DNN accelerators and DNN models
  - What are the **key metrics** that should be measured and compared?

TOPS or TOPS/W?

- TOPS = tera (10^{12}) operations per second
- TOPS/Watt or TOPS/Watt commonly reported in hardware literature to show efficiency of design
- However, does not provide sufficient insights on hardware capabilities and limitations (especially if based on peak throughput/performance)

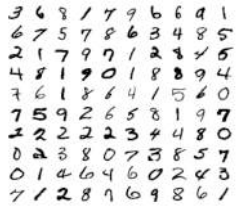


Example: high TOPS per watt can be achieved with inverter (ring oscillator)

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



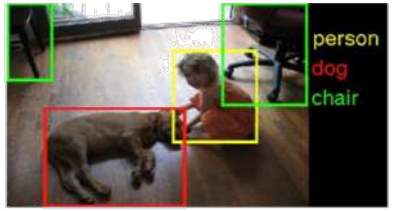
CIFAR-10

ImageNet

Embedded Device

Data Center

Computer Vision



Speech Recognition

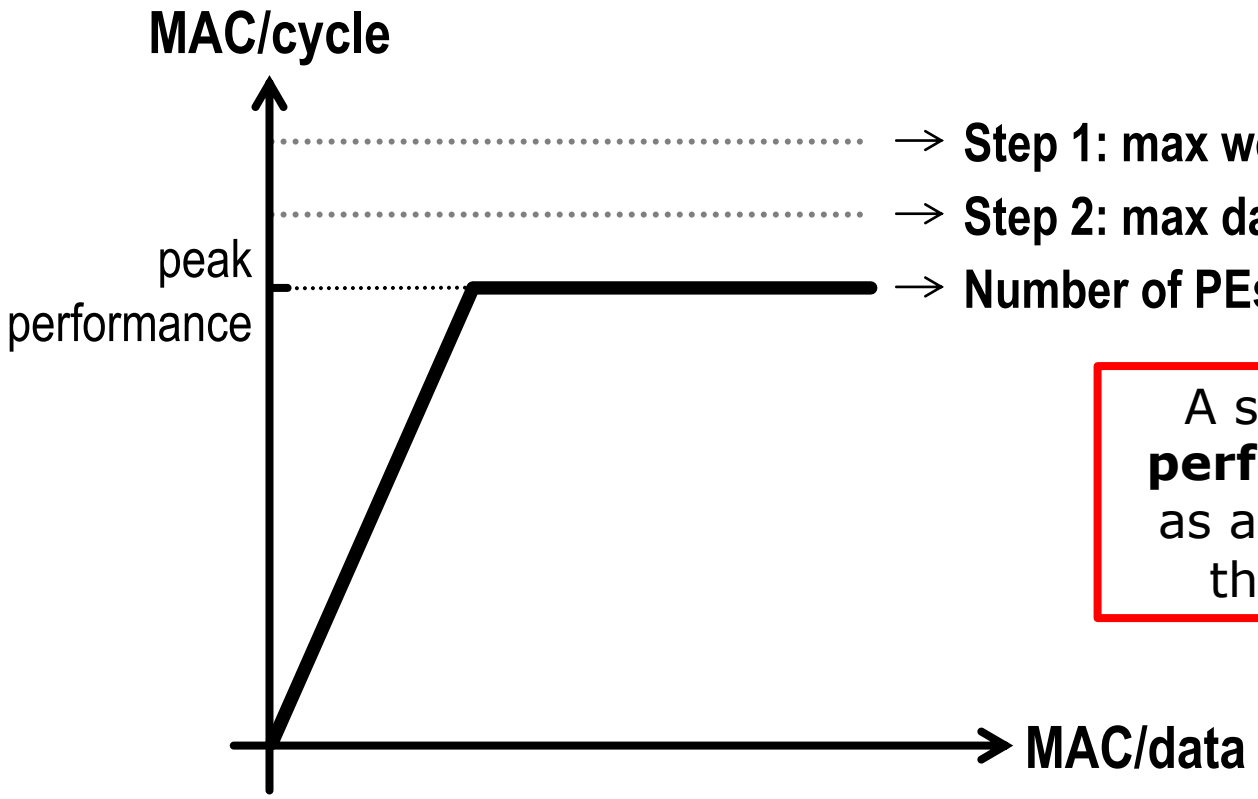
[**Sze**, *CICC* 2017]

Key Design Objectives of DNN Accelerators

- **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)**

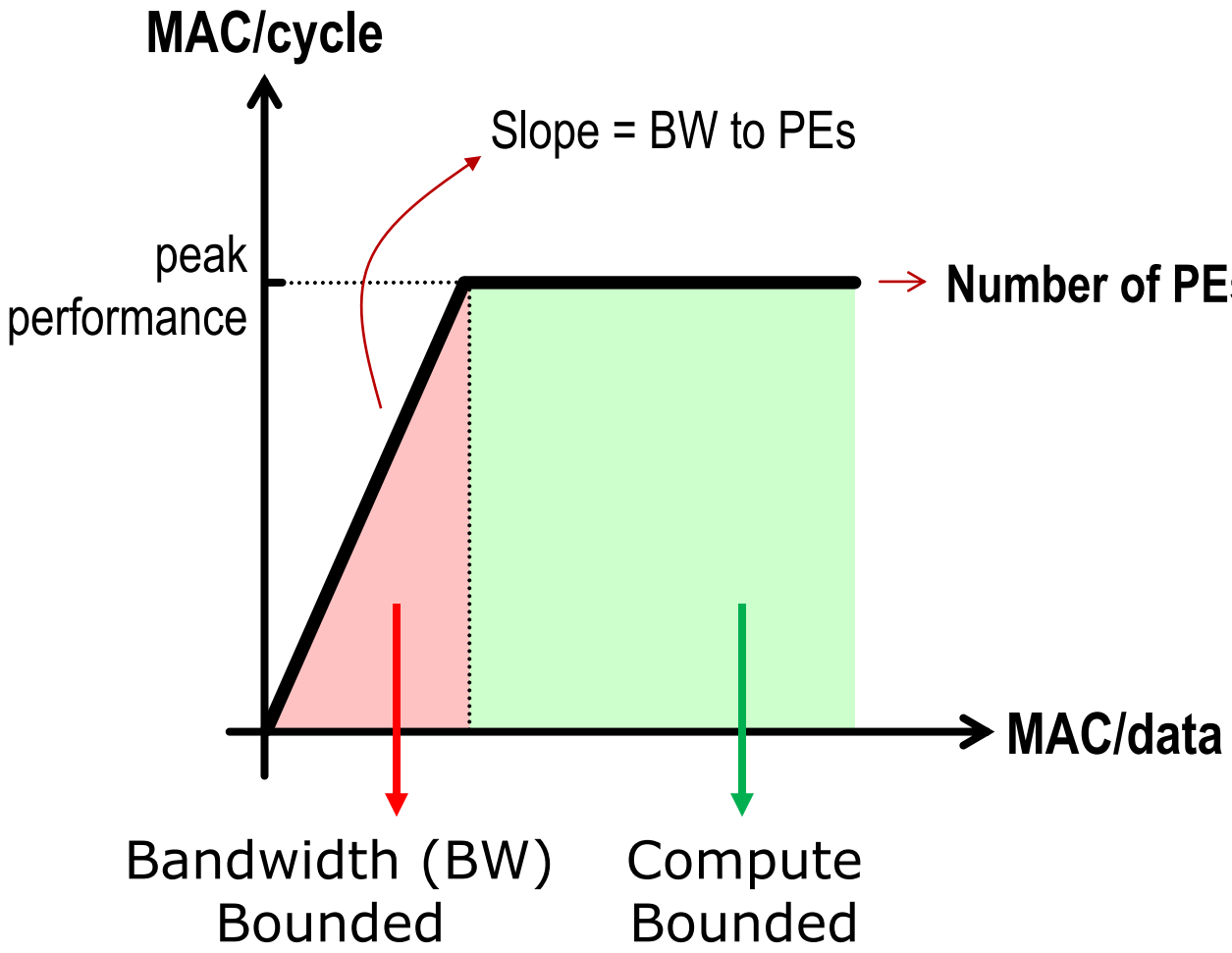
→ **Step 2: max dataflow parallelism**

→ **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]

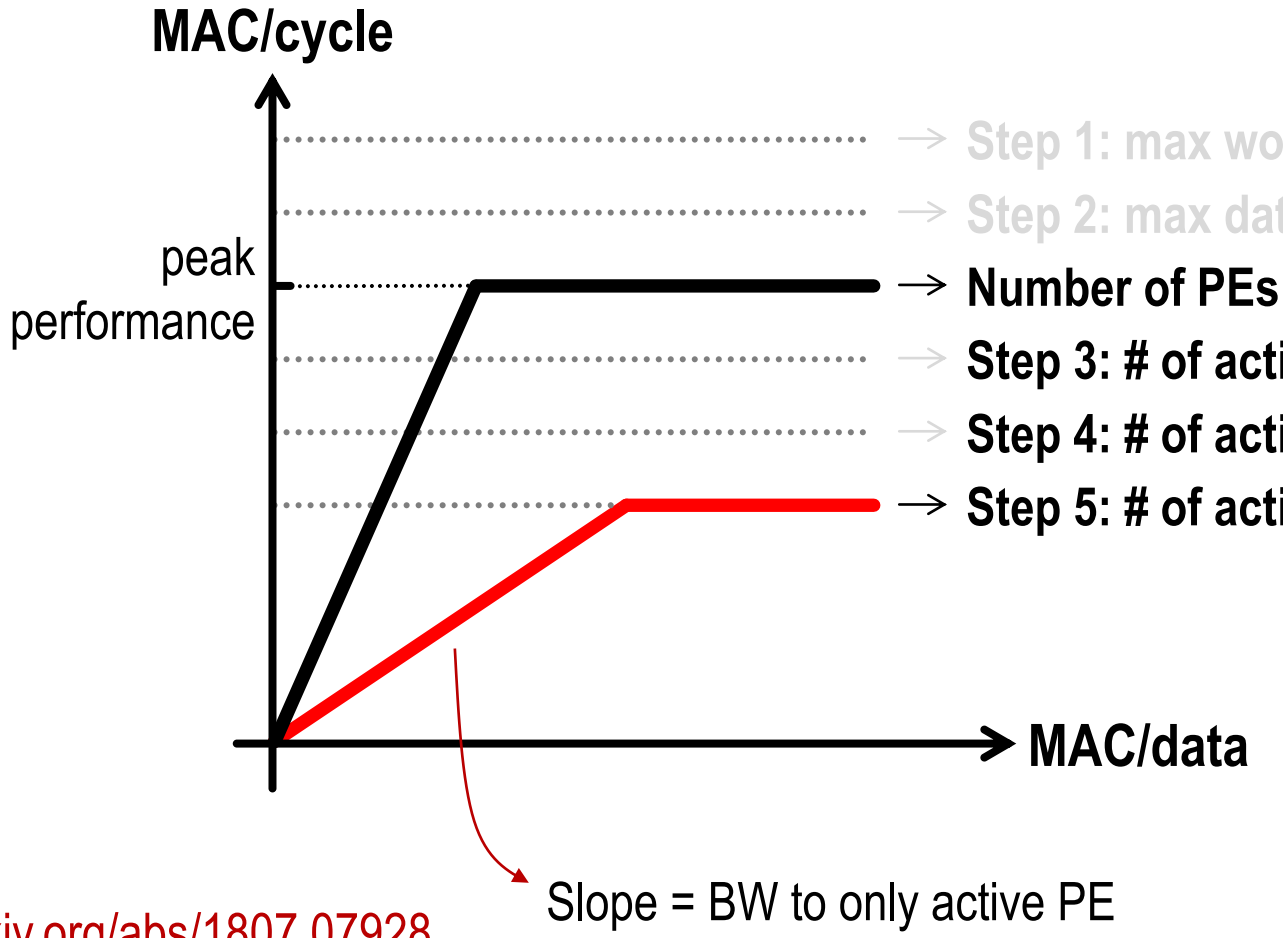
Eyexam: Performance Evaluation Framework



