Domain-Specific Architectures for AI and Robotics: Opportunities and Challenges

Vivienne Sze
Massachusetts Institute of Technology

Contact Info
email: sze@mit.edu
website: www.rle.mit.edu/eems
Wide Range of Compute-Intensive Applications

**Video Compression**

**AI:**
Deep Neural Networks

**Robotics:**
Autonomous Navigation

- Rapidly growing volume of data to be processed
- Increasingly complex algorithms for higher quality of result
- Require high throughput/low latency and energy efficiency

Need Domain Specific Architectures → 2 to 3 years to design!
Key Design Considerations

• Exploit properties of workloads
  – Specialized hardware translates parallelism, data access patterns and representation into increased throughput and energy efficiency

• Design more efficient workloads
  – Co-design of algorithms and hardware without affecting quality of result

• Define range of workloads
  – Balance flexibility and efficiency depending on application requirements

How can Agile and Open Hardware help accelerate the design process?
Deep Neural Networks (DNN)
Properties We Can Leverage

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

**Memory Read**
- DRAM
  - filter weight
  - image pixel
  - partial sum

**MAC**
- ALU
  - multiply-and-accumulate

**Memory Write**
- DRAM
  - updated partial sum

200x

1x
Properties We Can Leverage

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

• **Input data reuse** opportunities (up to 500x)

Convolutional Reuse (pixels, weights)

Image Reuse (pixels)

Filter Reuse (weights)
Data Movement is Expensive

Design memory hierarchy and dataflow to exploit data reuse at low cost memories.

Normalized Energy Cost*:

- 1× (Reference)
- 1×
- 2×
- 6×
- 200×

* measured from a commercial 65nm process
Eyeriss: Deep Neural Network Accelerator

Exploits data reuse for **100x** reduction in memory accesses from global buffer and **1400x** reduction in memory accesses from off-chip DRAM

Results for AlexNet

[Joint work with Joel Emer]
Features: Energy vs. Accuracy

Energy/ Pixel (nJ)

Accuracy (Average Precision)

Measured in on VOC 2007 Dataset
1. DPM v5 [Girshick, 2012]

Measured in 65nm*

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

[Suleiman et al., ISCAS 2017]
Design of Efficient DNN Algorithms

• Popular efficient DNN algorithm approaches

Network Pruning

before pruning

after pruning

pruning synapses

pruning neurons

Compact Network Architectures

Examples: SqueezeNet, MobileNet

... also reduced precision

• Focus on reducing number of MACs and weights
• Does it translate to energy savings and reduced latency?
Key Observations

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

Energy Consumption of GoogLeNet

Energy estimation tool available at: [https://energyestimation.mit.edu/](https://energyestimation.mit.edu/)

[Yang et al., CVPR 2017]
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 most energy first
- EAP reduces AlexNet energy by 3.7x and outperforms the previous work that uses magnitude-based pruning by 1.7x

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

[Yang et al., CVPR 2017]
Many Efficient DNN Design Approaches

Network Pruning

before pruning

pruning synapses

pruning neurons

after pruning

Compact Network Architectures

Reduce Precision

32-bit float: 101001010000000000001010000000000100

8-bit fixed: 01100110

Binary: 0

No guarantee that DNN algorithm designer will use a given approach.

Need flexible hardware!

[Chen et al., SysML 2018]
Need Flexible NoC 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** satisfies above but too expensive and not scalable.
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

[Chen et al., JETCAS 2019]

[Joint work with Joel Emer]
DNN Design Considerations

• Exploit properties of workloads
  – Efficient memory hierarchy and dataflow for data reuse; exploit natural sparsity in activation

• Design more efficient workloads
  – Design efficient DNN models with increased sparsity, reduced precision, and compact network architectures
  – Drive design of algorithms with direct metrics (i.e., energy, latency) rather than indirect metrics (i.e., # of ops, weights)

• Define range of workloads
  – Flexibility to support a wide range of DNNs, including different efficient DNN approaches
Autonomous Navigation
Robot Exploration

Decide where to go by computing **Shannon Mutual Information**

1. Select candidate scan locations
2. Compute **Shannon MI** and choose best location
3. Move to location and scan
4. Update Occupancy Map

Where to scan?

**Mutual Information (MI)**

Occupancy map with planned path

MI surface

Exploration with a mini race car using motion capture for localization

[Joint work with Sertac Karaman]
Challenge is Data Delivery to All Cores

Process multiple beams in parallel

Data delivery from memory is limited
Specialized Memory Architecture

Break up map into separate memory banks and novel storage pattern to minimize read conflicts when processing different beams in parallel.

Achieves throughput within 94% of theoretical limit (unlimited bandwidth). Compute entire 20mx20m map in under a second!

[Li et al., RSS 2019]
Robot Localization in Under 25mW

Localization is a key step in autonomous navigation (also AR and VR)

Entire system fully integrated on chip. Use compression/sparsity to reduce total storage to 854kB!