Based on Roofline Model

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

Eyexam: Performance Evaluation Framework



→ Step 1: max workload parallelism

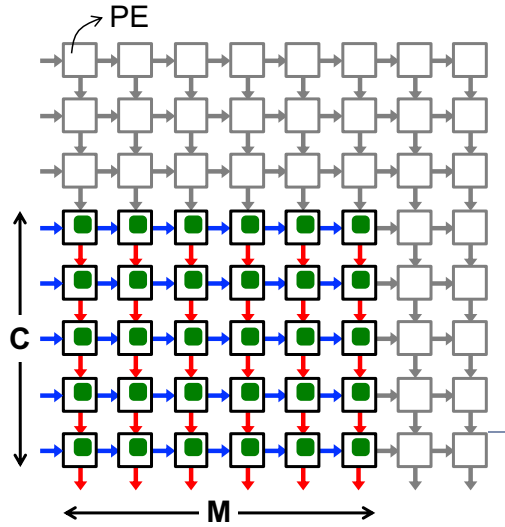
→ Step 2: max dataflow parallelism

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

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

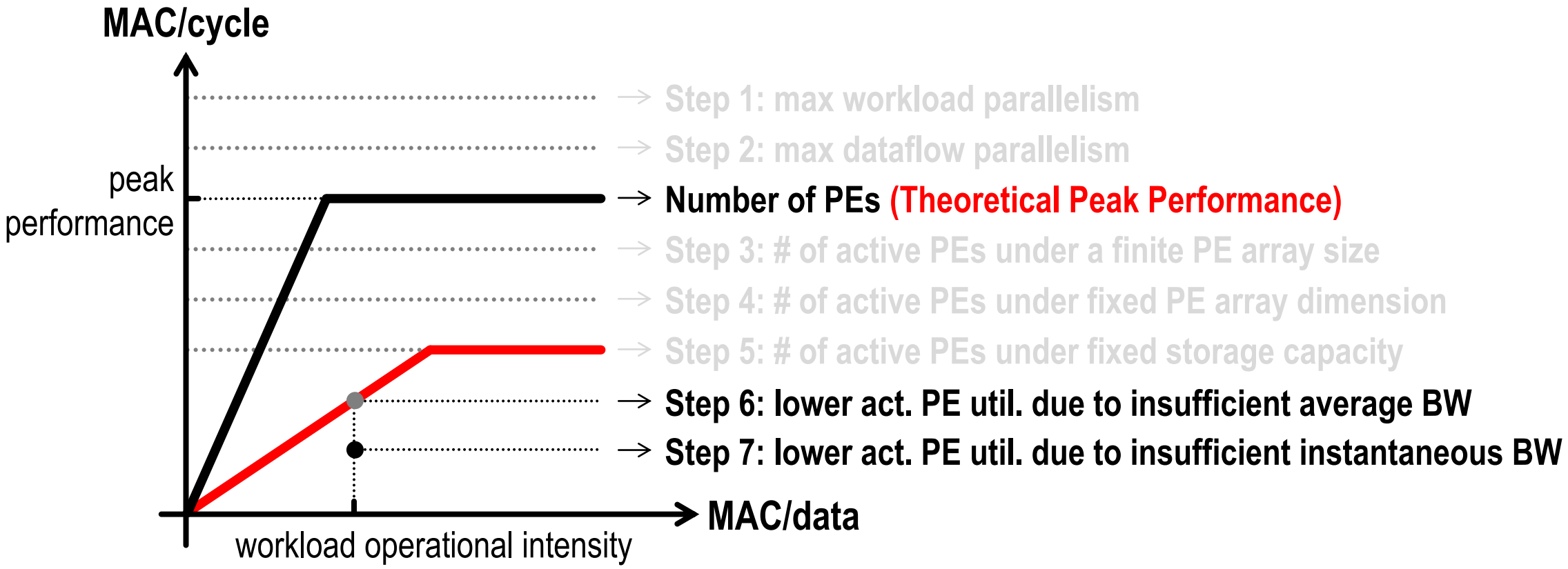
→ **Step 4: # of active PEs under fixed PE array dimension**

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



https://arxiv.org/abs/1807.07928

Eyexam: Performance Evaluation Framework



https://arxiv.org/abs/1807.07928

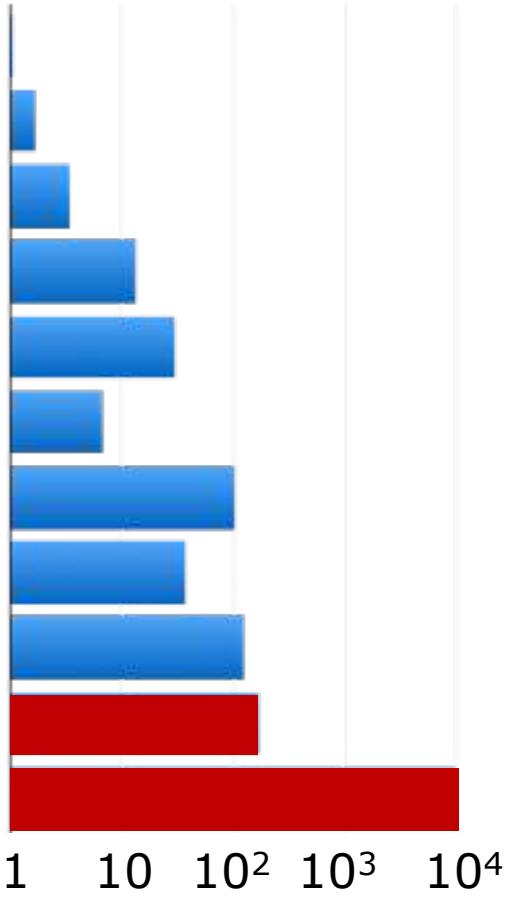
Key Design Objectives of DNN Accelerators

- **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)

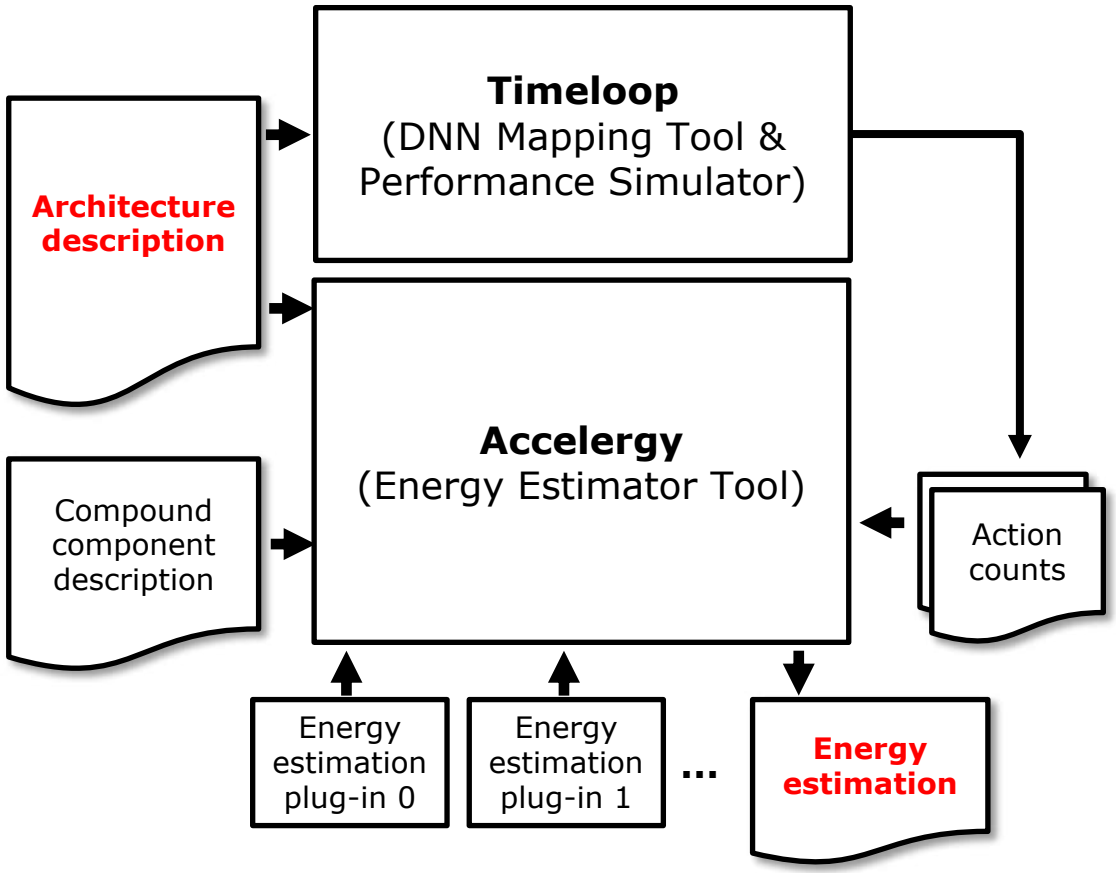
Operation:	Energy (pJ)
8b Add	0.03
16b Add	0.05
32b Add	0.1
16b FP Add	0.4
32b FP Add	0.9
8b Multiply	0.2
32b Multiply	3.1
16b FP Multiply	1.1
32b FP Multiply	3.7
32b SRAM Read (8KB)	5
32b DRAM Read	640

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



DNN Processor Evaluation Tools

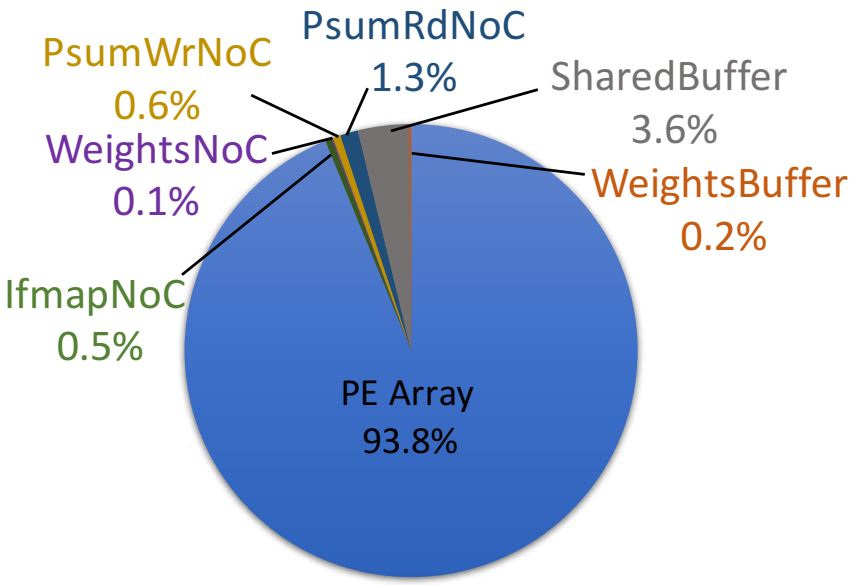
- Require systematic way to
  - Evaluate and compare wide range of DNN processor designs
  - Rapidly explore design space
- **Accelergy** [**Wu**, *ICCAD* 2019]
  - Early stage energy estimation tool at the architecture level
    - Estimate energy consumption based on architecture level components (e.g., # 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



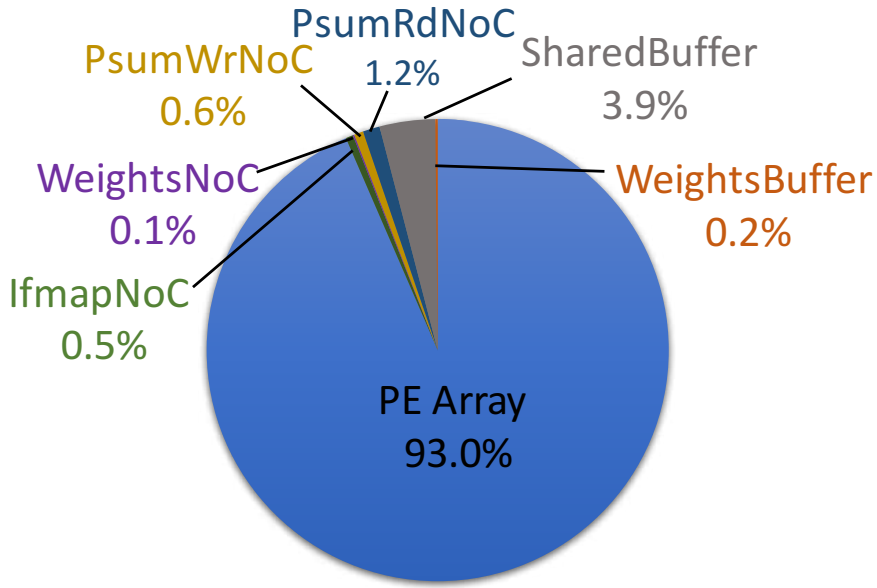
Open-source code available at:
http://accelergy.mit.edu

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

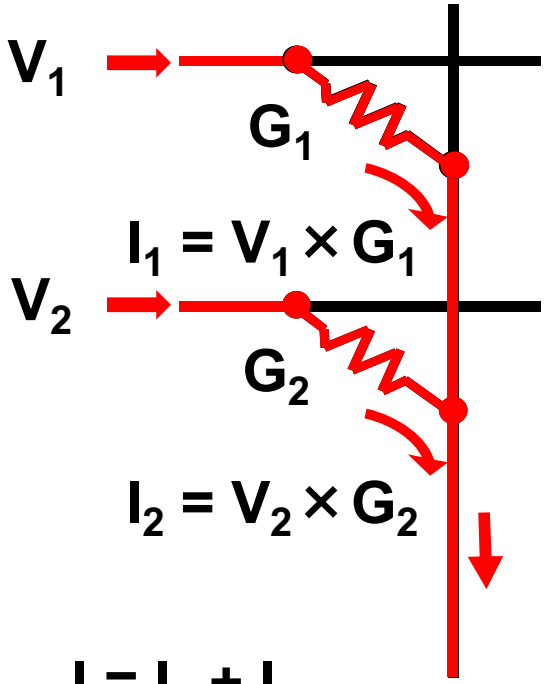


Accelergy Energy Breakdown

Open-source code available at: http://accelergy.mit.edu [**Wu**, *ICCAD* 2019]

Performing MAC with Memory Storage Element

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



Psum is output current

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

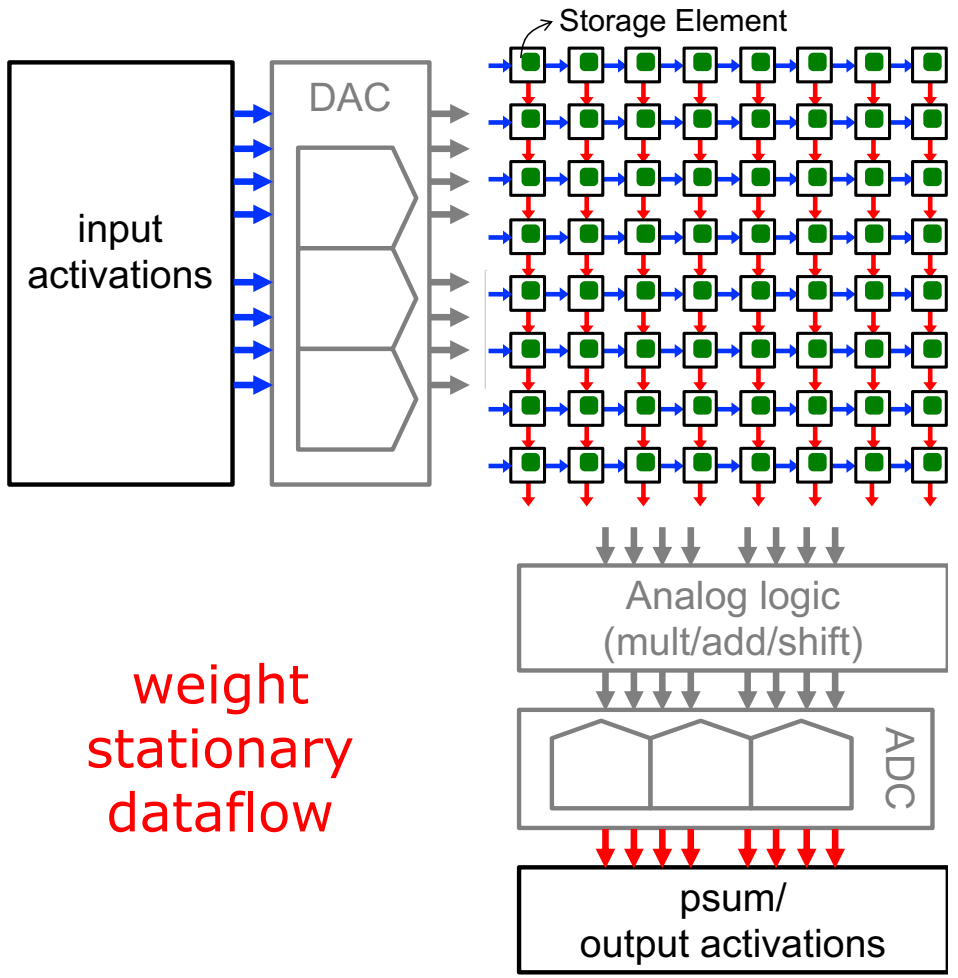
Image Source: [**Shafiee**, *ISCA* 2016]

- **Analog Compute**
  - Activations, weights and/or partial sums are encoded with analog voltage, current, or resistance
  - **Increased sensitivity** to circuit non-idealities: non-linearities, process, voltage, and temperature variations
  - Require A/D and D/A peripheral circuits to interface with digital domain
- **Multiplication**
  - eNVM (RRAM, STT-RAM, PCM) use **resistive device**
  - Flash and SRAM use **transistor** (I-V curve) or **local cap**
- **Accumulation**
  - Current summing
  - Charge sharing

Processing In Memory (PIM*)

* a.k.a. In-Memory Computing (IMC)

- Implement as **matrix-vector multiply**
  - Typically, matrix composed of stored weights and vector composed of input activations
- **Reduce weight data movement by moving compute into the memory**
  - Perform MAC with storage element or in peripheral circuits
  - Read out partial sums rather than weights → fewer accesses through peripheral circuits
- **Increase weight bandwidth**
  - Multiple weights accessed in parallel to keep MACs busy (high utilization)
- **Increase amount of parallel MACs**
  - Storage element can be higher area density than digital MAC
  - Reduce routing capacitance



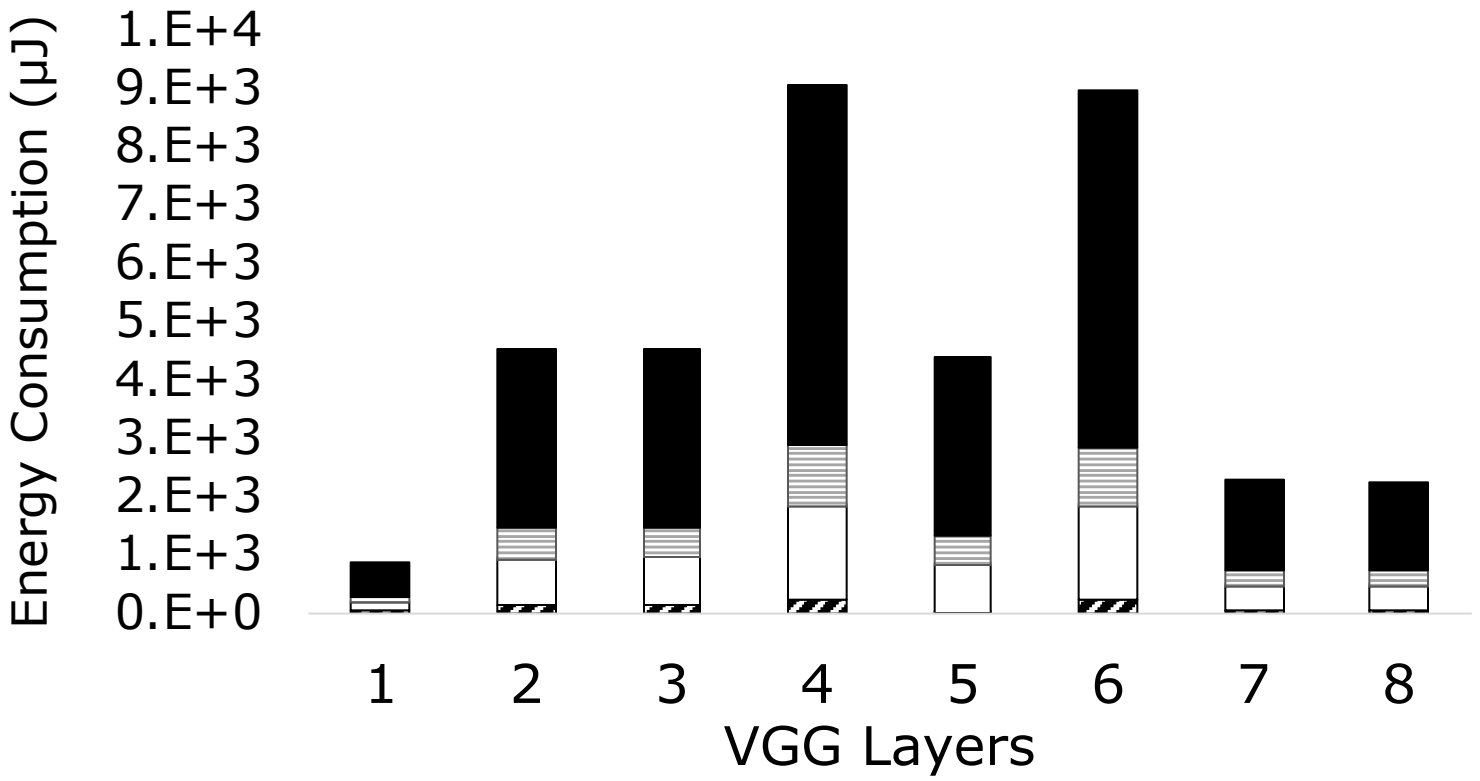
weight stationary dataflow

eNVM:[**Yu**, *PIEEE* 2018], SRAM:[**Verma**, *SSCS* 2019]

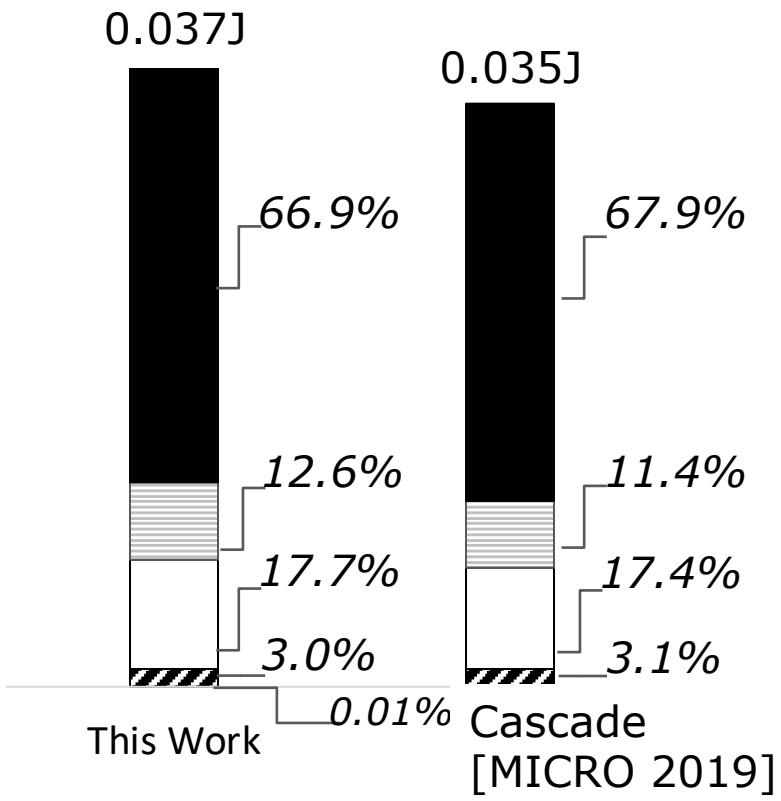
Accelergy for PIM

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

Energy breakdown across layers



Achieves ~95% accuracy



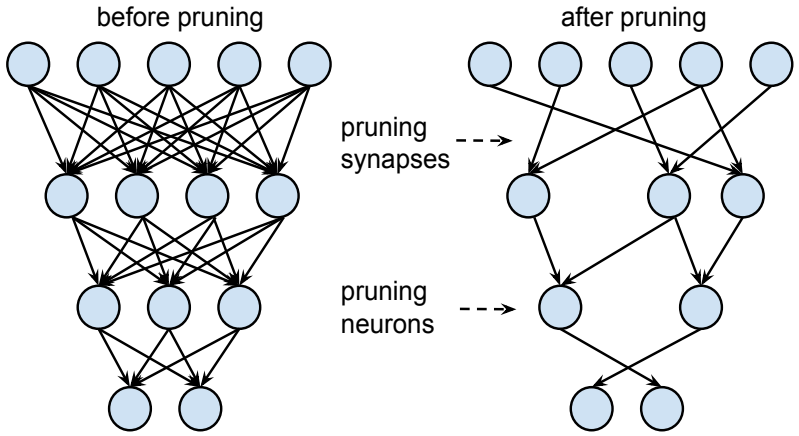
[**Wu**, *ISPASS* 2020]

Key Design Objectives of DNN Accelerators

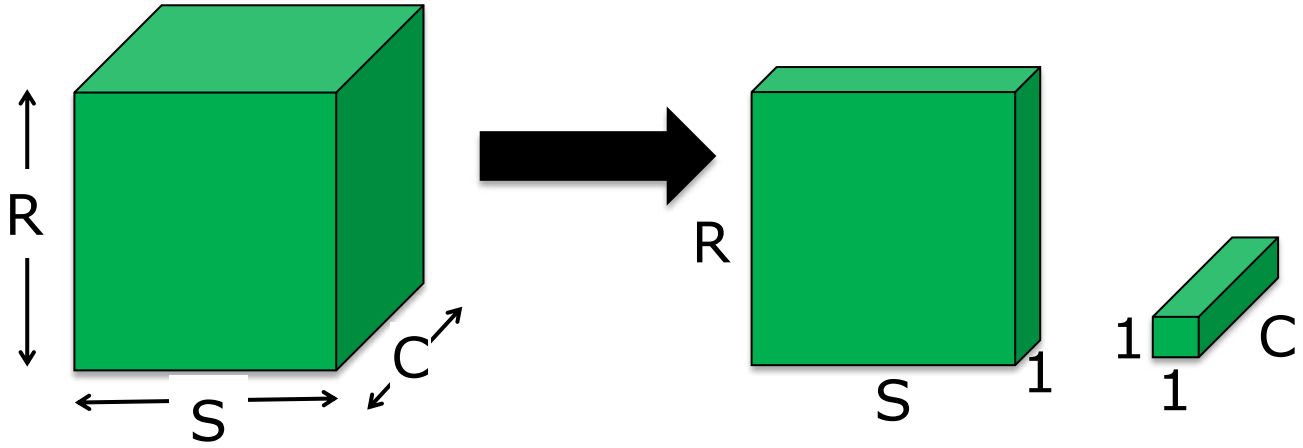
- **Flexibility**
  - Reduce overhead of supporting flexibility
  - Maintain efficiency across wide range of DNN models
    - Different layer shapes impact the amount of
      - Required storage and compute
      - Available data reuse that can be exploited
    - Different precision across layers & data 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.)