Consumes $684\times$ and $1582\times$ less energy than mobile and desktop CPUs, respectively

http://navion.mit.edu

[Joint work with Sertac Karaman]

[Zhang et al., RSS 2017], [Suleiman et al., VLSI 2018]
EuRoC dataset is a very challenging, and widely used UAV dataset with 11 sequences with three categories: easy, medium, and difficult.

Examples of Easy Sequences:
- MH_1
- V1_1

Examples of Difficult Sequences:
- Dark scenes (MH_4)
- Motion blur (V2_3)

Navion has over 250 configurable parameters to adapt to different sensors and environments. Adapting to the environment results in a 2-3x energy reduction.

[Suleiman et al., JSSC 2019]
Autonomous Navigation Design Considerations

• **Exploit properties of workloads**
  – Optimized memory banking and mapping to meet memory bandwidth requirements for high throughput parallel processing

• **Design more efficient workloads**
  – Compact data representation to reduce data movement, storage and accelerate computation

• **Define range of workloads**
  – Adapt to changing environment for improved efficiency
Video Compression
Video Compression

• Video codec composed of multiple heterogenous modules
  – entropy coding, transform, motion comp, intra coding, deblocking, etc.

• Specialized hardware for each module
  – Hardcode values for parameters defined by video coding standard (e.g., weights of interpolation filter and coefficients of transform)
  – Dedicated optimized memories and dataflow for each module

• Parallel and pipeline across and within modules
Parallelism Limited By Algorithm

- Advanced algorithms more difficult to parallelize
  - Limits throughput due to Amdahl’s law

*Context-Adaptive Binary Arithmetic Coding (CABAC)*

[Joint work with Anantha Chandrakasan]
Parallelism Limited By Algorithm

- Advanced algorithms more difficult to parallelize
- Re-design algorithms to be more hardware-friendly

*Context-Adaptive Binary Arithmetic Coding (CABAC)*

Parallel entropy coding algorithm gives >10x higher throughput than state-of-the-art with minimal impact on coding loss

[Joint work with Anantha Chandrakasan]
High Efficiency Video Coding (HEVC)

- H.265/HEVC is the successor to H.264/AVC
- Achieves 2x higher compression than H.264/AVC
- High throughput (Ultra-HD 8K @ 120fps) & low power

<table>
<thead>
<tr>
<th></th>
<th>Size</th>
<th>Energy</th>
<th>Coding Efficiency</th>
<th>Efficient Implementation</th>
</tr>
</thead>
<tbody>
<tr>
<td>MPEG-2 (1994)</td>
<td>1.5x</td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>H.264/AVC (2003)</td>
<td>4x</td>
<td>2x</td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>H.265/HEVC (2013)</td>
<td>2x</td>
<td>1.5x</td>
<td></td>
<td>X</td>
</tr>
</tbody>
</table>

Co-design of algorithm and hardware to address coding efficiency, throughput and power challenges
Flexibility Needed for Video Compression

- **Support multiple standards**

  Legacy (2003, 2013)  
  - H.264/MPEG-4 AVC
  - HEVC

  Emerging (2019, 2020)  
  - AV1
  - VVC

  *Shared resources (e.g., cache for motion compensation), but modules tend to be hardcoded due to tight power and speed requirements*

- **Encoder must be flexible**

  While decoder is standardize, encoder allows for product differentiation (e.g., better motion estimation)
Video Compression Design Considerations

• Exploit properties of workloads
  – Specialized hardware for heterogenous set of modules with hardcoded parameters; exploit parallelism and pipelining

• Design more efficient workloads
  – Parallel entropy coding algorithm to remove compute bottleneck
  – Co-design of algorithms and hardware in HEVC standard

• Define range of workloads
  – Flexibility to support multiple video standards and algorithm changes in encoder
Summary

• Domain-specific hardware can address the rising compute demands for many existing and emerging applications

• **Opportunities**
  – Exploit properties of workloads (e.g., parallelism, access patterns, representation)
  – Design efficient workloads using co-design of algorithms and hardware without affecting quality of result

• **Challenges**
  – Define range of workloads to support based on flexibility versus efficiency tradeoff
  – Workloads will evolve over time and across use cases/environments

• Agile design can be used for rapid exploration of workloads and tradeoff

• Open hardware can allow for rapid system development with **shared building blocks**
  – May need to configure for given application requirements
  – What is the granularity of the blocks?

Today’s slides available at [www.rle.mit.edu/eems](http://www.rle.mit.edu/eems)
Acknowledgements

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

- **Efficient Processing for Deep Neural Networks**
  - Project website: [http://eyeriss.mit.edu](http://eyeriss.mit.edu)
References

• Co-Design of Algorithms and Hardware for Deep Neural Networks
References

• Fast Shannon Mutual Information for Robot Exploration

• Energy-Efficient Visual Inertial Localization
  – Project website: http://navion.mit.edu
References

• Video Compression