Many Efficient DNN Design Approaches

Network Pruning



Efficient Network Architectures



Reduce Precision

32-bit float

8-bit fixed

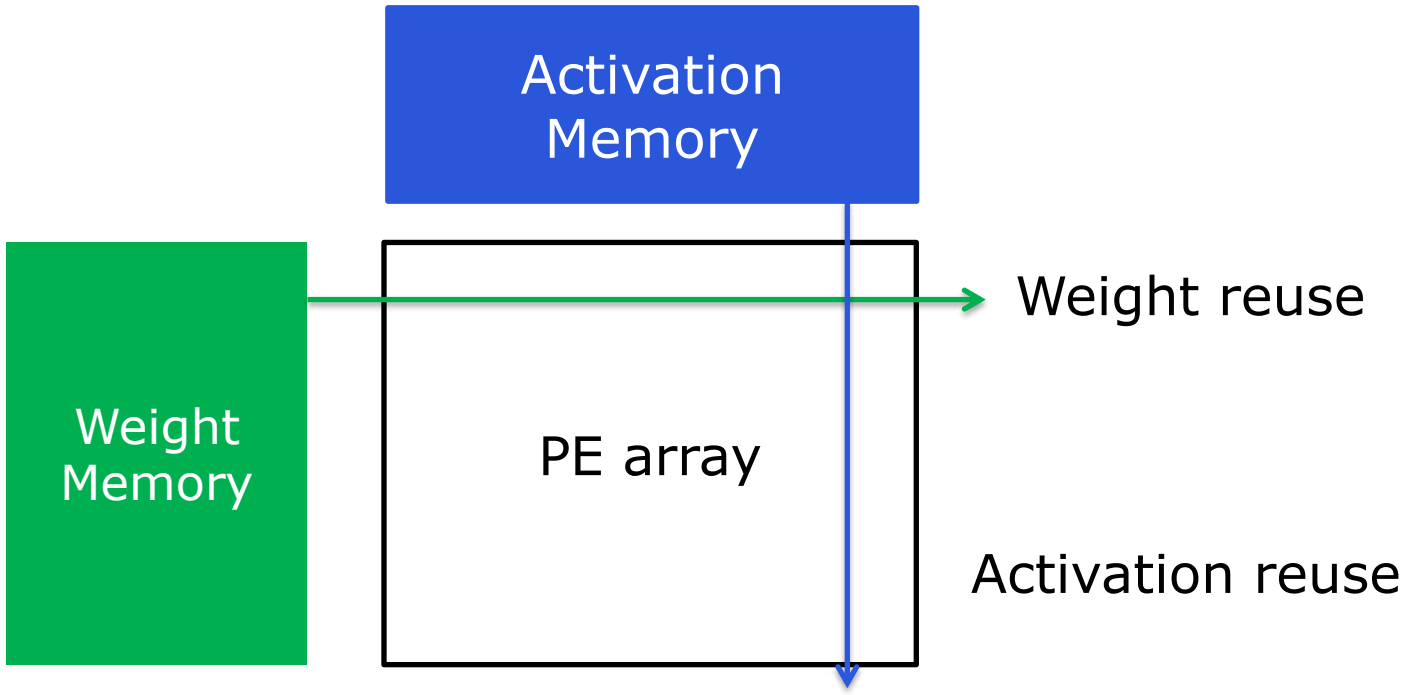
Binary

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

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

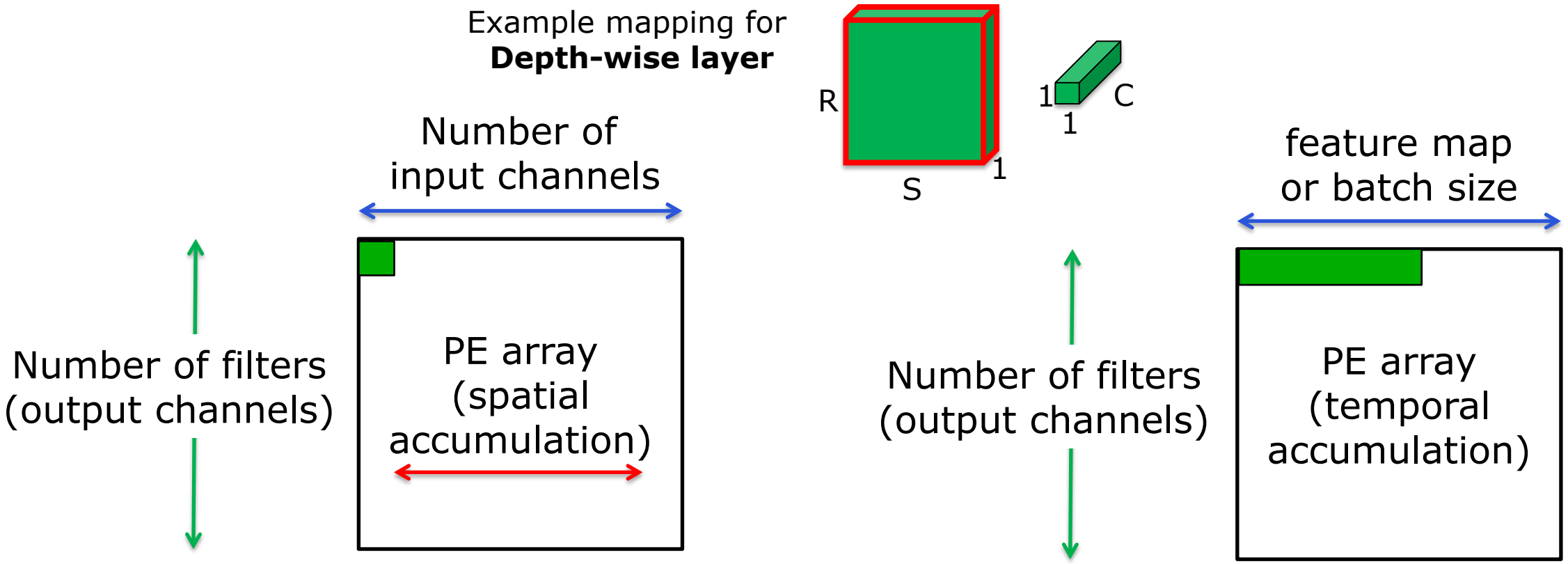
Limitations of Existing DNN Accelerators

- 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 Accelerators

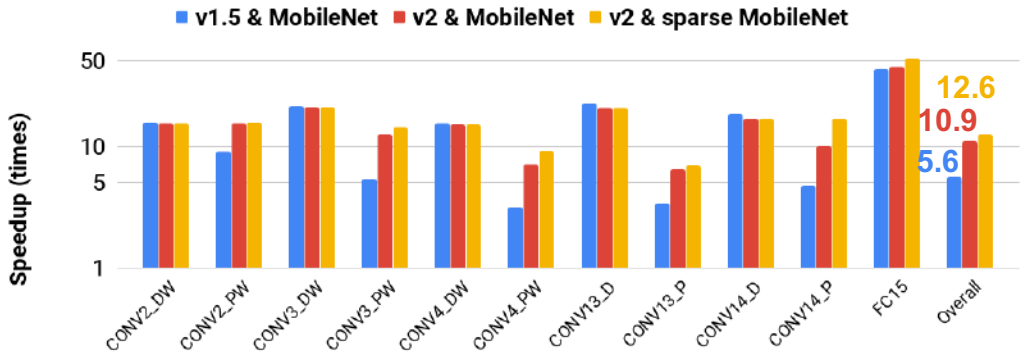
- Reuse depends on # of channels, feature map/batch size
  - Not efficient across all DNN models (e.g., efficient network architectures)



Eyeriss v2: Balancing Flexibility and Efficiency

- Uses a flexible hierarchical mesh on-chip network to efficiently support
  - Wide range of filter shapes
  - Different layers
  - Wide range of sparsity
- Scalable architecture

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



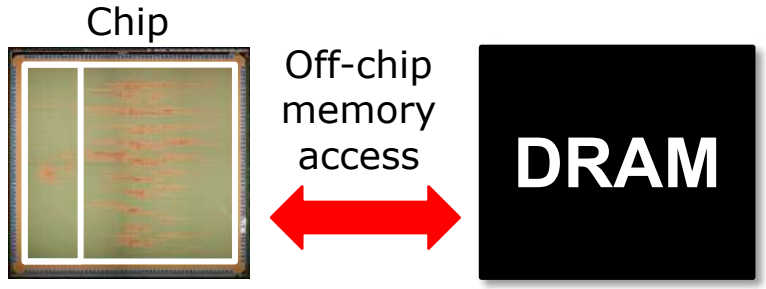
Speed up over Eyeriss v1 scales with number of PEs

# 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]

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



[**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

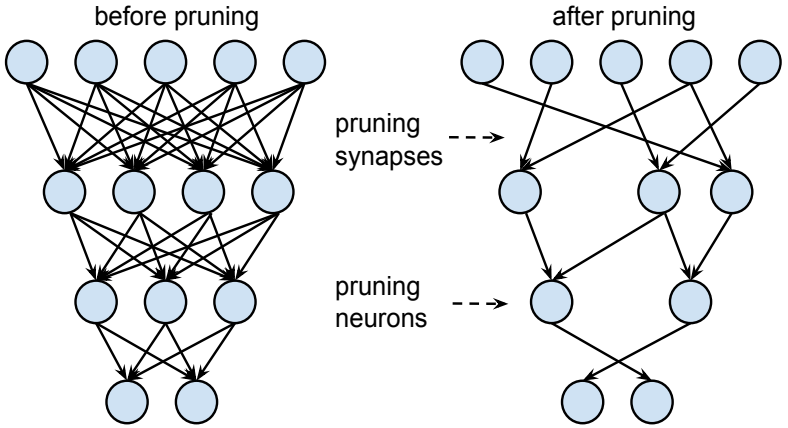
- **Does hardware cost exceed benefits?**
  - Need extra hardware to support variable precision and shapes or to identify sparsity
  - Granularity impacts hardware overhead as well as accuracy

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

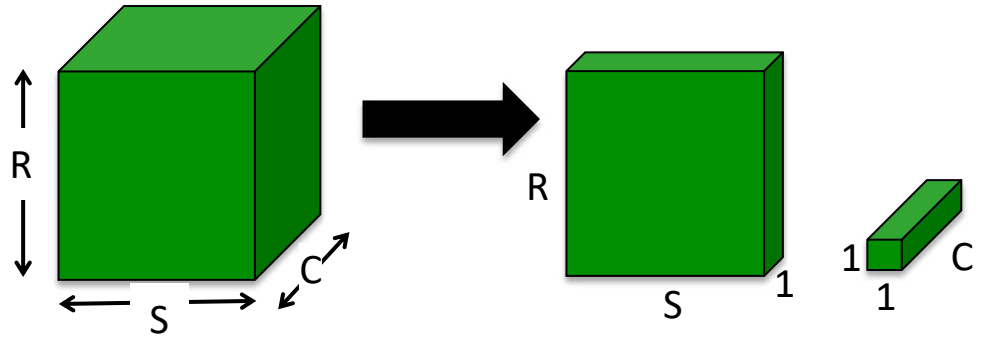
Design of Efficient DNN Algorithms

Popular efficient DNN algorithm approaches

Network Pruning



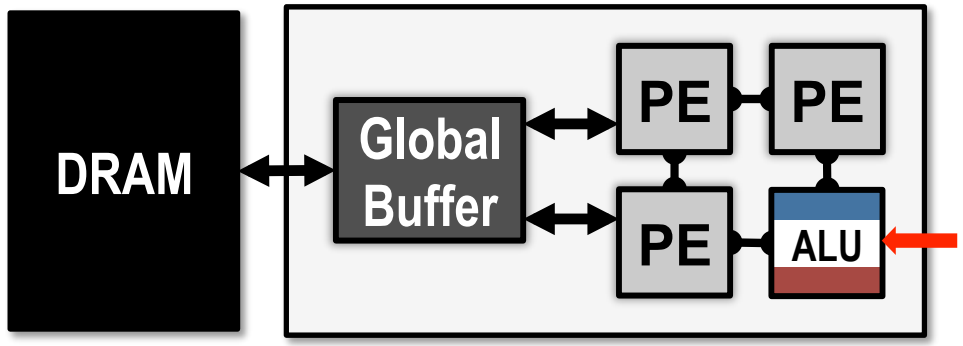
Efficient Network Architectures



… also reduced precision

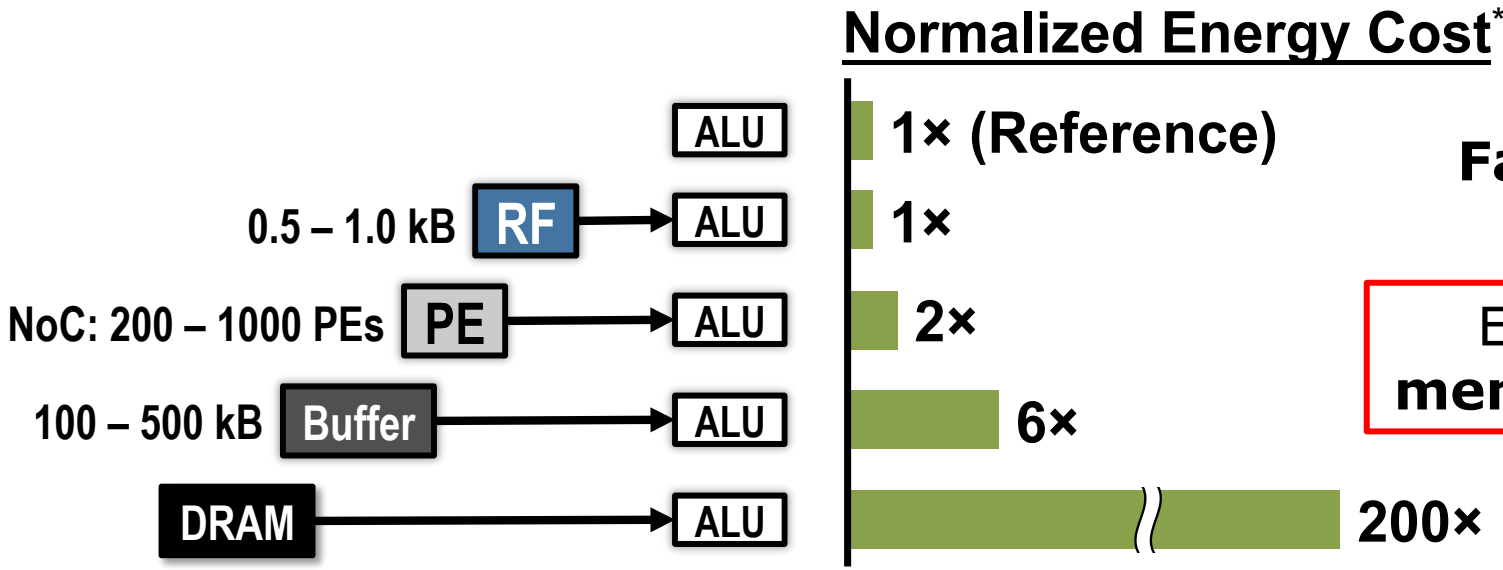
- Focus on reducing number of MACs and weights
- **Does it translate to energy savings and reduced latency?**

Data Movement is Expensive



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

fetch data to run a MAC here

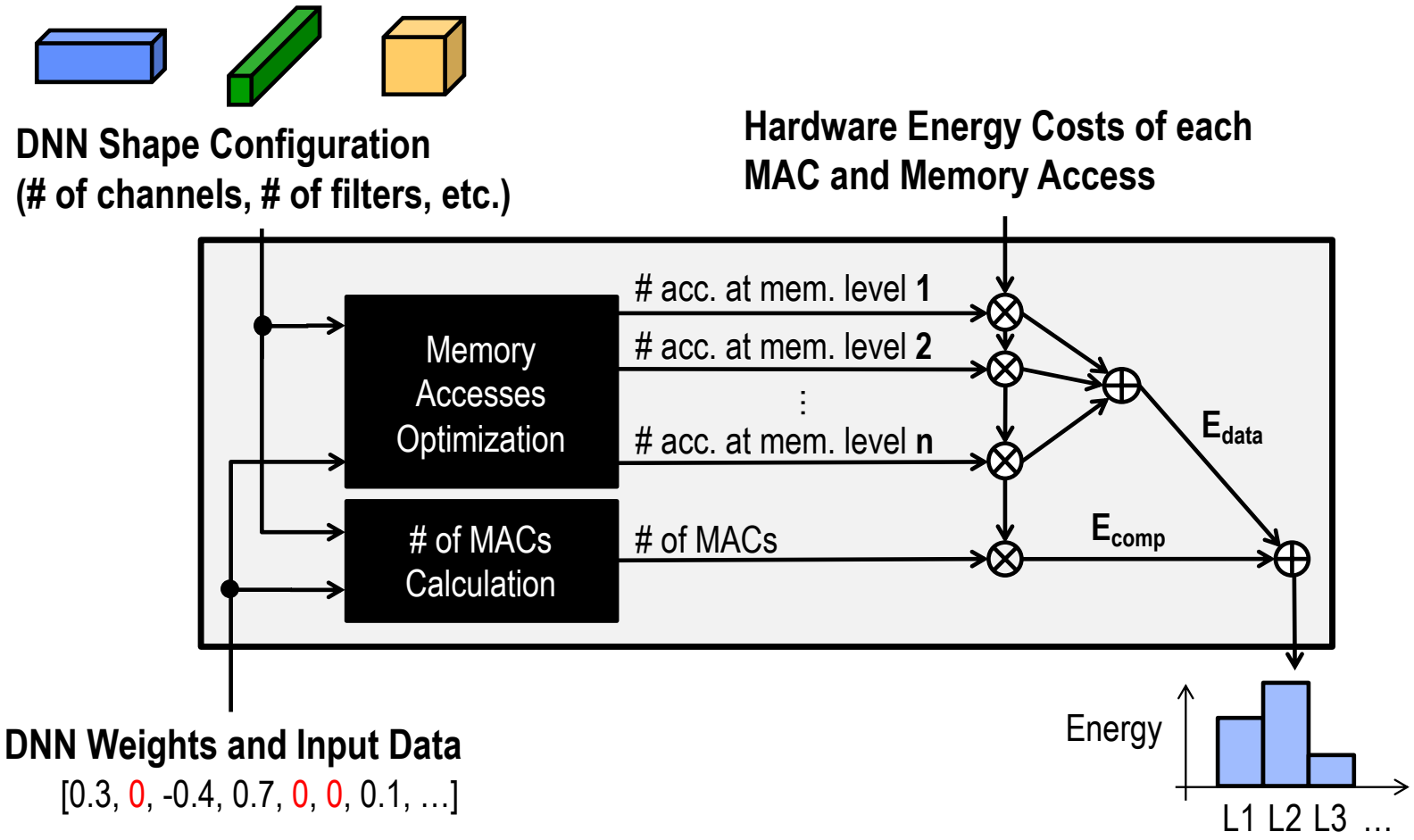


Farther and **larger** memories consume more power

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

* measured from a commercial 65nm process

Energy-Evaluation Methodology

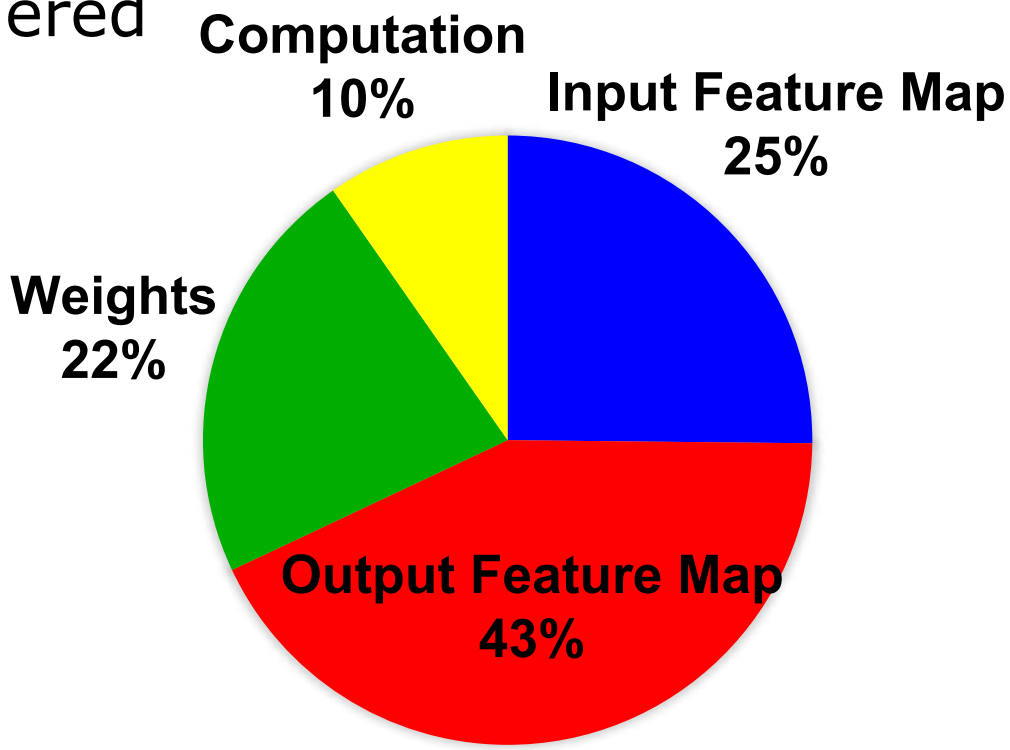


Tool available at https://energyestimation.mit.edu/ [**Yang**, *CVPR* 2017]

Key Observations

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

Energy Consumption of GoogLeNet



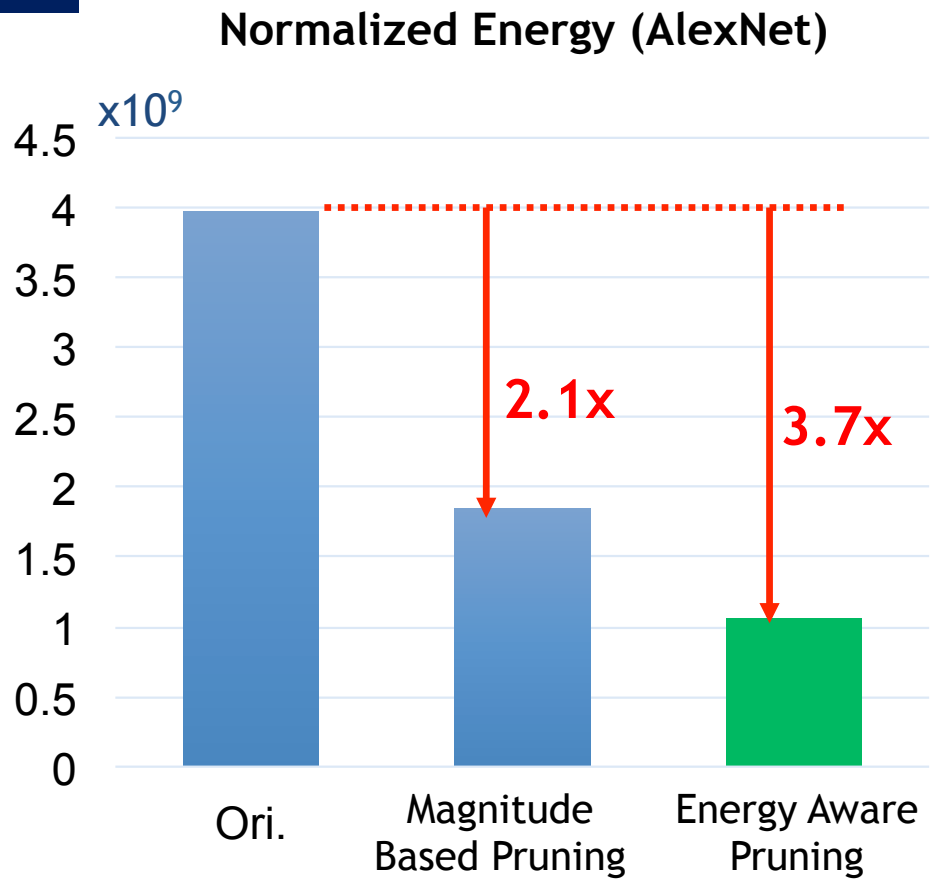
Tool available at https://energyestimation.mit.edu/ [**Yang**, *CVPR* 2017]

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**

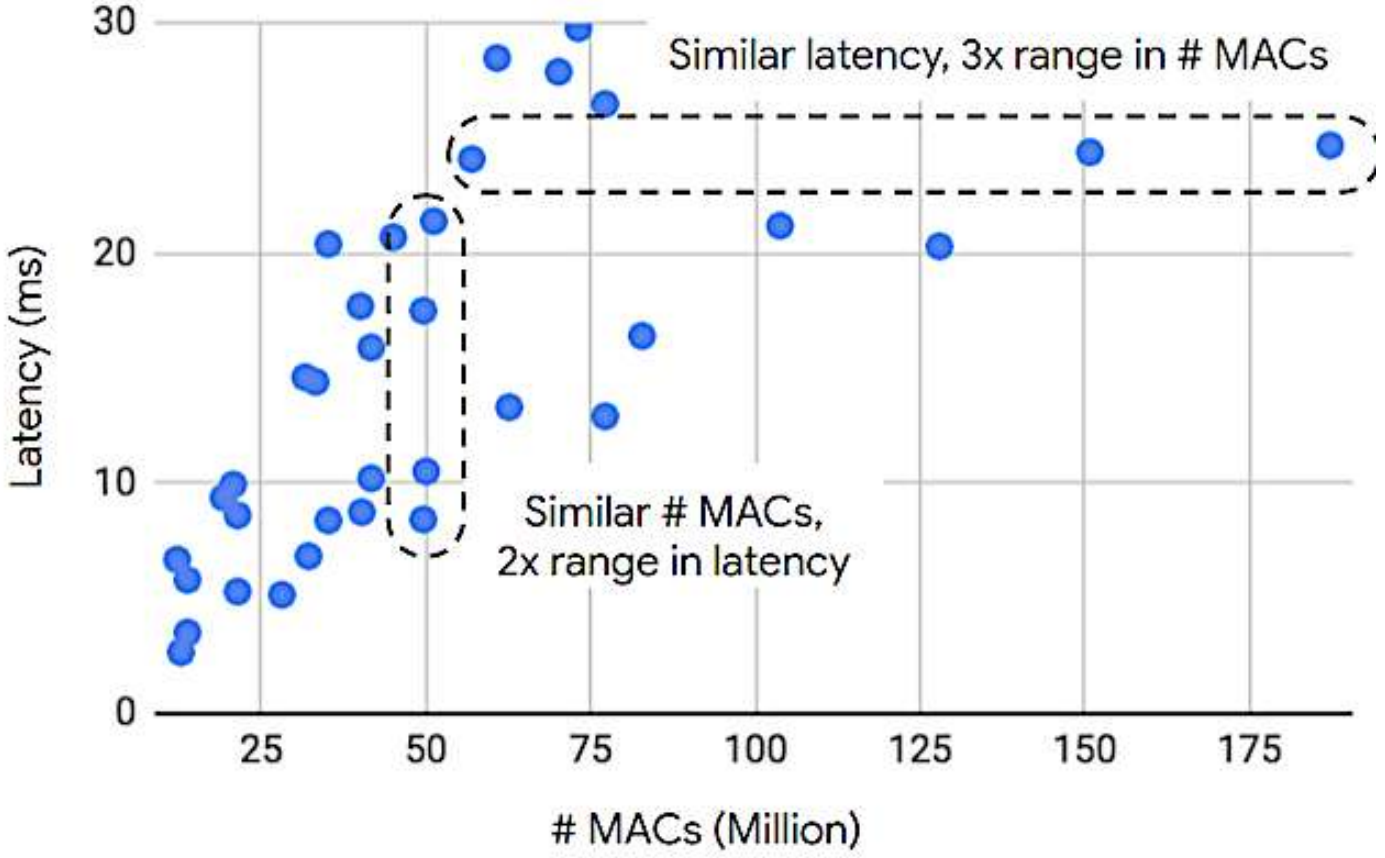
[**Yang**, *CVPR* 2017]



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

of Operations versus Latency

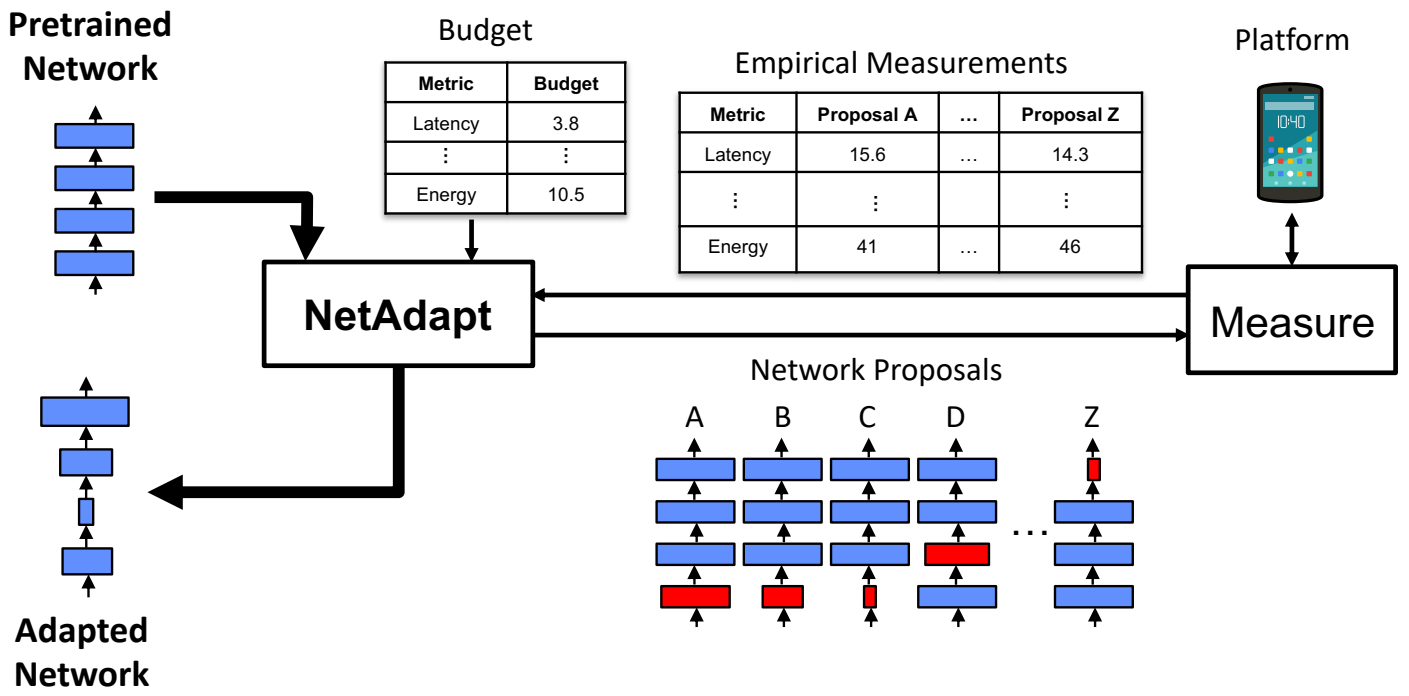
of operations (MACs) does not approximate latency well



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

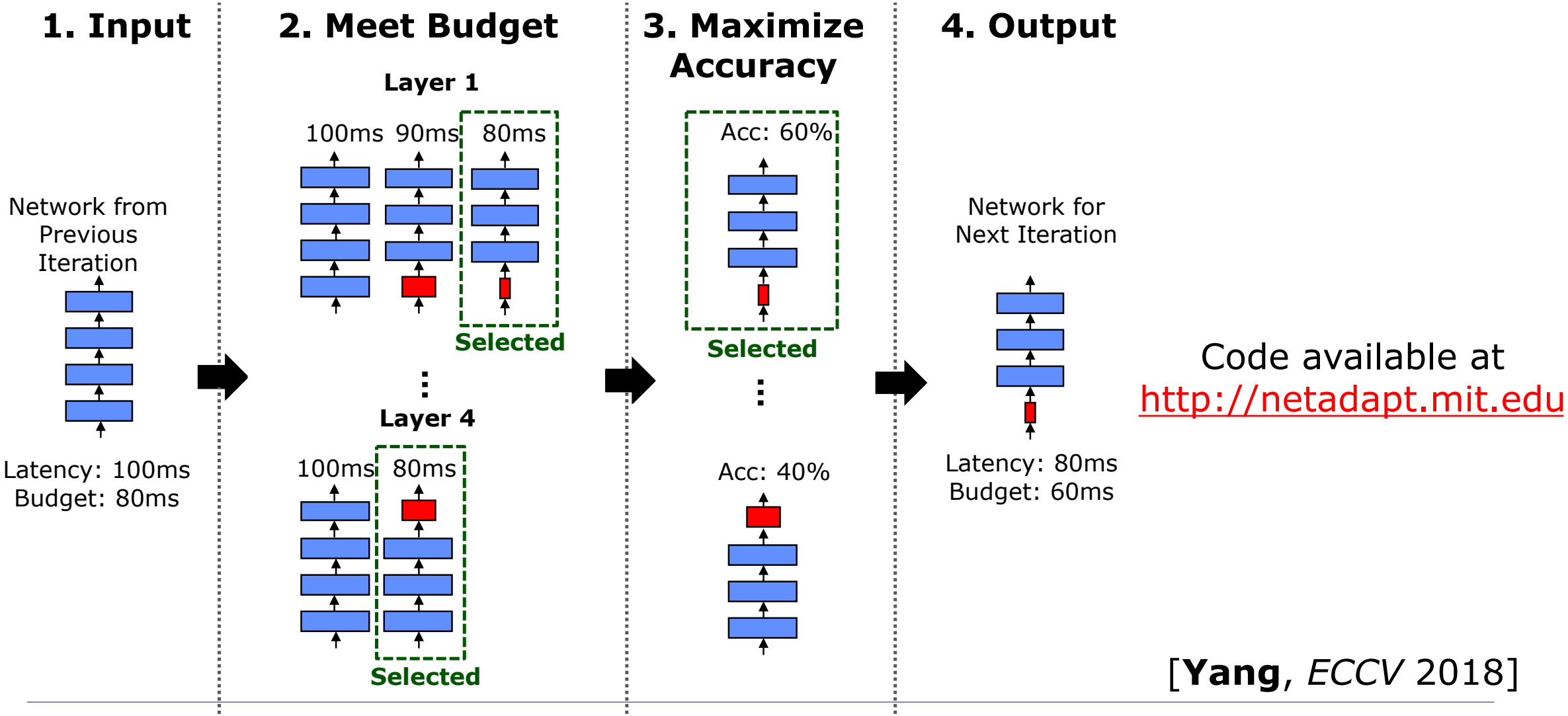


In collaboration with Google's Mobile Vision Team

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

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

NetAdapt: Simplified Example of One Iteration

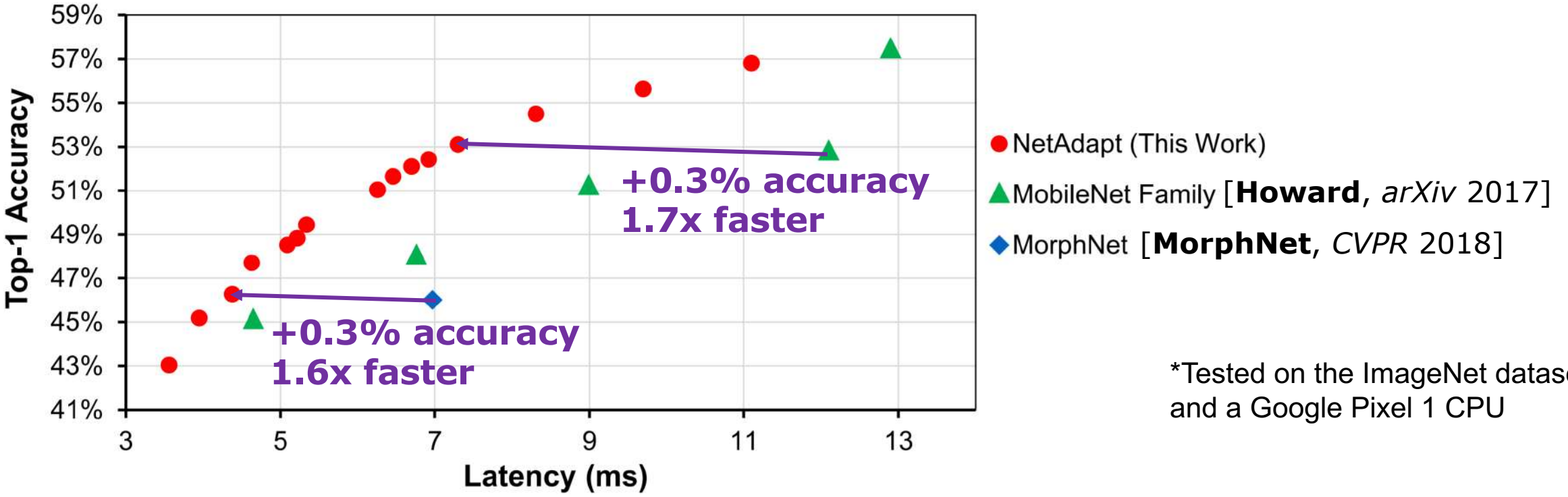


Code available at
http://netadapt.mit.edu

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

Improved Latency vs. Accuracy Tradeoff

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



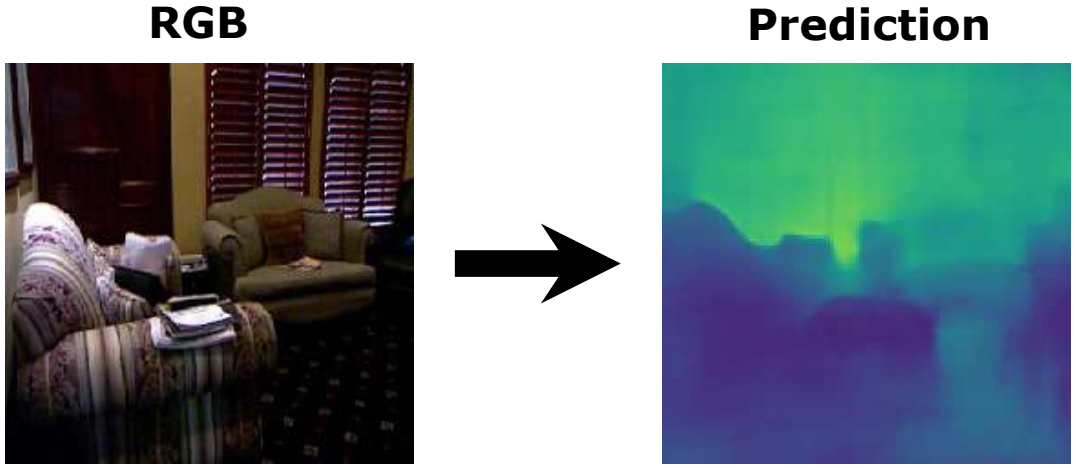
*Tested on the ImageNet dataset and a Google Pixel 1 CPU

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

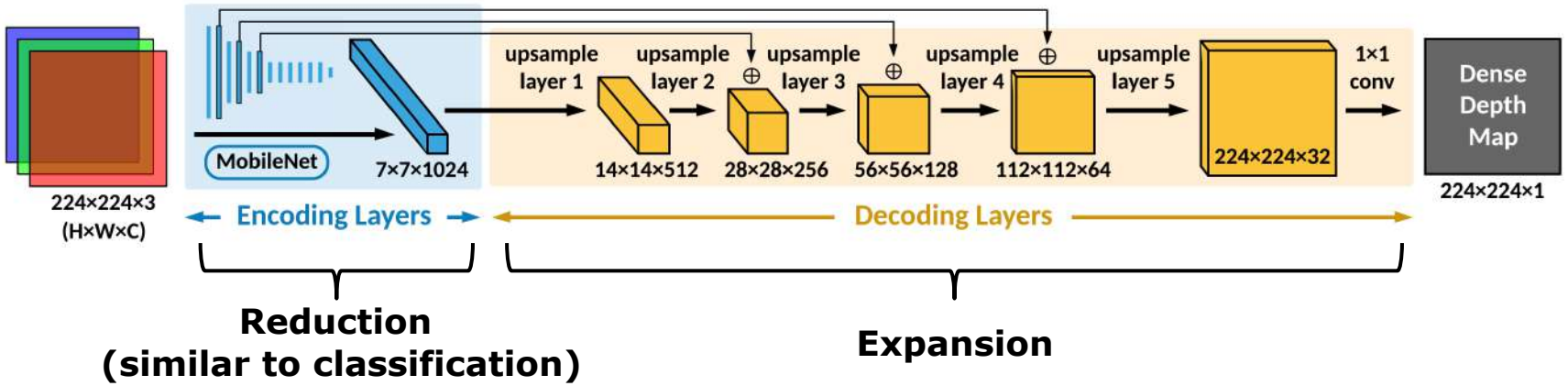
[**Yang**, *ECCV* 2018]

FastDepth: Fast Monocular Depth Estimation

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

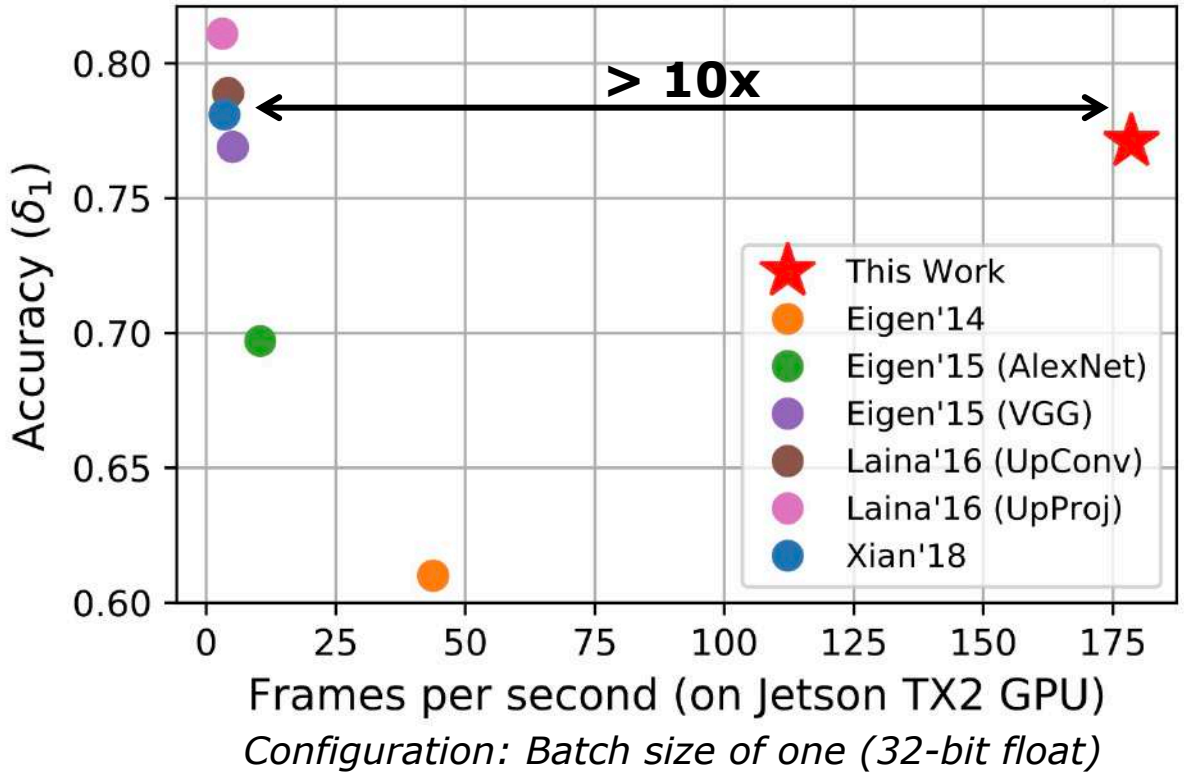


Auto Encoder DNN Architecture (Dense Output)



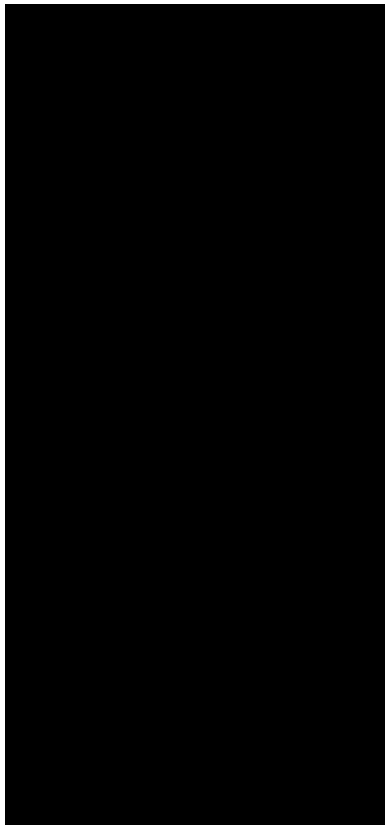
FastDepth: Fast Monocular Depth Estimation

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



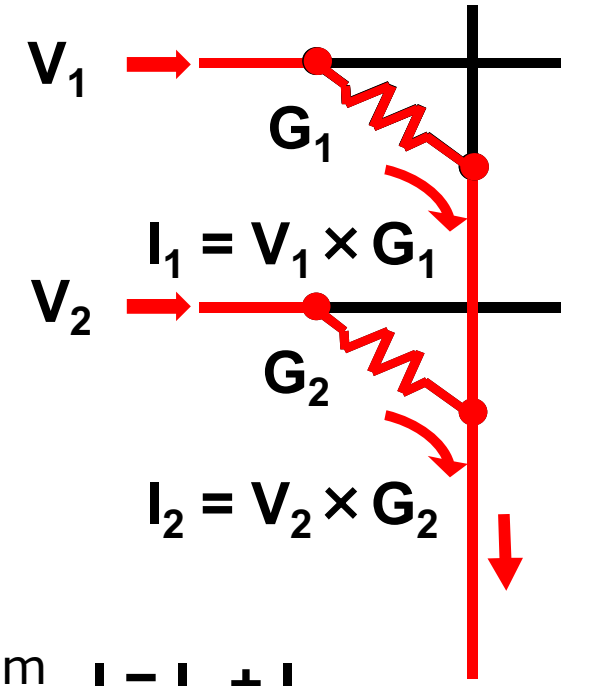
~40fps on an iPhone

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

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)



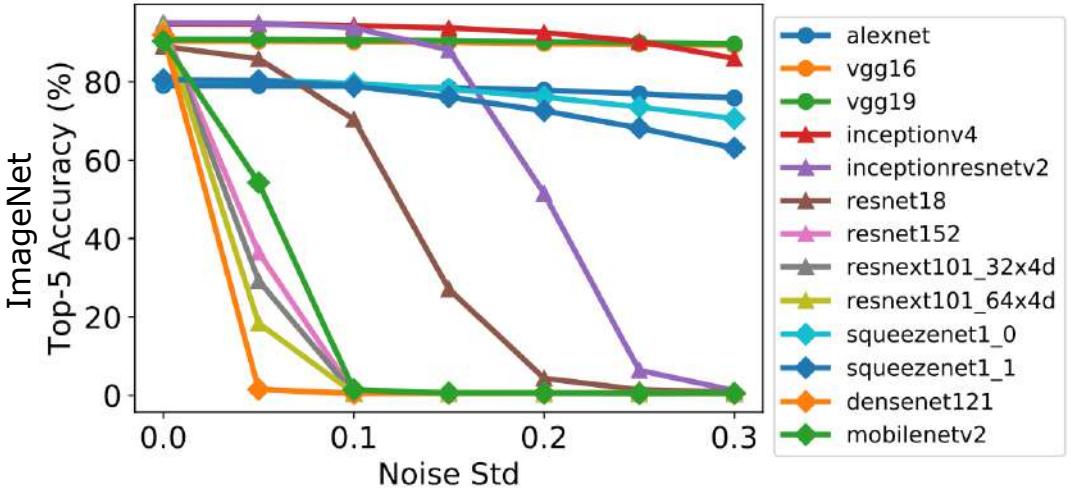
Partial sum is output current

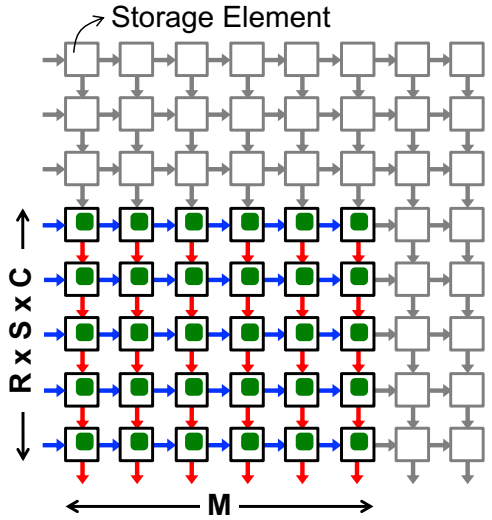
$$I = I_1 + I_2 = V_1 \times G_1 + V_2 \times G_2$$

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]

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:

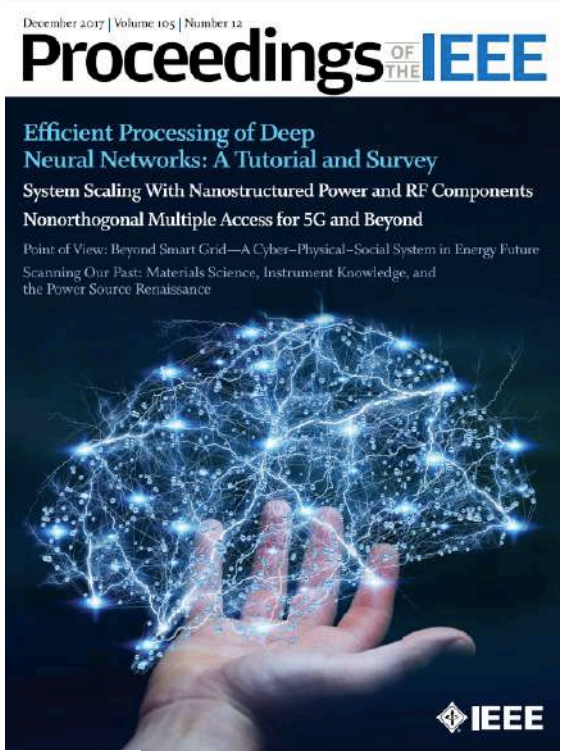
3M, AFOSR Air Force Office of Scientific Research, Analog Devices, Broadcom, DARPA, Google, intel, IBM, NSF, NVIDIA, Qualcomm, Samsung, SRC, Texas Instruments, tsmc

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



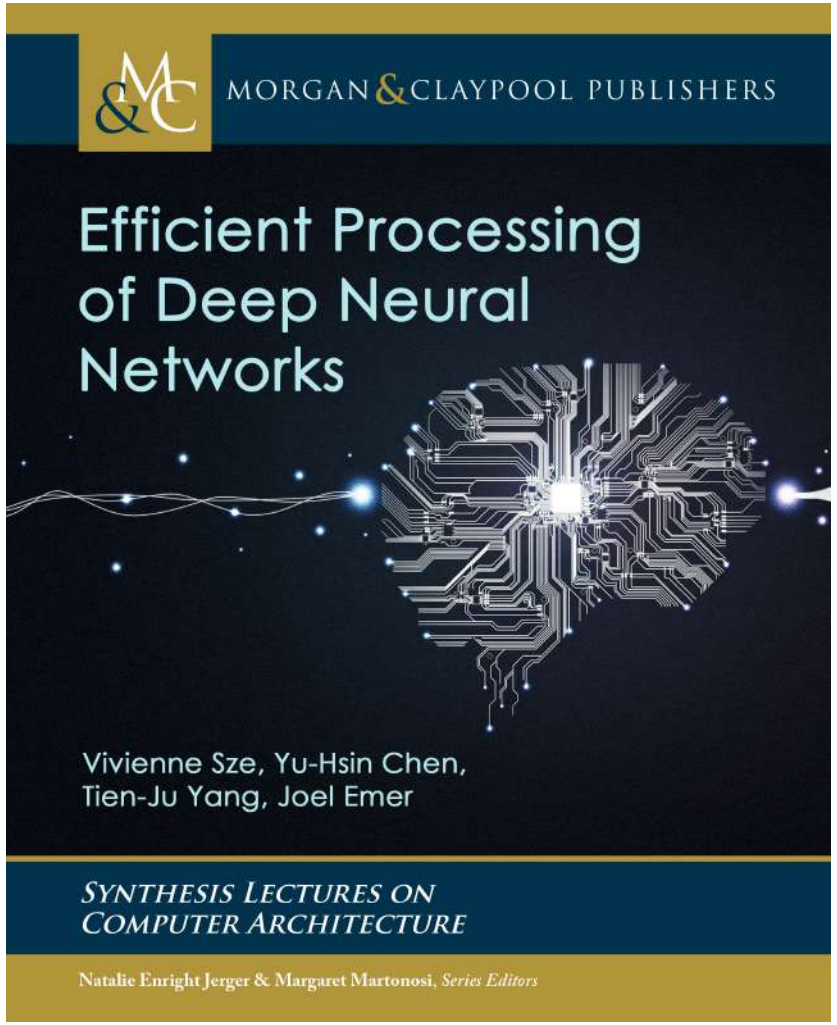
DNN tutorial website
http://eyeriss.mit.edu/tutorial.html

For updates

EEMS Mailing List

Follow @eems_mit

Book on Efficient Processing of DNNs



Part I Understanding Deep Neural Networks

Introduction

Overview of Deep Neural Networks

Part II Design of Hardware for Processing DNNs

Key Metrics and Design Objectives

Kernel Computation

Designing DNN Accelerators

Operation Mapping on Specialized Hardware

Part III Co-Design of DNN Hardware and Algorithms

Reducing Precision

Exploiting Sparsity

Designing Efficient DNN Models

Advanced Technologies

https://tinyurl.com/EfficientDNNBook

Excerpts of Book

CHAPTER 3

Key Metrics and Design Objectives

Over the past few years, there has been a significant amount of research on efficient processing of DNNs. Accordingly, it is important to discuss the key metrics that one should consider when comparing and evaluating the strengths and weaknesses of different designs and proposed techniques and that should be incorporated into design considerations. While efficiency is often only associated with the number of operations per second per Watt (e.g., floating-point operations per second per Watt as FLOPS/W or tera-operations per second per Watt as TOPS/W), it is actually composed of many more metrics including accuracy, throughput, latency, energy consumption, power consumption, cost, flexibility, and scalability. Reporting a comprehensive set of these metrics is important in order to provide a complete picture of the trade-offs made by a proposed design or technique.

In this chapter, we will

- discuss the importance of each of these metrics;
- breakdown the factors that affect each metric. When feasible, present equations that describe the relationship between the factors and the metrics;
- describe how these metrics can be incorporated into design considerations for both the DNN hardware and the DNN model (i.e., workload); and
- specify what should be reported for a given metric to enable proper evaluation.

Finally, we will provide a case study on how one might bring all these metrics together for a holistic evaluation of a given approach. But first, we will discuss each of the metrics.

3.1 ACCURACY

Accuracy is used to indicate the quality of the result for a given task. The fact that DNNs can achieve state-of-the-art accuracy on a wide range of tasks is one of the key reasons driving the popularity and wide use of DNNs today. The units used to measure accuracy depend on the task. For instance, for image classification, accuracy is reported as the percentage of correctly classified images, while for object detection, accuracy is reported as the mean average precision (mAP), which is related to the trade off between the true positive rate and false positive rate.

CHAPTER 10

Advanced Technologies

As highlighted throughout the previous chapters, data movement dominates energy consumption. The energy is consumed both in the access to the memory as well as the transfer of the data. The associated physical factors also limit the bandwidth available to deliver data between memory and compute, and thus limits the throughput of the overall system. This is commonly referred to by computer architects as the "memory wall."[1]

To address the challenges associated with data movement, there have been various efforts to bring compute and memory closer together. Chapters 5 and 6 primarily focus on how to design spatial architectures that distribute the on-chip memory closer to the computation (e.g., scratch pad memory in the PE). This chapter will describe various other architectures that use *advanced memory*, *process*, and *fabrication technologies* to brin

First, we will describe efforts to bring the off-chip
closer to the computation. These approaches are often ref
near-data processing, and include memory technologies s
stacked DRAM.

Next, we will describe efforts to integrate the comp
approaches are often referred to as *processing in memory* or *in-memory computing*, and include memory technologies such as Static Random Access Memories (SRAM), Dynamic Random Access Memories (DRAM), and emerging non-volatile memory (NVM). Since these approaches rely on mixed-signal circuit design to enable processing in the analog domain, we will also discuss the design challenges related to handling the increased sensitivity to circuit and device non-idealities (e.g., nonlinearity, process and temperature variations), as well as the impact on area density, which is critical for memory.

Significant data movement also occurs between the sensor that collects the data and the DNN processor. The same principles that are used to bring compute near the memory, where the weights are stored, can be used to bring the compute *near* the sensor, where the input data is collected. Therefore, we will also discuss how to integrate some of the compute *into* the sensor.

Finally, since photons travel much faster than electrons and the cost of moving a photon can be *independent* of distance, processing in the optical domain using light may provide significant improvements in energy efficiency and throughput over the electrical domain. Accordingly, we will conclude this chapter by discussing the recent work that performs DNN processing in the optical domain, referred to as *Optical Neural Networks*.

[1] Specifically, the memory wall refers to data moving between the off-chip memory (e.g., DRAM) and the processor.

Available DNN tutorial website
http://eyeriss.mit.edu/tutorial.html

Additional Resources

MIT Professional Education Course on

"Designing Efficient Deep Learning Systems"

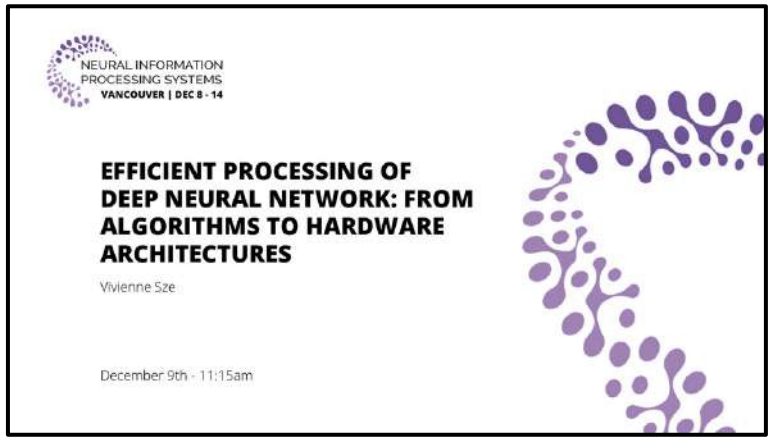
http://shortprograms.mit.edu/dls

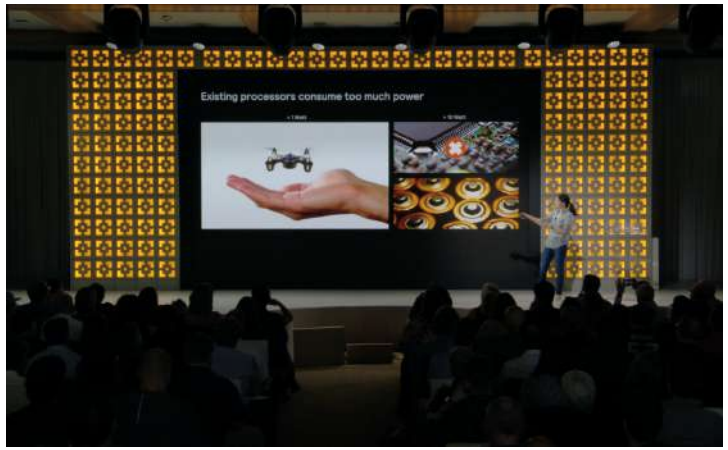
Next Offering: July 20-21, 2020 (Live Virtual)

Additional Resources

Talks and Tutorial Available Online

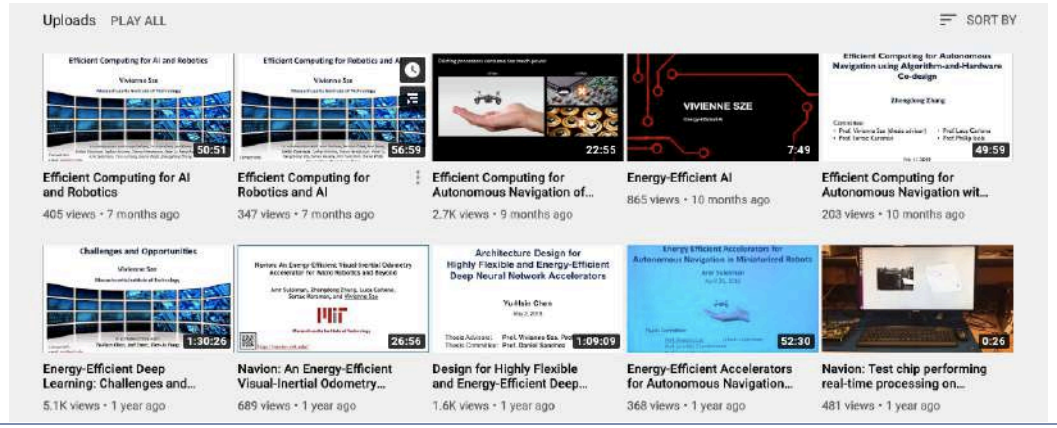
https://www.rle.mit.edu/eems/publications/tutorials/





YouTube Channel

EEMS Group – PI: Vivienne Sze



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/

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.

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

DNN Processor Evaluation Tools

- Wu et al., "Accelergy: An Architecture-Level Energy Estimation Methodology for Accelerator Designs," ICCAD 2019, http://accelergy.mit.edu
- Wu et al., "An Architecture-Level Energy and Area Estimator for Processing-In-Memory Accelerator Designs," ISPASS 2020, http://accelergy.mit.edu
- Parashar et al., "Timeloop: A Systematic Approach to DNN Accelerator Evaluation," ISPASS 2019